Studies in Computational Intelligence 1078

Hocine Cherifi ·
Rosario Nunzio Mantegna ·
Luis M. Rocha · Chantal Cherifi ·
Salvatore Micciche *Editors*

Complex Networks and Their Applications XI

Proceedings of The Eleventh International Conference on Complex Networks and Their Applications: COMPLEX NETWORKS 2022—Volume 2

Springer

Studies in Computational Intelligence

Volume 1078

The series "Studies in Computational Intelligence" (SCI) publishes new developments and advances in the various areas of computational intelligence—quickly and with a high quality. The intent is to cover the theory, applications, and design methods of computational intelligence, as embedded in the fields of engineering, computer science, physics and life sciences, as well as the methodologies behind them. The series contains monographs, lecture notes and edited volumes in computational intelligence spanning the areas of neural networks, connectionist systems, genetic algorithms, evolutionary computation, artificial intelligence, cellular automata, self-organizing systems, soft computing, fuzzy systems, and hybrid intelligent systems. Of particular value to both the contributors and the readership are the short publication timeframe and the world-wide distribution, which enable both wide and rapid dissemination of research output.

Indexed by SCOPUS, DBLP, WTI Frankfurt eG, zbMATH, SCImago.

All books published in the series are submitted for consideration in Web of Science.

Hocine Cherifi · Rosario Nunzio Mantegna ·
Luis M. Rocha · Chantal Cherifi ·
Salvatore Micciche
Editors

Complex Networks and Their Applications XI

Proceedings of The Eleventh International Conference on Complex Networks and Their Applications: COMPLEX NETWORKS 2022—Volume 2

Springer

Editors
Hocine Cherifi
University of Burgundy
Dijon, France

Luis M. Rocha
Thomas J. Watson College of Engineering and Applied Science
Binghamton University
Binghamton, NY, USA

Salvatore Micciche
Dipartimento di Fisica e Chimica—Emilio Segrè
Università degli Studi Palermo
Palermo, Italy

Rosario Nunzio Mantegna
Dipartimento di Fisica e Chimica—Emilio Segrè
Università degli Studi Palermo
Palermo, Italy

Chantal Cherifi
IUT Lumière—Université Lyon 2
University of Lyon
Bron, France

ISSN 1860-949X ISSN 1860-9503 (electronic)
Studies in Computational Intelligence
ISBN 978-3-031-21130-0 ISBN 978-3-031-21131-7 (eBook)
https://doi.org/10.1007/978-3-031-21131-7

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Organization and Committees

General Chairs

Hocine Cherifi, University of Burgundy, France
Rosario N. Mantegna, Università degli Studi di Palermo, Italy
Luis M. Rocha, Binghamton University, USA

Advisory Board

Jon Crowcroft, University of Cambridge, UK
Raissa D'Souza, University of California, Davis, USA
Eugene Stanley, Boston University, USA
Ben Y. Zhao, University of Chicago, USA

Program Chairs

Chantal Cherifi, University of Lyon, France
Salvatore Micciché, Università degli Studi di Palermo, Italy

Satellite Chairs

Sabrina Gaito, University of Milan, Italy
Javier Galeano, Universidad Politécnica de Madrid, Spain

Lightning Chairs

Alessandro Rizzo, Politecnico di Torino, Italy
Giancarlo Francesco Ruffo, Università degli Studi di Torino, Italy
Huijuan Wang, TU Delft, Netherlands

Poster Chairs

Manuel Marques-Pita, University Lusófona, Portugal
Michele Tumminello, Università degli Studi di Palermo, Italy
Laura Ricci, Università degli Studi di Pisa, Italy

Publicity Chairs

Christophe Cruz, University of Burgundy, France
Xiangjie Kong, Dalian University of Technology, China

Tutorial Chairs

Luca Maria Aiello, Nokia-Bell Labs, UK
Leto Peel, UC Louvain, Belgium

Special Issues Chair

Sabrina Gaito, University of Milan, Italy

Sponsor Chair

Roberto Interdonato, CIRAD—UMR TETIS, France
Claudio Fazio, Università degli Studi di Palermo, Italy

Local Committee Chair

Federico Musciotto, Università degli Studi di Palermo, Italy

Local Committee

Yuri Antonacci, Università degli Studi di Palermo, Italy
Giosué Lo Bosco, Università degli Studi di Palermo, Italy
Luca Faes, Università degli Studi di Palermo, Italy
Giacomo Fiumara, Università degli Studi di Messina, Italy
Vincenzo Giuseppe Genova, Università degli Studi di Palermo, Italy
Matteo Milazzo, University of Catania, Italy
Pasquale De Meo, Università degli Studi di Messina, Italy
Alessandro Pluchino, Università degli Studi di Catania, Italy
Andrea Rapisarda, Università degli Studi di Catania, Italy

Publication Chair

Matteo Zignani, University of Milan, Italy

Web Chair

Zakariya Ghalmane, LINEACT, CESI, France
Stephany Rajeh, University of Burgundy, France

Program Committee

Jacobo Aguirre, Centro de Astrobiología (CSIC-INTA), Spain
Amreen Ahmad, Jamia Millia Islamia, India
Masaki Aida, Tokyo Metropolitan University, Japan
Luca Maria Aiello, IT University of Copenhagen, Denmark
Marco Aiello, University of Stuttgart, Germany
Esra Akbas, Georgia State University, USA
Mehmet Aktas, University of Central Oklahoma, USA
Tatsuya Akutsu, Kyoto University, Japan
Reka Albert, The Pennsylvania State University, USA

Laura Maria Alessandretti, ENS Lyon, Italy
Antoine Allard, Université Laval, Canada
Claudio Altafini, Linköping University, Sweden
Lucila G. Alvarez-Zuzek, Fondazione Bruno Kessler, Italy
Luiz G. A. Alves, Northwestern University, USA
Diego Amancio, University of São Paulo, Brazil
G. Ambika, IISER, India
Fred Amblard, IRIT—University Toulouse 1 Capitole, France
Hamed Amini, Georgia State University, USA
Andre Franceschi De Angelis, Unicamp, Brazil
Marco Tulio Angulo, National Autonomous University of Mexico (UNAM), Mexico
Alberto Antonioni, Carlos III University of Madrid, Spain
Nuno Araujo, Universidade de Lisboa, Portugal
Saúl Ares, Centro Nacional de Biotecnología (CNB-CSIC), Spain
Malbor Asllani, University College Dublin, Ireland
Tomaso Aste, University College London, UK
Martin Atzmueller, Osnabrück University, Germany
Konstantin Avrachenkov, INRIA, France
Giacomo Baggio, University of Padova, Italy
Rodolfo Baggio, Bocconi University, Italy
Franco Bagnoli, University of Florence, Italy
Avner Bar-Hen, CNAM, France
Paolo Barucca, University College London, UK
Nikita Basov, St. Petersburg State University, Russia
Gareth Baxter, University of Aveiro, Portugal
Marya Bazzi, University of Oxford, UK
Andras A. Benczur, Hungarian Academy of Sciences, Hungary
Rosa M. Benito, Universidad Politécnica de Madrid, Spain
Kamal Berahmand, Queensland University of Technology, Australia
Cyrille Bertelle, Le Havre University, France
Marc Bertin, Université Claude Bernard Lyon 1, France
Ginestra Bianconi, Queen Mary University of London, UK
Christian Bick, University of Exeter, UK
Ofer Biham, The Hebrew University of Jerusalem, Israel
Livio Bioglio, University of Turin, Italy
Hanjo Boekhout, Leiden University, Netherlands
Johan Bollen, Indiana University Bloomington, USA
Anthony Bonato, Toronto Metropolitan University, Canada
Christian Bongiorno, Centralesupelec, France
Anton Borg Blekinge, Institute of Technology, Sweden
Pierre Borgnat, CNRS, Laboratoire de Physique ENS de Lyon, France
Stefan Bornholdt, Universität Bremen, Germany
Javier Borondo, Upm, Spain
Cécile Bothorel, IMT Atlantique, France
Federico Botta, The University of Exeter, UK

Alexandre Bovet, Mathematical Institute, University of Oxford, UK
Fabian Braesemann, University of Oxford, UK
Dan Braha, NECSI, USA
Ulrik Brandes, ETH Zürich, Switzerland
Rion Brattig Correia, Instituto Gulbenkian de Ciência, Portugal
Markus Brede, University of Southampton, UK
Iulia Martina Bulai, University of Sassari, Italy
Javier M. Buldu, U.R.J.C. & Center for Biomedical Technology, Spain
Raffaella Burioni, Dipartimento di Fisica, Università di Parma, Italy
Noah Burrell, University of Michigan, USA
Fabio Caccioli, University College London, UK
Carmela Calabrese, Aix-Marseille Université, Italy
Paolo Campana, University of Cambridge, UK
M. Abdullah Canbaz, University at Albany, SUNY, USA
Carlo Vittorio Cannistraci, TU Dresden, Germany
Vincenza Carchiolo, Universita' di Catania, Italy
Giona Casiraghi, ETH Zürich, Switzerland
Douglas Castilho, Federal Institute of South of Minas Gerais, Brazil
Costanza Catalano, Central Bank of Italy, Italy
Remy Cazabet, Université Lyon 1, France
Po-An Chen, National Yang Ming Chiao Tung University, Taiwan
Xihui Chen, University of Luxembourg, Luxembourg
Chantal Cherifi, Lyon 2 University, DISP Laboratory, France
Hocine Cherifi, University of Burgundy, France
Peter Chin, Boston University, USA
Matteo Chinazzi, Northeastern University, USA
Matteo Cinelli, Ca' Foscari, University of Venice, Italy
Richard Clegg, Queen Mary University of London, UK
Reuven Cohen, Bar-Ilan University, Israel
Alessio Conte, University of Pisa, Italy
Michele Coscia, IT University of Copenhagen, Denmark
Christophe Crespelle, Université Claude Bernard Lyon 1, France
Regino Criado, Universidad Rey Juan Carlos, Spain
Mihai Cucuringu, University of Oxford, USA
Bhaskar Dasgupta, University of Illinois at Chicago, USA
Joern Davidsen, University of Calgary, Canada
Toby Davies, University College London, UK
Pasquale De Meo, Università degli Studi di Messina, Italy
Fabrizio De Vico Fallani, Inria—ICM, France
Guillaume Deffuant, Cemagref, France
Charo I. del Genio, Coventry University, UK
Pietro Delellis, University of Naples Federico II, Italy
Karel Devriendt, University of Oxford, UK
Kuntal Dey, Accenture Tech Labs, India
Riccardo Di Clemente, University of Exeter, UK

Matías Di Muro, IFIMAR Institute, Argentina
Marcelo Do Vale Cunha, Federal Institute of Bahia, Brazil, Brazil
Constantine Dovrolis, Georgia Institute of Technology, USA
Ahlem Drif, Sétif 1 University, Algeria
Johan Dubbeldam, Delft University of Technology, Netherlands
Jordi Duch, Universitat Rovira i Virgili, Spain
Nicolas Dugué, LIUM, France
Victor M. Eguiluz, IFISC (CSIC-UIB), Spain
Alexandre Evsukoff, COPPE/UFRJ, Brazil
Mauro Faccin, CEPED, Université de Paris, France
Giorgio Fagiolo, Sant'Anna School of Advanced Studies, Italy
Daniel Figueiredo, UFRJ, Brazil
Marco Fiore, IMDEA Networks Institute, Spain
Alessandro Flammini, Indiana University Bloomington, USA
Manuel Foerster, Bielefeld University, Germany
Mattia Frasca, University of Catania, Italy
Angelo Furno, University of Lyon, France
Sabrina Gaito, University of Milan, Italy
Javier Galeano, Universidad Politécnica de Madrid, Spain
Stéphane Galland, UBFC—UTBM, France
Lazaros Gallos, Rutgers University, USA
Riccardo Gallotti, Fondazione Bruno Kessler, Italy
José Manuel Galán, Universidad de Burgos, Spain
Joao Gama, University of Porto, Portugal
Yerali Gandica, CY Cergy Paris Université, France
Jianxi Gao, Rensselaer Polytechnic Institute, USA
Floriana Gargiulo, University of Paris Sorbonne, France
Michael Gastner, Yale-NUS College, Singapore
Alexander Gates, Northeastern University, USA
Vincent Gauthier, Institut Mines Telecom CNRS SAMOVAR, France
Ralucca Gera, Naval Postgraduate School, USA
Raji Ghawi, Technical University of Munich, Germany
Jean-Louis Giavitto, IRCAM, France
Tommaso Gili, IMT, Italy
Silvia Giordano, SUPSI, Switzerland
Pierre-Louis Giscard, Université du Littoral Côte d'Opale, France
David Gleich, Purdue University, USA
Antonia Godoy, University College London, UK
Kwang-Il Goh, Korea University, South Korea
Jesus Gomez-Gardenes, Universidad de Zaragoza, Spain
Antonio Gonzalez-Pardo, Universidad Rey Juan Carlos, Spain
Bruno Gonçalves, Data For Science, Inc, USA
Carlos Gracia-Lázaro, BIFI, Spain
Jean-Loup Guillaume, Université de la Rochelle, France
Mehmet Gunes, Stevens Institute of Technology, USA

Sergio Gómez, Universitat Rovira i Virgili, Spain
Meesoon Ha, Chosun University, South Korea
Jürgen Hackl, University of Liverpool, Switzerland
Aric Hagberg, Los Alamos National Laboratory, USA
Chris Hankin, Imperial College London, UK
Yukio Hayashi, Japan Advanced Institute of Science and Technology, Japan
Mark Heimann, Lawrence Livermore National Laboratory, USA
Torsten Heinrich, Chemnitz University of Technology, Germany
Denis Helic, Graz University of Technology, Austria
Takayuki Hiraoka, Aalto University, Finland
Philipp Hoevel, University College Cork, Ireland
Seok-Hee Hong, University of Sydney, Australia
H. Ulrich Hoppe, University Duisburg-Essen, Germany
Yuichi Ikeda, Kyoto University, Japan
Roberto Interdonato, CIRAD—UMR TETIS, France
Giulia Iori, City University London, UK
Francesco Iorio, Human Technopole, Italy
Antonio Iovanella, Università degli Studi Internazionali di Roma, Italy
Sarika Jalan, IIT Indore, India
Mahdi Jalili, RMIT University, Australia
Jaroslaw Jankowski, West Pomeranian University of Technology, Poland
Marco Alberto Javarone, Centro Ricerche Enrico Fermi, Italy
Hawoong Jeong, Korea Advanced Institute of Science and Technology, South Korea
Tao Jia, Southwest University, China
Chunheng Jiang, Rensselaer Polytechnic Institute, USA
Jiaojiao Jiang, University of New South Wales, Australia
Di Jin, Amazon, USA
Hang-Hyun Jo, The Catholic University of Korea, South Korea
Ivan Jokić, Delft University of Technology, Netherlands
Marko Jusup, Tokyo Institute of Technology, Japan
Arkadiusz Jędrzejewski, Wrocław University of Science and Technology, Poland
Rushed Kanawati, Université Sorbonne Paris Nord, France
Eytan Katzav, The Hebrew University of Jerusalem, Israel
Mehmet Kaya, Firat University, Turkey
Domokos Kelen, Institute for Computer Science and Control, Hungary
Dror Kenett, Johns Hopkins University, USA
Mohammad Khansari, University of Tehran, Iran
Khaldoun Khashanah, Stevens Institute of Technology, USA
Hamamache Kheddouci, Universit Claude Bernard, France
Hyoungshick Kim, Sungkyunkwan University, South Korea
Jinseok Kim, University of Michigan, USA
Maksim Kitsak, Delft University of Technology, Netherlands
Mikko Kivela, Aalto University, Finland
Peter Klimek, Medical University of Vienna, Austria
Andrei Klishin, University of Pennsylvania, USA

Dániel Kondor, Complexity Science Hub Vienna, Austria
Xiangjie Kong, Zhejiang University of Technology, China
Ismo Koponen, University of Helsinki, Finland
Onerva Korhonen, Aalto University, Finland
Elka Korutcheva, Universidad Nacional de Educación a Distancia, Spain
Dimitris Kotzinos, CY cergy Paris University, France
Prosenjit Kundu, State University of New York at Buffalo, USA
Pascale Kuntz, Laboratoire d'Informatique de Nantes Atlantique, France
Ryszard Kutner, Faculty of Physics, University of Warsaw, Poland
Haewoon Kwak, Singapore Management University, Singapore
Richard La, University of Maryland, USA
José Lages, Université Bourgogne Franche-Comté, France
Claire Lagesse, TheMa UMR 6049, France
Mohammed Lalou, University of Claude Bernard Lyon1, France
Renaud Lambiotte, University of Oxford, UK
Aniello Lampo, Universidad Carlos Tercero Madrid (UC3M), Spain
Christine Largeron, Université de Lyon, France
Timothy LaRock, Oxford Mathematical Institute, UK
Jennifer Larson, Vanderbilt University, USA
Paul Laurienti, Wake Forest, USA
Anna T. Lawniczak, University of Guelph, Canada
Deok-Sun Lee, Korea Institute for Advanced Study, South Korea
Balazs Lengyel, Hungarian Academy of Sciences, Hungary
Maxime Lenormand, INRAE, France
Haiko Lietz, Leibniz Institute for the Social Sciences, Germany
Fabrizio Lillo, University of Bologna, Italy
Ji Liu, Stony Brook University, USA
Giacomo Livan, University College London, UK
Lorenzo Livi, University of Manitoba, Canada
Juan Carlos Losada, Universidad Politécnica de Madrid, Spain
Meilian Lu, Beijing University of Posts and Telecommunications, China
Leonardo Maccari, University of Venice, Italy
Matteo Magnani, Uppsala University, Sweden
Maria Malek, ETIS, CY Tech, France
Nishant Malik, Rochester Institute of Technology, USA
Fragkiskos Malliaros, Paris-Saclay University, France
Giuseppe Mangioni, University of Catania, Italy
Rosario Nunzio Mantegna, Università degli Studi di Palermo, Italy
Manuel Sebastian Mariani, University of Zurich, Switzerland
Radek Marik, Czech Technical University, Czechia
Daniele Marinazzo, Ghent University, Belgium
Andrea Marino, University of Florence, Italy
Manuel Marques-Pita, Universidade Lusofona, Portugal
Christoph Martin, Hamburg University of Applied Sciences, Germany
Cristina Masoller, Universitat Politècnica de Catalunya, Spain

Rossana Mastrandrea, IMT Institute of Advanced Studies, Italy
Ruth Mateos de Cabo, Universidad CEU San Pablo, Spain
John Matta, Southern Illinois University Edwardsville, USA
Fintan McGee, Luxembourg Institute of Science and Technology, Luxembourg
Matúš Medo, University of Bern, Switzerland
Jose Fernando Mendes, University of Aveiro, Portugal
Humphrey Mensah, Syracuse University, USA
Engelbert Mephu Nguifo, University Clermont Auvergne, France
Anke Meyer-Baese, FSU, USA
Salvatore Micciche, Università degli Studi di Palermo, Italy
Radosław Michalski, Wrocław University of Science and Technology, Poland
Letizia Milli, University of Pisa, Italy
Boris Mirkin, Higher School of Economics Moscow, UK
Shubhanshu Mishra, University of Illinois at Urbana-Champaign, USA
Marija Mitrović Dankulov, Institute of Physics Belgrade, Serbia
Andrzej Mizera, University of Luxembourg, Luxembourg
Osnat Mokryn, University of Haifa, Israel
Roland Molontay, Budapest University of Technology and Economics, Hungary
Raul Mondragon, Queen Mary University of London, UK
Misael Mongiovì, Consiglio Nazionale delle Ricerche, Italy
Alfredo Morales, MIT Media Lab, USA
Andres Moreira, Universidad Tecnica Federico Santa Maria, Chile
Esteban Moro, Universidad Carlos III de Madrid, Spain
Greg Morrison, University of Houston Department of Physics, USA
Sotiris Moschoyiannis, University of Surrey, UK
Igor Mozetič, Jozef Stefan Institute, Slovenia
Alberto P. Munuzuri, University of Santiago de Compostela, Spain
Masayuki Murata, Osaka University, Japan
Tsuyoshi Murata, Tokyo Institute of Technology, Japan
Katarzyna Musial, University of Technology Sydney, Australia
Muaz Niazi, National University of Science and Technology, Pakistan
Alexandre Nicolas, Université Claude Bernard Lyon 1, France
Peter Niemeyer, Leuphana Universität Lüneburg, Germany
Jordi Nin, ESADE, Universitat Ramon Llull, Spain
Rogier Noldus, Ericsson, Netherlands
El Faouzi Nour-Eddin, Université Gustave Eiffel, France
Masaki Ogura, Osaka University, Japan
Oleh Omelchenko, University of Potsdam, Germany
Andrea Omicini, Università di Bologna, Italy
Gergely Palla, ELTE Institute of Physics, Hungary
Pietro Panzarasa, Queen Mary University of London, UK
Fragkiskos Papadopoulos, Cyprus University of Technology, Cyprus
Symeon Papadopoulos, Information Technologies Institute, Greece
Han Woo Park, YeungNam University, South Korea
Juyong Park, Korea Advanced Institute of Science and Technology, South Korea

Pierre Parrend, EPITA Strasbourg, France
Philip E. Paré, Purdue University, USA
Leto Peel, Maastricht University, Belgium
Tiago Peixoto, Central European University, Germany
Matjaz Perc, University of Maribor, Slovenia
Anthony Perez, Université d'Orléans, France
Lilia Perfeito, Nova School of Business and Economics, Portugal
Giovanni Petri, CENTAI, Italy
Juergen Pfeffer, Technical University of Munich, Germany
Bruno Pinaud, Université de Bordeaux, France
Flavio Pinheiro, NOVA IMS—Universidade NOVA de Lisboa, Portugal
Clara Pizzuti, National Research Council of Italy (CNR), Italy
Pawel Pralat, Toronto Metropolitan University, Canada
Natasa Przulj, University College London, Spain
Oriol Pujol, University of Barcelona, Spain
Rami Puzis, Ben Gurion University of the Negev, Israel
Christian Quadri, University of Milan, Italy
Filippo Radicchi, Indiana University, USA
Andrea Rapisarda, Università di Catania, Italy
Miguel Rebollo, Universitat Politècnica de València, Spain
Juan Restrepo, University of Colorado Boulder, USA
Pedro Ribeiro, University of Porto, Portugal
Laura Ricci, University of Pisa, Italy
Alessandro Rizzo, Politecnico di Torino, Italy
Fernando Rosas, Imperial College London, UK
Giulio Rossetti, KDD Lab ISTI-CNR, Italy
Fabrice Rossi, Université Paris Dauphine PSL, France
Luca Rossi, IT University of Copenhagen, Denmark
Camille Roth, CNRS, Germany
Celine Rozenblat, University of Lausanne Institut de Géographie, Switzerland
Giancarlo Ruffo, Università degli Studi di Torino, Italy
Iraj Saniee, Bell Labs, Nokia, USA
Francisco C. Santos, Universidade de Lisboa, Portugal
Hiroki Sayama, Binghamton University, USA
Nicolas Schabanel, École Normale Supérieure de Lyon, France
Michael Schaub, RWTH Aachen University, Germany
Maximilian Schich, Professor for Cultural Data Analytics, Estonia
Caterina Scoglio, Kansas State University, USA
Hamida Seba, University Lyon1, France
Emre Sefer, Ozyegin University, Turkey
Santiago Segarra, Rice University, USA
Irene Sendiña-Nadal, Rey Juan Carlos University, Spain
Termeh Shafie, Leibniz Institute for the Social Sciences, Germany
Yilun Shang, Northumbria University, UK
Rajesh Sharma, University of Tartu, Estonia

Julian Sienkiewicz, Warsaw University of Technology, Poland
Anurag Singh, National Institute of Technology Delhi, India
Tiratha Raj Singh, Jaypee University of Information Technology, India
Rishabh Singhal, Dayalbagh Educational Institute, India
Per Sebastian Skardal, Trinity College, USA
Keith Smith, Nottingham Trent University, UK
Igor Smolyarenko, Brunel University, UK
Zbigniew Smoreda, Orange Labs, France
Annalisa Socievole, National Research Council of Italy (CNR), Italy
Albert Sole, Universitat Oberta de Catalunya, Spain
Sucheta Soundarajan, Syracuse University, USA
Semanur Soyyigit, Kirklareli University, Turkey
Jaya Sreevalsan-Nair, IIIT Bangalore, India
Christoph Stadtfeld, ETH Zurich, Switzerland
Massimo Stella, University of Exeter, UK
Fabian Stephany, University of Oxford, UK
Kashin Sugishita, Tokyo Institute of Technology, Japan
Xiaoqian Sun, Beihang University, China
Xiaoqian Sun, Chinese Academy of Sciences (CAS), China
Bosiljka Tadic, Jozef Stefan Institute, Slovenia
Andrea Tagarelli, DIMES, University of Calabria, Italy
Patrick Taillandier, INRAE/MIAT, France
Kazuhiro Takemoto, Kyushu Institute of Technology, Japan
Frank Takes, Leiden University, Netherlands
Fabien Tarissan, CNRS—ENS Paris-Saclay (ISP), France
Laura Temime, Conservatoire National des Arts et Métiers, France
Claudio Juan Tessone, Universität Zürich, Switzerland
François Théberge, Tutte Institute for Mathematics and Computing, Canada
Michele Tizzoni, ISI Foundation, Italy
Leonardo Torres, Max Planck Institute for Mathematics in the Sciences, Germany
Vincent Antonio Traag, Leiden University, Netherlands
Jan Treur, Vrije Universiteit Amsterdam, Netherlands
Milena Tsvetkova, London School of Economics and Political Science, UK
Liubov Tupikina, Universite de Paris, France
Janos Török, Budapest University of Technology and Economics, Hungary
Stephen Uzzo, New York Hall of Science, USA
Lucas D. Valdez, IFIMAR, Argentina
Pim van der Hoorn, Eindhoven University of Technology, Netherlands
Piet Van Mieghem, Delft University of Technology, Netherlands
Onur Varol, Sabanci University, Turkey
Christian Lyngby Vestergaard, CNRS and Institut Pasteur, France
Johannes Wachs, Central European University, Hungary
Huijuan Wang, Delft University of Technology, Netherlands
Wei Wang, Sichuan University, China
Ingmar Weber, Qatar Computing Research Institute, Qatar

Karoline Wiesner, University of Potsdam, Germany
Gordon Wilfong, Nokia Bell Labs, USA
Mateusz Wilinski, Los Alamos National Laboratory, USA
Richard Wilson, University of York, UK
Dirk Witthaut, Forschungszentrum Jülich, Germany
Jinshan Wu, Beijing Normal University, China
Haoxiang Xia, Dalian University of Technology, China
Fei Xiong, Beijing Jiaotong University, China
Xiaoke Xu, Dalian Minzu University, China
Gitanjali Yadav, University of Cambridge, UK
Gang Yan, Tongji University, China
Xiaoran Yan, Zhejiang Lab, China
Ying Ye, Nanjing University, China
Emilio Zagheni, Max Planck Institute for Demographic Research, Germany
An Zeng, Beijing Normal University, China
Yuexia Zhang, Beijing Information Science and Technology University, China
Zi-Ke Zhang, Hangzhou Normal University, China
Matteo Zignani, University of Milan, Italy
Eugenio Zimeo, University of Sannio, Italy
Lorenzo Zino, University of Groningen, Netherlands
Fabiana Zollo, Ca' Foscari University of Venice, Italy

Preface

The "Università degli Studi di Palermo" in Italy hosts the eleventh edition of the "International Conference on Complex Networks and Their Applications" (COMPLEX NETWORKS 2022) from November 8 to November 10, 2022. Every year, COMPLEX NETWORKS brings together researchers from various scientific backgrounds to review the field's current state and formulate new directions. The diversity of the attendees' scientific interests (Finance, Medicine and Neuroscience, Biology and Earth Sciences, Sociology and Politics, Computer Science and Physics, etc.) is a unique opportunity for crossfertilization between fundamental issues and innovative applications.

This edition attracted authors from all over the world, with 313 submissions from 54 countries. Each submission has been peer-reviewed by at least three independent reviewers from the international program committee. The 104 papers included in the proceedings result from this rigorous selection process.

The quality of the contributors is undoubtedly an essential element for a successful edition. The success also goes to the keynote speakers. These leaders and visionaries in their fields present fascinating plenary lectures with big-picture ideas and unique perspectives to help attendees deepen their understanding of scientific challenges. We are delighted to bring together this great line-up of speakers.

- Luís A. Nunes Amaral (Northwestern University, USA)
- Manuel Cebrian (Max Planck Institute for Human Development, Germany)
- Shlomo Havlin (Bar-Ilan University, Israel)
- Giulia Iori (City, University of London, UK)
- Melanie Mitchell (Santa Fe Institute, USA)
- Ricard Solé (Universitat Pompeu Fabra, Spain)

Our thanks also go to the speakers of the traditional tutorial sessions for delivering insightful talks on November 7, 2022.

- Michele Coscia (IT University of Copenhagen, Denmark)
- Adriana Iamnitchi (Maastricht University, Netherlands)

The success also relies in the deep involvement of many individuals, institutions, and sponsors.

We sincerely gratify the advisory board members for inspiring the essence of the conference:

Jon Crowcroft (University of Cambridge), Raissa D'Souza (University of California, Davis, USA), Eugene Stanley (Boston University, USA), and Ben Y. Zhao (University of Chicago, USA)

We record our thanks to our fellow members of the organizing committee:
The lightning sessions chairs:

Alessandro Rizzo (Politechnico di Torino, Italy), Giancarlo Francesco Ruffo (Università degli Studi di Torino, Italy), and Huijuan Wang (TU Delft, Netherlands)

The poster sessions chairs:

Manuel Marques Pita (Universidade Lusófona, Portugal), Michele Tumminello (Università degli Studi di Palermo, Italy), Laura Ricci (Università degli Studi di Pisa, Italy)

The tutorial chairs:

Luca Maria Aiello (Nokia Bell Labs, UK) and Leto Peel (Maastricht University, Netherland)

The special issue chair:

Sabrina Gaito (University of Milan, Italy)

The publicity chairs:

Christophe Cruz (University of Burgundy, France), Xiangjie Kong (Zhejiang University of Technology, China)

The sponsor chairs:

Claudio Fazio (Università degli Studi di Palermo, Italy) and Roberto Interdonato (CIRAD—UMR TETIS, Montpellier, France)

The Web chairs:

Zakariya Ghalmane (LINEACT, CESI, France) and Stephany Rajeh (University of Burgundy, France)

Our profound thanks go to Matteo Zignani (University of Milan, Italy), publication chair, for the tremendous work in managing the submission system and the proceedings publication process.

We would also like to record our appreciation for the work of the local committee chair, Federico Musciotto (Università degli Studi di Palermo, Italy), and all the local committee members, Yuri Antonacci (Università degli Studi di Palermo, Italy),

Giosuè Lo Bosco (Università degli Studi di Palermo, Italy), Luca Faes (Università degli Studi di Palermo, Italy), Giacomo Fiumara (Università degli Studi di Messina, Italy), Vincenzo Giuseppe Genova (Università degli Studi di Palermo, Italy), Matteo Milazzo (University of Catania, Italy), Pasquale de Meo (Università degli Studi di Messina, Italy), Alessandro Pluchino (Università degli Studi di Catania, Italy), Andrea Rapisarda (Università degli Studi di Catania, Italy), for their work in managing the sessions. They intensely participated to the success of this edition.

We would like to express our gratitude to our partner journals: *Advances in Complex Systems*, *Applied Network science*, *Complex Systems*, *Entropy*, *PLOS One* and *Social Network Analysis and Mining*.

We are thankful to all those who have contributed to the success of this meeting. Sincere thanks to the authors for their creativity.

Finally, we would like to express our most sincere thanks to the program committee members for their considerable efforts in producing high-quality reviews in a minimal time.

These volumes make the most advanced contribution of the international community to the research issues surrounding the fascinating world of complex networks. Their breath, quality, and novelty demonstrate the profound impact of complex networks in understanding our world. We hope you enjoy the papers as much as we enjoyed organizing the conference and putting this collection of articles together.

Dijon, France — Hocine Cherifi
Palermo, Italy — Rosario Nunzio Mantegna
Binghamton, USA — Luis M. Rocha
Bron, France — Chantal Cherifi
Palermo, Italy — Salvatore Micciche

Contents

Network Models

Modularity of the ABCD Random Graph Model with Community Structure

Bogumił Kamiński, Bartosz Pankratz, Paweł Prałat, and François Théberge

Abstract The Artificial Benchmark for Community Detection graph (**ABCD**) is a random graph model with community structure and power-law distribution for both degrees and community sizes. The model generates graphs with similar properties as the well-known **LFR** one, and its main parameter ξ can be tuned to mimic its counterpart in the **LFR** model, the mixing parameter μ. In this paper, we investigate various theoretical asymptotic properties of the **ABCD** model. In particular, we analyze the modularity function, arguably, the most important graph property of networks in the context of community detection. Indeed, the modularity function is often used to measure the presence of community structure in networks. It is also used as a quality function in many community detection algorithms, including the widely used *Louvain* algorithm.

Keywords ABCD model · Modularity function · Community detection

1 Introduction

One of the most important features of real-world networks is their community structure, as it reveals the internal organization of nodes [9]. In social networks communities may represent groups by interest, in citation networks they correspond to related

B. Kamiński
Warsaw School of Economics, Warsaw, Poland
e-mail: bkamins@sgh.waw.pl

B. Pankratz · P. Prałat (✉)
Ryerson University, Toronto, ON, Canada
e-mail: bartosz.pankratz@ryerson.ca

B. Pankratz
e-mail: pralat.pankratz@ryerson.ca

F. Théberge
The Tutte Institute for Mathematics and Computing, Ottawa, ON, Canada
e-mail: theberge@ieee.org

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_1

papers, in the Web communities are formed by pages on related topics, etc. Being able to identify communities in a network could help us to exploit this network more effectively.

Unfortunately, there are very few datasets with ground-truth identified and labelled. As a result, there is need for synthetic random graph models with community structure that resemble real-world networks in order to benchmark and tune clustering algorithms that are unsupervised by nature. The **LFR** (Lancichinetti, Fortunato, Radicchi) model [18, 20] generates networks with communities and at the same time it allows for the heterogeneity in the distributions of both node degrees and of community sizes. It became a standard and extensively used method for generating artificial networks.

In this paper, we analyze the Artificial Benchmark for Community Detection (**ABCD** graph) [14] that was recently introduced and implemented,[1] including a fast implementation that uses multiple threads (**ABCDe**).[2] Undirected variant of **LFR** and **ABCD** produce graphs with comparable properties but **ABCD**/**ABCDe** is faster than **LFR** and can be easily tuned to allow the user to make a smooth transition between the two extremes: pure (disjoint) communities and random graph with no community structure. More importantly from the perspective of this paper, it is easier to analyze theoretically.

The key ingredient for many clustering algorithms is *modularity*, which is at the same time a global criterion to define communities, a quality function of community detection algorithms, and a way to measure the presence of community structure in a network. The definition of modularity for graphs was first introduced by Newman and Girvan in [25].

Despite some known issues with this function such as the "resolution limit" reported in [10], many popular algorithms for partitioning nodes of large graphs use it [8, 19, 24] and perform very well. The list includes one of the mostly used unsupervised algorithms for detecting communities in graphs, the *Louvain* (hierarchical) algorithm [4]. For more details we direct the reader to any book on complex networks, including the following recent additions [15, 17].

1.1 Summary of Results

In this paper, we investigate the modularity function for the **ABCD** model $\mathcal{A}$. The paper is structured as follows. The **ABCD** model is introduced in Sect. 2.2 and the modularity function is defined in Sect. 2.3. Results for other random graph model in the context of the modularity function are summarized in Sect. 3.

We start analyzing the **ABCD** model by investigating some basic properties—see Sect. 4. These properties will be needed to establish results for the modularity function but they are important on their own. In particular, we show that the degree distribution

[1] https://github.com/bkamins/ABCDGraphGenerator.jl/.

[2] https://github.com/tolcz/ABCDeGraphGenerator.jl/.

is well concentrated around the corresponding expectations. Moreover, we show a concentration for the number of communities and well as for the distribution of their sizes. The same generating process is applied in **LFR** so the two results hold for that model as well. The **ABCD** model assigns nodes to communities randomly. Clearly, there is no hope to predict the volumes of small communities of constant size but sufficiently large communities have their volumes as well as the number of internal edges well concentrated around the corresponding expectations.

Then we move to the results for the modularity function. By design of the **ABCD** model, $1-\xi$ fraction of edges should become community edges and so should end up in some part of the ground truth partition $\mathbf{C}$. (ξ is the main parameter of the model responsible for the level of noise.) It is indeed the case but it turns out that a negligible fraction of the background graph join them there. As a result, the modularity function of the ground-truth partition $\mathbf{C}$ is asymptotic to $1-\xi$, as proved in Theorem 1.

Analyzing the maximum modularity is much more complex. We have two types of results. The first result (Theorem 2) shows that when the level of noise is sufficiently large (ξ close to one), then the maximum modularity $q^*(\mathcal{A})$ is asymptotically larger than $q(\mathbf{C})$, the modularity of the ground-truth. In this regime, the number of edges within community graphs G_i is relatively small so a partition of the background graph into small connected pieces yields a better modularity function. To show this result, we need to investigate the degree distribution of the background graph which might be of independent interest.

The second set of results is concerned with graphs with low level of noise (ξ close to zero). For these graphs, the situation is quite opposite. It turns out that the ground truth partition is asymptotically the best possible, that is, the maximum modularity $q^*(\mathcal{A})$ is only $o(1)$ away from $q(\mathbf{C})$, the modularity of the ground truth partition $\mathbf{C}$; both of them are asymptotic to $1-\xi$ (see Theorem 3). For some technical reason, it is assumed that δ, the minimum degree of $\mathcal{A}$, is sufficiently large: the lower bound of 100 easily works but it may be improved with more detailed treatment. Having said that, it seems that one needs a different approach to uncover the real bottleneck. On the other hand, the above property is not true if $\delta = 1$ (see Theorem 4): if $\delta = 1$, then $q^*(\mathcal{A})$ is substantially larger than $q(\mathbf{C})$, regardless of how close to zero ξ is.

Finally, let us mention that all proofs, statements of various technical lemmas, and results of simulations are omitted in this proceeding version of the paper. For much more details, we direct the reader to the journal counterpart of this short paper that is available on ArXiv [11].

1.2 Simulations

This paper focuses on asymptotic theoretical results of the ABCD model. Having said that, we performed a number of simulations and compared asymptotic predictions with graphs generated by computer. These simulations show that the behaviour of small random instances is similar to what is predicted by the theory. This is a good news for practitioners as it shows that, despite the fact that the generative algorithm

is randomized, the model has good stability. We discuss the results of simulations in the full journal version of the paper. The code with experiments is accessible on GitHub repository.[3]

1.3 Open Problems

Theoretical results and simulations suggest that if δ, the minimum degree of $\mathcal{A}$, satisfies $\delta \geq \delta_0$ for some $\delta_0 \geq 2$, then there exists a constant $\xi_0 = \xi_0(\delta)$ (that possibly depends also on other parameters of the **ABCD** model $\mathcal{A}$) such that the following holds *w.h.p.* (that is, with probability tending to one as $n \to \infty$):

- if $0 < \xi < \xi_0$, then $q^*(\mathcal{A}) \sim q(\mathbf{C})$, where $\mathbf{C}$ is the ground truth partition of the set of nodes of $\mathcal{A}$,
- if $\xi > \xi_0$, then $q^*(\mathcal{A})$ is separated by a constant from $q(\mathbf{C})$.

Our results make the first step towards this conjecture by showing upper and lower bounds for such threshold constant ξ_0, when $\delta_0 = 100$. The bounds for ξ_0 are not close to each other. The next step would be to narrow the gap down or perhaps to determine the threshold value exactly, provided that δ_0 is sufficiently large. Another natural direction would be to decrease the lower bound for δ, that is, to decrease the value of δ_0. We showed that $\delta = 1$ does not have the desired property but maybe $\delta_0 = 2$? Or maybe one can always construct a better partition than $\mathbf{C}$ when $\delta = 2$, regardless how small parameter ξ is? These questions are left as open questions for future investigation.

2 Definitions (of ABCD Model and Modularity)

2.1 Asymptotic Notation

Our results are asymptotic in nature, that is, we will assume that the number of nodes $n \to \infty$. Formally, we consider a sequence of graphs $G_n = (V_n, E_n)$ and we are interested in events that hold *with high probability* (*w.h.p.*), that is, events that hold with probability tending to 1 as $n \to \infty$. It would be also convenient to consider events that hold *with extreme probability* (*w.e.p.*), that is, events that hold with probability at least $1 - \exp(-\Omega((\log n)^2))$. An easy but convenient property is that if a polynomial number of events hold *w.e.p.*, then *w.e.p.* all of them hold simultaneously.

[3] https://github.com/bkamins/ABCDGraphGenerator.jl/.

Table 1 Parameters of the **ABCD** model

Parameter	Range	Description
n	$\mathbf{N}$	Number of nodes
γ	$(2, 3)$	Power-law exponent of degree distribution
δ	$\mathbf{N}$	Minimum degree at least δ
ζ	$\left(0, \frac{1}{\gamma-1}\right]$	Maximum degree at most n^{ζ}
β	$(1, 2)$	Power-law exponent of distribution of community sizes
s	$\mathbf{N} \setminus [\delta]$	Community sizes at least s
τ	$(\zeta, 1)$	Community sizes at most n^{τ}
ξ	$(0, 1)$	Level of noise

2.2 ABCD Model

The **ABCD** model is governed by 8 parameters summarized in Table 1. For a fixed set of parameters, we generate the **ABCD** graph $\mathcal{A}$ following the steps outlined below. Each time we refer to graph $\mathcal{A}$ in this paper, we implicitly (or explicitly, but it happens rather rarely) fix all of these parameters.

Degree Distribution Let $\gamma \in (2, 3)$, $\delta \in \mathbf{N}$, and $\zeta \in (0, 1)$. Degrees of nodes of **ABCD** graph $\mathcal{A}$ are generated randomly following the (truncated) *power-law distribution* $\mathcal{P}(\gamma, \delta, \zeta)$ with exponent γ, minimum value δ, and maximum value $D = n^{\zeta}$. In order to make sure the sum of degrees is even, if needed, we decrease by one the degree of one node of the largest degree.

It is easy to show that for any $\omega = \omega(n)$ tending to infinity as $n \to \infty$ *w.h.p.* the maximum degree of $\mathcal{A}$ is at most $n^{1/(\gamma-1)}\omega$ (of course, by definition, it is deterministically at most n^{ζ}). As a result, for any two values of $\zeta_1, \zeta_2 \in (\frac{1}{\gamma-1}, 1)$ one may couple the two corresponding ABCD graphs $\mathcal{A}$ so that *w.h.p.* they produce exactly the same graph. Hence, for convenience but without loss of generality, we will later on assume that $\zeta \in (0, \frac{1}{\gamma-1}]$.

Distribution of Community Sizes Let $\beta \in (1, 2)$, $s \in \mathbf{N} \setminus [\delta]$, and $\tau \in (\zeta, 1)$. Community sizes of **ABCD** graph $\mathcal{A}$ are generated randomly following the (truncated) *power-law distribution* $\mathcal{P}(\beta, s, \tau)$ with exponent β, minimum value s, and maximum value $S = n^{\tau}$. Communities are generated with this distribution as long as the sum of their sizes is less than n, the desired number of nodes. Suppose that the last community has size z and after adding it to the remaining ones, the sum of their sizes will exceed n by $k \in \mathbf{N} \cup \{0\}$. If $k = 0$, then there is nothing else to do. If $z - k \geq s$, then the size of the last community is reduced to $z - k$ so that the total number of nodes is exactly n. Otherwise, we select $z - k < s$ old communities at random, increase their sizes by one, and remove the last community so that the desired property holds.

The assumption that $\tau > \zeta$ is introduced to make sure large degree nodes have large enough communities to be assigned to. Similarly, the assumption that $s \geq \delta + 1$ is required to guarantee that small communities are not too small and so that they can accommodate small degree nodes.

Assigning Nodes into Communities At this point, the degree distribution ($w_1 \geq w_2 \geq \cdots \geq w_n$) and the distribution of community sizes ($c_1 \geq c_2 \geq \cdots \geq c_\ell$) are already fixed. The final **ABCD** graph $\mathcal{A}$ will be formed as the union of $\ell + 1$ independent graphs: ℓ community graphs $G_i = (C_i, E_i), i \in [\ell]$, and a single background graph $G_0 = (V, E_0)$, where $V = \bigcup_{i\in[\ell]} C_i$. Roughly ξw_i edges incident to node i will, by definition, belong to its own community but a few additional edges from the background graph might end up in that community. In order to create enough room for these edges, node of degree w_i will be allowed to be assigned to a community of size c_j if the following inequality is satisfied:

$$\lceil (1-\xi\phi)w_i \rceil \leq c_j - 1, \qquad \text{where } \phi = 1 - \sum_{k\in[\ell]} (c_k/n)^2.$$

Note that this condition is equivalent to the following one:

$$w_i \leq \frac{c_j - 1}{1-\xi\phi}. \tag{1}$$

An assignment of nodes into communities will be called *admissible* if the above inequality is satisfied for all nodes. We show that there are many admissible assignments. In particular, there are linearly many nodes of degree δ but, fortunately, *w.h.p.* communities of size more than n^ζ (more than the maximum degree) have space for almost all nodes. We select one admissible assignment uniformly at random. Sampling uniformly one of such assignments turns out to be relatively easy from both theoretical and practical points of view.

Distribution of Weights Parameter $\xi \in (0, 1)$ reflects the amount of noise in the network. It controls the fraction of edges that are between communities. Indeed, asymptotically (but not exactly) $1 - \xi$ fraction of edges are going to end up within one of the communities. Each node will have its degree w_i split into two parts: *community degree* y_i and *background degree* z_i ($w_i = y_i + z_i$). Our goal is to get $y_i \approx (1-\xi)w_i$ and $z_i \approx \xi w_i$. However, both y_i and z_i have to be non-negative integers and for each community $C \subseteq V$, $\sum_{i\in C} y_i$ has to be even. Note that since $\sum_{i\in V} w_i$ is even, so is

$$\sum_{i\in V} z_i = \sum_{i\in V} (w_i - y_i) = \sum_{i\in V} w_i - \sum_{C} \sum_{i\in C} y_i.$$

For each community $C \subseteq V$ we identify the *leader*, a node of the largest degree w_i associated with community C. (If many nodes in C have the largest degree, then we arbitrarily select one of them to be the leader.) For non-leaders we split the weights as follows:

$$y_i = \left\lfloor (1-\xi)w_i \right\rfloor \qquad \text{and} \qquad z_i = w_i - y_i,$$

where for a given integer $a \in \mathbf{Z}$ and real number $b \in [0, 1)$ the random variable $\lfloor a + b \rceil$ is defined as

$$\lfloor a + b \rceil = \begin{cases} a & \text{with probability } 1 - b \\ a + 1 & \text{with probability } b. \end{cases} \tag{2}$$

(Note that $\mathbf{E}[\lfloor a + b \rceil] = a(1 - b) + (a + 1)b = a + b$.) For the leader of community C we round $(1 - \xi)w_i$ up or down so that the sum of weights in each cluster is even. If $(1 - \xi)w_i \in \mathbf{N}$ and the sum of weights y_i in C is odd, then we randomly make a decision whether subtract or add one to make the sum to be even.

Creating Graphs As already mentioned, the final **ABCD** graph $\mathcal{A} = (V, E)$ will be formed as the union of $\ell + 1$ independent graphs: ℓ community graphs $G_i = (C_i, E_i)$, $i \in [\ell]$, and a single background graph $G_0 = (V, E_0)$, where $V = \bigcup_{i \in [\ell]} C_i$, that is, $E = \bigcup_{i \in [\ell] \cup \{0\}} E_i$. Each of these $\ell + 1$ graphs will be created independently. The partition $\mathbf{C} = \{C_1, C_2, \ldots, C_\ell\}$ will be called a *ground-truth* partition.

Suppose then that our goal is to create a graph on n nodes with a given degree distribution $\mathbf{w} := (w_1, w_2, \ldots, w_n)$, where $\mathbf{w}$ is any vector of non-negative integers such that $w := \sum_{i \in [n]} w_i$ is even. We define a random multi-graph $\mathcal{P}(\mathbf{w})$ with a given degree sequence known as the **configuration model** (sometimes called the **pairing model**), which was first introduced by Bollobás [5]. (See [3, 27, 28] for related models and results.) We start with w *points* that are partitioned into n *buckets* labelled with labels $v_1, v_2, \ldots, v_n$; bucket v_i consists of w_i points. It is easy to see that there are $\frac{w!}{(w/2)!2^w}$ pairings of points. We select one of such pairings uniformly at random, and construct a multi-graph $\mathcal{P}(\mathbf{w})$, with loops and parallel edges allowed, as follows: nodes are the buckets $v_1, v_2, \ldots, v_n$, and a pair of points xy corresponds to an edge $v_i v_j$ in $\mathcal{P}(\mathbf{w})$ if x and y are contained in the buckets v_i and v_j, respectively.

2.3 Modularity Function

The modularity function favours partitions of the set of nodes of a graph G in which a large proportion of the edges fall entirely within the parts but benchmarks it against the expected number of edges one would see in those parts in the corresponding **Chung-Lu** random graph model [7] which generates graphs with the expected degree sequence following exactly the degree sequence in G.

Formally, for a graph $G = (V, E)$ and a given partition $\mathbf{A} = \{A_1, A_2, \ldots, A_\ell\}$ of V, the *modularity function* is defined as follows:

$$q(\mathbf{A}) = \sum_{A_i \in \mathbf{A}} \frac{e(A_i)}{|E|} - \sum_{A_i \in \mathbf{A}} \left(\frac{\mathrm{vol}(A_i)}{\mathrm{vol}(V)} \right)^2, \tag{3}$$

where for any $A \subseteq V$, $e(A) = |\{uv \in E : u, v \in A\}|$ is the number of edges in the subgraph of G *induced by* set A, and $\mathrm{vol}(A) = \sum_{v \in A} \deg(v)$ is the *volume* of set A. In

particular, $\mathrm{vol}(V) = 2|E|$. The first term in (3), $\sum_{A_i \in \mathbf{A}} e(A_i)/|E|$, is called the *edge contribution* and it computes the fraction of edges that fall within one of the parts. The second one, $\sum_{A_i \in \mathbf{A}} (\mathrm{vol}(A_i)/\mathrm{vol}(V))^2$, is called the *degree tax* and it computes the expected fraction of edges that do the same in the corresponding random graph (the null model). The modularity measures the deviation between the two.

The maximum *modularity* $q^*(G)$ is defined as the maximum of $q(\mathbf{A})$ over all possible partitions $\mathbf{A}$ of V; that is, $q^*(G) = \max_{\mathbf{A}} q(\mathbf{A})$. In order to maximize $q(\mathbf{A})$ one wants to find a partition with large edge contribution subject to small degree tax. If $q^*(G)$ approaches 1 (which is the trivial upper bound), we observe a strong community structure; conversely, if $q^*(G)$ is close to zero (which is the trivial lower bound), there is no community structure. The definition in (3) can be generalized to weighted edges by replacing edge counts with sums of edge weights. It can also be generalized to hypergraphs [12, 13].

3 Related Results for Random Graphs

Analyzing the maximum modularity $q^*(G)$ for sparse random graphs is a challenging task. The most attention was paid to **random d-regular graphs** $\mathcal{G}_{n,d}$ but even for this family of graphs we only know upper and lower bounds for $q^*(\mathcal{G}_{n,d})$ that are quite apart from each other. For example, for random 3-regular graph $\mathcal{G}_{n,3}$ we only know that *w.h.p.*

$$0.667026 \leq q^*(\mathcal{G}_{n,3}) \leq 0.789998.$$

These bounds were recently proved in [21] but the main goal of that paper was to confirm the conjecture from [22] that *w.h.p.* $q^*(\mathcal{G}_{n,3}) \geq 2/3 + \varepsilon$ for some $\varepsilon > 0$. We refer the reader to [22, 26] for numerical bounds on $q^*(\mathcal{G}_{n,d})$ for other values of $d \geq 3$ and for some explicit but weaker bounds. It is also known that *w.h.p.* $q^*(\mathcal{G}_{n,2}) \sim 1$ [22].

The **binomial random graphs** $\mathcal{G}(n, p)$ were studied in [23] where it was shown that *w.h.p.* $q^*(\mathcal{G}(n, p)) \sim 1$, provided that $pn \leq 1$, On the other hand, *w.h.p.* $q^*(\mathcal{G}(n, p)) = \Theta(1/\sqrt{pn})$, provided that $pn \geq 1$ and $p < 1 - \varepsilon$ for some $\varepsilon > 0$. The modularity of the well-known **Preferential Attachment (PA) model** [2] and the **Spatial Preferential Attachment (SPA) model** [1] was studied in [26]. Finally, the modularity of a model of random geometric graphs on the hyperbolic plane [16], known as the **KPKBV model** after its inventors, was recently studied in [6].

4 Some Properties of ABCD

4.1 Degree Distribution

Let $\gamma \in (2, 3)$, $\delta \in \mathbf{N}$, and $\zeta \in (0, 1)$. Recall that the degrees of nodes of the **ABCD** model are generated randomly following the (truncated) *power-law distribution*

$\mathcal{P}(\gamma, \delta, \zeta)$ with exponent γ, minimum value δ, and maximum value $D = n^{\zeta}$. More precisely, if $X \in \mathcal{P}(\gamma, \delta, \zeta)$, then for any $k \in \{\delta, \delta+1, \dots, D\}$,

$$\begin{aligned} q_k = \Pr(X = k) &= \frac{\int_k^{k+1} x^{-\gamma} dx}{\int_\delta^{D+1} x^{-\gamma} dx} = \frac{k^{1-\gamma} - (k+1)^{1-\gamma}}{\delta^{1-\gamma} - (D+1)^{1-\gamma}} \\ &= (1 + \mathcal{O}(n^{-\zeta(\gamma-1)}) + \mathcal{O}(k^{-1}))\, k^{-\gamma}(\gamma-1)\delta^{\gamma-1}. \end{aligned} \tag{4}$$

The first lemma provides an upper bound for the maximum degree, which justifies our assumption that $\zeta \in (0, 1/(\gamma-1)]$. The second lemma shows that the degree distribution is well concentrated around the expectation. Since the statements are quite technical, we omit them. However, let us mention the following corollary. The volume of all nodes in $\mathcal{A}$ is *w.e.p.* equal to

$$\mathrm{vol}(V) = \sum_{k=\delta}^{D} kY_k = (1 + \mathcal{O}((\log n)^{-1}))\, dn, \qquad \text{where } d := \sum_{k=\delta}^{D} kq_k.$$

4.2 Distribution of Community Sizes

Let $\beta \in (1, 2)$, $s \in \mathbf{N}$, and $\tau \in (\zeta, 1)$. Recall that community sizes of the **ABCD** model are generated randomly following the (truncated) *power-law distribution* $\mathcal{P}(\beta, s, \tau)$ with exponent β, minimum value s, and maximum value $S = n^{\tau}$. More precisely, if $X \in \mathcal{P}(\beta, s, \tau)$, then after following exactly the same computation as in (4) we get that for any $k \in \{s, s+1, \dots, S\}$,

$$\begin{aligned} p_k = \Pr(X = k) &= \frac{\int_k^{k+1} x^{-\beta} dx}{\int_s^{S+1} x^{-\beta} dx} = \frac{k^{1-\beta} - (k+1)^{1-\beta}}{s^{1-\beta} - (S+1)^{1-\beta}} \\ &= (1 + \mathcal{O}(n^{-\tau(\beta-1)}) + \mathcal{O}(k^{-1}))\, k^{-\beta}(\beta-1)s^{\beta-1}. \end{aligned} \tag{5}$$

Our next lemma shows that community sizes of **ABCD** are well concentrated around their expectation. Again, we omit technical statements only reporting that *w.e.p.* the number of communities is equal to

$$\ell = \ell(n) = (1 + \mathcal{O}((\log n)^{-1}))\, \hat{c}\, n^{1-\tau(2-\beta)},$$

where

$$\hat{c} = \frac{2-\beta}{(\beta-1)s^{\beta-1}}.$$

4.3 Assigning Nodes into Communities and Distribution of Weights

Recall that at this point of the process, the degree distribution $(w_1 \geq w_2 \geq \cdots \geq w_n)$ and the distribution of community sizes $(c_1 \geq c_2 \geq \cdots \geq c_\ell)$ are already fixed. In order to assign nodes to communities we will use the following easy and natural algorithm. We consider nodes, one by one, starting from w_1 (high degree node) and finishing with w_n (low degree node). Recall that node i of degree w_i has to be assigned to a community of size c_j so that inequality (1) holds. We assign node w_i randomly to one of the communities that have size larger than $\lceil (1-\xi\phi)w_i \rceil$ and still have some "available spots". We do it with probability proportional to the number of available spots left. One can show that the above simple algorithm generates one of the admissible assignments uniformly at random.

The volumes of small communities are not well concentrated around their means. On the other hand, the volumes of very large communities are well concentrated around their means, as our next lemma shows. As usually, we skip the statement directing the reader to the journal version of this paper.

5 Modularity

5.1 Modularity of the Ground-Truth Partition: $q(\mathbf{C})$

Let us start by investigating the modularity of the ground-truth partition of $\mathcal{A}$.

Theorem 1 *Let* $\mathbf{C} = \{C_1, C_2, \ldots, C_\ell\}$ *be the ground-truth partition of the set of nodes of* $\mathcal{A}$*. Then,* w.e.p.

$$q^*(\mathcal{A}) \geq q(\mathbf{C}) = (1 + \mathcal{O}((\log n)^{-(\gamma-2)}))\,(1-\xi).$$

5.2 Maximum Modularity: $q^*(G)$

As mentioned in Sect. 3, analyzing the maximum modularity $q^*(G)$ for sparse random graphs is a challenging task and typically only bounds for $q^*(G)$ are known that are far apart from each other. Since the **ABCD** model $\mathcal{A}$ is more complex than other sparse random graphs, especially random d-regular graphs, there is no hope for tight bounds for the maximum modularity function but we will make some interesting observations below.

Large Level of Noise Let us start with investigating graphs with a large level of noise, that is, with ξ close to one. For such graphs, one should focus on the background

graph G_0 which involves all but a small fraction of edges. It turns out that G_0 is connected *w.h.p.*, provided that its minimum degree is at least 3, or otherwise *w.h.p.* it has a giant component. By restricting ourselves to a spanning tree of the giant component of G_0, we may partition the set of nodes into small parts such that each part induces a connected graph. This is not much, but for noisy graphs it yields the modularity that is larger than the modularity of the ground-truth partition.

Theorem 2 *Let* $\gamma \in (2,3)$, $\delta \in \mathbf{N}$, $\zeta \in \left(0, \frac{1}{\gamma-1}\right]$, *and* $\xi \in (0,1)$.

(a) If $\xi\delta \geq 3$, *then set* $\alpha = 1$.
(b) If $\xi\delta < 3$, *then there exists a universal constant* $\alpha > 0$ *which depends on the parameters of the model but it is always separated from 0 (that is,* α *is* not *a function of n).*

There exists a partition $\mathbf{C}$ *of the set of nodes V of* $\mathcal{A}$ *such that the following properties hold* w.h.p.

$$q^*(\mathcal{A}) \geq q(\mathbf{C}) \geq (1 + \mathcal{O}(n^{-(1-\zeta)/2}))\frac{2\alpha n}{vol(V)}$$
$$= (1 + \mathcal{O}((\log n)^{-1}))\,\frac{2\alpha}{d}, \qquad \text{where } d = \sum_{k=\delta}^{D} kq_k.$$

(Note that q_i *is defined in (4).).*

Recall that the modularity function of the ground-truth partition is *w.e.p.* asymptotic to $1-\xi$. The above theorem implies that if $\delta \geq 4$ and the graph has a large level of noise, namely, $\xi \geq 3/\delta$ and $\xi > 1 - 2/d$, then *w.h.p.* the modularity function obtained from dissecting the spanning tree of G_0 is larger! The same conclusion can be derived when $\delta \leq 3$ by considering ξ sufficiently close to one.

Low Level of Noise This time we will investigate graphs with a low level of noise, that is, with ξ close to zero. Let us fix a value of $\delta \in \mathbf{N}$ such that $\delta \geq 100$. For any $a \in \mathbf{N}$ and $b \in \mathbf{N} \setminus \{1,2\}$ such that $ab < \delta$, let

$$c(a,b) := \frac{b - 2\sqrt{b-1}}{2b}\,\frac{ab}{ab+b-1} - \frac{b-1}{ab+b-1} - 0.011. \tag{6}$$

Let

$$\xi_0(\delta) := \max_{a\in\mathbf{N}, b\in\mathbf{N}\setminus\{1,2\}, ab<\delta} \min\left(1 - \frac{ab}{\delta}, \frac{c(a,b)}{4}, \frac{1}{20}\right). \tag{7}$$

It is clear that $\xi_0(\delta)$ is a non-decreasing function of δ. Moreover, $\xi_0(100) \approx 0.0217$ (the maximum is achieved for $a = 8$ and $b = 12$), and $\xi_0(\delta) = 1/20$ for $\delta \geq 340$.

Our first result says that **ABCD** graph $\mathcal{A}$ with minimum degree $\delta \geq 100$ and $\xi \in (0, \xi_0(\delta))$ has *w.h.p.* the maximum modularity $q^*(\mathcal{A})$ asymptotically equal to the modularity function on the ground-truth.

Theorem 3 *Let $\delta \in \mathbf{N}$ such that $\delta \geq 100$ and $0 < \xi < \xi_0(\delta)$, where $\xi_0(\delta)$ is defined in (7). Let $\mathbf{C} = \{C_1, C_2, \ldots, C_\ell\}$ be the ground-truth partition of the set of nodes of $\mathcal{A}$. Then,* w.h.p. $q^*(\mathcal{A}) \sim q(\mathbf{C}) \sim 1 - \xi$.

The lower bound of 100 for δ as well as the constants $\xi_0(\delta)$ are not tuned for the strongest result. Since the proof technique we use will not allow us to close the gap anyway, we aimed for a simple argument that works for large enough δ and relatively simple constants. Having said that, the above property is not true for $\delta = 1$; that is, if $\mathcal{A}$ has minimum degree $\delta = 1$, then one may find a partition of the nodes of $\mathcal{A}$ that yields larger modularity than the one associated with the ground-truth.

Theorem 4 *Fix $\delta = 1$ and let $0 < \xi < 1$. Let $\mathbf{C} = \{C_1, C_2, \ldots, C_\ell\}$ be the ground-truth partition of the set of nodes of $\mathcal{A}$. Then,* w.e.p.

$$\begin{aligned} q^*(\mathcal{A}) &\geq (1 + \mathcal{O}((\log n)^{-(\gamma-2)})) \left((1-\xi) + \frac{\xi q_1}{d}\left(2 - \frac{q_1}{d}\right)\right) \\ &> (1 + \mathcal{O}((\log n)^{-(\gamma-2)}))\,(1-\xi) = q(\mathbf{C}), \end{aligned}$$

where q_k is defined in (4) and $d = \sum_{k=\delta}^{D} kq_k$.

References

1. Aiello, W., Bonato, A., Cooper, C., Janssen, J., Prałat, P.: A spatial web graph model with local influence regions. Internet Math. **5**(1–2), 175–196 (2008)
2. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)
3. Bender, E.A., Canfield, E.R.: The asymptotic number of labeled graphs with given degree sequences. J. Combin. Theor. Ser. A **24**(3), 296–307 (1978)
4. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mechan. Theor. Exper. **2008**(10), P10008 (2008)
5. Bollobás, B.: A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. Euro. J. Combin. **1**(4), 311–316 (1980)
6. Chellig, J., Fountoulakis, N., Skerman, F.: The modularity of random graphs on the hyperbolic plane. J. Complex Netw. **10**(1), cnab051 (2022)
7. Chung Graham, F., Lu, L.: Complex Graphs and Networks, no. 107. American Mathematical Society (2006)
8. Clauset, A., Newman, M.E., Moore, C.: Finding community structure in very large networks. Phys. Rev. E **70**(6), 066111 (2004)
9. Fortunato, S.: Community detection in graphs. Phys. Rep. **486**(3–5), 75–174 (2010)
10. Fortunato, S., Barthelemy, M.: Resolution limit in community detection. Proc. Nat. Acad. Sci. **104**(1), 36–41 (2007)
11. Kamiński, B., Pankratz, B., Prałat, P., Théberge, F.: Modularity of the abcd random graph model with community structure. arXiv:2203.01480 (2022)
12. Kamiński, B., Poulin, V., Prałat, P., Szufel, P., Théberge, F.: Clustering via hypergraph modularity. PloS One **14**(11), e0224307 (2019)
13. Kamiński, B., Prałat, P., Théberge, F.: Community detection algorithm using hypergraph modularity. In: International Conference on Complex Networks and Their Applications, pp. 152–163. Springer (2020)

14. Kamiński, B., Prałat, P., Théberge, F.: Artificial benchmark for community detection (ABCD)-fast random graph model with community structure. Netw. Sci. 1–26 (2021)
15. Kamiński, B., Prałat, P., Théberge, F.: Mining Complex Networks (2021)
16. Krioukov, D., Papadopoulos, F., Kitsak, M., Vahdat, A., Boguná, M.: Hyperbolic geometry of complex networks. Phys. Rev. E **82**(3), 036106 (2010)
17. Lambiotte, R., Schaub, M.: Modularity and dynamics on complex networks (2021)
18. Lancichinetti, A., Fortunato, S.: Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Phys. Rev. E **80**(1), 016118 (2009)
19. Lancichinetti, A., Fortunato, S.: Limits of modularity maximization in community detection. Phys. Rev. E **84**(6), 066122 (2011)
20. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Phys. Rev. E **78**(4), 046110 (2008)
21. Lichev, L., Mitsche, D.: On the modularity of 3-regular random graphs and random graphs with given degree sequences. arXiv:2007.15574 (2020)
22. McDiarmid, C., Skerman, F.: Modularity of regular and treelike graphs. J. Complex Netw. **6**(4), 596–619 (2018)
23. McDiarmid, C., Skerman, F.: Modularity of erdős-rényi random graphs. Random Struct. Algorithms **57**(1), 211–243 (2020)
24. Newman, M.E.: Fast algorithm for detecting community structure in networks. Phys. Rev. E **69**(6), 066133 (2004)
25. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E **69**(2), 026113 (2004)
26. Prokhorenkova, L.O., Prałat, P., Raigorodskii, A.: Modularity of complex networks models. Internet Math. (2017)
27. Wormald, N.C.: Generating random regular graphs. J. Algorithms **5**(2), 247–280 (1984)
28. Wormald, N.C., et al.: Models of random regular graphs. In: London Mathematical Society Lecture Note Series, pp. 239–298 (1999)

Learning Attribute Distributions Through Random Walks

Nelson Antunes, Shankar Bhamidi, and Vladas Pipiras

Abstract We investigate the statistical learning of nodal attribute distributions in homophily networks using random walks. Attributes can be discrete or continuous. A generalization of various existing canonical models, based on preferential attachment is studied, where new nodes form connections dependent on both their attribute values and popularity as measured by degree. We consider several canonical attribute agnostic sampling schemes such as Metropolis-Hasting random walk, versions of node2vec (Grover and Leskovec 2016) that incorporate both classical random walk and non-backtracking propensities and propose new variants which use attribute information in addition to topological information to explore the network. The performance of such algorithms is studied on both synthetic networks and real world systems, and its dependence on the degree of homophily, or absence thereof, is assessed.

Keywords Attributed networks · Homophily · Network model · Random walk samplings · Discrete and continuous attributes · Learning distributions

N. Antunes (✉)
Center for Computational and Stochastic Mathematics, University of Lisbon and University of Algarve, Avenida Rovisco Pais, 1049-001, Lisbon, Portugal
e-mail: nantunes@ualg.pt

S. Bhamidi · V. Pipiras
Department of Statistics and Operations Research, University of North Carolina, CB 3260, Chapel Hill, NC 27599, USA
e-mail: bhamidi@email.unc.edu

V. Pipiras
e-mail: pipiras@email.unc.edu

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_2

1 Introduction

Attributed networks, namely graphs in which nodes and/or edges have attributes, are at the center of network-valued datasets in many modern applications. In one direction, machine learning pipelines such as network representation learning [10], clustering [8], classification [17], and community detection [6] have been developed to study the entire network. Driven by the scale of data, the main motivation of this paper, is network sampling, where limited explorations are used to learn network level functionals such as the degree distribution [19].

One standard phenomenon in many such real world systems is *homophily* [18, 20, 22], i.e., node pairs with similar attributes being likelier connected than node pairs with discordant attributes. Performance of network sampling algorithms in such settings has received some attention including: the bias of several sampling methods in conserving position of nodes and visibility of groups [23]; the effect of homophily on centrality measures and visibility of minority groups and fairness questions [14]. This paper studies the estimation of the attribute distribution (both discrete and continuous) for homophily networks. We extend the attributed driven preferential attachment model [13, 14] where new nodes connect to existing ones based on the attributes of both end points of the potential edge and centrality of the existing vertex. Uniform random sampling of nodes or edges is the "gold standard", providing unbiased estimates of corresponding attribute distributions. However, owing to both computational and privacy issues in settings such as social networks, such sampling is often infeasible. In these cases, link trace sampling, such as random walks (RW) are typically used; see references in [3, 4] for estimation of functionals such as degree distribution and clustering. Much less is known in the context of attribute distribution estimation. In this paper, we consider several canonical attribute agnostic sampling schemes such as Metropolis-Hasting random walk, versions of node2vec [12] that incorporate both classical random walk and non-backtracking propensities and propose variants of node2vec where edge weights depend on attributes of the node pair. The performance of the considered random walk sampling schemes in terms of estimation error of the attribute distributions is studied across the following four dimensions in both synthetic and real world settings: (**a**) Inherent homophilc propensity of the network and underlying density of attributes; (**b**) Impact of centrality of nodes as measured by degree in the evolution of the network; (**c**) Nonlinear impact of incorporating "escape echo chamber" mechanisms in random walks by encouraging walks to jump across edges with discordant attributes; (**d**) Impact of reducing the backtracking propensity to encourage walks to explore the network.

Overview of findings and organization of the paper: We find that (i) RWs with attribute dependent weights can perform better over attribute agnostic RWs in homophilic networks; (ii) the weights need to balance the movements between/within nodes with different/same attributes; (iii) non-backtracking seems to improve performance, especially in conjunction with attribute dependent weights; (iv) the performance of RWs is well below the "gold standard" of random node sampling; (v) methods seem to work comparably well for discrete and continuous attributes.

The paper is organized as follows. A synthetic model with homophily is given in Sect. 2. Sampling schemes for learning attribute distribution are described in Sect. 3. Statistical learning tasks are discussed in Sect. 4. Numerical evaluation on synthetic and real data are described in Sect. 5. Section 6 concludes.

2 Attribute Network Models with Homophily

We now describe the main synthetic model, termed non-linear preferential attachment (NLPA) model with homophily. Fix an attribute (or latent) space $\mathcal{A}$ with probability measure μ. Fix a (potentially asymmetric) function $f: \mathcal{A} \times \mathcal{A} \to \mathbb{R}_+$ which measures propensities of node pairs to interact based on their attributes. Fix $\alpha \geq 0$ playing the role of degree in measuring popularity. Let N be the number of nodes (vertices) in the network. Nodes $\{v_t : 1 \leq t \leq N\}$ enter the system sequentially starting at $t = 1$ with a base connected graph $\mathcal{G}_1$ with every node having an attribute in $\mathcal{A}$. Every node v_t has attribute $a(v_t) \in \mathcal{A}$ generated independently using μ. The dynamics are recursively defined as follows: for any t and $v \in \mathcal{G}_t$, let $\deg(v, t)$ denote the degree of v at time t. Conditional on $\mathcal{G}_t$, the probability that v_{t+1} connects to $v \in \mathcal{G}_t$ is proportional to:

$$P_{v_{t+1}v} \propto f(a(v), a(v_{t+1}))[\deg(v, t)]^{\alpha}. \tag{1}$$

The model (1) extends various existing models including: Barabási-Albert model [5] ($f \equiv 1, \alpha = 1$), sublinear PA [16] ($f \equiv 1, 0 < \alpha < 1$), PA with multiplicative fitness [7] ($f(a, a') = a, \alpha = 1$), scale free homophilic model [9] ($f(a, a') = 1 - |a - a'|, \mathcal{A} = [0, 1], \alpha = 1$), and geometric versions with $\alpha = 1$, $\mathcal{A}$ a compact metric space and f an appropriate function of the distance [11, 13]. Most existing studies focus on asymptotics for either the degree distribution or maximal degree.

When the latent space $\mathcal{A} = \{1, 2, \ldots, K\}$ is finite, one can define, macroscopic measures of homophily, and the converse heterophily from an observed network $\mathcal{G}$ (either synthetic or empirically observed) on N nodes as follows [21]. Let $\mathcal{E}$ denote the total edge set; for $a \in \mathcal{A}$, $\mathcal{V}_a$ the set of nodes of type a, and for $a, a' \in \mathcal{A}$, let $\mathcal{E}_{aa'}$ be the set of edges between nodes of type a and a'. Let $p = |\mathcal{E}|/\binom{N}{2}$ be the edge density. For $a \in \mathcal{A}$, $D_a = |\mathcal{E}_{aa}|/(\binom{|\mathcal{V}_a|}{2} p)$ measures the contrast in edges within the cluster of nodes a as compared to a setting where all edges are randomly distributed; thus $D_a > 1$ signals homophilic characteristics of type a nodes while $D_a < 1$ signifies heterophilic nature of type a. Similarly, for $a \neq a'$, $H_{aa'} = |\mathcal{E}_{aa'}|/(|\mathcal{V}_a||\mathcal{V}_{a'}|p)$ denotes propensity of type a nodes to connect to type a' nodes as contrasted with random placement of edges at the same level as the global edge density.

An illustration of synthetic networks generated using the NLPA model (1) with finite latent space is given in Fig. 1. Here, $\mathcal{A} = \{1, 2, 3\}$ represent 70, 20 and 10% of the total $N = 1000$ nodes, resp.; $f(a, a) = 0.95$, $f(a, a') = 0.025$, for $a \neq a' = 1, 2, 3$. The network is plotted for different values of α—Fig. 1a–c. For $\alpha = 0.2$, the

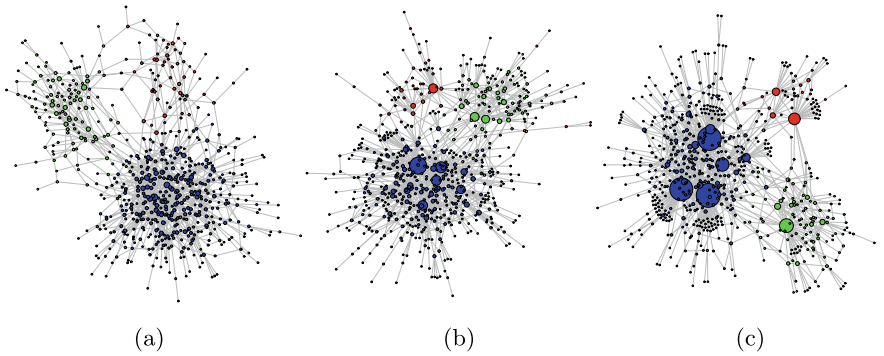


Fig. 1 Networks generated by the NLPA model with **a** $\alpha = 0.2$, **b** $\alpha = 1$, **c** $\alpha = 1.2$

corresponding homophily measures are $D_1 = 1.45$, $D_2 = 4.36$, $D_3 = 7.38$, $H_{12} = 0.07$, $H_{13} = 0.14$, $H_{23} = 0.45$. For $\alpha = 1.2$, the homophily measures are $D_1 = 1.38$, $D_2 = 4.84$, $D_3 = 9.12$, $H_{12} = 0.08$, $H_{13} = 0.08$, $H_{23} = 0.16$.

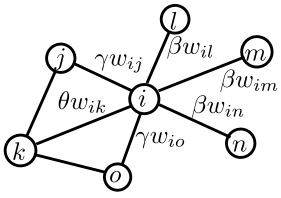
3 Network Sampling Schemes

This section describes sampling schemes for learning attribute distribution, both random walk based, as well as corresponding "gold standard" schemes. Throughout this section, for graph $\mathcal{G}$ and node $i \in \mathcal{G}$, d_i will denote its degree.

Metropolis Hasting Random Walk (MHRW). At each step, if the walk is currently at node i, a neighbor j is selected uniformly at random and the proposed move to j is accepted with probability $\min(1, d_i/d_j)$, else the walk stays at i. Thus proposed moves towards a node of smaller degree, are always accepted whilst we reject some of the proposed moves towards higher degree nodes. It is easy to check that the stationary distribution is uniform over the node set, i.e., $\pi_i = 1/N$ for $1 \leq i \leq N$.

Node2vec (N2V). As proposed in [12], in full generality, the transitions of N2V depend on the neighborhood both of the currently visited node, and the node visited prior to the current node. Let the previous and current visited nodes be k and i, resp. The next visited node j is chosen according to the transition probability proportional to:

$$p(j|k,i) \propto \begin{cases} \beta w_{ij}, & k \neq j, (k,j) \notin \mathcal{E}, \\ \gamma w_{ij}, & (k,j) \in \mathcal{E}, \\ \theta w_{ij}, & k = j, \end{cases}$$



where w_{ij} is the weight of edge (i, j)—see figure. We now describe specific variants of this class of random walks.

Node2vec-1(N2V-1): If the network is undirected, unweighted and $\theta = \beta = \gamma$, one obtains the classical RW with the well-known stationary distribution,

$$\pi_i = \frac{d_i}{2|\mathcal{E}|}. \tag{2}$$

Node2vec-2(N2V-2): If the network is undirected and $\theta = \beta = \gamma$, one obtains a weighted RW. This walk can use node attributes through weights in contrast to N2V-1. The stationary distribution in this case is given by

$$\pi_i \propto \sum_j w_{ij}. \tag{3}$$

Node2vec-3(N2V-3): If the network is simple (i.e. unweighted, undirected, without self-loops and multiple edges) and $\beta = \gamma, \theta > 0$, the stationary distribution for nodes is given by Eq. (2). With small θ, the walk approaches the non-backtracking random walk.

Node2vec-4(N2V-4): One can consider other variants of N2V. We consider below the combination of the last two schemes, with $\beta = \gamma, \theta > 0$ and weights w_{ij} dependent on the attributes of i and j. In this setting, one major technical hurdle is that, unlike the settings above, there is no explicit formula for the stationary distribution. Analogous to the stationary distribution for N2V-3 matching the usual RW in the stationary regime, it is expected that especially in the small θ setting, the stationary distribution can still be approximated by that in Eq. (3). We explore the efficacy of this approximation for moderate size synthetic networks below.

For comparison to RWs, we will also use the following baseline samplings. These can be viewed as "ideal" for sampling purposes and correspond to the limiting distributions of some RWs.

Node Sampling (NS). NS sampling requires full access to the network and is unavailable for many real networks. In the classical NS, nodes (and their attributes) are chosen independently and uniformly from the network (with replacement).

Edge Sampling (ES). In the classical ES, edges are chosen independently and uniformly from the network. Since ES selects edges rather than nodes to populate the sample, the node (attribute) set is constructed by including both incident nodes (attributes) in the sample when a particular edge is sampled.

4 Statistical Learning Methods

We now discuss the estimation of attribute distributions from the data collected through RWs, with discrete attributes described in Sect. 4.1 and continuous attributes in Sect. 4.2.

4.1 Discrete Attributes

Run a random walk (any of the schemes described in Sect. 3) for n steps and let i_s denote the s-th node sampled by a RW, for $1 \leq s \leq n$. Since nodes are sampled with replacement and with probabilities π_i in the stationary regime, the attribute distribution can be estimated as

$$\hat{p}(a) = \frac{1}{Nn} \sum_{s=1}^{n} \frac{\mathbf{1}\{a(i_s) = a\}}{\pi_{i_s}}, \qquad a \in \mathcal{A}, \tag{4}$$

where $\mathbf{1}\{B\} = 1$ if B is true and 0 otherwise [15] (Chap. 5). If the total number of nodes N is unknown, its estimator is given by $(1/n) \sum_s 1/\pi_{i_s}$. For N2V-2 this results in,

$$\hat{p}(a) = \frac{1}{\sum_{s=1}^{n} 1/w_{i_s}} \sum_{s=1}^{n} \frac{\mathbf{1}\{a(i_s) = a\}}{w_{i_s}}, \qquad a \in \mathcal{A}. \tag{5}$$

For fixed a, the MSE of $\hat{p}(a)$ is given by $E[(\hat{p}(a) - p(a))^2]$. In the stationary regime, $\hat{p}(a)$ in (4) is an unbiased estimator of $p(a)$ and the MSE is equal to the variance $V[\hat{p}(a)]$. The variance of $\hat{p}(a)$ can be related to the spectral gap of the RW. More specifically, let P be the associated transition matrix of the random walk with eigenvalues (real by reversibility): $1 = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N \geq -1$. The spectral gap is defined as $\delta = 1 - \lambda_2$. Equivalently, the relaxation time of the RW is the reciprocal of the spectral gap. A larger spectral gap implies a faster convergence of the RW to its stationary distribution. From [1] (Proposition 4.29), we have

$$V(\hat{p}(a)) \leq \frac{2\Lambda(a)}{\delta n} \left(1 + \frac{\delta}{2n}\right), \tag{6}$$

where $\Lambda(a) = \sum_{i=1}^{N} \mathbf{1}\{a(i) = a\}/(N^2 \pi_i)$. The error in estimating the proportion of nodes with attribute a is upper bounded by the inverse of the spectral gap and $\Lambda(a)$, the latter is small if the probability of sampling nodes with attribute a is large. We will see in Sect. 5 that for N2V-2, if edge weights w_{ij} are *inversely* related to the concordance of the attributes, thus encouraging the walk to explore vertices with different attributes, then in some settings, this increases δ and decreases $\Lambda(a)$ (for attributes with small proportions), resulting in a smaller variance of the estimator.

4.2 Continuous Attributes

Let $g(\cdot)$ be the density of the continuous attributes, and as before $(i_s: 1 \leq s \leq n)$ be the states visited by the RW with corresponding attributes $(a(i_s): 1 \leq s \leq n)$. Analogous to (5) the natural estimate for $g(\cdot)$ is through standard kernel smoothing as

$$\hat{g}(a) = \sum_{s=1}^{n} K\left(\frac{a - a(i_s)}{h}\right) \frac{1}{h} w_s, \tag{7}$$

where $h > 0$ is a bandwidth, K is a kernel function, and the weights w_s satisfy

$$w_s \propto \frac{1}{\pi_{i_s}}, \quad \sum_{s=1}^{n} w_s = 1. \tag{8}$$

The performance of the estimator can be assessed through the estimation error: for $q > 0$,

$$error = \left[\int |\hat{g}(a) - g(a)|^q da\right]^{1/q}. \tag{9}$$

The values of q usually considered are 1 and 2.

5 Numerical Studies

5.1 Synthetic Networks

We consider the NLPA model in (1) for networks with attributes and explore the effect of homophily on the accuracy of the RWs to estimate the attribute distribution in a controlled setting.

Discrete Attributes. The total number of nodes is $N = 2000$ with attributes labeled $a = 1, 2, 3$. If a node attribute is selected at random, its p.m.f. is given by $p(1) = 0.7$, $p(2) = 0.2$ and $p(3) = 0.1$. The tendency of two nodes to connect according to the NLPA model is $f(a, a) = 0.9$, $f(a, a') = 0.05$, $a, a' = 1, 2, 3$, $a \neq a'$. Consider first the case $\alpha = 0.2$, where the number of nodes with a large degree tends to be smaller—see Fig. 1a. For the largest component of the generated network, the homophily measures are $D_1 = 1.39$, $D_2 = 3.93$, $D_3 = 7.26$, $H_{12} = 0.17$, $H_{13} = 0.10$, $H_{23} = 0.35$.

The network attributes are sampled with the different RWs on the largest component and the p.m.f. of the attributes is estimated using (4). Table 1 shows the standard deviations of the estimates using 300 runs for each RW with length $0.15N$. The MH walk presents the worst performance. Compared to the baseline method NS that

Table 1 Standard deviations, spectral gaps and quantities $\Lambda(3)$, under the PA model with $\alpha = 0.2$ and attribute values $a = 1, 2, 3$

Random walks	MH	N2V-1	N2V-2^a	N2V-2^b	N2V-2^c	N2V-3	N2V-4	NS	ES
st. dev.; $a = 1$	0.235	0.172	0.199	0.142	0.169	0.123	0.102	0.029	0.051
st. dev.; $a = 2$	0.198	0.152	0.176	0.127	0.151	0.107	0.089	0.025	0.042
st. dev.; $a = 3$	0.137	0.077	0.098	0.069	0.086	0.065	0.049	0.018	0.035
Spectral gap (δ)	0.019	0.048	0.037	0.040	0.011	0.107	0.106	–	–
$\Lambda(3)$	0.088	0.149	0.165	0.135	0.243	0.149	0.135	–	–

For the RW weights: N2V-2^a ($\overline{w}_{aa} = 1.5, \overline{w}_{aa'} = 1$), N2V-$2^b$ ($\overline{w}_{aa} = 0.3, \overline{w}_{aa'} = 1$), N2V-$2^c$ ($\overline{w}_{aa} = 0.05, \overline{w}_{aa'} = 1$)

samples nodes according to the limit stationary distribution of MH, the difference in variability is large. The N2V-1 walk performs the worst among the variants of N2V. It represents the classical RW since edges are sampled at random in its stationary limit. However, the variability of the baseline method ES is smaller. The results for MH and N2V-1 can also be explained through the bound of the variance (6). The spectral gap δ is sufficiently larger for N2V-1, resulting in a lower variability for attribute $a = 3$, in spite of smaller $\Lambda(3)$ for MH.

We examine how the different choices of weights affect the performance of N2V-2. We write $\overline{w}_{aa}$ for the weights of nodes with the same attributes, and $\overline{w}_{aa'}$ with different attributes. If $\overline{w}_{aa}$ is greater than $\overline{w}_{aa'}$ (N2V-2^a in Table 1), the RW hardly transits from one attribute value to another, which creates a bottleneck for approaching the stationary probability. On the other hand, if $\overline{w}_{aa}$ is smaller than $\overline{w}_{aa'}$ (N2V-2^b), movements between different attribute values are more frequent, accelerating the convergence. In this case, the spectral gap increases. However, as the difference between $\overline{w}_{aa'}$ and $\overline{w}_{aa}$ increases (N2V-2^c), the convergence is decelerated because exploration within the same attribute is not sufficient due to the inter-attribute moves. We also see that if $\overline{w}_{aa'}$ is greater than $\overline{w}_{aa}$ until a certain point, the probability of the random walker of sampling nodes with attribute $a = 3$ increases and $\Lambda(3)$ decreases [see the discussion below (6)]. The tradeoff between δ and $\Lambda(a)$ explains the smaller variability for the three attribute values of N2V-2^b, which outperforms N2V-1.

In N2V-3, the parameter θ of the propensity for the random walk to backtrack is decreased to $\theta = 10^{-3}$ and $\beta = \gamma = 1$ are kept for the other two parameters. (Note that if the walker arrives at a node with degree 1, it always backtracks in the next time step since this is the only possible move.) In this case, a random walker tends to explore better the network within the same attribute value, which accelerates the convergence. The result is consistent with the non-backtracking RWs on regular graphs [2]. In many cases, they find spectral gap "twice as good" compared to the classical RW, as also in our case.

N2V-4 combines features of both weighted and non-backtracking RWs. We use the same weights and backtracking parameter as in N2V-2^b and N2V-3, resp. Since the stationary distribution is not known, we approximate it using (3). The choice is heuristic but the results show that N2V-4 has lower variability. This can be explained

Table 2 Standard deviations for various RWs (N2V-2 with $\overline{w}_{aa} = 0.3$, $\overline{w}_{aa'} = 1$), under the NLPA model with $\alpha = 1$ and with/without homophily and attribute values $a = 1, 2, 3$

	Random walks	MH	N2V-1	N2V-2	N2V-3	N2V-4	NS	ES
With homophily	st. dev.; $a = 1$	0.291	0.153	0.131	0.126	0.100	0.031	0.054
	st. dev.; $a = 2$	0.256	0.131	0.107	0.108	0.090	0.028	0.045
	st. dev.; $a = 3$	0.160	0.094	0.073	0.068	0.059	0.021	0.036
Without homophily	st. dev.; $a = 1$	0.146	0.059	0.055	0.049	0.045	0.029	0.040
	st. dev.; $a = 2$	0.116	0.055	0.051	0.041	0.039	0.025	0.036
	st. dev.; $a = 3$	0.110	0.039	0.036	0.031	0.024	0.018	0.027

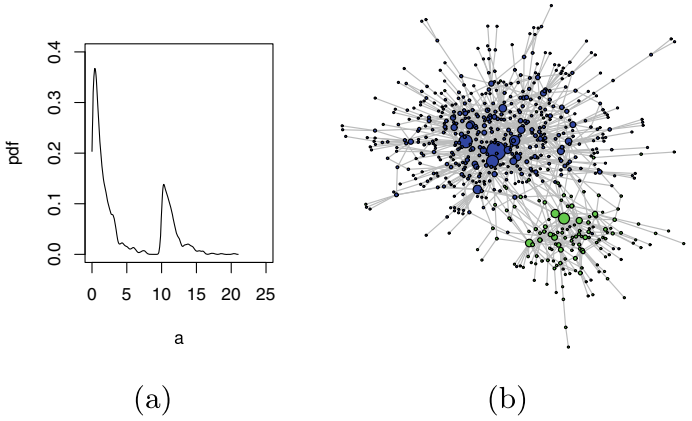


Fig. 2 **a** Probability density function of attributes estimated using kernel smoothing, **b** the generated NLPA network with attributes less than 10 (blue) and greater than 10 (green)

by the decrease of $\Lambda(a)$ for attribute values 2 and 3 (see $\Lambda(3)$ for N2V-3 and N2V-4 while δ is approximately equal). We have confirmed these findings by using the true stationary distribution of N2V-4 obtained through simulation.

We next consider the NLPA network with $\alpha = 1$ and take its remaining parameters as above. For the largest component of the network, the homophily measures are $D_1 = 1.38$, $D_2 = 4.30$, $D_3 = 6.25$, $H_{12} = 0.16$, $H_{13} = 0.23$, $H_{23} = 0.32$. The standard deviation of 300 runs for each RW is given in Table 2. In this case, the standard deviation of MH increases and of N2V-1 decreases. This can be explained by nodes with different attribute values attracted to high degree nodes— see Fig. 1b. Unlike the case $\alpha = 0.2$, the RWs which are attracted by high degree nodes will benefit from this to move between different attribute values. The same conclusions can be drawn as above for the other variants of N2V.

Finally, we consider a network without homophily where f is constant and $\alpha = 1$. The results are shown in Table 2. As seen from the table, if the homophily decreases, the differences between the RWs tend to be smaller.

Table 3 Estimation error and spectral gap for various RWs, under the NLPA model

Random walks	MH	N2V-1	N2V-2	N2V-3	N2V-4	NS	ES
Average error	0.818	0.487	0.457	0.385	0.364	0.186	0.270
Spectral gap (δ)	0.005	0.042	0.051	0.080	0.096	–	–

Continuous Attributes. We consider the NLPA model with N=2000 nodes and $\alpha = 1$. Nodes have continuous attributes with values drawn independently from the following probability distribution. Let X be a gamma random variable with shape and scale parameters 1 and 1.5, resp. For the attributes, we draw $0.7N$ and $0.3N$ independent random variables X and $10 + X$, resp. The density function of attributes estimated using kernel smoothing is shown in Fig. 2a. Additionally, we set

$$f(a(i), a(j)) = \begin{cases} 0.95, & a(i), a(j) < 10 \text{ or } a(i), a(j) > 10, \\ 0.05, & \text{otherwise.} \end{cases} \tag{10}$$

The network generated is plotted in Fig. 2b, where nodes are divided in two groups: with attributes less than 10 (group 1) and greater than 10 (group 2). The homophily measures are $D_1 = 1.378$, $D_2 = 3.092$, $H_{12} = H_{21} = 0.112$.

For N2V-2, the weights are taken as $w_{ij} = |a(i) - a(j)|^b$, which allows moving between the groups of nodes but also giving more weight to edges with different values within each group. The choice of b is motivated by similar arguments as in the case of discrete attributes. If the weights between edges of different groups are too large, then the convergence is decelerated because exploration within the same group attribute is not sufficient due to the inter-group moves. From the experiments, we found that values of b close to zero decrease the range of weights and show good results.

The network attributes are sampled with the different sampling methods on the largest component and the density function of the attributes is estimated using (7). Table 3 shows the average of the estimation error (9) with $q = 1$ and the spectral gap from 300 runs for each RW with length $0.15N$. We fixed $b = 0.3$ (N2V-2/4) and $\theta = 10^{-3}$ (N2V-3/4). The performance of the samplings methods is akin to the case of the discrete attributes.

5.2 *Real Networks*

We analyze two publicly available datasets of real networks with attributes and homophily.[1]

Discrete Attributes. The dataset is a webgraph of Facebook sites. Nodes represent pages while the links are mutual likes between sites. Node features were extracted

[1] https://snap.stanford.edu/data/.

Table 4 St. dev. of the estimates

Random walks	N2V-1	N2V-2	N2V-3
Politician (1)	0.052	0.0561	0.0489
Government (2)	0.051	0.046	0.043
TV show (3)	0.047	0.045	0.038
Company (4)	0.0669	0.058	0.054

from the site descriptions that the page owners created to summarize the purposes of the sites. The graph was collected through the Facebook Graph API and restricted to pages from four attributes which are defined by Facebook. These attributes are: politicians (1), governmental organizations (2), television shows (3) and companies (4). We consider the simplified network which has $N = 22,470$ nodes and 170, 823 edges. The distribution of node attributes is $p(1) = 0.31$, $p(2) = 0.26$, $p(3) = 0.29$, and $p(4) = 0.14$. The homophily measures are $D_1 = 3.28$, $H_{1*} = 0.17$, $D_2 = 5.08$, $H_{2*} = 0.21$, $D_3 = 3.46$, $H_{3*} = 0.14$, $D_4 = 1.41$, $H_{4*} = 0.11$, where H_{a*} denotes the propensity of attribute a nodes to connect to the other types of attributes.

To estimate the p.m.f. of the node attributes, we consider only the variants of N2V with known stationary distributions. For N2V-2, we set the weights as $\overline{w}_{aa} = 0.3$, $\overline{w}_{aa'} = 1, a, a' = 1, 2, 3, 4, a \neq a'$, and for N2V-3, we set $\theta = 10^{-3}$. Table 4 shows the standard deviations of the estimates using RWs of length $0.15N$ and 500 runs. The results are in line with the synthetic model with discrete attributes where sampling with N2V-3 produces more accurate estimates.

Continuous Attributes. Pokec is a social network with attributes from Slovakia. We use the age attribute viewed as continuous as in [24]. Considering only the nodes with age attributes results in a network with N=1,138,314 nodes and 22,301,601 edges—see Fig. 3. It is well known that the network is moderately homophilic with respect to age. If we divide the nodes in two groups: say, age less or equal to 37 (group 1) and greater than 37 (group 2), the homophilic measures of the groups are $D_1 = 1.166$, $H_{12} = 0.30$ and $D_2 = 1.46$. Group 2 represents 9% of the total number of nodes. The average of the estimation errors from 20 runs for each RW with length $0.05N$ are: 0.036 (N2V-1), 0.031 (N2V-2 with $b = 0.2$), 0.030 (N2V-3 with $\theta = 10^{-3}$).

6 Discussion and Future Directions

In this paper, we developed a statistical learning framework for the attribute distributions in networks and evaluated numerically the impact of homophily, degree centrality, and random walk exploration mechanisms on estimation accuracy. The results seem to indicate intricate non-linear relationship between intrinsic homophilic characteristics of the network, parameters modulating random walk exploration schemes and the error of proposed learning algorithms. Untangling the precise relationship

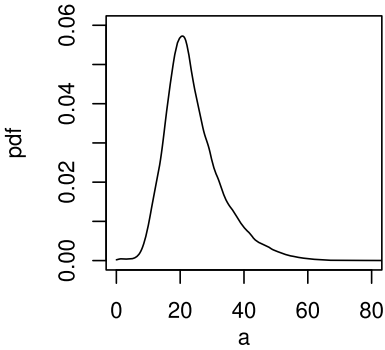


Fig. 3 P.d.f. of the age

will require careful theoretical understanding both of macroscopic functionals such as the spectral gap of proposed RWs and their relationship to parameters such as backtracking propensities and jump rates across different attribute sets, as well as microscopic functionals such as asymptotics for local neighborhoods of the underlying network. This should lead to more principled ways of choosing RWs and their parameters in terms of the network homophily, centrality and possibly other measures.

Random walks are also closely tied to ranking mechanisms such as the Page-rank centrality, and we plan to study the impact of the parameters driving the random walk on such centrality scores, thus looping back to one of the central motivations for studying attributed networks namely fairness of ranking mechanisms [14]. Other questions, including learning joint distributions of the degree and the attribute through sampling mechanisms, as well as multivariate attribute distributions, both in terms of developing synthetic models, as well as real world data will also be considered.

Acknowledgements The second and third authors were supported in part by the NSF grants DMS-2113662 and DMS-2134107.

References

1. Aldous, D., Fill, J.A.: Reversible Markov chains and random walks on graphs, 2002. Unfinished monograph, recompiled 2014. Available at http://www.stat.berkeley.edu/~aldous/RWG/book.html
2. Alon, N., Benjamini, I., Lubetzky, E., Sodin, S.: Non-backtracking random walks mix faster. Commun. Contempor. Math. **09**(04), 585–603 (2007)
3. Antunes, N., Bhamidi, S., Guo, T., Pipiras, V., Wang, B.: Sampling based estimation of in-degree distribution for directed complex networks. J. Comput. Graph. Stat. **30**(4), 863–876 (2021)
4. Antunes, N., Guo, T., Pipiras, V.: Sampling methods and estimation of triangle count distributions in large networks. Netw. Sci. **9**(S1), S134–S156 (2021)
5. Barabási, A.-L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)

6. Baroni, A., Conte, A., Patrignani, M., Ruggieri, S.: Efficiently clustering very large attributed graphs. In: 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 369–376 (2017)
7. Bianconi, G., Barabási, A.-L.: Bose-Einstein condensation in complex networks. Phys. Rev. Lett. **86**(24), 5632 (2001)
8. Chang, C.-H., Chang, C.-S., Chang, C.-T., Lee, D.-S., Lu, P.-E.: Exponentially twisted sampling for centrality analysis and community detection in attributed networks. IEEE Trans. Netw. Sci. Eng. **6**(4), 684–697 (2019)
9. de Almeida, M.L., Mendes, G.A., Madras Viswanathan, G., da Silva, L.R.: Scale-free homophilic network. Euro. Phys. J. B **86**(2), 38 (2013)
10. Fan, H., Zhong, Y., Zeng, G., Sun, L.: Attributed network representation learning via improved graph attention with robust negative sampling. Appl. Intell. **51**(1), 416–426 (2021)
11. Flaxman, A.D., Frieze, A.M., Vera, J.: A geometric preferential attachment model of networks II. Internet Math. **4**(1), 87–111 (2007)
12. Grover, A., Leskovec, J.: Node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, pp. 855–864. Association for Computing Machinery, New York, NY, USA (2016)
13. Jordan, J.: Geometric preferential attachment in non-uniform metric spaces. Electron. J. Probab. **18**, 1–15 (2013)
14. Karimi, F., Génois, M., Wagner, C., Singer, P., Strohmaier, M.: Homophily influences ranking of minorities in social networks. Sci. Rep. **8**(1), 11077 (2018)
15. Kolaczyk, E.D.: Statistical Analysis of Network Data. Springer Series in Statistics (2009)
16. Krapivsky, P.L., Redner, S.: Organization of growing random networks. Phys. Rev. E **63**(6), 066123 (2001)
17. Lee, D.J.L., Han, J., Chambourova, D., Kumar, R.: Identifying fashion accounts in social networks. In: In Proceedings of the KDD Workshop on ML Meets Fashion (2017)
18. McPherson, M., Smith-Lovin, L., Cook, J.M.: Birds of a feather: homophily in social networks. Ann. Rev. Sociol. **27**(1), 415–444 (2001)
19. Meng, L., Masuda, N.: Analysis of node2vec random walks on networks. Proc. R. Soc. A Math. Phys. Eng. Sci. **476**(2243), 20200447 (2020)
20. Mislove, A., Viswanath, B., Gummadi, K.P., Druschel, P.: You are who you know: inferring user profiles in online social networks. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp. 251–260 (2010)
21. Park, J., Barabási, A.-L.: Distribution of node characteristics in complex networks. Proc. Nat. Acad. Sci **104**(46), 17916–17920 (2007)
22. Shrum, W., Cheek, Jr., N.H., MacD, S.: Friendship in school: gender and racial homophily. In: Sociology of Education, pp. 227–239 (1988)
23. Wagner, C., Singer, P., Karimi, F., Pfeffer, J., Strohmaier, M.: Sampling from social networks with attributes. In: Proceedings of the 26th International Conference on World Wide Web, WWW'17, pp. 1181–1190. Republic and Canton of Geneva, CHE (2017)
24. Yang, H., Xiong, W., Zhang, X., Wang, K., Tian, M.: Penalized homophily latent space models for directed scale-free networks. PLoS One **16**(8), e0253873 (2021)

A More Powerful Heuristic for Balancing an Unbalanced Graph

Sukhamay Kundu and Amit A. Nanavati

Abstract We present a more powerful heuristic algorithm for the NP-complete problem of finding a minimum size subset of edges in an unbalanced signed graph G whose '+'/'−' labels can be flipped to balance G. Our algorithm finds a minimal flipping edge-set, starting with a given spanning tree T of G, by considering both the edges not in T and those in T because flipping a tree-edge can sometimes balance multiple fundamental unbalanced cycles at the same time. This can give a much smaller minimal flipping edge-set than the current algorithm where only the edges not in T are considered for flipping.

Keywords Balancing signed graph · Heuristic algorithm · Spanning tree

1 The Problem of Balancing an Unbalanced Graph

In a signed graph $G = (V,\ E)$, each edge $(x,\ y)$ has a '+'/'−' label (sign), denoted by $s(x,\ y)$. A political or social network can be modeled [1, 2] by a signed graph G, where the nodes represent individuals and the edges represent pairs of individuals who communicate directly with each other, with $s(x,\ y)$ = '+' indicating that x and y agree on some given issue such as voting the same way ('yes'/'no') on the issue, and $s(x,\ y)$ = '−' indicating that x and y disagree (voting differently). The signed graphs are also used in modelling intra-cellular regulatory system [3], where nodes represent proteins, metabolites, etc. and the edges represent excitation or inhibition interaction. G is called balanced if each cycle in G contains an even number of '−' edges. This is equivalent to saying [4] that we can partition $V = V_1 \cup V_2$ into disjoint non-empty subsets V_1 and V_2 (assuming that G has at least one '−' edge) such that each '+' edge connects two nodes in the same V_i and each '−' edge connects two

S. Kundu
Louisiana State University, Baton Rouge, LA 70803, USA

A. A. Nanavati (✉)
Ahmedabad University, Ahmedabad 380009, India
e-mail: amit.nanavati@ahduni.edu.in

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_3

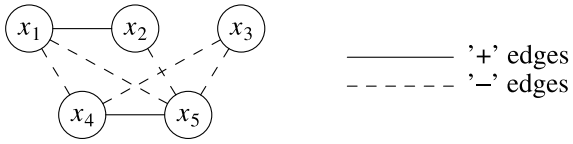


Fig. 1 A connected balanced signed graph $G = (V,\ E)$; the '−' edges form a bipartite graph, with the parts $V_1 = \{x_1,\ x_2,\ x_3\}$ and $V_2 = \{x_4,\ x_5\}$ forming a partition of V

nodes in different V_i's. For example, V_1 = the people who voted 'yes' and V_2 = the people who voted 'no'; see Fig. 1. Clearly, G is balanced if each of its connected components is balanced. The partition $V = V_1 \cup V_2$ is unique if G is connected. If G is unbalanced, then there are two ways to balance it: (1) by flipping the signs of a subset of the edges E, and (2) by deleting a subset of the edges E. It is known [5] that the optimal (minimum size) flipping edge-sets $E_{optFlip}$ whose labels can be flipped to balance G are the same as the optimal (minimum size) deletion edge-sets E_{optDel} whose deletion makes G balanced. If each edge in G is a '−' edge, then finding an E_{optDel} is the same as finding a maximum size bipartite subgraph of G. Because the latter problem is known to be NP-complete, finding an $E_{optFlip}$ for a general signed graph G is also NP-complete.

In [6], bounds on $|E_{optFlip}|$ are derived in terms of $|V|$ and $|E|$. In [7], a related NP-hard problem is considered where one wants to find a maximum size node set $V' \subseteq V$ such that the induced subgraph of G on V' is balanced. An $O(2^k|E|^2)$ algorithm for balancing G is given in [8], where $k = |E_{optFlip}|$. A heuristic algorithm based on signed spectral theory and perturbations of the graph Laplacian is given in [9]. The problem of balancing G as much as possible by deleting up to b edges is considered in [10].

2 Preliminaries

If $s(x,\ y)$ = '+' (short for '+1'), we call $(x,\ y)$ a p-edge; likewise, if $s(x,\ y)$ = '−' (short for '−1'), we call $(x,\ y)$ an n-edge. For a cycle $\xi = \langle x_1,\ x_2,\ \ldots,\ x_m,\ x_1\rangle$ of length $m \geq 3$, we write $s(\xi) = \prod_{j\leq m} s(x_j,\ x_{j+1})$, where $x_{m+1} = x_1$. We call ξ balanced if $s(\xi)$ = '+', i.e., #(n-edges in ξ) is even; otherwise, we call ξ unbalanced. The cycle ξ is called simple if the nodes $x_1, x_2, \ldots,$ and x_m are distinct. If ξ is not simple and unbalanced, then there is a simple unbalanced cycle ξ' whose edges are a subset of the edges of ξ. Henceforth, by a cycle we will mean a simple cycle. Thus, G is balanced if and only if every (simple) cycle in G is balanced. Because each cycle in G is contained in a bicomponent of G, it follows that G is balanced if and only if each of its bicomponents is balanced. Henceforth, we assume G is connected and $|V| \geq 3$.

Given a signed (connected) graph G and a spanning tree T in G, we can assign a '+'/'−' sign (label) $s(x)$ to each node x as follows. Choose an arbitrary node, say, x_1 as the root of T and let $s(x_1)$ = '+'. For each node $x_i \neq x_1$, if $\pi(x_i) = \langle x_1, x_2, \cdots, x_i\rangle$ is the unique x_1x_i-path in T, then let $s(x_i) = \prod_{j<i} s(x_j, x_{j+1})$, i.e., $s(x_i)$ = '+' if #(n-

edges in $\pi(x_i)$) is even and, otherwise, $s(x_i) =$ '$-$'. In particular, if node y is a child of node x (equivalently, $x = \text{par}(y)$, the parent of y), then $s(y) = s(x)s(x, y)$, which is the same as $s(x, y) = s(x)s(y)$. If we start with the opposite label $s(x_1) =$ '$-$' for root(T), then the new label of each x_i would be the opposite of its previous label. In this sense, we can say that the above method gives a unique '+'/'$-$' labeling of the nodes in G based on T. Henceforth, the node labels $s(x)$ will correspond to those obtained with $s(root(T)) =$ '+'. We say x is a p-node (resp., an n-node) if $s(x) =$ '+' (resp., '$-$'). Clearly, the computation of all node labels $s(x)$ takes $O(|V|)$ time. Note that if G is balanced, then the product $\prod s(x_j, x_{j+1})$ of the labels of the edges in an xy-path in G is independent of the xy-path because two xy-paths would form a cycle (which may not be simple) and that cycle is balanced. Thus, the node labels $s(x)$ are independent of T for a balanced G.

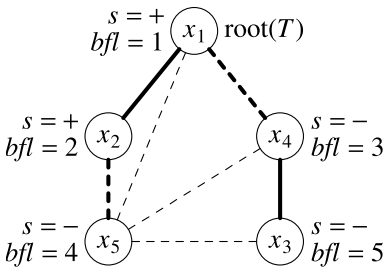
2.1 *Verifying Balancedness of G Via Node Labels s(x)*

Consider a fixed rooted spanning tree T in G. For each edge $(x, y) \in E - T$, we have a unique cycle $\xi_{x,y}$ in $T + (x, y)$; this is sometimes called the fundamental cycle of (x, y) with respect to T. If G is balanced, then each $\xi_{x,y}$ is balanced and that means $s(\xi_{x,y}) = s(x, y)s(\pi_{x,y}) = +1$, i.e., $s(x, y) = s(\pi_{x,y})$, where $\pi_{x,y}$ is the unique xy-path in T. If z is the nearest common ancestor in T of x and y, then $\pi_{x,y} = \pi_{x,z}\pi_{z,y}$, the concatenation of the xz-path $\pi_{x,z}$ in T and the zy-path $\pi_{z,y}$ in T. If $z = x$, say, then $\pi_{z,x}$ is taken to be empty-path (with no edges) and $s(\pi_{z,x}) = +1$. We have $s(x)s(y) = s(\pi(z))s(\pi_{z,x})s(\pi(z))s(\pi_{z,y}) = s(\pi_{x,y})$. This gives Eq. (1) below to test the balancedness of the fundamental cycle $\xi_{x,y}$ for an edge (x, y) not in T. Recall that the Eq. (1) holds when $(x, y) \in T$.

$$\xi_{x,y} \text{ is balanced: } s(x, y) = s(x)s(y). \tag{1}$$

2.2 *An Efficient Version of the Algorithm in [11] Based on Eq. (1)*

The algorithm MinimalFlipSetOfNonTreeEdges below is a more efficient version of the algorithm in [11] based on Eq. (1) by a factor of $O(|V|)$. It obtains the same minimal flipping edge-set $E_{malFlip} \subseteq E - T$ for balancing G for a given spanning tree T of G. The size of $E_{malFlip}$ obtained, which depends on T, can be much larger than that of an optimal flipping edge-set $E_{optFlip}$, in general. For each edge $(x, y) \in E - T$, determining whether $\xi_{x,y}$ is balanced or not based on Eq. (1) takes $O(1)$ time compared to $O(|V|)$ time in [11]. We use the breadth-first-label of a node bfl(x) = $k \geq 1$, if x is the kth node visited in a breadth-first traversal of T starting at root(T), to avoid adding an edge to $E_{malFlip}$ more than once. Figure 2 illustrates our



adj(x_1): $\langle(x_2,+),(x_4,-),(x_5,-)\rangle$
adj(x_2): $\langle(x_1,+),(x_5,-)\rangle$
adj(x_3): $\langle(x_4,+),(x_5,-)\rangle$
adj(x_4): $\langle(x_1,-),(x_3,+),(x_5,-)\rangle$
adj(x_5): $\langle(x_1,-),(x_2,-),(x_3,-),(x_4,-)\rangle$

Shown next to each node x are $s(x)$ and $bfl(x)$; the nodes are processed in the order: x_1, x_2, x_4, x_5, and x_3.

Fig. 2 Illustration of the algorithm MinimalFlipSetOfNonTreeEdges, giving $E_{malFlip} = \{(x_5, x_4), (x_3, x_5)\}$. The thick lines show the spanning tree T

algorithm. It gives $E_{malFlip} = \{(x_5, x_4), (x_3, x_5)\}$, which is also an optimal flipping edge-set. We add (x_5, x_4) first to $E_{malFlip}$ in processing x_5 because $\text{bfl}(x_5) = 4 > 3 = \text{bfl}(x_4)$; (x_3, x_5) is added next in processing x_3. Here, an alternate $E_{optFlip}$ is $\{(x_3, x_4), (x_4, x_5)\} \nsubseteq E - T$. It is shown in [5] that the removal of the edges in an $E_{malFlip}$ from G leaves G connected. This means given a particular $E_{malFlip} = E'$ and a spanning tree T of G with edges in $E - E'$, the algorithm in [11] or its modified form given here will give the minimal flipping edge-set E'. The following theorem is straightforward.

Theorem 1 *The algorithm MinimalFlipSetOfNonTreeEdges takes $O(|E|)$ time.*

Algorithm MinimalFlipSetOfNonTreeEdges (an efficient version of the algorithm in [2] based on eqn. (1)):

Input: A connected signed graph $G = (V, E)$, with $|V| \geq 3$, and a rooted spanning tree T of G. For each node x, assume par(x) = parent of x in T, with par(root(T)) = root(T), and adj(x) is a list of the pairs (y, $s(x, y)$) for nodes y adjacent to x in G.

Output: A minimal flipping edge set $E_{malFlip} \subseteq E - T$ to balance G.

1. Let s(root(T)) = '+', the breadth-first-label bfl(root(T)) = lastBFLassigned = 1, and $E_{malFlip}$ = empty-set.
2. Do a breadth-first traversal of T starting at root(T), and do one of the following for each node y in adj(x), where x = current node and $y \neq$ par(x):
   (a) [par(y) = x, hence y is not visited before and neither of $s(y)$ and bfl(y) is defined.] If (par(y) = x) then add 1 to lastBFLassigned and let bfl(y) = lastBFLassigned and $s(y) = s(x)s(x, y)$.
   (b) [both $s(y)$ and bfl(y) are defined; y may be an ancestor of x or a descendant of an ancestor of x in T but not a descendent of x, meaning all nodes in the fundamental cycle $\xi_{x,y}$ in $T + (x, y)$ have been visited.] If ($s(y)$ is defined, bfl(y) < bfl(x), and $\xi_{x,y}$ is not balanced, i.e., $s(x)s(y) \neq s(x, y)$), then add (x, y) to $E_{malFlip}$.

3 Flipping Edges in T to Balance G

For a given spanning tree T of G, flipping the label of an edge $(x,\ y) \in E - T$ changes the balancedness of only $\xi_{x,y}$ and has no effect on other fundamental cycles. However, flipping the label of $(u,\ v) \in T$ changes the balancedness of all fundamental cycles covered by $(u,\ v)$, i.e., $\xi_{cov}(u,\ v) = \{\xi_{x,y}\colon (x,\ y) \in E - T$ and $(u,\ v)$ is in the xy-path in T or, equivalently, $(u,\ v)$ is in $\xi_{x,y}\}$. If $\xi_{x,y} \in \xi_{cov}(u, v)$ is balanced, then flipping the label of $(u,\ v)$ makes $\xi_{x,y}$ unbalanced and we need to rebalance $\xi_{x,y}$ by flipping the label of some other edge in T that covers $\xi_{x,y}$ or by flipping the label of $(x,\ y)$ itself. We can sometimes obtain a smaller $E_{malFlip}$ when we use a combination of edges in T and edges in $E - T$ than that when we limit ourselves to use only the edges in $E - T$ as is done in [11]. For the signed graph $G = (V,\ E)$ and its spanning tree T shown in Fig. 3, which illustrates the notion of "cover", each edge $E - T$ is a p-edge and there are two $E_{optFlip} = \{(x_1, x_2), (x_5, x_6)\}$ and $\{(x_2, x_3), (x_6, x_7)\}$; they both have size 2 and consist of edges in T. Here, $E_{malFlip}$ based on T consists of 6 out of 7 edges in $E - T$, excluding the edge $(x_1,\ x_7)$.

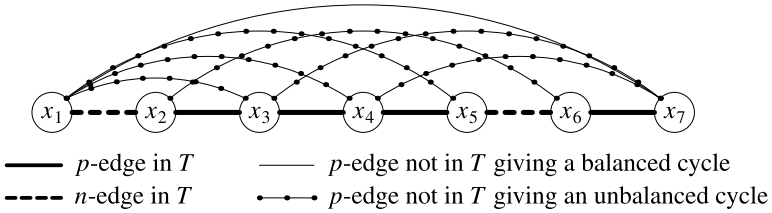
3.1 Selection Criteria for a Tree Edge $(u,\ v)$ for Flipping

Let T be a spanning tree of G, $(u,\ v)$ an edge in T, and $(x,\ y)$ and edge in $E - T$. Let $\xi_{bcov}(u,\ v) = \{\xi_{x,y} : \xi_{x,y}$ is balanced and covered by $(u,\ v)\}$ and $\xi_{ucov}(u,\ v) = \{\xi_{x,y} : \xi_{x,y}$ is unbalanced and covered by $(u,\ v)\}$. Clearly, $\xi_{cov}(u,\ v) = \xi_{bcov}(u,\ v) \cup \xi_{ucov}(u,\ v)$, a disjoint union. If we flip the label of $(u,\ v)$, then to rebalance each $\xi_{x,y} \in \xi_{bcov}(u,\ v)$ we can flip the label of $(x,\ y)$. Thus, flipping the label of $(u,\ v) \in T$ to balance G is better than flipping the labels of all edges in $\{(x,\ y) : \xi_{x,y} \in \xi_{ucov}(u,\ v)\}$ when inequality in Eq. (2) is strict.

$$\text{Condition for flipping } (u,\ v) \in T\colon |\xi_{bcov}(u,\ v)| \leq |\xi_{ucov}(u,\ v)| + 1 \qquad (2)$$

3.2 The New More Powerful Heuristic Algorithm

The algorithm MinimalFlipSet below repeatedly selects an edge $(u,\ v)$ in the given spanning tree T based on Eq. (2) to include in $E_{malFlip}$. When we cannot find such an $(u,\ v) \in T$, we choose the remaining edges $(x,\ y) \in E - T$ such that $\xi_{x,y}$ is still unbalanced. We need to select tree-edges before selecting non-tree edges partly because flipping the label of a selected $(x,\ y) \in E - T$ cannot help the selection of an $(u,\ v) \in T$ and partly because if we start selecting non-tree edges first and at some arbitrary point start selecting tree-edges, then we may still need to switch back to selecting non-tree edges again to complete balancing G. Note that if we let the loop in step 2 iterate any number of times (≥ 0) instead of terminating it when $M < 1$, then we can generate many alternative $E_{malFlip}$. The choice of $(u,\ v)$ in step 2(b) also lets us generate alternative $E_{malFlip}$.



(i) A signed graph G and a spanning tree T of it, whose edges are shown in bold lines.

Some edges $(u, v) \in T$	The edges $(x, y) \notin T$ such that the fundamental cycle $\xi_{x,y}$ is covered by (u, v)
(x_1, x_2)	(x_1, x_3), (x_1, x_4), (x_1, x_5), (x_1, x_7)
(x_2, x_3)	(x_1, x_3), (x_1, x_4), (x_1, x_5), (x_1, x_7), (x_2, x_6)
(x_5, x_6)	(x_1, x_7), (x_2, x_6), (x_3, x_7), (x_4, x_7)
(x_6, x_7)	(x_1, x_7), (x_3, x_7), (x_4, x_7)

(ii) The fundamental cycles $\xi_{x,y}$ covered by some edges $(u, v) \in T$.

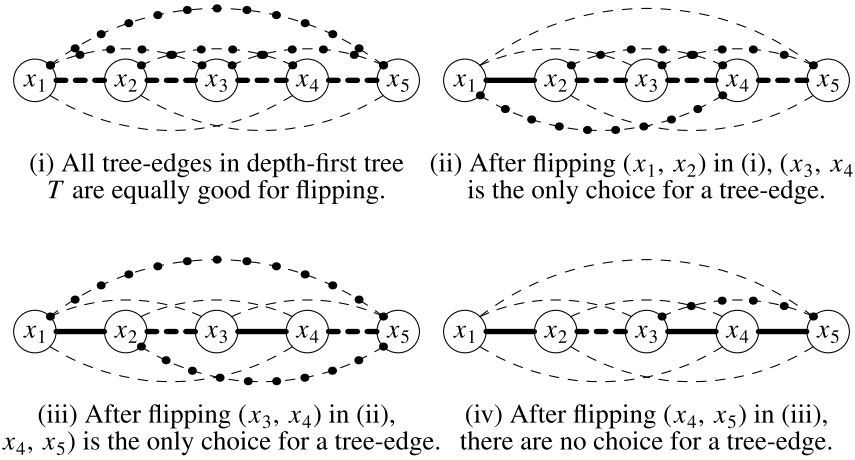
Fig. 3 Illustration of $\xi_{cov}(u, v)$ for $(u, v) \in T$

Algorithm MinimalFlipSet:

Input: A connected signed graph $G = (V, E)$ and a spanning tree T of G.
Output: A minimal flipping edge-set $E_{malFlip}$ consisting of possibly edges from both T and $E - T$ for balancing G.

1. Initialize $E_{malFlip}$ = empty-set.
2. Repeat steps (a)-(c):
   (a) For each edge $(u, v) \in T$, determine $\xi_{bcov}(u, v)$ and $\xi_{ucov}(u, v)$.
   (b) Determine $M = \max \{|\xi_{ucov}(u, v)| - |\xi_{bcov}(u, v)|: (u, v) \in T\}$ and an (u, v) that gives M. If (u, v) is not unique, choose one arbitrarily.
   (c) If $(M \geq 1)$, then flip the label of the edge (u, v) and add (u, v) to $E_{malFlip}$.
   while $(M \geq 1)$.
3. For each remaining unbalanced $\xi_{x,y}$, $(x, y) \notin T$, if any, add (x, y) to $E_{malFlip}$.

Example 1 Figure 4 illustrates algorithm MinimalFlipSet for $G = K_5^-$, the complete graph on 5 nodes with all n-edges. Figure 4i shows the depth-first tree T of G for root$(T) = x_1$. Figure 4ii–iv shows a sequence of choices of tree-edges for flipping and the results of flipping them. Finally, we flip $(x_3, x_5) \in E - T$ to balance G. The resulting $E_{malFlip}$ is also an $E_{optFlip}$. There are many other possible sequences of choice of tree-edges in our algorithm here; if we choose (x_2, x_3) and (x_3, x_4) in T in that order, then we must choose (x_1, x_5) and (x_2, x_4) in $E - T$ to balance G, again giving an $E_{optFlip}$. The algorithm in [11] here also give an $E_{optFlip}$. The situation is



(i) All tree-edges in depth-first tree T are equally good for flipping.

(ii) After flipping (x_1, x_2) in (i), (x_3, x_4) is the only choice for a tree-edge.

(iii) After flipping (x_3, x_4) in (ii), (x_4, x_5) is the only choice for a tree-edge.

(iv) After flipping (x_4, x_5) in (iii), there are no choice for a tree-edge.

Fig. 4 Illustration of MinimalFlipSet for $G = K_5^-$

quite different if we choose T with root$(T) = x_1$ having 4 children $\{x_2, x_3, x_4, x_5\}$. In this case, each $E_{malFlip}$ determined by MinimalFlipSet is an $E_{optFlip}$, involving 2 edges in T and 2 edges in $E - T$ but the algorithm in [11] gives an $E_{malFlip}$ of 6 edges in $E - T$, which is not an $E_{optFlip}$.

Theorem 2 *Given any spanning tree T in a connected signed graph G, we can compute all minimal flipping edge-sets $E_{malFlip}$, including all optimal flipping edge-sets $E_{optFlip}$, using different choices of edges in T and edges in $E - T$.*

4 An Efficient Implementation of MinimalFlipSet

We give below an efficient implementation of the algorithm MinimalFlipSet when T is a depth-first spanning tree of G. (The case when T is not a depth-first spanning tree of G will be discussed elsewhere.) Each edge $(x, y) \in E - T$ is now a back-edge, with one of x and y being an ancestor of the other but not the parent. We say a back-edge (x, y) is balanced (resp., unbalanced) if $\xi_{x,y}$ is balanced (resp., unbalanced). Let $dfl(x)$ be the depth-first label of x and $T(x)$ the subtree of T at x (including node x).

We define several counts related to balanced and unbalanced back-edges and their cumulative forms as shown in Table 1 for efficient computation of $|\xi_{bcov}(u, v)|$ and $|\xi_{ucov}(u, v)|$ used in Eq. (2) for all edges $(u, v) \in T$. Note that for a terminal node x in T, we have $cbt(x) = cut(x) = 0$ and for $x = \text{root}(T)$ we have $cbf(x) = cuf(x) = 0$, $ccbf(x) = ccbt(x) =$ #(balanced back-edges in T) and $ccuf(x) = ccut(x) =$ #(unbalanced back-edges in T). All these counts at a node x are determined when we backtrack from x in the depth-first traversal of G. Clearly, $cbf(x) + cuf(x) =$ #(back-edges from x) and $cbt(x) + cut(x) =$ #(back-edges to x). Table 2 gives

Table 1 Definitions of various basic counts related to back-edges

$cbf(x)$ = #(balanced back-edges from node x to its ancestors)
$cbt(x)$ = #(balanced back-edges to node x from its descendents)
$cuf(x)$ = #(unbalanced back-edges from node x to its ancestors)
$cut(x)$ = #(unbalanced back-edges to node x from its descendents)
$ccbf(x)$ = #(balanced back-edges from nodes in $T(x)) = \Sigma_{y \in T(x)} cbf(y)$
$ccbt(x)$ = #(balanced back-edges to nodes in $T(x)) = \Sigma_{y \in T(x)} cbt(y)$
$ccuf(x)$ = #(unbalanced back-edges from nodes in $T(x)) = \Sigma_{y \in T(x)} cuf(y)$
$ccut(x)$ = #(unbalanced back-edges to nodes in $T(x)) = \Sigma_{y \in T(x)} cut(y)$

Table 2 Definitions of counts related to back-edges covered by a tree-edge $(x, \text{par}(x))$

$cbcov(x)$	= #(balanced back-edges covered by the tree-edge $(x, \text{par}(x))$
	= #(balanced back-edges from nodes in T(x) to ancestors of x)
	$= \Sigma_{y \in T(x)}(cbf(y) - cbt(y)) = ccbf(x) - ccbt(x)$
$cucov(x)$	= #(unbalanced back-edges covered by the tree-edge $(x, \text{par}(x))$
	= #(unbalanced back-edges from nodes in T(x) to ancestors of x)
	$= \Sigma_{y \in T(x)}(cuf(y) - cut(y)) = ccuf(x) - ccut(x)$.

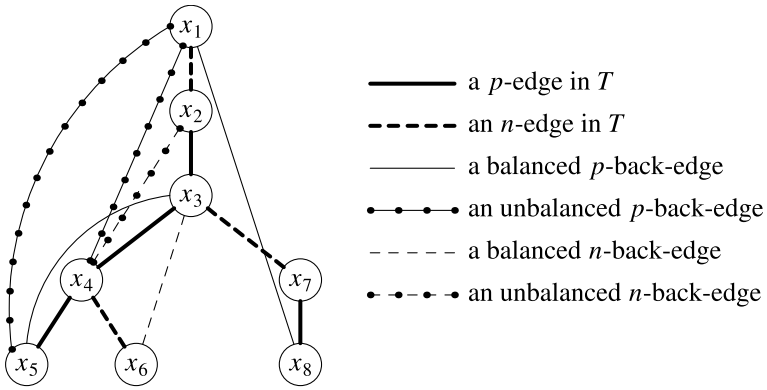


Fig. 5 Illustration of the balanced and unbalanced back-edges for a depth-first tree T in a signed graph G

the related counts of the fundamental cycles covered by a tree edge $(x, \text{par}(x))$; for example, $cbcov(u) = |\xi_{bcov}(u,\ par(u))|$.

Figure 5 shows a connected unbalanced signed graph G, a depth-first spanning tree T of G, the nodes labels $s(x_i)$ based on T, and the resulting balanced and unbalanced back-edges. Table 3 shows the various counts for the nodes in Fig. 5.

Table 3 Illustration of various counts at some of the nodes for G and T in Fig. 5; nodes are listed in the order they are back-tracked in the depth-first traversal

Node x_i	$cbf(x_i)$ $cbt(x_i)$,	$cuf(x_i)$, $cut(x_i)$	$ccbf(x_i)$, $ccbt(x_i)$	$ccuf(x_i)$, $ccut(x_i)$	$cbcov(x_i)$ and its associated edges	$cucov(x_i)$ and its associated edges
x_5	1, 0	1, 0	1, 0	1, 0	1 − 0=1 (x_5, x_3)	1 − 0 =1 (x_5, x_1)
x_6	1, 0	0, 0	1, 0	0, 0	1 − 0=1 (x_6, x_3)	0 − 0 = 0 –
x_4	0, 0	2, 0	2, 0	3, 0	2 − 0=2 (x_5, x_3), (x_6, x_3)	3 − 0 =3 (x_5, x_1), (x_4, x_1), (x_4, x_2)
...	...	...	...	...	...	...
x_1	0, 1	0, 2	3, 3	3, 3	3 − 3=0 –	3 − 3 = 0 –

4.1 Algorithm to Determine the Counts in Tables 1 and 2

The algorithm ModifiedDFSforVariousCounts below is a modified depth-first traversal of G to determine the various counts in Tables 1 and 2. The total of the counts $cbf(x)$, $cbt(x)$, $cuf(x)$, and $cut(x)$ for all nodes x equals $2(|E| - |V| + 1)$ because a back-edge $(x,\ y)$ contributes 1 to each of the counts $cbf(x)$ and $cbt(y)$ or to each of the counts $cuf(x)$ and $cut(y)$. The computation of the corresponding cumulative counts in Table 1 at a node x is done by combining the counts at x with with the cumulative counts at its children. Thus, their computation takes $O(|V|)$ time. Thus, the total computation time of the algorithm is $O(|E|)$. For each of the counts $cbf(x)$ and $cuf(x)$, we can also compute the associated list of nodes for the other end point of the back-edges $bbef(x) = \{y$: back-edge $(x,\ y)$ is balanced$\}$, and similarly for $ubef(x)$ that contribute to those counts. We do not compute similar list of nodes related to $cbt(x)$ and $cut(x)$ for reasons explained later. The computation of these lists require only additional $O(|E| - |V| + 1)$ time.

4.2 Selecting a Best Flipping Tree-Edge in a Depth-First Tree

A best flipping edge in a dept-first spanning tree T for balancing G is $(u,\ v)$, $v = par(u)$ and $u \neq$ root(T), is one that maximizes $cucov(u) - cbcov(u)$ is maximum and the maximum is > 1, based on Eq. (2). If there is more than one such u, then select any one arbitrarily.

For the selection of the next best tree-edge in T, we need to determine the back-edges $(x,\ y)$ covered by the selected tree-edge $(u,$ par$(u))$. For this, we form the union of the node-lists $bbef(x)$ and $ubef(x)$ for nodes $x \in T(u)$ and from this we subtract the nodes y with $dfl(y) \leq dfl(u)$. This can be done in $O(|E| - |V| + 1)$ computation time if we use depth-first labels of nodes in the lists $bbef(x)$ and $ubef(x)$ throughout instead of the original node names. As we flip the labels of the back-edges $(x,\ y)$ covered by the selected tree-edge $(u,$ par$(u))$, we update the counts related to $cbf(x)$ and $cbt(y)$ or $cuf(x)$ and $cut(y)$ depending on whether

(x, y) was balanced or not; we can also update the associated node-lists $bbef(x)$ or $ubef(x)$ accordingly. We also need to flip the sign of the selected tree-edge $(u, \text{par}(u))$ and flip the node labels of each node y in the subtree $T(u)$, including u. This takes $O(|V|)$ time. We are now ready to update the various cumulative counts in Table 1 for all nodes in $T(u)$ and the ancestors of u and then proceed to select the next best flipping edge in T, if any.

The process terminates because #(tree-edges selected) + #(unbalanced fundamental cycles with respect to T) keeps decreasing. In particular, it will not happen that we flip a tree edge $(u, \text{par}(u))$ at one point and then later we flip the same tree edge again. Note that there is no change in T itself in the whole process although the labels of some of its edges and nodes may change. Once a tree-edge selection provides no advantage, we make one more pass to select edges in $E - T$ for the remaining unbalanced fundamental cycles. The following theorem is now straightforward.

Algorithm ModifiedDFSforVariousCounts:

Input: A connected signed graph $G = (V, E)$, a start-node, and the lists adj(x) of the pairs $(y, s(x, y))$ for nodes y adjacent to x.

Output: A depth-first tree T with root(T) = start-node and for each node x its depth-first label dfl(x), '+'/'−' label $s(x)$, the counts in Table 1, and the node-lists bbef(x) and ubef(x) related to cbf(x) and cuf(x).

1. For each node x, initialize dfl(x) and each of the counts in Table 1 for node x to 0, and initialize the lists bbef(x) and ubef(x) to empty-list. Also, let root(T) = start-node, par(start-node) = start-node, s(root(T)) = '+', lastDflAssigned = dfl(root(T)) = 1, and the current node x = root(T).
2. Let y be the next node in adj(x) to visit, if any.
3. If (there is no y), then do one of the following to backtrack from x to par(x):
   (a) If (x = root(T)), then stop.
   (b) Otherwise, add ccbf(x) to ccbf(par(x)) and, likewise, for ccbt(x), ccuf(x), ccut(x), and also let cbcov(x) = ccbf(x) − ccbt(x) and cucov(x) = ccuf(x) − ccut(x) and then let the current node x = par(x).
4. Otherwise, i.e., if (y exists), then do one of the following for the first visit of either a depth-first tree-edge from x to a child y of x or a back-edge (x, y):
   (a) If (dfl(y) = 0), then add 1 to lastDflAssigned and let dfl(y) = lastDflAssigned, $s(y) = s(x)s(x, y)$, par$(y) = x$, and the current node $x = y$.
   (b) If (dfl$(y) > 0$ and $y \neq$ par(x), i.e., dfl$(y) <$ dfl(par(x))), then do one of the following for the back-edge (x, y):
      (b.1) If $((x, y)$ is balanced, i.e., $s(x)s(y) = s(x, y)$), then add y to bbef(x) and add 1 to each of cbf(x), cbt(y), ccbf(x), and ccbt(y).
      (b.2) Otherwise, add y to ubef(x) and add 1 to each of cuf(x), cut(y), ccuf(x), and ccut(y).

Theorem 3 *For a depth-first spanning tree T in G, we can compute a minimal flipping edge-set $E_{malFlip}$ combining edges in T and edges in $E - T$ in time $O((k + 1)|E|)$, where k = #(edges in T selected in $E_{malFlip}$).*

5 Conclusion

We first provide an $O(|E|)$ time heuristic algorithm, which is more efficient than the heuristic algorithm in [11] by a factor of $|V|$, to find a minimal flipping edge set $E_{malFlip}$ to balance a connected unbalanced signed graph $G = (V, E)$ based on a spanning tree T of G. Here, we use the concept of $\pm$-labeling $s(x)$ of the nodes $x \in V$ based on T and the fact that the fundamental cycle $\xi(x, y)$ in $T + (x, y)$ for an edge $(x, y) \notin T$ is balanced if and only if $s(x, y) = s(x)s(y)$. As in [11], we flip the label of $(x, y) \notin T$ if and only if $\xi(x, y)$ is not balanced; it balances only the fundamental cycle $\xi(x, y)$. We then provide a more powerful heuristic to find a $E_{malFlip}$ than in [11]. The key idea in our new heuristic algorithm is that if several $\xi(x, y)$ share a common edge $(u, v) \in T$, then flipping the label of (u, v) balances all those $\xi(x, y)$'s. Because it will also unbalance the balanced fundamental cycles that share (u, v), our algorithm uses a specific criteria to select the edges $(u, v) \in T$, in addition to some edges $(x, y) \notin T$, to balance G and it typically obtains a smaller minimal flipping edge-set for a given T than [11]. Our algorithm has, however, a higher time complexity as shown in Theorem 3.

References

1. Aref, S., Neal, Z.: Detecting coalitions by optimally partitioning signed networks of political collaboration. Sci. Rep. **10**(1), 1–10 (2020)
2. Cartwright, D., Harary, F.: Structural balance: a generalization of Heider's theory. Psychol. Rev. **63**(5), 277 (1956)
3. Ma'ayan, A., Lipshtat, A., Iyengar, R., Sontag, E.D.: Proximity of intracellular regulatory networks to monotone systems. IET Syst. Biol. **2**(3), 103–112 (2008)
4. Harary, F.: On the notion of balance of a signed graph. Michigan Math. J. **2**(2), 143–146 (1953)
5. Harary, F.: On the measurement of structural balance. Behavioral Sci. **4**(4), 316–323 (1959)
6. Akiyama, J., Avis, D., Chvátal, V., Era, H.: Balancing signed graphs. Discrete Appl. Math. **3**(4), 227–233 (1981)
7. Figueiredo, R., Frota, Y.: The maximum balanced subgraph of a signed graph: applications and solution approaches. Euro. J. Oper. Res. **236**(2), 473–487 (2014)
8. Hüffner, F., Betzler, N., Niedermeier, R.: Optimal edge deletions for signed graph balancing. In: International Workshop on Experimental and Efficient Algorithms, pp. 297–310. Springer (2007)
9. Ordozgoiti, B., Matakos, A., Gionis, A.: Finding large balanced subgraphs in signed networks. In: Proceedings of the Web Conference 2020, pp. 1378–1388 (2020)

10. Sharma, K., Gillani, I.A., Medya, S., Ranu, S., Bagchi, A.: Balance maximization in signed networks via edge deletions. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining, pp. 752–760 (2021)
11. Alabandi, G., Tešić, J., Rusnak, L., Burtscher, M.: Discovering and balancing fundamental cycles in large signed graphs. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1–17 (2021)

DC-RST: A Parallel Algorithm for Random Spanning Trees in Network Analytics

Lucas Henke and Dinesh Mehta

Abstract The Mantel Test, discovered in the 1960s, determines whether two distance metrics on a graph are related. We describe DC-RST, an algorithm to accelerate a key step of a network science statistical computation associated with DIMECOST, an approach that is faster the Mantel Test. DC-RST is a parallel, divide-and-conquer algorithm to compute a random spanning tree of a complete graph on n vertices. Relative to an implementation of Wilson's sequential random-walk algorithm, on a system with 48 cores, DC-RST was up to 4X faster when first creating random partitions and up to 20X faster without this sub-step. DC-RST is shown to be a suitable replacement for Wilson's sequential algorithm through a combination of theoretical and statistical results.

Keywords Wilson's algorithm · Mantel test · Dimecost

1 Introduction and Background

The Mantel Test [1] is a statistical test for characterizing the relation between two distance metrics. For example, in a road network, one would expect drive times to be closely related to driving distances. The Mantel Test provides a quantitative way for doing so—it entails computing Pearson's correlation coefficient on an $n \times n$ matrix for a number of permutations (denoted by t), making the overall computation $\Theta(t \cdot n^2)$. This is prohibitive for Big Data applications (social networks, the internet web graph, etc.), where $n \sim 10^9$. Bourbour et al. recently presented DIMECOST, similar to the Mantel Test, that utilizes uniform random spanning trees. DIMECOST has an improved complexity of $\Theta(t \cdot n)$ [2].

L. Henke · D. Mehta (✉)
Department of Computer Science, Colorado School of Mines, Golden, USA
e-mail: dmehta@mines.edu

L. Henke
e-mail: lhenke@mines.edu

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_4

$$D_1 = \begin{pmatrix} 0 & 96 & 220 & 322 & 56 \\ & 0 & 188 & 266 & 43 \\ & & 0 & 157 & 184 \\ & & & 0 & 308 \\ & & & & 0 \end{pmatrix} \qquad D_2 = \begin{pmatrix} 0 & 129 & 247 & 367 & 67 \\ & 0 & 215 & 278 & 53 \\ & & 0 & 190 & 220 \\ & & & 0 & 325 \\ & & & & 0 \end{pmatrix}$$

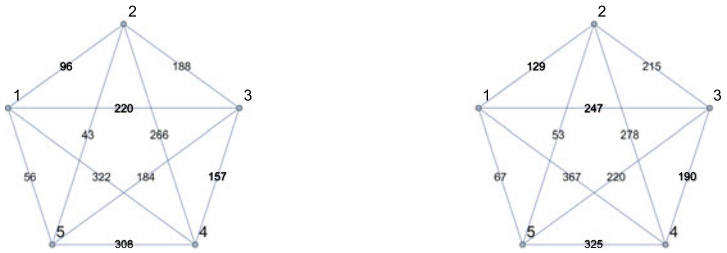


Fig. 1 Distance matrices D_1 and D_2 with corresponding graphs G_1 and G_2. Since both matrices are symmetric, only their upper triangles are drawn

This paper goes one step further: a key step in DIMECOST is the computation of a uniform random spanning tree of a complete graph using Wilson's algorithm; we aim to perform this step in parallel. While this problem has been studied before, this was only done theoretically [3]. Our proposed algorithm, DC-RST (Divide and Conquer—Random Spanning Tree), has been analyzed theoretically and implemented experimentally, and is shown to be a suitable replacement for this step.

Definitions Consider a set of n objects and two distance metrics d_1 and d_2, which can be used to compute pairwise distances between all pairs of objects[1]. A distance metric can be interpreted as a *symmetric matrix*: an $n \times n$ matrix, say D_1, where $D_1[i, j]$ is the distance between i and j using the d_1 metric. It may also be viewed as a *weighted network*: a weighted, undirected, complete graph, G_1, where each vertex corresponds to an object, and an edge from v_i to v_j has weight equal to the distance between i and j using the distance metric d_1. Both representations are equivalent: matrix D_1 can be viewed as the weighted adjacency matrix for G_1 as illustrated in Fig. 1.

Mantel Test The Mantel test answers the question: is there a statistically significant relationship between distance metrics d_1 and d_2? In the road network example, we expect drive times to be closely related to driving distances, and the Mantel test provides a quantitative method for measuring this relationship. Specifically, the Mantel test takes as input the matrices D_1 and D_2 and:

1. Computes the Pearson's correlation coefficient (r_{init}) between D_1 and D_2.
2. Randomly permutes the rows and columns of D_1 to obtain D_1'.
3. Compute r between D_1' and D_2.

[1] Note: distance metrics are assumed to be symmetric.

Steps 2 and 3 are run t times and the number of times x that $r > r_{init}$ is recorded. If $x/t \leq p$ (e.g., p could be 0.05), the test asserts that metrics d_1 and d_2 are related [1].

The Mantel Test (and its variants) have been a widely used tool for distance metric analyses (e.g., a package for the statistical language R implements the Mantel Test [4]). Schneider and Borlund review its use in anthropology, psychology, and geography [5]; see also [6–8]. Ricaut et al. used it to determine whether there is a correlation between genetic and discrete trait proximity matrices for individuals in the Egyin Gol necropolis in Mongolia [9]. Smouse et al. describes the importance of the Mantel Test in biology, on distance metrics of genetic markers, morphological traits, ecological divergence, etc. [10]. Kouri et al. used it in cheminformatics, to study the relationship between bond count distances and Tanimoto distances [11].

Dimecost Bourbour et al. proposed an algorithm called DIMECOST, which uses uniform, random spanning trees instead of matrices [2]. Recall that the distance matrix is equivalent to a weighted, undirected, complete graph on n vertices. DIMECOST computes a random spanning tree of this graph. r_{init} is then computed using the edges of the spanning tree and permutations of the d_1 weights are used to perform a test similar to Mantel. Bourbour et al. show this method works better than randomly selecting edges, has a lower complexity than the original Mantel test ($\Theta(t \cdot n)$ instead of $\Theta(t \cdot n^2)$), and results in similar correlation values between distance metrics.

Random Walks A key step in DIMECOST uses random walks on the graph to obtain a uniform, random spanning tree. Given an undirected graph $G = (V, E)$, a spanning tree is any undirected, acyclic, connected sub-graph $T = (V_T, E_T)$ that spans the graph. In general, undirected graphs have *many* spanning trees; thus, a *uniform random* spanning tree is *a* spanning tree, selected at random from the possible set of *all* spanning trees (where each tree is equally-likely [uniformly] to be selected). Aldous [12] and Broder [13] gave equivalent algorithms using a random walk to generate a uniform random spanning tree by tracking the edges traversed when discovering a vertex. Their algorithm has complexity equal to the mean cover time of the graph—for cliques, this is $O(n \log n)$ [14]. Later, Wilson [15] gave a different algorithm, whose complexity is equal to the mean *hitting* time of a graph—for cliques, this is $O(n)$ [14]. DIMECOST uses Wilson's algorithm.

2 Parallel Algorithm and Analysis

2.1 Algorithm Outline

We propose the following approach to create random spanning trees.

1. Randomly partition the original clique into k sub-cliques.
2. Run Wilson's algorithm on each sub-clique, forming a forest of k spanning sub-trees.
3. "Merge" the sub-trees together to form the final tree as follows:

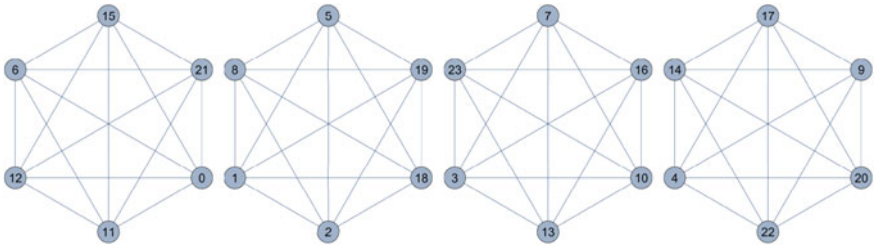


Fig. 2 Four random partitions of the original clique

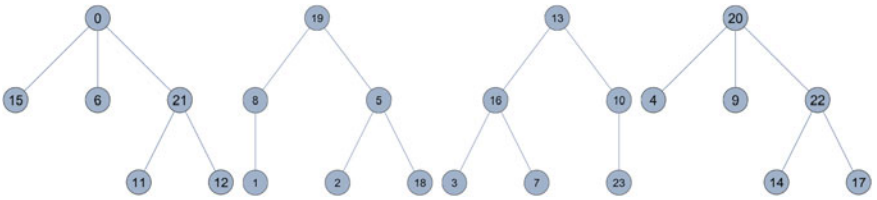


Fig. 3 Sub-trees from Step 2

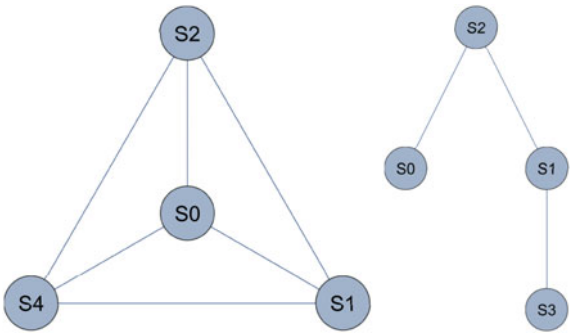


Fig. 4 Supergraph and supertree for this example

(a) Consider each sub-clique to be a "supernode", mutually connected to form a "supergraph" clique.
(b) Run Wilson's algorithm on the "supergraph" to get a "supertree".
(c) For each superedge in the supertree, obtain a "real" edge by choosing uniformly at random a vertex from both subtrees and join with an edge.

Consider a clique of 24 vertices randomly partitioned in Step 1 into four subgraphs of six vertices each, as shown in Fig. 2.

For each of the four sub-graphs, Step 2 runs four independent instances of Wilson's algorithm (one per sub-graph) in parallel resulting in a forest of spanning trees (referred to as "sub-trees") shown in Fig. 3. Recall, Wilson's algorithm generates *uniform random* spanning trees, thus this forest of spanning trees is just *one* of *many* possible forests.

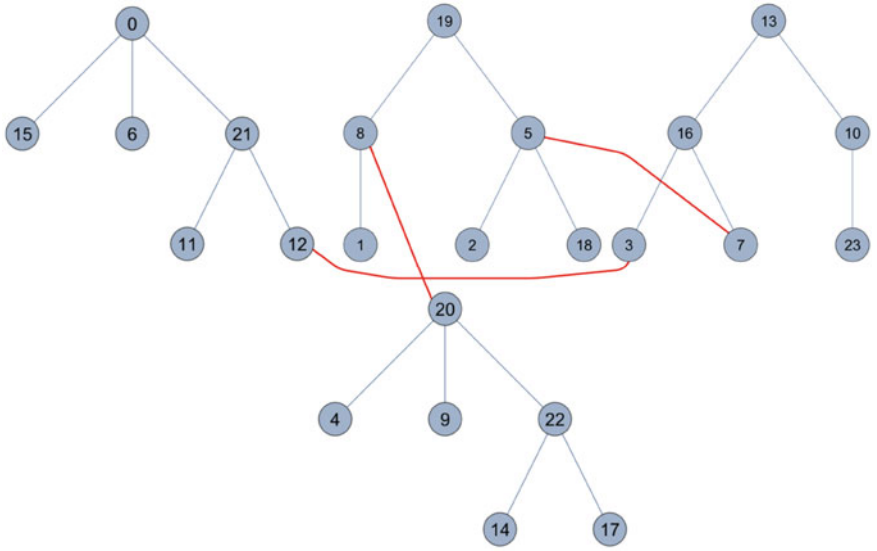


Fig. 5 Connecting the forest with edges using super tree

With these sub-trees, we now run the "Merge" step of Step 3. Step 3a creates the "supergraph", which is a clique of 4 vertices as shown in Fig. 4. Step 3b runs Wilson's algorithm on the supergraph, to form a "supertree", also shown in Fig. 4.

Step 3c converts each superedge in the supertree to a "real" edge by choosing uniformly at random a vertex from each tree and joining those two with an edge. The example supertree indicates trees 0 and 2 should be connected with an edge, thus a vertex from each tree is chosen at uniform random to be connected (vertex 12 and 3). This process, repeated for the remaining superedges, results in the final tree (new edges in red), shown in Fig. 5.

2.2 Theoretical Analysis

We first show that our algorithm meets the following necessary condition for a uniform random spanning tree. (Our theorems are stated without proof due to space limitations.)

Theorem 1 *Let $G = (V, E)$ be a complete, undirected graph (i.e., a clique), with $n = |V|$ and $m = |E|$. Let $(u, v) \in E$ be any edge in G. Let T be a spanning tree, chosen uniformly at random from all possible spanning trees of G (i.e., T is a uniform random spanning tree). Then, the probability $(u, v) \in T$ is: $P((u, v) \in T) = \frac{2}{n}$*

Theorem 2 *Let $G = (V, E)$ be a clique, let $(u, v) \in E$ be any edge in G, and let T be a spanning tree our algorithm creates. If the vertices V are partitioned uniformly at random into k groups of possibly unequal size, then the probability $(u, v) \in T$ is: $P((u, v) \in T) = \frac{2}{n}$*

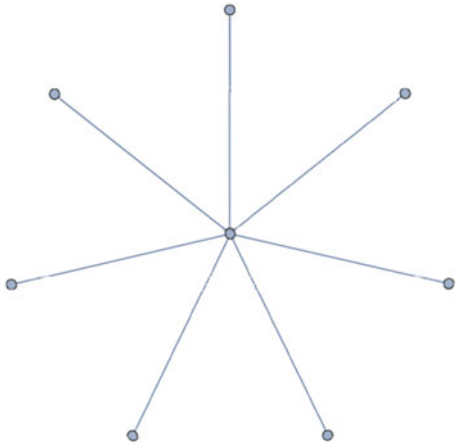


Fig. 6 The "star" tree for $n = 8$: all vertices are connected to one central vertex

However, a simple counterexample shows that DC-RST does not generate random spanning trees with uniform probability. Specifically, the "star" tree (Fig. 6) will never be generated by DC-RST unless $k = 1$ or $k = n$ (when DC-RST degenerates to Wilson's algorithm). Thus, DC-RST creates a random (but non-uniform) spanning tree of a clique such that edges are chosen with the same probability as a uniform random spanning tree.

2.3 Statistical Analysis

This raises the following question: if DC-RST does not generate uniform random spanning trees, then is DC-RST a suitable replacement for Wilson's algorithm in DimeCost? To answer this, we compared two versions of DimeCost—one using Wilson's algorithm, the other using DC-RST—on three data sets used by Bourbour et al. In each experiment, we use the Mantel Test to compute the correlation coefficient between a pair of distance matrices (r), and compute 95% confidence intervals (CIs) for DimeCost using both Wilson's algorithm and DC-RST for spanning tree generation. We then examine whether the CIs contain r.

Data Set 1: Comparison of Distance Norms For this first data set, we generated 25 (x, y) points and applied four distance matrices based on L_1, L_2, L_3, and L_∞ norms; considering these pairwise gives six data sets.[2] The results are summarized in Table 1.

The CIs using DC-RST differ slightly from those using Wilson's algorithm, but still contain r from the Mantel Test.

[2] The L_p-norm for $p \geq 1$ of a vector $\vec{x}$ is a commonly used measure of "distance" in machine learning for clustering, and is defined by $||\vec{x}||_p = (|x_1|^p + |x_2|^p + \cdots + |x_n|^p)^{\frac{1}{p}}$. Note that $L_\infty(\vec{x}) = \max\{|x_1|, |x_2|, ..., |x_n|\}$ is the limit of the L_p norm as $p \longrightarrow \infty$.

Table 1 Comparison between the Mantel test and DIMECOST on various L_p-norms

d_1	d_2	r	CI using Wilson	CI using DC-RST
L_1	L_2	0.9798	(0.9785, 0.9819)	(0.9776, 0.9806)
L_1	L_3	0.9584	(0.9550, 0.9613)	(0.9558, 0.9621)
L_1	L_∞	0.9167	(0.9096, 0.9209)	(0.9154, 0.9255)
L_2	L_3	0.9959	(0.9956, 0.9962)	(0.9955, 0.9961)
L_2	L_∞	0.9754	(0.9743, 0.9782)	(0.9751, 0.9785)
L_3	L_∞	0.9907	(0.9898, 0.9917)	(0.9901, 0.9918)

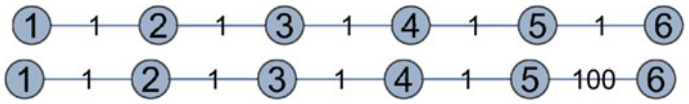


Fig. 7 Two line graphs with 6 vertices

Table 2 Comparison between the Mantel test and DIMECOST on data set 2

n	r	CI using Wilson	CI using DC-RST
6	0.400	(0.342, 0.504)	(0.314, 0.477)
10	0.350	(0.230, 0.358)	(0.232, 0.378)
30	0.422	(0.362, 0.432)	(0.387, 0.458)
50	0.596	(0.584, 0.630)	(0.570, 0.610)
100	0.872	(0.869, 0.887)	(0.866, 0.885)

Data Set 2: Line Graph For this example, consider a road network represented by a line graph, modeling a long continuous stretch of an interstate highway where each edge represents a highway segment. In the first graph, the weights of all edges (representing travel distances) are equal to 1. In the second graph (representing travel time), the weights of all edges are again 1 *except* the last edge, which possesses an unusually high weight, 100. This could represent an accident in that segment of the highway causing times, but not distances, to drastically increase. Both graphs are illustrated for the case of $n = 6$ in Fig. 7.

We then computed distance matrices for both metrics by computing the shortest path distance between each pair of vertices. Corresponding elements in the two matrices have identical values, except the last row and column, which represent edges to the last vertex. The results are summarized in Table 2.

As before, while the CIs using DC-RST differ slightly from those computed using Wilson's algorithm, they still contain the r value calculated by the Mantel Test. This test previously showed that simply choosing random edges from the complete graph does not suffice, as the CIs generated in this example with random edges do *not* contain the r value calculated by the Mantel Test [2].

Table 3 Comparison between the Mantel test and DIMECOST on data set 3

n	r	CI using Wilson	CI using DC-RST
10	−0.159	(−0.227, −0.108)	(−0.199, −0.083)
20	−0.136	(−0.200, −0.106)	(−0.217, −0.129)
30	0.043	(0.033, 0.105)	(−0.004, 0.071)
40	0.016	(−0.008, 0.045)	(0.003, 0.064)
50	0.072	(0.034, 0.090)	(0.037, 0.092)

Data Set 3: Random Lastly, we consider the case of two random distance metrics; i.e., the value returned by $d_1(a, b)$ and $d_1(b, a)$ is simply a random number (likewise for d_2). We expect these two distance metrics to be uncorrelated. The results are summarized in Table 3.

Again, the confidence intervals both contain the correlation coefficient calculated by the Mantel Test. These results suggests that DC-RST is indeed a suitable replacement for Wilson's algorithm in DIMECOST.

3 Performance Analysis

3.1 Implementation Details

We implemented DC- RST in C++ along with the OpenMP parallel library. The pseudocode is shown below in Fig. 8.

The function `random-partition()` was implemented by (1) initializing an array A with entries $1 \ldots n$ and (2) shuffling A uniformly at random. This array now defines the partitions. If parts $1 \ldots k$, have sizes $|p_1|, \ldots, |p_k|$, then the first $|p_1|$ elements of A are assigned to part 1, the next $|p_2|$ elements of A are assigned to part 2, etc. Step (2) of `random-partition()` involves *shuffling* an array of size

Fig. 8 Pseudocode for the parallel algorithm

```
V = [list of vertex numbers], F = [],
P = random-partition(V), k = |P|

#pragma omp parallel for
for (i in 0:k):
    F[i] = wilsons-algorithm(|P[i]|)

SuperTree = wilsons-algorithm(k)

T = merge(F, SuperTree)
```

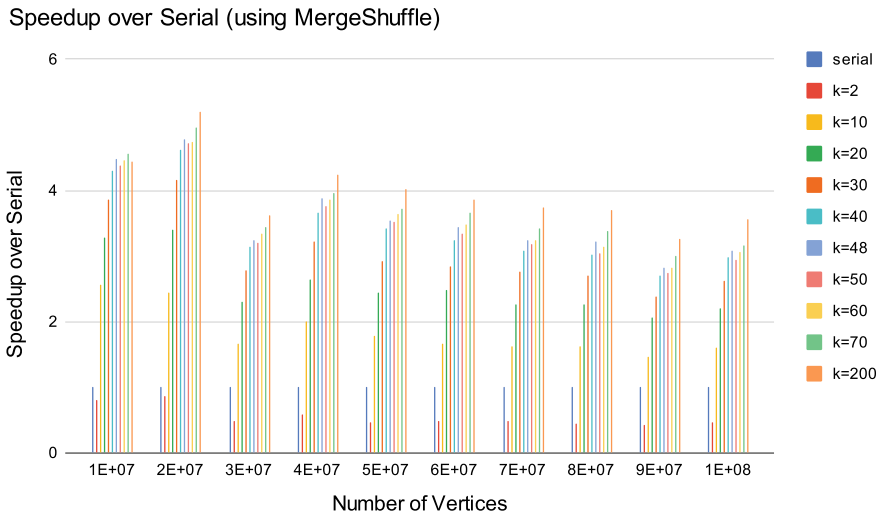


Fig. 9 Speedup over serial when using DC-RST with MERGESHUFFLE

n—the best known serial algorithm for this is Fisher-Yates, which runs in $O(n)$ time [16].

3.2 *Performance on a Symmetric Multiprocessor Architecture*

We compared DC- RST against the serial implementation of Wilson's algorithm on various combinations of k and n. The function `std::uniform_int_distribution` was used to generate the random numbers required in `wilsons-algorithm()`. This code was executed on Colorado School of Mines' "Isengard" server, which is a 48-core symmetric multiprocessor architecture with over 350 GB of main memory.

We implemented step (2) of `random-partition()` with a parallel shuffling algorithm: MERGESHUFFLE [16], which was chosen because of the availability of an implementation that uses OpenMP. The authors of MERGESHUFFLE suggest it is the current fastest parallel shuffling algorithm. The speedups are shown in Fig. 9. While this results in an improvement over Wilson's algorithm, observe that the speedup of DC-RST remains sub-linear w.r.t. the number of processors, with $k = 200$ partitions achieving a little less than 4x speedup on the largest clique.

To further assess our algorithm, we simply removed the shuffle step from `random-partition()`. For timing purposes this yields the equivalent of shuffling in 0 time. We called this variation NOSHUFFLE, and re-ran our benchmarks on

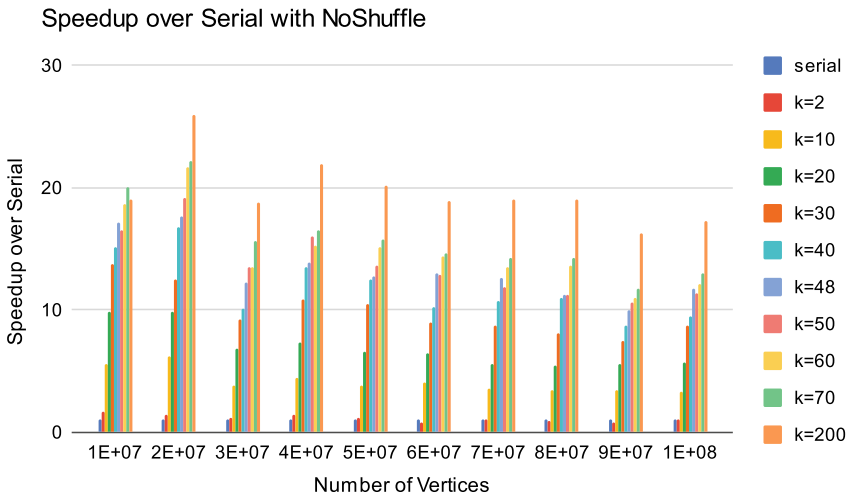


Fig. 10 Speedup over serial when using NoShuffle

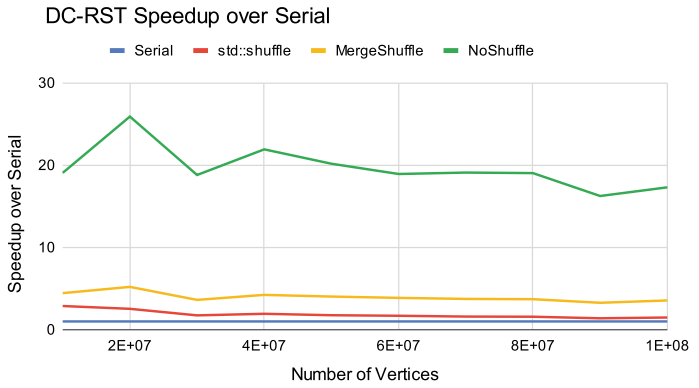


Fig. 11 Comparison of shuffling variations ($k = 200$)

NoShuffle to compare its performance to the previous results. This is shown in Fig. 10.

As expected, NoShuffle performs significantly better: on some inputs, NoShuffle is more than 20X faster than serial. This illustrates the potential for DC-RST to improve the performance of generating random spanning trees, should a better solution to Shuffle be found or if random partitions are pre-computed.

The results are summarized in Fig. 11. This figure illustrates the speedup the different variations of DC-RST (`std::shuffle`, MergeShuffle, and NoShuffle) over serial across the different clique sizes, all at $k = 200$ partitions. NoShuffle is by far the fastest, followed by MergeShuffle, with `std::shuffle` coming up last.

4 Conclusion

We have described DC-RST, a parallel divide-and-conquer algorithm for creating random spanning trees on a clique. While DC-RST does *not* create **uniform** random spanning trees, the impact on the proposed network science application, DIMECOST, is not significant and makes it suitable for replacement of Wilson's algorithm in DIMECOST. Additionally, the performance gains over Wilson's algorithm is significant: on a machine with 48 cores, DC-RST achieves 4X speedup when using MERGESHUFFLE, and when shuffling is not used at all, achieves 20X speedup. This points to the need for a faster parallel implementation of shuffling.

Note that to interface with DIMECOST, DC-RST only needs the size of the partitions (in order), and returns the list of edges in the computed spanning tree. This means the entire complete graph does *not* need to be generated to run DC-RST: instead, $n - 1$ queries could be executed to determine spanning tree edge-weights.

References

1. Mantel, N.: The detection of disease clustering and a generalized regression approach. Cancer Res. **27** (1967)
2. Bourbour, S., Mehta, D.P., Navidi, W.C.: Improved methods to compare distance metrics in networks using uniform random spanning trees (dimecost). Networks **76** (2020)
3. Anari, N., Hu, N., Saberi, A., Schild, A.: Sampling arborescences in parallel. In: Innovations in Theoretical Computer Science (2021)
4. Dray, S., Dufour, A.-B.: The ade4 package: implementing the duality diagram for ecologists. J. Stat. Softw. **22**(4), 1–20 (2007)
5. Schneider, J.W., Borlund, P.: Matrix comparison, part 2: measuring the resemblance between proximity measures or ordination results by use of the mantel and Procrustes statistics. J. Am. Soc. Inf. Sci. Technol. **58** (2007)
6. Sokal, R.R., Rohlf, F.J.: The comparison of dendrograms by objective methods. TAXON **11** (1962)
7. Corliss, J.O., Sneath, P.H.A., Sokal, R.R.: Numerical taxonomy: the principles and practice of numerical classification. Trans. Am. Microsc. Soc. **93** (1974)
8. Cooper-Ellis, S., Pielou, E.C.: The interpretation of ecological data. A primer on classification and ordination. Bryologist **97** (1994)
9. Ricaut, F.X., Auriol, V., Cramon-Taubadel, N.V., Keyser, C., Murail, P., Ludes, B., Crubézy, E.: Comparison between morphological and genetic data to estimate biological relationship: the case of the Egyin Gol necropolis (Mongolia). Am. J. Phys. Anthropol. **143** (2010)
10. Smouse, P.E., Long, J.C., Sokal, R.R.: Multiple regression and correlation extensions of the mantel test of matrix correspondence. Syst. Zool. **35** (1986)
11. Kouri, T.M., Awale, M., Slyby, J.K., Reymond, J.L., Mehta, D.P.: "Social" network of isomers based on bond count distance: algorithms. J. Chem. Inf. Model. **54** (2014)
12. Aldous, D.J.: The random walk construction of uniform spanning trees and uniform labelled trees. SIAM J. Discr. Math. **3** (1990)
13. Broder, A.: Generating random spanning trees (1989)
14. Lovász, L.: Random walks on graphs: a survey. Combinatorics **2** (1993)
15. Wilson, D.B.: Generating Random Spanning Trees More Quickly Than the Cover Time, vol. Part F129452 (1996)

16. Bacher, A., Bodini, O., Hollender, A., Lumbroso, J.: Mergeshuffle: A Very Fast, Parallel Random Permutation Algorithm, vol. 2113 (2018)

A Stochastic Approach for Extracting Community-Based Backbones

Zakariya Ghalmane, Mohamed-El-Amine Brahmia, Mourad Zghal, and Hocine Cherifi

Abstract Large-scale dense networks are very parvasive in various fields such as communication, social analytics, architecture, bio-metrics, etc. Thus, the need to build a compact version of the networks allowing their analysis is a matter of great importance. One of the main solutions to reduce the size of the network while maintaining its characteristics is backbone extraction techniques. Two types of methods are distinguished in the literature: similar nodes are gathered and merged in coarse-graining techniques to compress the network, while filter-based methods discard edges and nodes according to some statistical properties. In this paper, we propose a filtering-based approach which is based on the community structure of the network. The so-called "Acquaintance-Overlapping Backbone (AOB)" is a stochastic method which select overlapping nodes and the most connected nodes of the network. Experimental results show that the AOB is more effective in preserving relevant information as compared to some alternative methods.

Keywords Community structure · Weighted network · Backbone

1 Introduction

Complex networks are widely analyzed in various fields such as social, biological, communication and transportation [1–3]. Tremendously large real-world networks

Z. Ghalmane (✉) · M.-E.-A. Brahmia · M. Zghal
LINEACT CESI, Strasbourg, France
e-mail: zghalmane@cesi.fr

M.-E.-A. Brahmia
e-mail: abrahmia@cesi.fr

M. Zghal
e-mail: mzghal@cesi.fr

H. Cherifi
LIB, Burgundy University, Dijon, France
e-mail: hocine.cherifi@u-bourgogne.fr

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_5

are the results of the daily production of data. Because of memory and time constraints, it is more and more challenging and sometimes impossible to analyze such large-scale networks containing millions of nodes and billions of links. Therefore, the appropriate extraction of the essential nodes and links that conserve important information, while downsizing the network has become a matter of great importance. Backbone extraction methods provide a way to achieve this. The coarse-grained backbones and filter-based backbones are the two main research approaches that address this problem. In the first one, nodes sharing similarities are grouped together to reduce the size of the network [4, 5]. In the second case, nodes or edges are discarded from the network based on a given statistical criterion [6–10]. Serrano et al. propose the popular disparity filter method [10]. The latter one uses a null model of the link weights in order to retain statistically significant edges. The H-backbone [11] is another filter-based method which preserve only links with high betweenness and high h-index.

One of the important properties of real-world networks is community structure [14, 15]. Indeed, it strongly shapes their underlying functionality and dynamics [16–19]. It is usually represented as densely packed regions of connected nodes that are loosely connected with nodes from other regions. Communities can be non-overlapping [20–23] where each node belongs to one community, or overlapping [24–31] where nodes could belong to several communities simultaneously. Recent work has shown that community structure can be effectively exploited to extract backbones. The authors in [12] try to preserve only the overlapping nodes and the hubs in their backbone. By preserving these two types of nodes, they obtained a backbone that is more efficient than some alternative methods such as the Disparity filter backbone. In this context, the so-called "Acquaintance-Overlapping Backbone" selects these two types of nodes while following a stochastic approach. Our aim is to adapt this method for very large networks (composed of millions of nodes and billions of links) by proposing a technique having a very low computational complexity. The proposed method discards also links with very low weights while ensuring that the backbone remains composed of a single connected component. This is in order to preserve only very the relevant links that connects overlapping nodes and the nodes with the highest connectivity in the network. In this work, we suppose that the set of the overlapping nodes is already defined (if we have ground truth data) or can be easily defined by using a community detection algorithm. Then, the highly connected nodes of the network are defined by using a stochastic approach. This approach considers only the nodes that have been selected randomly as an acquaintance many times. The latter ones are likely to have a very high connectivity in the network.

Experiments are conducted on real-world weighted networks covering a variety of sizes and domains. The proposed method is compared to the previously developed community-based method "Overlapping nodes and hubs backbone" [12] as well as the "H-backbone" [11]. Results show that it is as effective as the Overlapping nodes and hubs backbone, and sometime it outperforms all the alternative methods in preserving the relevant information of the network. The major contributions of this manuscript can be briefly summarized as follows:

- The community structure of networks is exploited to create a backbone filtering method.
- Experiments conducted on real-worl weighted networks show that it performs as good or better than the other alternative methods.
- It can easily be adjusted to all types of networks (i.e., unweighted, directed and undirected networks).
- It can be also adapted to very large-scale networks (which is one of the most strong advantages of our method)

The remainder of this article is structured as follows. Section 2 introduces the Acaquaintance-Overlapping Backbone. Sections 3 presents all the datasets used in the experiments, while the evaluation measures are presented in Sect. 3. Section 5 of this article discusses the main results of the comparative analysis. Finally, Sect. 6 concludes this article.

2 Acquaintance-Overlapping Backbone (AOB)

The Acquaintance-Overlapping Backbone extraction process is based on the idea that hubs and overlapping nodes as well as the edges that connect them are the most influential elements in a network. The approach followed to extract the Acquaintance-Overlapping Backbone in weighted networks with an overlapping community structure is described as follows: the first step is to reveal the overlapping community structure of the network if the ground truth data is not available. After that, the Acquaintance-Overlapping Backbone can be defined. It combines two types of nodes; the overlapping nodes and the highly connected nodes of the network (i.e., hubs). The overlapping nodes are firstly selected in the backbone. These nodes may belong to more than one community simultaneously. Therefore, there is a great chance that they are connected to hubs located in different parts of the network, hence their big influence. Secondly, a stochastic approach is launched in order to preserve the most connected nodes in the backbone. To this end, nodes that verify two conditions are preserved. On one hand, random neighbors of randomly selected nodes that have been selected n times as an acquaintance are selected. In fact, most of real-world networks display the scale-free property [32]. They have, then, few hubs. Thus, selecting nodes that verify this first condition will ensure to reach the most connected nodes of the network. On the other hand, only acquaintances that have at least one overlapping node as a neighbor are maintained in the backbone. This second condition will allow us to select the most influential links in the network. Indeed, the overlapping nodes belong to several communities in the network. As a result, the links between them and the most connected nodes may act as a gateway to several areas in the network. On top of that, it will also ensure that there is only one connected component instead of many small ones.

The pseudo-code of the algorithm to extract the Acquaintance-Overlapping Backbone is given in Algorithm 1. Note that if the extracted backbone is composed of

Algorithm 1: Acquaintance-Overlapping Backbone Extraction

Input : $G(V, E)$: Graph,
$C = \{C_1, C_2, ..., C_\alpha\}$: Community set, where α is the number of the communities in the network,
n: Number of times nodes should be selected as an acquaintances,
s: Parameter that controls the size of the backbone
Output: $\widehat{G}(\widehat{V}, \widehat{E})$: Backbone or pruned graph

Define the variable n_{acq} as the number of hubs or acquaintances that should be preserved in the backbone $\widehat{G}$
Define the variable $counter$ as the the counter of the acquaintances
Define the set of the overlapping nodes A of size n_o
$n_{acq} \leftarrow s - n_o$
$counter \leftarrow 0$
while $counter < n_{acq}$ **do**
    Select a random node v
    Select randomly one of its acquaintances v_{acq}
    if v_{acq} *was selected as acquaintance n times* **then**
        Add v_{acq} to the set of acquaintances B
        $counter \leftarrow counter + 1$
    end
end
$\widehat{V} = A \cup B$
if $\widehat{G}$ *is disconnected* **then**
    $\widehat{G} \leftarrow LCC(\widehat{G})$ // LCC returns the Largest Connected Component of $\widehat{G}$
end
Sort the set of edges $\widehat{E}$ according to their weights
Remove the edges with small weights $e \in \widehat{E}$ while keeping the largest connected component connected
Return the backbone $\widehat{G}$

many components, only the largest connected component is maintained. Yet, with the second condition of the acquaintances selection, there is a great chance to extract only one single component. Moreover, all the links with small weights are removed from the obtained backbone to keep only the most important ones. This is while ensuring that the backbone stays composed as one connected component.

3 Datasets

A set of seven real-world networks of different sizes and from multiple fields (technological, social, transportation and collaborative) are used in the experiments. The size of the studied networks (number of nodes and edges) ranges from a few dozen to thousands. Small networks are included in this study for a better understanding to the extraction process. The topological properties of the networks are reported in Algorithm 1. The community structure of the network are used to uncover

the Speaker-Listener Label Propagation Detection Algorithm (SLPA) [13]. All the datasets used in this work are available on "github.com/zakariyaGH/Datasets" and "toreopsahl.com/datasets".

1. **Zachary's Karate Club**: the members of the karate club are represented by nodes and their friendship is represented by links. The links are also weighted by the relative interactions that take place between the members.
2. **Les Misérables**: the characters of the novel 'Les Misérables' are represented by the nodes, while the links indicate their co-appearance in the same chapter. The weights stand for the number of co-appearances.
3. **Madrid Train Bombing**: Nodes are the terrorists involved in the bombing of the train in Madrid on March 11, 2004. The links are the connections between the terrorists, while the weights are measured by the strength of their relationship.
4. **US Airport network**: The commercial airports of the United States are represented by nodes in this networks. Only the flights scheduled in 2010 are collected in this dataset.
5. **Openflights network**: This network was formed based on the dataset downloaded from "Openflights.org" which contains non-US-based airports. The Openflights network is weighted by the number of routes between two airports.
6. **Facebook-like Forum network**: This network contains 899 nodes representing facebook users sharing messages about 522 subjects in a forum. The messages exchanged are represented by links. The amount of messages posted by a user on a given topic represent the weights of links.
7. **Scientific Collaboration**: The authors of the articles published in the "Condensed Matter" category of arXiv are represented by nodes in this network, while the links indicate their co-authorship. The links are weighted by the number of times they were co-authors.

4 Evaluation Measures

In this paper, the similarity between the proposed backbone extraction technique and both the Overlapping hub backbone and the H-backbone is measured. Then, their effectiveness is compared based on three classical evaluation measures (Table 1).

1. **Proportion of common nodes**: This similarity measure indicates the number of elements in two different sets of the same size. By definition, it is the ratio of the size of the intersection of the two sets divided by their size. The proportion of common nodes A_n between two sets of nodes having the size n is:

$$A_n = \frac{|X \cap Y|}{n} \tag{1}$$

Where X and Y represent the sets of nodes of two different backbones. n denotes their size.

Table 1 The estimated parameters of the real-world networks

Networks	|V|	|E|	$\langle k \rangle$	ρ	r	c
Zachary's karate club	33	77	13.59	0.114	−0.476	0.256
Madrid Train Bombing	62	243	8.81	0.121	0.029	0.561
Les Misérables	77	254	21.3	0.087	−0.165	0.499
Facebook-like forum	899	7046	1.09	0.017	−0.108	0.06
US Airport network	1574	17215	66.15	0.014	−0.113	0.384
Openflights network	7976	15677	0.234	0.004	0.05	0.254
Scientific Collaboration	16726	47594	9.23	0.0003	0.185	0.36

N denotes the number of nodes in the network. $|E|$ is the number of edges. $\langle k \rangle$ is the average weighted degree. ρ is the density of the network. r indicates the assortativity while c refers to the transivity of the network

2. **Average Weighted degree**: The weighted degree of a node denotes the total sum of the weights of its immediate links (links connected to it). Thus, a backbone with a higher average weighted degree tends to keep the most significant nodes, highlighting its connectivity. The average weighted degree is defined as follows:

$$< k >= \frac{1}{|V|} \sum_{i,j \in V} w_{ij_{j \in N_i(1)}} \tag{2}$$

   $N_i(1)$ represents the set of the immediate neighbors of the node i.
3. **Average link weight**: The information flow of the original network should be preserved by the backbone links. Accordingly, a higher average link weight value implies a better backbone to preserve the network core information. The average weight is defined as follows:

$$< w >= \frac{1}{|V|} \sum_{i,j \in V} w_{ij} \tag{3}$$

4. **Average betweenness**: Information can be disseminated more rapidly by nodes with a high betweenness. As a result, backbones with higher average betweenness are more likely to preserve the speed of information dissemination of the original network. The betweenness of a node i is defined as follows:

$$\beta_i = \sum_{s,t \in G} \frac{\sigma_{st}^i}{\sigma_{st}} \tag{4}$$

where σ_{st} represents the number of shortest paths between the nodes s and t, and σ_{st}^{i} is the number of shortest paths between nodes s and t and passing through the node i. The average betweenness is de fined as:

$$< \beta >= \frac{1}{|V|} \sum_{i=1}^{|V|} \beta_i \tag{5}$$

5 Experimental Results

In this section, the Acquaintance-Overlapping Backbone is compared to the Overlapping nodes and hubs backbone and the H-backbone. The three backbones are compared in terms of both their similarity and effectiveness. In all the experiments, the backbone size is fixed to 30% of the original network. In a previous work [33], the results showed that the community structures uncovered by multiple community detection algorithms are quite stable in these networks. That is why, the SLPA is the only algorithm that was used in this work. At first, the proportion of common nodes is used to check if the proposed backbone select the same nodes as the alternative ones. Before reporting all the results, let's consider 'Les Misérables' network (represented in Fig. 1) which is a network having a very small size to illustrate the differences between the three backbones visually. One can point out form Fig. 2 that the Acquaintance-Overlapping Backbone is very similar to the Overlapping nodes and hubs backbone. Indeed, these two backbones select all the most important characters of Victor Hugo's novel (Valjean, Javert, Cosette, Mr. and Mme thenardier, Fantine, Marius and Gaveroche). Similarly to the Overlapping nodes and hubs backbone, the Acqauintance-Overlapping Backbone based method tend by following a stochastic approach to target the same type of nodes. That is the reason why these two backbones are quite similar. However, the H-backbone is relatively different to the other backbones. Actually, the latter one misses some characters such as Gaveroche which is one of the most important characters of this novel. This is because this technique is based on link selection instead of node selection as it is the case of the other backbone extraction techniques. The same results are noticed for all the other networks reported in Table 2.

The effectiveness of the proposed backbone and its alternatives is investigated and reported in Table 2. To this end, three backbone quality metrics are employed: the average weighted degree ($\langle k \rangle$), the average link weight ($\langle w \rangle$), the average betweenness ($\langle \beta \rangle$). The average weighted degree is firstly discussed. The backbone extraction technique with the highest value is the one that succeeds in preserving the salient nodes having the highest connectivity in the original network. According to the results reported in Table 2, the H-backbone has the lowest performance. Additionally, the Acquaintance-Overlapping Backbone has comparable results to the Overlapping nodes and hubs backbone. Yet, the average weighted degree of the proposed method is slightly even higher in both Madrid train bombing and Scientific Collaboration

Table 2 The estimated values of: the proportion of common nodes (A_n), the average weighted degree ⟨k⟩, the average link weight ($\langle w \rangle$) and the average betweenness ($\langle \beta \rangle$)

	$A_n(\%)$		$\langle k \rangle$			$\langle w \rangle$			$\langle \beta \rangle$		
Networks	AOB-OH	AOB-HB	AOB	OH	HB	AOB	OH	HB	AOB	OH	HB
Zachary's karate club	88.08	70.1	12.33	12.71	11.59	4.89	3.46	3.22	0.08	0.09	0.067
Madrid Train Bombing	91.13	74.7	17.88	16.77	13.21	2.01	1.38	1.25	0.068	0.077	0.061
Les Misérables	89.62	79.87	22.47	25.58	20.86	5.93	4.89	5.15	0.063	0.064	0.062
Facebook-like forum	90.04	56.71	1.11	1.15	1.03	8.55	6.99	7.45	0.009	0.005	0.003
US Airport network	96.26	64.09	70.4	71.89	67.31	886954	816292	799456	0.055	0,051	0.048
Openflights network	92.01	63.48	0.341	0.356	0.303	215.71	205,69	209.09	0.039	0,04	0.036
Scientific collaboration	90.07	62.13	16.89	16.53	15.98	49.9	49.7	48.9	0.011	0.008	0.007

For brevity, AOB stands for "Acquaintance-Overlapping Backbone", OH stands for "Overlapping nodes and hubs backbone", while HB stands for "H-backbone". Each reported value is the average of 10 SLPA simulation runs. The standard deviation is not in this table due to its small values ranging between 0 and 2%

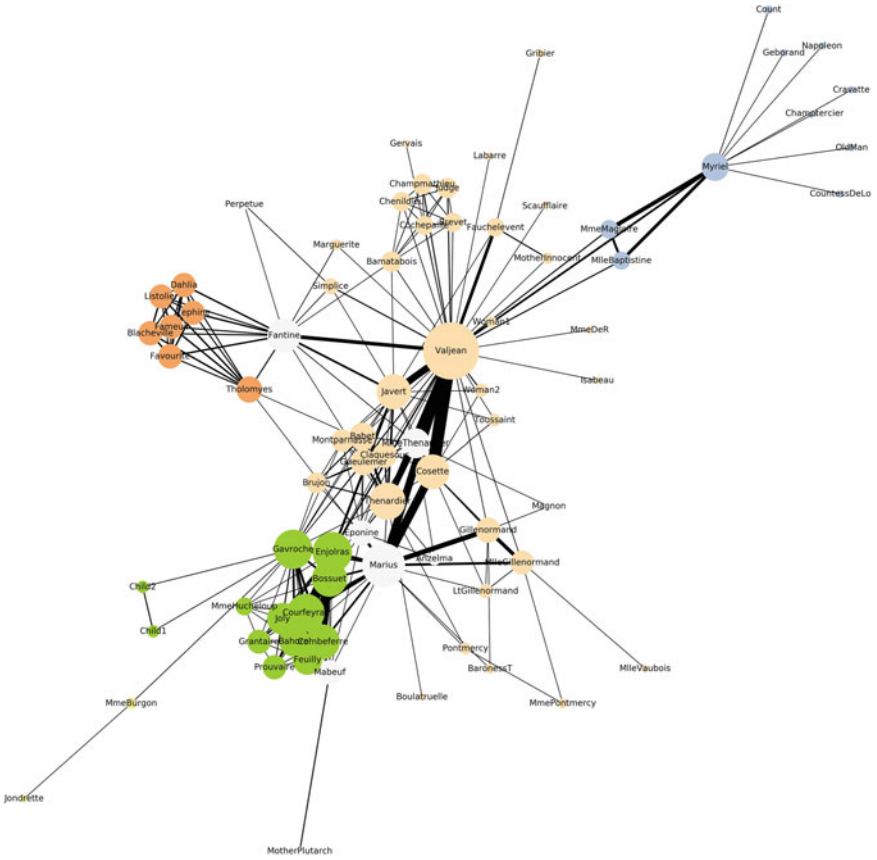


Fig. 1 Les Misérables network. The overlapping nodes are in grey while the other colors represent the different communities. The size of the nodes is proportional to their weighted degree. Likewise, the weight of links determine their size

networks. Indeed, as it was mentioned previously, these two backbones select about the same sets of nodes in all the networks under study. These results confirm our intuition that nodes that have been selected many times as acquaintances are more likely to be the highly connected nodes of the network.

Now, the results of the average link weight are discussed. The average link weight is an indicator of the pertinence of the links maintained in a backbone. Therefore, as its value increases, the more the backbone is likely to retain essential and critical links. The results show that the Acquaintance-Overlapping Backbone outperforms the other alternative methods. Indeed, the proposed backbone extraction technique keeps only the links with high weights connecting the overlapping nodes and the highly connected nodes of the network. These links connect all the various communities and zones of the networks. Note that the Overlapping nodes and hubs method

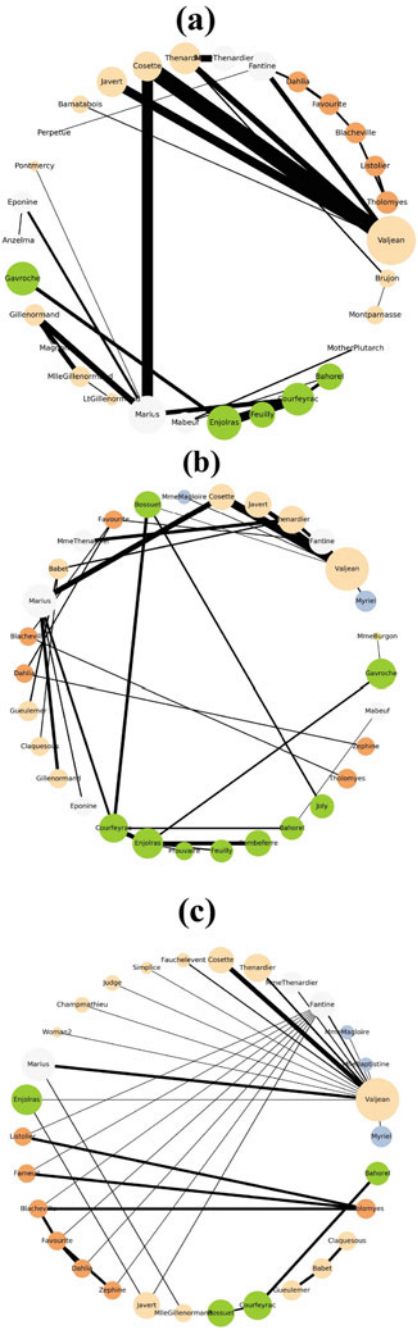


Fig. 2 Different methods for extracting the backbone of Les Misérables' network. **a** Acquaintance-overlapping backbone, **b** overlapping nodes and hubs backbone, **c** H-backbone. The overlapping nodes are in grey while the other colors represent the different communities. The size of the nodes is proportional to their weighted degree. Likewise, the weight of links determine their size

preserves all the links between the overlapping nodes and hubs which could include some links with minor pertinence. That shows that the proposed backbone can preserve links that play a very important role in information dissemination in the original network as compared to the other methods.

Finally, let's turn to the average betweenness. This measure reflects the volume of information flow that may pass over the nodes of a backbone. The greater a backbone's value, the greater its efficiency in spreading information. Results show that the Acquaintance-Overlapping Backbone as well as the Overlapping nodes and hubs backbone have very close values for all the networks under study. In some networks the proposed method outperforms all the others, whereas in others the Overlapping nodes and hubs backbone performs better. Therefore, they can maintain the nodes that participate the most in the information flow of the network. However, both backbone extraction techniques have higher performance than the H-backbone in terms of the average betweenness.

To summarize, the Acquaintance-Overlapping Backbone performs as good as the Overlapping nodes and hubs backbone and sometimes better, although it is a stochastic approach. This is because they preserve around the same set of nodes in all the networks. However, the H-backbone has always the lowest performance. Furthermore, the proposed backbone extraction method is also the most effective one in terms of the average link weight. Therefore, it preserves essential nodes and the most relevant links in the network. On top of that, the Overlapping nodes and hubs backbone extraction method requires information about each node of the network, while the H-backbone needs to compute the betweenness for each link of the network, making it a global method. However, the proposed method follows a stochastic approach. Thus, it only needs local information about some selected nodes of the network. It has, then, a very low computational complexity. It is around $O(k)$, where k is the number of nodes that must be preserved in the backbone. Therefore, the Acquaintance-Overlapping Backbone is the most suitable extraction method for very large network, given its very low computational complexity.

6 Conclusion

Understanding large-scale networks is crucial to better comprehend their underlying dynamics and topology. However, the massive size of some networks makes this difficult. Thus, it is important to remove all the redundant information while maintaining the relevant nodes and links of the network. Both coarse-grained and filter-based backbones are proposed to address this problem.

In this work, a filtering-based method is proposed which is based on community structure of networks. It selects firstly the overlapping nodes given their importance on connecting all the different communities of the network. Moreover, it follows a stochastic approach to select also the highly connected nodes given their big influence in the network. This is done by randomly selecting the nodes that have been selected as acquaintances multiple times. Additionally, this method preserve also the highly

weighted links between these to type of nodes which strategically connect nodes from different parts of the network, hence their importance.

Experiments show that the Acquaintance-Overlapping Backbone performs as good as the Overlapping nodes and hubs backbone in terms of the connectedness of the selected nodes measured by the average weighted degree, as well as the information flow measured by the average betweenness. However, it outperforms all the alternative methods in term of the relevancy of links (contributing to the efficiency of information spreading) measured by the average link weight. Globally, the H-backbone has always the lowest effectiveness in all these three aspects. Furthermore, the proposed method has a very low computational complexity since it is based on a stochastic approach, which makes it a perfect fit with very large networks.

References

1. Sporns, O.: Graph theory methods: applications in brain networks. Dialogues in clinical neuroscience (2022)
2. Herrera, M., Pérez-Hernández, M., Kumar Parlikad, A., Izquierdo, J.: Multi-agent systems and complex networks: Review and applications in systems engineering. Processes **8**(3), 312 (2020)
3. Soloviev, V., Solovieva, V., Tuliakova, A., Hostryk, A., Pichl, L.: Complex networks theory and precursors of financial crashes. In: CEUR Workshop Proceedings (2020)
4. Gfeller, D., De Los Rios, P.: Spectral coarse graining of complex networks. Phys. Rev. Lett. **99**(3), 038701 (2007)
5. Chen, M., Li, L., Wang, B., Cheng, J., Pan, L., Chen, X.: Effectively clustering by finding density backbone based-on kNN. Pattern Recogn. **60**, 486–498 (2016)
6. Goh, K.I., Salvi, G., Kahng, B., Kim, D.: Skeleton and fractal scaling in complex networks. Phys. Rev. Lett. **96**(1), 018701 (2006)
7. Grady, D., Thiemann, C., Brockmann, D.: Robust classification of salient links in complex networks. Nat. Commun. **3**(1), 1–10 (2012)
8. Zhang, X., Zhu, J.: Skeleton of weighted social network. Phys. A Stat. Mechan. Appl. **392**(6), 1547–1556 (2013)
9. Simas, T., Correia, R.B., Rocha, L.M.: The distance backbone of complex networks. J. Complex Netw. **9**(6), cnab021 (2021)
10. Serrano, M.Á., Boguná, M., Vespignani, A.: Extracting the multiscale backbone of complex weighted networks. Proc. Nat. Acad. Sci. **106**(16), 6483–6488 (2009)
11. Zhang, R.J., Stanley, H.E., Ye, F.Y.: Extracting h-backbone as a core structure in weighted networks. Sci. Rep. **8**(1), 1–7 (2018)
12. Ghalmane, Z., Cherifi, C., Cherifi, H., El Hassouni, M.: Extracting backbones in weighted modular complex networks. Sci. Rep. **10**(1), 1–18 (2020)
13. Xie, J., Szymanski, B.K., Liu, X.: SLPA: uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. In: 2011 IEEE 11th International Conference on Data Mining Workshops, pp. 344–349. IEEE (2011)
14. Cherifi, H., Palla, G., Szymanski, B.K., Lu, X.: On community structure in complex networks: challenges and opportunities. Appl. Netw. Sci. **4**(1), 1–35 (2019)
15. Fortunato, S., Hric, D.: Community detection in networks: a user guide. Phys. Rep. **659**, 1–44 (2016)
16. Xu, Y.: A spectral method to detect community structure based on the communicability modularity. Phys. A Stat. Mech. Appl. **537**, 122751 (2020)

17. Guo, K., He, L., Chen, Y., Guo, W., Zheng, J.: A local community detection algorithm based on internal force between nodes. Appl. Intell. **50**(2), 328–340 (2020)
18. Ghalmane, Z., El Hassouni, M., Cherifi, C., Cherifi, H.: Centrality in modular networks. EPJ Data Sci. **8**(1), 15 (2019)
19. Tulu, M.M., Hou, R., Younas, T.: Identifying influential nodes based on community structure to speed up the dissemination of information in complex network. IEEE Access **6**, 7390–7401 (2018)
20. Ghalmane, Z., El Hassouni, M., Cherifi, H.: Betweenness centrality for networks with non-overlapping community structure. In: 2018 IEEE Workshop on Complexity in Engineering (COMPENG), pp. 1–5. IEEE (2018)
21. Xu, E.H., Hui, P.M.: Uncovering complex overlapping pattern of communities in large-scale social networks. Appl. Netw. Sci. **4**(1), 1–16 (2019)
22. Jiang, H., Liu, Z., Liu, C., Su, Y., Zhang, X.: Community detection in complex networks with an ambiguous structure using central node based link prediction. Knowl.-Based Syst. **195**, 105626 (2020)
23. Saxena, R., Kaur, S., Bhatnagar, V.: Social centrality using network hierarchy and community structure. Data Mining Knowl. Discovery **32**(5), 1421–1443 (2018)
24. Palla, G., Derényi, I., Farkas, I., Vicsek, T.: Uncovering the overlapping community structure of complex networks in nature and society. Nature, **435**(7043), 814–818 (2005)
25. Lancichinetti, A., Fortunato, S., Kertész, J.: Detecting the overlapping and hierarchical community structure in complex networks. New J. Phys. **11**(3), 033015 (2009)
26. Ma, H., Yang, H., Zhou, K., Zhang, L., Zhang, X.: A local-to-global scheme-based multi-objective evolutionary algorithm for overlapping community detection on large-scale complex networks. Neural Comput. Appl. **33**(10), 5135–5149 (2021)
27. Ghalmane, Z., Cherifi, C., Cherifi, H., Hassouni, M.E.: Centrality in complex networks with overlapping community structure. Sci. Rep. **9**(1), 1–29 (2019)
28. Taghavian, F., Salehi, M., Teimouri, M.: A local immunization strategy for networks with overlapping community structure. Phys. A Stat. Mech. Appl. **467**, 148–156 (2017)
29. Yang, J., Leskovec, J.: Structure and overlaps of ground-truth communities in networks. ACM Trans. Intell. Syst. Technol. (TIST) **5**(2), 1–35 (2014)
30. Kudelka, M., Ochodkova, E., Zehnalova, S., Plesnik, J.: Ego-zones: non-symmetric dependencies reveal network groups with large and dense overlaps. Appl. Netw. Sci. **4**(1), 1–49 (2019)
31. Tarkowski, M., Szczepański, P., Rahwan, T., Michalak, T., Wooldridge, M.: Closeness centrality for networks with overlapping community structure. Proc. AAAI Conf. Artif. Intell. **30**(1) (2021)
32. Wang, X.F., Chen, G.: Complex networks: small-world, scale-free and beyond. IEEE Circ. Syst. Mag. **3**(1), 6–20 (2003)
33. Ghalmane, Z., Cherifi, C., Cherifi, H., El Hassouni, M.: Exploring hubs and overlapping nodes interactions in modular complex networks. IEEE Access **8**, 79650–79683 (2020)

Correcting Output Degree Sequences in Chung-Lu Random Graph Generation

Christopher Brissette, David Liu, and George M. Slota

Abstract Random graphs play a central role in network analysis. The Chung-Lu random graph model is one particularly popular model, which connects nodes according to their desired degrees to form a specific degree distribution in expectation. Despite its popularity, the standard Chung-Lu graph generation algorithms are susceptible to significant degree sequence errors when generating simple graphs. In this manuscript, we suggest multiple methods for improving the accuracy of Chung-Lu graph generation by computing node weights which better recreate the desired output degree sequence. We show that each of our solutions offer a significant improvement in degree sequence accuracy.

Keywords Graph theory · Random graphs · Graph generation

1 Introduction

Random graph generation is an important task in several fields of study, such as biology and the social sciences. Random graphs arising from graph generation algorithms have uses as null models and as algorithmic benchmarks [7, 10, 14]. Stochastic block models are random graph models in which a number of nodes is predefined and each possible edge is assigned a probability of existing [12]. A random graph can then be constructed by generating edges with respect to these assigned probabilities. Conversely, the common configuration model [3, 15] assigns to each vertex some number of stubs, equal to each vertex's desired degree, and then selects two stubs uniformly at random to create an edge. This process is repeated until all stubs are

C. Brissette (✉) · D. Liu · G. M. Slota
Rensselaer Polytechnic Institute, Troy, NY, USA
e-mail: brissc@rpi.edu

D. Liu
e-mail: liud9@rpi.edu

G. M. Slota
e-mail: slotag@rpi.edu

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_6

attached, and a graph exactly matching some input degree sequence can be output.

The expected pairwise degree probabilities (probability of a vertex of degree x attaching to a vertex of degree y) arising from the configuration model may be expressed as a stochastic block model via the Chung-Lu random graph model [6]. This model assigns a weight w_v to each node $v \in V$ in the graph, and then it attaches nodes $u, v \in V$ according to the probability $p_{uv} = w_u w_v / \sum_{a \in V} w_a$. If each weight is taken to be the desired degree of each given node, then this model produces a desired degree distribution, but only in expectation. This model is used as a subroutine in more complex graph generation algorithms [8, 13, 17, 18], and the given probabilities are also implicitly used to define network measures such as modularity for graph clusters [9, 16].

Despite its popularity and theoretical importance, Chung-Lu random graph generation often results in graphs with significant degree distribution errors [5, 8, 10, 13, 19]. While this can be resolved in many cases by using an explicit graph configuration model instead of Chung-Lu, these approaches have limited room for parallelism and are not scalable, particularly when a simple graph output is desired. An appealing feature of Chung-Lu graph generation is that the method is naïvely parallelized and is additionally expedited by techniques such as edge skipping [1, 2], even when generating simple graphs. Because of the algorithm's high scalability and wide usage, some work has been done to correct and quantify these errors. In Durak et al. [8], it is shown that Chung-Lu generation often under-estimates the number of low degree nodes, and they perform an artificial inflation in the number of nodes with unit weight to account for this. Other related work has used this specific approach [13] or a similar approach where unit weight nodes are instead manually configured [17]. Alternatively, in our prior work [5], we approximate the output degree distribution of Chung-Lu generation with a matrix equation, and we solve a linear system to determine an input distribution that will best generate the desired output. We also show that the inverse of many degree sequences with respect to this matrix yield vectors with negative entries. These vectors do not have a useful interpretation with respect to the Chung-Lu generation algorithm, and the method is greatly limited because of this. This manuscript aims to remedy the issues present in previous work while utilizing the same matrix model as an important building block.

In the Chung-Lu model, each set of nodes with the same weight w may be discussed as a block in a stochastic block model, and the degrees within that block should be approximately Poisson distributed about the weight w. Therefore, for degrees $\{w_1 = 1, w_2 = 2, \ldots, w_d = d\}$, one may generate a matrix $\mathbf{P}$ given by Eq. 1. Each column of the matrix represents the probability mass function of degrees within each block. The inner product of any given row r of this matrix with a degree sequence vector adds together the predicted number of nodes with degree r produced by each block. By considering the entire matrix simultaneously, the output vector predicts the output of Chung-Lu. That is, by representing an input degree sequence as a vector x, one can approximate the output degree sequence of Chung-Lu generation y as $\mathbf{P}x = y$. As presented, $\mathbf{P}$ is of infinite dimension and needs to be reduced to a finite dimension for computational use. For this purpose, the matrix is truncated by removing all rows beyond some maximum degree. This cut off can be chosen depending

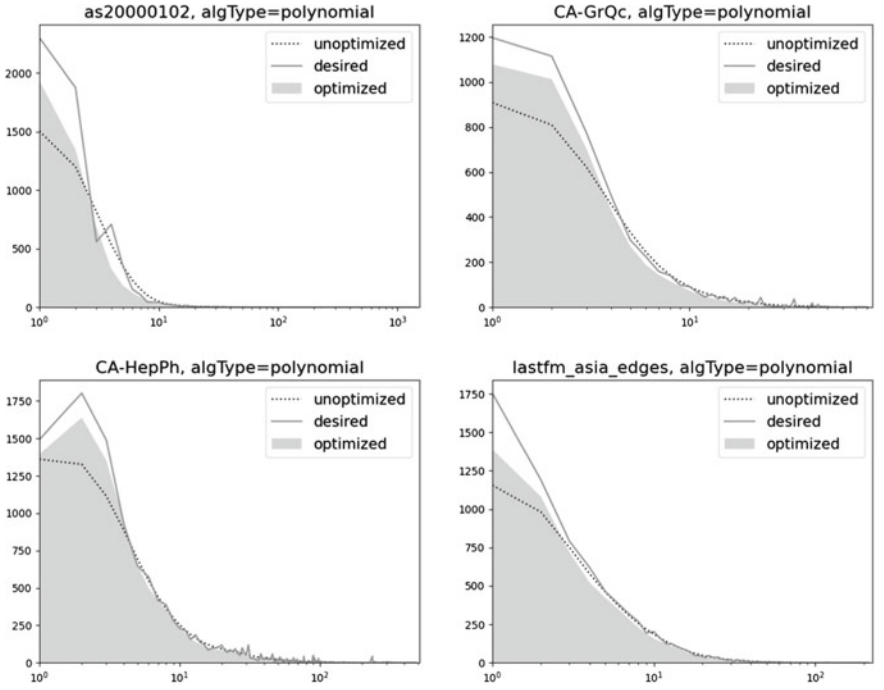


Fig. 1 Visualization of degree sequences: Comparisons of degree sequences for the as20, GrQc, HepTh, and lastfm graphs. The dotted lines denote the predicted output using standard Chung-Lu weights, the solid grey region denotes the output sequence using optimized Chung-Lu weights, and the solid line denotes the desired output sequence. Each optimization here was performed using our polynomial update method which is discussed in Sect. 2

on the desired output distribution y, and d may also be chosen to make P square, in which case an explicit inverse is known. For details regarding this analysis, a reader may consult the original citation [5].

$$\mathbf{P} = \begin{bmatrix} | & | & & | \\ poiss(w_1) & poiss(w_2) & \cdots & poiss(w_d) \\ | & | & & | \end{bmatrix} \tag{1}$$

We note that this matrix can be easily generalized. By choosing a set of arbitrary positive weights $w = \{w_1, w_2, \ldots, w_d\}$, instead of simply the nodal degrees, one obtains a matrix $\mathbf{P}(w)$ where the means of each Poisson distribution correspond to the given weights. This produces a new stochastic block model.

Our Contribution: This paper focuses on determining, for a given number of weight parameters d, the set of weights such that the error between the desired output and actual output of Chung-Lu graph generation will be minimized. We develop and optimize several novel methods to minimize this error. Visualized in Fig. 1 for several

graphs in the Stanford Large Network Dataset Collection[1] are their degree sequences, the unoptimized output from Chung-Lu generation using these degree sequences, and the generation output after applying one of our methods. We will discuss our varying methods in Sect. 2 and analyze their results in Sects. 3 and 4.

2 Methods

As we note in our prior work [5], there are numerous sequences that can not be reliably generated using naïve Chung-Lu weights. To remedy this short coming, there are two algorithm parameters which may be adjusted to alter the output. One parameter is the input sequence. This is the specific parameter studied in prior work. The other parameter is the set of weights $w = \{w_1, w_2, \ldots, w_d\}$. Conceptually, both methods are trying to approximate a distribution as a linear combination of Poisson distributions. In the former method, the Poisson distributions have means equal to the target degree classes, and the approximation is improved by altering the coefficients applied to each distribution. Alternatively, changing the weights equates to changing these means, effectively moving the Poisson distributions along the x-axis.

We present two methods incorporating weight alteration. Our first method relies on several greedy updates, where weights are chosen such that $\|\mathbf{P}(w)x - y\|$ is minimized at each step. The latter method uses maximum likelihood estimation [20] to solve for weights.

Before discussing either method, let us first formalize goals and definitions. Take $\mathbf{P}$(w) to be the square matrix given by weights $w = \{w_1, \ldots, w_d\}$ and removing both the first row and everything beyond row $d + 1$ in Eq. 1. The first row is removed because it corresponds to the number of zero-degree nodes. These nodes may be ignored after generation, so removing the first row of $\mathbf{P}$(w) mathematically represents this. We call our input degree sequence vector $x = [x_1, \ldots, x_d]$ and our desired output degree sequence vector $y = [y_1, \ldots, y_d]$. Additionally, call the output of the Chung-Lu algorithm with weight set w and input vector x, $CL(w, x)$. Then, our goal is to find a combination w, x such that $\|CL(w, x) - y\|_1$ is minimized. The 1-norm is specifically considered because it can be directly interpreted as the number of nodes with incorrect degrees. A $\log_2$-binned version of this error will additionally be considered later and is discussed in Sect. 3.

2.1 Greedy Updates

We first discuss the greedy update method. This is based off of a simple approximation and update loop. The basic idea is as follows. Given an input degree sequence x, determine the first k derivatives of each column of $\mathbf{P}$(w) with respect to their means and use these derivatives to approximate $\|\mathbf{P}(w + \epsilon)x - y\|$ for small perturbations in

[1] https://snap.stanford.edu/data/.

the elements of the mean-set $w + \epsilon$. Then, update the means in the optimal direction according to some minimization algorithm and repeat this process for some number of iterations.

Two update objectives are discussed in this section, which we call linear updates and polynomial updates. These objectives only differ in the number of derivatives considered. Linear updates approximate error based on the first derivative of each column in $\mathbf{P}$(w) . Alternatively, polynomial updates use an arbitrary number of k derivatives and the Taylor series to approximate error. As is shown later, both of these methods reduce the per-node degree error significantly; however, they require different numbers of iterations. All instances of the polynomial-update variant use $k = 2$ in this manuscript. In Algorithm 1, we show a general template of the greedy method. The main difference in each of these methods comes from how our objective changes the `opt_E`$(\cdot)$ function. The objectives are discussed in more detail in the following subsections.

Algorithm 1 Greedy-Update (x, y, $\{w_1, \ldots, w_d\}$, δ, t)

1: $\mathbf{P} \leftarrow$ fill_P($\{w_1, \ldots, w_d\}$)
2: **for** iters $\in$ [1…t] **do**
3: $\quad \mathbf{U} = \{\mathbf{U}_1, \ldots, \mathbf{U}_k\} \leftarrow compute\_U\_set(\{w_1, \ldots, w_d\})$
4: $\quad \mathbf{E} \leftarrow$ opt_E($\mathbf{P}$, $\mathbf{U}$, x, y, δ)
5: $\quad \mathbf{P} \leftarrow$ fill_P($\{w_1 + E_{11}, \ldots, w_d + E_{dd}\}$)
6: **return** $\{w_1, \ldots, w_d\}$

Algorithm 1 is initialized with an input degree sequence vector x, a desired output degree sequence vector y, a set of initial weights $\{w_1, \ldots, w_d\}$, a maximum update step-size δ, and an iteration number t. For this paper, initial degree sequences are taken to be $x = cy$ for some positive constant $c \in \mathbb{R}^+$. Additionally, initial weights are taken to be $\{w_1 = 1, \ldots, w_d = d\}$. The iteration number and step size will vary depending on desired accuracy and whether linear, or polynomial updates are being used. The algorithm proceeds as follows. $\mathbf{P}$(w) is initialized with the input weights. Then, within the loop, a set of matrices $\mathbf{U} = \{\mathbf{U}_1, \ldots, \mathbf{U}_k\}$ is computed within the `compute_U_set`$(\cdot)$ function. Each matrix $\mathbf{U}_i$ corresponds to the i^{th} derivative of each column. These matrices are then used in the `opt_E`$(\cdot)$ function to determine how much each mean in w should change. For the purposes of this manuscript, `opt_E`$(\cdot)$ uses the sequential least squares minimization [4] implementation from `scipy.optimize.minimize`$(\cdot)$ in `Python`. Then new weights are computed and $\mathbf{P}$(w) is updated.

Linear Updates: Linear updates are the simpler of the two greedy update methods. In the linear update method, $k = 1$ and only a single $\mathbf{U}$ matrix is computed in `compute_U_set`$(\cdot)$. This matrix has the same form given in Eq. 2 and the columns take the form of the derivatives of the columns in $\mathbf{P}$(w) as given in Eq. 3 with respect to their means. In Eq. 2, μ_j corresponds to the mean of the Poisson distribution.

$$\mathbf{U} = \begin{bmatrix} | & | & & | \\ \frac{\partial}{\partial \mu_1} poiss(\mu_1, x) & \frac{\partial}{\partial \mu_2} poiss(\mu_2, x) & \cdots & \frac{\partial}{\partial \mu_m} poiss(\mu_m, x) \\ | & | & & | \end{bmatrix} \tag{2}$$

$$\frac{\partial}{\partial \mu_i} poiss(\mu_i, x) = \frac{(x - \mu_i) e^{-\mu_i} \mu_i^{x-1}}{x!} \tag{3}$$

The linear update objective function used in `opt_E`$(\cdot)$ takes the form of minimizing $\gamma = \|(\mathbf{P}(w) + \mathbf{UE})x - y\|_2$ with respect to the diagonal matrix $\mathbf{E}$, where each entry is bounded by δ, $|E_{jj}| \leq \delta$. Unfortunately, linear approximations lack significant accuracy, and as such, the step size δ needs to be rather small to maintain stability within each optimization step `opt_E`$(\cdot)$. This ultimately leads to a method which requires many updates. This can be prohibitive for graphs with high maximum degree, since the dimensionality of our optimization problem depends on this. This issue is discussed in Sect. 4 at the end of the manuscript.

Polynomial Updates: The polynomial update method is very similar to the linear update method. In this method, higher order derivatives are considered in the Taylor series. This higher order error approximation is then used to predict degree sequence errors. The Taylor series approximation of the Poisson distribution is given by Eq. 4.

$$poiss(z, x) = \frac{e^{-\mu} \mu^x}{x!} + \sum_{j=1}^{\infty} \left(\frac{\partial^j}{\partial \mu^j} poiss(\mu, z) \right) (z - \mu)^j \tag{4}$$

For a given number of derivatives k, a truncated series is used to make approximations. Note that the term on the left of the sum is an entry of the matrix $\mathbf{P}$(w) . Additionally, the right hand sum consists of two components, the jth derivative, and a difference term. This allows us to rewrite this expression in terms of matrices as in Eq. 5.

$$\mathbf{P}(w') \approx \mathbf{P}(w) + \sum_{j=1}^{k} \mathbf{U}_j \mathbf{E}_j \tag{5}$$

In Eq. 5, $\mathbf{U}_j$ is the matrix corresponding to the jth derivative of each column, similar to Eq. 2. $\mathbf{E}_j$ is a diagonal matrix with entries $E_j(a, a) = e_a^j$, corresponding to the step size in each dimension. In the polynomial update function, the error to be minimized is of the form $\gamma = \|(\mathbf{P}(w) + \sum_{j=1}^{k} \mathbf{U}_j \mathbf{E}_j)x - y\|_2$. Because of the increased accuracy of the polynomial method, a larger bound δ may be used for the step size. While we do not present bounds for this here, the size of δ can be chosen to be larger for larger instances of the number of derivatives k.

2.2 Maximum Likelihood Estimation

Maximum likelihood estimation (MLE) based clustering is a popular statistical method for determining probabilistic clusters for a data set [20]. Given a pre-defined type of statistical distribution (e.g. normal, binomial, Poisson, etc.) and a number of distributions m, MLE clustering determines the parameters and coefficients for those distributions such that their mixture distribution has the highest likelihood of generating the data set. While MLE is most commonly used for clustering data, we instead use it here for function approximation. Consider the desired degree sequence y as a realization of a mixture distribution and the underlying statistical distributions as Poisson distributions. Then, the coefficients and means which are output as a mixture model from MLE may be interpreted as the input vector x and the μ values in $\mathbf{P}(\mathrm{w})$, respectively.

Algorithm 2 MLE-Update (m, y, d, iters)

1: x $\leftarrow [\frac{1}{m}, \ldots, \frac{1}{m}]$
2: $p_y \leftarrow \frac{y}{\|y\|_1}$
3: $\mu \leftarrow [\frac{d}{m}, \frac{2d}{m}, \cdots, d]$
4: (x,μ) $\leftarrow$ poiss_EM(x,μ,p_y,iters)
5: x $\leftarrow \|y\|_1 x$
6: **return** (x,μ)

Our MLE based method proceeds as follows, and is demonstrated in pseudocode in Algorithm 2. Begin by considering a desired output sequence y, a number of means m, and an interval $[0, d]$. Initialize a vector $x = [\frac{1}{m}, \ldots, \frac{1}{m}]$ and a vector of means $\mu = [\mu_1 = \frac{d}{m}, \mu_2 = \frac{2d}{m}, \ldots, \mu_m = d]$. Note that these means may be initialized randomly within the interval $[0, d]$, if desired. Then, normalize y to obtain a probability distribution $p_y = \frac{y}{\|y\|_1}$, from which points are sampled for maximum likelihood estimation. Maximum likelihood estimation is then run on these inputs, updating the entries of x and μ at each iteration. Once this has concluded, x is scaled by $\|y\|_1$ and each entry is rounded to the nearest natural number. This ensures that x now corresponds to the number of nodes instead of a proportion of all nodes.

As discussed earlier, there are two parameters which may be tuned when improving Chung-Lu graph generation. While our earlier work focused on changing the input sequence, and both the linear and polynomial methods focus on changing the means of Poisson distributions, Algorithm 2 simultaneously solves for both. Additionally, expectation maximization has a tune-able dimensionality. This means that one may take small samples from p_y, and consider fewer Poisson distributions to improve compute time. This is not an option that is readily available in the case of greedy linear and polynomial updates.

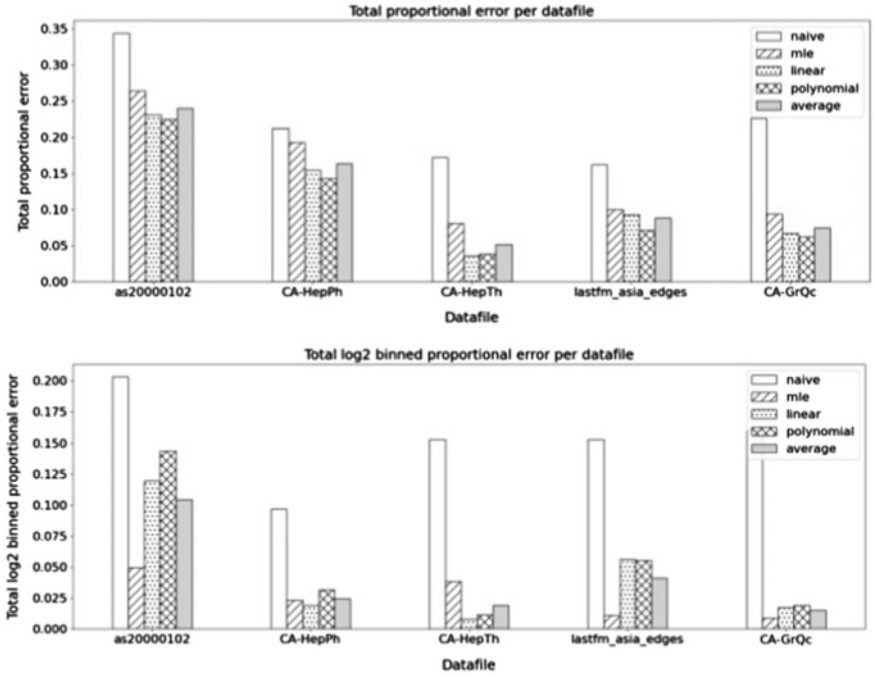


Fig. 2 Proportional errors: Degree error plots for all methods on a number of graphs. Both the proportional error (top), and $\log_2$-binned proportional error (bottom) metrics are as described in the Results section. As is seen, every method drastically reduces the proportional L1 error of the degree sequence when compared with naïve Chung Lu. However, different methods perform better on differing degree sequences

3 Results

In Fig. 2, the three methods discussed in the previous section are compared against naïve Chung-Lu generation on a set of degree sequences from the Stanford Large Network Dataset Collection. Graph generation is performed using the `expected_degree_graph(·)` function from the `NetworkX` [11] package in `Python`.

As can be seen, each method outperforms naïve Chung-Lu by a considerable margin. However, our different methods perform better on different degree sequences. The exact reason for this requires further analysis. Figure 2 considers two different proportional error functions. The first one is L1 proportional error which is computed as the ratio $\|CL(w,x) - y\|_1 / \|y\|_1$. This can be directly interpreted as the proportion of nodes which have the correct degree. Additionally, one can interpret this error function as a normalized version of the total variation distance. The $\log_2$-binned proportional error is also considered. In this case the sequences $CL(w,x)$ and y are partitioned into $b = \lceil log_2(d) \rceil$ bins, forming the sequences $\beta(CL(w,x))$ and $\beta(y)$, both of which are in $\mathbb{R}^b$. The entries of $\beta(CL(w,x))$ are

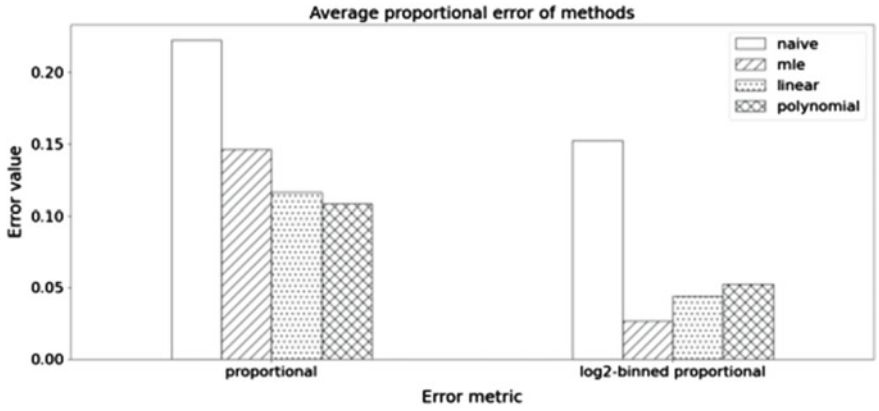


Fig. 3 Average proportional errors of methods: The proportional (left) and log2-binned proportional (right) errors are compared over all test graphs for each optimization method as well as naïve Chung-Lu. Both the proportional error, and $\log_2$-binned proportional error metrics are as described in the Results section. As is seen, on average the polynomial update method results in the more significant reduction of proportional error, however the MLE method results in the largest reduction in $\log_2$-binned proportional error

$\beta(CL(w,x))_i = \sum_{j=2^{(i-1)}}^{2^{(i-1)}+2^i} CL(w,x)_j$, and the entries of $\beta(y)$ follow similarly. The proportional binned error is then computed as $\|\beta(CL(w,x)) - \beta(y)\|_1 / \|\beta(y)\|_1$. The reason for defining this error function is that there are many applications where the exact degrees are less important than simply having the correct number of "low-degree", or "high-degree" nodes. For this purpose, the $\log_2$-binned proportional error provides a quantitative understanding of how many nodes are being generated for different "sections" of the sequence.

As is seen in Fig. 3, the polynomial update method outperforms the other optimization methods in proportional error. Additionally, the MLE optimization method outperforms the others for $\log_2$-binned proportional error. Conceptually, this implies that the polynomial update method may be the best at matching the degrees of nodes exactly, while the MLE method is superior for approximate reproduction of sequences.

4 Discussion

4.1 Parameters

When choosing parameters for Algorithm 1, a reader may be rightfully curious as to what constitutes a "good" choice. In Fig. 4, a parameter search over several choices of iteration number t and constant c, such that $x = cy$ are shown for two example graphs from the Stanford Large Network Dataset Collection. As is seen, the error

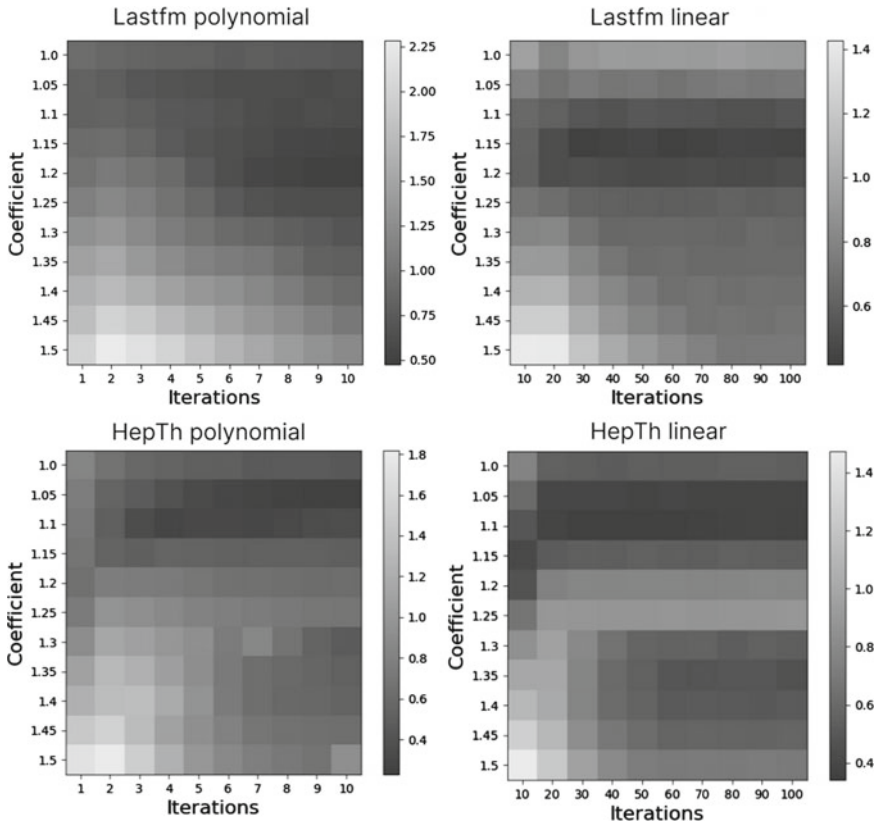


Fig. 4 A parameter search of error, varying the coefficient $c \in \mathbb{R}^+$ for $x = cy$, and the number of iterations for both the polynomial and linear update methods respectively. The polynomial update method in this case has $k = 2$. The colors indicate the proportional L1 error $\|CL(w, x) - y\|_1 / \|CL(y) - y\|_1$. As can be seen for the two sample graphs, the polynomial update method converges to a smaller proportional L1 error than the linear method does in the same number of iterations

reaches similar levels for both the polynomial and linear update methods for different parameters. We note that $1.05 < c < 1.15$ appears to work best for both graphs. While not shown, this behavior is also seen a across many other degree sequences. Furthermore, the number of iterations required to achieve a similar error reduction with polynomial updates versus linear updates is seen to be considerably smaller. In fact, for these two graphs, a similar error reduction is seen with an order of magnitude fewer update steps.

There is significant work to be done deciding parameters. While Fig. 4 suggests some best practices, it is far from definitive. Furthermore, the choice of step-size δ is currently somewhat arbitrary. In this paper, it is taken to be $0.05 \leq \delta \leq 0.2$ for linear updates, and $0.2 \leq \delta \leq 0.5$ for polynomial updates. Different step sizes drastically

alter the stability and number of requisite iterations of the method. This requires further experimental and theoretical results for varying degree sequences.

4.2 Timing Considerations

The methods presented in this manuscript require varying times to run. The linear update method uses a miniscule step size, and as such requires many iterations to terminate. This is a significant concern when the maximum degree of the desired output is large. This is because the maximum degree controls the dimensionality of the optimization step, which must be performed at every iteration. To this end, the polynomial update method can iterate with a larger step size, requiring less iterations. However, in the case of a significantly large maximum degree, the optimization step may still not be practical. The MLE-method does not suffer from these same drawbacks, because the sample number and number of distributions may be tuned. This means the MLE method should not perform slower on larger degree sequences, given constant sample and distribution numbers.

In the case of the greedy update methods, a simple change can be made which drastically speeds up compute time. This is the method of truncation. Note that, for most real world degree sequences the vast majority of the weight lies in the lowest degrees of the graph. Because of this, one may ignore a portion of the sequence when using either greedy update method. This drastically reduces compute time, but may introduce additional error. In our limited testing, removing the final 1% of the sequence by node count greatly improves run times and minimally affects error. Despite this, the best practice for truncation is an open problem.

5 Conclusion

In this manuscript, we presented two methods for improving the accuracy of Chung-Lu random graph generation. These methods consist of an iterative algorithm (Algorithm 1), which greedily updates the weights of nodes, and an algorithm (Algorithm 2) relying on maximum likelihood estimation. Both methods were shown to dramatically reduce degree sequence error in comparison to naïve Chung-Lu; however, they require different considerations. The greedy update methods suffer from long compute times in the case of sequences with high maximum degree, while the maximum likelihood method is significantly faster. While parameter choices for these algorithms are presented, a systematic study of their affect on resulting error is an avenue for further research.

References

1. Alam, M., Khan, M., Vullikanti, A., Marathe, M.: An efficient and scalable algorithmic method for generating large-scale random graphs. In: SC'16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. pp. 372–383. IEEE (2016)
2. Batagelj, V., Brandes, U.: Efficient generation of large random networks. Phys. Rev. E **71**(3), 036113 (2005)
3. Bollobás, B.: A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. Euro. J. Combin. **1**(4), 311–316 (1980)
4. Bonnans, J.F., Gilbert, J.C., Lemaréchal, C., Sagastizábal, C.A.: Numerical Optimization: Theoretical and Practical Aspects. Springer Science & Business Media (2006)
5. Brissette, C., Slota, G.M.: Limitations of Chung Lu random graph generation. In: International Conference on Complex Networks and Their Applications. pp. 451–462. Springer (2021)
6. Chung, F., Lu, L.: The average distances in random graphs with given expected degrees. Proc. Nat. Acad. Sci. **99**(25), 15879–15882 (2002)
7. Drobyshevskiy, M., Turdakov, D.: Random graph modeling: a survey of the concepts. ACM Comput. Surveys (CSUR) **52**(6), 1–36 (2019)
8. Durak, N., Kolda, T.G., Pinar, A., Seshadhri, C.: A scalable null model for directed graphs matching all degree distributions: in, out, and reciprocal. In: 2013 IEEE 2nd Network Science Workshop (NSW). pp. 23–30. IEEE (2013)
9. Fosdick, B.K., Larremore, D.B., Nishimura, J., Ugander, J.: Configuring random graph models with fixed degree sequences. SIAM Rev. **60**(2), 315–355 (2018)
10. Garbus, J., Brissette, C., Slota, G.M.: Parallel generation of simple null graph models. In: The 5th IEEE Workshop on Parallel and Distributed Processing for Computational Social Systems (ParSocial) (2020)
11. Hagberg, A., Swart, P., S Chult, D.: Exploring network structure, dynamics, and function using network. Tech. rep., Los Alamos National Lab. (LANL), Los Alamos, NM (United States) (2008)
12. Holland, P.W., Laskey, K.B., Leinhardt, S.: Stochastic blockmodels: first steps. Soc. Netw. **5**(2), 109–137 (1983)
13. Kolda, T.G., Pinar, A., Plantenga, T., Seshadhri, C.: A scalable generative graph model with community structure. SIAM J. Sci. Comput. **36**(5), C424–C452 (2014)
14. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., Alon, U.: Network motifs: simple building blocks of complex networks. Science **298**(5594), 824–827 (2002)
15. Molloy, M., Reed, B.: A critical point for random graphs with a given degree sequence. Random Struct. Algorithms **6**(2–3), 161–180 (1995)
16. Newman, M.E.: Modularity and community structure in networks. Proc. Nat. Acad. Sci. **103**(23), 8577–8582 (2006)
17. Slota, G.M., Berry, J., Hammond, S.D., Olivier, S., Phillips, C., Rajamanickam, S.: Scalable generation of graphs for benchmarking HPC community-detection algorithms. In: IEEE International Conference for High Performance Computing, Networking, Storage and Analysis (SC) (2019)
18. Slota, G.M., Garbus, J.: A parallel LFR-like benchmark for evaluating community detection algorithms. In: The 5th IEEE Workshop on Parallel and Distributed Processing for Computational Social Systems (ParSocial) (2020)
19. Winlaw, M., DeSterck, H., Sanders, G.: An in-depth analysis of the Chung-Lu model. Tech. rep., Lawrence Livermore National Lab. (LLNL), Livermore, CA (United States) (2015)
20. Zaki, M.J., Meira Jr, W., Meira, W.: Data Mining and Analysis: Fundamental Concepts and Algorithms. Cambridge University Press (2014)

Switching In and Out of Sync: A Controlled Adaptive Network Model of Transition Dynamics in the Effects of Interpersonal Synchrony on Affiliation

Sophie C. F. Hendrikse, Jan Treur, Tom F. Wilderjans, Suzanne Dikker, and Sander L. Koole

Abstract Interpersonal synchrony is associated with better interpersonal affiliation. No matter how well-affiliated people are, interruptions or transitions in synchrony rebound to occur. One might intuitively expect that transitions in synchrony negatively affect affiliation or liking. Empirical evidence, however, suggests that time periods with interruptions in synchrony may favor affiliation or liking even more than time periods without interruptions in synchrony. This paper introduces a controlled adaptive network model to explain how persons' affiliation might benefit from transitions in synchrony over and above mean levels of synchrony. The adaptive network model was evaluated in a series of simulation experiments for two persons with a setup in which a number of scenarios were encountered in different (time) episodes.

S. C. F. Hendrikse · T. F. Wilderjans · S. Dikker · S. L. Koole
Amsterdam Emotion Regulation Lab, Department of Clinical Psychology, Vrije Universiteit Amsterdam, Amsterdam, Netherlands
e-mail: s.c.f.hendrikse@vu.nl

T. F. Wilderjans
e-mail: t.f.wilderjans@fsw.leidenuniv.nl

S. L. Koole
e-mail: s.l.koole@vu.nl

S. C. F. Hendrikse · T. F. Wilderjans
Methodology and Statistics Research Unit, Institute of Psychology, Leiden University, Leiden, Netherlands

J. Treur (✉)
Social AI Group, Department of Computer Science, Vrije Universiteit Amsterdam, Amsterdam, Netherlands
e-mail: j.treur@vu.nl

T. F. Wilderjans
Research Group of Quantitative Psychology and Individual Differences, Faculty of Psychology and Educational Sciences, Katholieke Universiteit (KU) Leuven, Leuven, Belgium

S. Dikker
NYU—Max Planck Center for Language, Music and Emotion, New York University, New York, USA

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_7

Our controlled adaptive network model may serve as a foundation for more realistic virtual agents with regard to synchrony transitions and their role in affiliation.

Keywords Controlled adaptive network model · Interpersonal synchrony · Synchrony transitions · Liking · Affiliation

1 Introduction

When two or more individuals are interacting, their behavior tends to become mutually coordinated in time, or synchronized. Such interpersonal synchrony has been found to lead to behavioral adaptivity by enhancing, for example, closeness, concentration, cooperation, affiliation, alliance, connection, or bonding; e.g., [2, 19, 21, 29, 33, 37, 41]. However, perhaps counterintuitively, interruptions in interpersonal synchrony may positively affect behavioral adaptivity [12]. Furthermore, interpersonal liking is highest when a balance between interpersonal synchrony and complexity/unpredictability during a mirror game is obtained [30]. In this paper, we demonstrate how a controlled adaptive network model can computationally capture how behavioral adaptivity is induced not only by synchrony, but also by transitions of synchrony.

The controlled adaptive network model is based on an adaptive interplay of a number of mechanisms drawn from cognitive, behavioral, and affective neuroscience. For example, a neural basis for behavioral adaptivity in the form of adaptive affiliation can be found in recent work on the (nonsynaptic, intrinsic) adaptive excitability of (neural) states; e.g., [5, 8, 40, 42]. The extent of adaptation that a person requires may vary from situation to situation, which is called metaplasticity (e.g., [1, 31]). This controls the plasticity in a context-sensitive manner. The resulting network model yields a controlled adaptive network model.

The adaptive network model was evaluated in a series of simulation experiments for two persons with a setup in which a number of scenarios were encountered in different (time) episodes. The simulations included not only episodes with a common stimulus for the two persons, but also episodes with different stimuli for the persons. Moreover, to analyze the role of communication (interaction), circumstances were also included for episodes when communication was enabled by the environment and episodes when communication was not enabled.

2 Background Literature

Interpersonal synchrony usually leads to behavioral adaptivity in the form of mutual adaptation of interactive behavior; e.g., [2, 19–22, 26, 28, 33, 38, 39, 41; Fairhurst et al. 29]. For example, therapists were rated more favorably and as more empathic when, they were instructed to make their movements more synchronized with the

client [24, 29, 32, 36] found that initial movement synchrony between client and therapist was predictive of the client's experience of the quality of the therapeutic alliance at the end of each session. Moreover, Koole and Tschacher [21] reviewed converging evidence that movement synchrony has a positive effect on the working alliance between patient and therapist. Interpersonal synchrony in face-to-face interactions has been found to promote interpersonal alliance [11, 41].

However, it seems that more interpersonal synchrony is not always better for behavioral outcomes, such as good (working) relationships. A moderate range instead of a very low or very high behavioral synchrony between children and their parent is related to children who are more securely attached to their parent [3]. Another study [27] found a comparable pattern regarding the success of a therapy and movement synchrony. Patients who improved during therapy had a medium level of movement synchrony at the beginning of therapy, whereas patients who did not improve and consensually terminated their therapy had the highest level of movement synchrony at the beginning of therapy. Other findings indicating that high levels of synchrony do not always link to the best behavioral outcomes by Deres-Cohen et al. [9], suggest that therapists increase their levels of movement synchrony towards their patient to keep a strong alliance when difficulties occur.

Also, a more nuanced view of the link between interpersonal synchrony and behavioral adaptivity is found in several experimental paradigms with the mirror game, a task in which participants have the goal to move as synchronously as possible. On real world experimental data with the mirror game, Dahan et al. [6] have fitted multiple mathematical models. The models that only included moving in synchrony fit the real data worse than an alternative model that included a tendency to withdraw from synchrony (i.e., moving in and out of synchrony). As another example with the mirror game, Ravreby et al. [30] found that a combination of movement synchrony and interruptions by events with more complexity explains liking almost two times better than movement synchrony by itself does. Thus, in addition to synchrony, maintaining interest may be essential for bonding.

Partly inspired by the few available empirical studies, some conceptual studies have been published on the importance of both interpersonal synchrony and transitions in interpersonal synchrony with regard to other outcomes like affiliation. García and Di Paolo [12] propose that transitions in interpersonal synchrony play an important role, instead of only interpersonal synchrony itself, in the relation between movement synchrony and phenomena, such as therapeutic alliance. Similarly, Mayo and Gordon [25] claim that, to have an adaptive interpersonal system within a social context, people have two simultaneous tendencies: (1) to achieve interpersonal synchrony and (2) to switch in and out of interpersonal synchrony. The above empirical and conceptual literature on how interrupted synchrony can lead to stronger affiliation than 'just' synchrony was a main inspiration for the development of the controlled adaptive network model introduced here.

3 Self-Modelling Network Modelling

The presented controlled adaptive network model is based on network-oriented modelling. Following [34, 35], a temporal-causal network model uses nodes X and Y, also called states, with values $X(t)$ and $Y(t)$ over time t and the following characteristics:

- *Connectivity characteristics*

Connections from a state X to a state Y and their weights $\boldsymbol{\omega}_{X,Y}$

- *Aggregation characteristics*

For any state Y, some combination function $\mathbf{c}_Y(..)$ defines the aggregation that is applied to the impacts $\boldsymbol{\omega}_{X,Y}X(t)$ on Y from its incoming connections from states X

- *Timing characteristics*

Each state Y has a speed factor $\boldsymbol{\eta}_Y$ defining how fast it changes for given causal impact

The following difference (or related differential) equations that are used for simulation purposes and also for analysis of temporal-causal networks, incorporate these network characteristics $\boldsymbol{\omega}_{X,Y}$, $\mathbf{c}_Y(..)$ and $\boldsymbol{\eta}_Y$ in a standard numerical format:

$$Y(t + \Delta t) = Y(t) + \boldsymbol{\eta}_Y[\mathbf{c}_Y(\boldsymbol{\omega}_{X_1,Y}X_1(t), \ldots, \boldsymbol{\omega}_{X_k,Y}X_k(t)) - Y(t)]\Delta t \qquad (1)$$

for any state Y and where X_1 to X_k are the states from which Y gets its incoming connections. Within the software environment described in [34, Chap. 9], a large number of currently around 65 useful basic combination functions are included in a combination function library. The above concepts make it possible to design network models and their dynamics in a declarative manner, based on mathematically defined functions and relations. The examples of combination functions that were applied in the model introduced here can be found in the Appendix (as Linked Data at https://www.researchgate.net/publication/361435085). In Table 1, the new transition detection functions are shown. Here, W is a representation of the sliding windows.

Realistic network models are usually adaptive: often not only their states but also some of their network characteristics change over time. By using a *self-modelling network* (also called a *reified* network), a similar network-oriented conceptualization can also be applied to adaptive networks to obtain a declarative description using mathematically defined functions and relations for them as well; see [34, 35]. This works through the addition of new states to the network (called *self-model states*) which represent (adaptive) network characteristics. In the graphical 3D-format as shown in Sect. 4, such additional states are depicted at a next level (called *self-model level* or *reification level*), where the original network is at the *base level*.

For instance, the weight $\boldsymbol{\omega}_{X,Y}$ of a connection from state X to state Y can be represented (at a next self-model level) by a self-model state named $\mathbf{W}_{X,Y}$. Similarly,

Table 1 The new transition detection combination functions used in the introduced network model

	Notation	Formula	Parameters
Average transition	$\mathbf{transdetavabs}_{\boldsymbol{\iota\delta},\boldsymbol{\sigma\omega}}(W)$	$\lvert\text{mean}_{1\leq \boldsymbol{\nu}\leq \boldsymbol{\sigma\omega}}(W(\boldsymbol{\iota\delta}, \boldsymbol{\nu})) - \text{mean}_{\boldsymbol{\sigma\omega}+\mathbf{1}\leq \boldsymbol{\nu}\leq 2\boldsymbol{\sigma\omega}}(W(\boldsymbol{\iota\delta}, \boldsymbol{\nu}))\rvert$	$\boldsymbol{\iota\delta}$ state identifier $\boldsymbol{\sigma\omega}$ sliding window size
Maxmin transition	$\mathbf{transdetmaxminabs}_{\boldsymbol{\iota\delta},\boldsymbol{\sigma\omega}}(W)$	$\lvert \max_{1\leq \boldsymbol{\nu}\leq \boldsymbol{\sigma\omega}}(W(\boldsymbol{\iota\delta}, \boldsymbol{\nu})) - \min_{\mathbf{1}\leq \boldsymbol{\nu}\leq \boldsymbol{\sigma\omega}}(W(\boldsymbol{\iota\delta}, \boldsymbol{\nu}))\rvert$	$\boldsymbol{\iota\delta}$ state identifier $\boldsymbol{\sigma\omega}$ sliding window size
Standard deviation transition	$\mathbf{transdetstdev}_{\boldsymbol{\iota\delta},\boldsymbol{\sigma\omega}}(W)$	$2\sqrt{[\text{mean}_{1\leq \boldsymbol{\nu}\leq \boldsymbol{\sigma\omega}}(W(\boldsymbol{\iota\delta}, \boldsymbol{\nu}) - \text{mean}(W))^2]}$	$\boldsymbol{\iota\delta}$ state identifier $\boldsymbol{\sigma\omega}$ sliding window size

All Greek letters bold and also the words in the column under notation

all other network characteristics from $\boldsymbol{\omega}_{X,Y}$, $\mathbf{c}_Y(..)$ and $\boldsymbol{\eta}_Y$ can be made adaptive by including self-model states for them. For example, an adaptive excitability threshold $\boldsymbol{\tau}_Y$ (e.g., [5, 8, 42]) for a logistic combination function for state Y can be represented by a self-model state named $\mathbf{T}_Y$ and an adaptive speed factor $\boldsymbol{\eta}_Y$ can be represented by a self-model state named $\mathbf{H}_Y$.

As the outcome of such a process of network reification is also a temporal-causal network model itself, as has been shown in [34, Chap. 10], this self-modelling network construction can easily be applied iteratively to obtain multiple orders of self-models at multiple (first-order, second-order, …) self-model levels. For example, a second-order self-model may include a second-order self-model state $\mathbf{H}_{\mathbf{T}_Y}$ representing the speed factor $\boldsymbol{\eta}_{\mathbf{T}_Y}$ for the dynamics of first-order self-model state $\mathbf{T}_Y$ which in turn represents the adaptative excitability threshold $\boldsymbol{\tau}_Y$ for Y. In the current paper, this multi-level self-modelling network perspective is applied to obtain a second-order adaptive network architecture addressing controlled adaptation induced by detected synchrony and detected synchrony transitions. The control level is used to make the adaptation speed context-sensitive as addressed by the metaplasticity literature such as [1, 31]: the metaplasticity principle 'Adaptation accelerates with increasing stimulus exposure' formulated by [31] can easily be modelled by using second-order self-model states, this is discussed in Sect. 4.

4 The Adaptive Network Model

In this section, we introduce an adaptive neural agent model covering the detection of interpersonal synchrony and transitions of interpersonal synchrony and its related behavioral adaptivity. It applies a self-modelling network architecture of three levels: a base level, a first-order self-model level, and a second-order self-model level. The (middle) first-order self-model level models how excitability of states at the base

level are adapted over time, and the (upper) second-order self-model level models the control over this adaptation.

4.1 Base Level

Figure 1 shows a graphic overview of the base level of the person model (persons are indicated by the big boxes) and the above mentioned Appendix provides explanations for all of its states. Each person uses six states for the interaction with the other person: three states (indicated by sense) for sensing the other person on the left-hand side of each box, and three states for execution or expression of actions (move, exp_affect, talk) on the right-hand side. Within a box the person's internal mental states can be found, outside the boxes are the world states. Note that the ws-states are sensed by one agent and determined by another agent, while the state world_s represents an external world state that is independent from the agents. Each person also senses its own actions, modelled by the arrows from right to left outside the box.

The internal mental states cover sensory representation states (rep) and preparation states (prep) for each of the three modalities: movement *m*, expression of affect *b*, and verbal action *v*. In addition, each person has a conscious emotion state for affective response *b* (cons_emotion). Each of the mentioned states is depicted in Fig. 1 by a light pink circle shape. For each modality, the corresponding representation state has an outgoing response connection to the related preparation state and it has an incoming (prediction) connection back from that preparation state to model internal mental simulation [7, 18].

Six synchrony detector states are depicted in Figs. 1 and 2 by the darker pink diamond shapes: (1) three of them for intrapersonal synchrony for the three pairs of the three modalities movement—emotion (*m*-*b*), movement—verbal action (*m*-*v*), emotion—verbal action (*b*-*v*), and (2) the other three for interpersonal synchrony for each other three modalities [15]. In addition, we introduce three interpersonal

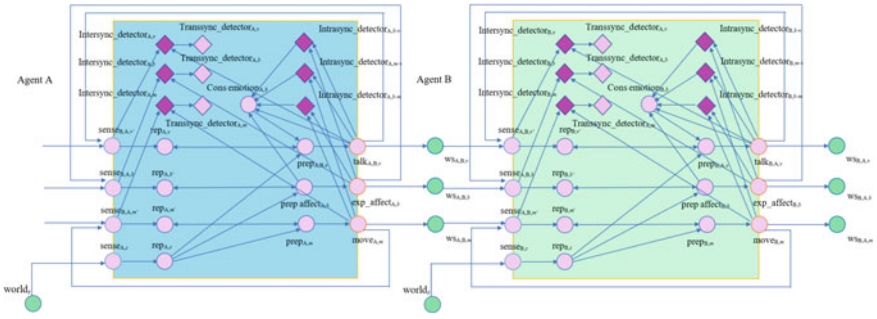


Fig. 1 Base level of the introduced adaptive network model with three modalities, six synchrony detection states for intrapersonal and interpersonal synchrony (dark pink diamonds) and three states for interpersonal synchrony transition detection (light pink diamonds)

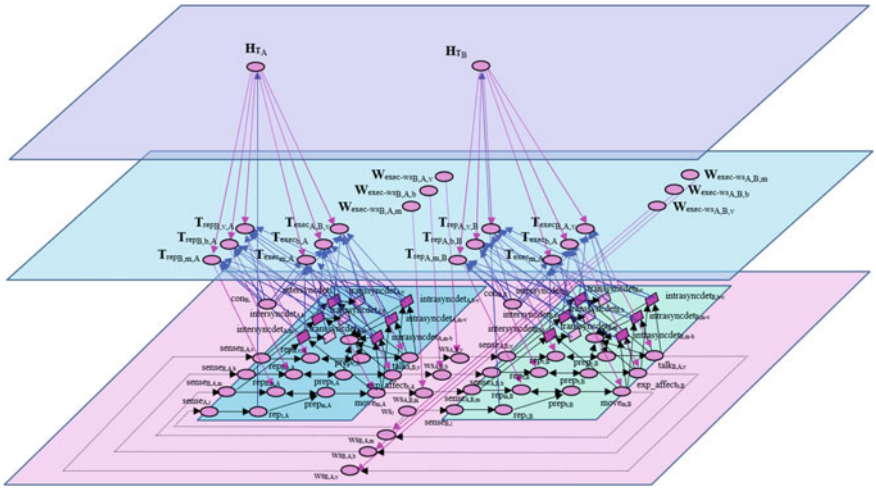


Fig. 2 Overview of the overall second-order adaptive network model, the purple plane represents the second-order self-model level (upper), the blue plane the first-order self-model level (middle) and the pink plane the base level (lower)

synchrony transition detector states (the light purple diamonds) that are able to note transitions in the synchrony.

Following (Ravreby et al. [30]), we assume that both the detected interpersonal synchrony and the detected transitions of interpersonal synchrony contribute to behavioral adaptivity of each person: detected synchrony and synchrony transition lead to becoming more sensitive to sensing a person and to expressing to that person (a form of homeostatic regulation). This is modelled through self-model **T**-states for adaptive excitability thresholds [5, 8, 42].

Thus, more synchrony and synchrony transition detection will enhance excitability for the representation and execution states. More sensitive states for representations will lead to having better images of the modalities of the other person, which will make the sensed signals better available and accessible for the brain. More sensitive states for execution will lead to better expressed own modalities, so that the other person will sense them better. In Sect. 4.2, we discuss in more detail how we modelled this behavioral adaptivity and its control.

At the base level some world states (world_s) are modelled for stimuli s that are sensed by the persons. The world's suitability for enabling communication between the two persons is modelled by **W**-states. Two general context states are included to model the conditions to maintain excitability thresholds represented by **T**-states.

4.2 Modelling Controlled Adaptation

Following what has been described in Sects. 3 and 4.1, the behavioral adaptivity is modelled by first-order self-model states $\mathbf{T}_Y$ for adaptation of the excitability thresholds $\boldsymbol{\tau}_Y$ for the internal representation states Y and execution states Y, covering the three considered modalities (movement, affective response and verbal action). In addition, second-order self-model $\mathbf{H}_\mathbf{T}$-states are used for context-sensitive control of the adaptation of these adaptive excitability thresholds $\boldsymbol{\tau}_Y$ for the internal representation states and execution states Y.

Figure 2 shows a 3D picture of the overall design of the model; here, the first-order self-model states are in the middle (blue) plane and the second-order self-model states in the upper (purple) plane. By changing the activation values of the **T**-states, the corresponding excitability thresholds change accordingly [4, 5, 42]. This change occurs due to the influences from the detected synchronies and synchrony transitions, modelled by the upward (blue) arrows in Fig. 2 to the **T**-states in the middle plane.

The two second-order self-model states $\mathbf{H}_{\mathbf{T}_A}$ and $\mathbf{H}_{\mathbf{T}_B}$ model context-sensitive excitability adaptation control, one for each person. They represent the adaptation speed for the excitability for the concerning person according to the second-order adaptation (or metaplasticity) principle 'Adaptation accelerates with increasing stimulus exposure' [31]. Therefore, they have incoming connections (blue upward arrows from base plane to upper plane) from the stimulus representation states at the base level.

In the Appendix, the full specification of this network model by role matrices and explanations for all of its states can be found.

5 Simulation Results

In this section, we study how our controlled adaptive network model behaves during an experimental setting with different episodes. In this experimental setting, it was manipulated whether or not person A and B each received the same stimulus and whether or not they were able to communicate, in such a way that each condition happened. Person A was always stimulated for 120 time units, followed by 60 time units of no stimulus (180 time units in total), and thereafter this process was repeated. Regarding person B, the first 60 time units no stimulus was present, followed by 180 time units with the stimulus (i.e., 240 time units in total), and this (non-)stimulus episode of 240 time units was thereafter repeated. The communication enabling conditions in the environment are indicated by the self-model states $\mathbf{W}_{\text{exec-ws}_{x,\,A,B}}$ (from A to B) and the states $\mathbf{W}_{\text{exec-ws}_{x,\,B,A}}$ (from B to A). They are activated to activation value 1 from time 30 to time 60 and then repeated every 60 time units; when they are not activated, they have activation value 0 (from time 0 to time 30 and then repeated every 60 time units). Limit cycles in the results and repetition of conditions occur after a certain amount of time. We will depict the simulations until 360 time units

were reached with a step size (Δt) of 0.5, meaning that our shown simulation run contains 720 computational steps. In Figs. 3 and 4 examples of simulation outcomes are shown. All initial state values can be consulted in the Appendix (Table 2).

5.1 Comparing Time Intervals with Transitions and Time Intervals Without Transitions

Since the focus of this paper is the role of intrapersonal synchrony, interpersonal synchrony and interpersonal synchrony transitions in relation to affiliation, we highlight the results of these processes in the controlled adaptive network model. The results of the other states in the model can be consulted in the Appendix that can be found as Linked Data at https://www.researchgate.net/publication/361435085. Within a simulation, in different time periods different phenomena can be observed. First of all, in the period 0–30 mainly startup phenomena can be observed. After that, the following more specific patterns can be observed.

- If a person does not get a stimulus and communication is not enabled then this results in a low level of detected subjective intrapersonal synchrony for that person; e.g., for person A in Fig. 3a, time period 120–150; when communication is enabled with just one person having a stimulus, then there is increasing intrapersonal synchrony for both persons
- Large values for interpersonal synchrony occur in periods of enabled communication, even when a person has no stimulus; e.g., Fig. 3b, time 30–60
- When the stimulus is absent for a person, then the **H**-state (not depicted) for that person becomes 0 and therefore the related **T**-states of that person show flat lines for: e.g., Fig. 3d, time 120–180 for person A
- When no communication is enabled, then sensing becomes 0 and when the stimulus is absent also actions become 0 as well; therefore, interpersonal synchrony increases; e.g., Fig. 3b, time 240–270 (person B); note that this does not really hold at time 120–150 (only a small effect of movement only of person A)
- When detected interpersonal synchrony goes from high to low, then a peak in the transition follows results in lower **T**-states due to the transitions and not due to the interpersonal synchrony; e.g., Fig. 3b–d, time 180–210
  Oppositely, when detected interpersonal synchrony goes from low to high, then a peak in the transition follows resulting in lower **T**-states due to both the transitions and the interpersonal synchrony; e.g., Fig. 3b–d, time 90–120
- The lowest values for the **T**-states are reached when both interpersonal synchrony and transition are high; e.g., Fig. 3b–d, time 90–120, time 210–240

As shown in Fig. 3, for each person the activation of intrapersonal synchrony detection seems to rely heavily on the stimulus the person itself receives. Moreover, in time intervals in which transitions occur the **T**-states reach lower values than in time intervals where only synchrony occurs; this means stronger behavioral adaptivity in

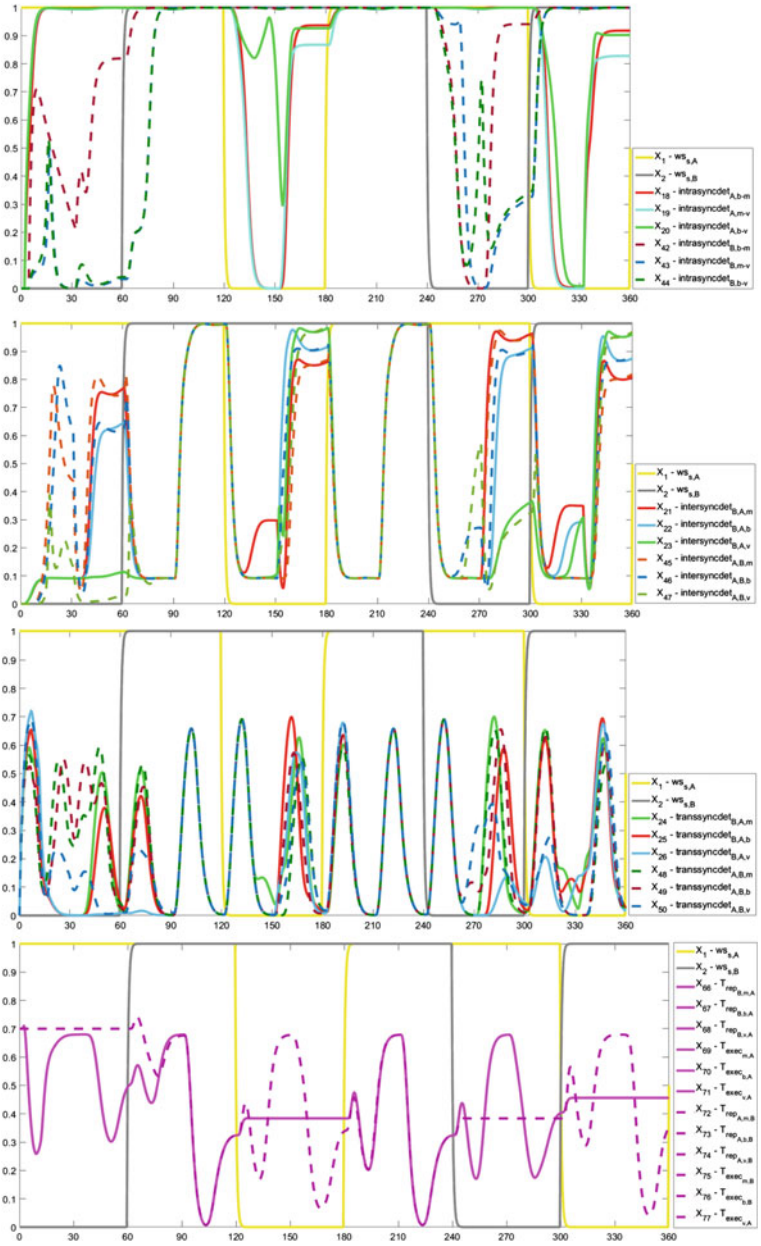


Fig. 3 The trajectory over the stimuli episodes of **a** the intrapersonal synchrony detector states (first), **b** the interpersonal synchrony detector states (second), **c** the interpersonal synchrony transition detector states (third), **d** the affiliation states when both interpersonal synchrony and interpersonal synchrony transitions are connected with the affiliation (fourth). The horizontal axis represents time and the vertical axis represents the activation levels (from 0 to 1)

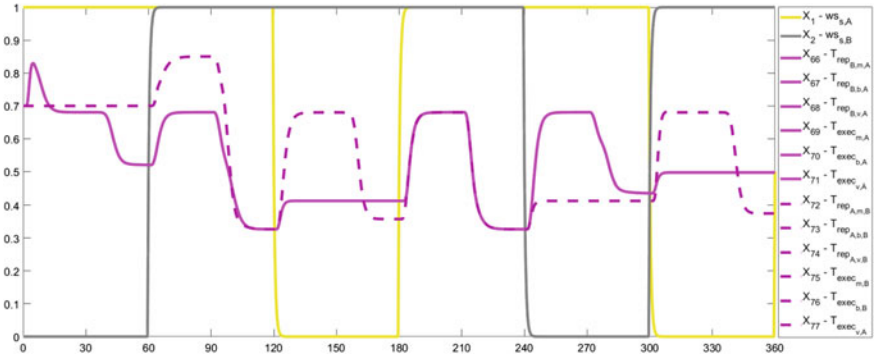


Fig. 4 The trajectory over the stimuli episodes of the affiliation states when only interpersonal synchrony (and not interpersonal synchrony transitions) is connected with the affiliation. The horizontal axis represents time and the vertical axis represents the activation levels (from 0 to 1)

Table 2 Stimuli and communication for the first 4 episodes

Type	Stimulus person A, B	Communication enabled	Duration	Time intervals
Episode 1	Different	No	30	0–30
Episode 2	Common	Yes	30	30–60
Episode 3	Different	No	30	60–90
Episode 4	Common	Yes	30	90–120

the former type of time intervals. So, overall, the results are in accordance with findings reported in literature such as [30].

5.2 Comparing Continuous Synchrony Without Transitions with Synchrony with Transitions

In Fig. 4 the simulation results of the **T**-states (regarding other simulation results, see Appendix) are shown in case that the connections from the transition detectors to the **T**-states are disabled. For both persons, the excitability thresholds represented by the **T**-states reach lower values in the case with enabled transition detectors in Fig. 3, which means a stronger short-term behavioral adaptivity. Moreover, the peaks of the **T**-states in Fig. 3 (transition detector states enabled) last for less time units than those in Fig. 4, likely due to the effect of the enabled transition detector states that change at a higher frequency.

Table 3 Average behavioral adaptivity through excitability thresholds (represented by **T**-states) for enabled and disabled transition detection over 1080 time units

Average excitability threshold	Person A	Person B	Person A and B
Enabled transition detection	0.394	0.400	0.397
Disabled transition detection	0.507	0.526	0.517
Difference % enabled of disabled (%)	22	24	23

To quantify the precise role of the transition detector states, Table 3 displays the specific averages of the adaptive excitability thresholds over 1080 time units for both simulations in Figs. 3 and 4. It turns out that the transitions make that the excitability thresholds are 20–25% lower, which is substantial: it makes activation much easier. This once more is in accordance with findings reported in the literature such as [30].

6 Discussion

Within psychotherapy sessions, more interpersonal synchrony usually leads to a better therapeutical affiliation. However, in practice also switching in and out of interpersonal synchrony often occurs, which breaks synchrony for some period of time. Given that in general interpersonal synchrony positively affect affiliation, a reasonable expectation would be that transitions in synchrony negatively influence affiliation. In contrast, it has been put forward [12, 25, 30] that such transitions also positively affect affiliation and do this even to such an extent that time periods with interpersonal synchrony interrupted by transitions may positively affect affiliation or liking more than time periods with synchrony without such transitions. This paper has introduced a controlled adaptive network model that addresses this effect.

Computational modelling of interpersonal synchrony was already addressed in earlier work such as [13–17]. However, in the models described in [13, 14], no (subjective) internal detection of synchrony takes place. Furthermore, in [14] no adaptivity was covered, whereas in [13] another type of adaptivity was incorporated, namely of internal connections from representation states to preparation states. As far as we know, [15–17] describe the only publications on other computational models where subjective synchrony detection is addressed in relation to affiliation. A difference is that in these publications no subjective detection of synchrony transitions was addressed, which is the novelty introduced by the current paper. The controlled adaptive network model introduced in the current paper has adopted part of the model of [16] as point of departure but has extended it by detector states for interpersonal synchrony transitions and their effect on behavioral adaptivity. In this way, the introduced controlled adaptive network model was obtained for the way in which detected interpersonal synchrony transitions also lead to different types of behavioral adaptivity concerning affiliation between two persons. The model can provide a basis to develop adaptive virtual agents that are able to concentrate on each other by

short-term behavioral adaptivity in a human-like manner not only during periods of interpersonal synchrony but also during periods in which from time to time transitions of the interpersonal synchrony occur.

References

1. Abraham, W.C., Bear, M.F.: Metaplasticity: the plasticity of synaptic plasticity. Trends Neurosci. **19**(4), 126–130 (1996)
2. Accetto, M., Treur, J., Villa, V.: An adaptive cognitive-social model for mirroring and social bonding during synchronous joint action. In: Proceedings of the 9th International Conference on Biologically Inspired Cognitive Architectures, BICA'18, vol. 2; Proc. Comput. Sci. **145**, 3–12. https://doi.org/10.1016/j.procs.2018.11.002
3. Beebe, B., Steele, M.: How does microanalysis of mother—infant communication inform maternal sensitivity and infant attachment? Attach. Hum. Dev. **15**(5–6), 583–602 (2013). https://doi.org/10.1080/14616734.2013.841050
4. Bloch, C., Vogeley, K., Georgescu, A.L., Falter-Wagner, C.M.: INTRApersonal synchrony as constituent of INTERpersonal synchrony and its relevance for autism spectrum disorder. Front. Robot. AI **73** (2019)
5. Chandra, N., Barkai, E.: A non-synaptic mechanism of complex learning: modulation of intrinsic neuronal excitability. Neurobiol. Learn. Mem. **154**, 30–36 (2018)
6. Dahan, A., Noy, L., Hart, Y., Mayo, A., Alon, U.: Exit from synchrony in joint improvised motion. PLoS ONE **11**(10), e0160747 (2016). https://doi.org/10.1371/journal.pone.0160747
7. Damasio, A.R.: The Feeling of What Happens: Body and Emotion in the Making of Consciousness. Houghton Mifflin Harcourt (1999)
8. Debanne, D., Inglebert, Y., Russier, M.: Plasticity of intrinsic neuronal excitability. Curr. Opin. Neurobiol. **54**, 73–82 (2019)
9. Deres-Cohen, K., Dolev-Amit, T., Peysachov, G., Ramseyer, F.T., Zilcha-Mano, S.: Nonverbal synchrony as a marker of alliance ruptures. Psychotherapy **58**(4), 499–509 (2021)
10. Fairhurst, M.T., Janata, P., Keller, P.E.: Being and feeling in sync with an adaptive virtual partner: Brain mechanisms underlying dynamic cooperativity. Cereb. Cortex. 23(11), 2592–2600 (2013). https://doi.org/10.1093/cercor/bhs243
11. Feldman, R.: Parent–infant synchrony biological foundations and developmental outcomes. Curr. Dir. Psychol. Sci. **16**, 340–345 (2007). https://doi.org/10.1111/j.1467-8721.2007.00532.x
12. García, E., Di Paolo, E.A.: Embodied coordination and psychotherapeutic outcome: beyond direct mappings. Front. Psychol. **1257** (2018)
13. Hendrikse, S.C.F., Kluiver, S., Treur, J., Wilderjans, T.F., Dikker, S., Koole, S.L.: How virtual agents can learn to synchronize: an adaptive joint decision-making model of psychotherapy. Cogn. Syst. Res. (2022a)
14. Hendrikse, S.C.F., Treur, J., Wilderjans, T.F., Dikker, S., Koole, S.L.: On the same wavelengths: emergence of multiple synchronies among multiple agents. In: Proceedings of the 22nd International Workshop on Multi-Agent-Based Simulation, MABS'21. Lecture Notes in AI, vol. 13128, pp. 57–71. Springer Nature (2022b)
15. Hendrikse, S.C.F., Treur, J., Wilderjans, T.F., Dikker, S., Koole, S.L.: On becoming in sync with yourself and others: an adaptive agent model for how persons connect by detecting intra- and interpersonal synchrony, under submission. In: Proc. of the 2022 Joint Conference on Robotics and AI, JCRAI'22. See also: In sync with yourself and with others: detection of intra- and interpersonal synchrony within an adaptive agent model. In: Face2face: Advancing the Science of Social Interaction. Royal Society, London (2022c). https://www.researchgate.net/publication/358964043

16. Hendrikse, S.C.F., Treur, J., Wilderjans, T.F., Dikker, S., Koole, S.L.: On the interplay of interpersonal synchrony, short-term affiliation and long-term bonding: a second-order multi-adaptive neural agent model. In: Maglogiannis et al. (eds.) Proceedings of the 18th International Conference on Artificial Intelligence Applications and Innovations, AIAI'22. Advances in Information and Communication Technology, vol. 646, pp. 37–57. Springer Nature (2022d)
17. Hendrikse, S.C.F., Treur, J., Wilderjans, T.F., Dikker, S., Koole, S.L.: Becoming attuned to each other over time: a computational neural agent model for the role of time lags in subjective synchrony detection and related behavioral adaptivity. In: Proceedings of the 15th International Conference on Brain Informatics, BI'22. Lecture Notes in AI, vol 13406, pp. 369–383. Springer Nature (2022e)
18. Hesslow, G.: Conscious thought as simulation of behaviour and perception. Trends Cogn. Sci. **6**, 242–247 (2002)
19. Hove, M.J., Risen, J.L.: It's all in the timing: interpersonal synchrony increases affiliation. Soc. Cogn. **27**(6), 949–960 (2009)
20. Kirschner, S., Tomasello, M.: Joint music making promotes prosocial behavior in 4-year-old children. Evol. Hum. Behav. **31**, 354–364 (2010). https://doi.org/10.1016/j.evolhumbehav.2010.04.004
21. Koole, S.L., Tschacher, W.: Synchrony in psychotherapy: a review and an integrative framework for the therapeutic alliance. Front. Psychol. **7**, 862 (2016)
22. Koole, S.L., Tschacher, W., Butler, E., Dikker, S., Wilderjans, T.F.: In sync with your shrink. In: Forgas, J.P., Crano, W.D., Fiedler, K. (eds.) Applications of Social Psychology, pp. 161–184. Taylor and Francis, Milton Park (2020)
23. Lisman, J., Cooper, K., Sehgal, M., Silva, A.J.: Memory formation depends on both synapse-specific modifications of synaptic strength and cell-specific increases in excitability. Nat. Neurosci. **2018**(21), 309–314 (2018)
24. Maurer, R.E., Tindall, J.H.: Effect of postural congruence on client's perception of counselor empathy. J. Counseling Psychol. **30**, 158 (1983). https://doi.org/10.1037/0022-0167.30.2.158
25. Mayo, O., Gordon, I.: In and out of synchrony—behavioral and physiological dynamics of dyadic interpersonal coordination. Psychophysiol. **57**(6), e13574 (2020)
26. Palumbo, R.V., Marraccini, M.E., Weyandt, L.L., Wilder-Smith, O., McGee, H.A., Liu, S., Goodwin, M.S.: Interpersonal autonomic physiology: a systematic review of the literature. Pers. Soc. Psychol. Rev. **21**(2), 99–141 (2017)
27. Paulick, J., Deisenhofer, A.-K., Ramseyer, F., Tschacher, W., Boyle, K., Rubel, J., Lutz, W.: Nonverbal synchrony: a new approach to better understand psychotherapeutic processes and drop-out. J. Psychother. Integr. **28**(3), 367–384 (2018). https://doi.org/10.1037/int0000099
28. Prince, K., Brown, S.: Neural correlates of partnered interaction as revealed by cross-domain ALE meta-analysis. Psychol. Neurosci. **15**(1), 1–13 (2022). https://doi.org/10.1037/pne0000282
29. Ramseyer, F., Tschacher, W.: Nonverbal synchrony in psychotherapy: coordinated body movement reflects relationship quality and outcome. J. Consult. Clin. Psychol. **79**, 284–295 (2011). https://doi.org/10.1037/a0023419a
30. Ravreby, I., Shilat, Y., Yeshurun, Y.: Liking as a balance between synchronization, complexity and novelty. Sci. Rep. **12**(1), 1–12 (2022). https://doi.org/10.1038/s41598-022-06610-z
31. Robinson, B.L., Harper, N.S., McAlpine, D.: Meta-adaptation in the auditory midbrain under cortical influence. Nat. Commun. **7**, e13442 (2016)
32. Sharpley, C.F., Halat, J., Rabinowicz, T., Weiland, B., Stafford, J.: Standard posture, postural mirroring and client-perceived rapport. Couns. Psychol. Q. **14**, 267–280 (2001). https://doi.org/10.1080/09515070110088843
33. Tarr, B., Launay, J., Dunbar, R.I.M.: Silent disco: dancing in synchrony leads to elevated pain thresholds and social closeness. Evol. Hum. Behav. **37**(5), 343–349 (2016)
34. Treur, J.: Network-Oriented Modeling for Adaptive Networks: Designing Higher-Order Adaptive Biological, Mental and Social Network Models. Springer Nature (2020a)
35. Treur, J.: Modeling multi-order adaptive processes by self-modeling networks (keynote speech). In: Tallón-Ballesteros, A.J., Chen, C.-H. (eds.) Proceedings of the 2nd International Conference

on Machine Learning and Intelligent Systems, MLIS'20. Frontiers in Artificial Intelligence and Applications, vol. 332, pp. 206–217. IOS Press (2020b)
36. Trout, D.L., Rosenfeld, H.M.: The effect of postural lean and body congruence on the judgment of psychotherapeutic rapport. J. Nonverbal Behav. **4**, 176–190 (1980)
37. Tschacher, W., Ramseyer, F., Koole, S.L.: Sharing the now in the social present: duration of nonverbal synchrony is linked with personality. J. Pers. **86**(2), 129–138 (2018)
38. Valdesolo, P., DeSteno, D.: Synchrony and the social tuning of compassion. Emotion **11**, 262 (2011). https://doi.org/10.1037/a0021302
39. Valdesolo, P., Ouyang, J., DeSteno, D.: The rhythm of joint action: synchrony promotes cooperative ability. J. Exp. Soc. Psychol. **46**(4), 693–695 (2010)
40. Williams, A.H., O'Leary, T., Marder, E.: Homeostatic regulation of neuronal excitability. Scholarpedia **8**, 1656 (2013)
41. Wiltermuth, S.S., Heath, C.: Synchrony and cooperation. Psychol. Sci. **20**(1), 1–5 (2009)
42. Zhang, A., Li, X., Gao, Y., Niu, Y.: Event-driven intrinsic plasticity for spiking convolutional neural networks. IEEE Trans. Neural Netw. Learn. Syst. (2021). https://doi.org/10.1109/tnnls.2021.3084955

Uniformly Scattering Neighboring Nodes of an Ego-Centric Network on a Spherical Surface for Better Network Visualization

Emily Chao-Hui Huang and Frederick Kin Hing Phoa

Abstract Ego-centric networks are an important class of networks to represent a particular node's connections to its neighbors. This work aims at providing an efficient method to represent an ego-centric network so that all neighboring nodes are scattered on the surface of the unit sphere uniformly. Such uniformity is not just a simple space-filling distribution with maximum Euclidean distance among nodes, but with the consideration of existing edges among these nodes and without overlapping of node clusters. Our proposed method is a three-step method that partitions the spherical surface associated to a criterion on the edge-to-node ratio, then scatters the nodes on the respective subspace according to the relationship between nodes and modularity. To compute efficiently, the particle swarm optimization method is employed in all three steps to allocate the respective points. We show the connection between our space-filling distribution of points on a spherical surface to the minimum energy design on a two-dimensional flat plane with a specific gradient. We provide a demonstration on allocating nodes of an ego-centric network of 50 nodes, and some distance statistics show the good performance of our method when compared to four state-of-the-art methods via self-organizing maps and force-driven approaches.

Keywords Ego-centric networks · Space-filling · Particle swarm optimization · Modularity · Minimum energy design

1 Introduction

The analysis of large-scale networks has grown in importance since the turn of the millenium. The structure of a network helps to describe the relationship among individuals in the network, and its wide applications include anthropology, biomedical research, communication studies, and social sciences. Among all networks with spe-

E. C.-H. Huang
National Tsing Hua University, Hsinchu, Taiwan

F. K. H. Phoa (✉)
Institute of Statistical Science, Academia Sinica, Taipei, Taiwan
e-mail: fredphoa@stat.sinica.edu.tw

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_8

cial structure, an ego-centric network is an important class to represent a particular node's connections to its neighbors. The traditional presentation of an ego-centric network is given as a two-dimensional circular form, where the center node is located at the center of the circle and all neighboring nodes are located on the circle's perimeter. Although this presentation is simple to draw, the existence of edges between neighboring nodes greatly reduces the visualization quality of the network, not mentioning if node clusters exist. Rather than a two-dimensional circle with limited space to allocate neighboring nodes, a three-dimensional sphere not only provides additional space for node allocations, its extra dimension allows edges between neighboring nodes to be drawn with possibly fewer overlaps. This brings up the research question of this work: How to allocate neighboring nodes "uniformly" on a spherical surface with the consideration of the existence of edges between nodes and node clusters?

Distributing points uniformly on a spherical surface is a well-known problem. It was first proposed in [1], which tried to determine equilibrium configurations of electrons constrained to the surface. The spiral method was first proposed to distribute points over a surface. For example, [2] investigated the energy of allocations of N points on a sphere, and constructed a suitable partition with N parts of equal areas to obtain bounds of extremal energy. Inspired by the phyllotaxis, [3] proposed a fast and effective approach called the Fibonacci grid, which optimizes the packing efficiency by the mapping from golden spiral, and each point on the surface represent almost the same area. Some mathematical works also introduced the spherical t-design [4] to allocate points uniformly for numerical integration with equal weights. It was defined by a set of points locating on an unit spherical surface if the integral of any polynomial of degree almost t is equal to the average value of the polynomial over the set of points. However, all above methods are appropriate for allocating independent points, which is hardly fulfilled when we consider network nodes as the points.

There are many recent algorithms being developed for drawing a network on a spherical surface. Fu et al. [5] extended the self-organizing map (SOM) algorithm to layout email network on a spherical surface. SOM is an unsupervised artificial neural network training approach via competitive learning as a substitution of backpropagation with gradient descent to minimize the loss function. It is useful in visualizing network clusters, especially in small-word network, but its overlapping between nodes and edges lowers the quality in network visualization. Thus, [6] proposed a two-stage SOM algorithm, where the next stage after the original SOM is a circular layout algorithm to adjust the node positions to prevent overlapping of nodes and edges. Other than SOM-type algorithm, [7] proposed a stochastic neighbor embedding method called Doubly Stochastic Neighbor Embedding on Spheres (DOSNES). It overcomes the problem of crowd nodes with highly imbalanced data by a novel normalization method.

In fact, point allocation on a surface is called the space-filling problem in statistics and experimental design, computer experiments in particular. The separation distance of an experimental design D^N is its minimal pairwise Euclidean distance $\rho(D) = \min_{x,y\in D}\left\{\sum_{k=1}^{N}(x_k - y_k)^2\right\}$. Johnson et al. [8] introduced a popular space-filling

design that achieves the greatest separation distance $\rho(D)$. Readers who are interested in space-filling designs on a flat and regular space are referred to [9].

In this work, we propose a new approach to allocate the neighboring nodes of an ego-centric network on a spherical surface that is uniform with the consideration of the existence of edges and node clusters. Notations and definitions are provided in Sect. 2. Section 3 describes our method and algorithm, together with the connection to the minimum energy design. We show the performance via numerical simulations in Sect. 4 and a discussion is given in the last section.

2 Notations and Definitions

We consider an undirected ego-centric network $G(V, E)$ with a set of nodes V, which includes a center node denoted as v_{00}, and a set of edges (E) between pairs of nodes. We define a cluster in G as connected components of the maximal subgraph with vertex set $V \setminus \{v_{00}\}$. Let k be the number of clusters and we denote $\{C_1, \ldots, C_k\}$ as these k clusters with sizes $|C_i| = c_i$ for $i = 1, \ldots, k$. Note that our definition of node cluster may differ from traditional definitions of network community, and the method of detecting communities is not the main focus of this work. A brief review of the literature of community detection can be found in [10]. Here we assume to use a simple node cluster method, but if network communities are detected in prior, they can be implemented accordingly.

For all neighboring nodes in the node cluster, we denote them as v_{ij} for $i = 1, \ldots, k$ and $j = 1, \ldots, c_i$. For all remaining neighboring nodes with degree 1, we denote them as v_{0j} for $j = 1, \ldots, N$, and $N = |V| - \sum_i c_i - 1$. For each cluster C_i, we denote E_i as the set of edges in C_i with size $|E_i| = e_i$ for $i = 1, \ldots, k$. Let A be the $(|V| - 1) \times (|V| - 1)$ adjacency matrix of $V \setminus \{v_{00}\}$ with each element $a_{st} = 1$ if $(v_{is}, v_{it}) \in E_i$ and 0 otherwise for $s, t = 1, \ldots, (|V| - 1)$.

In practice, we recast all nodes v_{ij} and all clusters C_i in the form of spherical caps. Every nodes v_{ij} is located on the apex of a spherical cap characterized by a cone angle θ_i^j and a solid angle $\Omega_i^j = 2\pi(1 - \cos\theta_i^j)$. Every cluster C_i can also be similarly characterized with a different solid angle Ω_i.

To define optimality on uniform node allocation, we first define the distance between two points v_{ij} and $v_{ij'}$ on a unit sphere with the center point v_{00} at the origin of coordinate as the angle $\phi_{jj'} = cos^{-1}\left(|v_{ij} \cdot v_{ij'}|\right)$, for $i = 0, \ldots, k$. Then the uniform allocation is optimal if they maximize the minimum of $\phi_{jj'}$ for all nodes v_{ij} and clusters C_i in $G(V, E)$. Note that the distance between a pair of nodes in a cluster needs to be adjusted by the cluster's edge density. Conventionally, a simple measure of edge density is given by the Beta index $\beta = \frac{\text{total number of edges}}{\text{total number of nodes}}$, but it is equal to zero for a single point and it causes zero-denominator problem in the first step of our three-stage optimization. Therefore, we define an adjusted Beta index $\beta_i = \frac{e_i + c_i}{c_i}$ to represent the degree of connectivity of C_i.

3 Method

3.1 Preliminaries

Particle Swarm Optimization. Space-filling distribution is an NP-hard problem, so we suggest to use the Particle Swarm Optimization (PSO) [11] for its computational efficiency. PSO has been widely used in computational intelligence, industrial optimization, and many engineering problems. It starts with an initial set of particles randomly assigned in the solution domain. These particles are iteratively updated for quality improvements by its two best particles: the personal best, $pBest$, and global best, $gBest$. The personal best for each individual particle indicates the one with the best location that the particle has ever visited, and the global best for all particles is the best one among all personal best particles. The update of particle i at iteration t is conducted via the velocity $\mathbf{w}_i(t)$ update and the position $X_i(t)$ update: $\mathbf{w}_i(t+1) = \omega \times \mathbf{w}_i(t) + a_1\gamma_1(pBest - X_i(t)) + a_2\gamma_2(gBest - X_i(t))$ and $X_i(t+1) = X_i(t) + V_i(t+1)$, where γ_1 and γ_2 are random numbers in $(0, 1)$, constants ω, a_1, and a_2 are predefined parameters.

The PSO algorithm we use in our method is the standard version as in [11]. Each particle is a vector $\mathbf{v}$ recording the positions of node on the spherical surface. The objective function is $f(\mathbf{v}) = -\min_{\mathbf{v}\in V}\left\{\phi_{jj'}\right\}$. It aims to find the optimal $\mathbf{v}$ before the maximum iteration reached or the objective change is less than 10^{-8}.

Eigenvector of Modularity. Modularity is a common tool to detect network communities. A node cluster with high modularity implies a high connection probability between nodes within a cluster while low connection probability for nodes in the cluster to connect to nodes outside cluster. The standard definition of modularity [12] is $Q = \sum_{i=1}^{N}\sum_{j=1}^{N}\left(a_{ij} - \frac{k_i k_j}{2m}\right)\mathbf{1}_{i,j\in S}$, where a_{ij} is an element of the $N \times N$ adjacency matrix A, k_i, k_j are degrees of the vertices, $m = \frac{1}{2}\sum_{i=1}^{N} k_i$, and S stands for the same group. To improve the efficiency, [13] proposed the modularity-based spectral approach that helps to approximate the partition problem using the eigenvector of modularity matrix. The modularity matrix B is a real symmetric matrix with elements $B_{ij} = a_{ij} - \frac{k_i k_j}{2m}$, which the sign of elements of the leading eigenvector divide all nodes in this communities to two different groups. By implementing the same algorithm over the newly formed communities, the network can be partitioned to small clusters until all the communities are indivisible.

3.2 Three-Stage Optimization

We consider an ego-centric network that consists of clusters and scattered points among the neighboring nodes. In order to allocate these neighboring nodes uniformly

on a spherical surface, we propose a three-stage optimization algorithm to solve this maximin distance problem.

Step 1: Allocation of Cluster Center. Every cluster is viewed as points with different weights in this step. We aim to allocate the connected nodes closer than the unconnected ones, so a smaller polar angle is expected to a cluster with a higher edge density. The weight of each cluster C_i is defined by a weight function $w(c_i, e_i) = \frac{c_i}{\beta_i} = \frac{c_i^2}{e_i + c_i}$ and the corresponding proportion is $r_i = w_i(c_0 + \sum_{i=1}^{k} w(c_i, e_i))^{-1}$. This implies that the solid angle of C_i is $\Omega_i = 4\pi r_i$ and its corresponding polar angle $\theta_i = \cos^{-1}(1 - 2r_i)$. Then this step aims at maximizing the following criterion: $f(p_i, p_i') = \min_{i \neq i', i, i' \in \{1,\ldots,k\}} \frac{\phi_{ii'}}{\theta_i + \theta_{i'}}$, where $\phi_{ii'}$ is angle between cluster center p_i and $p_{i'}$. We employ the PSO algorithm to solve this optimization problem.

Step 2: Allocation of Nodes within Every Cluster. All nodes that belong to their respective clusters are scattered within the limit of the spherical cap formed in Step 1. Because a latent community structure may exist within clusters, we suggest to use the leading eigenvector method to detect them in the beginning of Step 2. To avoid overfitting that creates tiny fragments, a user-defined small number of communities is allowed. Within each community, we use a circular sector to constrain the locations of communities in the cluster spherical cap. For communities with only one point, they are allocated in the same circular sector. Suppose there are M communities in cluster C_i, the angle of circular section m is defined as $\alpha_m = 2\pi w_m (\sum_{m=1}^{M} w_m)^{-1}$, where w_m is the weight function mentioned in step 1. Inside each circular sector, all angles of node pairs $\min_{j \neq j'} \phi_{jj'}$ are maximized.

Step 3: Allocation of Remaining Nodes outside Clusters. All remaining nodes outside clusters are degree 1 that connects only to the center node. They are allocated to the spherical surface that maximizes the angle between pairs of nodes. We also avoid the overlap between the spherical caps of these nodes and those of the clusters' spherical caps.

Regarding the overlap of the spherical cap, [14] proved that for n points on a unit sphere, the angle between any two points $\phi \leq \sin^{-1} \frac{\sqrt{4 - \csc^2\left(\frac{n\pi}{6(n-2)}\right)}}{2}$. A failure to satisfy this constraints for the polar angle of clusters in Step 1 results in an overlap of the spherical caps, then we consider the set difference of spherical caps as the new constraint region instead of the whole circular sector, thus the resulting allocation is slightly less uniform but simpler to compute than uniformity.

3.3 A Connection to Minimum Energy Designs

Minimum energy design (MED) is a space-filling design with the domain gradient following a given function instead of being uniform [15]. The basic idea of MED is to allocate more points on the more important regions than other less important regions when a prior knowledge on the functional of the solution domain is given.

A criterion to optimize MED is defined as: $\max_{D} \min_{i,j} \frac{d(x_i,x_j)}{q(x_i)q(x_j)}$, where $d(x_i, x_j)$ is the Euclidean distance between points x_i and x_j, $q(x) = \frac{1}{f(x)^{1/(2p)}}$ is the charge function and $f(x)$ is a desired density function.

For the space-filling problem in the previous subsection, the node allocation on the three-dimensional spherical surface can be viewed as a two-dimensional MED on a flat surface with a specific gradient. Specifically, the Lambert cylindrical equal area projection is used as the gradient. Consider a point (a, b, c) with $a^2 + b^2 + c^2 = 1$ on a unit spherical surface, it can be mapped to $(x, y) \in R^2$ according to the transformation equations: $a = \cos(x)\sqrt{1 - y^2}$, $b = \sin(x)\sqrt{1 - y^2}$, and $c = y$. On the two-dimensional space, the distance of high-latitude region is elongated. Therefore, the amount of points on high-latitude region should be less than those on low-latitude region. The gradient function is given by $f(x, y) = \cos(\sin^{-1} y)$. Roshan Joseph et al. [15] proved that the limiting distribution of points in MED converges to uniform distribution for an arbitrary density function. Based on the one-to-one projection from the spherical surface space-filling problem to MED with our specific projection as gradient, the distribution of points calculated by our method is also uniform.

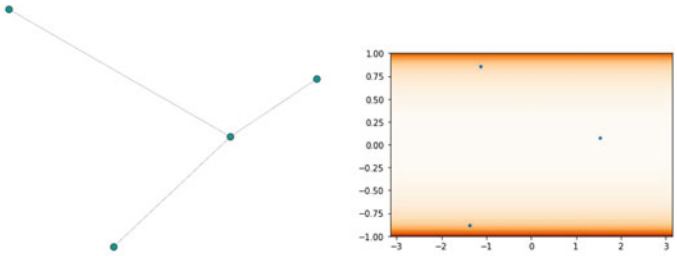
3.4 A Demonstration

We demonstrate how our three-step method uniformly allocates 49 neighboring nodes of an ego-centric network on a spherical surface. This ego-centric network is generated randomly ourselves and for verification purpose, we include the network structure in the appendix. In addition to the step-by-step illustration of the three-dimensional network on the left of Fig. 1, we also include their corresponding two-dimensional MED side-by-side on the right.

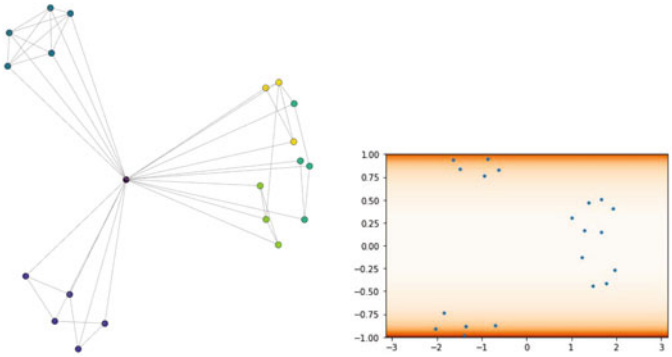
First, there are three node clusters in the network, which are a 5-node clique, a 5-node pentagon, and a 10-node subgraph. Step 1 identifies the spherical caps of three clusters and Fig. 1a allocates the apex of these caps. In Step 2, five nodes of the clique and five nodes of the pentagon are allocated to their respective spherical caps. For ten nodes of the subgraph, our method further identifies three separate communities among these ten nodes. Figure 1c shows the allocated positions of these 20 nodes. The remaining 29°C-1 nodes are allocated to the open area outside the clusters' spherical caps, and Fig. 1e shows the node allocations of all 49 neighboring nodes. The point allocation of the respective MED in Fig. 1f looks uniform.

4 Performance Comparison

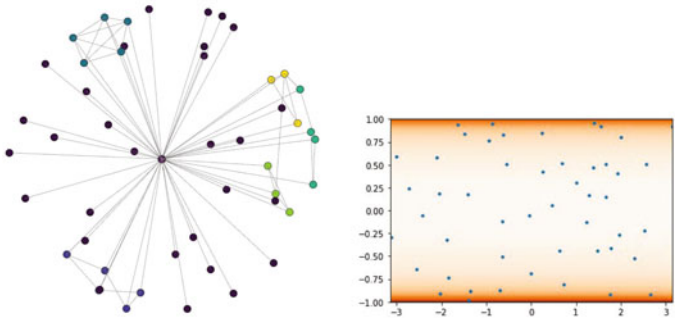
We compare the performance of our method with state-of-the-art methods in the literature, including SOM [5], two-step SOM [6], Schulz's method [16], and DOSNES



(a) Step 1. Our Method (b) Step 1. MED



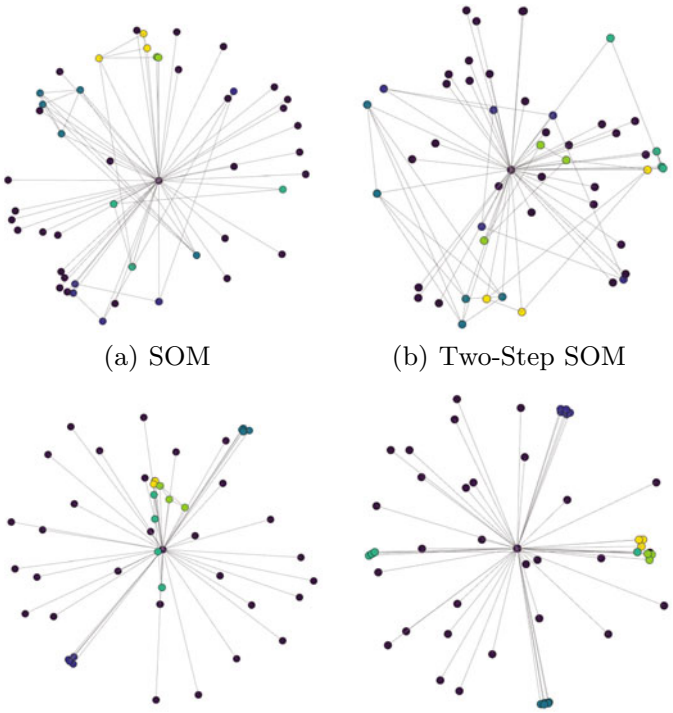
(c) Step 2. Our Method (d) Step 2. MED



(e) Step 3. Our Method (f) Step 3. MED

Fig. 1 Step-by-step result

[7]. The first two are good self-organizing map algorithms and the last two are good force-directed algorithms. The test network is the same ego-centric network as in the demonstration above. The simulation repeats 10 times for each of the five methods. We use the solid angle of single nodes Ω_0 to evaluate the degree of uniformity of node positions on the spherical surface. Ω_0 can be estimated by the solid angle of all nodes in this network multiplying their corresponding Beta index, thus



(a) SOM (b) Two-Step SOM

(c) Christian Schulz (d) DOSNES

Fig. 2 Results from four state-of-the-art methods

$\hat{\Omega}_0 = \frac{1}{|V|-1}\sum_{i=0}^{k}\sum_{j=1}^{c_i}\beta_i\hat{\Omega}_i^j = \frac{1}{|V|-1}\sum_{i=0}^{k}\sum_{j=1}^{c_i}(\hat{\Omega}_0)_i^j$, where $(\hat{\Omega}_0)_i^j$ is the solution of $\phi = \cos^{-1}(1-\frac{\Omega_0}{2\pi\beta_i}) + \cos^{-1}(1-\frac{\Omega_0}{2\pi\beta_p})$, ϕ is the angle between v_{ij} and its nearest point v_{pq}. The variance of $\hat{\Omega}_0$ indicates the degree of uniform allocation.

Figure 2 shows the results obtained from the four state-of-the-art methods, and the result of our method is in Fig. 1e. Table 1 provides the statistics of Ω_0 of each node allocations by five methods. SOM is highly efficient in terms of computational time but it fails to allocate the nodes that belong to the same clusters to the surrounding positions, not mentioning the node overlap that appears as the minimum Ω_0 is close to 0. Its two-step variant improves the overlapping situation by adding an extra step for node separation. Although its mean Ω_0 is also improved from 0.0746 to 0.0911, its variance (i.e. distance variation among node pairs) and the computational time increase substantially. The two force-directed algorithms suffer the same problem of node overlap as SOM, but their mean Ω_0 are at the same par as two-step SOM and their variance Ω_0 are lower than that of SOM. This indicates that the two force-directed methods achieve better uniformity than SOM in general.

Compared to the four methods, our method successfully achieves the highest minimum Ω_0, the highest mean Ω_0 and the lowest variance Ω_0. This implies that the

Table 1 Performance comparison of five methods

Method	Ours	SOM	Two-step SOM	Schulz	DOSNES
Time (s)	113.982	4.769	359.040	23.291	1.092
Minimum ($\hat{\Omega}_0$)	0.0986	8.73×10^{-5}	0.0015	9.41×10^{-5}	0.0001
Mean ($\hat{\Omega}_0$)	0.1898	0.0746	0.0911	0.0991	0.0874
Variance ($\hat{\Omega}_0$)	0.1261	0.3912	0.6462	0.2939	0.2469

distance between two closest nodes and the average distance among all node pairs are both maximized while the distance variation among node pairs are minimized, indicating a good uniformity on the node distance and thus a uniform node allocation. The only weakness of our method is the computational time. It requires additional works to perfect the program codes to achieve lower computational time.

5 Conclusion

In this article, we develop a new method to scatter the neighboring nodes of an ego-centric network on a spherical surface, and the node allocation is uniform with the consideration of edges and node clusters. Our method is a three-step optimization process optimized via PSO. Given a close connection to the minimum energy design, we ensure that the node allocation is uniformly distributed on the spherical surface. We compare our result with other network visualization methods, showing that our method can generate a spherical-looking network with a better uniform distribution of nodes, although the computational time of our method still needs further improvement via better code writing. Besides, it is possible to have other algorithms for allocating nodes on a spherical surface that we do not include in this paper. We will compare them in the extended version of this paper. In addition, there are better metaheuristic algorithms other than PSO to handle discrete optimization problem, like the swarm intelligence based (SIB) method [17, 18]. Since this SIB method is employed to the search of optimal MED in [19], it has a great potential to be applied to the search of optimal node allocation in our work. Finally, it is efficient to use modularity for an ego-centric network with only distance-1 neighboring nodes, and it faces challenges when we add neighboring nodes with distances greater than 1. Such scenario commonly exists in many large-scale networks like scientific networks [20, 21]. A remedy to this obstacle is to use another metric that considers neighboring nodes with larger distances, like the scan statistics [22] and its generalized version [23], or many others.

Acknowledgements This work was supported by Academia Sinica (Taiwan) Thematic Project grant number AS-TP-109-M07 and the Ministry of Science and Technology (Taiwan) grant numbers 107-2118-M-001-011-MY3 and 111-2118-M-001-007-MY2.

Appendix

For the network in the demonstration, Node 01 is the center node that connects to all other 49 nodes. Here is the remaining edge list for the network: 04-06, 04-11, 05-07, 05-10, 05-12, 05-14, 06-09, 07-10, 07-12, 07-14, 08-09, 08-11, 10-12, 10-14, 12-14, 21-22, 21-23, 21-24, 21-29, 22-23, 24-31, 28-29, 28-30, 29-30, 31-32, 32-33.

References

1. Thomson, J.J.: XXIV. On the structure of the atom: an investigation of the stability and periods of oscillation of a number of corpuscles arranged at equal intervals around the circumference of a circle; with application of the results to the theory of atomic structure. London Edinburgh Dublin Philos. Mag. J. Sci. **7**, 237–265 (1904)
2. Rakhmanov, E.A., Saff, E.B., Zhou, Y.M.: Minimal discrete energy on the sphere. Math. Res. Lett. **1**, 647–662 (1994)
3. Swinbank, R., Purser, J.: Fibonacci grids: a novel approach to global modelling. Q. J. R. Meteorol. Soc. **132**, 1769–1793 (2007)
4. Delsarte, P., Goethals, J., Seidel, J.J.: Spherical codes and designs. Geometriae Dedicata **6**, 363–388 (1977)
5. Fu, X., Hong, S., Nikolov, N.S., Shen, X., Wu, Y., Xuk, K.: Visualization and analysis of email networks. In: 2007 6th International Asia-Pacific Symposium on Visualization, pp. 1–8 (2007)
6. Wu, Y., Takasuka, M.: Visualizing multivariate network on the surface of a sphere. In: Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation, vol. 60, pp. 77–83 (2006)
7. Lu, Y., Corander, J., Yang, Z.: Doubly stochastic neighbor embedding on spheres. Pattern Recogn. Lett. **128**, 100–106 (2019)
8. Johnson, M.E., Moore, L.M., Ylvisaker, D.: Minimax and maximin distance designs. J. Stat. Plann. Inference **26**, 131–148 (1990)
9. Roshan Joseph, V.: Space-filling designs for computer experiments: a review. Qual. Eng. **28**, 28–35 (2016)
10. Sun, W.H., Phoa, F.K.H.: Network community detection via an improved swarm intelligence approach. In: Advances in Swarm Intelligence, vol. 13344, pp. 419–431. Springer (2022)
11. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995)
12. Newman, M.E.J.: Modularity and community structure in networks. Proc. Nat. Acad. Sci. **103**, 8577–8582 (2006)
13. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E **74**, 036104 (2006)
14. Fejes, L.: Über eine Abschätzung des kürzesten Abstandes zweier Punkte eines auf einer Kugelfläche liegenden Punktsystems. Jahresbericht der Deutschen Mathematiker-Vereinigung **53**, 66–68 (1943)
15. Roshan Joseph, V., Dasgupta, T., Tuo, R., Wu, C.F.J.: Sequential exploration of complex surfaces using minimum energy designs. Technometrics **57**, 64–74 (2015)
16. Schulz, C.: Visualizing spreading phenomena on complex networks. arXiv:1807.01390 (2018)
17. Phoa, F.K.H., Chen, R.B., Wang, W.C., Wong, W.K.: Optimizing two-level supersaturated designs via swarm intelligence techniques. Technometrics **58**, 43–49 (2016)
18. Phoa, F.K.H.: A swarm intelligence based (SIB) method for optimization in designs of experiments. Nat. Comput. **16**, 597–605 (2017)
19. Phoa, F.K.H., Tsai, T.C.: A two-step approach to the search of minimum energy designs via swarm intelligence. In: Advances in Swarm Intelligence, vol. 12145, pp. 37–45. Springer (2020)

20. Chang, L.L.N., Phoa, F.K.H., Nakano, J.: A new metric for the analysis of the scientific article citation network. IEEE Access **7**, 132027–132032 (2019)
21. Chang, L.L.N., Phoa, F.K.H., Nakano, J.: A generative model of article citation networks of a subject from a large-scale citation database. Scientometrics **126**, 7373–7395 (2021)
22. Wang, T.C., Phoa, F.K.H.: A scanning method for detecting clustering pattern of both attribute and structure in social networks. Phys. A **445**, 295–309 (2016)
23. Wang, T.C., Phoa, F.K.H.: A generalized framework for detecting social network communities by the scanning method. In: Complex Networks and Their Applications, vol. 881, pp. 250–261. SCI (2020)

The Hyperbolic Geometric Block Model and Networks with Latent and Explicit Geometries

Stefano Guarino, Enrico Mastrostefano, and Davide Torre

Abstract In hyperbolic geometric networks the vertices are embedded in a latent metric space and the edge probability depends on the hyperbolic distance between the nodes. These models allows to produce networks with high clustering and scale-free degree distribution, where the coordinates of the vertices abstract their centrality and similarity. Based on the principles of hyperbolic models, in this paper we introduce the Hyperbolic Geometric Block Model, which yields highly clustered, scale-free networks while preserving the desired group mixing structure. We additionally study a parametric network model whose edge probability depends on both the distance in an explicit euclidean space and the distance in a latent geometric space. Through extensive simulations on a stylized city of 10K inhabitants, we provide experimental evidence of the robustness of the HGBM model and of the possibility to combine a latent and an explicit geometry to produce data-driven social networks that exhibit many of the main features observed in empirical networks.

Keywords Urban social network · Graph model · Simulator · Hyperbolic geometric graph · Data-driven

1 Introduction and Background

Defining accurate models for real-world social networks is instrumental in several research fields, e.g., in sociology [1], epidemiology [2] or marketing [3]. Dynamic processes, such as the spread of a disease or a rumour, can be represented on appropriate networks that encode patterns of connection and interaction among individuals

S. Guarino · E. Mastrostefano · D. Torre (✉)
Institute for Applied Mathematics "Mauro Picone", CNR, Rome, Italy
e-mail: davidetorre92@gmail.com

D. Torre
University Campus Bio-Medico of Rome, Rome, Italy

LUISS Guido Carli, Rome, Italy

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_9

in a population. The structure of the network has a direct impact on the process [4], e.g., the topology of urban social networks, their size and demography, can affect disease spreading [5] in and within cities [6]. The efforts towards a deeper understanding of the mechanisms underlying the formation of real world networks have led to the development of a number of network models, mostly driven by the desire to reproduce—and possibly explain—specific observed features of such complex networks [7].

A recent line of research builds on the intuition that the vertices of the network can be embedded into a hidden metric space [8], so that notions of centrality and homophily in the network find a direct counterpart in the position and proximity of the vertices in this space. Assuming a hyperbolic latent geometry, rather than an euclidean geometry, allows to generate networks with a high clustering, a scale-free degree distribution and, possibly, a soft community structure [9].

Computational social sciences require network models that encode real data and empirical findings about the socio-demographic and geographic features of the considered population. Age and geographic distance emerged as two critical factors in guiding the formation of social ties [10, 11]. This led to the definition of data-driven spatial social network models [12–14] that rely upon the wide availability of spatial density data [15] and age-based mixing patterns deduced from census and/or survey data [16].

The aim of this study is to investigate ways to endow data-driven network models with desirable topological properties, thanks to a latent hyperbolic geometry. The Geometric Block Model (GBM), proposed in [17], generalizes the Stochastic Block Model (SBM) embedding the vertices in an euclidean metric space and considering a different connectivity threshold for each possible block pair. Including group mixing in a hyperbolic setting is not equally straightforward, because of the interplay between the rules governing the distribution of the vertices in the hyperbolic space and the rules determining whether two vertices must be connected based on their hyperbolic distance. To the best of our knowledge, there is no previous model that considers both a latent and an explicit metric space, making edge probabilities dependent on the distances computed in both spaces.

Among the class of hyperbolic geometric graph models, we focus on the 0-temperature model, which is the one guaranteeing the stronger transitivity. We first propose, in Sect. 2, the Hyperbolic Geometric Block Model (HGBM), where a different hyperbolic latent space is considered for each group pair in order to guarantee the desired group mixing. We then analyze, in Sect. 3, the topological features of a parametric model obtained as a linear combination of the HGBM with the spatially-explicit USN model proposed in [13, 18]. We simulate both models for a stylized urban population, with age-based mixing patterns inferred from survey data for Italy and distance-based mixing adjusted according to previous empirical findings. We provide experimental evidence of the robustness of our HGBM model and of the possibility to obtain suitable data-driven social networks by combining a latent and

an explicit geometry. All software used in this paper is available as open-source under the GPLv3.[1]

1.1 Related Work

A number of random graph models, proposed across decades of research, have been widely used in computational social sciences. Ideally, the models should reproduce the main features of real-world social networks, well summarized in [19]. These networks show a heavy-tailed (e.g., lognormal) degree distribution, often with a finite cutoff in agreement with Dunbar's number. The transitivity of the networks is high, compared to a random graph model, as a consequence of the well-established principle that "friends of my friends are my friends." Moreover, they show positive assortativity by degree and *type*.

Defining simple models that capture all of these features is not an easy task. Models designed to mimic the scale-free degree distribution emerging in many real networks, for instance, may fail to yield the expected clustering structure [20, 21]. Exponential random graphs have been shown to overcome some of these limitations [22, 23].

Recently, network instances having suitable features have been generated by means of the so-called *random geometric* models [8, 24, 25], where the popularity and similarity of the nodes depend on their position in some *latent* metric space [26]. The distance function chosen for this space impacts on the properties of the obtained network. Embedding the vertices into a hyperbolic disk [8] has proved a way to obtain both high clustering and heavy-tailed degree distribution.

In real social networks, individuals tend to socialize with their peers [27]. Among other aspects, such as education or economy, age emerged as a critical element in the formation of social ties [10, 11], possibly thanks to the availability of age-related data at different spatial scales [15]. Another widely studied type of homophily is spatial proximity, which gives rise to the so-called *spatial networks*. Most authors considered variations of well known random network models obtained by embedding the vertices in a metric space. The imposed spatial constraints influence the topological properties of the network [28] and the imposed penalty on "long" edges causes the spatial distribution of the vertices to impact on clusters, path lengths, degree distribution, and more [29].

Stochastic Block Models (SBM) are commonly used for generating networks with a known community structure [30, 31], which is a typical feature in the presence of some homophily principle. In this type of networks the nodes are partitioned into disjoint sets named *blocks* and the probability of an edge existing between two nodes

[1] Both the HGBM model and the parametric model presented in Sect. 3 are included in the USN package at https://gitlab.com/cranic-group/usn; the HGBM model is also released as a standalone software at https://gitlab.com/cranic-group/hgbm.

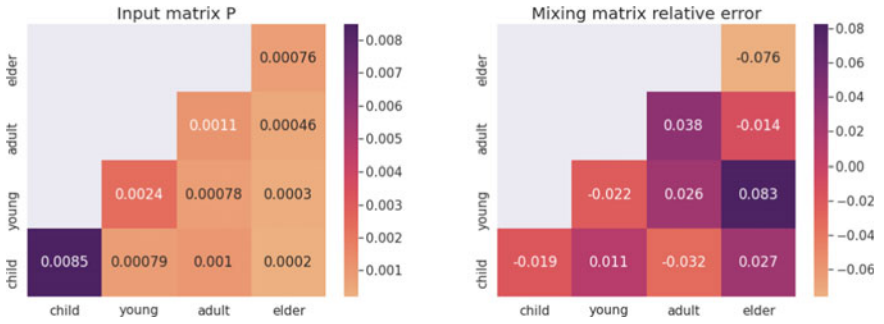
depends on the blocks to which the two nodes belong. The SBM and its generalization have gained their success in the last decades as they can be used to discover and understand the structure of a network, as well as for clustering purposes [32, 33].

2 Hyperbolic Geometric Block Model

Let $G = (V, E)$ be an undirected simple graph with vertex set V and edge set E. The set V is partitioned into n blocks $\{V_i\}_{1 \leq i \leq n}$. The imposed mixing patterns between different blocks is expressed in terms of a $n \times n$ mixing matrix P, where, for each $i, j \in \{1, \ldots, n\}$, P_{ij} measures the connection strength between V_i and V_j, i.e., the average probability the $(u, v) \in E$ over all $u \in V_i$ and $v \in V_j$. Based on the mixing matrix P, on the given demographics and on the chosen average degree $\bar{k}$, we can compute the expected number K_{ij} of edges between blocks i and j (see Appendix 1.2 for more details).

We define the Hyperbolic Geometric Block Model (HGBM) as follows. Let $p_{\mathbb{H}^2}(x)$ be the probability that two vertices at distance x in the hyperbolic disk are connected by an edge in the $\mathbb{H}^2$ model with fixed parameters T, γ and ζ (see Appendix 1.1 for more details on the $\mathbb{H}^2$ model). In the HGBM model, for each pair of blocks $i \leq j$, let E_{ij} be the set of edges that connect vertices in V_i and V_j. If $i = j$, the set E_{ii} of intra-block connections for block i is generated according to the standard $\mathbb{H}^2$ model, using V_i as vertex set and setting the target average degree to $\bar{k}_{ii} = \frac{2K_{ii}}{|V_i|}$. If $i < j$, instead, E_{ij} is the set of inter-block connections between blocks i and j. To obtain E_{ij}, we consider the bipartite analogous of the $\mathbb{H}^2$ model with vertex set $V_i \cup V_j$: for each u, v at distance x, the probability that the edge (u, v) exists is $p_{\mathbb{H}^2}(x)$ if $u \in V_i$ and $v \in V_j$, whereas it is 0 otherwise. In this case, the parameter $\bar{k}_{ij}$ is set to $\bar{k}_{ij} = \frac{K_{ij}(|V_i|+|V_j|-1)}{|V_i||V_j|}$, with a multiplicative factor that accounts for the fact that a vertex $u \in V_i$ can establish links with just a fraction $|V_j|/(|V_i| + |V_j| - 1)$ of the vertex set $(V_i \setminus \{u\}) \cup V_j$. Throughout the paper, we consider the following parametrization of the $\mathbb{H}^2$ model: the power-law exponent of the degree distribution is set to $\gamma = 2.5$, which is typical for real-world social networks; the curvature is set to $\zeta = 1$, to make sure that the graph is generated in the hyperbolic regime [34]; the temperature is set to $T = 0$, to maximize clustering. The impact of these parameters on the topology of the obtained graph has been studied extensively in the literature [8, 34], and we leave a deeper assessment of their impact on the HGBM model to future work.

The experimental results presented in the following consider a synthetic population of 10K vertices representing people subdivided into $n = 4$ different age groups, labeled *child* (0 to 17), *young* (18 to 34), *adult* (35 to 64), or *elder* (65+). The age distribution is taken from the Italian Institute of Statistics (ISTAT) (further details can be found in [13]). The age-based social mixing matrix P (shown in Fig. 1a) is obtained from aggregated contact data from [16], collected through the SOCRATES Data Tool [35]. The expected average degree of the network is set to $\bar{k} = 10$. We ran



(a) Input mixing matrix P. (b) Relative error matrix ϵ (see (2)).

Fig. 1 The mixing matrix P and the relative empirical error matrix ϵ. The relative error range is 1–8%

20 simulations of the HGBM model and analyzed the resulting graphs by looking at the group mixing matrix, the local clustering coefficient distribution and the degree distribution.

Social Mixing First of all, we verify that the simulated networks respect the imposed age-based social mixing structure. In Fig. 1, we show the input matrix P, as defined in 2, and the relative error of the connection probability evaluated between the matrix P and the experimental contact matrix obtained averaging over the 20 simulations. Formally, the simulated mixing matrix is defined as:

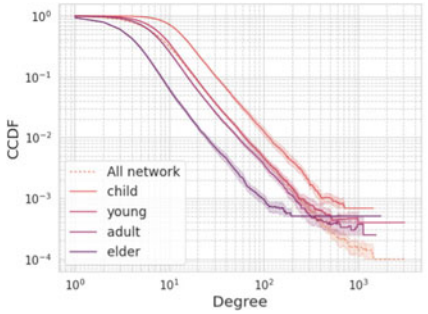
$$P_{ij}^{\text{sim}} = \frac{\sum_{t=1}^{20} |E_{ij}^t|}{20 M_{ij}} \tag{1}$$

where $|E_{ij}^t|$ is the number of edges between age-group i and age-group j in run t, whereas M_{ij} is the maximum number of possible such edges (see (6) in the Appendix). The relative error between the elements P_{ij} and P_{ij}^{sim} is:

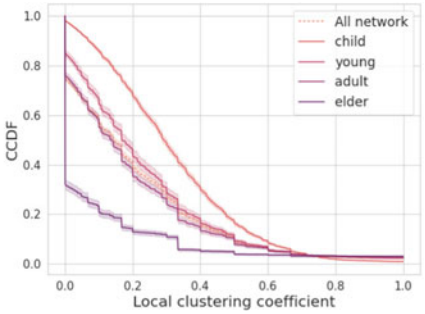
$$\epsilon_{ij} = \frac{P_{ij} - P_{ij}^{\text{sim}}}{P_{ij}}. \tag{2}$$

From Fig. 1 we observe that each element of the average simulated mixing matrix differs from the input matrix by about 5%, thus confirming that the HGBM model preserves, within a reasonable error, the desired mixing structure.

Clustering Figure 2b shows the complementary cumulative distribution function (CCDF) of the local clustering coefficient C_u^{loc}, for the entire network and separately for each age-group. C_u^{loc} is the fraction of neighbors of u that are themselves adjacent [7]. As common practice, the vertices with less than 2 neighbors have been ignored. The average of C_u^{loc} over the entire network is $\langle C_u^{\text{loc}} \rangle \sim 0.2$ (with a variance



(a) CCDF of the vertex degree k_u, averaged over 20 simulations, for the entire network (dashed) and for the different age-groups (solid). The distribution is scale-free and relatively heavy-tailed, as expected, and the average degree of different groups is in line with the matrix P.



(b) CCDF of the local clustering coefficient C_u^{loc}, averaged over 20 simulation runs, for the entire network (dashed) and for the different age-groups (solid). Stronger clustering emerges for age-groups with greater average degree and stronger internal cohesion, as expected.

Fig. 2 Vertex degree and local clustering coefficient CCDF of HGSM

of ~ 0.013 across different simulations), while the global clustering coefficient is $C^{\text{glo}} \sim 0.064$ (with a variance of ~ 0.018). The obtained value $\langle C_u^{\text{loc}} \rangle \sim 0.2$ is compatible with real-world network such as the networks of email address books and of email messages [7], to name two. Moreover Fig. 2b shows that 70% of the network has $C_u^{\text{loc}} > 0.02$. As a benchmark, in a SBM with mixing matrix P and an identical population, we obtained $\langle C_u^{\text{loc}}(SBM) \rangle \sim 0.00177 \pm 0.00002$ over 1000 experiments, generated and evaluated with the Python library IGRAPH [36]. Our HGBM thus provides a significant increase of the local clustering coefficient—100-fold on average, and at least 10-fold for most of the networks—with respect to the SBM, while preserving the mixing structure. Finally, Fig. 2b shows the same trend for all age-groups, and an especially high clustering for those age-groups, i.e., *child* and *young*, having a greater average degree and a stronger internal cohesion, based on the input mixing matrix P—the average degree of group i is proportional to $\sum_j P_{ij}$.

Degree Distribution Figure 2a shows CCDF of the vertex degree k_u, on a log-log scale, for the entire network and separately for each age-group. The expected scale-free heavy-tailed distribution, given by the latent hyperbolic geometry, is clearly visible, with a similar trend for all groups. The average degree of the simulated graphs is $\bar{k}^{\text{sim}} = 10.003 \pm 0.336$, while only $\sim 0.1\%$ of the network has $k_u > 200$, in line with the Dunbar's number [19]. Again, the expected ranking of the age-groups by their average degree is preserved.

3 Combining Latent and Explicit Geometries

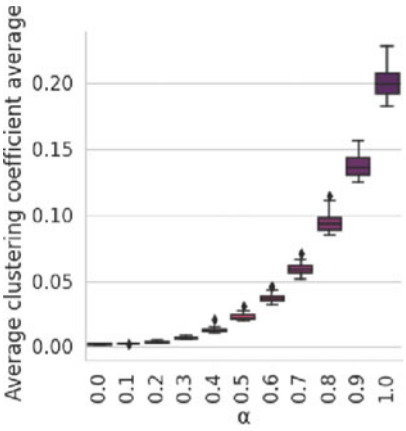
To analyze the possible combined effect of a latent and an explicit geometry, we embed our synthetic population into a stylized physical territory represented as a disk of radius 2.5Km, with each vertex position sampled uniformly at random in the disk. We consider a parametric model in which the edge probability is a linear combination of two terms:

$$p_{uv}(\alpha) = \alpha p_{uv}^{HGBM} + (1-\alpha) p_{uv}^{USN} \tag{3}$$

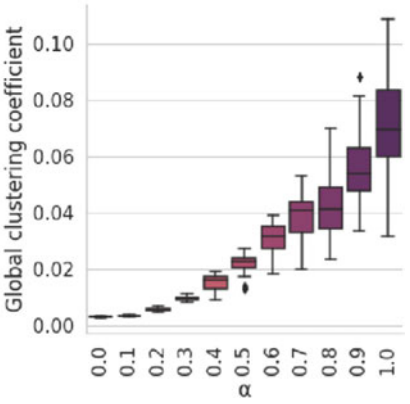
In (3), p_{uv}^{HGBM} is the probability that edge (u, v) exists in the HGBM model defined in Secion 2. p_{uv}^{USN} is instead the probability that the same edge (u, v) exists in a simplified version of the Urban Social Network (USN) model proposed in [13]. In short, $p_{uv}^{USN} \propto P_{g_u g_v} \cdot d_{uv}^{-1}$, where g_u is u's age-group and d_{uv} is the euclidean distance between u and v in the synthetic territory. A brief description of the USN model can be found in Appendix 1.3, while we refer the interested reader to [13, 18] for further details. With respect to the original USN model [13], here we ignore the households and the vertex-intrinsic fitness.

As α varies in [0, 1], (3) shifts from a model only based on an explicit euclidean geometry to a model only based on a latent hyperbolic geometry, with values $\alpha \in (0, 1)$ guaranteeing that the probability that edge (u, v) exists depends on the distance between u and v in both metric spaces. By generating and analyzing 20 networks for each $\alpha \in [0, 0.1, 0.2, \ldots, 1]$, we experimentally evaluated how some characteristics of the obtained network vary as a function of the parameter α. Ideally, we look for a suitable α that provides a network where the frequency of social ties decays as an inverse power of the geographic distance, as agreed by many empirical studies [28, 37], and having high clustering and scale-free degree distribution thanks to the contribution of the hyperbolic model [8].

Figure 3 shows the distribution, over the 20 simulation runs, of the average local clustering coefficient $C_{\text{avg}}^{\text{loc}}$ and of the global clustering coefficient C^{glo}, as a function of α. In Fig. 4a, instead, we compare the CCDF of the vertex degree for different values of α. Finally, Fig. 4b shows the CCDF of the geographical distance between neighboring nodes, again for different values of α. We see that in the USN model ($\alpha = 0$) both $C_{\text{avg}}^{\text{loc}}$ and C^{glo} are negligible and the degree distribution decays exponentially fast. The topological properties of the network improve slowly for small α, but both the clustering and the degree distribution become reasonably good for $\alpha \geq 0.7$. On the other hand, the distribution of distances changes quite smoothly with α and even fairly high values of α show a significant prevalence of "short" edges.

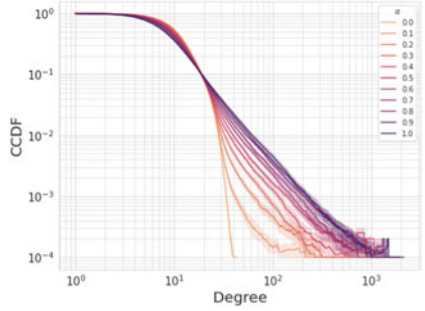


(a) Average local clustering coefficient as a function of α.

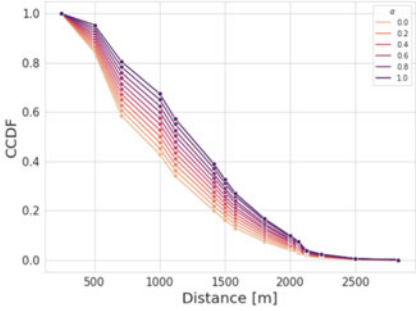


(b) Global clustering coefficient as a function of α.

Fig. 3 Distribution of the average local and global clustering coefficients over 20 simulations for each α. Reasonably good clustering is obtained for $\alpha \geq 0.7$



(a) CCDF of the vertex degree for different values of $\alpha \in [0, 1]$.



(b) CCDF of the distance between adjacent nodes, for different $\alpha \in [0, 1]$.

Fig. 4 CCDF of the vertex degree and of the geographic distance between adjacent nodes, for different values of $\alpha \in [0, 1]$. For $\alpha > 0.5$, the degree distribution glides smoothly towards the one given by the HGBM model. The distance distribution changes smoothly with α and, even for $\alpha \geq 0.7$, it is still quite similar to the one of the USN model (i.e., to $\alpha = 0$)

4 Discussion and Conclusions

Hyperbolic geometric graph models are gaining increasing attention, mostly due to their ability to produce networks with high clustering coefficient and scale-free degree distribution in a simple and elegant way. Combining different ingredients into a single model to obtain suitable topological features is relatively common in the literature—e.g., see [38], that blends Stochastic Block Models and Chung-Lu random graphs,

or [39], that generalizes preferential attachment with a vertex-intrinsic fitness. Until now, however, it was not clear whether a latent hyperbolic geometry could be used to enrich data-driven social network models with desirable topological properties. We made a first step in that direction, through the definition and the analysis of two novel network models.

Our Hyperbolic Geometric Block Model (HGBM) incorporates social group mixing patterns (e.g., age-based mixing inferred from survey data) into the framework of hyperbolic geometric graphs. Through extensive simulations on a stylized population of 10K individuals, we verified that the obtained networks respect the imposed age-based social mixing patterns, and show a high clustering coefficient and a scale-free heavy-tailed degree distribution, in line with empirical findings from social science. To the best of our knowledge, the HGBM is the first block-structured extension of hyperbolic geometric graphs. It is a static model that allows to generate a random graph with high clustering coefficient and heavy-tailed degree distribution, while preserving the input mixing structure. Moreover, it works regardless of the specific configuration chosen for the underlying $\mathbb{H}^2$ model.

We also defined a composite model whose edge probability is the linear combination between the edge probability of the HGBM and that of the USN model [13]. Among the many possible ways to combine a latent hyperbolic and an explicit euclidean geometry, the proposed model has a flexible design and a simple interpretation. A single parameter α controls to which extent the topology of the obtained network depends on the latent and/or data-driven spatial density patterns. If two vertices are very *similar*—in some sense encoded in the latent geometry—they have a positive edge probability regardless of where they live (e.g., because they share common habits or passions); on the other hand, if two vertices live very close to each other, they have a positive edge probability regardless of their similarity (e.g., because they use the same neighborhood facilities). To simulate the model, we randomly placed the vertices on a disk that represents a small urban area of diameter 2.5 km. Our simulations provide preliminary evidence that we can effectively get the most of the two models: at least for some intermediate values of $\alpha \approx 0.7$, we obtain a relatively high clustering coefficient, a scale-free and heavy-tailed degree distribution, and the frequency of social ties that decays as an inverse power of the geographic distance.

We believe that this work paves the way towards the incorporation of hyperbolic latent spaces into data-driven network models, with the potential of producing more realistic social networks while preserving data-driven and empirical features, such as age-based and distance-based mixing. In the next future, we plan to refine our models and to further investigate how the parameters of the underlying hyperbolic graph impact on the properties of the obtained network.

Appendix 1

1.1 $\mathbb{H}^2$ *Hyperbolic Geometric Graph*

The $\mathbb{H}^2$ model [8, 34] is a hyperbolic geometric model with five parameters: the number of nodes N, the temperature T, the target average degree $\bar{k}$, the exponent γ of the desired power-law degree distribution, and the curvature ζ of the latent space. It works by assigning to each node a radial coordinate r and an angular position θ according to the distributions

$$\begin{aligned} \rho(r) &= a\frac{\sinh(ar)}{\cosh(aR)-1}, \\ \rho(\theta) &= \mathcal{U}(0, 2\pi), \end{aligned} \tag{4}$$

where, in (4), $a = \frac{\zeta}{2}(\gamma - 1)$ and R is the radius of the hyperbolic disk, which depends on N, T and $\bar{k}$. The probability that any two vertices are connected by an edge is a function of their hyperbolic distance x, with a functional form that depends on R and T. In the special case $T = 0$ considered in this paper, the connection probability reads

$$p(x) = \Theta(x - R), \tag{5}$$

where $\Theta(\cdot)$ is the Heaviside step function.

For the generation of the hyperbolic graph we used the C++ library [34].

1.2 Data-Driven Social Mixing Matrix

Given a vertex set V, let us consider a partition of V into disjoint sets $\{V_i\}_{1\le i\le n}$ called *blocks*. The total number of pairs of nodes u, v with $u \in V_i$ and $v \in V_j$ is

$$M_{ij} = \begin{cases} \frac{|V_i|(|V_i|-1)}{2}, & \text{if } i = j \\ |V_i||V_j|, & \text{if } i \neq j \end{cases} \tag{6}$$

Let P be a $n \times n$ mixing matrix, i.e., P_{ij} is the frequency of edges between blocks V_i and V_j. If we set the average degree of the whole graph G to $\bar{k}$, the expected number of edges K_{ij} linking blocks V_i and V_j is

$$K_{ij} = \bar{k}\frac{|V|}{2}\frac{P_{ij}M_{ij}}{\sum_{i\le j} P_{ij}M_{ij}} \tag{7}$$

where $|V|$ is the total number of nodes in the graph G.

1.3 Urban Social Network

The Urban Social Network [13] is a model for generating a network of strong ties that captures the social fabric of an urban region. Each node of the network represents an agent u having coordinates in the given territory. The population is partitioned into age-groups based on census data provided by the Italian Institute of Statistics (ISTAT) (all details can be found in [13]). A social fitness score f_u, drawn from an adjustable distribution, accounts for agents having variable sociability.

For the purposes of this work, we only consider edges that represent *acquaintance* ties, ignoring household links. Further, we consider a constant fitness. In this case, the connection probability of the USN model is given by

$$p_{uv}^{USN} = \bar{k}\frac{|V|}{2}\frac{P_{ij}M_{ij}}{\sum_{i\leq j}P_{ij}M_{ij}}\frac{d_{uv}^{-1}}{\sum_{u'\in V_u, v'\in V_v} d_{u'v'}^{-1}} \tag{8}$$

where: $\bar{k}$ is the imposed average degree, $|V|$ is the total number of nodes in the network, P_{ij} and M_{ij} are defined as in Appendix 4, d_{uv} is the euclidean distance between node u and node v, and V_u is the set of all vertices having the same age as u. More details on the USN model can be found in [13].

References

1. Wasserman, K.F.S.: Social Network Analysis: Methods and Applications (Structural Analysis in the Social Sciences). Cambridge University Press (1995)
2. Cauchemez, S., Bhattarai, A., Marchbanks, T.L., Fagan, R.P., Ostroff, S., Ferguson, N.M., Swerdlow, D.: Role of social networks in shaping disease transmission during a community outbreak of 2009 H1N1 pandemic influenza. Proc. Nat. Acad. Sci. **108**(7), 2825–2830 (2011)
3. Webster, C.M., Morrison, P.D.: Network analysis in marketing. Austr. Market. J. (AMJ) **12** (2004)
4. Keeling, M.: The implications of network structure for epidemic dynamics. Theor. Popul. Biol. **67**(1), 1–8 (2005)
5. Ribeiro, H.V., Sunahara, A.S., Sutton, J., Perc, M., Hanley, Q.S.: City size and the spreading of covid-19 in brazil. PloS One **15**(9), e0239699 (2020)
6. Colizza, V., Barrat, A., Barthélemy, M., Vespignani, A.: The role of the airline transportation network in the prediction and predictability of global epidemics. Proc. Nat. Acad. Sci. **103**(7), 2015–2020 (2006)
7. Newman, M.: Networks. Oxford University Press (2018)
8. Krioukov, D., Papadopoulos, F., Kitsak, M., Vahdat, A., Boguná, M.: Hyperbolic geometry of complex networks. Phys. Rev. E **82**(3), 036106 (2010)
9. Zuev, K., Boguná, M., Bianconi, G., Krioukov, D.: Emergence of soft communities from geometric preferential attachment. Sci. Rep. **5**(1), 1–9 (2015)
10. Palla, G., Barabási, A.-L., Vicsek, T.: Quantifying social group evolution. Nature **446**(7136), 664–667 (2007)
11. Huang, Y., Shen, C., Contractor, N.S.: Distance matters: exploring proximity and homophily in virtual world networks. Dec. Support Syst. **55**(4), 969 – 977 (2013); 1. Social Media Research and Applications 2. Theory and Applications of Social Networks

12. Arentze, T., van den Berg, P., Timmermans, H.: Modeling social networks in geographic space: approach and empirical application. Environ. Plann. A **44**(5), 1101–1120 (2012)
13. Guarino, S., Mastrostefano, E., Bernaschi, M., Celestini, A., Cianfriglia, M., Torre, D., Zastrow, L.R.: Inferring urban social networks from publicly available data. Future Internet **13**(5) (2021)
14. Jiang, N., Crooks, A.T., Kavak, H., Burger, A., Kennedy, W.G.: A method to create a synthetic population with social networks for geographically-explicit agent-based models. Comput. Urban Sci. **2**(1), 1–18 (2022)
15. Worldpop. https://www.worldpop.org/ (2020)
16. Mossong, J., Hens, N., Jit, M., Beutels, P., Auranen, K., Mikolajczyk, R., Massari, M., Salmaso, S., Tomba, G.S., Wallinga, J., Heijne, J., Sadkowska-Todys, M., Rosinska, M., John Edmunds, W.: Social contacts and mixing patterns relevant to the spread of infectious diseases. PLOS Med. **5**(3), 1 (2008)
17. Galhotra, S., Mazumdar, A., Pal, S., Saha, B.: The geometric block model. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
18. Celestini, A., Colaiori, F., Guarino, S., Mastrostefano, E., Zastrow, L.R.: Epidemics in a synthetic urban population with multiple levels of mixing. In: International Conference on Complex Networks and Their Applications, pp. 315–326. Springer (2021)
19. Kertész, J., Török, J., Murase, Y., Jo, H.-H., Kaski, K.: Modeling the complex network of social interactions. In: Pathways Between Social Science and Computational Social Science, pp. 3–19. Springer (2021)
20. Cointet, J.-P., Camille Roth, C.: How realistic should knowledge diffusion models be? J. Artif. Soc. Social Simul. **10**(3), 1–11 (2007)
21. Iskhakov, L., Kamiński, B., Mironov, M., Prałat, P., Prokhorenkova, L.: Local clustering coefficient of spatial preferential attachment model. J. Complex Netw. **8**(1), cnz019 (2020)
22. Robins, G., Snijders, T., Wang, P., Handcock, M., Pattison, P.: Recent developments in exponential random graph (p*) models for social networks. Social networks **29**(2), 192–215 (2007)
23. Daraganova, G., Pattison, P., Koskinen, J., Mitchell, B., Bill, A., Watts, M., Baum, S.: Networks and geography: modelling community network structures as the outcome of both spatial and network processes. Social Netw. **34**(1), 6–17 (2012)
24. Boguná, M., Papadopoulos, F.: Krioukov, Dmitri: Sustaining the internet with hyperbolic mapping. Nat. Commun. **1**(1), 1–8 (2010)
25. Angeles Serrano, M., Krioukov, D., Boguná, M.: Self-similarity of complex networks and hidden metric spaces. Phys. Rev. Lett. **100**(7), 078701 (2008)
26. Papadopoulos, F., Kitsak, M., Serrano, M., Boguná, M., Krioukov, D.: Popularity versus similarity in growing networks. Nature **489**(7417), 537–540 (2012)
27. McPherson, M., Smith-Lovin, L., Cook, J.M.: Birds of a feather: homophily in social networks. Ann. Rev. Sociol. **27**(1), 415–444 (2001)
28. Barthélemy, M.: Spatial networks. Phys. Rep. **499**(1–3), 1–101 (2011)
29. Alizadeh, M., Cioffi-Revilla, C.: Crooks, Andrew: Generating and analyzing spatial social networks. Comput. Math. Organ. Theor. **23**(3), 362–390 (2017)
30. Karrer, B., Newman, M.E.J.: Stochastic blockmodels and community structure in networks. Phys. Rev. E **83**, 016107 (2011)
31. Peixoto, T.P.: Hierarchical block structures and high-resolution model selection in large networks. Phys. Rev. X **4**, 011047 (2014)
32. McCallum, A., Wang, X., Corrada-Emmanuel, A.: Topic and role discovery in social networks with experiments on Enron and academic email. J. Artif. Intell. Res. **30**, 249–272 (2007)
33. Zhou, D., Manavoglu, E., Li, J., Lee Giles, C., Zha, H.: Probabilistic models for discovering e-communities. In: Proceedings of the 15th international conference on World Wide Web, pp. 173–182 (2006)
34. Aldecoa, R., Orsini, C., Krioukov, D.: Hyperbolic graph generator. Comput. Phys. Commun. **196**, 492–496 (2015)
35. Willem, L., Van Hoang, T., Funk, S., Coletti, P., Beutels, P., Hens, N.: SOCRATES: an online tool leveraging a social contact data sharing initiative to assess mitigation strategies for COVID-19. BMC Res. Notes **13**(1), 06 (2020)

36. Csardi, G., Nepusz, T.: The igraph software package for complex network research. InterJ. Complex Syst. **1695** (2006)
37. Lambiotte, R., Blondel, V.D., De Kerchove, C., Huens, E., Prieur, C., Smoreda, Z., Van Dooren, P.: Geographical dispersal of mobile communication networks. Phys. A Stat. Mech. Appl. **387**(21), 5317–5325 (2008)
38. Burstein, D.: Asymptotics of the spectral radius for directed Chung-Lu random graphs with community structure. arXiv:1705.10893 (2017)
39. Bianconi, G., Barabási, A.-L.: Competition and multiscaling in evolving networks. EPL (Europhys. Lett.) **54**(4), 436 (2001)

A Biased Random Walk Scale-Free Network Growth Model with Tunable Clustering

Rajesh Vashishtha, Anurag Singh, and Hocine Cherifi

Abstract Complex networks appear naturally in many real-world situations. A power law is generally a good fit for their degree distribution. The popular Barabasi-Albert model (BA) combines growth and preferential attachment to model the emergence of the power law. One builds a network by adding new nodes that preferentially link to high-degree nodes in the network. One can also exploit random walks. In this case, the network growth is determined by choosing parent vertices by sequential random walks. The BA model's main drawback is that the sample networks' clustering coefficient is low, while typical real-world networks exhibit a high clustering coefficient. Indeed, nodes tend to form highly connected groups in real-world networks, particularly social networks. In this paper, we introduce a Biased Random Walk model with two parameters allowing us to tune the degree distribution exponent and the clustering coefficient of the sample networks. This efficient algorithm relies on local information to generate more realistic networks reproducing known real-world network properties.

Keywords Network model · Complex network · Clustering coefficient · Biaised random walk · Barabasi Albert model

1 Introduction

In complex networks, nodes follow connectivity properties and the system's topology [10, 13, 17, 19, 20, 26, 27]. Connectivity property is simply "who connects to who". The World Wide Web (WWW) [3], biological networks [1, 38], and the Internet [37] are typical examples of complex networks. In the WWW network, nodes represent the

R. Vashishtha · A. Singh (✉)
National Institute of Technology Delhi, New Delhi, India
e-mail: anuragsg@nitdelhi.ac.in

H. Cherifi
University of Burgundy, Dijon, France
e-mail: hocine.cherifi@u-bourgogne.fr

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_10

web pages, and edges are the hyperlinks. On the internet, The nodes are computers, and the edges represent the physical and wireless connection. In a biological network, nodes are neurons, and there is an edge between two nodes if a synapse connects the two neurons. The primary purpose of studying complex networks is to characterize the network's properties and develop the correct model for generating real-world networks. The ubiquitous properties of real-world networks are: (1) Small World. Indeed, generally, one needs to visit a few numbers of hops to go from one node to another node as compared to the total number of nodes present in the networks. (2) High clustering coefficient. Indeed, they usually contain a high proportion of triangles. (3) Scale-free degree distribution. Certainly, real-world networks are non-homogeneous. They include a small proportion of high degree nodes and a vast majority of small degree nodes. Hence, a power law is a good fit for their degree distribution.

In recent decades, numerous works have been conducted to better understand real-world network structure [4, 5, 9, 12, 16, 30, 34]. Network modeling is still a very active research field [23, 24, 32]. Network modeling is done with an emphasis on the structural properties of the network, e.g., diameter, degree distribution, clustering coefficient, etc. These properties are ubiquitous in many real-world networks, e.g., social networks [35], technological networks, information networks, and so on. Generally, these networks follow the power-law degree distribution $p(d) \sim d^{-\gamma}$. The value of γ lies within the range, $2 \leq \gamma \leq 3$. Where d denotes the degree of the node. Based on the preferential attachment mechanism, the Barabasi Albert Model (BA) explains real-world network formation. Several models have been developed for generating static real-world networks in recent years. In the famous Erdos-Renyi (ER) model, the number of nodes is fixed, and edges are formed randomly. Note that this model does not respect the high clustering property typical of small-world networks. property (i.e., clustering coefficient) of the real network.

Watts and Strogatz also proposed a static model for generating real-world networks [36]. We consider a network static when the number of nodes is fixed [22]. This model is unsuitable for satisfying the nodes' power-law degree distribution. In this model, the main result is that the diameter of real-world networks increases slowly with increasing network size (number of nodes). They also show that the clustering coefficient of their small-world network model is high compared with the ER model. Some models are used to generate growing network [2]. The BA model by Barabasi and Albert is the most famous example of a growing network model.

In this model, preferential attachment defines the rule for attaching a new node to the existing network. A new node in the networks tends to link with high-degree nodes rather than being attracted by low-degree nodes. In the BA model, one needs global information about the network connectivity to attach a new node [25]. In contrast, in real-world networks, e.g., social networks or WWW networks, one does not assume such knowledge when adding a new node to the network. Real-world network formation may not use global information to attach a new node to the existing one.

The main idea proposed in this paper is to generate a real-world network that does not use global information. We suggest using biased random walk [?] to develop the network model. With the local knowledge, the generated networks satisfy all the

characteristic properties of complex networks. The biased random walk provides a way to understand the network structure, using the local information of the network [11]. Note that one can use the biased random walk to define structural centrality [6] of the networks. One achieves the desired goal using a random walk, where the transition probability is proportional to the degree of node reached through a random edge [15].

Networks may represent many complex systems, nodes (V), and their interconnection by edges (E). A graph can represent each social network, and edges represent peoples' relationships. How strongly they are connected is represented by the weight matrix. The graph is a collection of three tuples (V, E, W). Where V is the set of vertices, E is the set of edges, W is the weight matrix, $W(i, j) = w_{ij}$ if an edge exists between i and j and 0 otherwise. For an unweighted graph, there is no weight matrix. Instead of a weight matrix, one uses the adjacency matrix.

The various classical models (WS, BA, ER) used to analyze real networks' properties are not suited to many real-world applications. For example, in the WS model, the network follows the Poisson degree distribution. However, many real networks exhibit a power-law degree distribution. The BA model fulfills the power-law degree distribution but uses global information about the network in its growing process. The main drawback is that one needs to estimate the preferential attachment probability of each node before linking a new node to the existing network. Indeed, by adding a new node to the existing network, one needs to know the degree of all the existing nodes in the network. In other words, knowledge about the network needs to be global to perform the growing process. Furthermore, the BA model's clustering coefficient of the generated real networks is almost zero.

We develop a model to generate real-world networks based on a biased random walk in the proposed work. It uses the existing seed network's local information (i.e., information about the vertex and its first-order neighbors). One can also control the clustering coefficient of the generated network.

Section 2 briefly presents the existing generating models with their limitations. Section 3 describes the biased random walk and the proposed model. Section 4 reports its experimental evaluation. Section 5 concludes the work and discusses future research directions.

2 Literature Review

Network modeling is essential to understanding the property of real-world networks. Alexei et al. [32] proposed a model for growing the network based on a local rule. According to them, one can rely on local information to control the network's various properties (degree distribution, clustering, hierarchy) [7]. They propose three models for network modeling: (1) the Random Walk model (RWM), (2) the Recursive search model (RSM), and (3) Connecting the nearest-neighbors model (CNNM). The RWM model answer two questions (1) How to add a new node in the network? (2) How to add an edge to the existing network? Initially, the network contains only one node

as a seed network. To add a new node to the existing network, a random node is selected in the existing network and connected to the new node. Therefore, an edge is created between the randomly selected node and a new node. One can also connect a new node to one of the neighbors of a randomly selected node with probability p_e. The limitation of the RWM model is that a new node can only be connected to a random node and its neighbor. Recursive Search Model (RSM) overcomes these limitation. The RSM model uses a recursive approach for connecting new nodes in the existing network. Now, a new node can connect with any randomly selected node's neighbors. Hence, it affects a larger fraction of the network. In the CNNM model, two non-adjacent nodes are connected if they share at least one common neighbor. Poster et al. [33] show that the clustering coefficient and average neighbor degree depend on the vertex degree. Alexei et al. [32] concluded that the local clustering coefficient of a vertex is inversely proportioning to the vertex degree. Saramaki et al. [28] proposed a model based on the random walk. They found that the random walk can generate a similar network as the BA model (BA model at γ=3). They also conclude that a random walk is sufficient to achieve the preferential attachment, which Albert Barabasi previously suggests. They focused on the power-law degree distribution and ignored other network properties. Toivonen et al. [31] proposed a model for a social network. Their work aims to capture the features of a real-world social network. The model contains two growth processes, (1) random attachment and (2) implicit preferential attachment. This model has an initial seed network with n_0 vertices. Choose some vertices randomly, called the initial vertex. The neighbor of these initial vertex is known as secondary contacts. A new node is connected to some initial and secondary contacts. This process repeats until the desired size network is reached. The limitation of this model is that one cannot control triangle formulation. Indeed, every new node is connected to a randomly selected node and its neighbor.

Serrano et al. [29] proposed a model for generating real-world networks. This model has three different parts for generating the real-world networks, (1) assignment of a degree to each node and assignment of several triangles to each degree class according to the expected distributions, (2) closure of triangles, and (3) closure of the remaining free stubs. The limitations of this model are that (1) the number of nodes, (2) the degree distribution, and (3) the clustering coefficient of generated networks are fixed.

Remember that some other models use global information for generating real-world networks. WS, BA, and ER models are the most popular examples of models that use global information.

3 Biased Random Walk Model

We propose a model for generating a real-world network using local information, i.e., information about the vertex and its neighbors. It uses a biased random walk for attaching a new node to the existing network. Using local information generates the network quicker than models relying on the global network information.

We show that the resultant network follows the power-law distribution. The main advantage of the proposed solution is that it helps control the network's clustering coefficient.

3.1 Random Walk (RW)

A random walk is simply based on a Markov-chain model. It is a finite Markov chain that is time-reversible [18]. Time-reversible Markov chain can be viewed as random walks on an undirected graph [18]. A random walker jumps from node to node, and each node represents the state of the Markov process [8]. In a random walk, a walker randomly selects a node and jumps to its neighbors according to a transition matrix.

The random walk is defined with the help of single step transition matrix, T_m [11]. Where, element of T_m, p_{ij} is the probability to jump from node i to node j. Transition probability is defined as $p_{ij} = \frac{a_{ij}}{a_i}$, where, $a_i = \sum_j^n a_{ij}$. Basically, $\sum_j a_{ij}$ is the degree of node i.

3.2 Biased Random Walk (BRW)

Initially, a walker is placed at any vertex randomly. It can move to one of its neighbors with some probability p. But in the case of a biased random walk, we add one parameter β controlling the biasedness of the walker [14]. In a biased random walk, initially, a walker is placed at vertex x. It jumps to the neighbor of x, i.e., y with some probability p_{xy} defined as follows:

$$p_{xy} = \frac{d_x^\beta}{\sum_{x=1}^{d_x} d_x^\beta} \tag{1}$$

where, d_x is the degree of node x. β is the biased parameter used to manage the biasedness of the random walker.

The value of β lies between $-\infty$ to $+\infty$. If the value of β is very low, i.e., approaching $-\infty$, then the random walker visits dangling ends more often. Alternatively, if the value of the β is very high, i.e., approaching $+\infty$, then the random walker is stuck at the central part or hub of the network.

3.3 Algorithm of the Biased Random Walk Model

Initially, the network contains a fixed number of nodes as seed nodes. New nodes are added one at a time in the network with a fixed number of edges. A new node will

form a certain number of connections or edges with the existing nodes of the network. Biased random walk helps to mark the nodes in the existing network to which new node will get connected. There are other essential network properties to remember during network formation. One of the properties is the clustering coefficient. The clustering coefficient is managed by controlling triangles formation in the network (transitive connections).

The biased random walk path length controls the Triangle formulation, which eventually controls the clustering coefficient. If the path length of a random walker is greater than or equal to 2, i.e., $l \geq 2$, no triangle is added to the network. In contrast, if the path length is equal to 1, i.e., $l = 1$, then adding a new node with m edges results in the formation of $m - 1$ triangles in the network. Therefore, a parameter α manages the mixing of the path length in a biased random walk. α helps in controlling the transitive connections in the network. $0 \leq \alpha \leq 1$. The following algorithm 1 describes the procedure for generating a network using BRW. In a biased random walk, another parameter β helps control the network's power-law degree distribution.

Algorithm 1 Algorithm for generating real world networks through BRW

1: **Set parameters**: (i) n_0: number of initial nodes, (ii) $0 \leq \alpha \leq 1$, (iii) target network size $n_0 + n$, (iv) the number of edges m that a newly node is connected with that edges to the existing nodes. The value of m is less or equal to number of nodes in the seed network i.e., ($m \leq n_0$).
2: **Input**: Initial seed network with n_0 nodes. $0 \leq \alpha \leq 1$. The number of nodes are to be added in the seed network n. The number of edges that are uses to add a new node.
3: **Output**: Real networks with contain $n_0 + n$ nodes. The degree distribution of nodes is follow power law.
4: **Initialization** $\{Mark\ nodes\} \leftarrow \phi$, $Current\_node \leftarrow \phi$, Assign the $prob\_value$ of each existing nodes using binomial distribution i.e., $p(n, k)$.
5: **while** Until n new node are added to the network. **do**
6: Select a node v_s randomly.
7: From node v_s start a $l \geq 1$ step biased random walk and reach a end node v_e and mark that node and add that node into the mark nodes set i.e., $\{mark\ nodes\} \leftarrow \{mark\ nodes\} + \{v_e\}$
8: **while** size$\{mark\ nodes\} \leq m$ **do**
9: Set $Current\_node \leftarrow v_e$
10: **if** $Prob\_value(Current\_node) \leq \alpha$ **then** make a 1-step BRW reach a vertex v_e and mark v_e node. Add that v_e to the mark node set i.e., $\{mark\ nodes\} \leftarrow \{mark\ nodes\} + \{v_e\}$
11: **else** make a 2-step BRW and mark that node v_e. Add that mark node to the mark node set i.e., $\{mark\ nodes\} \leftarrow \{mark\ nodes\} + \{v_e\}$.
12: **end if**
13: **end while**
14: Add new node to all the nodes that are present in the mark node set i.e., $\{mark\ nodes\}$.
15: **end while**

The inputs of Algorithm 1 are the seed network, the number of links of the new node, and the number of nodes to add to the existing network. If the value of α is near zero, then, most of the time, the random walker performs a 2-step walk, and there is no new triangle in the network. Consequently, the clustering coefficient of

the sample network is low. In contrast, if α is near 1, the clustering coefficient is very high because the walker often performs a single-step walk adding more triangles to the network.

4 Experimental Results

This section reports the results of the simulations performed with the proposed model. First, we illustrate the behavior of the biased random walk on a toy network example. Then we investigate the degree distribution and the clustering coefficient of the generated sample networks.

4.1 Biased Random Walk

The biasedness of the random walker depends on the value of the parameter β. Figure 1 present a toy example network used to illustrate the walker's biasedness for visiting it. Table 1 reports the number of visits of each node by a random walker in the toy example network for various values of the biased parameter β. If the value of β is very low, then the random walker visits dangling ends more often. In contrast, if the value of β is very high, then the random walker is stuck in the central part or in the hub node of the network.

4.2 Degree Distribution Analysis

To analyze the degree distribution of the generated network, we use a star network with five nodes as seeds ($n_0 = 5$). Four thousand new nodes are added at timestamp to the seed network. Each new node is added with $m = 2$ edges. Figure 2a shows the generated network degree distribution estimate. The maximum degree of the generated network is 77. There is a unique node with a maximum degree value.

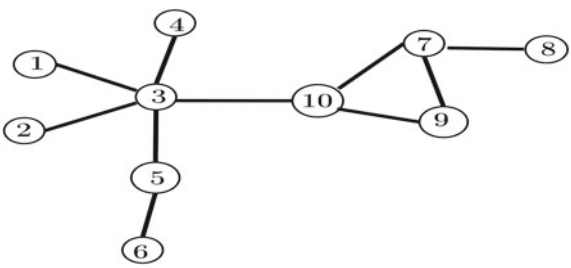
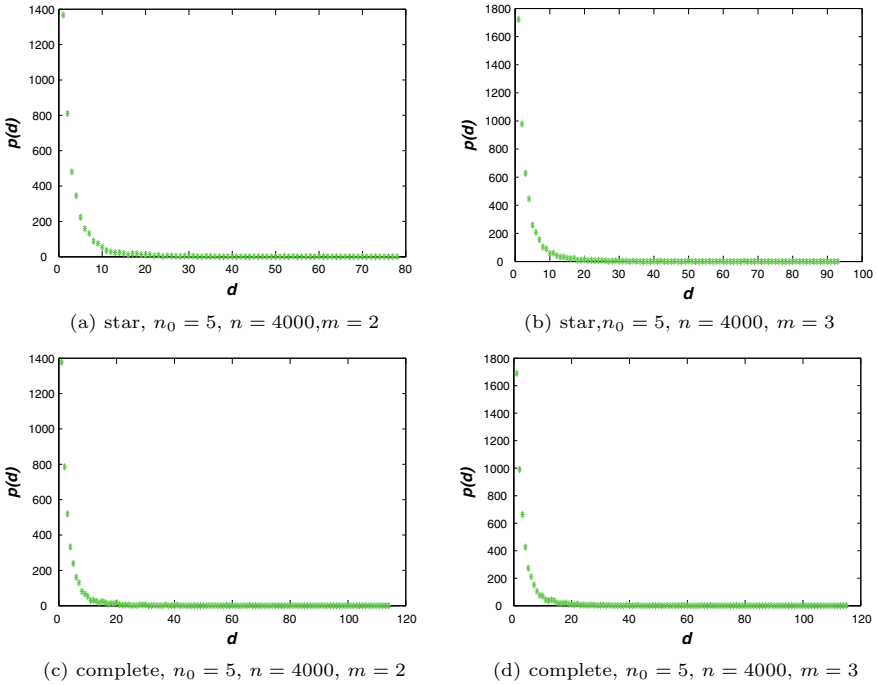


Fig. 1 Toy example network for investigating the BRW behavior. It contains a triangle (7, 10, 9) linked to a Hub (3) and low degree nodes (1, 2, 4, 6, 8)

Table 1 Number of visits of the nodes by the Biased random walk for the toy network example of Fig. 1

	$\beta = 100$	$\beta = 10$	$\beta = 0$	$\beta = -10$	$\beta = -100$
1	0	1	403	2517	0
2	0	0	424	2483	0
3	1698	1666	1745	5001	0
4	0	32	916	0	0
5	1617	1680	1405	0	0
6	0	0	495	0	0
7	3302	3330	1861	0	5000
8	1	0	493	0	5001
9	0	27	920	0	0
10	3383	3265	1339	0	0



(a) star, $n_0 = 5$, $n = 4000$,$m = 2$ (b) star,$n_0 = 5$, $n = 4000$, $m = 3$

(c) complete, $n_0 = 5$, $n = 4000$, $m = 2$ (d) complete, $n_0 = 5$, $n = 4000$, $m = 3$

Fig. 2 Power law degree distribution of the generated networks

The minimum degree of nodes is two, and the number of nodes with minimum degree nodes is 1388, as shown in Fig. 2a. Starting with the same seed network, we add the same number of nodes one at a time with $m = 3$ edges. Once again, the generated network degree distribution estimate follows a power law as illustrated in

Fig. 2b. In this case, the maximum degree is 93, and the frequency of the maximum degree is 1. The minimum degree of generated network is three, and the number of minimum degree nodes is 1753. Rather than starting with a star, the original network used as a seed is a complete network of 5 nodes ($n_0 = 5$). We add 4000 new nodes at timestamp to the seed network. Each new node has $m = 2$ edges. Once again, the generated network follows the power law shown in Fig. 2c. The maximum degree of this network is 115, and the frequency of the maximum degree is 1. Its minimum degree is two, and the number of minimum degree nodes is 1398. Now to the same seed network, we add the same number of nodes one at a time with $m = 3$ edges. Again, the generated network follows the power law as shown in Fig. 2d. The maximum degree of the generated network is 117, and the frequency of the maximum degree is 1. The minimum degree is three, and the number of minimum degree nodes is 1705. The power law exponent estimation ranges between 2 and 3. As desired, these values are typical of real-world power-law degree distributions.

4.3 Clustering Coefficient Analysis

Clustering Coefficient The clustering coefficient is a measure of the nodes ability to form clusters. The local clustering coefficient of a node v is defined as follows:

$$CC_l\,(v) = \frac{2N_v}{K_v\,(K_v - 1)} \tag{2}$$

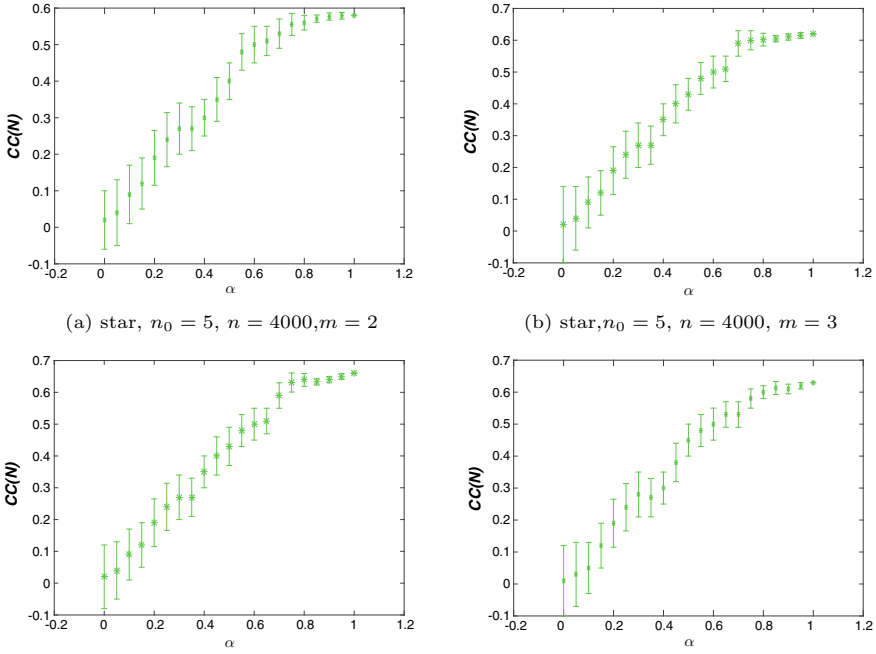
where, N_v the number of links between neighbours of v. K_v the number of neighbours of node v.

Global Clustering Coefficient The Global clustering coefficient is the measure of a number of triad closures in graph relative to connected triples [21]. Three nodes form connected triple if we can reach from one node to other two nodes. Three nodes form a triad closure if each pair of nodes has a direct link. Global clustering coefficient is the ratio of the number of triad closures to the connected triples. The clustering coefficient of a graph is denoted by $CC_g(G)$.

$$CC_g(G) = \frac{3TC}{CT} \tag{3}$$

where, TC is the total number of triad closure in the graph. CT is the total number of connected triples. One can control the clustering coefficient of the generated network by tuning α. When the values of α are small, i.e., approaching 0, the clustering coefficient of the sample network is small. It grows almost linearly with α until it reaches an asymptotic value, as shown in Fig. 3.

In Fig. 3, the initial seed network is the star of 5 nodes, and we add 4000 nodes. Each new node has two edges. Figure 3 reports the evolution of the clustering coefficient versus the value of α. When the value of α is zero, the clustering coefficient is



(a) star, $n_0 = 5$, $n = 4000$,$m = 2$

(b) star,$n_0 = 5$, $n = 4000$, $m = 3$

(c) complete, $n_0 = 5$, $n = 4000$, $m = 2$

(d) complete, $n_0 = 5$, $n = 4000$, $m = 3$

Fig. 3 Clustering coefficient versus α

small, and the variance of the clustering coefficient is high. Increasing alpha increases the clustering coefficient's value. The value of the variance also decreases. In Fig. 3a, the maximum value of clustering coefficient is 0.58 at $\alpha = 1$ and the minimum value of the clustering coefficient is 0.01 at $\alpha = 0$. Now in the same seed network, the same number of nodes are added one at a time with $m = 3$ edge. In Fig. 3b, the maximum value of clustering coefficient is 0.62 at $\alpha = 1$, and the minimum value of clustering coefficient is 0.02 at $\alpha = 0$. When the initial seed network is a complete network of 5 nodes, and one adds 4000 nodes with two edges, the maximum value of the clustering coefficient is 0.63 at alpha 1 (see Fig. 3c). The minimum value of the clustering coefficient is 0.03. In the same situation, if one adds nodes one at a time with $m = 3$ edges, the maximum value of the clustering coefficient is 0.65 at alpha 1, and the minimum value of the clustering coefficient is 0.03 (see Fig. 3d).

5 Conclusions

Real-world networks are characterized by a power-law degree distribution and a high clustering coefficient. We propose a model based on a biased random walk to generate networks with these typical properties. The random walk has the advantage of using

local information about the seed network rather than global information. Therefore, it is more efficient than popular methods such as the BA model. Experiments show that the generated real-world networks follow the power law degree distribution. Additionally, one can control the clustering coefficient of the sample networks. The value of the clustering coefficient Increases almost linearly with the tuning parameter α value until it reaches an asymptotic value. In future work, we plan to investigate controlling the average diameter of the network with a biased random walk and exploit this property to design a community detection algorithm.

References

1. Arquam, M., Singh, A., Cherifi, H.: Impact of seasonal conditions on vector-borne epidemiological dynamics. IEEE Access **8**, 94510–94525 (2020)
2. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)
3. Boudourides, M., Antypas, G.: A simulation of the structure of the world-wide web. Sociol. Res. Online **7**(1), 9–25 (2002). https://doi.org/10.5153/sro.684
4. Chakraborty, D., Singh, A., Cherifi, H.: Immunization strategies based on the overlapping nodes in networks with community structure. In: International Conference on Computational Social Networks, pp. 62–73. Springer, Cham (2016)
5. Cherifi, H., Palla, G., Szymanski, B.K., Lu, X.: On community structure in complex networks: challenges and opportunities. Appl. Netw. Sci. **4**(1), 1–35 (2019)
6. Courtain, S., Leleux, P., Kivimäki, I., Guex, G., Saerens, M.: Randomized shortest paths with net flows and capacity constraints. Inf. Sci. **556**, 341–360 (2021)
7. Erdős, P., Rényi, A.: On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci **5**(1), 17–60 (1960)
8. Fouss, F., Pirotte, A., Renders, J.M., Saerens, M.: Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. IEEE Trans. Knowl. Data Eng. **19**(3), 355–369 (2007)
9. Ibnoulouafi, A., El Haziti, M., Cherifi, H.: M-centrality: identifying key nodes based on global position and local degree variation. J. Stat. Mech. Theor. Exper. **2018**(7), 073407 (2018)
10. Jebabli, M., Cherifi, H., Cherifi, C., Hamouda, A.: User and group networks on YouTube: a comparative analysis. In: 2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA), pp. 1–8. IEEE (2015)
11. Klein, D.J., Randić, M.: Resistance distance. J. Math. Chem. **12**(1), 81–95 (1993)
12. Kumar, M., Singh, A., Cherifi, H.: An efficient immunization strategy using overlapping nodes and its neighborhoods. In: Companion Proceedings of the The Web Conference 2018, pp. 1269–1275 (2018)
13. Lasfar, A., Mouline, S., Aboutajdine, D., Cherifi, H.: Content-based retrieval in fractal coded image databases. In: Proceedings 15th International Conference on Pattern Recognition. ICPR-2000, vol. 1, pp. 1031–1034. IEEE (2000)
14. Lee, S., Yook, S.H., Kim, Y.: Centrality measure of complex networks using biased random walks. Euro. Phys. J. B **68**(2), 277–281 (2009)
15. Leleux, P., Courtain, S., Guex, G., Saerens, M.: Sparse randomized shortest paths routing with Tsallis divergence regularization. Data Min. Knowl. Disc. **35**(3), 986–1031 (2021)
16. Lewis, T.G.: Network Science: Theory and Applications. John Wiley & Sons (2011)
17. Li, M., Liu, R.R., Lü, L., Hu, M.B., Xu, S., Zhang, Y.C.: Percolation on complex networks: theory and application. Phys. Rep. **907**, 1–68 (2021). https://doi.org/10.1016/j.physrep.2020.12.003, https://www.sciencedirect.com/science/article/pii/S0370157320304269

18. Lovász, L., et al.: Random walks on graphs: a survey. Combinatorics Paul Erdos Eighty **2**(1), 1–46 (1993)
19. Messadi, M., Cherifi, H., Bessaid, A.: Segmentation and ABCD rule extraction for skin tumors classification. arXiv:2106.04372 (2021)
20. Mourchid, Y., El Hassouni, M., Cherifi, H.: A new image segmentation approach using community detection algorithms. In: 2015 15th International Conference on Intelligent Systems Design and Applications (ISDA), pp. 648–653. IEEE (2015)
21. Newman, M.: Networks. Oxford University Press (2018)
22. Newman, M.E., Strogatz, S.H., Watts, D.J.: Random graphs with arbitrary degree distributions and their applications. Phys. Rev. E **64**(2), 026118 (2001)
23. Orman, G.K., Labatut, V., Cherifi, H.: Towards realistic artificial benchmark for community detection algorithms evaluation. arXiv:1308.0577 (2013)
24. Orman, K., Labatut, V., Cherifi, H.: An empirical study of the relation between community structure and transitivity. In: Complex Networks, pp. 99–110. Springer, Berlin, Heidelberg (2013)
25. Pastor-Satorras, R., Vázquez, A., Vespignani, A.: Dynamical and correlation properties of the internet. Phys. Rev. Lett. **87**(25), 258701 (2001)
26. Pastrana-Vidal, R.R., Gicquel, J.C., Colomes, C., Cherifi, H.: Frame dropping effects on user quality perception. In: Proceedings of 5th International WIAMIS (2004)
27. Rital, S., Bretto, A., Cherifi, H., Aboutajdine, D.: A combinatorial edge detection algorithm on noisy images. In: International Symposium on VIPromCom Video/Image Processing and Multimedia Communications, pp. 351–355. IEEE (2002)
28. Saramäki, J., Kaski, K.: Scale-free networks generated by random walkers. Phys. A Stat. Mech. Appl. **341**, 80–86 (2004)
29. Serrano, M.A., Boguná, M.: Tuning clustering in random networks with arbitrary degree distributions. Phys. Rev. E **72**(3), 036133 (2005)
30. Singh, A., Cherifi, H., et al.: Centrality-based opinion modeling on temporal networks. IEEE Access **8**, 1945–1961 (2019)
31. Toivonen, R., Onnela, J.P., Saramäki, J., Hyvönen, J., Kaski, K.: A model for social networks. Phys. A Stat. Mech. Appl. **371**(2), 851–860 (2006)
32. Vázquez, A.: Growing network with local rules: preferential attachment, clustering hierarchy, and degree correlations. Phys. Rev. E **67**(5), 056104 (2003)
33. Vázquez, A., Pastor-Satorras, R., Vespignani, A.: Large-scale topological and dynamical properties of the internet. Phys. Rev. E **65**(6), 066130 (2002)
34. Vespignani, A.: Twenty years of network science (2018)
35. Vitevitch, M.S., Chan, K.Y., Roodenrys, S.: Complex network structure influences processing in long-term and short-term memory. J. Memory Lang. **67**(1), 30–44 (2012)
36. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature **393**(6684), 440 (1998)
37. Wu, X., Yu, K., Wang, X.: On the growth of internet application flows: a complex network perspective. In: 2011 Proceedings IEEE INFOCOM, pp. 2096–2104 (2011). https://doi.org/10.1109/INFCOM.2011.5935019
38. Wuchty, S., Ravasz, E., Barabási, A.L.: The Architecture of Biological Networks, pp. 165–181. Springer US, Boston, MA (2006)

The Distance Backbone of Directed Networks

Felipe Xavier Costa, Rion Brattig Correia, and Luis M. Rocha

Abstract In weighted graphs the shortest path between two nodes is often reached through an indirect path, out of all possible connections, leading to structural redundancies which play key roles in the dynamics and evolution of complex networks. We have previously developed a parameter-free, algebraically-principled methodology to uncover such redundancy and reveal the distance backbone of weighted graphs, which has been shown to be important in transmission dynamics, inference of important paths, and quantifying the robustness of networks. However, the method was developed for undirected graphs. Here we expand this methodology to weighted directed graphs and study the redundancy and robustness found in nine networks ranging from social, biomedical, and technical systems. We found that similarly to undirected graphs, directed graphs in general also contain a large amount of redundancy, as measured by the size of their (directed) distance backbone. Our methodology adds an additional tool to the principled sparsification of complex networks and the measure of their robustness.

Keywords Directed networks · Weighted graphs · Network backbones · Sparsification · Shortest path · Redundancy

Felipe Xavier Costa and Rion Brattig Correia—Contributed equally.

F. X. Costa · R. B. Correia · L. M. Rocha (✉)
Systems Science and Industrial Engineering Department, Binghamton University State University of New York, Binghamton, NY 13902, USA
e-mail: rocha@binghamton.edu

F. X. Costa
Department of Physics, State University of New York at Albany, Albany, NY 12222, USA

F. X. Costa · R. B. Correia · L. M. Rocha
Instituto Gulbenkian de Ciência, Oeiras 2780-156, Portugal

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_11

1 Introduction

Networks are a canonical method to model complex multivariate interactions and have been proven useful in the study of a variety of problems, such as social interaction and human mobility to predicting epidemic spreading [6, 16]. and modeling biochemical networks to predict the onset of diseases [8, 12]. This modeling approach allows for a shift from the traditional scientific focus on the (reductionist) study of things (e.g., animals or proteins), to the study of system-wide interactions among these things, such as friendships among animals, or bonding among proteins. In network science, typically, these multivariate interactions are represented as edges that connect variables as nodes in a graph. In addition, networks built to represent real-world complex systems often denote variable interaction with a weight that is proportional to the strength of interaction between nodes, such as a proximity (similarity) or a distance (dissimilarity). For instance, edge weights can represent the probability of interaction between genes [8], similarity between concepts in a knowledge space [10], or a measure of how much time two individuals spent together in close vicinity [9]. In its simplest form, edge weights are non-directed, meaning interactions between nodes are symmetric. This is especially the case when distance and shortest paths between nodes are relevant for analysis—e.g. inferring the likelihood that a person infects another in a population under epidemic spread—because distance measures are by default symmetric (in addition to being non-negative and anti-reflexive [26]).

Redundancy is considered a fundamental aspect in the evolution of complex systems [7]. Distinct aspects of the phenomenon have been shown to greatly contribute to our understanding of network dynamics, controllability, and robustness [13, 14, 24]. In particular, we have shown that most networks where edges represent distance (or dissimilarity) contain large amounts of topological redundancy in computing shortest paths, which can be identified through our algebraically-principled and parameter-free *distance backbone* [24]. This means our method differs from other backbones by requiring no tunning parameter, null model comparisons, or Monte Carlo approximations. However, even though distance is typically considered to be symmetric [11], many real-world complex systems are best modeled by *directed, weighted graph*s. Indeed, asymmetric interactions have been to shown to be important in a variety of domains, ranging from unreciprocated friendships [2], food-webs and host-parasite ecological networks [15], to designing smarter urban traffic and cities [1, 22].

Here, our main contribution is the extension of the distance backbone methodology to *directed* weighted graphs. Specifically, we build upon the concepts of transitive and distance closure for undirected weighted graphs [26] to identify a subgraph whose edges do not break a generalized triangle inequality and which are sufficient to compute all shortest *directed* paths. In other words, we obtain a *directed distance backbone* that preserves the distribution of shortest paths in directed weighted graphs. This in turn allows us to quantify both the structural redundancy of such networks and their robustness to random attacks. Real-world examples also show preliminary results that having directed edges yields a larger distance backbone than it does for undirected graphs.

2 Closures in Complex Networks

In social networks, indirect associations are often exemplified as "the friend of my friend is also my friend". These indirect associations can be described in a graph $G(X)$, defined on the set of nodes X, in terms of the transitive and distance closures. Transitive closures assume edge weights to measure a similarity while distance closures assume weights to be a dissimilarity between nodes [26]. The formalism for closures in weighted undirected networks has been introduced in Simas et al. [24]. We revise this mathematical construction in this section and, in Sect. 3, we relax the symmetry condition previously considered while showing that the formalism of closures in complex networks is applicable to both undirected and directed networks.

2.1 Transitive Closure

The strength of interactions between the nodes $x_i \in X$ can be measured by a proximity graph, $P(X)$. This is a reflexive network with edges weights $p_{ij} \in [0, 1]$, a continuous range of values, with $p_{ii} = 1$. Transitivity is computed via the composition of generalized, weighted logical operators. These are extensions of the binary logic operators, derived from probabilistic metric spaces and fuzzy logic, and are called triangular norms and conorms [17, 24, 26].

A triangular norm (t-norm) is a generalized logical conjunction given by the operation $\wedge: [0, 1] \times [0, 1] \to [0, 1]$. It satisfies the properties of commutativity ($p \wedge q = q \wedge p$), associativity ($p \wedge (q \wedge w) = (p \wedge q) \wedge w$), monotonicity ($p \wedge q \leq w \wedge v$ implies $p \leq w$ and $q \leq v$), and having 1 as its identity element ($p \wedge 1 = p$). Similarly, a triangular conorm (t-conorm) is a generalized logical disjunction given by the operation $\vee : [0, 1] \times [0, 1] \to [0, 1]$. It is also commutative, associative, monotonic, but has 0 as its identity element ($p \vee 0 = p$). Combining them gives us the compositions of P with itself as

$$P^{\eta} = P \circ P^{\eta-1} \iff p_{ij}^{(\eta)} = \bigvee_{k} \left(p_{ik} \wedge p_{kj}^{(\eta-1)} \right), \tag{1}$$

considering $\eta \in \mathbb{Z} \geq 2$ and $P^1 = P$. This leads to the transitive closure of $P(X)$ given by

$$P^T(X) = \bigcup_{\eta=1}^{\kappa} P^{\eta} \iff p_{ij}^T = p_{ij} \vee p_{kj}^{(2)} \vee \cdots \vee p_{ij}^{(\kappa-1)} \vee p_{ij}^{(\kappa)}. \tag{2}$$

For general t-norms and t-conorms the closure is reached as $\kappa \to \infty$. But with proximity graphs, as long as $\wedge \equiv \min$, the closure $P^T(X)$ converges for a finite κ no larger than the graph diameter [17, 26]. The adjacency matrix $P^{\eta}(X)$ measures the proximity for paths of size η, while the transitive closure $P^T(X)$ accounts for the strongest proximity for paths *up to* size κ.

We say that a proximity graph is transitive with respect to the algebraic structure $([0, 1], \vee, \wedge)$ if for every weighted edge p_{ij} in the graph we have:

$$p_{ij} \geq \bigvee_k (p_{ik} \wedge p_{kj}) \tag{3}$$

for any node $x_k \in X$. By construction, all edges of $P^T(X)$ obey this generalized transitivity constraint, while only a subset of edges of $P(X)$ typically do. In the context of the generalized transitivity criterion given by Eq. 3, fully transitive graphs denote a similarity multivariate relation, whereas graphs that break transitivity for at least one edge denote a proximity relation [17].

For connected, undirected graphs, this leads to a closure where $p_{ij}^T > 0$ for all x_i and x_j in X, i.e. a complete or fully connected graph. Unfortunately, this does not generalize for directed graphs, where there can be nodes that only have outwards connections, and therefore can never be reached from other nodes.

2.2 Distance Closure

In network science, we often need to compute shortest paths on graphs to infer the (direct and indirect) influence of variables on one another. This requires casting the network as a distance (or dissimilarity) graphs, $D(X)$ on the set of node variables X. These graphs have non-negative weights, i.e. adjacency matrix elements $d_{ij} \in [0, \infty)$, and are anti-reflexive: $d_{ii} = 0$. They are also isomorphic to proximity graphs [26] via a strictly monotonic decreasing map $\varphi: [0, 1] \to [0, \infty)$ constrained by:

$$\underset{k}{f}\{g(\varphi(p_{ik}), \varphi(p_{kj}))\} = \varphi(\bigvee_k (p_{ik} \wedge p_{kj})) \quad \forall x_i, x_j, x_k \in X, \tag{4}$$

where f and g are isomorphic operations to $\wedge$ and $\vee$, respectively, in the sense that they are associative, commutative, monotonic, and having identity elements given by $\varphi(0) \to \infty$ for f and $\varphi(1) = 0$ for g. Due to this construction, g and f are named triangular distance norm (td-norm) and conorm (td-conorm), respectively [26].

Though an infinite number of maps satisfy the isomorphism, the simplest, which we use here unless otherwise noted, is the familiar distance function:

$$d_{ij} = \varphi(p_{ij}) = \frac{1}{p_{ij}} - 1, \tag{5}$$

that easily converts between proximity $P(X)$ and distance $D(X)$ graphs. In addition to being non-negative and anti-reflexive, distance measures are typically symmetrical, and if transitive, are also known as metric [11].

Equation (4) allows us to study transitivity of distance graphs by establishing an isomorphism with transitive closures of proximity graphs. Thus, the distance closure $D^T(X)$ is obtained via compositions of f and g:

$$d_{ij}^{(\eta)} = \underset{k}{f}\, g\left(d_{ik}, d_{kj}^{(\eta-1)}\right) \quad \& \quad d_{ij}^{T} = f\left(d_{ij}, d_{ij}^{(2)}, \ldots, d_{ij}^{(\kappa-1)}, d_{ij}^{(\kappa)}\right), \tag{6}$$

where, because of the isomorphism, κ is the same as for the transitive closure (Eq. 2). The adjacency matrix $D^{\eta}(X)$ measures the shortest distance for paths including η connections, while the distance closure $D^{T}(X)$ accounts for the shortest path length *up to* κ links. For distance graphs, the transitivity criterion is defined by each algebraic structure $([0, \infty), f, g)$:

$$d_{ij} \leq \underset{k}{f}\, g(d_{ik}, d_{kj}) \quad \forall x_i, x_j, x_k \in X. \tag{7}$$

The distance closure $D^{T}(X)$ is transitive by construction, but generally only a subset of edges $D(X)$ obey Eq. (7).

2.3 Shortest-Path, Metric and Ultrametric Closures

The general transitive and distance closures of Sects. 2.1 and 2.2 yield a number of well-known cases used in network science [24, 26]. When $f \equiv \min$ (or $\vee \equiv \max$ in proximity graphs), we have the large class of *shortest-path closures*, $D^{T,g}(X)$, for any distance function g (or $\wedge$ in proximity graphs), as the closure selects the minimum path with length given by g. This leads to a *generalized triangle inequality* [24] as a transitivity criterion:

$$d_{ij} \leq g(d_{ik}, d_{kj}) \quad \forall x_i, x_j, x_k \in X. \tag{8}$$

For instance, when $g \equiv +$, we obtain the familiar *metric closure*, $D^{T,m}(X)$, where the length of the path is obtained by summing the distance edge weights. Similarly, when $g \equiv \max$, we instead obtain the *ultrametric closure*, $D^{T,u}$, where the length of the path is obtained by the maximum distance weight in path (the weakest link).

Many other shortest-path distance closures—and thus different path length measures and transitivity criteria—can be usefully employed in network science [26]. Here we exemplify the approach with these two well-known cases because the metric closure is the most common way to compute shortest path on weighted graphs, and the ultra-metric closure is the lower bound of distance closures [24].

2.4 Distance Backbone Subgraph

The *distance backbone* $B^{g}(X)$ of a distance graph $D(X)$ is the invariant subgraph under a shortest-path distance closure $D^{T,g}(X)$ with $f \equiv \min$ and some g [24]. It is sufficient to compute all shortest paths in $D(X)$ given a path length measure g. The

distance backbone is invariant because its edges are the ones that obey the generalized triangle inequality (Eq. 8) and are thus called *triangular* edges. That is, the distance backbone is defined by edges that have the same weight in the shortest-path closure:

$$b_{ij}^{g} = \begin{cases} d_{ij}, & \text{if } d_{ij} = d_{ij}^{T,g} \\ \infty, & \text{if } d_{ij} > d_{ij}^{T,g} \end{cases}, \quad \forall x_i, x_j \in X, \tag{9}$$

where $d_{ij}^{T,g}$ are the adjacency matrix weights of the distance closure graph $D^{T,g}(X)$. The edges that break the generalized triangle inequality are called *semi-triangular* and are not on the backbone, i.e. $b_{ij}^{g} = \infty$. If (and only if) an edge between x_i and x_j is semi-triangular (i.e., not present on the backbone), there exists a shorter indirect path (i.e., which is present on the backbone) connecting them via some x_k [24].

The metric ($g \equiv +$) and ultrametric ($g \equiv \max$) backbones of distance graph $D(X)$ are denoted by $B^m(X)$ and $B^u(X)$, respectively. Similarly, edges on these backbones are called metric and ultrametric, while those off are known as semi-metric and semi-ultrametric, respectively [24].

3 Directed Distance Backbone

Here we extend the concept of distance backbone by relaxing the symmetry constraint of distance functions, thus considering distance graphs $D(X)$ where $d_{ij} \neq d_{ji}$, or directed distance graphs. As summarized above, distance backbones exist when enforcing a generalized triangle inequality (Eq. 8) as a transitive closure criterion. This is the same as computing all shortest paths of $D(X)$ using a measure of path length determined by g.

Computation of the all pairs shortest path problem (APSP) for undirected weighted graphs with $g \equiv +$ is straightforward using the Dijkstra algorithm [5] (though it can also be computed with the distance product directly via Eqs. (2) and (6) [26, 28]). Since all shortest-path distance closures are based on setting $f \equiv \min$ in Eqs. (6) and (7), they can also be computed as a APSP problem by adjusting the chosen algorithm with a different path length measure for each g used, such as $g \equiv \max$ for the ultrametric backbone [24].

We also know that the standard triangle inequality, Eq. (8) with $g \equiv +$, is valid for directed distances [18]. This way, the APSP of directed distance graphs based on this transitivity criterion can also be computed via the Dijkstra algorithm [5] or the distance product [28]. Indeed, the methodology of closures in complex networks is found to be applicable to both undirected and directed weighted graphs. The latter is shown in the real world examples of Sect. 4.

3.1 Redundancy and Robustness

The fraction of edges in the backbone

$$\tau^g(X) = \frac{|B^g(X)|}{|D(X)|} = \frac{|\{d_{ij} : d_{ij} = d_{ij}^{T,g}\}|}{|\{d_{ij}\}|} \; \forall_{x_i,x_j \in X: i \neq j} \tag{10}$$

measures the proportion of triangular (or topologically invariant) edges, while its complement $\sigma(X) = 1 - \tau(X)$ quantifies the proportion of semi-triangular edges. The latter measures the structural redundancy of complex networks given a specific transitivity criterion (Eq. 8). That is, the edges that are redundant for shortest-path computation given the path length measure g chosen. Note that due to the introducing of directionality, now τ^g must be computed for *all* entries of the adjacency matrix, and not just for the upper or lower diagonal as previously done for the undirected case [24].

If a network has a small backbone (small τ^g), most of its edges are semi-triangular and do not affect the shortest path distribution. This way, *random* attacks would most likely not interfere with the backbone itself, a robustness[1] that can be inferred from the measure of topological redundancy $\sigma^g(X)$.

4 Experimental Analysis

Now we investigate the backbone of nine real-world networks pertaining to three distinct domains: biomedical, social, and man-made technological systems. Here we discuss in more detail the backbones of a giraffe social network [3], the U.S airport transportation system [23, 24], and the bike-sharing system of the City of London [21]. Additional details for this and the remaining networks can be found in the accompanying digital supplemental material. Descriptive data for each directed weighted graph, and the size of their respective metric and ultrametric backbone are shown in Table 1.

4.1 Giraffe Socialization

Evidence suggests that giraffes have complex social structures, with females having social preferences and suggestive that adult giraffes have friendships beyond only mother-child interactions [3]. We analyze a network of social interaction of captive giraffes at the San Diego Zoo's Wild Animal Park. The original observational study included 6 adult female Rothschild's giraffe (*Giraffa camelopardalis*) housed in

[1] A finer characterization of robustness in terms of edge properties [25] in the case of directed graphs is left for future work.

Table 1 Topological invariance of weighted directed graphs modeling real-world systems

	Network	$\|X\|$	$\|d_{i\neq j}\|$	δ	τ^m	τ^u	τ^u/τ^m
Biomedical	Co-morbidity risk	95	8,930	1.0	47.44	2.17	4.57
	Drug interaction	412	2,966	1.75e-2	59.00	40.49	68.63
	Species-Species inter.	10,578	18,529	1.66e-4	99.47	99.46	99.99
Social	Giraffe socialization	6	30	1.0	76.67	30.00	39.13
	Telephone calls	322	609	5.89e-3	91.63	84.89	92.64
Technological	Bicycle trips (min. 7)	725	53,118	0.1	59.53	2.75	4.62
	U.S. Airports 2006	1,075	18,906	1.64e-2	27.59	18.99	68.83
	Water pipes	1,836	2,351	6.98e-4	99.62	95.83	96.20

The number of nodes $|X|$ and edges $|d_{i\neq j}|$ are used to compute the network density δ. The relative size of the metric (τ^m) and ultrametric (τ^u) backbones are presented as percentages

a single herd. In the study, they were observed 5 mornings a week for a total of 300 d, and the behavior of each subject was recorded for a 20-min focal sample in random order. Data on nearest neighbor and proximity (measured at 2 neck lengths) were collected at 1-min intervals for the focal subject. Affiliative social interactions involving the focal subject were recorded and included: approach, necking, head rub, bumping, social exam, muzzle, co-feed, and sentinel (details in [3]). In total, 600 h of observation time and 2,748 affiliative interactions were observed.

In the social network directed edge weights represent the frequency in which giraffe x_i interacts with giraffe x_j as a measure of similarity p_{ij} (see Fig. 1a). This is a small network containing only 6 nodes and fully connected with 30 directed edges (density $\delta = 1.0$). The metric backbone consists of 23 edges ($\tau^m = 76.7\%$) and the ultrametric of only 9 ($\tau^u = 30\%$) edges. Interestingly, the metric backbone completely removes the edge between giraffes *Yanahmah* and *Chokolati*, both the oldest giraffes in the herd. In the metric backbone the mother-daughter relationships are also kept between *Yanahman-Ykeke* and *Chokolati-Chinde*. In other words, and as previously noted for human contact networks [9], the backbone preserves the hierarchical structure of social networks.

4.2 London Bike-Sharing Trips

The SARS-Cov-2 pandemic caused unprecedented shifts in urban mobility with bike-share systems having a significant increased in demand in several major capitals

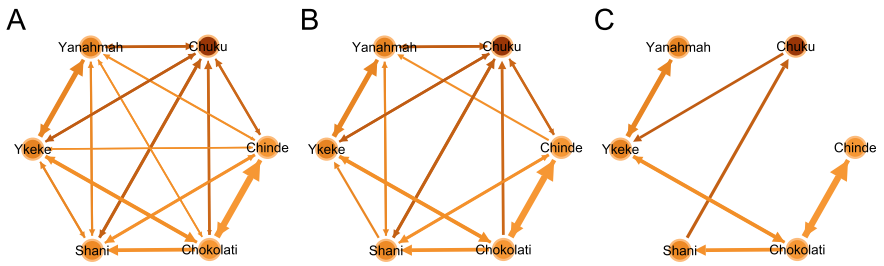


Fig. 1 Giraffe socialization network in the San Diego Zoo [3]. **a** Directed distance graph; **b** metric backbone subgraph; and **c** ultrametric backbone subgraph. The original distance graph contains 30 edges, while the metric backbone contains 23 (76.7%) and the ultrametric backbone only 9 (30%) edges. Plotted with Gephi [4]

[19, 27]. We analyze the City of London's bike-sharing system, available through the Transport for London Open Data API and previously analyzed in Munoz-Mendez et al. [21]. Data contains records for each unique bicycle and their rental transactions, including timestamped information on which bike-sharing station it was picked up and then returned in a network of 770 stations through the city. A month's worth of bike-sharing transactions is analyzed, from June to July 2014. Transactions that started or ended in a repair station, as well as stations with too few transactions, were discarded. This means we only included stations that accounted for 75% of all transactions (i.e., a minimum of 7 monthly trips per station), which in turn resulted in 726 bike-sharing stations and 948,339 bike-sharing transactions.

In this network a node represents a bike-sharing station, x_i, and edges are weighted by the average trip duration between stations as a directed distance, d_{ij}. This network has 725 nodes and 53,118 nodes (density $\delta = 0.1$). The metric and ultrametric backbones consist of $\tau^m = 59.53\%$ and $\tau^u = 2.75\%$, respectively, of the directed network. Along with the co-morbidity risk network, the bike sharing network has one of the largest differences in the sizes of the metric to the ultrametric backbone ($\tau^u/\tau^m = 4.6\%$). This means that a directed attack on the metric backbone will have a small impact on ultrametric backbone and thus in the distribution of shortest paths [24]. In other words, the network of the bike-sharing system for the City of London is very robust to directed attacks, translated to the possible closure of bike-sharing stations or street changes that cyclists use (Fig. 2).

4.3 U.S. Airport Transportation

This network is the domestic nonstop segment of the U.S. airport transportation system for the year 2006, retrieved from http://www.transtats.bts.gov. Each node is an airport, and edge weights are the normalized number of passengers traveling between two airport-nodes. This network was analyzed in Simas et al. [24] and is a reconstruc-

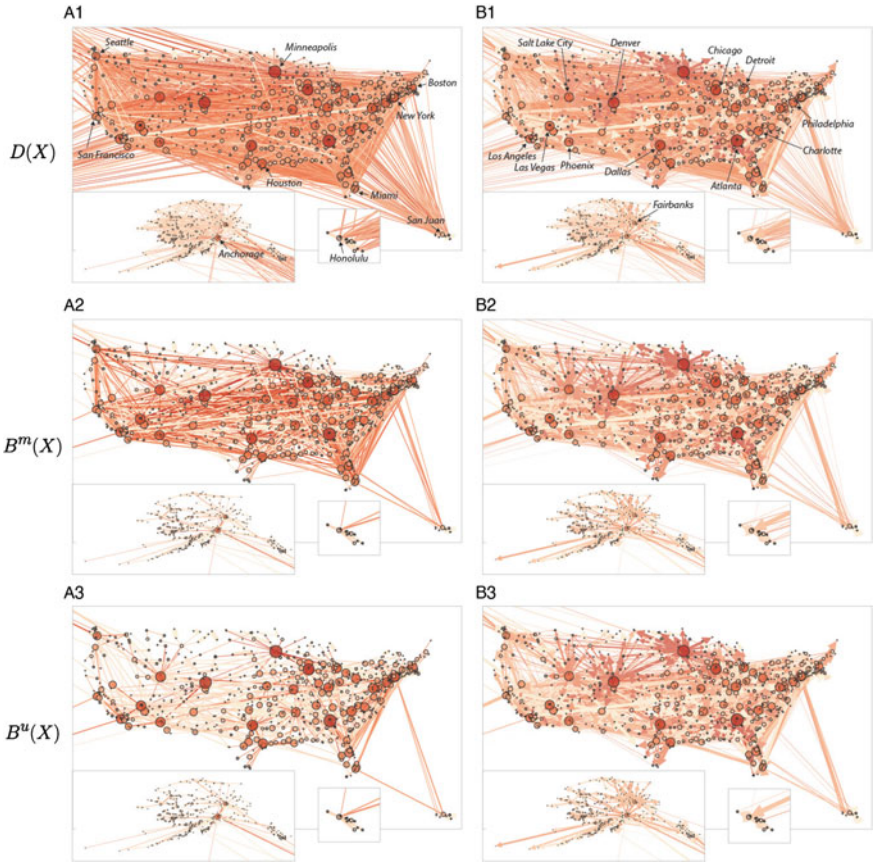


Fig. 2 Domestic nonstop segment of the U.S. airport transportation system [23]. **(a)** Undirected distance graph with its respective **(b)** metric, and **(c)** ultrametric backbone subgraphs [24]. **(d)** Directed distance graph with its respective **(e)** metric, and **(f)** ultrametric backbone subgraphs. The original directed (undirected) distance graph contains 18,906 (11,973) edges. From those, 27.59% (16.14%) are in the metric backbone, and 18.99% (8.98%) in the ultrametric backbone. The difference in number of edges between the undirected and directed representation comes from the fact that 5040 (26.65%) of all flights are only in one direction. Network plotted with Gephi [4]

tion of the one used by Serrano et al. [23]. Differently from previous work, however, here we consider directionality in the flow of passengers as 5,040 (approximately 27%) of all flights are only in one direction In other words, flight routes may include stops in multiple airports from initial to final destination, and not necessarily contain a direct return to the initial departure airport. Airports in the American Samoa, Guam, Northern Marianas, and Trust Territories of the Pacific Islands have been removed from the analysis. This is a large but relatively sparse network with 1075 nodes and 18,906 edges (density δ =1.64e−2). The relative size of the metric and ultrametric backbone are $\tau^m = 27.59\%$ and $\tau^u = 18.99\%$, respectively, (see Table 1).

5 Discussion

Directionality and strength of interactions are relevant properties of real complex networks. The structure of such networks can be reduced in a principled manner, while preserving the entire distribution of shortest paths (for a given length measure g), with the computation of the distance backbone.

In the nine networks we analyzed, we found that the size of the metric backbone ranges from 27.59 to 99.6%—three networks have metric backbones above 92% of the distance graph. Ultrametric backbones range from 2.17 to 99.5%, with two networks having ultrametric backbones above 95.8%. In contrast, for undirected graphs studied in Simas et al. [24] the metric (ultrametric) backbones range from 1.75 to 83.59% (0.2–78.45%), which shows a substantial increase in the size of backbones due to directionality. A direct comparison can be made for the U.S. airports network. Its undirected representation has a relative size of the metric and ultrametric backbone of $\tau^m = 16.14\%$ and $\tau^u = 8.98\%$, respectively [24]. Here, we found that the relative size of the metric and ultrametric backbone are $\tau^m = 27.59\%$ and $\tau^u = 18.99\%$, respectively (see Table 1). This increase is likely due to the fact that the closure for directed graphs does not lead to a complete graph—unlike what happens to connected undirected graphs. In other words, having many connections in only one direction (approximately 27% in this case) can make them necessary for shortest paths irrespective of the edge weight, which emphasized the importance of directionality when studying real-world networks. The large difference between the size of backbones in directed and undirected graphs warrants future studies of the effect of directionality vis a vis various topological parameters.

The metric and ultrametric backbones of the networks we analyzed (Table 1) exemplify networks which are robust to random edge removal, as is the case of the comorbidity risk and bicycle trips networks, for having a smaller backbone (small τ^g). On the other hand, the species-species interaction network and water pipes networks have a large τ^g and little redundancy. That is, the backbone is most of the network, suggesting that they mostly contain necessary interaction information, or were perhaps optimized to minimize the cost of implementing redundant edges, being susceptible to random edge removal or failure. In the case of the water pipe network, little redundancy is expected because its distance weights represent an actual physical distance between nodes, which must conform to a naturally metric topology. Thus, it is an expected result that its metric backbone is almost the entire distance graph (99.6%). This highlights the fact that semi-metric (and semi-triangular) behavior can only occur in high-dimensional spaces [24]. In contrast, the metric backbone of the passenger traffic between U.S. airports is only 27.59%, making its shortest path distribution very robust to random attacks, as the odds of randomly removing semi-metric edges are much higher than removing metric ones that contribute to the backbone. The precise impact in the shortest path distribution for those networks requires the computation of edge distortion [24, 25] and is left for future work.

6 Conclusion

We introduced directionality to study shortest-path redundancy in *weighted directed* graphs via a novel directed distance backbone subgraph, the computation of which we showed to be feasible. This consideration brings improvement over other sparsification methods that considers only undirected networks [20, 24] or that treat incoming and outgoing edges independently [23]. We focused on the metric (where $g \equiv +$) and the ultrametric (where $g \equiv \max$) backbones, but the methodology is applicable for any length measure g, allowing other backbones to be considered in the future.

We applied the methodology to study redundancy of a variety of real-world weighted directed graphs modeling biomedical, social, and technological systems. The size of the metric (ultrametric) backbone ranges from 27 to 99% (2–99%), but is typically much smaller than the original distance graph. However, the size of the directed backbones observed are larger than the undirected backbones previously reported, emphasizing the difference in shortest-path robustness for the two different classes of graphs. The comparison using the same underlying U.S. airports network is particularly illuminating. We found that both the metric and the ultrametric backbone for the directed graph are larger than the ones for the undirected version—71% and 112%, respectively. Thus, asymmetric airline seat capacity between cities (27% of all connections exist only in one direction) has a large impact on shortest paths between them. This exemplifies the importance of our contribution in the study of distance backbones for directed networks, which will lead to a study with additional networks in the future.

The methodology further allows us to infer the robustness of shortest path distributions to random attack, via the relative size of the metric and ultrametric backbones. This can aid the design of more resilient social and technological systems or the identification of key evolutionary properties in biomedical systems. We are confident that the study of directed distance backbones can help the understanding and control of a variety of complex multivariate systems where both strength and directionality of interactions is key.

Acknowledgements This work was partially funded by the National Institutes of Health, National Library of Medicine Program, grant 01LM011945-01, and the Fundação para a Ciência e a Tecnologia, grants PTDCMEC-AND-30221-2017 and DSAIPA/AI/0102/2019 (LMR and RBC). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

References

1. Alves, L.G.A., Rybski, D., Ribeiro, H.V.: Scientific Rep. **11**(1), 22918 (2021)
2. Ball, B., Newman, M.E.: Netw. Sci. **1**(1), 16–30 (2013)
3. Bashaw, M.J., Bloomsmith, M.A., Maple, T.L., Bercovitch, F.B.: J. Comp. Psychol **121**(1), 46–53 (2007)

4. Bastian, M., Heymann, S., Jacomy, M.: In: International AAAI Conference on Weblogs and Social Media (2009)
5. Brandes, U., Erlebach, T.: SSBM 3418 (2005)
6. Colizza, V., Barrat, A., Barthélemy, M., Vespignani, A.: Proc. Nat. Acad. Sci. **103**(7), 2015–2020 (2006)
7. Conrad, M.: BioSystems **24**(1), 61–81 (1990)
8. Correia, R.B., Almeida, J., Wyrwoll, M., Julca, I., Sobral, D., Misra, C., Guilgur, L., Schuppe, H., Silva, N., Prudêncio, P., Nóvoa, A., Leocádio, A., Bom, J., Mallo, M., Kliesch, S., Mutwil, M., Rocha, L.M., Tüttelmann, F., Becker, J., Navarro-Costa, P.: bioRxiv:2022.03.02.482557 (2022)
9. Correia, R.B., Barrat, A., Rocha, L.M.: bioRxiv:2022.02.02.478784 (2022)
10. Correia, R.B., Li, L., Rocha, L.M.: In: Pacific Symposium on Biocomputing. vol. 21, pp. 492–503 (2016)
11. Galvin, F., Shore, S.: Am. Math. Monthly **98**(7), 620–623 (1991)
12. Gates, A.J., Correia, R.B., Wang, X., Rocha, L.M.: Proc. Nat. Acad. Sci. **118**(12), e2022598118 (2021)
13. Gates, A.J., Rocha, L.M.: Sci. Rep. **6**, 24456 (2016)
14. Gates, A.J., Wood, I.B., Hetrick, W.P., Ahn, Y.Y.: Sci. Rep. **9**, 8574 (2019)
15. Ings, T.C., Montoya, J.M., Bascompte, J., Blüthgen, N., Brown, L., Dormann, C.F., Edwards, F., Figueroa, D., Jacob, U., Jones, J.I., Lauridsen, R.B., Ledger, M.E., Lewis, H.M., Olesen, J.M., Van Veen, F.F., Warren, P.H., Woodward, G.: J. Animal Ecol. **78**(1), 253–269 (2009)
16. Karsai, M., Kivelä, M., Pan, R.K., Kaski, K., Kertész, J., Barabási, A.L., Saramäki, J.: Phys. Rev. E **83**(2), 025102 (2011)
17. Klir, G., Yuan, B.: Prentice Hall. New Jersey (1995)
18. Lawvere, F.: Seminario Matematico e Fisico di Milano **43**, 135–166 (1973)
19. Li, Q., Xu, W.: Cambridge J. Reg. Econ. Soc. (2022)
20. Mercier, A.M., Scarpino, S.V., Moore, C.: arXiv:2111.02449 (2021)
21. Munoz-Mendez, F., Han, K., Klemmer, K., Jarvis, S.: Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers, pp. 1015–1023. Association for Computing Machinery, New York (2018)
22. Salgado, A., Li, W., Alhasoun, F., Caridi, I., Gonzalez, M.: Appl. Netw. Sci. **6**(1), 43 (2021)
23. Serrano, M.A., Boguñá, M., Vespignani, A.: Proc. Nat. Acad. Sci. USA **106**, 6483–6488 (2009)
24. Simas, T., Correia, R.B., Rocha, L.M.: J. Complex Netw. **9** (2021)
25. Simas, T., Rocha, L.M.: 2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, pp. 175–179 (2012)
26. Simas, T., Rocha, L.M.: Netw. Sci. **3**(2), 227–268 (2015)
27. Song, J., Zhang, L., Qin, Z., Ramli, M.A.: Phys. A **592**, 126819 (2022)
28. Zwick, U.: J. ACM (JACM) **49**(3), 289–317 (2002)

Community Structure

Structure of Core-Periphery Communities

Junwei Su and Peter Marbach

Abstract It has been experimentally shown that communities in social networks tend to have a core-periphery topology. However, there is still a limited understanding of the precise structure of core-periphery communities in social networks including the connectivity structure and interaction rates between agents. In this paper, we use a game-theoretic approach to derive a more precise characterization of the structure of core-periphery communities.

Keywords Core-Periphery communities · Social networks · Game-Theoretic model

1 Introduction

Experimental results have shown that communities in social networks tend to have a core-periphery topology consisting of two types of agents, core agents and periphery agents, that differ in their objectives for participating in the community [12, 13]. The objective of periphery agents is to obtain content that is of interest to them. As a result, periphery agents follow other agents in the community to obtain the content that is of most interest to them. The objective of the core agents is to attract followers, and attention, from the periphery agents. To achieve their objective, core agents aggregate/collect content from the community and make it available to the periphery agents [3, 4, 9]. These two different objectives lead to a community structure where the core agents follow periphery agents in the community in order to collect content, and the periphery agents connect with the core agents and other periphery agents in order to obtain the content they are interested in [12, 13].

In this paper, we provide a mathematical model that allows us to derive these structural properties of core-periphery communities in social networks in a formal manner. The results of our analysis provide a precise characterization of the connec-

J. Su · P. Marbach (✉)
Dept. of Computer Science, University of Toronto, Toronto, Canada
e-mail: marbach@cs.toronto.edu

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_12

tivity structure, and interactions rates, of core-periphery communities. This allows us to use the structural properties to formally design algorithms for social networks such as (core-periphery) community detection algorithms, and content recommendation algorithms based on the users' interaction with a content item.

For our analysis, we use a game-theoretic framework where we assume that agents in the community make the decision on which other agents to interact with in a manner that maximizes their own objective. The proofs of the results presented in paper are provided in [11].

2 Related Work

Experimental studies have shown that communities in social networks tend to have a core-periphery topology with two types of agents, core agents and periphery agents [12, 13], where the core agents collect (aggregate) content from the community, and make it available to the periphery agents. While experimental studies show that this structure exists, they do not provide a (precise) characterization of the connectivity structure, as well as the interaction rates between the different agents. The goal of this paper is to provide such a characterization. An interesting result from the experimental studies is that core-periphery communities in online networks tend to have a small set of core agents, typically in the order of 1-6 core agents [12, 13].

Theoretical results on the structural properties of communities were obtained in the context of network formation games[1, 2, 5, 7, 8, 10]. Jackson and Wolinsky presented one of the first, and most influential, analyses of network formation games [8]. For their analysis, Jackson and Wolinsky assume that a) agents in the network obtain a benefit from having paths to other agents and b) pay a cost for each direct connection (link) that they have with another agent. The benefit that an agent obtains from another agent is discounted by a factor δ^d, where d is the length of the path (distance) between the two agents and δ, $0 < \delta < 1$, is a discount factor. Assuming bi-directional links, Jackson and Wolinsky show that the star topology is a Nash equilibrium for the game that they consider. The paper by Jackson and Wolinsky makes several important contributions. First, it shows that a game-theoretic model can be used to derive the structural properties of communities. Second, the star-topology of the Nash equilibrium suggests that a core-periphery topology might indeed naturally emerge as the community topology in social networks.

Bala and Goyal use in [1] the model of Jackson and Wolinsky, except that they consider unidirectional links instead of bidirectional links. For this model, Bala and Goyal show that the star topology again emerges as a Nash equilibrium, and side payments from the periphery agents to the core agent are required for the star topology to emerge as a Nash equilibrium.

A limitation of the analysis by Jackson and Wolinsky is they assume a homogeneous set of periphery agents. Hegde et al. consider in [6] a more general model that allows for a heterogeneous population of periphery agents where periphery agents differ in the benefit they obtain from other agents. To model the heterogeneous pop-

ulation, Hedge et al. embed agents in a Euclidean space. Agents that are close (in the Euclidean distance) to a given agent provide a higher benefit to the agent compared with agents that are further away. Assuming that all agents have the same number of connections, Hegde et al. consider the game where agents choose connections to other agents in order to maximize their own benefit. For this model, Hedge et al. show that there exists a Nash equilibrium. However, due to the complexity of the model, Hedge et al. were not able to derive and characterize the structural properties of the Nash equilibrium.

In summary, existing mathematical models are either too simple (as it is the case for [1, 8]) and lead to a core-periphery community structure that does not accurately reflect the community structures observed in real-life social networks; or they are too complex and can not be used to derive the structural properties of core-periphery communities (as it is the case for [6]). The goal of this paper is to propose a model that is simple enough to characterize the structural properties of a community, and yet is complex enough to lead to results that accurately reflect the community structures that are observed in real-life social networks and can be used to design algorithms for social networks.

3 Core-Periphery Community

We use the following model for our analysis.

Core-Periphery Community C: A core-periphery community consists of a set of core agents and periphery agents. To simplify the notation and analysis, we assume that there exists a single core agent y_c. This assumption is also motivated by the experimental results which show that core-periphery communities tend to have a small set of core agents, typically in the order of 1-6 core agents [12, 13]. The results that we obtain for a single core agent can be extended to the case of multiple core agents. Using this assumption, a core-periphery community C is then given by a core agent y_c and a set C_p of periphery agents, i.e. we have that $C = C_p \cup \{y_c\}$.

Periphery Agents C_p: For our analysis, we assume that periphery agents both produce and consume content. In addition, we assume a heterogeneous set of periphery agents, where agents differ in the content (topics) that they are interested in. To model this situation we use a similar approach as in [6], and assume a "topic space" that specifies how closely two topics are related to each other. The topic space that we consider is given by the interval $I_C = [I_0 - L_C, I_0 + L_C] \subset R$. Each periphery agent is then characterized by its main interest $y \in I_C$, which is the topic that the agent is most interested in. For content production, we assume that each agent produces content on the topic that is their main interest. For the content assumption, we use the following model. The probability that a periphery agent with main interest y is interested in a content produced by an agent with main interest x is given by

$$p(x|y) = f(||x - y||), \qquad x, y \in I_C, \tag{1}$$

where $f: [0, \infty) \mapsto [0, 1]$ is a decreasing concave function. Note that this definition implies that periphery agents are more interested in content that is produced by agents whose main interest is close to their own main interest.

For our analysis we assume that the (main interests of the) periphery agents are"uniformly" distributed in the interval I_C, with equal distance δ between two agents. That is, we assume that the set of periphery agents C_l consists of K agents with main interests y_k, $k = 1, ..., K$, given by

$$C_p = \{y_1, ..., y_K\} \subset I_C = [I_0 - L_C, I_0 + L_C],$$

with $y_1 = I_0 - L_C$ and $y_{k+1} = y_k + \delta$, $k = 1, ..., K - 1$, where $\delta = \frac{2L_C}{K-1}$. In the following, we identify periphery agents by their main interest y.

4 Utility of Periphery Agents

For our analysis, we assume that periphery agents can obtain content from three different sources: a) directly from other periphery agents by following these agents, b) indirectly from the core agent, where the content provided by the core agent is the content that the core agent obtains by following periphery agents in the community, and c) by following content platforms outside the community.[1]

We use the following notation to characterize the following rates between the agents, and the following rates of periphery agents to content platforms outside the community.

Let $\mu_c(y)$ be the rate with which core agent y_c follows periphery agent $y \in C_p$, and let $\mu_c = (\mu_c(y))_{y \in C_p}$ be following rate vector of the core agent y_c to all periphery agents $y \in C_p$.

Similarly, let $\mu(y) = (\mu(z|y))_{z \in C \setminus \{y\}}$ be the following rate vector of periphery agent $y \in C_p$ to all other agents $z \in C \setminus \{y\}$ in the community. Furthermore, let $\lambda(y)$ be the rate with which periphery agent y follows content platforms outside the community, and let $\mu_p(y) = (\mu(y), \lambda(y))$ be the overall following rate vector of periphery agent $y \in C_p$.

Finally, let $\Lambda_p = (\mu_p(y))_{y \in C_p}$ be the following rate vectors of all periphery agents.

We next define the utilities that periphery agents obtain from following a) other periphery agents directly, b) the core agent, and c) content platforms outside the community.

Utility from Following a Periphery Agent Directly: We first define the utility that a periphery agent y obtains by following another periphery agent z with rate $\mu(z|y)$. Suppose that agent y receives a reward of value 1 for each content item that is of interest to agent y. Furthermore, suppose that each content item that agent y receives

[1] For example, users on Twitter will generally also get content from additional content platforms such as other news or other social media sites.

incurs a processing (reading) cost c, $0 < c < 1$. If agent y receives content from agent z with delay $d(z|y)$, then the (expected) utility rate of agent y is given by

$$U_{C,p}(z|y) = r_p\Big[p(z|y)e^{-\alpha d(z|y)} - c\Big]I(\mu(z|y)), \tag{2}$$

where $p(z|y)$ is the probability that a content item of agent z is of interest to agent y, $I(\mu(z|y))$ is the indicator function of whether agent y follows agent z and is equal to 1 if $\mu(z|y) > 0$, r_p is the rate at which z produces content, and α is a given constant that captures how sensitive the content produced by agent z is towards delay. This utility function captures the intuition that the longer the delay $d(z|y)$ is, the lower is the utility of the received content.

For our analysis we define the delay $d(z|y)$ by

$$d(z|y) = \frac{1}{\mu(z|y)},$$

where $\mu(z|y)$ is the rate with which agent y follows agent z. Note that this definition implies that the higher the rate with which agent y follows agent z, the lower the delay $d(z|y)$ will be.

Utility from Following the Core Agent: We next define the utility that a periphery agent $y \in C_p$ receives from content of periphery agent $z \in C_p$, when the content is received through the core agent y_c. For this, suppose that the core agent y_c follows periphery agent z with rate $\mu_c(z)$, and periphery agent y follows the core agent y_c with rate $\mu(y_c|y)$. The total delay with which agent y receives content from agent z through y_c is given by

$$d(z|y_c) + d(y_c|y) = \frac{1}{\mu_c(z)} + \frac{1}{\mu(y_c|y)}.$$

Using this result, the utility rate of periphery agent y for getting the content of agent z via the core agent y_c is given by

$$U_{C,c}(z|y) = r_p\Big[p(z|y)e^{-\alpha\left(\frac{1}{\mu_c(z)}+\frac{1}{\mu(y_c|y)}\right)} - c\Big]I(\mu_c(z))I(\mu(y_c|y)). \tag{3}$$

Utility from Following other Content Platforms: Finally we define the utility that a periphery agent $y \in C_p$ obtains by getting content from other platforms. For this, we assume that the overall rate (over all content platforms) at which new content items are generated by the other platforms is equal to $r_0 > 0$, and that each content item is of interest to agent y with probability B_0. If periphery agent y follows other content platforms with rate $\lambda(y)$, then the corresponding utility rate is given by

$$U_0(y) = r_0\Big[B_0 e^{-\frac{\alpha}{\lambda(y)}} - c\Big]I(\lambda(y)). \tag{4}$$

5 Agents' Decisions and Interactions

In this section, we model the interaction among agents in a core-periphery community where we assume that each agent decides on its following rates in order to maximize its own objective function.

5.1 Core Agent's Decision Problem

Recall from Sect. 1 that the objective of the core agent y_c is to attract attention from periphery agents by aggregating/collecting content that is of most interest to the periphery agents [3, 4, 9]. We formulate the resulting decision problem of the core agent as an optimization problem as follows.

Recall that Λ_p is the rate allocation vector over the all periphery agents $y \in C_p$, and $\mu_c = (\mu_c(y))_{y \in C_p}$ is the rate allocation of the core agent y_c. Furthermore recall Eq. (3) that defines the utility $U_{C,c}(z|y)$ that periphery agent y obtains from getting content of agent z through the core agent y_c. For a given rate allocation Λ_p of the periphery agents, the decision problem of the core agent y_c is given by the following optimization problem $OPT(\mu_c|\Lambda_p)$,

$$\begin{aligned} \underset{\mu_c}{\text{maximize}} \quad & \sum_{y \in C_p} \sum_{z \in C_p \setminus \{y\}} U_{C,c}(z|y) \\ \text{subject to} \quad & \sum_{y \in C_p} \mu_c(y) \leq M_c, \\ & \mu_c(y) \geq 0, \ \ y \in C_p, \end{aligned} \tag{5}$$

where M_c is a constraint on the total rate that the core agent can allocate to follow periphery agents $y \in C_p$. This constraint reflects that the core agent y_c has limited resources (time) to follow periphery agents in the community. Note that the optimization problem $OPT(\mu_c|\Lambda_p)$ captures the goal of the core agent: the core agent y_c wants to use its limited resources to attract attention from the periphery agents by aggregating content that is of most interest to the periphery agents.

5.2 Periphery Agents' Decision Problem

Recall that the objective of a periphery agent y is to obtain as "much content that is of interest as possible". A periphery agent can achieve this goal by following other periphery agents directly, by getting content through the core agent y_c, and by getting content from other content platforms. We formulate the resulting decision problem of a periphery agent as follows.

Let $\mu_c = (\mu_c(y))_{y \in C_p}$ be a given rate allocation of the core agent y_c, and let

$$U_p(\mu_p(y)|\mu_c, y) = \sum_{z \in C_p \setminus \{y\}} U_{C,c}(z|y) + \sum_{z \in C_p \setminus \{y\}} U_{C,p}(z|y) + U_0(y) \tag{6}$$

be the total utility rate that periphery agent y obtains under its rate allocation $\mu_p(y)$ and the allocation μ_c of the core agent. For a given rate allocation μ_c of the the core agent, the decision problem of the periphery agent y is given by the following optimization problem $OPT(\mu_p(y)|\mu_c, y)$,

$$\begin{aligned} \underset{\mu_p(y)}{\text{maximize}} \quad & U_p(\mu_p(y)|\mu_c, y) \\ \text{subject to} \quad & \mu(y_c|y) + \lambda(y) + \sum_{z \in C_p \setminus \{y\}} \mu(z|y) \le M_p, \\ & \mu(z|y), \lambda(y), \mu(y_c|y) \ge 0, \ z \in C_p \setminus \{y\}, \end{aligned} \tag{7}$$

where $M_p > 0$ is a constraint on the total rate that periphery agent y can allocate. To simplify the notation and analysis, we assume that the rate budget M_p is the same for all periphery agents.

5.3 Nash Equilibrium

The optimal solution of the maximization problem $OPT(\mu_c|\Lambda_p)$ of the core agent depends on the given rate allocation Λ_p of the periphery agents. Similarly, the optimal solution of the maximization problem $OPT(\mu_p(y)|\mu_c, y)$ of periphery agent y depends on the given rate allocation μ_c of the core agent. This coupling creates a strategic interaction (game) between the agents in the community. A Nash equilibrium for the resulting game is given as follows.

Let $\Lambda = (\mu_c, \Lambda_p)$ be the rate allocation vector that characterizes the rate allocation μ_c of the core agent, and the rate allocation vector $\Lambda_p = (\mu_p(y))_{y \in C_p}$ over all periphery agents.

Definition 1 An allocation $\Lambda^* = (\mu_c^*, \Lambda_p^*)$ is a Nash equilibrium if we have that

$$\mu_c^* = \underset{\mu_c \ge 0}{\text{argmax}}\, OPT(\mu_c|\Lambda_p^*) \quad \text{and} \quad \mu_p^*(y) = \underset{\mu_p(y) \ge 0}{\text{argmax}}\, OPT(\mu_p(y)|\mu_c^*, y).$$

Definition 1 states that under a Nash equilibrium $\Lambda^* = (\mu_c^*, \Lambda_p^*)$ no agent is able to increase the value of their objective function by unilaterally changing their allocation. In Sect. 5.4 we show that there exists a unique Nash equilibrium, and in Sect. 6 we characterize the structural properties of the Nash equilibrium.

5.4 Existence of Unique Nash Equilibrium

For our analysis we make the following assumptions.

Assumption 1 For all periphery agents $y \in C_p$ we have that

$$\sum_{z \in C_p \backslash \{y\}} [p(z|y) - c] > 0 \quad \text{and} \quad \sum_{z \in C_p \backslash \{y\}} [p(y|z) - c] > 0.$$

Assumption 1 states that if the content of agent y is received by all other agents $z \in C_p \backslash \{y\}$ without delay, then the resulting total utility is positive. Similarly, if agent y receives content from all other agents $z \in C_p \backslash \{y\}$ without delay, then the resulting total utility y is positive.

In addition we make the following assumption for the processing cost c.

Assumption 2 We have that $c > e^{-1}$.

Assumption 2 implies that if agent y follows agent z with rate $\mu(z|y) < \alpha$, then the utility from content received through agent z will be negative. As a result we have that if agent y follows agent z with a positive rate $\mu(z|y) > 0$, then we have that $\mu(z|y) > \alpha$. We then obtain the following results.

Proposition 1 *There exists a unique Nash equilibrium* $\Lambda^* = (\mu_c^*, \Lambda_p^*)$.

6 Structural Properties of Core-Periphery Communities

In this section, we derive the structural properties of a core-periphery community at the Nash equilibrium $\Lambda^* = (\mu_c^*, \Lambda_p^*)$.

6.1 Condition for Core-Periphery Communities to Emerge

We first characterize how the rate budget M_c of the core agent, and the rate budgets M_p of the periphery agents, impact the structural properties of the Nash equilibrium. We have the following result.

Proposition 2 *There exists constants* m_c *and* m_p *such that if for the rate budget* M_c *of the core agent and the rate budget* M_p *of the periphery agents we have that*

$$M_c > m_c \quad \textit{and} \quad M_p > m_p,$$

then the following is true for the resulting Nash equilibrium $\Lambda^* = (\mu_c^*, \Lambda_p^*)$. *For all periphery agents* $y \in C_p$ *we have that*

$$\mu_c^*(y) > 0 \quad \textit{and} \quad \mu^*(y_c|y) > 0.$$

Proposition 2 states that if the rate budgets M_c and M_p are high enough then all periphery agents follow the core agent, and the core agent will follow all periphery agents.

Proposition 2 provides conditions for a core-periphery community to emerge. In a core-periphery community, the core agent collects content from (almost) all periphery agents and makes it available to the periphery agents. In addition, in a core-periphery community (almost) all periphery agents follow the core agent in order to obtain content from the community. Proposition 2 states that in order for this structure to emerge, the agents have to be sufficiently interested in getting content and allocated a sufficient amount of time (a sufficiently large rate budget) to sharing online content.

6.2 Connectivity Between Periphery Agents

We next study the structural properties of how periphery agents follow each other in a core-periphery community. We have the following result.

Proposition 3 *For a Nash equilibrium $\Lambda^* = (\mu_c^*, \Lambda_p^*)$ as given in Proposition 2 the following is true. For each periphery agent $y \in C_p$ there exists a threshold $t(y) > 0$ such that $I(\mu^*(z|y)) = 1$, if, and only if, $p(z|y) > t(y)$.*

Note that the value of $p(z|y)$ is higher for agents z that are close to agent y. As a result, Proposition 3 states that each periphery agent y follows other periphery agents z that are not too far away from y. Combining this result with Propositions 2 and 3 states that core-periphery communities have the structural property that periphery agents follow the core agent, as well as other periphery agents that produce content close to the agents' main interest. This result provides insight into how content is propagated within a core-periphery community. In particular, the result implies that content propagates in the following two manners: it spreads (globally) through the core agent within the community, as well as locally through the connection between periphery agents that have similar interests.

6.3 Following Rates

Next, we characterize the following rates between the core agent and periphery agents. We obtain the following result.

Proposition 4 *For a Nash equilibrium $\Lambda^* = (\mu_c^*, \Lambda_p^*)$ as given in Proposition 2 the following is true. If for two periphery agents $y, y' \in C_p$ we have*

$$||y - I_0|| < ||y' - I_0||,$$

then we have that

$$\mu^*(y_c|y) > \mu^*(y_c|y') \quad \textit{and} \quad \mu_c^*(y) > \mu_c^*(y').$$

Proposition 4 states that periphery agents that are close to the center I_0 of the community have higher interaction rates compared with a periphery agents further away from I_0. More precisely, both the rate $\mu^*(y_c|y)$ with which periphery agent y follows the core agent y_c, and the rate $\mu_c^*(y)$ with which the core agent follows periphery agent y, is higher for an agent y closer to the center of the community I_0.

Proposition 4 provides a "ranking" or "ordering" of periphery agents $y \in C_p$ based on how close they are to the center I_0 of the community. While it is impossible to directly measure how close a periphery agent is with respect to the center of the community, it is possible to measure/estimate the interaction rates of the agent with the core agent. These measurements/estimates can be used in return to infer how close an agent is to the center of the community.

7 Conclusions

We characterized the structural properties of core-periphery communities using a game-theoretic framework. Assuming that agents allocate a sufficient rate (as given by Proposition 2), we obtain the following results:

(a) **Connectivity of Core Agents**: Core agents follow all periphery agents (Proposition 2). This confirms the results obtained from experimental studies that core agents serve as a "hub" for the community by collecting (aggregating) content and making it available to the other agents in the community [12, 13].
(b) **Connectivity of Periphery Agents**: Periphery agents have two types of connections. First, they all follow the core agents (Proposition 2). Second, they also follow other periphery agents whose main interest closely matches their interest (Proposition 3). This result implies that the structure of a core-periphery is not given by a star structure, but has a more complex structure with connections between periphery agents. In addition, this result provides insight into how content propagates within a community (see discussion after Proposition 3).
(c) **Interaction Rates**: Periphery agents whose main interest is closer to the center of the community I_0 have higher interaction rates with the core agent compared with agents further away from I_0 (Proposition 4). One possible application of this result is to rank periphery agents with respect to how close their main interest is to the community center I_0 (see discussion after Proposition 4).

The obtained results provide a mathematical characterization of the structure of core-periphery communities, that can be used to design algorithms. We are currently using these structural properties to derive community detection algorithms that require only local information, and community-based content recommendation algorithms. The obtained allow us to derive these algorithms in a formal manner, and provide formal performance guarantees.

References

1. Bala, V., Goyal, S.: A noncooperative model of network formation. Econometrica **68**(5), 1181–1229 (2000)
2. Belleflamme, P., Bloch, F.: Market sharing agreements and stable collusive network (2002)
3. Dellarocas, C., Katona, Z., Rand, W.: Media, aggregators and the link economy: strategic hyperlink formation in content networks. School Manage. Res. Paper **2010–30**, 06–131 (2010)
4. Dellarocas, C., Katona, Z., Rand, W.: Media, aggregators, and the link economy: strategic hyperlink formation in content networks. Manage. Sci. **59**(10), 2360–2379 (2013)
5. Furusawa, T., Konishi, H.: Free trade networks. J. Int. Econ. **72**(2), 310–335 (2007)
6. Hegde, N., Massoulié, L., Viennot, L.: Self-organizing flows in social networks. In: International Colloquium on Structural Information and Communication Complexity, pp. 116–128. Springer (2013)
7. Jackson, M.O.: A survey of network formation models: stability and efficiency. Group Form. Econ. Netw. Clubs Coalitions **664**, 11–49 (2005)
8. Jackson, M.O., Wolinsky, A.: A strategic model of social and economic networks. J. Econ. Theor. **71**(1), 44–74 (1996)
9. Jordan, P.R., Nadav, U., Punera, K., Skrzypacz, A., Varghese, G.: Lattice games and the economics of aggregators. In: Proceedings of the 21st international conference on World Wide Web, pp. 549–558. ACM (2012)
10. Starr, R.M., Stinchcombe, M.: Efficient Transportation Routing and Natural Monopoly in the Airline Industry: An Economic Analysis of Hub-Spoke and Related Systems. University of California, Department of Economics (1992)
11. Su, J., Marbach, P.: Structure of core-periphery communities (2022). https://arxiv.org/abs/2207.06964
12. Warmbrodt, J., Sheng, H., Hall, R.: Social network analysis of video bloggers' community. In: Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008), p. 291. IEEE (2008)
13. Yang, J., Zhang, M., Shen, K.N., Ju, X., Guo, X.: Structural correlation between communities and core-periphery structures in social networks: evidence from twitter data. Expert Syst. Appl. **111**, 91–99 (2018)

Outliers in the ABCD Random Graph Model with Community Structure (ABCD+o)

Bogumił Kamiński, Paweł Prałat, and François Théberge

Abstract The **A**rtificial **B**enchmark for **C**ommunity **D**etection graph (**ABCD**) is a random graph model with community structure and power-law distribution for both degrees and community sizes. The model generates graphs with similar properties as the well-known **LFR** one, and its main parameter ξ can be tuned to mimic its counterpart in the **LFR** model, the mixing parameter μ. In this paper, we extend the **ABCD** model to include potential outliers. We perform some exploratory experiments on both the new **ABCD+o** model as well as a real-world network to show that outliers posses some desired, distinguishable properties.

Keywords ABCD model · Outliers · Community detection

1 Introduction

One of the most important features of real-world networks is their community structure, as it reveals the internal organization of nodes [7]. In social networks communities may represent groups by interest, in citation networks they correspond to related papers, in the Web communities are formed by pages on related topics, etc. Being able to identify communities in a network could help us to exploit this network more effectively. Grouping like-minded users or similar-looking items together is important for a wide range of applications including recommendation systems, anomaly or outlier

B. Kamiński
SGH Warsaw School of Economics, Warsaw, Poland
e-mail: bkamins@sgh.waw.pl

P. Prałat (✉)
Toronto Metropolitan University, Toronto, ON, Canada
e-mail: pralat@ryerson.ca

F. Théberge
The Tutte Institute for Mathematics and Computing, Ottawa, ON, Canada
e-mail: theberge@ieee.org

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_13

detection, fraud detection, rumour or fake news detection, etc. [10]. For more discussion around various aspects of mining complex networks see, for example, [14, 19].

It was identified as one of the major current challenges in detecting communities that most of the existing algorithms treat all nodes the same way, that is, they try to assign them to precisely one community. On the other hand, many complex networks (regardless whether their nodes correspond to, say, users of some social media or movies on Netflix) consist of nodes that are more active participants of their own communities while others are not [17]. As a result, there is a need to detect outlier nodes that are not part of any of the communities. Moreover, some communities might be overlapping which is reflected by some of the nodes belonging to a few communities via fuzzy membership. Some recent algorithms (see, for example [2, 8] or **NI-Louvain** [22]) try to incorporate these notions but more research is expected to be pursued in the near future. For more on anomalies and outliers in graphs see, for example, the survey [1].

Another well-known challenge recognized by many researchers is that there are very few datasets with ground-truth identified and labelled. As a result, there is need for synthetic random graph models with community structure that resemble real-world networks in order to benchmark and tune clustering algorithms that are unsupervised by nature. The **LFR** (Lancichinetti, Fortunato, Radicchi) model [15, 16] generates networks with communities and at the same time it allows for the heterogeneity in the distributions of both node degrees and of community sizes. It became a standard and extensively used method for generating artificial networks with (non-overlapping) community structure.

Unfortunately, the situation is much more challenging if one needs a synthetic model with outliers. There seems to be no standard model that one may use. For example, in [8] the authors adjust the classical Stochastic Block Model to simultaneously take into account the community structure and outliers by introducing different probabilities of connection between inliers and pairs involving outliers. To validate algorithms tested in [2], the authors start with a synthetic **LFR** network or a real-world one and then randomly perturb edges around some randomly selected nodes in order to create artificial outliers. **LFR** itself [15] has some basic functionality to create overlapping clusters but not outliers.

In this paper, we revisit the Artificial Benchmark for Community Detection (**ABCD** graph) [13] that was recently introduced and implemented,[1] including a fast implementation that uses multiple threads (**ABCDe**) [11].[2] Undirected variant of **LFR** and **ABCD** produce graphs with comparable properties but **ABCD**/**ABCDe** is faster than **LFR** and can be easily tuned to allow the user to make a smooth transition between the two extremes: pure (disjoint) communities and random graph with no community structure. Moreover, it is easier to analyze theoretically. For example, various theoretical asymptotic properties of the **ABCD** model are analyzed in [12], including the modularity function that is, arguably, the most important graph property of networks in the context of community detection.

[1] https://github.com/bkamins/ABCDGraphGenerator.jl/.

[2] https://github.com/tolcz/ABCDeGraphGenerator.jl/.

We extend the original **ABCD** model to include potential outliers (see Sect. 2). We examine one of the few real-world networks with identified outliers, the College Football Graph (see Sect. 3.1), and identify a few distinctive properties of outliers that are present in this network. We then perform a few simulations with our new **ABCD+o** model to show that its outliers posses similar properties (see Sects. 3.2 and 3.3). Future directions are briefly mentioned in Sect. 4.

2 Adjusting the ABCD Model to Include Outliers

We start this section with a brief description of the **ABCD** model taken from [11]; details can be found in [13] or in [12]. We then carefully explain the adjustments needed to incorporate the existence of outliers.

2.1 The Original Model

As in **LFR** model [15, 16], for a given number of nodes n, we start by generating a power law distribution both for the degrees and community sizes. Those are governed by the power law exponent parameters (γ, β). We also provide additional information to the model, again as it is done in **LFR**, namely, the average and the maximum degree, and the range for the community sizes. The user may alternatively provide a specific degree distribution and/or community sizes.

For each community, we generate a random *community* subgraph on the nodes from a given community using either the **configuration model** [4] (see [3, 23, 24] for related models and results) which preserves the exact degree distribution, or the **Chung-Lu model** [5] which preserves the expected degree distribution. On top of it, we independently generate a *background* random graph on all the nodes. Everything is tuned properly so that the degree distribution of the union of all graphs follows the desired degree distribution (only in expectation in the case of the Chung-Lu variant). The mixing parameter ξ guides the proportion of edges which are generated via the background graph. In particular, in the two extreme cases, when $\xi = 1$ the graph has no community structure while if $\xi = 0$, then we get disjoint communities. In order to generate simple graphs, we may have to do some re-sampling or edge re-wiring, which are described in [13].

During this process, larger communities will additionally get some more internal edges due to the background graph. As argued in [13], this "global" variant of the model is more natural and so we recommend it. However, in order to provide a variant where the expected proportion of internal edges is exactly the same for every community (as it is done in **LFR**), we also provide a "local" variant of **ABCD** in which the mixing parameter ξ is automatically adjusted for every community.

Two examples of **ABCD** graphs on $n = 100$ nodes are presented in Fig. 1. Degree distribution was generated with power law exponent $\gamma = 2.5$ with minimum and

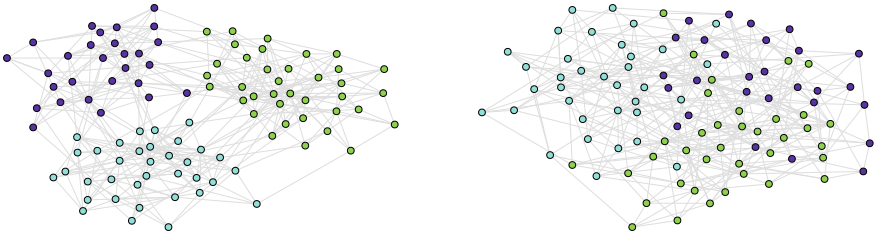


Fig. 1 Two examples of **ABCD** graphs with low level of noise ($\xi = 0.2$, left) and high level of noise ($\xi = 0.4$, right)

maximum values 5 and 15, respectively. Community sizes were generated with power law exponent $\beta = 1.5$ with minimum and maximum values 20 and 40, respectively; communities are shown in different colours. The global variant and the configuration model was used to generate the graphs. The left plot has the mixing parameter set $\xi = 0.2$ while the "noisier" graph on the right plot has the parameter fixed to $\xi = 0.4$.

2.2 Adjusting the Model to Include Outliers

The adjusted model, **ABCD+o** (**ABCD** with **outliers**), will have additional parameter s_0 which is equal to the number of outliers. Because of a well structured and flexible design of the original model, adjusting it to include outliers is simple. One trivial adjustment needed is in the way the distribution of community sizes is generated. Slightly more delicate modification is needed in the process of assigning nodes to communities. However, before that the algorithm needs to select suitable nodes for outliers. Below, we independently discuss these issues and explain how they are generalized.

The **ABCD+o** extension is defined only for the default settings of the original **ABCD** algorithm, namely, for the global version of the algorithm, configuration model used to generate community and background graphs, and accepts only parameter ξ as the level of noise.

Distribution of Community Sizes

As in the original **ABCD** model, the degree distribution is generated randomly following the (truncated) *power-law distribution* $\mathcal{P}(\gamma, \delta, \Delta)$ with exponent $\gamma \in \mathbf{R}_+$, minimum value δ, and maximum value $\Delta \geq \delta$. No adjustment is needed.

Let $\beta \in \mathbf{R}_+$, $s, S \in \mathbf{N}$ such that $\delta < s \leq S$. Community sizes in the original **ABCD** model are generated randomly following the (truncated) *power-law distribution* $\mathcal{P}(\beta, s, S)$ with exponent β, minimum value s, and maximum value S. It is recommended to use $\beta \in (1, 2)$, some relatively small value of s such as 100 or 500, and S larger than Δ. The condition for S is needed to make sure large degree nodes have large enough communities to be assigned to. Similarly, the assumption that $s \geq \delta + 1$ is required to guarantee that small communities are not too small and

so that they can accommodate small degree nodes. These conditions are needed to make sure that generating a simple graph with the desired properties is feasible.

Communities in the original model are generated with this distribution as long as the sum of their sizes is less than n, the desired number of nodes. After drawing a predetermined number of samples from this distribution, the algorithm is selecting one sequence with the sum as close to n as possible and carefully adjusts it, if needed.

Since there are s_0 outliers in the new model, the community sizes $(s_i, i \in [\ell] := \{1, \ldots, \ell\})$ are generated as in the original model but this time with the condition that the sum of their sizes is equal to $n - s_0$ (instead of n).

Assigning Nodes to Outliers

Parameter $\xi \in (0, 1)$ reflects the amount of noise in the network. It controls the fraction of edges that are between communities. Indeed, in the original **ABCD** model, asymptotically (but not exactly) $1 - \xi$ fraction of edges end up within one of the communities. Each node in the original model has its degree w_i split into two parts: *community degree* y_i and *background degree* z_i ($w_i = y_i + z_i$). The goal is to get $y_i \approx (1 - \xi)w_i$ and $z_i \approx \xi w_i$. However, both y_i and z_i have to be non-negative integers and for each community $C \subseteq V$, $\sum_{i \in C} y_i$ has to be even. Fortunately, this can be easily achieved by an appropriate random rounding of $(1 - \xi)w_i$ to the nearest integers.

In the generalized **ABCD+o** model, each non-outlier has its degree w_i split into y_i and z_i, as in the original model. These nodes will be assigned into one community. On the other hand, outliers will not get assigned to any community and all of their neighbours will be in the background graph and so they will be "sprinkled" across the whole graph. As a result, their degrees will satisfy $w_i = z_i$. Note that the only potential problem with outliers that might occur is when ξ is close to zero. At the extreme case when $\xi = 0$, only outliers have non-zero degree in the background graph. In order to make sure that there exists a simple graph that satisfies the required degree distribution, in such extreme situations all outliers must have degrees smaller than s_0. The model needs to be prepared for such potential problems but in practice (when the number of nodes n is large, the number of outliers s_0 is relatively small, and the level of noise ξ is not zero) there are plenty of nodes with non-zero degree in the background graph and so there is no restriction for outliers.

To prepare for a potential problem we do the following. Once the degree of each node w_i is split into y_i and z_i, we get a lower bound for the number of nodes that will have non-zero degree in the background graph, namely, $L := |\{v \in V : z_i \geq 1\}|$. Note that $\bar{L} = \mathbf{E}[L] = \sum_{i \in V} \min(1, \xi w_i)$ since each node with $\xi w_i \geq 1$ satisfies $z_i \geq 1$ and each node with $\xi w_i < 1$ has $z_i = 1$ with probability ξw_i and $z_i = 0$ otherwise. Moreover, since by default outliers have $z_i = w_i \geq 1$, there will be at least s_0 vertices of positive degree in the background graph. Assuming that outliers are selected uniformly at random, we expect $L + (n - L)(s_0/n)$ nodes of positive degree in the background graph. (In fact, since there is a slight bias toward selecting small degree nodes for outliers and L has a bias toward large degree nodes, we expect slightly more nodes of positive degree in the background graph, which is good.) We introduce the following constraint: a node of degree w_i can become an outlier if

$$w_i \leq \bar{L} + s_0 - \bar{L}s_0/n - 1. \tag{1}$$

Finally, s_0 nodes satisfying (1) are selected uniformly at random to become outliers. (In the implementation, these nodes simply form an independent "community" with $y_i = 0$ and $z_i = w_i$.)

Assigning Nodes to Communities

Similarly to the potential problem with outliers, we need to make sure that non-outliers of large degree are not assigned to small communities. Based on the parameter ξ we know that roughly $(1-\xi)w_i$ neighbours of a node of degree w_i will be present in its own community. However, this is only the lower bound as some neighbours in the background graph might end up there by chance. Hence, in order to make enough room in the community graph for all neighbours of a given node, the original **ABCD** algorithm needs to compute x_i, the expected number of neighbours of a node of degree w_i that end up in its own community. We need to recompute x_i to incorporate the existence of outliers.

Assuming that nodes are assigned randomly with a distribution close to the uniform distribution, we expect Ws_0/n points (in the corresponding configuration model) in the background graph to be associated with outliers, where $W := \sum_{i\in[n]} w_i$ is the volume of the graph (equivalently, the total number of points in the corresponding configuration model). Similarly, we expect ξ fraction of the points associated with non-outliers to end up in the background graph, that is, $W(1-s_0/n)\xi$ points. In order to estimate what fraction of neighbours of a given non-outlier node is expected to be within the same community, we need to answer the following question: what is the probability that a random point in the background graph associated with a non-outlier is matched with a point within the same community? It is equal to

$$\sum_{j\in[\ell]} \frac{s_j}{n-s_0} \cdot \frac{\frac{s_j}{n-s_0} W(1-s_0/n)\xi}{W(1-s_0/n)\xi + Ws_0/n} = \sum_{j\in[\ell]} \left(\frac{s_j}{n-s_0}\right)^2 \frac{(n-s_0)\xi}{(n-s_0)\xi + s_0}.$$

Indeed, with probability $\frac{s_j}{n-s_0}$ a random point belongs to community j. There are $\frac{s_j}{n-s_0} W(1-s_0/n)\xi$ points associated with community j and the total number of points in the background graph is $W(1-s_0/n)\xi + Ws_0/n$. Hence, one can easily estimate the probability that the point from community j is matched with another point from the same community. The expected number of neighbours of a node of degree w_i that stay within the same community is then

$$x_i := \left(1 - \xi + \xi \sum_{j\in[\ell]} \left(\frac{s_j}{n-s_0}\right)^2 \frac{(n-s_0)\xi}{(n-s_0)\xi + s_0}\right) w_i = (1-\xi\phi)w_i,$$

where

$$\phi := 1 - \sum_{j\in[\ell]} \left(\frac{s_j}{n-s_0}\right)^2 \frac{(n-s_0)\xi}{(n-s_0)\xi + s_0}.$$

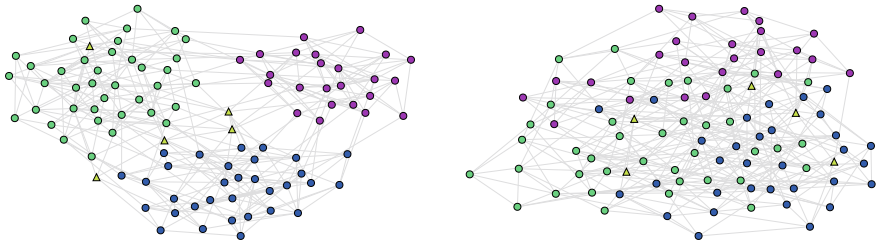


Fig. 2 Two examples of **ABCD+o** graphs with low level of noise ($\xi = 0.2$, left) and high level of noise ($\xi = 0.4$, right). The number of outliers is $s_0 = 5$

In particular, we expect $(1 - \xi\phi)(1 - s_0/n)$ fraction of edges to stay within one of the communities. Moreover, as expected, if $s_0 = 0$, then we recover the value of ϕ used in the original **ABCD** model, namely,

$$\phi = 1 - \sum_{j \in [\ell]} \left(\frac{s_j}{n}\right)^2 .$$

As in the original **ABCD** model, a node of degree w_i can be assigned to community of size s_j if $x_i \leq s_j - 1$. We select one admissible assignment of non-outliers to communities uniformly at random which turns out to be relatively easy from both theoretical and practical points of view.

Two examples of **ABCD+o** graphs on $n = 100$ nodes are presented in Fig. 2. The number of outliers is $s_0 = 5$ and the remaining parameters are exactly the same as the ones to produce Fig. 1. Communities are shown in different colours and outliers are displayed with triangular shape. The left plot has the mixing parameter set $\xi = 0.2$ while the "noisier" graph on the right plot has the parameter fixed to $\xi = 0.4$. In the left plot it is visible that 4 out of 5 outliers are clearly located *between* the communities (one of them is within a community as outlier can, by pure chance, get many edges within one community). In the right plot, which is more noisy, we still see that outliers are surrounded by nodes belonging to different communities.

3 Experiments—Distinguishing Properties of Outliers

In order to better understand properties of outliers, we perform a few simple and exploratory experiments on the well-known College Football real-world network with known community structure and the presence of outliers. We identified three natural properties that distinguish outliers from non-outliers.

In order to show that our new **ABCD+o** model exhibits similar desired properties, we generated graphs on $n = 10{,}000$ nodes and $s_0 = 500$ outliers (5%). Degree distribution was generated with power law exponent $\gamma = 2.5$ with minimum and max-

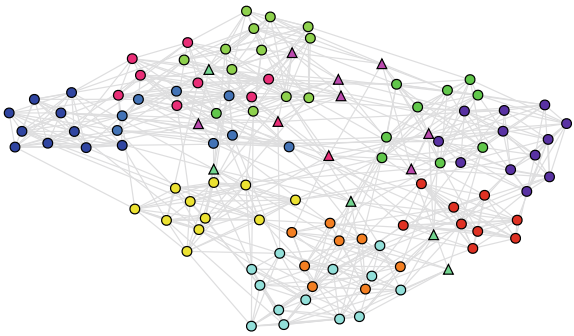


Fig. 3 The college football graph; outliers are displayed with triangular shape

imum values 5 and 500, respectively. Community sizes were generated with power law exponent $\beta = 1.5$ with minimum and maximum values 100 and 1,000, respectively. We independently generated graphs for all values of $\xi \in \{0.0, 0.1, \ldots, 1.0\}$ but the degree distribution and the distribution of community sizes were coupled (it is easy to do in our implementation) so that all 11 graphs use the same distributions.

3.1 The College Football Graph

The College Football real-world network represents the schedule of United States football games between Division IA colleges during the regular season in Fall 2000 [9]. The data consists of 115 teams (nodes) and 613 games (edges). The teams are divided into conferences containing 8–12 teams each. In general, games are more frequent between members of the same conference than between members of different conferences, with teams playing an average of about seven intra-conference games and four inter-conference games in the 2000 season. There are a few exceptions to this rule, as detailed in [18]: one of the conferences is really a group of independent teams, one conference is really broken into two groups, and 3 other teams play mainly against teams from other conferences. We refer to those 14 teams as outlying nodes, which we represent with a distinctive triangular shape in Fig. 3.

3.2 Participation Coefficient

The following definitions are commonly used in the literature [6, 21] (see also [14]). We say that a set of nodes $C \subseteq V$ forms a *strong community* if each node in C has more neighbours in C than outside of C. One may relax this strong notion and say that C forms a *weak community* if the average degree inside the community C (over all nodes in C) is larger than the corresponding average number of neighbours outside of C. In this context, an *outlier* could be formally defined as a node that

does not have majority of its neighbours in any of the communities. In the **ABCD+o** model, non-outliers are expected to have more than half of their neighbours in its own community, provided that $\xi < 0.5$. On the other hand, outliers are expected to satisfy the desired property, unless there is an enormous community spanning more than 50% of nodes.

A more refined picture is provided by the next coefficient that is a natural measure of concentration. For any partition $\mathbf{A} = \{A_1, \ldots, A_\ell\}$ of the set of nodes, the *participation coefficient* of a node v (with respect to $\mathbf{A}$) is defined as follows:

$$p(v) = 1 - \sum_{i=1}^{\ell} \left(\frac{\deg_{A_i}(v)}{\deg(v)} \right)^2,$$

where $\deg_{A_i}(v)$ is the number of neighbours of v in A_i. The participation coefficient $p(v)$ is equal to zero if v has neighbours exclusively in one part. Members of strong communities satisfy, by definition, $p(v) < 3/4$. In the other extreme case, the neighbours of v are homogeneously distributed among all parts and so $p(v)$ is close to the trivial upper bound of

$$1 - \sum_{i=1}^{\ell} \left(\frac{\deg(v)/\ell}{\deg(v)} \right)^2 = 1 - \frac{1}{\ell} \approx 1.$$

For the experiments shown below, even though we have the ground truth communities available to use, we computed the participation coefficients using communities (partition $\mathbf{A}$) we obtained with the **ECG** clustering algorithm which we describe in the following subsection. The distribution of the participation coefficient among outliers and non-outliers for the College Football Graph is presented on box plot in Fig. 4 (left). We see that outliers have significantly larger average value of $p(v)$ than the corresponding value for non-outliers: 0.709 vs. 0.439. The corresponding averages (together with associated standard deviations) for the **ABCD+o** model with different level of noise are presented in Fig. 4 (right). For low level of noise (small values of ξ) there is a clear difference between outliers and non-outliers but the discrepancy diminishes for noisy graphs (large values of ξ). In the extreme case when $\xi = 1$ there is no difference between the two classes and so the averages are close to each other as they should.

3.3 ECG Votes

Ensemble Clustering algorithm for Graphs (**ECG**) [20][3] is a consensus clustering method based on the classical **Louvain** algorithm. In its first phase, several low-level partitions are computed with different randomization, and for each edge the

[3] https://github.com/ftheberge/graph-partition-and-measures.

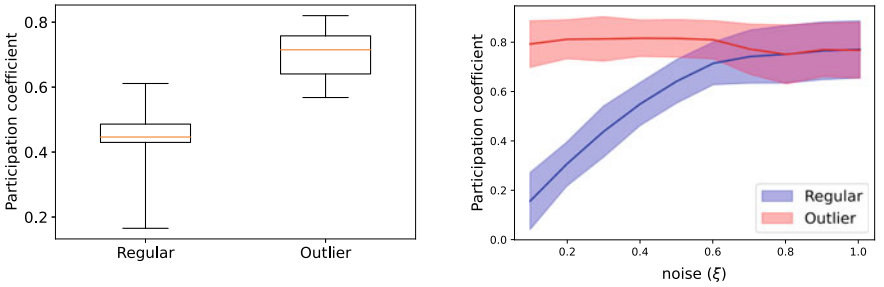


Fig. 4 Distribution of the participation coefficient for regular and outlier nodes: college football graph (left) and **ABCD+o** model (right)

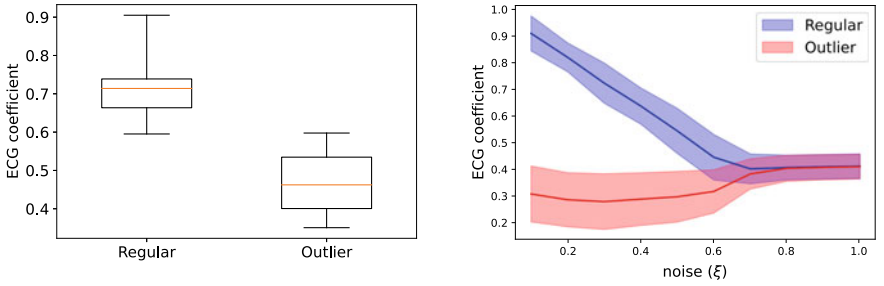


Fig. 5 Distribution of the ECG coefficient for regular and outlier nodes: college football graph (left) and **ABCD+o** model (right)

proportion of times both nodes ended up in the same part is computed. Those are the *ECG edge scores*. High scores are indicative of stable pairs that often appear in the same part. For a given node v, we define $E(v)$ to be the average ECG score over all edges incident to v, and we call it the *ECG coefficient* of a node v. It is expected that outliers are more challenging to cluster which should be manifested by relatively small ECG coefficients $E(v)$ associated with these nodes.

As it was done for the participation coefficient, we investigate the distribution of the ECG coefficient among outliers and non-outliers for the College Football Graph—see Fig. 5 (left). We see that it is another distinguishing coefficient—outliers have significantly smaller average value of $E(v)$ than the corresponding value for non-outliers: 0.465 vs. 0.701. Similar conclusions can be derived from the corresponding averages for the **ABCD+o** model—see Fig. 5 (right). As before, the difference becomes less visible as more noise is introduced.

4 Future Directions

In this paper, we extended the **ABCD** model to **ABCD+o** which incorporates the presence of outliers. We investigated a few properties that are able to distinguish outliers from regular nodes. One may try to extend these ideas further and build an outlier detection algorithm. Another important extension of the original **ABCD** that we leave for the future is to design a variant of the model to include overlapping clusters. An orthogonal future direction that we (and industry partners that we collaborate with) are interested in is to design a hypergraph model with known community structure.

References

1. Akoglu, L., Tong, H., Koutra, D.: Graph based anomaly detection and description: a survey. Data Min. Knowl. Disc. **29**(3), 626–688 (2015)
2. Bandyopadhyay, S., Vivek, S.V., Murty, M.N.: Integrating network embedding and community outlier detection via multiclass graph description. arXiv:2007.10231 (2020)
3. Bender, E.A., Canfield, E.R.: The asymptotic number of labeled graphs with given degree sequences. J. Combin. Theor. Ser. A **24**(3), 296–307 (1978)
4. Bollobás, B.: A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. Euro. J. Combin. **1**(4), 311–316 (1980)
5. Chung Graham, F., Lu, L.: Complex Graphs and Networks. no. 107. American Mathematical Soc. (2006)
6. Flake, G.W., Lawrence, S., Giles, C.L.: Efficient identification of web communities. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 150–160 (2000)
7. Fortunato, S.: Community detection in graphs. Phys. Rep. **486**(3–5), 75–174 (2010)
8. Gaucher, S., Klopp, O., Robin, G.: Outlier detection in networks with missing links. Comput. Stat. Data Anal. **164**, 107308 (2021)
9. Girvan, M., Newman, M.E.: Community structure in social and biological networks. Proc. Nat. Acad. Sci. **99**(12), 7821–7826 (2002)
10. Javed, M.A., Younis, M.S., Latif, S., Qadir, J., Baig, A.: Community detection in networks: a multidisciplinary review. J. Netw. Comput. Appl. **108**, 87–111 (2018)
11. Kamiński, B., Olczak, T., Pankratz, B., Prałat, P., Théberge, F.: Properties and performance of the ABCDE random graph model with community structure. arXiv:2203.14899 (2022)
12. Kamiński, B., Pankratz, B., Prałat, P., Théberge, F.: Modularity of the abcd random graph model with community structure. arXiv:2203.01480 (2022)
13. Kamiński, B., Prałat, P., Théberge, F.: Artificial benchmark for community detection (ABCD)-fast random graph model with community structure. Netw. Sci. 1–26 (2021)
14. Kamiński, B., Prałat, P., Théberge, F.: Mining complex networks (2021)
15. Lancichinetti, A., Fortunato, S.: Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Phys. Rev. E **80**(1), 016118 (2009)
16. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Phys. Rev. E **78**(4), 046110 (2008)
17. Liu, F., Wang, Z., Deng, Y.: GMM: a generalized mechanics model for identifying the importance of nodes in complex networks. Knowl.-Based Syst. **193**, 105464 (2020)
18. Lu, Z., Wahlström, J., Nehorai, A.: Community detection in complex networks via clique conductance. Sci. Rep. **8**(1), 1–16 (2018)

19. Newman, M.E.J.: Networks, 2nd ed. Oxford University Press, Oxford, New York (2018)
20. Poulin, V., Théberge, F.: Ensemble clustering for graphs. In: International Conference on Complex Networks and their Applications, pp. 231–243. Springer (2018)
21. Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., Parisi, D.: Defining and identifying communities in networks. Proc. Nat. Acad. Sci. **101**(9), 2658–2663 (2004)
22. Singh, D., Garg, R.: Ni-louvain: A novel algorithm to detect overlapping communities with influence analysis. J. King Saud Univ. Comput. Inf. Sci. (2021)
23. Wormald, N.C.: Generating random regular graphs. J. Algorithms **5**(2), 247–280 (1984)
24. Wormald, N.C., et al.: Models of random regular graphs. In: London Mathematical Society Lecture Note Series, pp. 239–298 (1999)

Influence-Based Community Deception

Saif Aldeen Madi and Giuseppe Pirrò

Abstract This paper studies the novel problem of influence-based community deception. Tackling this problem amounts to devising tools to protect the users of a community from being discovered by community detection algorithms. The novel setting considers networks that have both edge directions and models the influence of nodes as edge weights. We present a deception strategy based on modularity. We conducted an experimental evaluation that shows the feasibility of our proposal.

Keywords Community deception · Community hiding

1 Introduction

Social network analysis has been an active area of research thanks to the accelerating growth of social media platforms with billions of users worldwide. A particular example is *community detection* which has become a relatively well-established research problem, owing to its wide range of applications, including recommendation systems [17], fraud detection [19], and citation networks [1].

Naturally, social networking platforms, such as Facebook or Twitter, constitute an important application area for community detection. Therefore, it is expected that such algorithms will play an increasingly influential role in the lives of millions of users. This raises subtle ethical dilemmas, particularly regarding user privacy [23], freedom of speech, and security. Such concerns ignited serious efforts toward designing algorithms that help communities of users protect their privacy by evading community detection algorithms. This new research field has been variously referred to as community deception [6] or community hiding [23], where the goal is to hide a target community from detection by rewiring connections incidents to its nodes.

S. A. Madi · G. Pirrò (✉)
Department of Computer Science, Sapienza University of Rome, Rome, Italy
e-mail: pirro@di.uniroma1.it

S. A. Madi
e-mail: madi@di.uniroma1.it

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_14

Table 1 State-of-the-art deception algorithms

Quality function	Mathematical formulation	Deception algorithms
Modularity [15, 16]	$\frac{1}{m}\sum_{C\in\bar{C}}\sum_{ij\in C}\mathcal{I}_{ij} - \frac{d_i^{out}d_j^{in}}{m}$	Nagraja [14], DICE [23]
Safeness [6]	$\tau\frac{\lvert V^u(C)\rvert-\lvert E(u,C)\rvert}{\lvert C\rvert-1} + \chi\frac{\lvert\bar{E}(u,V)\rvert}{\delta(u)}$	SADDEN [6]
Permanence [2, 3]	$Perm(u,G) = \frac{\lvert E(u,V)\rvert}{E^{max}(u)} \times \frac{1}{\delta(u)} - (1 - C_{in}(u))$	NEURAL [13]
Normalized mutual information [5]	$I(X,Y) = \sum_{x,y} p(x,y)log\frac{p(x,y)}{p(x)p(y)}$	Q-Attack [4]

Related Work Community deception [6] or community hiding [13, 23] is a relatively new research area, which came into existence as countermeasure against some issues posed by detection algorithms, particularly privacy-related concerns [6, 23]. Its noteworthy that some authors refer to deception as an *attack* [4], which reflects the perspective of the detector. Table 1 gives an overview of the state-of-the-art deception algorithms.

However, previous work on deception algorithms overlooked an essential component of social relationships, namely, *influence* [8, 11]. Indeed, to the best of our knowledge, all previous deception algorithms have been devised using influence-less 0–1 edge. Such a representation assumes that a pair of nodes is either connected or not, ignoring the strength of influence a node exerts on its neighbors. We consider this a serious drawback of state-of-the-art community deception for several reasons. First, real-world social relationships vary in their influence. For example, a person, say, Bob, probably exerts a greater influence on his son than on his neighbor. Indeed, while it might be reasonable to assume that some social connections share relatively similar influences, it is certainly not a universal truth. Such variation in influence has not, so far, been covered by the deception literature. Secondly, the notion of influence has already been incorporated as an important component in several detection algorithms [8, 11, 12, 22]. This necessarily leaves state-of-the-art deception algorithms lagging, as they fail to account for such influence-aware detection methods. Moreover, deception algorithms have been applied to undirected networks only. This was another critical drawback in previous approaches. Suppose that Bob follows a celebrity, say, Alice, on Twitter. This does not necessarily imply that Alice follows Bob as well. We can argue that the followed—Alice, influences the follower—Bob, but not vice-versa. Indeed, she probably does not even know who Bob is. Finally, and maybe more importantly, deception algorithms can make much more intelligent decisions when considering influence. Specifically, several previous deception algorithms have utilized an edge modification budget, implying the desire to perform deception with the least number of edge updates. By considering influence and direction, we can distinguish between edges' importance and carefully choose those which make the most deceptive effect to be modified.

Contributions and Outline. The main aim of this work is to address all previous issues by introducing *directed influence* as an essential component of the deception process.

We comprehensively study community deception in Directed Influence Networks (DIN), using modularity as the quality function of choice. We focus in this study on modularity because it is a popular quality function [7], subsequently making our deception algorithm intuitive. Specifically, we present the following contributions: (i) We formally present community deception in the context of DIN; (ii) We introduce an upgraded version of modularity that accommodates the concept of influence; (iii) We develop an influence-aware deception algorithm called InfluDec and compare it with the state-of-the-art.

The remainder of the paper is organized as follows. Section 2 introduces the community deception problem. Section 3 introduces the influence-based community deception problem in directed networks and a greedy algorithm called InfluDec. Section 4 reports on an experimental evaluation. We conclude in Sect. 5.

2 Background

The objective of community deception is to devise an algorithm that can be used by a group of nodes to conceal their relationship from community detection algorithms. Particularly, we assume a directed network $G(V, E)$, where V is the set of vertices and E is the set of edges. This network is represented with an adjacency matrix $\mathbf{A} = [\mathcal{I}_{ij}]$, where $\mathcal{I}_{ij}$ is the influence of node i (the *influencer*) on node j (the *influenced*). We emphasize here that since G is directed, the influence $\mathcal{I}_{ij}$ need not be the same as $\mathcal{I}_{ji}$. A non-overlapping community detection algorithm $\mathcal{A}_{det}$ partitions V into a community structure $\bar{C} = \{C_1, C_2, ..., C_k\}$, where $C_i \subseteq V$, and $C_i \cap C_j = \phi$, for all $i, j \in \{1, ..., k\}$ and $i \neq j$. We also define two types of edges. Considering a community $C \subseteq V$, an edge $(u, v) \in E$ is called an *intra-community* edge, if and only if $u, v \in C$. On the other hand, (u, v) is called an *inter-community* edge, if and only if $u \in C, v \in C'$, and $C' \neq C$. The notation used in this paper is summarized in Table 2.

Problem Statement. In a typical deception scenario, there are two players. The first is the *detector*, who possesses a detection algorithm $\mathcal{A}_{det}$. The second is the *target community* $\mathscr{C}$—a group of nodes that wants to hide itself by utilizing a deception algorithm $\mathcal{A}_{dec}$. By running $\mathcal{A}_{det}$, the detector reveals a set of communities $\bar{C}$, which we call the *revealed* community structure. The nodes of the target community can be either dispersed among different communities in $\bar{C}$, or in a worst-case scenario, totally located in a single community: $\mathscr{C} \subseteq C$ where $C \in \bar{C}$. On the opposite side, the target community's goal is to conceal its identity as a single community from the detector. Specifically, members of $\mathscr{C}$ should be distributed over $\bar{C}$ in a way that prevents their being identified as a single community. To this end, $\mathscr{C}$ uses $\mathcal{A}_{dec}$ in order to maximize a certain *deception score* [6, 23] that measures the concealment level of the target community. Eventually, $\mathcal{A}_{dec}$ produces two sets of edges E^- and E^+, for edges to be deleted and added, respectively. After applying these modifications, the target community $\mathscr{C}$ minimizes the chance of its being detected again either as community or even as a subset of a larger community, the next time $\mathcal{A}_{det}$ is run.

Table 2 Notation table for influence-based modularity

Symbol	Meaning	Formula
$E(C)$	The set of intra-community edges in community C	$\{(u, v)\|u, v \in C\}$
$\tilde{E}(C)$	The set of inter-community edges having one side in community C	$\{(u, v)\|u \in C \vee v \in C\}$
$\mathcal{I}_i^{\uparrow}$	Total influence outgoing from node i	$\sum_j \mathcal{I}_{ij}$
$\mathcal{I}_j^{\downarrow}$	Total influence received by node j	$\sum_i \mathcal{I}_{ij}$
$\mathcal{I}$	Total influence of the entire network	$\sum_{i,j \in V} \mathcal{I}_{ij}$
η	Total intra-community influence of the network	$\sum_{C \in \bar{C}} \sum_{ij \in C} \mathcal{I}_{ij}$
θ	Total inter-community influence in network G	$\mathcal{I} - \eta$
$\bar{C}'$	Community structure excluding community C, where $(u, v) \in C$ is the edge to be modified.	$\bar{C}' = \{C\|C \in \bar{C}$ and $u, v \notin C\}$
C^*	The community for which an edge (u, v) is added/deleted	$C^* = \{C\|C \in \bar{C}$ and $(u, v) \in E(C^*)\}$
$\mathcal{I}_C^{\uparrow}$	Let $C \in \bar{C}$	$\mathcal{I}_C^{\uparrow} C = \sum_{i \in C} \mathcal{I}_i^{\uparrow}$
$\mathcal{I}_C^{\downarrow}$	Let $C \in \bar{C}$	$\mathcal{I}_C^{\downarrow} = \sum_{i \in C} \mathcal{I}_i^{\downarrow}$

Now, let G^* be the whole network after applying the aforementioned edge modifications. If s_1 and s_2 are the deception scores after running $\mathcal{A}_{det}$ on G and G^*, respectively, then C_t's goal would be to maximize $s_2 - s_1$. Typically, $\mathcal{A}_{dec}$ indirectly improves the deception score by choosing edge modifications E^- and E^+ that optimize some deception optimization function. Thus, letting Q be such function, we define the net loss as $\Delta Q = Q_G - Q_{G^*}$, where Q_G and Q_{G^*} are the values before and after applying edge modifications, respectively. Naturally, the deceiver's goal is to introduce edge modifications that maximizes the net loss. Hence, making it no longer feasible for A_D to select C_t as a member of the community structure $\bar{C}$. This can be formulated as an optimization as follows:

$$\underset{G^*}{\text{argmax}} \quad \phi(G, G^*, \mathscr{C}) \tag{1}$$

where $G^* = (V, E')$ and $E' = (E \cup E^+) \backslash E^-$
$E^+ \subseteq \{(u, v)|u \in \mathscr{C} \vee v \in \mathscr{C}, (u, v) \notin E\}$
$E^- \subseteq \{(u, v)|u \in \mathscr{C} \vee v \in \mathscr{C}, (u, v) \in E\}$
$|E^-| + |E^+| \leq \beta$
where β is the budget of edge updates. Hiding a community requires a measure of concealment, i.e., quantifying how much hidden a community is. Several previous papers proposed such measures, including [6, 23].

3 Deception in Directed Influence Networks (DIN)

This section introduces InflDec, our novel influence-based deception approach for directed networks. We based InflDec on modularity. This is because modularity is very intuitive; it neatly captures our intuition of what forms a community: a group of people sharing denser connections between each other than with people from outside. This intuitive appeal is crucial because we consider modularity a clustering index rather than an objective function. Therefore, we argue that a target community applying our deception mechanism generally improves its concealment level, even if the detector utilizes different clustering indices. In what follows, we incorporate influence into directed modularity and study the effect of intra/inter-community edge modifications that will represent the toolbox of our deception strategy.

Directed Modularity. In this paper, we deal with directed graphs that incorporate influence between nodes as an edge coefficient. Modularity for a directed network can be expressed as in Eq. (2), which is a slightly modified version of the one described by [10]:

$$Q = \frac{1}{m} \sum_{C \in \tilde{C}} \sum_{ij \in C} \mathcal{I}_{ij} - \frac{d_i^{out} d_j^{in}}{m} \tag{2}$$

where d_i^{out} and d_j^{in} are the out/in degrees of nodes i, j respectively. However, since we are considering a DIN, we further modify the preceding function:

$$Q = \frac{1}{\mathcal{I}} \sum_{C \in \tilde{C}} \sum_{ij \in C} \mathcal{I}_{ij} - \frac{\mathcal{I}_i^{\uparrow} \mathcal{I}_j^{\downarrow}}{\mathcal{I}} \tag{3}$$

Now, we consider the effect of edge modifications on ΔQ. We can simplify eq. (3) as follows:

$$\begin{aligned} Q &= \frac{1}{\mathcal{I}} \sum_{C \in \tilde{C}} \sum_{ij \in C} \mathcal{I}_{ij} - \frac{\mathcal{I}_i^{\uparrow} \mathcal{I}_j^{\downarrow}}{\mathcal{I}} = \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2} \sum_{C \in \tilde{C}} \sum_{ij \in C} \mathcal{I}_i^{\uparrow} \mathcal{I}_j^{\downarrow} \\ &= \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2} \sum_{C \in \tilde{C}} \sum_{i \in C} \mathcal{I}_i^{\uparrow} \sum_{j \in C} \mathcal{I}_j^{\downarrow} = \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2} \sum_{C \in \tilde{C}} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} \end{aligned} \tag{4}$$

3.1 *Effect of Edge Modification on the Modularity Loss*

In this section, we study the effect of the addition/deletion of each type of edge on modularity loss ΔQ. This will drive the InflDec deception strategy presented in Sect. 3.2.

3.1.1 Intra-community Edges

Let (u, v) be an intra-community edge deleted from community C^*.

Theorem 1 *Deleting an intra-community edge results in modularity loss, $\Delta Q > 0$, if and only if the following condition holds:*

$$\frac{\eta \mathcal{I}_{uv}}{\mathcal{I}} + \theta + \frac{2\mathcal{I} - \mathcal{I}_{uv}}{\mathcal{I}^2} \sum_{C \in \bar{C}} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} > \mathcal{I}_{C^*}^{\uparrow} + \mathcal{I}_{C^*}^{\downarrow} \tag{5}$$

Proof First, note that the modularity before modification can be expressed as:

$$Q_G = \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2} \sum_{C \in \bar{C}} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} = \frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2} \left(\mathcal{I}_{C^*}^{\uparrow} \mathcal{I}_{C^*}^{\downarrow} + \sum_{C \in \bar{C}'} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} \right) \tag{6}$$

With slight modification of Eq. 6, we obtain modularity after intra-community edge deletion:

$$Q_{G^*} = \frac{\eta - \mathcal{I}_{uv}}{\mathcal{I} - \mathcal{I}_{uv}} - \frac{1}{(\mathcal{I} - \mathcal{I}_{uv})^2} \left((\mathcal{I}_{C^*}^{\uparrow} - \mathcal{I}_{uv})(\mathcal{I}_{C^*}^{\downarrow} - \mathcal{I}_{uv}) + \sum_{C \in \bar{C}'} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} \right) \tag{7}$$

With Eqs. 6 and 7, we obtain:

$$\begin{aligned} \Delta Q = Q_G - Q_{G^*} &= \left[\frac{\eta}{\mathcal{I}} - \frac{1}{\mathcal{I}^2} \left(\mathcal{I}_{C^*}^{\uparrow} \mathcal{I}_{C^*}^{\downarrow} + \sum_{C \in \bar{C}'} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} \right) \right] \\ &\quad - \left[\frac{\eta - \mathcal{I}_{uv}}{\mathcal{I} - \mathcal{I}_{uv}} - \frac{1}{(\mathcal{I} - \mathcal{I}_{uv})^2} \left((\mathcal{I}_{C^*}^{\uparrow} - \mathcal{I}_{uv})(\mathcal{I}_{C^*}^{\downarrow} - \mathcal{I}_{uv}) + \sum_{C \in \bar{C}'} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} \right) \right] \\ &= \frac{\mathcal{I}_{uv}(\mathcal{I} - \eta)}{\mathcal{I}(\mathcal{I} - \mathcal{I}_{uv})} + \frac{\mathcal{I}_{uv}}{\mathcal{I}^2(\mathcal{I} - \mathcal{I}_{uv})^2} \\ &\quad \left[\left((2\mathcal{I} - \mathcal{I}_{uv}) \sum_{C \in \bar{C}} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} \right) - \left(\mathcal{I}^2 \left(\mathcal{I}_{C^*}^{\uparrow} + \mathcal{I}_{C^*}^{\downarrow} - \mathcal{I}_{uv} \right) \right) \right] \end{aligned} \tag{8}$$

With basic algebraic operations we get to:

$$\Delta Q = \frac{\mathcal{I}_{uv}}{(\mathcal{I} - \mathcal{I}_{uv})^2} \left[\mathcal{I} - \eta + \frac{\eta \mathcal{I}_{uv}}{\mathcal{I}} - \mathcal{I}_{C^*}^{\uparrow} - \mathcal{I}_{C^*}^{\downarrow} + \frac{(2\mathcal{I} - \mathcal{I}_{uv})}{\mathcal{I}^2} \sum_{C \in \bar{C}} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} \right] \tag{9}$$

Equation. 9 shows that the sign of ΔQ depends on the term between square brackets, which is positive if and only if:

$$\frac{\eta \mathcal{I}_{uv}}{\mathcal{I}} + \theta + \frac{2\mathcal{I} - \mathcal{I}_{uv}}{\mathcal{I}^2} \sum_{C \in \bar{C}} \mathcal{I}_C^{\uparrow} \mathcal{I}_C^{\downarrow} > \mathcal{I}_{C^*}^{\uparrow} + \mathcal{I}_{C^*}^{\downarrow} \tag{10}$$

The proofs of the following theorems develop along with a similar line Theorem 5; we omit them for the sake of space.

Theorem 2 *Adding an intra-community edge results in modularity loss, $\Delta Q > 0$, if and only if the following condition holds:*

$$\frac{\eta \mathcal{I}_{uv}}{\mathcal{I}} + \mathcal{I}^{\uparrow}_{C^*} + \mathcal{I}^{\downarrow}_{C^*} > \theta + \frac{(2\mathcal{I} + \mathcal{I}_{uv})}{\mathcal{I}^2} \sum_{C \in \bar{C}} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} \tag{11}$$

3.1.2 Inter-Community Edges

Let $C^u, C^v \in \bar{C}$ be two arbitrary communities, then consider a inter-community edge (u, v), where $u \in C^u$ and $v \in C^v$. In this section, we study how does the deletion/addition of (u, v) affect ΔQ.

Theorem 3 *Deleting an inter-community edge will increase modularity if and only if:*

$$\eta \mathcal{I} \mathcal{I}_{uv} + (2\mathcal{I} - \mathcal{I}_{uv}) \sum_{C \in \bar{C}} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} > \mathcal{I}^2 \left(\eta - \mathcal{I}^{\downarrow}_{C^u} - \mathcal{I}^{\uparrow}_{C^v} \right) \tag{12}$$

Theorem 4 *Adding an inter-community edge results in modularity loss, $\Delta Q > 0$, if and only if:*

$$\eta \mathcal{I} \mathcal{I}_{uv} + \mathcal{I}^2 \left(\eta + \mathcal{I}^{\downarrow}_{C^u} + \mathcal{I}^{\uparrow}_{C^v} \right) > (2\mathcal{I} + \mathcal{I}_{uv}) \sum_{C \in \bar{C}} \mathcal{I}^{\uparrow}_{C} \mathcal{I}^{\downarrow}_{C} \tag{13}$$

3.2 *The InflDec Deceptor*

Now we present our InflDec deceptor, a greedy algorithm that leverages the theorems to produce edge updates *that maximizes the modularity loss*. We note that only intra-community edge deletion and inter-community edge addition have the highest potential of inflicting modularity loss. Other edge modifications, such as adding an intra-community edge, do not cause modularity loss, except in the case of an extensive target community, see inequality (11). Therefore, the algorithm has two main tasks. First, it has to choose the best intra/inter-community edge to be deleted or added. Secondly, it has to choose which of the two operations will cause more modularity loss and subsequently execute it. Algorithm 1 takes as input the network with influence coefficients on the edges and the target community. Line 3 calls a procedure that searches $\mathscr{C}$ for the edge with the strongest influence, which will be the candidate for deletion. Next, line 4 selects the highest influence inter-community edge with its destination node in $\mathscr{C}$. Alternatively, line 5 selects the highest influence inter-community edge with its source being inside $\mathscr{C}$. The inter-community

Algorithm 1 The INFLDEC deception algorithm.

1: **procedure** MODIFYNETWORK($G, \mathscr{C}$)
2: **do**
3: intraEdg $\leftarrow$ getHighestIntra($\mathscr{C}$)
4: select$(n_u, n_v) \notin E, n_u \in C_i, n_v \in \mathscr{C} \cap C_j, C_i \in \text{argmax}(\mathcal{I}^{\downarrow}_{C_i}), C_j \in \text{argmax}(\mathcal{I}^{\uparrow}_{C_j})$
5: select$(n_p, n_t) \notin E, n_p \in C_l \cap \mathscr{C}, n_t \in C_m, C_l \in \text{argmax}(\mathcal{I}^{\downarrow}_{C_l}), C_m \in \text{argmax}(\mathcal{I}^{\uparrow}_{C_m})$
6: $\mathcal{ML}_{del} \leftarrow$ getDelLoss($intraEdg, \bar{C}, G$)
7: $\mathcal{ML}^{\uparrow}_{add} \leftarrow$ getAddLoss($(n_p, n_t), \bar{C}, G$)
8: $\mathcal{ML}^{\downarrow}_{add} \leftarrow$ getAddLoss($(n_u, n_v), \bar{C}, G$)
9: **if** $\mathcal{ML}_{del} \geq \max(\mathcal{ML}^{\uparrow}_{add}, \mathcal{ML}^{\downarrow}_{add})$ and $\mathcal{ML}_{del} > 0$ **then**
10: $G \leftarrow (V, E \backslash \{\text{intraEdg}\})$
11: **else**
12: **if** $\mathcal{ML}^{\uparrow}_{add} \geq \mathcal{ML}^{\downarrow}_{add}$ and $\mathcal{ML}^{\uparrow}_{add} > 0$ **then**
13: $G \leftarrow (V, E \cup \{(n_p, n_t)\})$
14: **else**
15: **if** $\mathcal{ML}^{\downarrow}_{add} > 0$ **then**
16: $G \leftarrow (V, E \cup \{(n_u, n_v)\})$
17: **end if**
18: **end if**
19: **end if**
20: $\beta \leftarrow \beta - 1$
21: **while** $\beta > 0$ and ($\mathcal{ML}^{\uparrow}_{add} > 0$ or $\mathcal{ML}^{\downarrow}_{add} > 0$ or $\mathcal{ML}_{del} > 0$)
22: **end procedure**

edges selected are our candidates for addition. Lines 6–8 compute the modularity loss caused by deleting/adding the three selected edges. Moreover, the algorithm performs the modification using the edge yielding the highest possible modularity loss. Line 20 keeps track of the available budget for modifications.

4 Experimental Evaluation

This section reports on an experimental evaluation. The overall goal is to gain insight into how our approach, which considers both edge directions and node influence, is effective. Moreover, we want to compare our influence-based community deception approach for directed networks with the state-of-the-art focused on undirected networks. The comparison will shed further light on our novel techniques' effectiveness in hiding capabilities. In what follows, we describe the experimental setting and then report on the experimental results. The algorithm has been implemented in Python. Code and datasets are available online.[1]

Detectors. We considered community detection algorithms (detectors) that will act as adversaries to the deception techniques focusing on approaches that work on directed networks and support edge weights, that in our case model influence. We considered the following algorithms available in the cdlib library[2]: Leiden [21] (`leiden`);

[1] https://communitydeception.wordpress.com/.

[2] https://cdlib.readthedocs.io.

Table 3 Datasets

Network	[4] |V|	[4] |E|	Number communities			
			`leiden`	`dm`	`surprise`	`gemsec`
`Freeman`	50	500	5	5	7	5
`Email`	∼1K	∼25K	28	32	21	16
`Anybeat`	∼12K	∼67K	129	81	43	112
`WikiVote`	∼7K	∼103K	30	34	43	49
`Facebook`	∼9K	∼142K	6	5	6	5
`Epinions`	∼75K	∼508K	795	896	Timeout	Timeout
`Slashdot`	∼77K	∼905K	825	1115	Timeout	Timeout

Directed modularity [10] (`dm`); Surprise community [20] (`surprise`); Gemsec [18] (`gemsec`).

Deceptors. We considered the following deceptors: Delete Internal Connect External [23] (`DICE`): this community deception algorithm is based on the heuristic of deleting intra-community edges and adding inter-community edges; Modularity Minimization [6] (`modMin`): this approach corrects for some issues with `DICE`; the authors of `modMin` showed that in some cases, `DICE` fails to perform edge updates that minimize modularity; Safeness-based deception [6] (`SAF`): this approach introduces safeness maximization for community deception; Permanence-based deception [13] (`NEUR`): this approach is based on permanence minimization; Random edge updates (`RND`): we consider an approach that randomly selects both the type of update and the endpoints of the edge addition/deletion.

Datasets. As this paper aims to introduce influence-based deception for directed networks, we focused on various real directed social networks. In order to compute influence between nodes, we used the approach described in Kumar et al. [9], which is also able to predict the influence of missing links; we leave as a future work the investigation of further influence measures. These networks are available online.[3],[4],[5]

Table 3 gives an overview of the networks considered. The table also reports, for each network, the number of communities found by the Detectors considered. We note that some of the detectors could not complete community detection on the more extensive networks after a timeout of 3h.

Evaluation Methodology. To test deception algorithms, we cannot directly apply the deception score introduced in Fionda et al. [6] This is because this score was devised for undirected and unweighted networks. Therefore, as similarly done by the state-of-the-art [13], we considered a combination of *community spread* and *community ratio*, that is, in how many communities the member of $\mathscr{C}$ are scattered and in which percentage; the large the value the better the hiding.

[3] https://data4goodlab.github.io/dataset.html.

[4] https://snap.stanford.edu.

[5] https://toreopsahl.com/datasets.

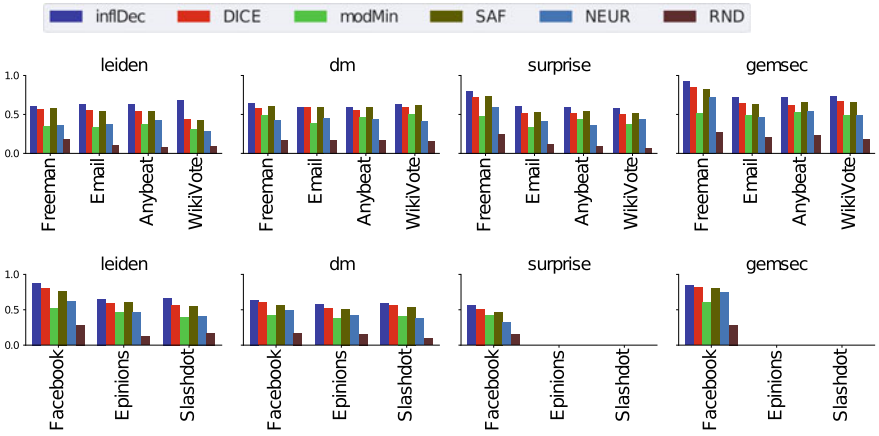


Fig. 1 Comparison between directed and undirected deception approaches

4.1 Deception Score Comparison

We now compare our novel approach for community deception based on influence in directed networks with the state-of-the-art. We note that competitors were not designed to work on directed networks; neither these approaches support edge weights, that in our case model influence. We also note that most of the datasets considered are directed networks, which underlines the importance of adding directions in social network relations.

For each experiment round, we chose one of the communities found by the detection algorithm we want to deceive by looking at the distribution of the sizes of the communities found; this represents the worst-case scenario where the community is completely revealed to the detector. Hence, the initial deception score is zero. For the competitors, we treated the networks as undirected. Moreover, we focus on a budget of updates equal to 60% of the edges of $\mathscr{C}$ as this configuration worked best for all approaches. Figure 1 reports results in terms of deception score. In the figure, each column represents a detection algorithm; the x-axis represents a network in each subfigure, while the y-axis is the deception score.

In all networks, INFLDEC performs better than the undirected approaches; this is true for all detectors. However, we observe that for `leiden` in the smaller networks, the deception score tends to be lower than in the larger ones. As an example, on Freeman, the smallest network, this value reaches 0.61, while on `Slashdot` the largest network is 0.66. Moreover, `gemsec` seems to be the approach less robust to all deception techniques; deception score values are higher than those obtained when deceiving other detectors. It is interesting to look at the relative performance of INFLDEC, `DICE`, and `modMin` all based on modularity. `DICE`, which is the simplest approach based on randomly deleting internal edges and adding external, seems to perform better than `modMin`, which adopts a more elaborated strategy to determine the best set of edge updates that also looks at the degree of communities. However,

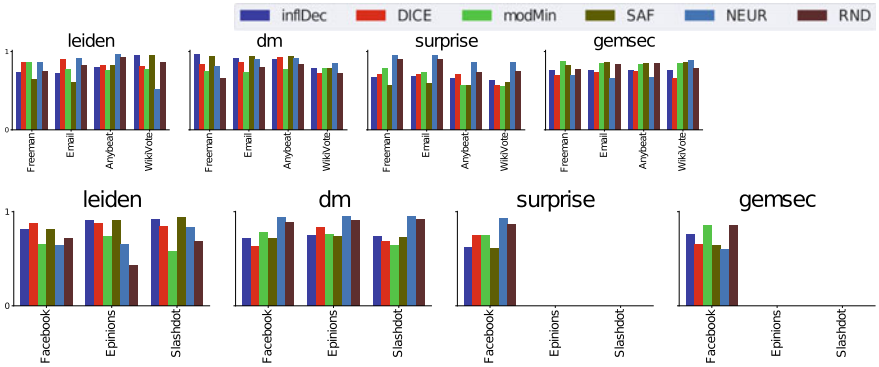


Fig. 2 NMI comparison

we note that these two approaches were devised for undirected networks, while the networks considered here are directed. Nevertheless, INFLDEC's approach that optimizes the modularity loss for deception by taking into account both edge directions and influence performs consistently better. The other deceptors `SAF` and `NEUR`, based on node safeness and node permanence performed better than `DICE` and `modMin`. As one would expect, the worst-performing deceptor is `RND`, which adds/remove edges randomly starting from $\mathscr{C}$'s members. This is an important result underlining that achieving deception requires some strategy and does not just amount to deleting and adding edges.

Normalized Mutual Information. To shed more light on the impact of deception on the community structure found by a detection algorithm, we computed the normalized mutual information (NMI) score comparing the communities before and after applying deception techniques. Each column in Fig. 2 represents a detection algorithm where the x-axis represents one of the networks and the y-axis the value of NMI.

The values of NMI seem to be lower for smaller networks meaning that the community structure changes more than in the case of larger networks. However, in both cases, the value is greater than for all deception approaches but `RND`. In general, INFLDEC seems to be the algorithm that best preserves the original community structure. This again, points out how a carefully designed deception strategy can act to hide a community while not impacting too much on the other part of the community structure. Again, `RND` significantly changes the community structure while not providing a good deception score.

5 Concluding Remarks and Future Work

We introduced the novel problem of influence-based community deception, that is, hiding the members of a community from community detection algorithms in a

setting that considers both directed networks and edge weights that model influence. We devised a strategy based on modularity, which performs better than the state-of-the-art. In the future, we plan to tackle the problem from the perspective of node updates.

References

1. Bedi, P., Sharma, C.: Community detection in social networks. Wiley Interdisc. Rev. Data Min. Knowl. Disc. **6**(3), 115–135 (2016)
2. Chakraborty, T., Srinivasan, S., Ganguly, N., Mukherjee, A., Bhowmick, S.: On the permanence of vertices in network communities. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'14, pp. 1396–1405. Association for Computing Machinery, New York, NY, USA (2014)
3. Chakraborty, T., Srinivasan, S., Ganguly, N., Mukherjee, A., Bhowmick, S.: Permanence and community structure in complex networks. ACM TKDD **11**(2), 1–34 (2016)
4. Chen, J., Chen, L., Chen, Y., Zhao, M., Yu, S., Xuan, Q., Yang, X.: Ga-based q-attack on community detection. IEEE Trans. Comput. Soc. Syst. **6**(3), 491–503 (2019)
5. Danon, L., Diaz-Guilera, A., Duch, J., Arenas, A.: Comparing community structure identification. J. Stat. Mech. Theor. Exper. **9** (2005)
6. Fionda, V., Pirrò, G.: Community deception or: how to stop fearing community detection algorithms. IEEE Trans. Knowl. Data Eng. **30**(4), 660–673 (2018)
7. Fortunato, S.: Community detection in graphs. Phys. Rep. **486**(3), 75–174 (2010)
8. Ghosh, R., Lerman, K.: Community detection using a measure of global influence. In: Proceedings of the Second International Conference on Advances in Social Network Mining and Analysis, SNAKDD'08, pp. 20–35. Springer-Verlag, Berlin, Heidelberg (2008)
9. Kumar, S., Spezzano, F., Subrahmanian, V., Faloutsos, C.: Edge weight prediction in weighted signed networks. In: 2016 IEEE 16th International Conference on Data Mining (ICDM), pp. 221–230. IEEE (2016)
10. Leicht, E.A., Newman, M.E.: Community structure in directed networks. Phys. Rev. Lett. **100**(11), 118703 (2008)
11. Lu, Z., Zhu, Y., Li, W., Wu, W., Cheng, X.: Influence-based community partition for social networks. Comput. Soc. Netw. **1**(1), 1 (2014)
12. Ma, T., Liu, Q., Cao, J., Tian, Y., Al-Dhelaan, A., Al-Rodhaan, M.: LGIEM: global and local node influence based community detection. Future Gener. Comput. Syst. **105**, 533–546 (2020)
13. Mittal, S., Sengupta, D., Chakraborty, T.: Hide and seek: outwitting community detection algorithms. IEEE Trans. Comput. Soc. Syst. (2021)
14. Nagaraja, S.: The Impact of Unlinkability on adversarial community detection: effects and countermeasures. In: PETS, pp. 253–272 (2010)
15. Newman, M.E.: Modularity and community structure in networks. PNAS **103**(23), 8577–8582 (2006)
16. Newman, M.E.J.: Fast algorithm for detecting community structure in networks. Phys. Rev. E **69**, 066133 (2004)
17. Rezaeimehr, F., Moradi, P., Ahmadian, S., Qader, N.N., Jalili, M.: TCARS: time- and community-aware recommendation system. Future Gener. Comput. Syst. **78**, 419–429 (2018)
18. Rozemberczki, B., Davies, R., Sarkar, R., Sutton, C.: GEMSEC: graph embedding with self clustering. In: Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 65–72 (2019)
19. Sarma, D., Alam, W., Saha, I., Alam, M.N., Alam, M.J., Hossain, S.: Bank fraud detection using community detection algorithm. In: 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), pp. 642–646 (2020)

20. Traag, V.A., Aldecoa, R., Delvenne, J.-C.: Detecting communities using asymptotical surprise. Phys. Rev. E **92**(2), 022816 (2015)
21. Traag, V.A., Waltman, L., Van Eck, N.J.: From Louvain to Leiden: guaranteeing well-connected communities. Sci. Rep. **9**(1), 1–12 (2019)
22. Wang, W., Street, W.N.: A novel algorithm for community detection and influence ranking in social networks. In: 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014), pp. 555–560 (2014)
23. Waniek, M., Michalak, T.P., Wooldridge, M.J., Rahwan, T.: Hiding individuals and communities in a social network. Nat. Human Behaviour **2**(2), 139–147 (2018)

AutoGF: Runtime Graph Filter Tuning for Community Node Ranking

Emmanouil Krasanakis, Symeon Papadopoulos, and Ioannis Kompatsiaris

Abstract A recurring graph analysis task is to rank nodes based on their relevance to overlapping communities of shared metadata attributes (e.g. the interests of social network users). To achieve this, approaches often start with a few example community members and employ graph filters that rank nodes based on their structural proximity to the examples. Choosing between well-known filters typically involves experiments on existing graphs, but their efficacy is known to depend on the structural relations between community members. Therefore, we argue that employed filters should be determined not during algorithm design but at runtime, upon receiving specific graphs and example nodes to process. To do this, we split example nodes into training and validation sets and either perform supervised selection between well-known filters, or account for granular graph dynamics by tuning parameters of the generalized graph filter form with a novel optimization algorithm. Experiments on 27 community node ranking tasks across three real-world networks of various sizes reveal that runtime algorithm selection selects near-best AUC and NDCG among a list of 8 popular alternatives, and that parameter tuning yields similar or improved results in all cases.

Keywords Node ranking · Graph signal processing · Parameter tuning

1 Introduction

When graph nodes are attributed (e.g. they are social network users and attributes are their areas of interest), they can be organized into communities of shared metadata

E. Krasanakis (✉) · S. Papadopoulos · I. Kompatsiaris
Centre for Research and Technology Hellas, Information Technologies Institute, Thessaloniki, Greece
e-mail: maniospas@iti.gr

S. Papadopoulos
e-mail: papadop@iti.gr

I. Kompatsiaris
e-mail: ikom@iti.gr

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_15

attributes [26]. By definition, these communities are not tied to specific high-level structural characteristics, such as strong connectivity between nodes. Still, it is commonly accepted that attributes could correlate to low-level dynamics leading to the creation of edges, in which case graph structure can help predict metadata. For example, nodes of social network graphs often exhibit homophilous behavior [23], a term describing their tendency to form edges with others of similar attributes. Then, tightly knit structural communities become good predictors of parts of -but not of whole- metadata communities [35].

A recurring graph analysis task, which we tackle in this work, is to rank nodes based on their relevance to communities sharing metadata attributes of interest [16, 19, 29, 32, 33]. Ranking provides greater granularity than clear-cut predictions, for example in the scope of recommending more community members. It also respects overlaps and fuzzy boundaries between communities [21]. Furthermore, node relevance scores obtained during ranking are often the core of more sophisticated systems, such as graph neural networks for classification after initial neural estimations [13, 15] and post-processing strategies that threshold transformations of scores to predict community membership [3].

A popular use case for community node ranking, which we also follow, is to start with a few known community members serving as examples, and inferring the relatedness of all nodes to respective communities based on their structural proximity to the examples [19, 34, 37]. This task is performed independently for one or more communities. Assumptions about what constitutes proximity have coalesced under the field of graph signal processing [Sect. 2], where they are modeled with ad-hoc graph filters and controlled by a small number of parameters [8, 25].[1] Different filters and parameters match different types of communities. For example, filter efficacy could depend on the number of community members [1, 12, 19]. As a result, deployed filters may work well in certain graphs but not necessarily in others. By extension, running filter-based tools 'off-the-shelf' in deployed systems risks producing node ranks of lesser quality.

In this work, we address the above issue by exploiting autotune principles [17] for runtime selection of graph filters. We explore two strategies: a) choosing the best among a list of promising filters, and b) tuning the parameters of a generalized filter form. For the second strategy, we also introduce a novel tuning algorithm that keeps examining a wide search breadth in the solution space but converges within a bounded number of filter runs. The effectiveness of our approach is corroborated on 27 community node ranking tasks across 3 real-world graphs of different domains. Results indicate that neither strategy falls significantly behind best-performing ad-hoc filters when optimizing popular node rank quality measures. Furthermore, parameter tuning frequently captures structural proximity better than ad-hoc assumptions and improves rank quality.

[1] Most non-filter node ranking algorithms, such as k-shell decomposition and variations [36], blindly rank the importance of nodes within graph structures and can not personalize ranks in terms of importance to specific communities.

This paper is organized as follows. In Sect. 2 we present graph filters as an approach for ranking nodes with respect to metadata communities, alongside a generalized literature filter form. In Sect. 3, we describe our runtime filter selection approach and its implementation choices. We also present a novel algorithm for tuning parameters of generalized graph filters. In Sects. 4 and 5 we evaluate our approach in real-world data and discuss practical applicability and potential risks. Finally, in Sect. 6 we summarize our findings and present promising research directions.

2 Background

Graph edges are often represented by adjacency matrices A with elements $A[u, v] = \{1 \text{ if edge } (u, v) \text{ exists, } 0 \text{ otherwise}\}$. These are symmetrically normalized by weighing edges to mitigate the importance of highly connected nodes per:

$$W = D^{-1/2} A D^{-1/2}$$

where D with elements $D[u, v] = \{\sum_{v'} A[u, v'] \text{ if } u = v, 0 \text{ otherwise}\}$ are diagonal matrices of node degrees. The graph's spectrum can be defined as the eigenvalues of the normalized adjacency matrix.[2] In detail, eigenvalue decomposition yields $W = U \Lambda U^{-1}$, where Λ are diagonal matrices of eigenvalues $\Lambda = diag([\lambda_1, \lambda_2, \ldots, \lambda n])$ and U are orthogonal matrices whose columns hold the corresponding eigenvectors. For connected graphs, eigenvalues of the normalized adjacency matrix are real-valued and reside in the unit range $\lambda_i \in [-1, 1]$.

Graph signal processing [24, 27, 28] manipulates signals p whose elements $p[u]$ correspond to values stored at nodes u. To do this, it defines their graph Fourier transform as $\mathcal{F}\{p\} = U^{-1} p$ and its inverse as $\mathcal{F}^{-1}\{\mathcal{F}\{p\}\} = U \mathcal{F}\{p\}$. Then, it observes that $W^n = U \Lambda^n U^{-1} \Rightarrow H(W) = U H(\Lambda) U^{-1}$ for function forms $H(\cdot)$ whose Taylor expansions exist around zero, and defines filters $H_{\mathcal{F}} = [H(\lambda_1), H(\lambda_2), \ldots H(\lambda_n)]$ in the Fourier space, whose parameters arise through transformations $H(\lambda_i)$ of eigenvalues λ_i. Graph filters can be applied on signals via an element-wise multiplication $\odot$ on their Fourier transform $\mathcal{F}\{p\}$. The outcome of filtering in the node space becomes:

$$\mathcal{F}^{-1}(H_{\mathcal{F}} \odot \mathcal{F}\{p\}) = U H(\Lambda) U^{-1} p = H(W) p$$

During the above analysis, the function forms $H(\cdot)$ determining graph filters can be parameterized in terms of their Taylor coefficients $h_0, h_1, \ldots$ per:

$$H(W) = \sum_{k=0}^{\infty} h_k W^k$$

[2] The graph's spectrum can also be defined as the eigenvalues $1 - \lambda_i$ of its normalized Laplacian $I - W$. This, too, can express filters as infinite-degree polynomials of W.

As $W^k p$ propagates graph signals p at k hops away through normalized adjacency matrices W, the above formula describes a weighted aggregation of multi-hop signal propagation. Filters matching different structural assumptions arise from different coefficients h_k. Two well-known filters are personalized PageRank [3, 4] and heat kernels [16]. These respectively adopt degrading hop weights $h_k = (1-a)a^k$ and the kernel $h_k = e^{-t}t^k/k!$ for parameters $a \in [0, 1)$ and $t \in \{1, 2, 3, \dots\}$.

Given the above formulation, graph filtering can rank how nodes pertain to communities of interest [19, 34, 37]. Approaches start with signals p whose values capture whether nodes v belong to sets $\mathcal{C}$ of known community members per:

$$p[v] = \{1 \text{ if } v \in \mathcal{C}, 0 \text{ otherwise}\}$$

Then, for graphs with normalized adjacency matrices W, graph filters $H(W)$ yield new signals $r = H(W)p$ with elements $r[u]$ corresponding to how proximate nodes u are to known member sets C under some understanding of proximity. Finally, nodes are ranked by order of their proximity to known members.

3 Tuning Graph Filters at Runtime

As previously mentioned, graph filters for community node ranking should ideally be selected at runtime, after graphs and example community members become known and therefore can be used to understand underlying structural features. We consider best-performing filters those with higher node rank quality, for instance measured with the area under curve of the receiver operating characteristics (AUC) [6] and the normalized discounted cumulative gain across all graph nodes (NDCG) [14]. Employed measures should coincide with practical objectives on unknown test data. For example, high AUC indicates higher ranks for community members than non-members, whereas high NDCG verifies the community membership of top-ranked nodes.

To optimize node rank quality at runtime, we follow an autotune paradigm that searches through the parameter space of black box algorithms to optimize validation objectives. Originally, the term was associated with specific approaches [17], but nowadays broadly describes automatic selection of machine learning model parameters. This comes at the expense of multiple algorithm runs, but there exist mature solutions for fast computation of graph filters [18].

Our approach starts with sets $\mathcal{C}$ of known community members among graph nodes, which are organized into binary graph signals p per the formulation of Sect. 2. We split known members into non-overlapping subsets $\mathcal{C}_{train}, \mathcal{C}_{valid} \subseteq \mathcal{C}$, which correspond to "training" graph signals p_{train} to be used as filter inputs, and desired output validation signals p_{valid}. We employ evaluation measures $\mathcal{M}(\cdot, \cdot)$, such as AUC or NDCG, that assess node rank quality via pairwise comparison between predictions and ground truth, and select filters with high $\mathcal{M}(r_{train}, p_{valid})$ for predicted ranking scores $r_{train} = H(W)p_{train}$. We avoid overfitting by computing measures

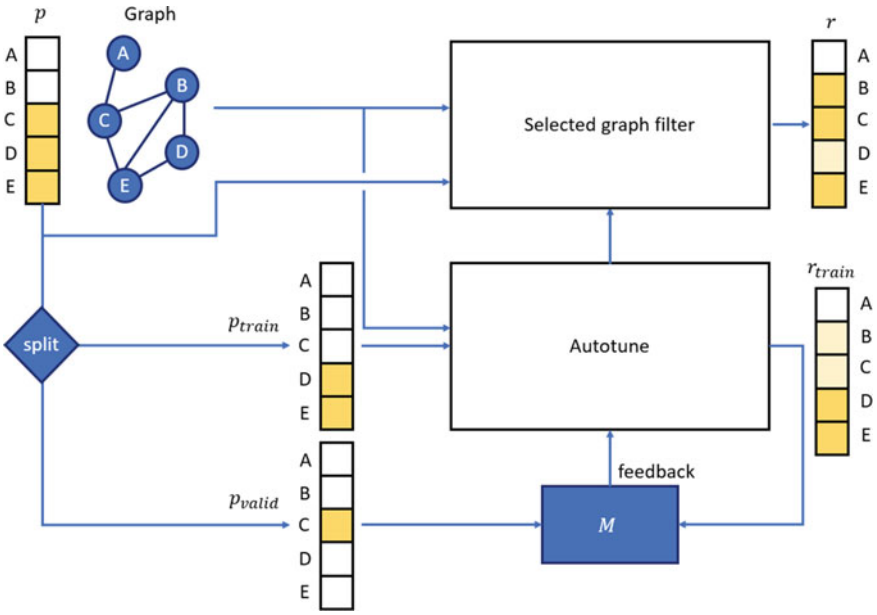


Fig. 1 Overview of graph filter autotuning under measures $\mathcal{M}$. Example nodes are split between training and validation graph signals, where the latter assume the role of ideal training outputs. Highlighted signal elements correspond to higher node values

only across non-training nodes. As long as graphs exhibit homogeneous correlations between communities and edges, filters maximizing validation evaluation are expected to also maximize $\mathcal{M}(r, p_{test})$ for $r = H(W)p$ on nodes other than known community members, where p_{test} are unknown ideal test labels. Our pipeline's data flow is summarized in Fig. 1.

We follow two strategies for graph filter selection by the autotune component of our approach. The first is to perform a simple selection among a list of popular filters, such as those we experiment with later on. The second is to start with the parameterized graph filter form presented in Sect. 2 and tuning a vector of its parameters $h = [h_0, h_1, \ldots, h_K]^T$ to optimize validation objectives. We explore only non-negative parameters to match the widespread literature practice of introducing only non-negative correlations between hops and high-quality node ranks. Then, without loss of generality, we tune all parameters in the range [0, 1].

When tuning graph filter parameters on non-differentiable (potentially even non-convex) validation objectives, a first take is to adopt existing generic black box optimization algorithms [7, 10]. However, these do not guarantee convergence for all deployed system inputs. At the same time, adjusting one graph filter parameter to control the importance of propagating graph signals a fixed number of hops away could drastically affect the validity of other propagation weights. This hypothesis is also corroborated by experiments later on.

To address the above concerns, we propose an algorithm for graph filter parameter tuning that maintains a broad parameter search space while converging in finite time. This involves cycling through parameters, and progressively minimizing a loss function $\ell(h) = 1 - \mathcal{M}(H(W)p_{train}, p_{valid})$ by finding the best permutation around each parameter with coarse linear search. As tuning progresses, we shrink the search range, so that small permutations around ideal values are eventually found. Intuitively, this is equivalent to moving the center of the selected rectangle chosen for each parameter based on subsequent selections of other parameters. If shrinking is slow enough, by the time when parameter permutation breadths become small, potential combinations with drastically different permutations of other parameters have already been considered.

Conceptually, this procedure is a variation of divided rectangles (DIRECT) [9] that, instead of keeping many candidate rectangles to divide, keeps only one, though of larger width than the partition. This practice corresponds to the shrinking radius technique proposed for non-convex block coordinate optimization [22], although the two are not mathematically equivalent due to the finite sum of rectangle widths that limits the optimization within the hypercube of searched parameters instead of looking at an unconstrained range.

In detail, we start from the center of the parameter hypercube and cycle through parameters i. For each of those, we consider the range $\Delta h[i]$ in which to search for new solutions and partition it uniformly to $2P + 1$ candidate points, P of which examine higher parameter values and an equal number lower values. Values are snapped to the search bounds 0 or 1 if they subceed or exceed those respectively. Perturbations form a set H_{search} of potential parameter vectors, of which we select the one minimizing the loss. Finally, we contract the search range by division with constant $T > 1$ and move on to the next parameter. Cycling through parameters stops when loss reduction becomes smaller than a tolerance ϵ across all parameters. This process is outlined in Algorithm 1.

Algorithm 1 Parameter tuning

Inputs: parameter loss $\ell(h)$, tolerance ϵ, line search partitions P, range shrinking T
Outputs: near-optimal vector of K parameters
$h \leftarrow [0.5] \times K$, $\Delta h \leftarrow [0.5] \times K$, $err \leftarrow [\infty] \times K$, $i \leftarrow 0$
while $\max_i err[i] > \epsilon$ **do**
  $u_i \leftarrow$unit vector with $u_i[j] = \{1 \text{ if } i = j, 0 \text{ otherwise}\}$
  $H_{search} \leftarrow \{max(0, min(1, h + u_i \cdot \Delta h[i] \cdot (p/P - 1))) \mid p = 0, 1, \ldots, 2P\}$
  $err[i] \leftarrow \ell(h) - \min_{h \in H_{search}} \ell(h)$
  $h \leftarrow \arg\min_{h \in H_{search}} \ell(h)$
  $\Delta h[i] \leftarrow \Delta h[i]/T$
  $i \leftarrow (i + 1) \, mod \, K$
return h

If the objective $\ell(h)$ is Lipschitz continuous with Lipschitz constant $L < \infty$ (when the loss is differentiable, this means that $\sup \|\nabla \ell(h)\| \leq L$), it is easy to see that the division of the parameter permutation radius by T every K iterations lets the algo-

rithm run in amortized time $O\big(K(\text{run } \ell(h)) \log_T \frac{L}{\epsilon}\big)$. If graph nodes are fewer than edges (as happens for connected graphs), in which case the running time of $\ell(h)$ is not dominated by node validation. Using sparse matrix multiplication to iteratively compute Krylov space elements $\{W^k p_{train} |, k = 0, \ldots, K\}$ by left-multiplying previous ones with W, graph filters run in time $O(KE)$, where E is the number of graph edges. Thus, our graph filter parameter tuning mechanism can be implemented to run in amortized time:

$$O\big(K^2 E(\log_T L - \log_T \epsilon)\big)$$

Running time scales linearly with the number of edges and quadratically with the number of parameters. We recommend and employ default parameters $P = 2, T = 1.01$, which suffice to minimize the Beale and Booth functions often used in optimization benchmarks [2] to 10^{-6} parameter (instead of loss) tolerance.

4 Experiment Setup

We experiment on three publicly available real-world graphs with metadata communities. First is the Amazon co-purchasing graph [20], whose nodes and edges correspond to products and frequent co-purchases. Products are organized into metadata communities based on their type (e.g. book, movie) attribute. Second is the Citeseer citation graph [11], whose nodes and edges correspond to scientific publications and citations. Publications are organized into communities based on scientific field. Third is the Maven dependency graph [5], whose nodes and edges correspond to software projects and dependencies. Projects are organized into communities based on the organization responsible for their development.

These graphs were chosen for experimentation on merit of comprising metadata communities with enough member nodes to conduct robust validation. To not overrepresent graphs with many communities and obtain enough validation nodes later on, we experiment with the first three communities of each graph with at least 500 nodes. We treat all edges as undirected so that symmetric normalization of filters is applicable. Community details are summarized in Table 1.

For each of the the above-described communities, we generate three splits of known-test members by assuming that known members are uniformly sampled to comprise 10, 20, or 30% of total members. We remind that validation nodes can only be subsampled from known members. Sampling is seeded to ensure reproducibility and fair comparison between approaches. In total, experiments on 9 communities create $9 \cdot 3 = 27$ different known-test member splits. For each split, we consider two different node ranking objectives; optimizing AUC, and optimizing NDCG. Thus, we obtain $27 \cdot 2 = 54$ experiment setups. Sampling and splits are seeded so that evaluations of different graph filters in the same setups are comparable.

We investigate the ability of our approach to produce high-quality community node ranks compared to ad-hoc graph filters and parameters often encountered in the

Table 1 Details of communities we experiment on

Community	Graph	Nodes	Edges	Members
amazon0	Amazon	554,789	3,577,450	280,507
amazon1	Amazon	554,789	3,57,7450	64,915
amazon2	Amazon	554,789	3,577,450	17,966
citeseer0	Citeseer	3327	9464	596
citeseer1	Citeseer	3327	9464	668
citeseer2	Citeseer	3327	9464	701
maven0	Maven	1,965,359	19,431,302	1687
maven1	Maven	1,965,359	19,431,302	1043
maven2	Maven	1,965,359	19,431,302	49,883

literature. We compare the following alternatives, all of which we integrated in the *pygrank* Python library [18] alongside experiment setups:

- *ppr a* [3, 4, 25]. Personalized PageRank that performs stochastic random walks with restart probabilities $1 - a$ at each step [29]. We test common values $a \in \{0.5, 0.85, 0.9, 0.99\}$ and compute filters to numerical tolerance 10^{-9}.
- *hk k* [8]. Heat kernels that form bandpass windows around desired propagation hops k. We test common window centers $k \in \{2, 3, 5, 7\}$.
- *select* [this work]. Runtime selection of the best among ppr a and hk k by withholding a 10% validation subset of known community members. When graphs are unknown during algorithm selection, this becomes a baseline for tuning.
- *tune* [this work]. Tuning a generalized graph filter with 40 parameters, where the filter is obtained with non-zero Taylor coefficients $h_0 = 1$ and tuned $h_1, \ldots, h_{40}$ via Algorithm 1 towards maximizing measures of choice on the same 10% validation subset as in *select*. Optimization absolute deviation tolerance is set to $\epsilon = 10^{-6}$.
- *tuneLBFGSB* [ablation study]. A variation of *tune* that substitutes our tuning algorithm with the L-BFGS-B optimizer [7] provided by the *scipy* library [31] with default parameters and 10^{-6} percentage decrease on the evaluation function as a stropping criterion to make sure that tuning does not stop early. This is a popular optimizer still used for parameter search [30] and approximates Neuton's method while limiting the number of computations to only first-order gradients. Experiments with the Nelder-Mead optimizer yielded similar or worse results that we do not report due to space constraints.

5 Experiment Results

Tables 2 and 3 present the quality of community node ranking across experiment setups in terms of AUC and NDCG respectively. Before exploring graph filter selection, we verify that individual ad-hoc filter efficacy varies across communities and

training-test splits. Indeed, no explored filter outperforms the rest in all experiments. For instance, ppr0.99 is often the best in Amazon communities, but also the worst in Maven communities, where it lags behind others up to 0.035 AUC. Runtime filter selection would be useful as long as it lags less behind.

Choosing between ad-hoc filters with our validation strategy does not always retrieve the best-performing ones. We attribute this behavior to few missing examples still impacting the ideal filter propagation weights needed for high-quality node ranking. Withholding fewer nodes could degrade validation robustness and future research could investigate new mechanisms to improve generalization. For the time being, selection of best among existing alternatives at runtime chooses the best filters in 31/54 settings. But, even when this scheme fails to identify the best filter, it often retrieves near-best ones that at worst lag behind only by 0.011 in terms of AUC or NDCG, where this gap usually shrinks to 0.001.

Parameter tuning with Algorithm 1 outperforms all ad-hoc filters in 40/54 experiment settings. This induces up to 0.010 AUC and 0.033 NDCG improvements, indicating that it manages to discover nuanced notions of structural proximity. It lags behind by at worst 0.007 on account of either measure, and often by much less. Compared to selecting among filters, tuning yields better evaluation outcomes in 49/54 of experiment settings. As such, we recommend it as an out-of-the-box solution for community node ranking in new graphs, especially if structural characteristics correlating to the formation of communities are not known beforehand. Finally, comparing our optimization algorithm to L-BFGS-B, the latter induces marginal improvements in the Citeseer graph, but falls significantly behind -even compared to filter selection- in the Amazon and Maven graphs. This corroborates the need for retaining a wide parameter search space.

In relation to applying our methodology, we experimented on communities with enough example members to achieve a robust evaluation when randomly withholding 10% of them. Fewer known members may not yield robust validation strategies and we hereby caution against blindly applying our methodology when too few members are known. In principle, we expect our approach to work well -and therefore be applicable on- community node ranking based on at least the same number of known members (at least 50) as in our experiments.

As evidence that tuning discovers non-trivial graph propagation schemes, Fig. 2 shows the first 41 parameters of high-AUC filters for citeseer0 with 30% known members. There, tuning discovers a different propagation strategy than ad-hoc filters, which subsequently manages to (slightly) improve the best filter in Table 2. Moreover, Fig. 3 shows that tuning is tailored not only to communities but even to specific sets of example nodes, yielding drastically different filters for the same communities. Given that tuned graph filters generally outperform others, this finding corroborates our hypothesis that filters should be selected at runtime to match the characteristics of data they are about to process. Finally, filter differences between different fractions of community examples support our practice of withholding only a small fraction of validation nodes.

Table 2 Test set AUC of community node ranks for ad-hoc filters and those obtained through runtime tuning on the same measure

		Ad-hoc								Autotune		
Com.	Examples (%)	ppr0.5	ppr0.85	ppr0.9	ppr0.99	hk2	hk3	hk5	hk7	Select	Tune	tuneLBFGSB
amazon0	10	0.825	0.844	0.853	0.901	0.823	0.825	0.832	0.844	0.901	**0.903**	0.883
amazon0	20	0.820	0.855	0.870	0.910	0.817	0.824	0.844	0.865	0.910	**0.918**	0.900
amazon0	30	0.815	0.868	0.884	0.914	0.810	0.824	0.855	0.880	0.914	**0.924**	0.908
amazon1	10	0.930	0.934	0.935	**0.944**	0.930	0.930	0.931	0.933	**0.944**	0.943	0.940
amazon1	20	0.941	0.946	0.948	0.954	0.941	0.941	0.944	0.946	0.954	**0.955**	0.953
amazon1	30	0.946	0.953	0.955	0.960	0.946	0.947	0.951	0.954	0.960	**0.961**	0.959
amazon2	10	0.960	0.964	0.965	**0.968**	0.959	0.960	0.961	0.963	**0.968**	**0.968**	0.967
amazon2	20	0.970	0.974	0.974	0.975	0.970	0.971	0.972	0.974	0.975	**0.977**	0.976
amazon2	30	0.973	0.976	0.977	0.976	0.972	0.973	0.975	0.976	0.977	**0.979**	0.978
citeseer0	10	0.778	0.789	0.792	0.793	0.775	0.777	0.781	0.784	0.793	**0.795**	**0.795**
citeseer0	20	0.841	0.849	0.850	0.845	0.838	0.840	0.844	0.847	0.845	**0.852**	**0.852**
citeseer0	30	0.859	0.867	0.868	0.861	0.856	0.859	0.863	0.866	0.868	**0.869**	**0.869**
citeseer1	10	0.805	**0.807**	**0.807**	0.798	0.806	0.805	0.805	0.806	0.798	0.805	0.806
citeseer1	20	0.820	0.823	**0.824**	0.813	0.820	0.820	0.821	0.823	0.820	0.819	0.823
citeseer1	30	0.816	0.821	0.821	0.811	0.816	0.816	0.819	0.820	0.816	**0.821**	**0.821**
citeseer2	10	0.659	0.669	0.672	0.675	0.656	0.657	0.661	0.664	0.656	**0.670**	0.676
citeseer2	20	0.718	0.727	0.730	0.731	0.716	0.717	0.721	0.724	0.716	0.732	**0.733**
citeseer2	30	0.767	0.773	0.774	0.769	0.766	0.767	0.770	0.773	0.773	0.775	**0.776**
maven0	10	**0.998**	0.997	0.995	0.942	**0.998**	**0.998**	**0.998**	0.997	**0.998**	0.994	0.989
maven0	20	0.997	0.995	0.994	0.925	0.997	0.997	0.997	0.996	0.997	**0.998**	0.984
maven0	30	0.997	0.995	0.993	0.911	**0.998**	0.997	0.997	0.995	**0.998**	**0.998**	0.982
maven1	10	0.996	0.993	0.991	**0.967**	0.996	0.996	0.996	0.995	0.996	0.995	0.986
maven1	20	**0.996**	0.992	0.990	0.961	**0.996**	**0.996**	0.995	0.994	0.994	0.995	0.984
maven1	30	0.997	0.992	0.989	0.956	**0.998**	0.997	0.996	0.995	0.992	0.992	0.982
maven2	10	**0.998**	**0.998**	**0.998**	0.993	0.997	**0.998**	**0.998**	**0.998**	**0.998**	**0.998**	0.997
maven2	20	**0.997**	**0.997**	**0.997**	0 .991	**0.997**	**0.997**	**0.997**	**0.997**	**0.997**	**0.997**	0.996
maven2	30	0.997	0.997	0.997	0.990	0.997	0.997	**0.998**	0.997	0.997	0.997	0.996
Average		0.897	0.903	0.905	0.898	0.896	0.897	0.901	0.904	0.909	**0.912**	0.908

The highest value is bolded

Table 3 Test set NDCG of community node ranks for ad-hoc filters and those obtained through runtime tuning on the same measure

		Ad-hoc								Autotune		
Com.	Examples (%)	ppr0.5	ppr0.85	ppr0.9	ppr0.99	hk2	hk3	hk5	hk7	select	tune	tuneLBFGSB
amazon0	10	0.970	0.976	0.978	0.985	0.970	0.971	0.975	0.978	0.985	**0.987**	0.983
amazon0	20	0.968	0.976	0.979	0.984	0.967	0.970	0.975	0.978	0.984	**0.987**	0.983
amazon0	30	0.965	0.976	0.978	0.983	0.964	0.968	0.974	0.978	0.983	**0.987**	0.982
amazon1	10	0.955	0.962	0.964	**0.972**	0.954	0.956	0.960	0.963	**0.972**	**0.972**	0.969
amazon1	20	0.954	0.963	0.966	0.972	0.953	0.956	0.962	0.965	0.972	**0.976**	0.971
amazon1	30	0.951	0.963	0.966	0.970	0.950	0.955	0.961	0.966	0.970	**0.976**	0.970
amazon2	10	0.912	0.923	0.927	**0.933**	0.910	0.913	0.920	0.925	0.920	0.927	0.934
amazon2	20	0.914	0.928	0.932	0.933	0.913	0.917	0.926	0.931	0.933	**0.941**	0.937
amazon2	30	0.912	0.926	0.929	0.929	0.910	0.915	0.924	0.929	0.929	**0.938**	0.933
citeseer0	10	0.896	0.900	0.902	0.902	0.894	0.895	0.897	0.899	0.902	0.903	**0.904**
citeseer0	20	0.922	0.931	0.933	0.926	0.920	0.923	0.928	0.931	0.926	0.932	**0.934**
citeseer0	30	0.915	0.925	0.926	0.923	0.907	0.918	0.923	0.926	0.923	**0.930**	0.929
citeseer1	10	0.905	0.905	0.905	0.904	0.905	0.905	0.904	0.903	0.904	**0.907**	**0.907**
citeseer1	20	0.901	0.903	0.904	0.895	0.900	0.901	0.903	0.903	0.903	0.897	**0.905**
citeseer1	30	0.892	0.893	0.892	0.877	0.891	0.892	**0.893**	0.892	0.892	0.890	0.889
citeseer2	10	0.869	0.876	0.878	0.879	0.867	0.868	0.872	0.875	0.868	**0.883**	0.881
citeseer2	20	0.887	0.896	0.897	0.899	0.886	0.888	0.890	0.895	0.886	0.898	**0.900**
citeseer2	30	0.894	0.904	0.905	0.903	0.893	0.896	0.902	0.905	0.905	**0.908**	**0.908**
maven0	10	0.803	0.786	0.778	0.723	0.803	0.804	0.796	0.774	0.796	**0.813**	0.745
maven0	20	0.738	0.708	0.693	0.615	0.743	0.741	0.724	0.694	0.743	**0.744**	0.637
maven0	30	0.673	0.637	0.625	0.554	**0.676**	0.674	0.651	0.629	0.674	0.670	0.581
maven1	10	0.812	0.823	0.821	0.781	0.807	0.817	0.828	0.830	0.828	**0.863**	0.814
maven1	20	0.806	0.814	0.812	0.767	0.788	0.800	0.817	0.818	0.818	**0.831**	0.801
maven1	30	0.770	0.774	0.773	0.728	0.767	0.775	0.780	0.773	0.773	**0.825**	0.762
maven2	10	0.904	0.907	0.906	0.845	0.903	0.908	0.911	0.909	0.911	**0.935**	0.889
maven2	20	0.863	0.864	0.861	0.785	0.862	0.867	0.869	0.865	0.869	**0.903**	0.840
maven2	30	0.812	0.812	0.808	0.729	0.811	0.816	0.818	0.813	0.818	**0.861**	0.786
Average		0.880	0.883	0.883	0.863	0.878	0.882	0.885	0.883	0.888	**0.899**	0.877

The highest value is bolded

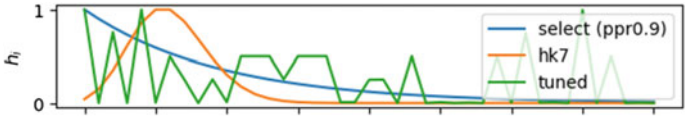


Fig. 2 Parameters h_i of filters with high AUC on citeseer0 with 30% examples

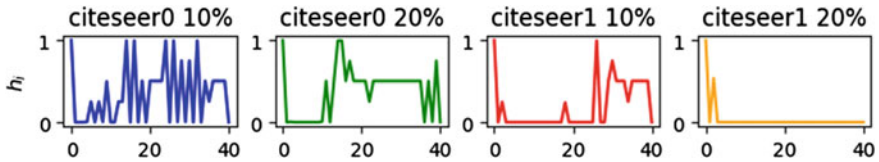


Fig. 3 Parameters tuned on citeseer0 and citeseer1 with 10 and 20% examples

6 Conclusions and Future Work

This work introduces a runtime graph filter selection scheme for community node ranking based on known member nodes. Selection involves either choosing between promising filters or tuning the parameters of a generalized filter form. For the latter, we introduced a novel algorithm that meshes parts of previous alternatives to satisfy both scalability and a wide parameter search breadth needed by graph filters. We verified the efficacy of our approach with experiments across real-world graph communities, where we found that, given enough example community members to satisfy robust evaluation by withholding a few of them, our methodology (especially tuning) yields filters with similar or better AUC and NDCG than alternatives. Thus, we recommend its adoption in practice.

In the future, we are interested in experimenting on more graphs, improving our tuning algorithm, and theoretically probing its optimality. More robust evaluation could also be devised to autotune from fewer known community members.

Acknowledgements This work was partially funded by the European Commission under contract number H2020-951911 AI4Media.

References

1. Abu-El-Haija, S., Kapoor, A., Perozzi, B., Lee, J.: N-gcn: Multi-scale graph convolution for semi-supervised node classification. In: Uncertainty in Artificial Intelligence, pp. 841–851. PMLR (2020)
2. Al-Roomi, A.R.: Unconstrained Single-Objective Benchmark Functions Repository (2015). https://www.al-roomi.org/benchmarks/unconstrained
3. Andersen, R., Chung, F., Lang, K.: Local partitioning for directed graphs using pagerank. Internet Math. **5**(1–2), 3–22 (2008)
4. Bahmani, B., Chowdhury, A., Goel, A.: Fast incremental and personalized pagerank. Proc. VLDB Endow. **4**(3) (2010)

5. Benelallam, A., Harrand, N., Valero, C.S., Baudry, B., Barais, O.: Maven central dependency graph (2018)
6. Bradley, A.P.: The use of the area under the roc curve in the evaluation of machine learning algorithms. Pattern Recogn. **30**(7), 1145–1159 (1997)
7. Byrd, R.H., Lu, P., Nocedal, J., Zhu, C.: A limited memory algorithm for bound constrained optimization. SIAM J. Sci. Comput. **16**(5), 1190–1208 (1995)
8. Chung, F.: The heat kernel as the pagerank of a graph. Proc. Nat. Acad. Sci. **104**(50), 19735–19740 (2007)
9. Finkel, D.E., Kelley, C.: Additive scaling and the direct algorithm. J. Glob. Optim. **36**(4), 597–608 (2006)
10. Galántai, A.: Convergence of the Nelder-Mead method. Numer. Algorithms, 1–30 (2021)
11. Getoor, L.: Link-based classification. In: Advanced Methods for Knowledge Discovery from Complex Data, pp. 189–207. Springer (2005)
12. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864 (2016)
13. Huang, Q., He, H., Singh, A., Lim, S.N., Benson, A.R.: Combining label propagation and simple models out-performs graph neural networks. arXiv:2010.13993 (2020)
14. Järvelin, K., Kekäläinen, J.: Ir evaluation methods for retrieving highly relevant documents. In: ACM SIGIR Forum, vol. 51, pp. 243–250. ACM, New York, NY, USA (2017)
15. Klicpera, J., Bojchevski, A., Günnemann, S.: Predict then propagate: graph neural networks meet personalized pagerank. arXiv:1810.05997 (2018)
16. Kloster, K., Gleich, D.F.: Heat kernel based community detection. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1386–1395 (2014)
17. Koch, P., Golovidov, O., Gardner, S., Wujek, B., Griffin, J., Xu, Y.: Autotune: a derivative-free optimization framework for hyperparameter tuning. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 443–452 (2018)
18. Krasanakis, E., Papadopoulos, S., Kompatsiaris, I., Symeonidis, A.: pygrank: a python package for graph node ranking. arXiv:2110.09274 (2021)
19. Krasanakis, E., Schinas, E., Papadopoulos, S., Kompatsiaris, Y., Symeonidis, A.: Boosted seed oversampling for local community ranking. Inf. Process. Manage. **57**(2), 102053 (2020)
20. Leskovec, J., Adamic, L.A., Huberman, B.A.: The dynamics of viral marketing. ACM Trans. Web (TWEB) **1**(1), 5-es (2007)
21. Leskovec, J., Lang, K.J., Dasgupta, A., Mahoney, M.W.: Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters. Internet Math. **6**(1), 29–123 (2009)
22. Lyu, H.: Convergence of block coordinate descent with diminishing radius for nonconvex optimization. arXiv:2012.03503 (2020)
23. McPherson, M., Smith-Lovin, L., Cook, J.M.: Birds of a feather: homophily in social networks. Ann. Rev. Sociol. **27**(1), 415–444 (2001)
24. Ortega, A., Frossard, P., Kovačević, J., Moura, J.M., Vandergheynst, P.: Graph signal processing: overview, challenges, and applications. Proc. IEEE **106**(5), 808–828 (2018)
25. Page, L., Brin, S., Motwani, R., Winograd, T.: The Pagerank Citation Ranking: Bringing Order to the Web. Tech. rep. Stanford InfoLab (1999)
26. Papadopoulos, S., Kompatsiaris, Y., Vakali, A., Spyridonos, P.: Community detection in social media. Data Min. Knowl. Disc. **24**(3), 515–554 (2012)
27. Shuman, D.I., Narang, S.K., Frossard, P., Ortega, A., Vandergheynst, P.: The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process. Mag. **30**(3), 83–98 (2013)
28. Stanković, L., Daković, M., Sejdić, E.: Introduction to graph signal processing. In: Vertex-Frequency Analysis of Graph Signals, pp. 3–108. Springer (2019)
29. Tong, H., Faloutsos, C., Pan, J.Y.: Fast random walk with restart and its applications. In: Sixth International Conference on Data Mining (ICDM'06), pp. 613–622. IEEE (2006)

30. Tooley, R.: Auto-tuning spark with Bayesian optimisation (2021)
31. Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., SciPy 1.0 Contributors: SciPy 1.0: fundamental algorithms for scientific computing in python. Nat. Methods **17**, 261–272 (2020)
32. Whang, J.J., Gleich, D.F., Dhillon, I.S.: Overlapping community detection using seed set expansion. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, pp. 2099–2108 (2013)
33. Whang, J.J., Gleich, D.F., Dhillon, I.S.: Overlapping community detection using neighborhood-inflated seed expansion. IEEE Trans. Knowl. Data Eng. **28**(5), 1272–1284 (2016)
34. Wu, F., Huberman, B.A.: Finding communities in linear time: a physics approach. Euro. Phys. J. B **38**(2), 331–338 (2004)
35. Yang, J., Leskovec, J.: Defining and evaluating network communities based on ground-truth. Knowl. Inf. Syst. **42**(1), 181–213 (2015)
36. Zareie, A., Sheikhahmadi, A.: A hierarchical approach for influential node ranking in complex social networks. Expert Syst. Appl. **93**, 200–211 (2018)
37. Zhang, T., Wu, B.: A method for local community detection by finding core nodes. In: 2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 1171–1176. IEEE (2012)

Dynamic Local Community Detection with Anchors

Konstantinos Christopoulos, Georgia Baltsou, and Konstantinos Tsichlas

Abstract Community detection is a challenging research problem, especially in dynamic networks, since in this case communities cannot remain stable as they evolve. In evolving networks new communities may emerge and existing communities may disappear, grow or shrink. There are many cases where someone is more interested in the evolution of a particular community, to which an important node belongs, rather than in the global partitioning of a dynamic network. However, due to the drifting problem where one community can evolve into a completely different one, it is difficult to track the evolution of communities. Our aim is to identify the community that contains a node of particular importance, called anchor, and its evolution over time. The framework we propose circumvents the identity problem by allowing the anchor to define the core of the relevant community partially or fully. Preliminary experiments with synthetic datasets demonstrate the positive aspects of the proposed framework in identifying communities with high accuracy.

Keywords Local community detection · Networks · Dynamic · Anchor

1 Introduction

Networks are used to represent entities and their relations for systems of almost any domain like biology, society, transportation etc. In such systems, there is a huge amount of data that is constantly generated. Community detection constitutes an important task of network analysis. It's aim is to uncover groups of densely

K. Christopoulos (✉) · K. Tsichlas
Department of Computer Engineering and Informatics, University of Patras, Patras, Greece
e-mail: kchristopou@upnet.gr

K. Tsichlas
e-mail: ktsichlas@ceid.upatras.gr

G. Baltsou
School of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece
e-mail: georgipm@auth.gr

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_16

connected entities called communities. Most of the existing literature is concerned with the global community detection problem, i.e. a whole network's division into communities. However, in many cases we are interested only on the communities around a few particular nodes. For this reason, local community detection has lately attracted scientists' interest. Consequently, there are different cases where global approach is preferred over the local and vice versa. Generally, local community detection is more suitable for discovering the communities for nodes of interest on complex networks with low computational cost. Most existing work on community detection problems is on static networks. Static networks do not change their structure (nodes and edges) over time. However, most real-world networks change rapidly, and sometimes relationships are established only instantaneously. Networks that are time evolving are called dynamic or temporal. Moreover, with the rapid growth of the internet and its applications, real-world network datasets are extremely large, making it unreasonable to process them in their entirety as it must be done in the case of global community detection. One way to model such rapidly changing networks is by assuming that updates/actions[1] come in a streaming fashion. This means that in each time instance one update is performed on the network. In this sense, time is defined with respect to these updates and the life span of an edge is defined by the updates between its insertion and its removal from the graph [2, 9]—that is, we use transaction time as our main notion of time.

In the present work, we focus on the detection of local communities of particular nodes in temporally evolving networks by revising the theoretical framework of [1] and providing preliminary experimental results that verify its effectiveness. More specifically, our goal is to uncover the community evolution of a node of particular interest, called *anchor*. The importance of this node for the evolving community is considered external knowledge, that is, knowledge that cannot be inferred from the structure of the network. This node defines the evolving community and functions as an anchor for the community circumventing in this way the identity problem. As an example, one can think of a football team community in a social network. This community evolves since new fans may connect or existing fans may stop supporting the team. However, the core fans (e.g., ultras) of the team are more stable and in some sense behave as an anchor for this community.

1.1 Contributions

Our present work focuses on the identification of the community of a specific node called anchor, which is assumed to be of particular importance to this community based on external knowledge in temporal networks. To achieve this, we propose a multi-step framework that firstly applies a static algorithm to discover the initial

[1] We use the terms updates and actions interchangeably to refer to a small change in the network, such as edge insertion or/and deletion.

anchor's community and then for each update "near" the anchor its community is updated. We experimentally show the promise of the suggested framework when compared to other methods in synthetic datasets. Our contribution is twofold:

- From a **modeling perspective**, our contribution lies in the introduction of the notion of the anchor node in the local community detection problem in time evolving networks.
- From an **algorithmic perspective**, a general multi-step framework is suggested to be used in order to uncover stable communities of an important node in time evolving networks.

The remaining sections are organized as follows. In Sect. 2, we review the literature on local community detection in dynamic networks. The proposed framework is described in Sect. 3. In Sect. 4, we present experimental results illustrating our algorithmic framework. Finally, we discuss future expansions of the suggested framework and conclude in Sect. 5.

2 Related Work

Local community detection, which is also known as the seed set expansion problem, has attracted the attention of researchers as it is very common to process only a small part of the network, either because of its large size or because it is dynamic or someone might be interested in focusing on a specific part of the network. Consequently, there have been many different approaches proposed. However, the literature in dynamic networks is much smaller when compared to the case of static networks. In the following, we discuss local community detection algorithms that are closely related to our work, i.e. in dynamic networks processed in a streaming fashion

In [14] the authors adopt the static L-metric approach [3] in order to find dynamic communities in an incremental way. L-metric is a measure based on the assumption that a community has fewer connections to nodes outside of this community. At each snapshot communities are uncovered using information from previous snapshots and at the end communities found in different snapshots are matched based on their similarity (L-metric). Experiments showed that the method resulted in meaningful communities. In addition, a dynamic seed set expansion method is proposed in [16, 17] where the authors suggest updating the fitness score of each snapshot incrementally. In order to keep a community centered around the seed, their method ensures that the order of fitness scores remains monotonically increasing by tracking the order of nodes added. Experiments showed that the suggested method is quite fast and the performance is better when low-latency updates are required. Furthermore, in [4] a method called PHASR to find the temporal community with the lowest conductance is proposed. This work aims to find communities with stable membership over time. Experiments showed that the suggested method has low runtime and achieves to find high quality communities. Moreover, the authors of [7] use a metric called local fitness to firstly find the starting nodes of a community and run

a static algorithm to define the communities in the first snapshot. In the following snapshots, they use a node contribution metric to incrementally reveal communities. Their experiments showed that the proposed method uncovered communities with high accuracy. Finally, to the best of our knowledge, two methods, [10, 15], have been proposed for local community detection in graph streams. In [10], the CoEuS algorithm is suggested. The constraint of this method lies in the fact that only a single access to the stream is possible and the working memory is limited. Experiments on networks showed that the algorithm is able to discover local communities with high accuracy. More recently, the algorithm called SCDAC that is suggested in [15], seeks an optimal community on the subgraph intercepted by the streaming model. Experiments showed that SCDAC is more effective and efficient than CoEuS in real networks.

3 Dynamic Local Community Detection with Anchors

3.1 Preliminaries and Problem Formulation

Let $G = (V, E_t)$ be a dynamic network which is composed of a node set V and a set of time-stamped edges E_t. E_t represents interactions among the nodes at time t, where $t \in \mathbb{N}$, generated by an interaction streaming source S. The interaction streaming source S may produce new interactions between nodes which can be either already part of the network or new ones. In particular, it forms a sequence of actions in which interactions flow in streams over time. In this paper, we assume that an action may be an edge insertion or an edge deletion. As a result, the corresponding network's communities also change as the network evolves. Dynamic community detection is the process by which we can observe the evolution of the network's communities.

Given a node A called anchor, the network G and an interaction generator S, our aim is to discover the community C which includes A. We assume that the anchor is of particular importance for the community (external knowledge) it belongs and thus, it operates as a reference point for this community, i.e., anchor defines in a sense the community it belongs to.

In order to minimize the avalanche effect[2] [13], we suggest to limit the community updating only to an influence range around the anchor. The *influence range*, R, defines the radius of the ball centered around the anchor in each time instance of the evolving network. In this ball, all nodes with maximum length of their shortest path to the anchor $\leq R$ are contained. For example, an influence range equal to 1 means that we should update the community structure considering both the anchor as well as its adjacent nodes. Generally, a high influence range value would increase the process demands, since a larger network area would be examined.

[2] The avalanche effect corresponds to the phenomenon where communities can experience substantial drifts compared to what a static algorithm computes in each time instance.

Besides, with a view to discover the most stable anchor's community, we can use a node rewarding method. That is, for each update, we suggest to reward the edges in the anchor's influence range by a weight increase. In our setting, we use three different rewarding methods. Assuming that R is the influence range and d the distance between the anchor and a node, we define the rewarding methods as follows:

1. *Dynamic reward* 1: $w = 2R - 2(d - 1)$. For instance, if $R = 3$, the edges of the anchor to its adjacent nodes get a weight of $w = 6$, as $d = 1$. Consequently, all the edges of the anchor's adjacent nodes to their adjacent nodes, where the distance from anchor is $d = 2$, get a weight of $w = 4$, and so on until $d = R$.
2. *Dynamic reward* 2: $w = R^{R/d}$. Similar to the previews one, if $R = 3$, the edges of the anchor to its adjacent nodes get a weight of $w = 27$, as $d = 1$. Consequently, all the edges of the anchor's adjacent nodes to their adjacent nodes, where the distance from anchor is $d = 2$, get a weight of $w = \sqrt{27}$, and so on until $d = R$.
3. *Dynamic reward* 3: this is a rewarding system that takes into account in a very simple manner the history of an edge. When a new edge arrives that its nodes have minimum distance d from the anchor, we initially set its weight to $w = R - d + 1$. Then, if the edge persists after a batch of y actions we assign an extra unit reward to it. y is a user-determined parameter. If an edge is reinserted and lies in the influence range of the anchor then the reward that receives is estimated by the ratio of edge appearances to total number of actions (edge insertion or deletion) in the influence range.

The quality of a community C can be measured by different quality metrics. We use three such metrics in order to objectively evaluate the performance of our suggested framework. The first quality metric is f_{monc}, which is defined as the ratio of the sum of the degrees of internal nodes to other nodes within the community divided by the total sum of the degrees of nodes in C [8]:

$$f(C)_{monc} = \frac{2k_{in}^{C} + 1}{(2k_{in}^{C} + k_{out}^{C})^{\alpha}},$$

where k_{in}^{C} and k_{out}^{C} are the total internal and external degrees of the nodes of community C, and α is a positive real-valued parameter, controlling the size of the communities. The second quality metric we choose to use is LWP which is the ratio of interior edges to edges leaving community C defined in [11] as:

$$LWP(C) = \frac{k_{in}^{C}}{k_{out}^{C}}.$$

The third community quality metric that we use is *conductance* as defined in [6]. For community C and its complement $\overline{C} = V \backslash C$ conductance is defined as:

$$cond(C) = \frac{c(C)}{\min(l(C, V), (\overline{C}, V))},$$

Table 1 Community evolution depending on the fitness scores order

Sequence	0	1	2	...	n
Nodes	u_0	u_1	u_2	...	u_n
Interior edges	$K_{0,in}$	$K_{1,in}$	$K_{2,in}$	...	$K_{n,in}$
External edges	$K_{0,out}$	$K_{1,out}$	$K_{2,out}$	...	$K_{n,out}$
Fitness score	f_0	f_1	f_2	...	f_n

where $c(C)$ is equal to $cut(C)$, which is defined as the number of edges between nodes in C and nodes in its complement $\overline{C}$. $l(A, B)$ is the number of edges between nodes in A and nodes in B

3.2 Proposed Framework

We assume network G with node set V and edge set E where edges are unweighted. Our method is divided into five steps. The first two are the initialization and the rest the streaming process.

Initialization: The first step of the suggested framework, is to apply weights on the edges according to the anchors influence range. More precisely, we apply a weighting scheme that rewards the edges being closer to the anchor. The depth till which edges are rewarded starting from the anchor is predefined by the chosen influence range R.

The second step of our proposed framework is the application of a greedy static algorithm [8] on the initial state of network G, at timestamp defined as $t = 0$. At this timestamp, the community C contains only the anchor (u_0), and then new nodes are iteratively added. A node is added to C (e.g. u_1) only if the fitness score (e.g. $f(C)_{monc}$) is increased ($f_0 < f_1$). The static algorithm terminates when the fitness score can not be increased anymore. At the end of the second step, a community evolution sequence[3] is created and interior($K_{n,in}$)/external($K_{n,out}$) community edges are also recorded (i.e. the sequence in which each node entered the community, see Table 1).

Streaming process: In the third step of the process, a stream of network updates i is applied. These updates can be either edge insertions or deletions. If i occurs in the anchors' influence range: (1) influence range has to be recomputed and (2) edge weights have to be updated considering a rewarding method. Consequently, if i occurs in the anchor's community C or the updated weights affects it, then the C quality measure has to be updated. More precisely, the measure has to be recalculated and if after this recalculation the fitness scores are not anymore in an increasing order, then the node that disrupts this order must be removed and interior/border edges must be

[3] If the quality metric is Conductance then the fitness score must be decreased and so, the fitness scores should be in a decreasing order.

Fig. 1 The proposed approach for dynamic local community detection with anchors

modified as well. Then, the same procedure is applied for community nodes that are neighbors of the removed node, and repeat as long as there are neighbors that are not affected by these changes. Even if i does not occur in the anchors' influence range, we should still check the anchors' community because is may extended beyond the influence range. In the fourth step, after fixed-size batches of actions: (1) we check if the sequence of fitness scores is in increasing order and if not, we remove all nodes from the sequence from the leftmost violation up to end and (2) we apply the static algorithm on the updated community of the anchor. The number of actions in each batch is calculated as the ratio of the total number of actions to a user-defined constant value x. That is, the static algorithm will run no more than x times. We need to note here that the more times we choose to run the static algorithm, the more accurate is the outcome of the process. However, the computational cost is higher. Thus, after experimental evaluation, we conclude that the value of the constant should be equal to 20. Figure 1 depicts the steps of the proposed framework.

4 Experiment Design

4.1 Datasets

The synthetic datasets we use in our experiments are generated by RDyn [12], an approach capable of generating dynamic networks that respect well-known real-world network properties along with time-dependent ground truth communities with adjustable quality, i.e., allowing both merging and splitting communities. The generator contains two significant, user-defined, parameters. The first is the number of nodes of the produced dynamic network and the second is the number of iterations. Each iteration consist of a batch of actions (edge insertion/deletion) and the number of these actions are not necessary equal in every iteration. The first iteration of each synthetic dataset is utilized for the purpose of the creation of the initial graph. So that, in each dataset the actions of first iteration are not taken into consideration to the total amount of actions. In our experiments, we use three different datasets produced by the RDyn generator. The basic characteristics of these datasets are described in Table 2.

Table 2 Synthetic datasets with number of nodes, iterations, initial/final edges and actions

Dataset	Nodes	Iterations	Initial edges	Final edges	Actions
$SD1$	100	100	99	478	6268
$SD2$	500	1000	495	1648	40,939
$SD3$	5000	1000	4917	25,590	246,191

4.2 Evaluation Metrics

To evaluate our proposed framework, we compare the results of our community detection with the ground truth communities produced by the synthetic dataset generator. However, an eligible (to some extent) argument against using the ground truth communities of the synthetic generator is that the discovered community is affected by the anchor. To this end, on the one hand we tried to setup the generator so that communities are not so intertwined while on the other hand we are more interested in comparing the methods between each other rather than looking at values of the metrics w.r.t. the ground truth. The evaluation metrics that are suitable for our purposes are precision, recall, and the F1 score. Precision is the ratio of elements found correctly to the total number of elements found. Recall is the proportion of relevant elements that were successfully retrieved. The F1 score is the harmonic mean of precision and recall [5]. The harmonic mean is used instead of the simple average because in this way the extreme values are penalised.

4.3 Experiment Results

In our experiments we use three different quality metrics, f_{monc}, $Conductance$ and LWP. The node we use as an anchor for each experiment is determined based on its degree centrality. That is, in the first dataset we use a node with low degree as an anchor, in the second we choose two nodes with medium degree, and in the third we use a node with high degree. In the experiments with the first dataset, we use the three quality metrics, while for the others we use the f_{monc} since it provides the best results. We choose the user-defined parameter for f_{monc} to be $a = 1$ and the influence range equal to 2.

Regarding the first synthetic graph, Figs. 2, 3 and 4 shows the results of the three quality metrics. The values on the x-axis represents the number of actions. The graph generator provides the graph partition after one or more iterations, where in each iteration the number of actions are not equal. As a consequence, each interval on the x-axis consist of the same number of iterations but different number of actions. The vertical lines on the x-axis shows the events (merge or split) that occurred in the ground truth communities, right after an action. One of these communities contains our anchor.

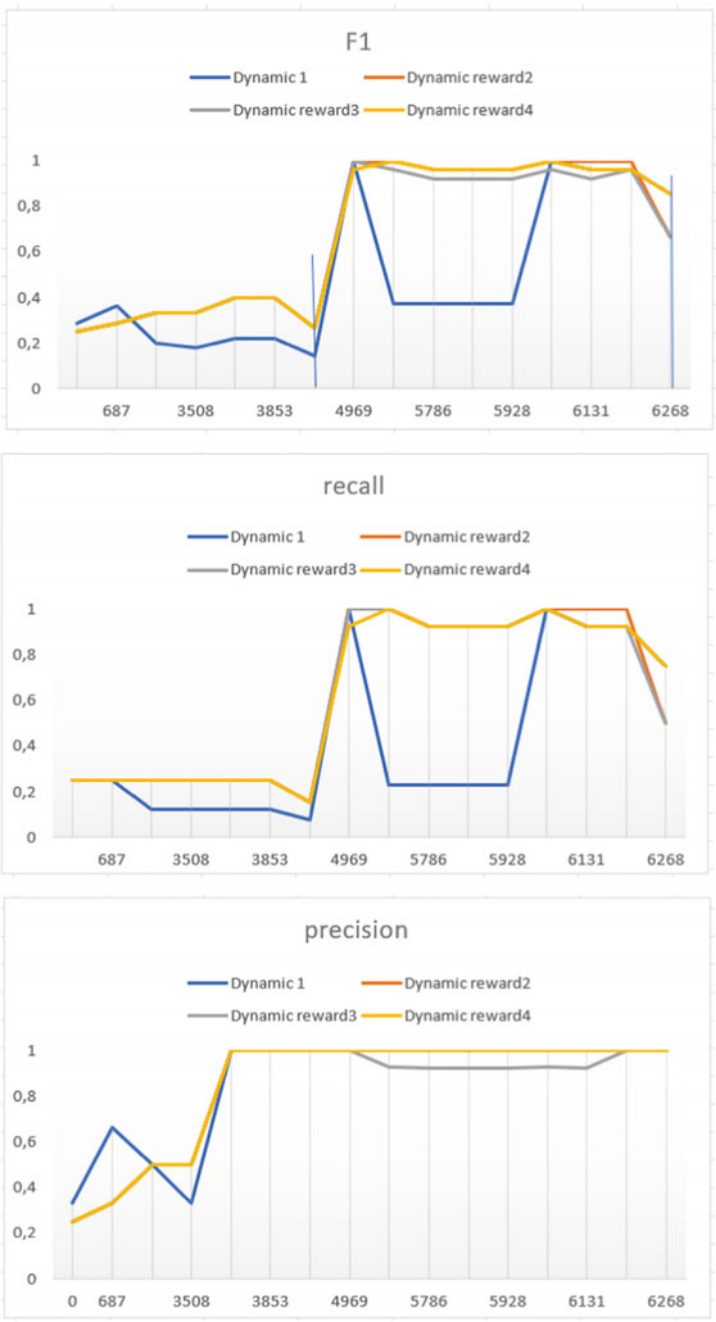


Fig. 2 Results using the synthetic dataset with 100 nodes and conductance as a quality metric

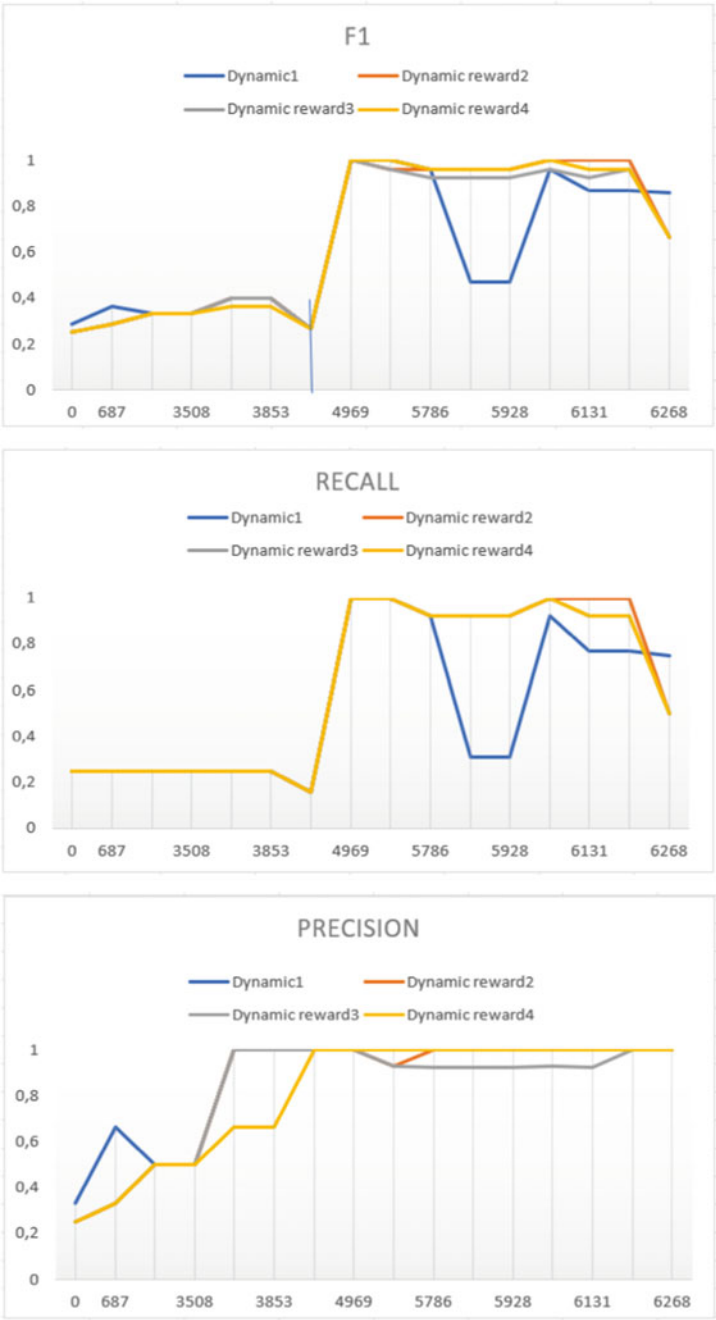


Fig. 3 Results using the synthetic dataset with 100 nodes and LWP as a quality metric

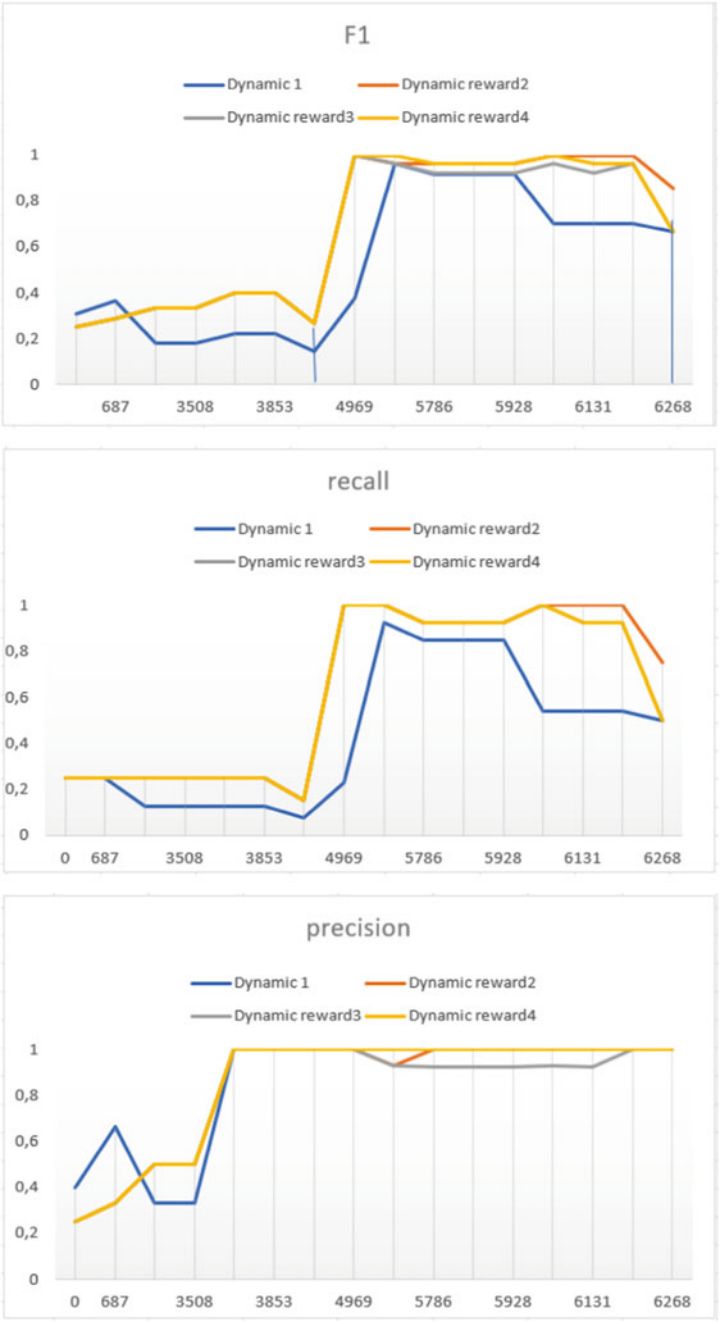


Fig. 4 Results using the synthetic dataset with 100 nodes and f_{monc} as a quality metric

Analyzing the experimental results, we find that all three dynamic methods with rewards (Dynamic Reward 1, Dynamic Reward 2, and Dynamic Reward 3) outperform dynamic method without rewards [17] (Dynamic 1). More precisely, the critical point of community evolution is the time when the first event (merge) occurs. Before this point, recall and precision were very low. There is an explosion in the recall metric and as a result, an improvement in the F1 score. In addition, between actions 4969 and 5786, the static algorithm is activated and as a consequence, the nodes affiliation in anchor's community is nearly the same with those in ground truth community. This is true for all quality metrics and for all dynamic methods except for Dynamic 1. For the latter method, we observe large variations in recall values, which is reflected in the F1 score. In more detail, looking carefully in dataset we can observe the following: (1) just before the action 5700, few edges (not related directly to anchor) that belong to the influence range of our community are inserted and at the same time the static algorithm is activated. As a consequence, our methods using conductance as a quality metric, take advantage of rewards and remain stable with high recall values. On the contrary, the same is not true for Dynamic 1. (2) In Fig. 3 using LWP as a quality metric, we notice the same outcome. Here, our reference point is the action 5896. Again, before this action we observe few edge insertions in the influence range with similar results as before. (3) On the other hand, for both quality metrics the recall values of Dynamic1 method rises between actions 5928 and 6091. Here, we notice an edge insertion between our anchor an its adjacent node. This action helps the method without rewards to reach the high recall values of other methods. Lastly, f_{monc} provides more stable and better results but still our methods outperform Dynamic1.

Next, we consider the second synthetic graph containing 500 nodes. Figures 5 and 6 shows the results of the three evaluation metrics using the f_{monc}. Here we run the experiment twice with two different anchors. In Fig. 5 we see 10 events. In the first actions, 6 events take place, which affect the recall and precision of each method. More specifically, all dynamic methods have extreme ups and downs, and Dynamic1 being the most affected. After these events, it is clear to see that our methods outperform Dynamic 1. The small time interval in which Dynamic 1 outperform the other methods is due to an edge deletion just before action 37,154, which belong to the influence range. In Fig. 6, a different anchor was chosen to show the dominance of the rewarding methods. In particular, we observe six events that cause negative fluctuations, mainly for Dynamic 1 method. After the fourth event, Dynamic1 is overlapped by Dynamic reward 3 and at the end by Dynamic reward 2.

Finally, Fig. 7 shows the results of the generated graph with 5000 nodes. Here, a split event take place at the beginning, which affects the performance of all methods. Nevertheless, not long after the third event and in combination with the activation of the static algorithm, Dynamic reward1 and reward 2 have a significant improvement in recall and precision values, which has a consistently positive impact on the F1 score. On the other hand, Dynamic 1 and Dynamic reward 3 are almost congruent for a long time, but after action 218,895 and until the end, the latter method performs much better. For the other two rewarding methods, it is obvious that recall and precision reflect the positive result of F1-Score.

Fig. 5 Results using the synthetic dataset with 500 nodes and f_{monc} as a quality metric

Fig. 6 Results using the synthetic dataset with 500 nodes and f_{monc} as a quality metric

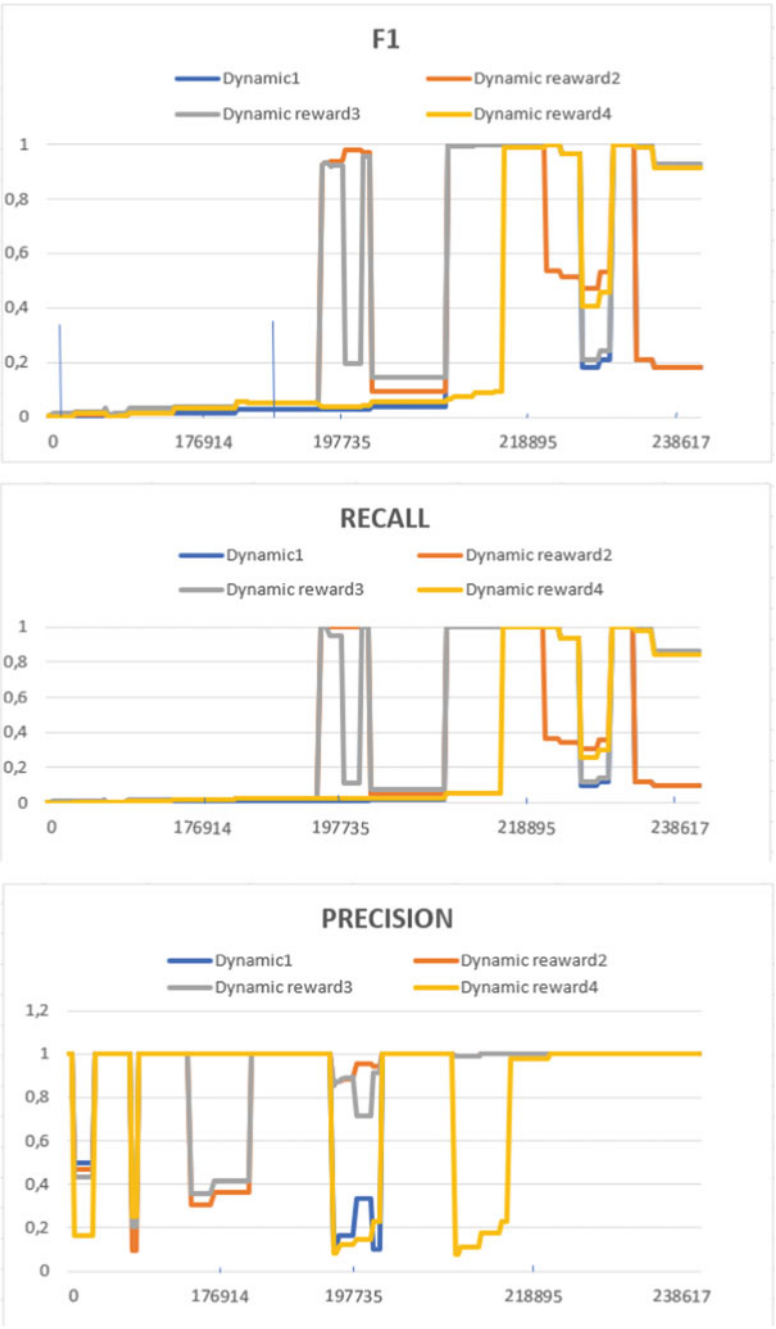


Fig. 7 Results using the synthetic dataset with 5000 nodes and f_{monc} as a quality metric

The fluctuations we observe could be explained in two ways. First of all, many actions in the influence range of anchor are occurred. For instance, after a missing of an edge with extra reward the community coherence breaks, which reflects on quality metrics. Secondly, as mentioned previously, due to the fact that the static algorithm is used after a fixed batch of actions and not whenever the graph generator provides the ground truth communities, this have an effect on our outcome.

5 Conclusions

Dynamic local community detection constitutes a research field that has drawn scientists' interest the last years. In the present work, we focus on the discovery of local communities that contain important nodes termed anchors. Our aim is not only to identify such communities but also track their evolution over time as new edge insertions and/or deletions occur in a network. To achieve this, we suggest a multi-step framework that firstly applies a static algorithm to discover the initial anchor's community and then for each incoming edge change in the influence range of the anchor, we update the anchor's community. Influence range is used to minimize the avalanche effect. With a view to discover the most stable anchor's community, we suggest using a node rewarding method. That is, for each update, we suggest to reward the stable edges in the anchor's influence range by a weight increase. A preliminary experimental evaluation of the proposed framework is conducted using three different synthetic datasets. We also used three proposed rewarding methods and compared the results with the case where no rewarding method is used. Our findings indicate that all three dynamic methods with rewards (Dynamic reward1, Dynamic reward2 and Dynamic reward3) outperform the dynamic method without rewards in terms of recall, precision and F1 score.

This work contains preliminary results and we intend to extend these results along the following axis: 1. Extended experimental evaluation of more rewarding schemes that take into account the history of edges. 2. Experimentation on real temporal networks. 3. Elaborate tuning of the various parameter of the rewarding schemes and 4. Efficiency comparison between different rewarding schemes since the more complicated a scheme is the more time it needs per action.

Acknowledgements "Georgia Baltsou is co-financed by Greece and the European Union (European Social Fund-ESF)" through the Operational Programme "Human Resources Development, Education and Lifelong Learning" in the context of the project "Strengthening Human Resources Research Potential via Doctorate Research—2nd Cycle" (MIS-5000432), implemented by the State Scholarships Foundation (IKY)." "This research was supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the "2nd Call for H.F.R.I. Research Projects to support Faculty Members & Researchers" (Project Number: 3480)."

References

1. Baltsou, G., Tsichlas, K.: Dynamic community detection with anchors (2022)
2. Casteigts, A., Flocchini, P., Quattrociocchi, W., Santoro, N.: Time-varying graphs and dynamic networks. In: International Conference on Ad-Hoc Networks and Wireless, pp. 346–359. Springer (2011)
3. Chen, J., Zaiane, O.S., Goebel, R.: Detecting communities in large networks by iterative local expansion. In: 2009 International Conference on Computational Aspects of Social Networks, pp. 105–112. IEEE (2009)
4. DiTursi D.J., Ghosh, G., Bogdanov, P.: Local community detection in dynamic networks. In: 2017 IEEE International Conference on Data Mining (ICDM), pp. 847–852. IEEE (2017)
5. F1 score lemma. F1 score lemma—Wikipedia, the free encyclopedia (2020)
6. Gao, Y., Zhang, H., Zhang, Y.: Overlapping community detection based on conductance optimization in large-scale networks. Phys. A Stat. Mech. Appl. **522**, 69–79 (2019)
7. Guo, K., He, L., Huang, J., Chen, Y., Lin, B.: A local dynamic community detection algorithm based on node contribution. In: CCF Conference on Computer Supported Cooperative Work and Social Computing, pp. 363–376. Springer (2019)
8. Havemann, F., Heinz, M., Struck, A., Gläser, J.: Identification of overlapping communities and their hierarchy by locally calculating community-changing resolution levels. J. Stat. Mech. Theory Exp. **2011**(01), P01023 (2011)
9. Kostakos, V.: Temporal graphs. Phys. A Stat. Mech. Appl. **388**(6), 1007–1023 (2009)
10. Liakos, P., Papakonstantinopoulou, K., Ntoulas, A., Delis, A.: Rapid detection of local communities in graph streams. IEEE Trans. Knowl. Data Eng. (2020)
11. Luo, F., Wang, J.Z., Promislow, E.: Exploring local community structures in large networks. Web Intel. Agent Syst. Int. J. **6**(4), 387–400 (2008)
12. Rossetti, G.: Rdyn: graph benchmark handling community dynamics. J. Complex Netw. **5**(6), 893–912 (2017)
13. Rossetti, G., Cazabet, R.: Community discovery in dynamic networks: a survey. ACM Comput. Surv. (CSUR) **51**(2), 1–37 (2018)
14. Takaffoli, R.R., Zaïane, O.R.: Incremental local community identification in dynamic social networks. In: 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013), pp. 90–94. IEEE (2013)
15. Yang, Y., Wang, M., Bindel, D., He, K.: Streaming local community detection through approximate conductance, pp. arXiv–2110 (2021)
16. Zakrzewska, A., Bader, D.A.: A dynamic algorithm for local community detection in graphs. In: 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 559–564. IEEE (2015)
17. Zakrzewska, A., Bader, D.A.: Tracking local communities in streaming graphs with a dynamic algorithm. Social Netw. Anal. Mining **6**(1), 1–16 (2016)

Community Detection Supported by Node Embeddings (Searching for a Suitable Method)

Bartosz Pankratz, Bogumił Kamiński, and Paweł Prałat

Abstract Most popular algorithms for community detection in graphs have one serious drawback, namely, they are heuristic-based and in many cases are unable to find a near-optimal solution. Moreover, their results tend to exhibit significant volatility. These issues might be solved by a proper initialization of such algorithms with some carefully chosen partition of nodes. In this paper, we investigate the impact of such initialization applied to the two most commonly used community detection algorithms: **Louvain** and **Leiden**. We use a partition obtained by embedding the nodes of the graph into some high dimensional space of real numbers and then running a clustering algorithm on this latent representation. We show that this procedure significantly improves the results. Proper embedding filters unnecessary information while retaining the proximity of nodes belonging to the same community. As a result, clustering algorithms ran on these embeddings merge nodes only when they are similar with a high degree of certainty, resulting in a stable and effective initial partition.

Keywords Machine learning · Community detection · Complex networks · Network embedding methods

B. Pankratz (✉) · P. Prałat
Department of Mathematics, Toronto Metropolitan University, Toronto, ON, Canada
e-mail: bartosz.pankratz@ryerson.ca

P. Prałat
e-mail: pralat@ryerson.ca

B. Pankratz · B. Kamiński
Decision Analysis and Support Unit, SGH Warsaw School of Economics, Warsaw, Poland
e-mail: bpankra@sgh.waw.pl; bkamins@sgh.waw.pl

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_17

1 Introduction

The main trait of most empirical complex networks is the fact that they tend to display a modular organization where one can easily separate sets of nodes (*subgraphs*) with considerably larger density of edges between nodes in such sets than between two different sets. This property is widely referred to as a community structure [8]. Finding such partitions is interesting not only from a theoretical perspective. Indeed, often communities that are extracted, or nodes inside them, exhibit different properties than the entire graph, so identifying them might give a meaningful insight into the data. However, in most cases such underlying structure is unknown beforehand, thus we must use an unsupervised algorithm that is able to detect it. There are many existing solutions; the most common ones are built around a heuristic optimization of some carefully chosen score function.

Communities are somewhat elusive; without the full knowledge about the graph generating process (which is obviously the case for most real-world networks) it is not clear what score function or measure should be used to assess them and, consequently, what algorithm should be used to detect them, especially since no algorithm can uniquely solve community detection task [25]. This problem is widely discussed, see, for example, [19, 21, 34], and plenty of different score functions were proposed up to them. The modularity function [23] is possibly the most often used one.

Modularity measures the difference between the number of the edges within groups induced by a given partition $\mathcal{A}$ and the expected number of such edges given by an appropriately selected null-model, usually **Chung-Lu** random graph model [1]. For a graph $G = (V, E)$ and a given partition $\mathcal{A} = \{A_1, A_2, \ldots, A_\ell\}$, the modularity function is defined as follows:

$$q_G(\mathcal{A}) = \frac{1}{|E|} \sum_{A_i \in \mathcal{A}} \Big(e_G(A_i) - \mathbb{E}_{G' \sim \mathcal{G}(d)}[e_{G'}(A_i)] \Big), \tag{1}$$

where $|E|$ is the number of edges in G, $e_G(A_i) = |\{v_j v_k \in E : v_j, v_k \in A_i\}|$ is the number of edges in the subgraph of G induced by set A_i, and $\mathbb{E}_{G' \sim \mathcal{G}(d)}[e_{G'}(A_i)]$ is the corresponding expectation in the null-model.

However, optimizing modularity function is a NP-hard problem [5]; thus, basically all proposed solutions are heuristic in nature. One of the most popular, fastest, and best performing [18] ones is the **Louvain** algorithm [4]. Its core idea is simple yet effective, it is a two-step technique: it first moves each node to the community that provides the largest increase of the score function, ensuring that the score will be locally optimal; during the second step, it aggregates the communities into supernodes. Then, both phases are repeated until there is no improvement of the score function. By default, the **Louvain** algorithm starts from a singleton partition in which each node belongs to its own community but it is possible to initialize the algorithm with a preexisting partitioning.

Despite the fact that **Louvain** is a great algorithm, it has some serious and known drawbacks. First, the obtained results are heavily stochastic, that is, each run of the algorithm on the same network may lead to the vastly different partitions. Moreover, it may create a weakly connected or even internally disconnected communities [32]. These problems are caused by two factors, both inherent to the nature of the algorithm. It is a greedy algorithm; sometimes, especially on early iterations, nodes might be added to communities that they should not belong to because the algorithm finds the local best solution without considering the broader structure of the graph. Then, during the second phase, it merges the community into a supernode which makes it impossible to backtrack and fix these bad early connections.

This shortcoming might be addressed in two manners; either by allowing the algorithm to backtrack and refine the created communities in each step, which was proposed by **Leiden** algorithm [32] or by ensuring that the initial partitioning is stable and contains the nodes that certainly belong to the same community, as in **ECG** (Ensemble Clustering algorithm for Graphs) algorithm [27].

The latter idea is the center of this work, namely, we want to propose a method of community detection based on the modularity optimization with a spectral clustering initialization step. The procedure starts with an embedding of the nodes of the graph in the high dimensional space of real numbers, then the clustering algorithm is run on the obtained representation. The algorithm is fine-tuned to obtain many small clusters where only nodes that are very close in the latent space are merged together. As a result, we obtain a stable partition which is finally used to initialize the **Louvain** algorithm. In the same manner, such initial partition might also be used to improve other greedy optimization algorithms such as the **Leiden** one. In the experiments presented in this paper, we will test both of them but when describing the reasoning behind the proposed method we will use the **Louvain** algorithm as an example.

Our motivation is simple; we believe that carefully selected embeddings preserve the proximity of nodes belonging to the same community and clearly separate them from the other ones, reducing the chance of misguided connections at the early stages of the algorithm. Having said that, relying only on the embedded representation is causing problems on its own; by their nature (typically local), embeddings preserve some properties of the nodes but filter some other ones, resulting in the inherent information loss that might induce a significant bias if we decide to run the clustering algorithm only on the embedded data and use it as the final partition. Therefore, the most promising approach that we propose in this paper is to combine both methods.

The goal of this paper is to test this premise. In order to do this, we perform an experiment aimed to answer the following four questions:

(1) *How the proposed method performs compared to the other extensions of the* **Louvain** algorithm (**Leiden** and **ECG**)?
(2) *How stable is the proposed method? How volatile are the results compared to the* **Louvain** *algorithm?*
(3) *Is this method able to improve the* **Leiden** *algorithm?*

(4) *Which embedding methods and clustering algorithms give the best results? What is the relation between the graph's properties and the way how it is embedded into the latent space?*

The rest of the paper is organized as follows. In Sect. 2 we further describe the proposed method and motivate it. Sections 3 and 4 introduce an experiment designed to test the hypothesis and, respectively, present obtained results. Finally, Sect. 5 provides some concluding remarks.

2 Method Description

Let $G = (V, E)$ be a graph on the set of n nodes $V = \{v_1, v_2, \ldots, v_n\}$ and the set of m edges $E = \{e_1, e_2, \ldots, e_m\}$. In order to find the partition $\mathcal{A} = \{A_1, A_2, \ldots, A_\ell\}$ of V that tries to maximize the modularity function $q_G(\mathcal{A})$, we perform the following three steps:

Step 1: Find the embedding function $\mathcal{E}\colon V \to \mathbb{R}^s$ which embeds each node of graph G into a s-dimensional latent vector $\mathcal{E}(v) = \{z_1, z_2, \ldots, z_s\}$, where $s \ll n$.
Step 2: Run the clustering algorithm on the obtained latent representation $\mathcal{E}$ to get the partition $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$. The goal is to use $\mathcal{C}$ as an initializing partitioning for the **Louvain** (or **Leiden**) algorithm so the number of clusters k should be significantly larger than the desired number of parts in the partition $\mathcal{A}$: $k \gg \ell$.
Step 3: Run the **Louvain** (or **Leiden**) algorithm on graph G using the partition $\mathcal{C}$ as a starting point. The result of this procedure, partition $\mathcal{A}$, is the outcome of our algorithm.

2.1 Motivation

First of all, let us discuss the reasons why one might want to use the embeddings at all. One issue is a nature of graphs as data structures; they are discrete objects, which reduces the number of possible approaches to the problem of community detection. It basically forces one to use the heuristic-based approaches such as the classical **Louvain** algorithm. On the other hand, embedded latent representation is a vector of real numbers which creates new possibilities, mostly because there are more algorithms designed for working with real numbers and they are often more efficient [2]. Also, properly selected embeddings might be considered as a form of "denoising" data; they retain only the properties of nodes that are important for the task at hand, removing the remaining useless relations, resulting in a representation of data that is significantly lower dimensional and possibly easier to cluster.

In the case of community detection algorithms these advantages are clearly visible. Instead of greedily merging nodes into communities, we merge them when that

are close in the latent space, ensuring that the connections are more stable (not merely a result of a random enumerating). Indeed, equipped with a properly selected embedding, nodes that are close in the latent space will almost surely be part of the same community.

However, experimental results (see: [30]) show that using only an embedded representation to obtain the desired partition is not enough. An obvious explanation is a fact that embeddings are usually too reductive, that is, the representation gap between the graph G and its latent representation $\mathcal{E}$ is too large. Indeed, embeddings preserve some proximity of nodes but remove other useful global information that might be crucial to achieve a satisfactory result. Overcoming this issue is the main reason why the proposed solution consists of two separated partitioning steps. The reasoning is pretty straightforward: starting the **Louvain** with a visibly smaller starting set of nodes in which most sensitive elements are already connected should improve the results and decrease the volatility of the method. Similarly, it is expected that these ideas will also improve the quality of the results obtained by the **Leiden** algorithm—because of the additional refinement stage it gives a significantly better results compared to **Louvain** algorithm but still it is a greedy algorithm with all the inherent issues mentioned above.

Starting any of the two clustering algorithms from a properly generated initial partition seems to be a good idea but there are two problematic issues that we need to deal with: selection of the embedding $\mathcal{E}$ and selection of clustering algorithm. There are plenty of different embedding methods to choose from (see, for example, [6, 9, 11, 17]), that measure the proximity between nodes in different manners, which makes the selection of the algorithm a demanding task, often requiring a domain expert knowledge or time-consuming experiments. One of the goals of this work is to look at various embedding algorithms and test their behaviour in this particular task in order to find the best solution to create a guidance for future users. We also want to compare the results with divergence scores obtained by the **CGE** [12, 15]—unsupervised framework created to compare and asses different embeddings. We believe that this framework might become a useful tool, significantly simplifying the selection process of a suitable embedding.

Similarly, finding a clustering algorithm for the first step might be challenging. There are plenty of the well-known, efficient, and scalable algorithms; they might result in vastly different behaviour of the initial partitioning. For example, density-based algorithms will cluster only the points occupying the same densely connected regions whereas the points in the sparsely inhabited areas will be considered as noise and will not be assigned to any cluster. Thus, the initial partition C will contain only the nodes which are almost surely the parts of the same communities, leaving the more ambiguous nodes for the **Louvain** or **Leiden** algorithm. On the other hand, distribution-based methods of clustering will return the probability of a node belonging to each cluster, not a fixed assignment. As a result, one might fine-tune the certainty of the partition C instead of leaving it to the algorithm. Obviously, it is necessary to validate the described above intuitions which will be an important part of the experiment described in the next section.

3 Experiment Design

The main body of the experiment was written in Julia 1.7.0 programming language with additional code and packages written in Python 3.7.10. The code for execution and analysis of the experiments is available on GitHub repository[1] and so are Jupyter notebooks with a more details and further result analysis.[2] The experimental design was as follows. At the beginning, a comprehensive family of graphs with various properties was generated using the **ABCDe** (**A**rtificial **B**enchmark for **C**ommunity **D**etection) model [13, 16] and following parameter sweep: the number of nodes $n = 1000$, exponents of the power-law distributions for community sizes $\beta \in \{1.1, 1.5, 1.9\}$ and degree distributions $\gamma \in \{2.1, 2.5, 2.9\}$, community sizes $c_{\min} = 0.005n$ and $c_{\max} = 0.2n$, the minimum degree $\delta \in \{1, 2, 5\}$, the maximum degree $\Delta = \sqrt{n}$ and, finally, we set the mixing parameter $\xi \in \{0.15, 0.25, 0.35, 0.5, 0.65, 0.75, 0.85\}$ that controls the level of noise in the resulting graph. Detailed explanation on how these parameters impact the graph structure is available in [13, 14, 16].

Louvain, **Leiden**, and **ECG** algorithms were each run 50 times for every given graph in order to obtain the baseline for the comparison. Then, every graph was embedded using the following algorithms taken from the Python *OpenNE*[3] package: **Locally Linear Embedding (LLE)** [29], **Laplacian Eigenmaps (LE)** [3], **deepWalk** [26], **node2vec** [10], **LINE** [31], **SDNE** [33], **GraRep** [7] and **HOPE** [24]. For each of the selected algorithms, we tested dimensions $d \in \{8, 16, 32, 64, 128, 256\}$. To find the most suitable clustering algorithm and get the best initial partitioning C, for every embedding $\mathcal{E}$ we tested the following three methods: **k-means** [20], **HDBSCAN** [22] and **Gaussian Mixture Model (GMM)** [28]. Parameters of all the embedding and clustering algorithms used in this experiment are further described in the aforementioned accompanying Jupyter notebook. Finally, every partition C was used as the initial partitioning for both **Leiden** and **Louvain** algorithm. To achieve comparable results, both methods were run 50 times on every C.

Roughly 55,000 different embeddings were tested with more than 1,500,000 initial partitions. Experiments were performed on the machines with 32 Intel Xeon Processors (Cascadelake) 2.30 GHZ vCPUs with 160 GB RAM memory, 120 GB disk space and Ubuntu 20.04.1 operating system. Computations were run simultaneously on eight machines for five consecutive days, totalling in around 960 vCPU hours.

[1] https://github.com/bartoszpankratz/ECCD.

[2] https://github.com/bartoszpankratz/ECCD/blob/main/Embedding-Clustering_Community_Detection_Experiment.ipynb.

[3] https://github.com/thunlp/OpenNE.

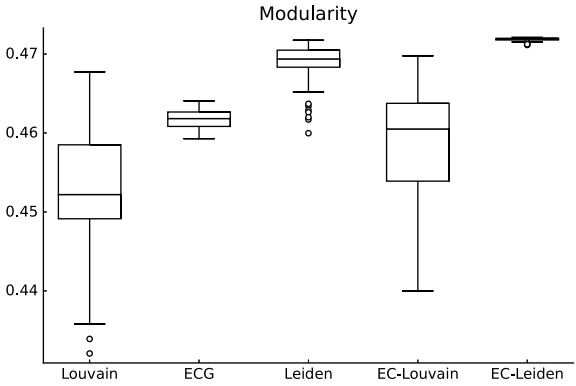


Fig. 1 Comparison of the modularity function for a single but representative set of parameters: $\xi = 0.5$, $\beta = 1.5$, $\gamma = 2.5$, and $\delta = 5$

4 Results

Figure 1 presents the results for one representative set of parameters: $\xi = 0.5$, $\beta = 1.5$, $\gamma = 2.5$, and $\delta = 5$. As one can easily see, the results obtained by **Louvain** after using a better initialization procedure are clearly improved. Both **ECG** and **EC–Louvain**[4] are able to improve over the vanilla **Louvain**. In some rare cases, **EC–Louvain** is able to achieve performance similar to **Leiden**. But what is the most interesting, adding the initial partitioning C to **Leiden** significantly improves its quality and reduces the volatility.

The presented figure shows the results only for a single case; Table 1 shows how different values of ξ impact the performance of the algorithms. The relation here is pretty obvious; ξ is a *noise parameter*, it controls the expected fraction of edges between communities. As a result, with an increasing value of ξ one should expect the modularity to decrease, but also relative better performance of the augmented methods. We could see that **EC–Louvain** gives a relatively small improvement over the baseline **Louvain**, but **Leiden** with initial partitioning is able to outclass the rest of the algorithms with a large margin. Also it reduces the volatility to the negligible levels. Interestingly, for $\xi = 0.75$ **ECG** gives worse results than **Louvain**; **ECG** seems to be very sensitive to the graph's parametrization. In some cases it performs very well (see, for example, Fig. 1), but it can also be weaker than **Louvain**. In comparison, it is never a case for the **EC** methods—in the worst case scenario, they return the same value of the modularity as **Louvain**.

One can see similar pattern for other parameters of the **ABCD** model[5]: when change of the parameter distorts the community structure of the graph, then the advantage from using the augmented methods is more visible. However, in almost

[4] **EC** stands for **Embedding–Clustering** and denotes the proposed extension of **Louvain** and **Leiden** algorithms. If not otherwise stated, the results for the **EC** algorithm uses the best possible initial partitioning C.

[5] For details please refer to: https://github.com/bartoszpankratz/ECCD/blob/main/Embedding-Clustering_Community_Detection_Experiment.ipynb.

Table 1 Comparison of the algorithms for different values of ξ ($\beta = 1.5$, $\gamma = 2.5$, and $\delta = 5$). Column *Louvain (baseline)* shows the average modularity obtained by this algorithm. Other columns present the average difference between the results of each algorithm and **Louvain**. Standard deviation is given in parenthesis

ξ	Louvain (baseline)	ECG	Leiden	EC–Louvain	EC–Leiden
0.35	0.58132	0.00027	0.0029	0.00145	0.00302
	(0.00502)	(0.00019)	(0.00042)	(0.00237)	(0.0)
0.5	0.45263	0.00907	0.01593	0.00596	0.0192
	(0.00847)	(0.00124)	(0.00289)	(0.00696)	(0.0002)
0.75	0.30533	−0.01987	0.01955	0.00976	0.03096
	(0.00357)	(0.00279)	(0.0029)	(0.00448)	(0.00206)

all cases **EC–Louvain** gives small to mediocre improvement, but **EC–Leiden** gives a significant performance boost. Why is this happening? The answer is pretty straightforward and lies in the very nature of both algorithms, **Louvain** and **Leiden**.

As it was mentioned before, **Louvain** merges two nodes if such move maximizes the modularity locally, without any broader context. The initial partitioning C was designed to overcome this issue, guaranteeing the stability of the first step of the algorithm. But this problem is prevailing in later steps until the algorithm reaches the stage when the communities are large enough. As a result, the impact of the initial "good" partitioning is minimized. This problem might be fixed by repeating the embedding process after every iteration up to the moment when the algorithm reaches its stable stage but obviously such procedure would be unfeasible for large graphs as it is very time consuming.

On the other hand, refinement stage in **Leiden** solves this issue. After every iteration, when communities are created in the same manner as in **Louvain**, they are split and recombined into new, better partitions, ensuring that all nodes are optimally assigned in the context of the given subgraph induced by a single community. But still, **Leiden** backtracks only in a limited scope; early on, when initialized with a singleton partition, it might still merge nodes that should not belong to the same community and that will be irreversible. By initializing it with a fine-tuned initial partitioning C we ensure that this will not happen.

The last question remaining concerns the way how one should design the procedure. Clearly, proper selection of embedding and clustering gives a significant boost of the performance of **Leiden** (and to the lesser extend **Louvain**), but how should one chooses them?

Figure 2 shows the relation between the modularity and the CGE scores obtained by the unsupervised framework for comparing graph embeddings [12, 15], both local and global. Results show some interesting behavior. Let is first focus on the **EC–Louvain**. As can be seen on the two upper plots, the relation between the quality of the embedding and the achieved modularity is pretty insignificant, basically any kind of the reasonable embedding could give us a similar performance. These observations

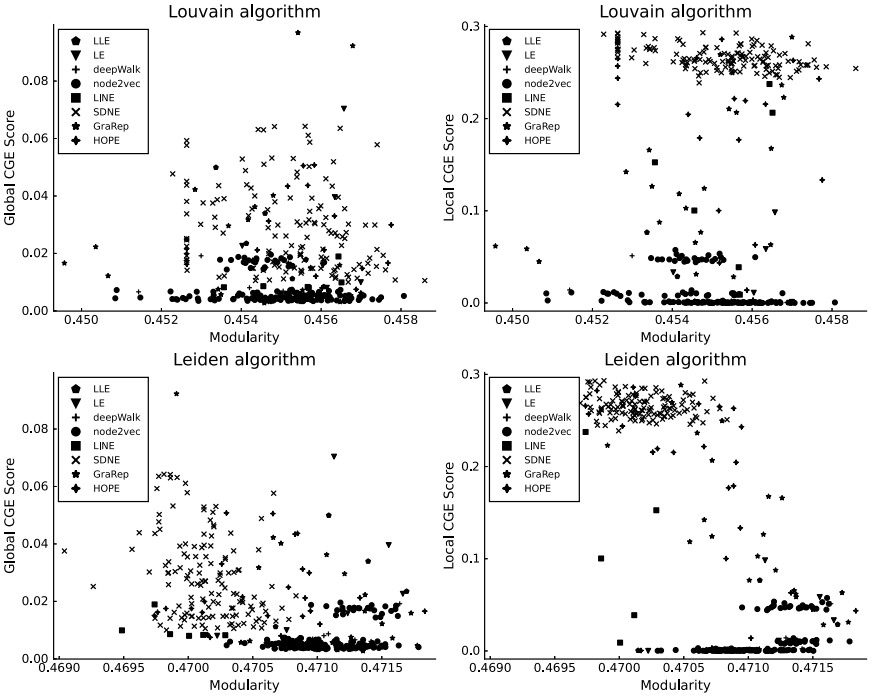


Fig. 2 Comparison of the modularity function and global/local CGE scores for different embedding algorithms and a single set of parameters: $\xi = 0.5$, $\beta = 1.5$, $\gamma = 2.5$, and $\delta = 5$

are in line with previous results showing that the inherent volatility of **Louvain** decreases the relevance of the initial partitioning C. However, it is not a case for **EC–Leiden**. We could clearly see that there is a strong relation between the quality of the embedding and the final modularity value. Moreover, plots show that **node2vec** is usually the best performing embedding algorithm. It is quite intuitive; it represents the nodes through the use of random walks and in the case of the community detection it is a natural form of representing proximities—nodes that are parts of the same communities are likely to be present close to each other in the associated random walks.

Let us now briefly comment on the performance of different clustering algorithms. **HDBSCAN** was usually the best one, which was somewhat foreseeable—this density based algorithm clusters nodes only when they are certainly a part of the same community. Figure 3 shows the result for an example but representative parametrization. Further analysis of the impact of parametrization of both embeddings and clustering algorithms is available in the accompanying Jupyter notebook.

However, presented method have one serious limitation—the execution time of augmented algorithms is significantly (about two orders of magnitude) higher than the execution time of the baseline methods, which is a directly caused by time-

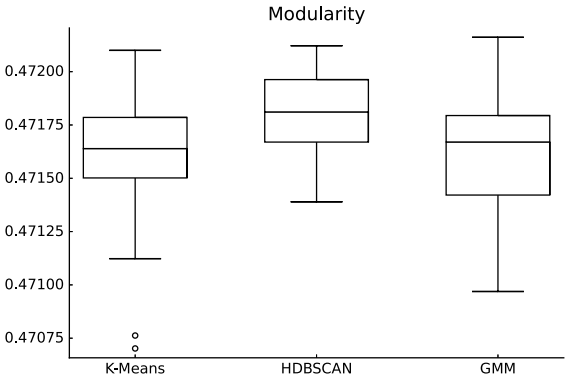


Fig. 3 Comparison of the modularity function for a graph embedded with **node2vec** into a 16-dimensional space and one set of parameters: $\xi = 0.5$, $\beta = 1.5$, $\gamma = 2.5$, and $\delta = 5$

complexity of the embedding algorithms. At the moment, the method presented in this paper might be somewhat infeasible for some applications, but it shows how the community detection algorithms could be further refined in order to obtain better and more stable solutions.

5 Final Remarks

The results presented in this paper show that the usage of the initial partitioning C obtained by clustering of nodes in graph embeddings improves the results of the popular community detection algorithms. In the case of **Louvain** the impact is rather small, almost negligible, but the initial partitioning of **Leiden** significantly improves its performance and reduce the volatility. We also provided results showing that there are some certain classes of embeddings (such as **node2vec**) and clustering algorithms (such as **HDB-SCAN**) that are the most suitable for this particular task.[6]

Acknowledgements Hardware used for the computations was provided by the SOSCIP consortium. Launched in 2012, the SOSCIP consortium is a collaboration between Ontario's research-intensive post-secondary institutions and small- and medium-sized enterprises (SMEs) across the province. Working together with the partners, SOSCIP is driving the uptake of AI and data science solutions and enabling the development of a knowledge-based and innovative economy in Ontario by supporting technical skill development and delivering high-quality outcomes. SOSCIP supports industrial-academic collaborative research projects through partnership-building services and access to leading-edge advanced computing platforms, fuelling innovation across every sector of Ontario's economy.

[6] https://www.soscip.org/.

References

1. Aiello, W., Chung, F., Lu, L.: A random graph model for massive graphs. In: Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, pp. 171-180. STOC '00, Association for Computing Machinery, New York, NY, USA (2000). https://doi.org/10.1145/335305.335326
2. Bartz-Beielstein, T., Zaefferer, M.: Model-based methods for continuous and discrete global optimization. Appl. Soft Comput. **55**, 154–167 (2017). https://www.sciencedirect.com/science/article/pii/S1568494617300546
3. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic, pp. 585–591. NIPS'01, MIT Press, Cambridge, MA, USA (2001)
4. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. Theory Exp. **2008**(10), 10008 (2008)
5. Brandes, U., Delling, D., Gaertler, M., Gorke, R., Hoefer, M., Nikoloski, Z., Wagner, D.: On modularity clustering. IEEE Trans. Knowl. Data Eng. **20**(2), 172–188 (2008)
6. Cai, H., Zheng, V.W., Chang, K.C.C.: A comprehensive survey of graph embedding: problems, techniques and applications (2018)
7. Cao, S., Lu, W., Xu, Q.: Grarep: Learning graph representations with global structural information. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 891–900. CIKM '15, Association for Computing Machinery, New York, NY, USA (2015). https://doi.org/10.1145/2806416.2806512
8. Fortunato, S.: Community detection in graphs. Phys. Rep. **486**(3–5), 75–174 (2010). https://doi.org/10.1016/j.physrep.2009.11.002
9. Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: a survey. Knowl. Syst. **151**, 78–94 (2018). https://doi.org/10.1016/j.knosys.2018.03.022
10. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks (2016)
11. Hamilton, W.L., Ying, R., Leskovec, J.: Representation learning on graphs: methods and applications (2018)
12. Kamiński, B., Kraiński, l., Prałat, P., Théberge, F.: A multi-purposed unsupervised framework for comparing embeddings of undirected and directed graphs (2021). https://arxiv.org/abs/2112.00075
13. Kamiński, B., Olczak, T., Pankratz, B., Prałat, P., Théberge, F.: Properties and performance of the abcde random graph model with community structure (2022). https://arxiv.org/abs/2203.14899
14. Kaminski, B., Pankratz, B., Pralat, P., Theberge, F.: Modularity of the abcd random graph model with community structure (2022). https://arxiv.org/abs/2203.01480
15. Kamiński, B., Prałat, P., Théberge, F.: An unsupervised framework for comparing graph embeddings. J. Complex Netw. **8**(5), cnz043 (2020)
16. Kamiński, B., Prałat, P., Théberge, F.: Artificial benchmark for community detection (abcd)-fast random graph model with community structure. Netw. Sci. 1–26 (2021)
17. Kamiński, B., Prałat, P., Théberge, F.: Mining Complex Networks. Chapman and Hall/CRC (2021)
18. Lancichinetti, A., Fortunato, S.: Community detection algorithms: a comparative analysis. Phys. Rev. E **80**(5) (2009). https://doi.org/10.1103/PhysRevE.80.056117
19. Leskovec, J., Lang, K.J., Mahoney, M.W.: Empirical comparison of algorithms for network community detection (2010). https://arxiv.org/abs/1004.3539
20. Lloyd, S.P.: Least squares quantization in pcm. IEEE Trans. Inf. Theory **28**, 129–137 (1982)
21. McCarthy, A.D., Chen, T., Ebner, S.: An exact no free lunch theorem for community detection. In: Complex Networks and Their Applications VIII, pp. 176–187. Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-36687-2_15
22. McInnes, L., Healy, J., Astels, S.: hdbscan: Hierarchical density based clustering. J. Open Source Softw. **2**(11), 205 (2017). https://doi.org/10.21105/joss.00205

23. Newman, M.E.J.: Modularity and community structure in networks. Proc. Nat. Acad. Sci. **103**(23), 8577–8582 (2006). https://doi.org/10.1073/pnas.0601602103
24. Ou, M., Cui, P., Pei, J., Zhang, Z., Zhu, W.: Asymmetric transitivity preserving graph embedding. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1105–1114. KDD '16, Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2939672.2939751
25. Peel, L., Larremore, D.B., Clauset, A.: The ground truth about metadata and community detection in networks. Sci. Adv. **3**(5), e1602548 (2017). https://doi.org/10.1126/sciadv.1602548
26. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk. Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2014). https://doi.org/10.1145/2623330.2623732
27. Poulin, V., Théberge, F.: Ensemble clustering for graphs: comparisons and applications. Appl. Netw. Sci. **4**(1) (2019). https://doi.org/10.1007/s41109-019-0162-z
28. Rasmussen, C.E.: The infinite gaussian mixture model. In: Proceedings of the 12th International Conference on Neural Information Processing Systems, pp. 554–560. NIPS'99, MIT Press, Cambridge, MA, USA (1999)
29. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**(5500), 2323–2326 (2000). https://science.sciencemag.org/content/290/5500/2323
30. Tandon, A., Albeshri, A., Thayananthan, V., Alhalabi, W., Radicchi, F., Fortunato, S.: Community detection in networks using graph embeddings. Phys. Rev. E **103**, 022316 (2021). https://link.aps.org/doi/10.1103/PhysRevE.103.022316
31. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line. Proceedings of the 24th International Conference on World Wide Web (2015). https://doi.org/10.1145/2736277.2741093
32. Traag, V., Waltman, L., van Eck, N.J.: From Louvain to Leiden: guaranteeing well-connected communities. Sci. Rep. **9**, 5233 (03 2019)
33. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1225–1234. ACM (2016)
34. Yang, J., Leskovec, J.: Defining and evaluating network communities based on ground-truth (2012). https://arxiv.org/abs/1205.6233

Modeling Node Exposure for Community Detection in Networks

Sameh Othman, Johannes Schulz, Marco Baity-Jesi, and Caterina De Bacco

Abstract In community detection, datasets often suffer a sampling bias for which nodes which would normally have a high affinity appear to have zero affinity. This happens for example when two affine users of a social network were not exposed to one another. Community detection on this kind of data suffers then from considering affine nodes as not affine. To solve this problem, we explicitly model the (non-) exposure mechanism in a Bayesian community detection framework, by introducing a set of additional hidden variables. Compared to approaches which do not model exposure, our method is able to better reconstruct the input graph, while maintaining a similar performance in recovering communities. Importantly, it allows to estimate the probability that two nodes have been exposed, a possibility not available with standard models.

Keywords Networks · Community detection · Latent variable models

1 Introduction

Modeling the mechanisms of how nodes interact in networks is a relevant problem in many applications. In social networks, we observe a set of interactions between people, and one can use this information to cluster them into communities based on some notion of similarity [7]. Broadly speaking, the connections between users can be used to infer users' membership, and this in turns determines the likelihood that a pair of users interacts. Real networks are often sparse, people interact with a tiny amount of individuals, compared to the large set of possible interactions that they could in principle explore. Traditionally, models for community detection in networks treat an existing link as a positive endorsement between individuals: if two

S. Othman · J. Schulz · C. De Bacco (✉)
Max Planck Institute for Intelligent Systems, Cyber Valley, Tuebingen 72076, Germany
e-mail: caterina.debacco@tuebingen.mpg.de

M. Baity-Jesi
Eawag, Überlandstrasse 133, 8600 Dübendorf, Switzerland

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_18

people are friends in a social network, this means they like each other. In assortative communities, where similar nodes are more likely to be in the same group [8, 13], this encourages the algorithm to put these two nodes into the same community. On the contrary, a non-existing link influences the model to place them into different communities, as if the two non-interacting individuals were not compatible. However, many of these non-existing links—especially in large-scale networks—are absent because the individuals are not aware of each other, rather than because they are not interested in interacting. This is a general problem in many network datasets: we know that interacting nodes have a high affinity, but we can not conclude the contrary about non-interacting nodes.

This problem has been explored in the context of recommender systems [2, 10, 18, 19], where it is crucial to learn what items that a user did not consume could be of interest. In this context, items' exposure is often modeled by means of propensity scores or selection biases assigned to user-item pairs that increase the probability of rare consumption events.

It is not clear how to adapt these techniques to the case of networks of interacting individuals, hence the investigation of this problem in the context of networks is still missing. Existing approaches partially account for this by giving more weight to existing links, as in probabilistic generative models that use a Poisson distribution for modeling the network adjacency matrix [1, 5, 17, 20]. These methods are effective, but may be missing important information contained in non-existing links.

2 Community Detection with Exposure

We address this problem by considering a probabilistic formulation that assigns probabilities to pairs of nodes of being exposed or not. These are then integrated into standard probabilistic approaches for generative networks with communities. For this, as a reference model we consider MULTITENSOR [5], as it is a flexible model that takes in input a variety of network structures (e.g. directed or undirected networks, weighted or unweighted) and detects overlapping communities in a principled and scalable way.

2.1 Representing Exposure

Consider an $N \times N$ *observed* network adjacency matrix $\mathbf{A}^{(o)}$, where $A_{ij}^{(o)} \geq 0$ is the weight of the interaction between nodes i and j, this is the input data. For instance, $A_{ij}^{(o)}$ could be the number of times that i and j met or exchanged messages. If a link $A_{ij}^{(o)}$ exists, this indicates an affinity between individuals i and j, triggered by both individuals' inner preferences. If the link does not exist ($A_{ij}^{(o)} = 0$), one usually assumes that this indicates a lack of affinity between i and j. However, the link might

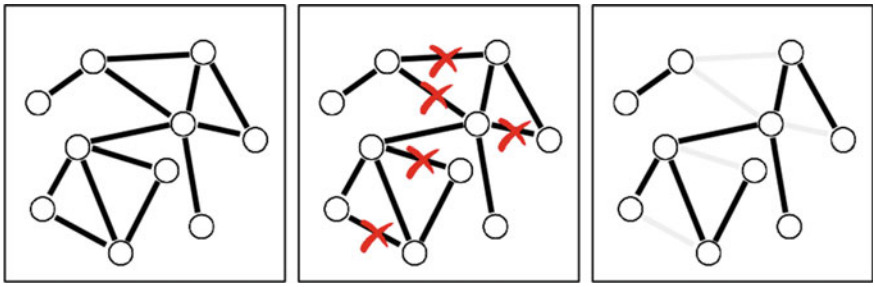


Fig. 1 Diagram of the exposure mechanism. On the **left** we have the full graph ($\mathbf{A}^{(g)}$). We then set to zero the probability of some connections through a mask $\mathbf{Z}$ (**center**), and reconstruct the true graph and communities based solely on the visible links, $\mathbf{A}^{(o)}$ (**right**)

not exist simply because i and j never met. This is the case in social networks, where an *ego* might follow an *alter* because of personal preference, but this choice is subject to being exposed to the *alter* in the first place. This suggests that the event of being exposed to someone influences the patterns of interactions observed in networks. We are interested in incorporating this notion of exposure in modeling network data, and investigate how results change.

To represent this, we postulate the existence of a *ground-truth adjacency matrix*, $\mathbf{A}^{(g)}$, that indicates the affinity between nodes i and j regardless of whether the two nodes were exposed to each other (Fig. 1—left). In addition, we introduce a *dilution matrix* $\mathbf{Z}$ (red crosses in Fig. 1—center), with values $Z_{ij} = 0, 1$ indicating whether nodes i and j were exposed ($Z_{ij} = 1$) or not ($Z_{ij} = 0$). The observed matrix is then the element-wise product of the ground truth network times the dilution matrix,

$$\mathbf{A}^{(o)} = \mathbf{A}^{(g)} \otimes \mathbf{Z}\,, \tag{1}$$

where $\otimes$ indicates an element-by-element multiplication. A diagram of the resulting matrix is shown in Fig. 1—right. Through this representation, a zero-entry $A^{(o)}_{ij} = 0$ can be attributed to $A^{(g)}_{ij} = 0$ (lack of affinity), $Z_{ij} = 0$ (lack of exposure) or both.

Standard models for community detection do not account for exposure, therefore they treat a zero-entry $A^{(o)}_{ij} = 0$ as a signal for non-affinity. We aim at measuring both communities and exposure, given the observed data $A^{(o)}_{ij}$. In other words, for a given node i, we would like to estimate its community membership and for a given pair (i, j) we want to estimate the probability that they were exposed to each other. For simplicity, we show derivations for the case of undirected networks, but similar ones apply to directed ones.

2.2 The Ground Truth Adjacency Matrix

In our notation, we use θ to denote the latent variables affecting community detection, *i.e.* determining the probability of observing an interaction between i and j given that they have been exposed. Following the formalism of Ref. [5], we assign a K-dimensional hidden variable u_i to every node i. Since different communities may interact in different ways, we also introduce a $K \times K$ affinity matrix w, regulating the density of interactions between different groups. The latent variables related to the ground truth matrix are then $\theta = (u, w)$.

We express the expected interaction between two nodes through a parameter

$$\lambda_{ij} = \sum_{k,q}^{K} u_{ik} u_{jq} w_{kq} \,, \tag{2}$$

and extract the elements of $\mathbf{A}^{(\mathrm{g})}$ from a Poisson distribution with mean λ_{ij},

$$P(A_{ij}^{(\mathrm{g})}|u_i, u_j, w) = \mathrm{Pois}\left(A_{ij}^{(\mathrm{g})}; \lambda_{ij}\right) = \frac{e^{-\lambda_{ij}} \lambda_{ij}^{A_{ij}^{(\mathrm{g})}}}{A_{ij}^{(\mathrm{g})}!} \,. \tag{3}$$

We then assume conditional independence between different pairs of edges given the latent variables $P(\mathbf{A}^{(\mathrm{g})}|u, u, w) = \prod_{i<j} P(A_{ij}^{(\mathrm{g})}|u_i, u_j, w)$, but this can be generalized to more complex dependencies [4, 15, 16]. We do not explore this here.

2.3 The Observed Adjacency Matrix

The observed adjacency matrix depends on whether two nodes were exposed or not, through the matrix $\mathbf{Z}$. If $Z_{ij} = 1$, the two nodes are exposed, and the edge comes from the ground truth matrix, *i.e.* $P(A_{ij}^{(\mathrm{o})}|Z_{ij} = 1, \theta) = P(A_{ij}^{(\mathrm{g})}|\theta) = \mathrm{Pois}(A_{ij}^{(\mathrm{g})}; \lambda_{ij})$. If $Z_{ij} = 0$, then $A_{ij}^{(\mathrm{o})} = 0$ regardless of λ_{ij}. Therefore, the elements of $\mathbf{A}^{(\mathrm{o})}$ are extracted from the distribution

$$P(A_{ij}^{(\mathrm{o})}|Z_{ij}, \theta) = \mathrm{Pois}(A_{ij}^{(\mathrm{o})}; \lambda_{ij})^{Z_{ij}} \, \delta(A_{ij}^{(\mathrm{o})})^{1-Z_{ij}} \,. \tag{4}$$

Since Z_{ij} is binary, we assign it a Bernoulli prior with parameter μ_{ij},

$$P(\mathbf{Z}|\mu) = \prod_{i<j} P(Z_{ij}|\mu_{ij}) = \prod_{i<j} \left(\mu_{ij}\right)^{Z_{ij}} \left(1 - \mu_{ij}\right)^{1-Z_{ij}} \,. \tag{5}$$

The parameter μ_{ij} will depend on some latent variable related to nodes i and j. There are several possible choices for that. Here, we consider a simple setting:

$$\mu_{ij} = \mu_i \, \mu_j \,, \tag{6}$$

$$\mu_i \in [0, 1] \,, \tag{7}$$

This allows to keep the number of parameters small and has an easy interpretation. In fact, the parameter μ_i acts as the propensity of an individual to be exposed to others: the higher its value, the higher the probability that node i will be exposed to other nodes. This way of modeling exposure only adds one more parameter per node, allowing for heterogeneous behaviors among users while keeping the model compressed. The full set of variables that need to be inferred consists of the u, the w and the μ variables, which amounts to $NK + K^2 + N$ parameters, which is one order of magnitude smaller than the N^2 elements of $\mathbf{A}^{(o)}$.

2.4 Inference and Expectation-Maximization

Given the data $\mathbf{A}^{(o)}$, our goal is to first determine the values of the parameters θ, which fixes the relationship between the hidden indicator Z_{ij} and the data, and then to approximate Z_{ij} given the estimated θ.

We perform this using statistical inference as follows. Consider the posterior distribution $P(\mathbf{Z}, \theta | \mathbf{A}^{(o)})$. Since the dilution $\mathbf{Z}$ is independent from the parameters θ and all the edges are considered conditionally independent given the parameters, Bayes' formula gives

$$P(\mathbf{Z}, \theta | \mathbf{A}^{(o)}) = \frac{P(\mathbf{A}^{(o)} | \mathbf{Z}, \theta) \, P(\mathbf{Z} | \mu) \, P(\theta)}{P(\mathbf{A}^{(o)})} \,. \tag{8}$$

Summing over all the possible indicators we have:

$$P(\theta | \mathbf{A}^{(o)}) = \sum_{\mathbf{Z}} P(\mathbf{Z}, \theta | \mathbf{A}^{(o)}) = \prod_{i<j}^{N} \sum_{Z_{ij}=0,1} P(Z_{ij}, \theta | \mathbf{A}^{(o)}) \quad , \tag{9}$$

which is the quantity that we need to maximize to extract the optimal θ. It is more convenient to maximize its logarithm, as the two maxima coincide. We use Jensen's inequality:

$$\log P(\theta | \mathbf{A}^{(o)}) = \log \sum_{\mathbf{Z}} P(\mathbf{Z}, \theta | \mathbf{A}^{(o)}) \geq \sum_{\mathbf{Z}} q(\mathbf{Z}) \, \log \frac{P(\mathbf{Z}, \theta | \mathbf{A}^{(o)})}{q(\mathbf{Z})} := \mathcal{L}(q, \theta, \mu) \tag{10}$$

where $q(\mathbf{Z})$ is any distribution satisfying $\sum_{\mathbf{Z}} q(\mathbf{Z}) = 1$, we refer to this as the variational distribution.

Inequality (10) is saturated when

$$q(\mathbf{Z}) = \frac{P(\mathbf{Z}, \theta | \mathbf{A}^{(o)})}{\sum_{\mathbf{Z}} P(\mathbf{Z}, \theta | \mathbf{A}^{(o)})}, \tag{11}$$

hence this choice of q maximizes $\mathcal{L}(q, \theta, \mu)$ with respect to q. Further maximizing it with respect to θ gives us the optimal latent variables. This can be done in an iterative way using Expectation-Maximization (EM), alternating between maximizing with respect to q using Eq. (11) and then maximizing $\mathcal{L}(q, \theta, \mu)$ with respect to θ and μ.

To obtain the updates for the parameters we need to derive the equations that maximize $\mathcal{L}(q, \theta, \mu)$ with respect to θ and μ and set these derivatives to zero. This leads to the following closed-form updates:

$$u_{ik} = \frac{\sum_j Q_{ij} A_{ij} \sum_q \rho_{ijkq}}{\sum_j Q_{ij} \sum_q u_{jq} w_{kq}} \tag{12}$$

$$w_{kq} = \frac{\sum_{i,j} Q_{ij} A_{ij} \rho_{ijkq}}{\sum_{i,j} Q_{ij} u_{ik} u_{jq}} \tag{13}$$

$$\rho_{ijkq} = \frac{u_{ik} u_{jq} w_{kq}}{\sum_{k,q} u_{ik} u_{jq} w_{kq}} \tag{14}$$

$$\mu_i = \frac{\sum_j Q_{ij}}{\sum_j \frac{(1-Q_{ij})\mu_j}{(1-\mu_i \mu_j)}}, \tag{15}$$

where we defined $Q_{ij} = \sum_{\mathbf{Z}} Z_{ij} q(\mathbf{Z})$ the expected value of Z_{ij} over the variational distribution.

As μ_i appears on both sides of Eq. (15), this can be solved with root-finding methods bounding μ_i to the interval [0, 1], to be compatible as a parameter of the Bernoulli prior.[1]

Finally, to evaluate $q(Z)$, we substitute the estimated parameters inside Eq. (8), and then into Eq. (11) to obtain:

$$q(\mathbf{Z}) = \prod_{i<j} Q_{ij}^{Z_{ij}} (1 - Q_{ij})^{(1-Z_{ij})}, \tag{16}$$

where

$$Q_{ij} = \frac{\text{Pois}(A_{ij}; \lambda_{ij}) \mu_{ij}}{\text{Pois}(A_{ij}; \lambda_{ij}) \mu_{ij} + \delta(A_{ij}) (1 - \mu_{ij})}. \tag{17}$$

In other words, the optimal $q(\mathbf{Z})$ is a product $\prod_{i<j} q_{ij}(\mathbf{Z}_{ij})$ of Bernoulli distributions q_{ij} with parameters Q_{ij}. This parameter is also a point-estimate of the exposure variable, as for the Bernoulli distribution $Q_{ij} = \mathbb{E}_q [\mathbf{Z}_{ij}]$.

[1] In practice, we limit the domain of μ_i to the interval $[\epsilon, 1 - \epsilon]$, where ϵ is a small hyperparameter chosen to avoid numerical overflows of $\mathcal{L}$. To maintain the model interpretable in terms of exposure, at the end of the optimization we set to zero each $\mu_i \equiv \epsilon$ and to one each $\mu_i \equiv 1 - \epsilon$.

The algorithmic EM procedure then works by initializing at random all the parameters and then iterating Eqs. (12)–(15) for fixed q, and the calculating Eq. (17) given the other parameters, and so on until convergence of $\mathcal{L}$. The function $\mathcal{L}$ is not convex, hence we are not guaranteed to converge to the global optimum. In practice, one needs to run the algorithm several times with different random initial parameters' configurations and then select the run that leads to best values of $\mathcal{L}$. In the following experiments we use 5 of such realizations.

3 Results

We test our algorithm on synthetic and real data, and compare it to its formulation without exposure, *i.e.* the MULTITENSOR algorithm described in Ref. [5]. In the following, we refer to our algorithm as EXP, and we use NOEXP for the algorithm that does not utilize exposure.

3.1 Synthetic Data

Synthetic data experiments are particularly interesting, because we can validate our model performances on the ground truth values. The creation of a synthetic dataset follows the generative model described in Sect. 2.1:

1. For a graph with $N = 500$ nodes, we generate the latent parameters θ and μ as follows. We draw overlapping communities by sampling u_i from a Dirichlet distribution with parameter $\alpha_k = 1, \forall k$; we choose an assortative w by selecting the off-diagonal entries to be 0.001 times smaller then the on-diagonal ones. We then vary $K \in [3, 5, 8]$. We draw μ_i from a Beta distribution $\text{Beta}(\mu_i; 2, \beta)$, where we vary $\beta \in [0.1, 10]$ to tune the fraction of unexposed links.
2. Sample $\mathbf{A}^{(\text{g})}{}_{ij}$ from a Poisson distribution with means $\lambda_{ij} = \sum_{k,q} u_{ik} u_{jq} w_{kq}$.
3. Sample $\mathbf{Z}$ from a Bernoulli distribution of means $\mu_{ij} = \mu_i \mu_j$.
4. Calculate the matrix $\mathbf{A}^{(\text{o})} = \mathbf{A}^{(\text{g})} \otimes \mathbf{Z}$. This matrix has on average $\langle k \rangle$ links per node.

We repeat this procedure 10 times for each set of parameters to obtain different random realizations of synthetic data. We then apply the EXP and NOEXP algorithms to $\mathbf{A}^{(\text{o})}$ to learn the parameters and study the performance as a function of $\langle k \rangle$, controlling the density of observed edges.

Reconstructing hidden links We start by testing the ability of the model to predict missing links, a procedure often used as a powerful evaluation framework for comparing different models [11, 12]. We use a 5-fold cross-validation scheme where we hide 20% of the edges in $\mathbf{A}^{(\text{o})}$ and train the model on the remaining 80%. Performance is then computed on the hidden 20% of the edges. As a performance evaluation

metric we measure the area under the receiver operating characteristic curve (AUC) between the inferred values and the ground truth used to generate $\mathbf{A}^{(o)}$ on the test set. The AUC is the probability that a randomly selected existing edge is predicted with a higher score than a randomly selected non-existing edge. A value of 1 means optimal performance, while 0.5 is equivalent to random guessing. As the score of an edge $\mathbf{A}^{(o)}_{i,j}$ we use the quantity $Q_{ij}\,\lambda_{ij}$ for EXP, and λ_{ij} for NOEXP. In both cases, these are the expected values of $A^{(o)}_{ij}$ using the estimates of the latent parameters and, for EXP, over the inferred $q(\mathbf{Z})$. We find that the EXP algorithm outperforms NOEXP by a large margin, which increases as the network becomes more dense, going above 10%, as shown in Fig. 2—left. At low densities, the performance increase of the EXP algorithm is narrow for models with a large number of communities, while at large densities it becomes bigger and independent of the number of communities. This result suggests that EXP is capturing the input data better–consistently for varying dilution densities–than a model that does not account for exposure.

Guessing unexposed links Our algorithm not only allows us to predict missing edges but also gives interpretable estimates of the probability of exposure between nodes. These probabilities follow naturally from the posterior distribution on $\mathbf{Z}$, which is the Bernoulli distribution in Eq. (16). Standard algorithms as NOEXP cannot estimate this. We can use the mean value Q_{ij} as in Eq. (17) as a score of an edge to compute the AUC between inferred and ground truth values of $\mathbf{Z}$, analogously to what was done for reconstructing $\mathbf{A}^{(o)}$. We report in Fig. 2—center the ability of EXP to reconstruct the matrix $\mathbf{Z}$, i.e. to infer which edges were removed in the dilution step. The AUC varies between 0.65 and 0.75, well above the random baseline of 0.5. We notice how the values increase as the density of connection increases, but stay above 0.65 even at small density values, where reconstruction is more challenging.

Inferring communities In Fig. 2—right, we can see that EXP and NOEXP show similar performances in reconstructing communities. From this plot we can also notice how reconstruction improves for larger densities and fewer communities. The similar performances may be due to selecting a simple prior as in Eq. (6). For a more structured prior, the inferred communities would likely change and potentially improve. Given this similar community detection abilities but the better predictive power in reconstructing $\mathbf{A}^{(o)}$, we argue that the learned Q_{ij}'s are important to boost prediction compared to a model that does not properly account for exposure. This is true even for a simple prior.

Dependence on the number of communities All of these metrics exhibit a scaling w.r.t. the variable $\langle k \rangle / K$, as can be seen in the insets of Fig. 2. This suggests that the curves seem to be independent of the number of communities when accounting for this rescaling. Thus observing the behavior for one particular value of K should be informative enough to understand how the model behaves for various densities.

Suggesting good matches Since the EXP algorithm is good at predicting which nodes were removed from the original graph (Fig. 2—center), we can use this to address the following question: Is the EXP algorithm able to suggest two nodes that have high affinity despite not having any connection? In other words, we are asking whether

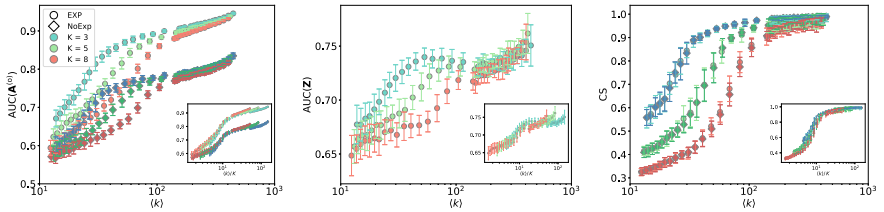


Fig. 2 Performance of the EXP and NoEXP algorithms on synthetic data. The matrix $\mathbf{A}^{(g)}$ has $K = 3, 5, 8$ communities and $N = 500$. The exposure mask Z is extracted from a binomial distribution with parameter $\mu_{ij} = \mu_i \mu_j$. **Left**: AUC between the inferred values and the ground truth used to generate $\mathbf{A}^{(o)}$. **Center**: AUC of the reconstruction of the exposure mask $\mathbf{Z}$. **Right**: Cosine similarity between inferred and ground truth communities. **Inset**: We show the same data as in the main plots by rescaling the average number of links by the number of communities

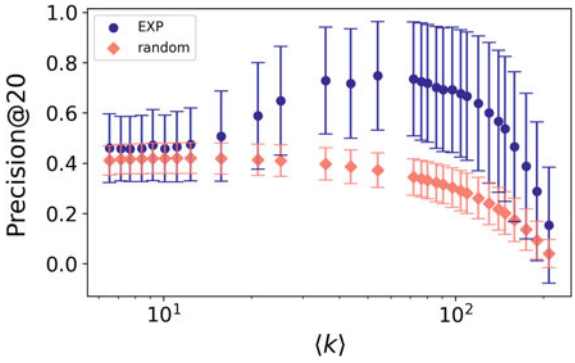


Fig. 3 Suggesting unexposed compatible nodes. For each node i, we suggest the 20 links with highest λ_{ij} inferred by our algorithm from the non-observed links where $A^{(o)}_{ij} = 0$. We show the P@20 averaged across all nodes and compare with a uniform-at-random baseline (random) where 20 nodes are selected at random among the available ones. Error bars are standard deviations. Here we use a synthetic network generated as in Sect. 3.1 with $N = 500$ and $K = 5$

we are able to find links that in $\mathbf{A}^{(o)}$ are absent, but have a high expected value in $\mathbf{A}^{(g)}$. To test this ability, we take for each node i: a) all the possible neighbors j such that $A^{(o)}_{ij} = 0$; b) select among them the 20 with the largest inferred affinity λ_{ij}; and c) check how many of those are present in $\mathbf{A}^{(g)}$. We call Precision@20 (P@20) the fraction of links which were correctly inferred, averaging across all nodes. In Fig. 3 we show that for intermediate dilution values, the P@20 reaches around 80%, and outperforms random guessing at any value of the dilution. Notice that random guessing in not constant in $\langle k \rangle$. This is because this depends on the number of missing links in $\mathbf{A}^{(o)}$, and those depend both on the density of $\mathbf{A}^{(g)}$ and on the dilution mask $\mathbf{Z}$. Specifically, P@20 of the random baseline goes as $(\langle k \rangle_g - \langle k \rangle)/(N - \langle k \rangle)$, where $\langle k \rangle_g$ is degree of $\mathbf{A}^{(g)}$. This is a decreasing function of $\langle k \rangle$, for $\langle k \rangle_g < N$.

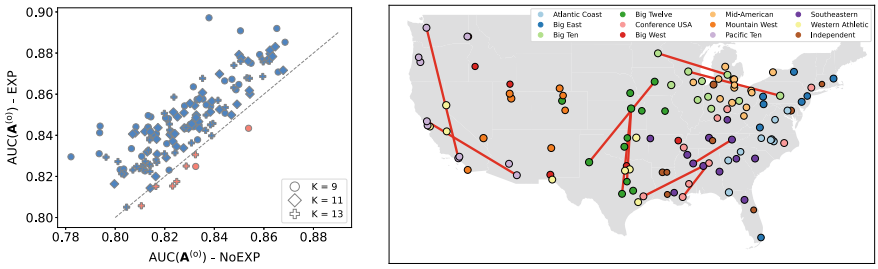


Fig. 4 **Left**: Performance in predicting missing links of the EXP and NOEXP algorithms on the ACFN dataset. Different marker shapes correspond to different numbers of communities, while blue (red) markers denote instances where EXP (NOEXP) has better performance than NOEXP (EXP). There is a total of 150 markers, denoting 5 folds repeated for 10 random seeds for each value of K. **Right**: Top 10 games that are recommended by the EXP algorithm with $K = 11$, which were not played in the AFCN data set. Different colors indicate different conferences

3.2 *Real Data*

To test our algorithm on real data, we use the American College Football Network (ACFN) dataset provided in Ref. [9], which represents the schedule of Division I games for the season of the year 2000. Each node in the data set corresponds to a team, and each link is a game played between teams. Teams are grouped in conferences, and each team plays most of its games within a same conference (though not all teams within a conference encounter each other). Conferences group teams of similar level, but another main criterion is geographic distance. Therefore, this dataset has a community structure which is not based on affinity. Here, affinity indicates that teams are of similar level, and therefore should play in the same conference, if conferences were based solely on affinity.

We randomly hide 20% of the links in the ACFN and check how well the EXP and NOEXP algorithms are able to reconstruct which links are missing. We run the algorithm with various number of communities $K = 9, 11, 13$, finding the best result at $K = 11$, which is also the number of conferences in the dataset. In Fig. 4—left we show a scatter plot of the AUC trial-by-trial. This reveals a superior performance of the EXP method which outperforms NOEXP in 142 out of 150 trials (5 folds per 10 random seeds for each of $K = 9, 11, 13$). This suggest that EXP is better capturing the data.

In Fig. 4—right we show the top 10 recommendations that we can extract from the EXP algorithm by taking, among the links missing from $\mathbf{A}^{(o)}$, those with smallest predicted exposure Q_{ij} and the highest affinity λ_{ij}. Although, in the absence of ground truth, we are not able to assess the validity of these suggestions, we note that all the suggested links represent unplayed games within the the same conference and that games within teams in different conferences were ranked lower.

4 Conclusions

In networks, nodes that would enjoy a high mutual affinity often appear disconnected for reasons that are independent of affinity. This is the case, for example, with people or entities in social networks that have never met, or due to some kind of sampling bias. This introduces a sampling bias in the datasets used for community detection. We studied this problem through a general framework, where we postulate that affinity in terms of compatibility of communities is not enough in order to explain the existence of a link, but rather a mechanism of exposure between nodes should be taken into account as well.

We proposed a principled probabilistic model, EXP, that takes into account this type of bias and is able to estimate the probability that two non-connected nodes are exposed while jointly learning what communities they belong to. We tested the EXP algorithm against a version of itself that does not account for exposure, NOEXP. On artificial data, where we could validate our results on ground truth parameters and unobserved ground truth data, we found that EXP is as good as NOEXP in learning communities, but it outperforms it when it comes to reconstructing missing links. In addition, the EXP approach allows us to satisfactorily infer which links remained unexposed, an estimate that cannot be done with standard method as, for example, NOEXP. We finally tested our algorithm on a real dataset which has a hidden structure that is independent of the affinity between links, finding that also here the EXP algorithm is better at reconstructing missing links.

The principled approach that we used based on statistical inference is general. It can be made more specific depending on the application at hand. For example, we considered the simple case where exposure only depends on each individual's propensity towards being exposed. However, this could depend on a more fine structure of society, and we could think of introducing an exposure mechanism that mimics the presence of communities which are independent of affinity (e.g. different schools, or different classes in a school). Allowing for community-dependent exposure has the potential to better mimic the kind of dilution that occurs in many real datasets. This can also apply to the AFCN dataset, where a better way to model exposure may be one that allows a structure that is able to account for different conferences or geographical regions. We leave this for future work. Additionally, exposure could be driven by covariate information on nodes, as also used in recommender systems [10]. This could be integrated using variants of community detection methods that account for this extra information [3, 6, 14]. Exposure could also change through time, and it could also have some dependence on the structure of $\mathbf{A}^{(g)}$. These are all interesting avenues for future work.

References

1. Ball, B., Karrer, B., Newman, M.E.: Efficient and principled method for detecting communities in networks. Phys. Rev. E **84**(3), 036103 (2011)
2. Chuklin, A., Markov, I., Rijke, M.d.: Click models for web search. Synthesis Lect. Inf. Concepts Retrieval Serv. **7**(3), 1–115 (2015)
3. Contisciani, M., Power, E.A., De Bacco, C.: Community detection with node attributes in multilayer networks. Sci. Rep. **10**(1), 1–16 (2020)
4. Contisciani, M., Safdari, H., De Bacco, C.: Community detection and reciprocity in networks by jointly modeling pairs of edges. J. Complex Netw. **10**(4), cnac034 (2022)
5. De Bacco, C., Power, E.A., Larremore, D.B., Moore, C.: Community detection, link prediction, and layer interdependence in multilayer networks. Phys. Rev. E **95**(4), 042317 (2017)
6. Fajardo-Fontiveros, O., Guimerà, R., Sales-Pardo, M.: Node metadata can produce predictability crossovers in network inference problems. Phys. Rev. X **12**(1), 011010 (2022)
7. Fortunato, S.: Community detection in graphs. Phys. Rep. **486**(3–5), 75–174 (2010)
8. Fortunato, S., Hric, D.: Community detection in networks: a user guide. Phys. Rep. **659**, 1–44 (2016)
9. Girvan, M., Newman, M.E.: Community structure in social and biological networks. Proc. Nat. Acad. Sci. **99**(12), 7821–7826 (2002)
10. Liang, D., Charlin, L., McInerney, J., Blei, D.M.: Modeling user exposure in recommendation. In: Proceedings of the 25th International Conference on World Wide Web, pp. 951–961. International World Wide Web Conferences Steering Committee (2016)
11. Liben-Nowell, D., Kleinberg, J.: The link-prediction problem for social networks. J. Am. Soc. Inf. Sci. Technol. **58**(7), 1019–1031 (2007)
12. Lü, L., Zhou, T.: Link prediction in complex networks: a survey. Phys. A Stat. Mech. Appl. **390**(6), 1150–1170 (2011)
13. Newman, M.: Networks. Oxford University Press (2018)
14. Newman, M.E., Clauset, A.: Structure and inference in annotated networks. Nat. Commun. **7**, 11863 (2016)
15. Safdari, H., Contisciani, M., De Bacco, C.: Generative model for reciprocity and community detection in networks. Phys. Rev. Res. **3**(2), 023209 (2021)
16. Safdari, H., Contisciani, M., De Bacco, C.: Reciprocity, community detection, and link prediction in dynamic networks. J. Phys. Complex. **3**(1), 015010 (2022)
17. Schein, A., Zhou, M., Blei, D., Wallach, H.: Bayesian Poisson tucker decomposition for learning the structure of international relations. In: International Conference on Machine Learning, pp. 2810–2819. PMLR (2016)
18. Wang, X., Bendersky, M., Metzler, D., Najork, M.: Learning to rank with selection bias in personal search. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 115–124 (2016)
19. Yang, L., Cui, Y., Xuan, Y., Wang, C., Belongie, S., Estrin, D.: Unbiased offline recommender evaluation for missing-not-at-random implicit feedback. In: Proceedings of the 12th ACM Conference on Recommender Systems, pp. 279–287 (2018)
20. Zhao, Y., Levina, E., Zhu, J.: Consistency of community detection in networks under degree-corrected stochastic block models. Annals Stat. **40**(4), 2266–2292 (2012)

Community Detection for Temporal Weighted Bipartite Networks

Omar F. Robledo, Matthijs Klepper, Edgar van Boven, and Huijuan Wang

Abstract Community detection of temporal (time-evolving) bipartite networks is challenging because it can be performed either on the temporal bipartite network, or on various projected networks, composed of only one type of nodes, via diverse community detection algorithms. In this paper, we aim to systematically design detection methods addressing both network choices and community detection algorithms, and to compare the community structures detected by different methods. We illustrate our methodology by using a telecommunications network as an example. We find that three methods proposed identify evident community structures: one is performed on each snapshot of the temporal network, and the other two, in temporal projections. We characterise the community structures detected by each method by an evaluation network in which the nodes are the services of the telecommunications network, and the weight of the links between them are the number of snapshots that both services were assigned to the same community. Analysing the evaluation networks of the three methods reveals the similarity and difference among these methods in identifying common node pairs or groups of nodes that often belong to the same community. We find that the two methods that are based on the same projected network identify consistent community structures, whereas the method based on the original temporal bipartite network complements this vision of the community structure. Moreover, we found a non-trivial number of node pairs that belong consistently to the same community in all the methods applied.

Keywords Community detection · Temporal networks · Bipartite networks

O. F. Robledo · E. van Boven · H. Wang (✉)
Delft University of Technology, Mekelweg 4, 2628 CD, Delft, The Netherlands
e-mail: H.Wang@tudelft.nl

M. Klepper · E. van Boven
KPN, Rotterdam, The Netherlands

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_19

1 Introduction

Networks [10] have been used to represent complex systems. In a network, nodes represent the elements of a system, and their interactions or relations are represented by links. Community detection has been a fundamental network characterisation method to discover communities of nodes where nodes within a community are more similar or more strongly connected, whereas two nodes from different communities are less similar or weakly connected.

The detection of disjoint communities has been broadly studied, especially for static networks [4–6, 12]. Modularity [7], defined by Newman and Girvan, is one classic quantification of the quality of a partition of network nodes into disjoint groups, among many other possibilities. A partition of network nodes that maximises the modularity is recognised as the community structure of the network, and the corresponding maximal modularity is called the modularity of the network. Algorithms to detect communities that optimise the modularity have been widely proposed and applied; e.g., the greedy techniques proposed by Newman [8], and Blondel et al. [3]. These algorithms do not require the number of communities as an input.

Many real-world networks evolve over time. In a physical (virtual) contact network, two individuals are connected only when there is a face-to-face (email) contact instead of constantly. Community detection algorithms for static networks could be applied to detect the communities at each snapshot of the temporal (time evolving) network independently. Algorithms have been further developed for temporal networks to enhance the stability of the community structure over time, especially between two consecutive time steps [13]. Many real-world networks are static bipartite networks, where the nodes can be divided in two disjoint sets (such as authors and papers), and links (authorship relations) can only connect nodes from different sets. Bipartite graphs have been projected to networks composed of only one set of nodes in various ways, and classic static network community detection algorithms can be applied to the projected networks. Moreover, the definition of modularity has been further updated for static bipartite networks [2]. Correspondingly, algorithms to detect communities in a static bipartite network that optimised the bipartite network modularity have been designed [14].

A challenging problem is the community detection of a temporal weighted bipartite network [11] (e.g., a telecommunications network that records the data transfer between services and base stations, over time). For such networks, communities can be detected by diverse combinations of the network (original network or projected ones) and community detection algorithms (to detect the community structure per snapshot independently, or stably overtime). Each detection method identifies the communities, with possibly a specific community definition. The foundational questions are two. First, how to systematically design detection methods that utilise existing network projection methods and community detection algorithms, and second, and most importantly, how to compare the community structures detected by different methods, so that we can have an integrated overview.

In this work, we develop methodologies to address these two questions, illustrated by using a telecommunications network as an example. We introduce a basic framework to design community detection methods that systematically consider diverse network or network projection choices, and, correspondingly, various community detection algorithms. Three of the proposed methods recognise relatively evident community structures, at least in a fraction of network snapshots. To compare the community structures identified by these algorithms, we propose to construct an evaluation network that characterises the evident community structures detected by a method. By analysing the evaluation networks of these three methods, we obtain insights regarding, e.g., when different methods are applied, whether the frequency that a node pair belong to the same community is consistent, and whether the group of nodes that frequently belong to the same community differ. Our work may shed light on how to utilise existing community detection and network projection algorithms to obtain a multi-perspective vision of the community structure(s) of a network.

This paper is organised as follows. In Sect. 2, we design community detection methods. In Sect. 3, we evaluate and compare the community structures found by these methods. Finally, we present our conclusions in Sect. 4.

2 Methods

In this section, we propose methods to detect the community structure of a temporal bipartite weighted network from different perspectives. We start by introducing the temporal bipartite network. Second, we propose methods to project a temporal bipartite network to one or multiple networks composed of only one type of nodes. Finally, we briefly review the community detection algorithms that will be applied to the temporal bipartite network and to the projected networks, respectively.

2.1 Weighted Temporal Bipartite Network

Static bipartite networks are a type of networks in which the nodes can be divided in two disjoint sets, $\mathbb{S}$, of size S, and $\mathbb{U}$, of size U, and links ($\mathcal{L}$) can only connect nodes from different sets. A weighted bipartite network can be represented by its biadjacency matrix R, an $S \times U$ rectangular matrix in which each element $R_{s,u}$ represents the weight between nodes s and u.

Take the data transference between services and base stations in a telecommunications network as an example. It could be represented as a temporal weighted bipartite network (see Fig. 1). A temporal bipartite network observed or measured at discrete time $\mathbb{T} = [1, 2, ..., T]$, and composed of a set $\mathbb{S}$ of S services and a set $\mathbb{U}$ of U base stations can be represented by a $S \times U \times T$ temporal biadjacency matrix

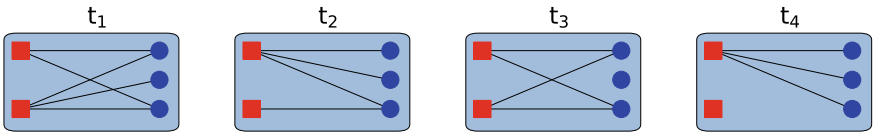


Fig. 1 Example of a temporal bipartite network at $T = 4$ time steps

Table 1 Basic properties of the bipartite telecommunications network

Number of services (S)	253
Number of base stations (U)	5166
Time window length (T) in steps	1440
Time window length in days	60
Time resolution per step	1 h

$\mathcal{R}$. Each element $\mathcal{R}_{s,u,t}$ represents the amount of data that has been transferred from service s to base station u at time t, where $s \in [1, S]$, $u \in [1, U]$ and $t \in [1, T]$. Basic properties of this telecommunications network can be found in Table 1.

2.2 *Projections of Weighted Temporal Bipartite Network*

Static bipartite networks have been projected to networks that contain only one type of nodes. The projected network, resulted from a given projection method, captures a specific relation among the same type of nodes. Such projection is motivated by the following. First, we might be interested in detecting communities within one type of nodes. Second, classic community detection methods can be further applied to a projected network. Projected networks are usually weighted networks, with the weights representing, e.g., a given kind of similarity between nodes. In this section, we will introduce diverse ways of projecting a temporal bipartite network, either per snapshot or as a whole, resulting in T projected networks or one projected network respectively. To illustrate our method, we project the temporal telecommunications network to networks among the services.

Static projection based on average cosine similarity. First, we explain a basic method that projects the temporal network as a whole to a static network of services. The volume of data transfer between a service i_1 and a base station j per step over time can be represented as a time series $\mathbf{w_{i_1,j}}$, where each element $\mathbf{w_{i_1,j}(t)} = \mathcal{R}_{\mathbf{i_1,j,t}}$ describes the volume of the data transfer between service i_1 and station j at time t. In this projection, the weight $\widehat{w}_{i_1,i_2}$ between two services i_1 and i_2 is the average cosine similarity between the two services' data transfer $\mathbf{w_{i_1,j}}$ and $\mathbf{w_{i_2,j}}$ with a base station j. Mathematically,

$$\widehat{w}_{i_1,i_2} = \frac{1}{U} \sum_{j \in \mathbb{U}_{i_1,i_2}} \frac{\mathbf{w_{i_1,j}} \cdot \mathbf{w_{i_2,j}}}{||\mathbf{w_{i_1,j}}||_2 \cdot ||\mathbf{w_{i_2,j}}||_2}, \tag{1}$$

where $\mathbb{U}_{i_1,i_2}$ represents the set of base stations that have transferred data from both nodes i_1 and i_2 at least once in $\mathbb{T}$. A large weight $\widehat{w}_{i_1,i_2}$ between two services implies that they are demanded by a base station over time in term of traffic in a similar way.

Temporal projection based on number of common neighbours. Temporal projection refers to methods that project each snapshot $\mathcal{G}(t)$ of the temporal bipartite network $\mathcal{G}$ to a network of the services. The first temporal projection method is defined as follows. In the projected network of $\mathcal{G}(t)$, two services are connected if they share any common neighbours in $\mathcal{G}(t)$, and the corresponding weight $\widehat{w}_{i_1,i_2}(t)$ is the number of common neighbours they have in $\mathcal{G}(t)$. That is,

$$\widehat{w}_{i_1,i_2}(t) = \sum_{j \in \mathbb{U}} \mathbf{1}_{\mathcal{W}(i_1,j,t)\mathcal{W}(i_2,j,t)>0}, \tag{2}$$

where the indicator function $1_{\mathcal{W}(i_1,j,t)\mathcal{W}(i_2,j,t)>0}$ equals one when the amount of data transfer $\mathcal{W}(i_1, j, t)$ and $\mathcal{W}(i_1, j, t)$ are both positive, or equivalently when j is a common neighbour for i_1 and i_2 at time t. A large weight $\widehat{w}_{i_1,i_2}(t)$ between two services indicates that both services have traffic with a large number of stations in common at time t.

Temporal projection based on the average geometric mean. At each time step t, we may wonder whether two services tend to have a large amount of data transfer with a common station, beyond their number of common neighbours. Hence, in the second temporal projection method, the weight $\widehat{w}_{i_1,i_2}(t)$ between two services projected from $\mathcal{G}(t)$ is defined as the geometric mean $\sqrt{w_{i_1,j}(t) \cdot w_{i_2,j}(t)}$ of their traffic with a common neighbour j, averaged over all common neighbours. Specifically,

$$\widehat{w}_{i_1,i_2}(t) = \frac{\sum_{j \in \mathbb{U}, \mathcal{W}(i_1,j,t)\mathcal{W}(i_2,j,t)>0} \sqrt{w_{i_1,j}(t) \cdot w_{i_2,j}(t)}}{\sum_{j \in \mathbb{U}} \mathbf{1}_{\mathcal{W}(i_1,j,t)\mathcal{W}(i_2,j,t)>0}}. \tag{3}$$

A large weight $\widehat{w}_{i_1,i_2}(t)$ between two services indicates that they tend to have a large amount of traffic with a station in common at time t.

2.3 Community Detection Methods

We adopt the concept of community and community detection algorithms that originated from modularity optimisation proposed by Newman [9] for networks of one type of nodes. We will illustrate how classic concepts and algorithms can be applied to detect the community structure of a weighted temporal bipartite network systematically.

2.3.1 Community Detection of Projected Networks

Classic community detection algorithms for static networks can be applied to the static projected network and the temporal projected network at each time step, to detect the communities of projected networks.

Consider first an undirected weighted network G that is composed of one type of nodes. It can be represented by a weighted adjacency matrix A. Given a weighted network and a partition of all the nodes into non-overlapping communities, the quality of this community partition can be measured by the modularity

$$Q = \frac{1}{2L} \sum_{i,j} \left[A_{i,j} - \frac{k_i \cdot k_j}{2L} \right] \delta_{c_i,c_j}, \tag{4}$$

where $k_i = \sum_j A_{i,j}$ is the sum of the weights of all the links connected to node i, so-called node strength; c_i is the label of the community to which node i belongs; the Kronecker delta $\delta_{c_i,c_j} = 1$, if $c_i = c_j$, and 0 otherwise; and $L = \frac{1}{2} \sum_{i,j} A_{i,j}$ is the total weight in the network.

The modularity of a partition describes the extent to which the weight of links within each community is bigger than the weight of those between communities. The modularity $Mod(G) \in [0, 1]$ of a network is the maximal modularity that could be obtained via community detection. Computing the modularity of a network is an NP-hard problem. We adopt the classic Louvain method [3] to obtain the approximate optimal modularity of a static network and its corresponding community partition.

The Louvain method [3]. This method starts with every node in its own community. For each node, it checks whether the modularity increases or not when changing its community to that of one of its neighbours. If there is an increase in modularity, then the community of that node is changed. This assignment step is repeated until there is no increase in modularity. The final community structure is considered as the optimal partition and the corresponding modularity is the modularity of the network. We will apply the Louvain method to detect the community structure of the static projected network and of the temporal projected network at each time t independently.

Stabilised Louvain method. To maintain the consistency of the community structures at two consecutive snapshots, we will also apply the stabilised Louvain method [1] to the temporal projected networks. Aynaud et al. modified the Louvain method, such that it considers the resulting community partition from the previous snapshot as the initialisation, whereas the modularity optimisation procedure remains the same.

2.3.2 Community Detection of a Temporal Bipartite Network

In the previous section, we have shown how to detect the communities of a temporal bipartite network by applying classic community detection methods to its projected

Table 2 Summary of the community detection methods proposed that combine the network and community detection algorithm differently. The methods that find evident community structures are highlighted in bold

Network	CD algorithm	Method name
Bipartite network	*Bi-Louvain*	*BiLouvain*
Cosine similarity static projection	Louvain	CS-Louvain
Common neighbours temporal projection	Louvain	CN-Louvain
	Stabilised Louvain	CN-stabilised
Geometric mean temporal projection	*Louvain*	*Geometric-Louvain*
	Stabilised Louvain	*Geometric-Stabilised*

networks. However, we can also apply a community detection algorithm for static bipartite networks to each snapshot $\mathcal{G}(t)$ of the temporal bipartite network.

The modularity definition for a static bipartite weighted network has been adapted by Barber [2] by redefining the null model to which we compare the weights within each community. We can express it as

$$Q = \frac{1}{L} \sum_{i=1}^{S} \sum_{j=1}^{U} \left[R_{i,j} - \frac{k_i \cdot d_j}{L} \right] \delta_{c_i, c_j}, \tag{5}$$

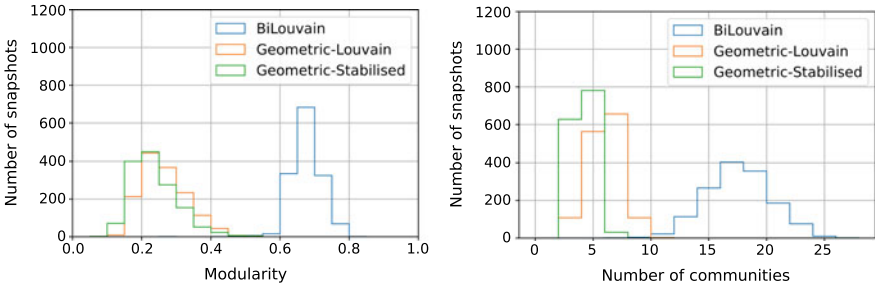
which considers the random weighted bipartite network with the same node strength as the given bipartite network as the null model.

The Bi-Louvain method [14]. Zhou et al. have proposed this community detection algorithm for static bipartite networks based on the Louvain method and modularity definition (5).

In summary, combinations of the aforementioned network choices, projected or not, and community detection algorithms lead to in total six community detection methods, as shown in Table 2.

3 Results

In this section, we evaluate the communities of services detected by the methods that we have proposed. First, we study to what extent the community structures found are evident through their modularity. Second, we investigate how the evident community structures (partition of services) detected by diverse methods provide a complementary or consistent vision.



(a) Modularity. (b) Number of communities.

Fig. 2 Representation of the **a** modularity and **b** number of communities of each of the methods

3.1 Modularity

Only the BiLouvain, Geometric-Louvain and Geometric-Stabilised methods have found evident community structure, i.e., the modularity is higher than 0.3 in, at least, a portion of the snapshots. Hence, the other methods will not be discussed further. Each of the three considered methods, partitions the nodes (services) into communities for each snapshot of the bipartite temporal network, or of the geometric mean temporal projection. In Fig. 2, we show the distribution of the modularity in a snapshot. The BiLouvain method shows the largest modularity of the three methods. For each of the three methods, we will further analyse the community structures in snapshots when the corresponding modularity is larger than 0.3.

3.2 Community Structure Comparison

We aim to compare the evident community structures found by these methods. In order to do that, we define an evaluation network to characterise the evident community partitions detected by a method.

3.2.1 Evaluation Network

An evaluation network contains the set $\mathbb{S}$ of services as nodes, and is constructed based on the community structures detected by a given method in every snapshot. Two nodes are connected if they have been assigned to the same community in, at least, one snapshot in which the modularity is larger than 0.3. The weight of the link is the total number of snapshots in which both nodes belong to the same community and the modularity is larger than 0.3. We build a weighted static evaluation network for each of the three methods. The weight distributions of the three evaluation networks are shown in Fig. 3. The average link weight in BiLouvain evaluation network

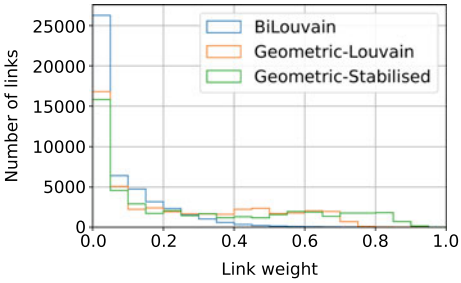


Fig. 3 Link weight distribution in each evaluation network

is evidently smaller than that in Geometric-Louvain and Geometric-Stabilised evaluation networks. This could be due to the larger number of communities detected by BiLouvain. This could also imply that the community structure detected by BiLouvain changes more significantly over time. The average link weight in the Geometric-Stabilised evaluation network is slightly larger than that in Geometric-Louvain evaluation network, supporting that Stabilised Louvain detects more stable community structure over time than Louvain.

3.2.2 Recognition Rate

First, we aim to understand whether two nodes that more frequently belong to the same community according to one method, or, equivalently, have a high weight in the corresponding evaluation network, also tend to belong to the same community more often according to another method. This is evaluated via the recognition rate between two methods, defined as follows. We rank the links in each evaluation network according to their weights. The set of fL links, with f being the ratio of links considered, with the highest link weights in the evaluation network derived from, e.g., the BiLouvain (Geometric-Louvain) method can be represented as J_f^{BL} (J_f^{GL}), where $L = \binom{S}{2}$ is the maximal possible number of links among S services, and $f \in [0, 1]$. The top f fraction recognition rate between, e.g., (the evaluation networks of) BiLouvain and Geometric-Louvain methods is defined as $r_{BL,GL}(f) = \frac{|J_f^{BL} \cap J_f^{GL}|}{|J_f^{BL}|}$, which measures the number of links in common between the two sets J_f^{BL} and J_f^{GL} normalised by the number of links fL in each set.

The link densities of the evaluation networks are all slightly above 0.7. Therefore, we compute the recognition rate for $f \in (0, 0.7]$. The top f recognition rate between random ranking of links and any ranking of links is f. As we can see in Fig. 4, the top f recognition rate between any two community detection method is higher than f, suggesting that all the evaluation networks share similarity in identifying similar set of links with a large weight. Moreover, the co-occurrence between the top links in Geometric-Louvain and Geometric-Stabilised is the highest. This is in line with the fact that Geometric-Louvain and Geometric-Stabilised use the same network

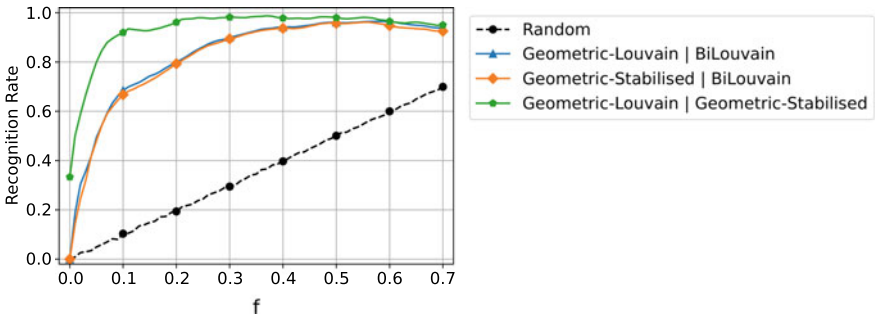


Fig. 4 Top f recognition rate between the proposed methods

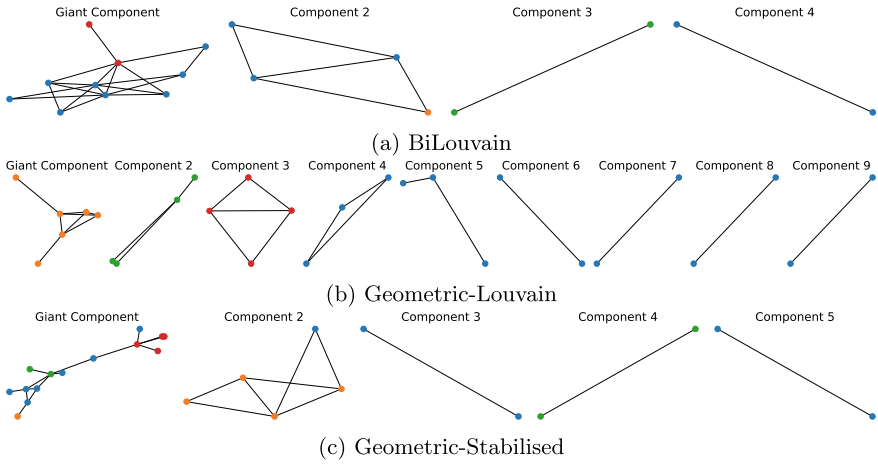


Fig. 5 Sub-evaluation network composed of 50 links with the largest weights derived by **a** BiLouvain, **b** geometric-Louvain and **c** geometric-stabilised respectively. All nodes are coloured in blue except those in the largest, second largest, and third largest component in the geometric-Louvain sub-evaluation network, which are coloured in orange, green and red, respectively

projection for community detection. The visualisation of the sub-evaluation network composed of the top 50 links with the largest link weight derived by each method in Fig. 5 reveals the same. For example, nodes in the largest, second largest, and third largest components of the Geometric-Louvain sub-evaluation network (coloured in orange, green and red respectively) are more likely to appear, or to be connected, in the Geometric-Stabilised sub-evaluation network in comparison to BiLouvain.

3.2.3 Persistent Community Component

Besides the similarity of two evaluation networks in identifying links with large weight, measured by the recognition rate, we explore further the similarity between

Table 3 Number of snapshots in which all the nodes in the indicated component (or clique) in the sub-evaluation network with 50 links belong to the same community

Method	BiLouvain	Geometric-Louvain	Geometric-stabilised
Giant component	11	174	95
Second largest component	124	372	336
Largest clique within giant component	141	311	472

two methods in identified groups of nodes that frequently belong to the same community, so-called persistent community component. Finding the persistent groups of size m requires the counting of the number of snapshots in which each of the $\binom{S}{m}$ groups belong to the same community, according to a given community detection method. Its computational complexity is high and it is difficult to be simplified when the community structure changes over time.

Identifying whether components in the aforementioned sub-evaluation network, composed of links with the highest weight, are persistent, could be an intuitive and insightful start. The motivation is that a group of nodes may frequently belong to the same community if pairs of them often belong to the same community.

The number of snapshots in which all the nodes in the largest (second largest) component of a sub-evaluation network fall into the same community is shown in Table 3. We find that nodes in the largest component of the BiLouvain sub-evaluation network belong to the same community less frequently compared to that of other sub-evaluation networks, although the largest component of the BiLouvain sub-evaluation network is denser. This difference in frequency is evident, especially in view that the total number of snapshots that have a modularity larger than 0.3 is far larger when BiLouvain is applied. For the Geometric-Louvain and Geometric-Stabilised methods, nodes in the biggest component and, especially, in the second biggest component belong to the same community in up to almost a quarter of the snapshots that have an evident community structure. The same observation holds when examining whether nodes in the second largest component and the largest clique within each giant component are persistent community components. This difference could be due to the lower average link weight in the BiLouvain sub-evaluation network, and the highly dynamic community structure detected by BiLouvain over time.

We find that each component in Fig. 5 tends to be persistent and composed of a specific type of services, e.g., related to social networks or provided by the same brand. The biggest component of the Geometric-Stabilised method, though persistent, is an exception, containing various types of services.

4 Conclusions

In this paper, we define multiple methods to detect community structures of a temporal weighted bipartite network. We study how the partitions found by different community detection methods align or complement each other, illustrated via a telecommunications network. The three community detection methods that find evident community structures are performed either on the original bipartite temporal network or on a temporal projection; i.e., projecting each temporal network snapshot independently. To compare them beyond their difference in community definition, we define an evaluation network to characterise the community structures found by each method, in which the nodes are the services of the telecommunications network, and the weight of the links between them is the number of snapshots in which both services belong to the same community. Then, we compare which nodes are the ones that are most commonly clustered together, first in terms of node pairs through the recognition rate, and then in terms of groups of nodes by studying the components of the sub-evaluation network with the highest-weight links. The two methods that partition the network based on the same temporal projection, using Louvain and stabilised Louvain, respectively, identify consistent community structures, whereas the third method, based on the original temporal bipartite network, provides a complementary perspective of the community structure. Moreover, we find that all three methods share a non-trivial number of common node-pairs that are often in the same community.

Our methodology, exemplified by a limited choice of candidate algorithms and one network, is the starting point to explore the multi-perspective vision of the community structure of a temporal bipartite network. It could be further improved by investigating, e.g., the time series associated to each link of an evaluation network that records the time stamps when two nodes belong to the same community, and networks with known ground truth community structure.

Acknowledgements We thank NExTWORKx, a collaboration between TU Delft and KPN on future telecommunication networks, for the support.

References

1. Aynaud, T., Guillaume, J.: Static community detection algorithms for evolving networks. In: 8th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, pp. 513–519 (2010)
2. Barber, M.: Modularity and community detection in bipartite networks. Phys. Rev. E. **76**, 066102 (2007)
3. Blondel, V., Guillaume, J., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. Theory Exp., P10008 (2008)
4. Delvenne, J., Yaliraki, S., Barahona, M.: Stability of graph communities across time scales. Proc. Nat. Acad. Sci. **107**, 12755–12760 (2010)
5. Fortunato, S.: Community detection in graphs. Phys. Rep. **486**, 75–174 (2010)

6. Ge, X., Wang, H.: Community overlays upon real-world complex networks. Eur. Phys. J. B **85**, 26 (2012)
7. Newman, M., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E. **69**, 026113 (2004)
8. Newman, M.: Fast algorithm for detecting community structure in networks. Phys. Rev. E. **69**, 066133 (2004)
9. Newman, M.: Modularity and community structure in networks. Proc. Nat. Acad. Sci. **103**, 8577–8582 (2006)
10. Newman, M.: Networks: An Introduction. Oxford University Press (2010)
11. Peters, L., Cai, J., Wang, H.: Characterizing temporal bipartite networks—sequential- versus cross-tasking. Complex Netw. Appl. VII 28–9 (2019)
12. Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., Parisi, D.: Defining and identifying communities in networks. Proc. Nat. Acad. Sci. **101**, 2658–2663 (2004)
13. Rossetti, G., Cazabet, R.: Community discovery in dynamic networks: a survey. ACM Comput. Surv. **51** (2018)
14. Zhou, C., Feng, L., Zhao, Q.: A novel community detection method in bipartite networks. Phys. A Stat. Mech. Appl. **492**, 1679–1693 (2018)

Robustness and Sensitivity of Network-Based Topic Detection

Carla Galluccio, Matteo Magnani, Davide Vega, Giancarlo Ragozini, and Alessandra Petrucci

Abstract In the context of textual analysis, network-based procedures for topic detection are gaining attention as an alternative to classical topic models. Network-based procedures are based on the idea that documents can be represented as word co-occurrence networks, where topics are defined as groups of strongly connected words. Although many works have used network-based procedures for topic detection, there is a lack of systematic analysis of how different design choices, such as the building of the word co-occurrence matrix and the selection of the community detection algorithm, affect the final results in terms of detected topics. In this work, we present the results obtained by analysing a widely used corpus of news articles, showing how and to what extent the choices made during the design phase affect the results.

Keywords Text network analysis · Community detection · Topic detection

1 Introduction

The need to gather information from large textual datasets has led to the development of automated information extraction methods [12, 18]. Among these methods, those aimed at identifying topics have become very popular in machine learning and natural language processing [1].

Recently, network-based procedures have gained attention in the context of textual analysis as an alternative to classical topic models for detecting topics in large collections of documents [9]. These methods are based on the idea that any text can be represented as a word co-occurrence network, where topics emerge as groups of strongly connected words. In addition, the network can be used to explore and

C. Galluccio (✉) · A. Petrucci
University of Florence, Florence, FI 50134, Italy
e-mail: carla.galluccio@unifi.it

M. Magnani · D. Vega
InfoLab, Uppsala University, 752 37, Uppsala, Sweden

G. Ragozini
University of Naples Federico II, Naples, NA 80133, Italy

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_20

present the relations between the topics. Although many works have used network-based procedures for detecting topics in textual data, there is a lack of systematic analysis of how different design choices affect the final results in terms of detected topics.

Essentially, a network-based topic discovery process takes the following form:

- pre-processing the text, a step-by-step procedure during which the researcher selects which methods to apply to clean the text and make it ready for the analysis (e.g. removal of non-alphanumeric characters, removal of stopwords, reduction of terms to a common root);
- forming of the word co-occurrence matrix by defining the context in which two words will be considered semantically related. This is usually done by defining what is meant by "co-occurrence" between words;
- building of the network and selection of the community detection algorithm.

This procedure requires the researcher to make decisions in each of these steps.

In this work, we focus on the two defining steps of this process, as they are unique to network-based approaches: building the word co-occurrence matrix and selecting the community detection algorithm. From our point of view, the definition of the word co-occurrence matrix, which determines the shape of the network, and the community detection algorithm employed are strongly related to the characteristics of the discovered topics. Moreover, the impact of other design choices on text classification has already been studied in a non-network context. For instance, Uysal and Gunal have investigated the impact of text pre-preprocessing on text classification, revealing that choosing an appropriate combination of pre-processing steps may improve the classification accuracy [17].

As an example, Fig. 1 shows four different networks built using the same documents. They represent the word co-occurrence matrices of 9 news extracted from the BBC news articles collection [8] concerning business, sport, and tech. More specifically, in the first (Fig. 1a) and the third (Fig. 1c) networks two words belonging to the same document are adjacent, or co-occur, if they are at most 2 words apart (that is, if between the two words there is at most one word in between). On the other hand, the second (Fig. 1b) and the fourth (Fig. 1d) networks have been built considering that two words in the same document co-occur if they are at most 10 words apart. Furthermore, in order to identify the topics, we applied the Louvain community detection algorithm [4] on the first and the second networks (Fig. 1a, b), while on the other two networks we applied Newman's leading eigenvector method for detecting communities [13]. It is possible to observe how the shape of the networks and the detected communities change. For example, we can observe more defined communities in the networks with a window size equal to 10, some communities recognised by one method are split into two by the other, and some nodes are assigned to a different community.

Analysing the effect of the relevant design choices on the final results allows us to identify the fundamental aspects that should be taken into account when using network-based procedures to analyse textual data and discover topics, and those which may require further research. Therefore, the main contribution of this work

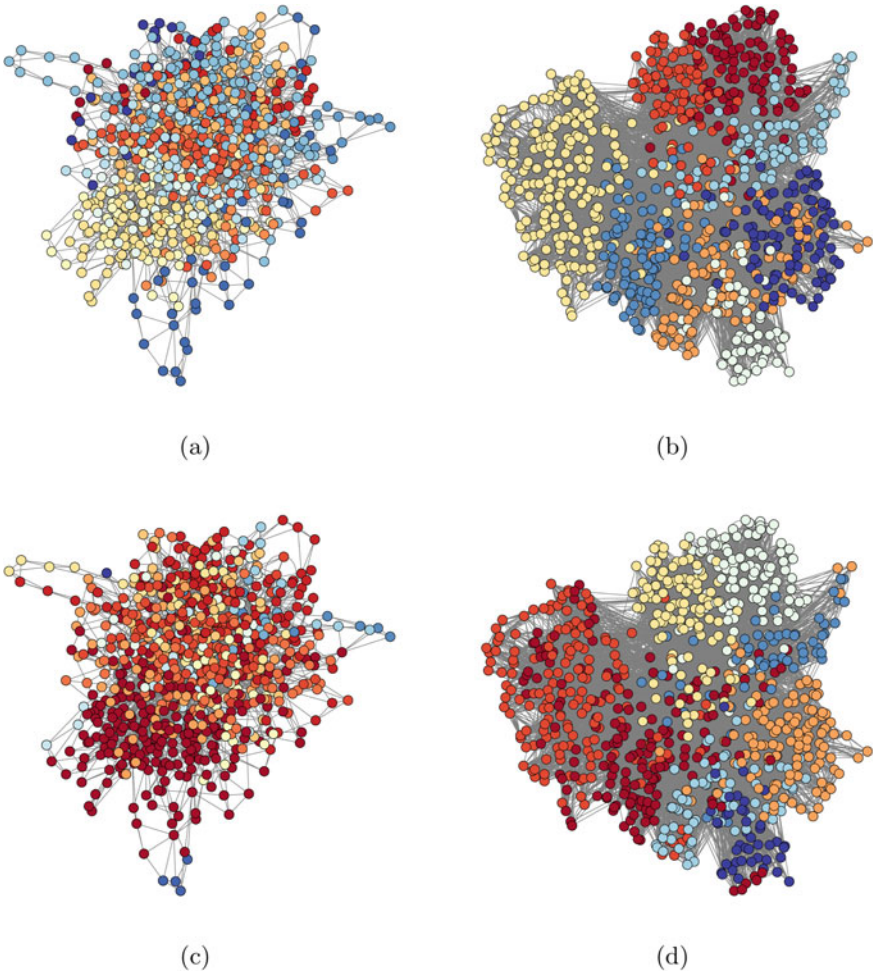


Fig. 1 Example of networks obtained from 9 news of the BBC news article collection. In networks **a** and **c** the window size is equal to 2, while it is equal to 10 in networks **b** and **d**. The colours represent the community to which each node belongs according to community detection algorithms: the Louvain algorithm in (**a**) and (**b**) and Newman's leading eigenvector method in (**c**) and (**d**). Note that the organization of nodes in communities varies between networks. Indeed, while in (**b**) and (**d**) the organization in communities is clear, in (**a**) and (**c**) the partition is much less defined

is to evaluate the relationship between the shape of the network, which changes depending on the word co-occurrence matrix, the community detection algorithm employed, and the features of the discovered topics.

Another unexplored question about network-based topic detection is about its relationship with probabilistic topic models, such as Latent Dirichlet Allocation (LDA) [3]. While this question is also important, before addressing it we need to develop a

deeper understanding of optimal design choices for network-based methods. Therefore, this paper is a first step towards enabling a comparison between these different approaches.

2 State of the Art

In recent years, many works have been written about applying community detection methods for topic discovery.

For example, Sayyadi and Raschid find topics as communities in a keyword co-occurrence matrix using the Girvan-Newman community detection algorithm based on the betweenness centrality measure [16]. They build the keyword co-occurrence matrix considering that two keywords are connected if they co-occur in at least one document, and the weight of that link is given by the number of documents in which both keywords co-occur. Then, they compute each word's document frequency and remove the links with a value below a specific threshold.

Another example is given by Salerno et al., who apply the Louvain community detection algorithm for discovering topics on a weighted network in which nodes represent individual words in the vocabulary and links indicate the co-occurrence of a pair of words within a document [15]. The weight of the links between words is determined by the context in which two words co-occur: for example, a co-occurrence within the same sentence carries more weight than a co-occurrence within the same paragraph. Then, they evaluate their results using modularity and comparing the error rate to the results achieved by two baselines: one that classifies documents randomly and another one that classifies documents based on the most common label in the training set. Similar approaches can be found in Dang and Nguyen [6].

Instead, de Arruda et al. investigate how specific definitions of the co-occurrence between words favour the emergence of communities of semantically related words, allowing for the identification of relevant topics [7]. In particular, they consider three different ways to define the co-occurrence between two words in the pre-processed text: two words are connected if they are separated by at most a given number of other words; words belonging to the same paragraph are linked together in a clique, disregarding links between words further from each other than the given maximum distance; finally, the statistical significance of co-occurrences with regard to random, shuffled texts is tested. The fast-greedy method is used to find communities of high modularity.

Lancichinetti et al. discover topics using the Infomap algorithm on networks built considering that two words are connected if they co-occur in the same document [12]. More specifically, they compute the dot product similarity of each pair of words that co-occur in at least one document in order to compare it against the expectation for a null model where words are randomly shuffled across documents. Then, a threshold is defined for retaining words for which the co-occurrence between them cannot be explained by the null model. However, because Informap is run as a non-overlapping community detection algorithm, to cope with generic words used in multiple topics,

they refine the results obtained from applying the community detection algorithm using a latent topic model that allows for non exclusivity.

Some of the most recent contributions in this area are given by Kim and Sayama [11] and Hamm and Odrowski [9]. The former transform the textual data into a vector form by computing the *tf-idf* (term frequency inverse document frequency) score considering each sentence as a document. Afterwards, they compute the pair-wise cosine similarity of the *tf-idf* vectors to build adjacency matrices of the sentences, and then they use the Louvain community detection algorithm on the sentence networks, where the nodes are the sentences, and the cosine similarity of *tf-idf* representations between every node pair represents the link weight. Hamm and Odrowski apply the Leiden community detection algorithm on undirected weighted networks investigating the effects of the resolution parameter on modularity maximisation [9]. Moreover, they define a measure to identify the most significant words within a topic.

This work contributes to this research line by considering the relationship between the definition of the word co-occurrence matrix, the selection of the community detection algorithms, and the final results.

3 Method and Material

In this section, we describe the data and the tested design choices.

Data. For the analysis, we used the corpus of BBC news articles, a collection of documents widely used as a benchmark for machine learning research [8]. The collection is composed of 2,225 complete news articles collected from 2004 to 2005 and divided into five topics: business, entertainment, politics, sport, and tech. The total number of articles and unique words per topic is reported in Table 1. We considered both the headline and the body of each news in the analysis.

Data pre-processing. We removed non-alphanumeric characters, numbers, and words composed of 1 or 2 characters. Afterwards, we divided the text into tokens, choosing single words (*uni-grams*) as unit of analysis. Then, we removed the stopwords using a list provided with the dataset, and stemmed the text in order to reduce the size of the vocabulary, that is the set of unique words used in the text corpus.

Table 1 Number of documents and unique words for each topic of the BBC collection

Topics	Documents	Unique words
Business	510	10,790
Entertainment	386	11,040
Politics	417	10,636
Sport	511	9,997
Tech	401	11,444

Finally, to remove very common words not included in the stopword list, we filtered out words with a value of *tf-idf* less than 0.01 [2]. After the pre-processing stage, the number of unique word tokens was equal to 18,422.

Word co-occurrence matrix. Once we pre-processed the corpus and obtained the vocabulary, we built the word co-occurrence matrices. To generate the word co-occurrence matrices we counted the number of times two words co-occur in the same document within a specific window size.

There are three ways of positioning the window: to the left of the word, to the right, or on either side [5]. Herein, we considered windows of different sizes placed to the right of the words, as usually done in the literature. More specifically, in this work we have considered window sizes equal to 2, 5, 10, 15 and 20.

Furthermore, in the literature many authors apply different filters to the word co-occurrence matrix based on the distribution of the words or their frequency in order to reduce the size of the matrix. For this reason, we decided to test this aspect by using different filters for the word co-occurrence matrices. More specifically, we removed the 100, 500, and 1000 words with the lowest co-occurrence values and the 50, 100, and 500 words with the highest co-occurrence values. We also filtered words with the highest or lowest co-occurrence values considering specific percentages of the total, but the results were similar to those obtained in the first two cases, so we do not report them here.

Afterwards, inspired by Salerno et al., who applied different weights based on the context in which two words co-occur [15], we defined an experimental condition by modifying the co-occurrence values assigned to words within the window size. In particular, we assigned weights proportional to the words' proximity. For example, for a window size equal to 3, the word adjacent to the target word gets a value equal to 1; the next word takes a value equal to 2/3; then, we assign a value equal to 1/3 to the last word.

Network and community detection algorithm. Starting from the word co-occurrence matrices, interpreted as weighted adjacency matrices, we built the undirected weighted networks on which we applied three different community detection algorithms.

Since almost all the works reported in Sect. 2 applied modularity optimisation algorithms, we decided to use the Louvain community detection algorithm as one of the most popular among them. Then, to investigate the performance of a different kind of approach we employed a spectral algorithm, namely Newman's leading eigenvector method. The rationale behind this choice is that if the network obtained after the pre-processing phase presents clearly separated topics, different algorithms should find similar results, while for networks with a less clear community structure the specific types of community that each different method is designed to identify would potentially lead to significantly different results.

Finally, we argue that despite the absence of methods finding overlapping communities in the literature on network-based topic detection, in theory these methods are the most appropriate. In general, we cannot exclude that a word belongs to multiple topics at the same time, but using a partitioning method prevents the identification

of such cases. As a consequence, we also tested the SLPA algorithm as a method designed to discover overlapping community [19].

4 Results and Discussion

In this section, we present the results of our experiments, focusing on how the different choices we made in the definition of the word co-occurrence matrix and the selection of the community detection algorithm affect the features of the detected topics.

4.1 The Effect of the Window Size

The main result we observe is that the number of communities obtained by the three algorithms is generally higher for smaller window sizes. Indeed, as the window size increases, the number of communities the algorithms find decreases, remaining constant for a window size greater than 5.

Figure 2 shows the number of communities found applying the three algorithms on the word co-occurrence matrices without filters: here, the number of communities identified by the non-overlapping community detection algorithms, that is, the Louvain and Newman's leading eigenvector methods, is always greater than the number of communities identified by SLPA for window sizes greater than 2. In particular, SLPA finds only one community with these settings.

4.2 Filters on the Word Co-occurrence Matrix

The results remain stable when we remove the words with the lowest co-occurrence values from the word co-occurrence matrix. Instead, removing the words with the highest co-occurrence values changes the number of detected communities only for a window size equal to 2: the Louvain community detection algorithm found 47 communities, Newman's algorithm found 27 communities, while the SLPA found 112 communities. The results for window sizes greater than 2 remain stable.

4.3 Weighting Scheme

Finally, we assessed the effect of using a different weighting scheme within the window sizes. We evaluated this aspect in the condition without any filters on the word co-occurrence matrix. In this case, results were significantly different from those obtained in the other experimental conditions for the Louvain and the SLPA

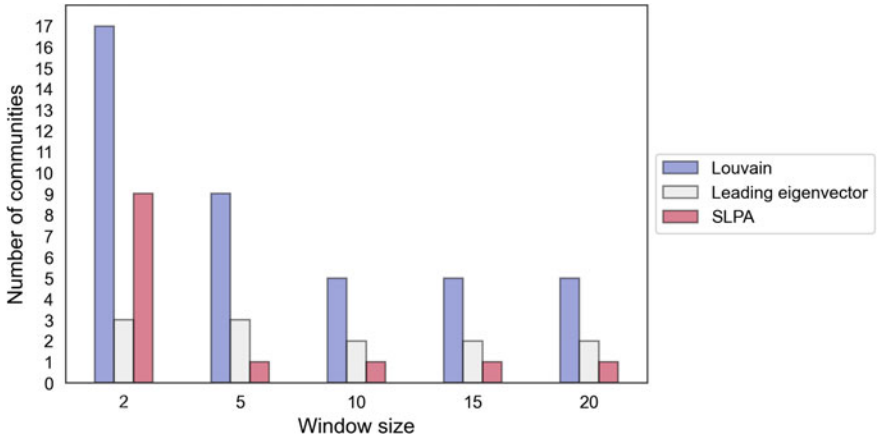


Fig. 2 Number of communities per window size and community detection algorithm. Observe that the number of communities decreases as the window size increases

community detection algorithms, with a number of communities ranging from 10 to 51 for the former and from 30 to 179 for the latter. However, also in this case the number of communities decreases when we increase the window size.

4.4 Selection of the Community Detection Algorithm

Regarding the community detection algorithm, the Louvain algorithm showed the most interesting results. In almost all the experimental conditions, this algorithm found a number of communities equal to the number of the actual topics in the document collection for window sizes greater than 5. Moreover, as shown in Fig. 3a–c, the communities are coherent with the content of the actual topics in the BBC document collection, with each community representing mainly one topic.

Note that Fig. 3 was built by matching the communities' words with the actual topics' words, enabling possible overlapping. Therefore, in the representation of the correspondence between communities' words and topics' words, generic words such as "month" or "show" could be included in more than one topic.

To better understand these results, take as an example the communities found by the Louvain community detection algorithm for a window size equal to 10 (Fig. 3a). First, the size of communities is quite balanced, with a number of words ranging from 3162 to 4283. Then, from an inspection of the words with the highest node degree within each community, we observed that they are coherent with the topic they represent. So, for example, among the top 15 words with the highest node degree in the first community there are words such as "show", "film", "record", "star", and

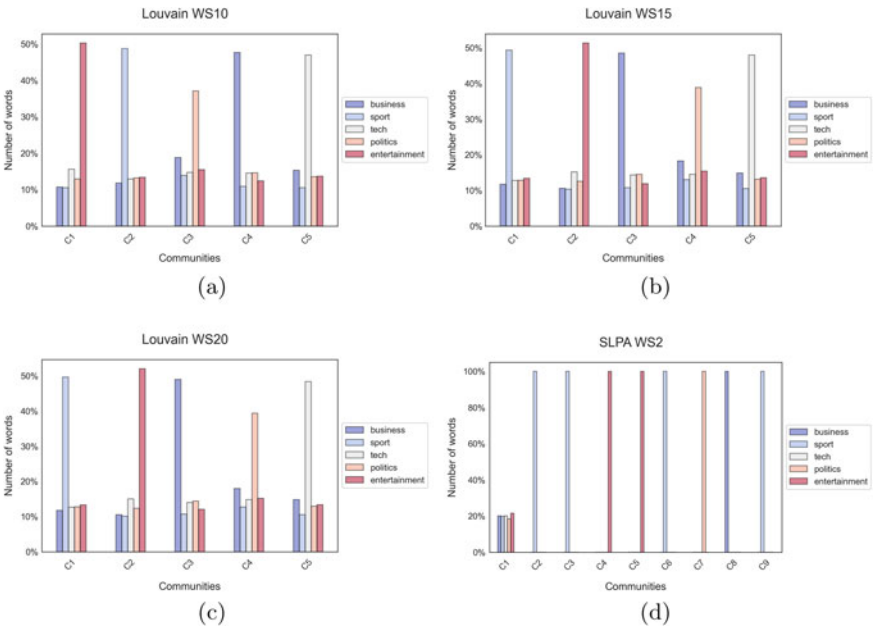


Fig. 3 Matching of the communities' words to actual topics' words for the Louvain community detection algorithm for window sizes equal to 10 (**a**), 15 (**b**), and 20 (**c**), and for the SLPA algorithm for window size equal to 2 (**d**). In this case, no filter was applied to the word co-occurrence matrix. On the y axis we reported the percentage of words in each community belonging to each actual topic (every group of bars sum up to 100%). "C" means "community", while "WS" means "window size"

"music", coherent with the topic "entertainment". We observed the same for window sizes equal to 15 and 20.

Instead, in the cases in which the Louvain algorithm finds more than 5 communities, namely for window sizes equal to 2 and 5, we observed that there are always 5 bigger communities coherent with the original topics and a variable number of smaller communities. Moreover, the largest communities generally include a number of words greater than 2000, whereas the smallest are composed of hundreds, tens, or just a few words.

To provide a more detailed analysis of the communities identified by the Louvain algorithm under different settings, we computed the Adjusted Rand Index (ARI) [10], a metric for comparing disjoint clustering solutions. Table 2 shows the ARI for different window sizes. Observe that the ARI is generally high, particularly between the partitions obtained considering window sizes greater than 5. More specifically, for window sizes greater than 5, ARI values range from 0.604 to 0.878, showing high similarities, but also that the algorithm finds the same number of communities but the communities are not identical.

Table 2 Value of the ARI computed between all the partitions obtained by the Louvain community detection algorithm applied to networks built from the different word co-occurrence matrices

	WS2	WS5	WS10	WS15	WS20
WS2	1				
WS5	0.348	1			
WS10	0.289	0.651	1		
WS15	0.279	0.624	0.828	1	
WS20	0.277	0.604	0.793	0.878	1

Here, "WS" means "window size"

The lowest ARI values are associated to the partitions obtained using smaller window sizes, requiring an additional analysis to show how these communities relate to those found with larger window sizes. Therefore, we computed the contingency table between the partitions obtained with window sizes equal to 5 and 10, respectively, to better understand the tendency of the algorithm to merge communities related to the same topic by increasing the window size. The table is not reported here for space reasons, but it shows that some of (but not all) the clusters obtained using a window size equal to 5 are assimilated into some of the larger clusters found in the partition obtained using a window size equal to 10.

The two other algorithms failed to find a reasonable number of communities, with the SLPA algorithm finding only one community for window sizes greater than 2 in all the experiments. Even in those cases where SLPA finds more than one community, the communities are not balanced, with almost all the words within one of the detected communities. Figure 3d shows the results we obtained applying the SLPA algorithm on the word co-occurrence matrix without filters using a window size equal to 2. Note that in the first community there are 18,402 words, while in the others the number of words ranges from 1 to 5. As an overlapping community detection algorithm, we also tried to use the K-clique algorithm [14] with different values for the k parameter, but we did not manage to obtain results because of the presence of large dense subgraphs, making this approach computationally intractable.

5 Conclusions

In this work we assess the effect of different design choices in network-based procedures for topic detection. In particular, we tested different ways of building the word co-occurrence matrix found in the literature and the selection of different community detection algorithms.

Our findings show that, for all tested algorithms, increasing the window size initially decreases the number of communities, which becomes stable for window sizes equal to or greater than 5 depending on the algorithm. This suggests that some of

topics identified in the literature may have been influenced by this design choice, and leads to the consideration that the window size should be regarded as an important hyperparameter in future studies.

In addition, considering the number of detected topics applying different filters on the word co-occurrence matrix, we observe that the Louvain community detection algorithm generally performs better than the other tested algorithms. Indeed, considering the information available on the actual number of topics in the BBC document collection, the Louvain algorithm always detects the correct number of topics for a window size greater than 5, whereas the other two algorithms fail. This does not lead to a rejection of our hypothesis that overlapping community detection methods are more appropriate to find topics in word co-occurrence networks: it is still possible that the Louvain algorithm could correctly cluster together words belonging to a single topic, while arbitrarily including multi-topic words in only one of the communities where they should have been included. However, we can conclude that some of the typical overlapping community detection methods are not able to identify significant topics under the experimental settings tested in this paper. The fact that these settings are taken from the literature suggests that more research should be done to identify pre-processing schemes leading to networks better suited to the application of these methods. One feature of the networks obtained in our experiments that may have determined the poor results of the tested methods is their high density, suggesting that stronger filtering schemes should be considered.

Finally, regarding the weighting scheme, our results show that while weighting the links can significantly affect the results, finding a good setting is not straightforward, with the number of communities suddenly becoming very high after imposing the basic scheme considered in this paper. This shows that this aspect should be analysed in more depth, also testing different combinations of pre-processing steps to select the words and to define co-occurrence weights and values.

In summary, on the one hand our preliminary results confirm what is stated in the literature, where network-based procedures for topic discovery show promising results; on the other hand, they highlight how different design choices, such as choosing specific algorithms or window sizes, applying filters on the word co-occurrence matrix, or defining different weighting schemes, may significantly affect the results in terms of detected topics.

Most importantly, this study highlights a number of aspects deserving additional attention. First, as further developments, we plan to extend our study considering additional community detection algorithms, to evaluate which methods are appropriate depending on the applied pre-processing steps. Second, additional ways to define the word co-occurrence matrix should also be studied, to enable the application of a broader range of algorithms and consequently the discovery of different types of communities. Third, we plan to define additional measures aimed at evaluating the quality of the detected topics, going beyond the basic measure of word overlapping used in this paper. Finally, we aim to assess the effects of these design choices on different kinds of texts. For example, we can expect different window sizes to be relevant for shorter documents, such as social media posts, and different vocabulary sizes to lead to networks with different sizes and densities.

Acknowledgements The authors acknowledge the financial support provided by the "Dipartimenti Eccellenti 2018–2022" ministerial funds. This work has also been partly funded by eSSENCE, an e-Science collaboration funded as a strategic research area of Sweden, and by EU CEF grant number 2394203 (NORDIS—NORdic observatory for digital media and information DISorder).

References

1. Alghamdi, R., Alfalqi, K.: A survey of topic modeling in text mining. Int. J. Adv. Comput. Sci. Appl. (IJACSA) **6**, 147–153 (2015)
2. Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E.D., Gutierrez, J.B., Kochut, K.: A brief survey of text mining: Classification, clustering and extraction techniques, pp. 1–13 (2017). arXiv:1707.02919
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
4. Blondel, V.D., Guillaume, J., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. 1–12 (2008)
5. Bullinaria, J.A., Levy, J.P.: Extracting semantic representations from word co-occurrence statistics: a computational study. Behav. Res. Methods **39**, 510–526 (2007)
6. Dang, T., Nguyen, V.T.: ComModeler: topic modeling using community detection. In: Tominski, C., von Landesberger, T. (eds.), EuroVis Workshop on Visual Analytics (EuroVA). The Eurographics Association, pp. 1–5. (CH) (2018)
7. de Arruda, H.F., Costa, L.F., Amancio, D.R.: Topic segmentation via community detection in complex networks. Chaos **26**, 1–10 (2015)
8. Greene, D., Cunningham, P.: Practical solutions to the problem of diagonal dominance in kernel document clustering. In: Proceedings 23rd International Conference on Machine learning (ICML'06), pp. 377–384. ACM Press, New York (2006)
9. Hamm, A., Odrowski, S.: Term-community-based topic detection with variable resolution. Information **12**, 221–252 (2021)
10. Hubert, L., Arabie, P.: Comparing partitions. J. Classif. **2**, 193–218 (1985)
11. Kim, M., Sayama, H.: The power of communities: a text classification model with automated labeling process using network community detection. In: International Conference on Network Science, pp. 231–243. Springer, Berlin (2020)
12. Lancichinetti, A., Sirer, M.I., Wang, J.X., Acuna, D., K öording, K., Amaral, L.A.N.: High-reproducibility and high-accuracy method for automated topic classification. Phys. Rev. X. **5**, 1–11 (2015)
13. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E **74**, 1–2 (2006)
14. Palla, G., Derényi, I., Farkas, I., Vicsek, T.: Uncovering the overlapping community structure of complex networks in nature and society. Nature **435**, 814–818 (2005)
15. Salerno, M.D., Tataru, C.A., Mallory, M.R.: Word community allocation: discovering latent topics via word co-occurrence network structure (2015). http://snap.stanford.edu/class/cs224w-2015/projects_2015/Word_Community_Allocation.pdf
16. Sayyadi, H., Raschid, L.: A graph analytical approach for topic detection. ACM Trans. Internet Technol. 1–23 (2013)
17. Uysal, A.K., Gunal, S.: The impact of preprocessing on text classification. IInf. Process. Manage. **50**, 104–112 (2014)
18. Usai, A., Pironti, M., Mital, M., Mejri, C.A.: Knowledge discovery out of text data: a systematic review via text mining. J. Knowl. Manag. **22**, 1471–1488 (2018)
19. Xie, J., Kelley, S., Szymanski, B.K.: Overlapping community detection in networks: The state-of-the-art and comparative study. ACM Comput. Surv. **45**, 1–35 (2013)

Community Detection Using Moore-Shannon Network Reliability: Application to Food Networks

Ritwick Mishra, Stephen Eubank, Madhurima Nath, Manu Amundsen, and Abhijin Adiga

Abstract Community detection in networks is extensively studied from a structural perspective, but very few works characterize communities with respect to dynamics on networks. We propose a generic framework based on Moore-Shannon network reliability for defining and discovering communities with respect to a variety of dynamical processes. This approach extracts communities in directed edge-weighted networks which satisfy strong connectivity properties as well as strong mutual influence between pairs of nodes through the dynamical process. We apply this framework to food networks. We compare our results with modularity-based approach, and analyze community structure across commodities, evolution over time, and with regard to dynamical system properties.

Keywords Moore-Shannon network reliability · Networked dynamical systems · Food networks · Community detection · Modularity

1 Introduction

A community in a static network is usually defined as a set of vertices that are more densely connected with each other than with other vertices [23]. There are many ways to generalize this notion for a networked dynamical system. We suggest using two criteria

(1) The states of nodes in a community evolve more coherently with each other than with those in other communities.
(2) This coherence is robust against removing a few interactions both within and outside the community.

R. Mishra · S. Eubank · M. Amundsen · A. Adiga (✉)
Biocomplexity Institute and Initiative, University of Virginia, Charlottesville, USA
e-mail: abhijin@virginia.edu

M. Nath
Slalom Consulting, LLC, White Plains, NY, USA

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_21

The fundamental object of study in a dynamical system is the probability P that a system in a given configuration at time t_1 evolves into a particular configuration at t_2. This probability is a function of the configurations, the time duration, the nature of the dynamics and any dynamical parameters, as well as the network of interactions when the system consists of discrete interacting elements. Moore-Shannon Network Reliability (MSNR) [21] highlights its dependence on the network, and Birnbaum importance [2] measures how it changes as interactions are removed from the system. Communities are subgraphs with certain user-defined properties that are resilient to the removal of these interactions, thus meeting both the defining criteria above. Network reliability is famously hard to evaluate [28] or even approximate [25]. We take advantage of a statistical physics perspective on the underlying probability P: it is the value of the propagator, which can be well-approximated by strong- and weak-coupling perturbation expansions [5]. See Eubank et al. [8] for more details.

Application to food networks. Advances in technology and the resulting globalization have made it possible to break down geographical barriers to the movement of agricultural commodities. This has led to an increased reliance on the long-distance trade of commodities, making food systems vulnerable to extreme weather events, pests and pathogens, contamination, and politics [3, 7, 17]. Understanding the community structure of international food networks particularly in the context of spread processes representing cascading failures and biological invasions can help inform surveillance and control strategies. Food networks are directed and weighted, with weights representing the volume or value of trade between the exporter and importer. In this work, we apply our methods to identify communities that satisfy the two criteria stated in the beginning in country-to-country networks from Food and Agriculture Organization's Trade Matrix database [10] and domestic food networks from Freight Analysis Framework [9].

Contributions. We apply the MSNR-based approach to detect communities in directed edge-weighted networks that are *well-interacting* with respect to the underlying dynamical process, which in our case is the discrete-time Susceptible-Infected-Removed (SIR) diffusion process, and involves Monte-Carlo simulations and perturbation techniques. Our community detection algorithm is parameterized by the maximum size of the community, a transmission probability parameter that determines every edge probability as a function of its weight, and a quantization parameter that fixes the maximum possible number of distinct edge weights. We apply this framework to the four food networks. We compare our method with the Directed-Louvain algorithm [6] as well as from a network dynamics perspective. We also analyze the community structure resulting from varying the three parameters. The communities discovered in the food networks are extensively analyzed.

2 Related Work

Community detection is an extensively studied topic, with many methods that have been developed and applied to many fields [11, 16, 20]. Modularity-based approaches are very popular. Here, we use the Louvain algorithm adapted for directed edge-weighted graphs as a baseline to compare with our approach [6]. There has been recent work on community detection by local approaches which offer lower time complexity, valuable in large complex networks [1, 29]. In another line of work, spectral clustering has been utilized for community detection [24].

However, characterizing communities with respect to dynamical processes is an emerging body of work. Ghosh et al. [13] define a generalized Laplacian matrix that captures a class of linear dynamical processes. They introduce the notion of generalized conductance to measure the quality of communities with respect to the dynamical process. However, their work is limited to undirected networks. In another line of work, Zhang et al. [30] consider the problem of discovering clusters of nodes that have similar roles in a dynamical process (e.g., influential nodes or bridges).

There are very few works that have analyzed food networks from a dynamics perspective. Ercsey-Ravasz et al. [7] analyze country-to-country trade networks of agricultural commodities induced by ComTrade database [4]. Using a diffusion model called the food flux model, the authors show that contaminants can rapidly spread through a food network while, due to network effects, their origin becomes hard to trace. Sutrave et al. [26] use a diffusion model to evaluate surveillance strategies for the detection of pathogens. The precursor to this work, Nath et al. [22], adapts a network reliability framework for unweighted networks to analyze food networks from FAO (more details in Sect. 4). Lin et al. [19] analyze US domestic food flow networks using the Commodity Flow Survey (CFS) database [27] that belong to same class as the FAF networks analyzed in this paper, and Gephart and Pace [12], analyze global seafood trade. None of these works (except [22]) study community structure of these networks.

3 Preliminaries

Let $G(V, E)$ be a directed, edge-weighted graph. We use C to denote a community. For an edge $e \in E$, let w_e denote its weight. For a community C, let $d_G(C)$ denote the maximum distance between any pair of nodes belonging to C in G. Let $F(C)$ denote the sum of weights on all edges belonging to the subgraph of G induced by C. The diffusion probability on each edge e is $1 - \exp(-x w_e)$, where x is a tunable parameter. An induced subgraph is *strongly connected* (SC) if there is a directed path from any source to any target in the subgraph. A maximal strongly connected subgraph is a strongly connected component (SCC).

Diffusion model. Here, we apply the discrete-time SIR diffusion process, where each node that has transitioned from state S to I at time t infects each of its susceptible

out-neighbors with edge probability $1 - \exp(-x w_e)$ (where e is the corresponding edge) at time $t + 1$, and then moves to state R, never to participate in the diffusion process again. For some applications such as plant disease or pest epidemiology, SIR-like models have been considered at various spatial scales [14, 15]. The likelihood of spread via an edge is considered to be a direct function of its weight.

We used the Directed-Louvain (DL) method as the baseline to compare against. It partitions the node set with the objective of maximizing directed modularity for weighted networks [6, 18]: $Q = \frac{1}{m} \sum_{i,j} \left[A_{ij} - \frac{w_i^{in} w_j^{out}}{m} \right] \delta(c_i, c_j)$, where m is the sum of weights of all edges, A_{ij} represents the weight of the edge (if present) between nodes i and j belonging to communities c_i and c_j, w_i^{in} (resp. w_i^{out}) is the sum of weights of incoming edges (resp. outgoing edges) of i, while $\delta(c_i, c_j)$ is the indicator of the event: $c_i = c_j$. To evaluate communities from the perspective of the SIR process, we introduce the concept of *minimum influence*. Let p_{uv} denote the probability that a perturbation introduced at node u propagates to node v. The minimum influence for a node pair (u, v): $p_{\min}(u, v) = \min(p_{uv}, p_{vu})$.

4 Community Detection Framework

Here, we describe the Moore-Shannon Network Reliability community detection (MSNR-CD) method applied to directed edge-weighted graphs. Our approach is as follows. Given a networked dynamical system, let $\mathcal{E}$ be the desired outcome of a diffusion process whose probability, $\Pr(\mathcal{E})$, is a monotone non-decreasing function of iterative edge removal. This $\Pr(\mathcal{E})$ is the MSNR, and is the sum of all such configurations for which $\mathcal{E}$ is true. This is calculated using the Inclusion-Exclusion expansion to avoid over-counting. The probability of picking a single edge is a polynomial in e^{-x}, where x is the probability parameter (see Sect. 3), the probability of picking any particular random subgraph is also a polynomial in e^{-x}.

In our case, we calculate $\Pr(\mathcal{E})$ as the probability of the event that the graph of infected nodes resulting from the SIR process starting from a single random node contains an SCC of size $\leq n_{\text{SCC}}$, the maximum community size. In this iterative process, an edge whose removal maximizes $\Pr(\mathcal{E})$ in the residual graph is chosen and removed from the graph. The iterative process is terminated when the residual graph has no SCC of size $> n_{\text{SCC}}$, in which case, $\Pr(\mathcal{E}) = 1$. The SCCs in the residual graph constitute the communities. This process of discovering communities can be described from an adversarial viewpoint. An adversary desires to minimize the likelihood of many nodes being infected (akin to maximizing the $\Pr(\mathcal{E})$) by making minimal structural changes to the network (in this case, edge removal). The SCCs that survive this process emerge out as communities. The greater the number of iterations needed to destroy an SCC, the stronger the corresponding community.

Since computing the significance of edges in order to rank them is a computationally hard problem [25, 28], we apply perturbation techniques and Monte-Carlo simulations to estimate them. Here, we provide a sketch of our approach to calculate

$\Pr(\overline{\mathcal{E}})$. The details are in Eubank et al. [8]. In our case, a subgraph H is minimal if its size is at least $n_{\text{SCC}} + 1$ and it is strongly connected, but none of its subgraphs satisfy this property. Let $\mathcal{E}_H$ be the event that H occurred and let $\mathcal{H}$ denote all such H. If the sampled random subgraph (containing the random seed node) contains any $H \in \mathcal{H}$, then $\mathcal{E}$ does not hold. Therefore, $\overline{\mathcal{E}}$ is the disjunction of $\mathcal{E}_H$ events, and its probability can be expressed as an inclusion-exclusion expansion involving only conjunctions of one or more $\mathcal{E}_H$. The probability of a conjunction of $\mathcal{E}_H$ is given by the probability of the union of the corresponding subgraphs, which is, as above, a polynomial in e^{-x}.

There are three challenges in evaluating this probability expression. Firstly, the size of $\mathcal{H}$ could be very large. Second, there are $2^{|\mathcal{H}|}$ terms in the inclusion-exclusion expansion. Finally, heterogeneous weights can lead to extremely high degree polynomials. To cope with the first problem, we sample from $\mathcal{H}$. Even with only a sample set, though, evaluating the exact probability may be infeasible. To cope with the second problem, we truncate the inclusion-exclusion expansion at combinations of no more than k events $\mathcal{E}_H$. We denote the smallest power of e^{-x} appearing in the polynomial for the probability of any union (conjunction) of $k + 1$ subgraphs (clauses) by $m(k)$. Then the truncated inclusion-exclusion expansion yields the first $m(k)$ terms of a Taylor series expansion for the probability of $\overline{\mathcal{E}}$ in the limit as $e^{-x} \to 0$. Finally, we expect that the larger the degree of the polynomial, the worse the Taylor series approximation is. The true degree is given by the sum of the weights on all the edges, divided by their greatest common factor. To reduce the degree, we can bin the weights so that they are all small positive integers.

This work significantly extends the work by Nath et al. [22], where an iterative process of removal of significant interactions reveals strong communities. Our approach accounts for weights on the edges, while in the previous work, the approach was limited to unweighted networks; weighted edges were converted to multi-edges and post-processed to obtain communities.

5 Experimental Results

Data and networks. We analyzed food networks at two spatial scales: (a) country-to-country commodity-specific trade networks induced by the data from FAO [10], and (b) sub-national coarse food class flows between FAF zones in the US [9]. For FAO flows, we considered two commodities, tomato and corn. These are representative crops for vegetables and cereal respectively. We obtained the Detailed Trade Matrices at the country level for each of these crops corresponding to multiple years (2000–2019). Each edge is directed and associated with a weight corresponding to the quantity of trade (measured in tonnes) from source to destination. Henceforth, we refer to these networks as `tomato` and `corn`. Since there is some inconsistency in the reporting of trade volume by exporting and importing countries, we considered only edges with volume at least 10 tonnes. For the sub-national FAF flows, we looked at two commodity classes, cereal and other-ag-prod-mixed-freight and considered edges with volume at least 100 tonnes. We refer to these, as `cereal`

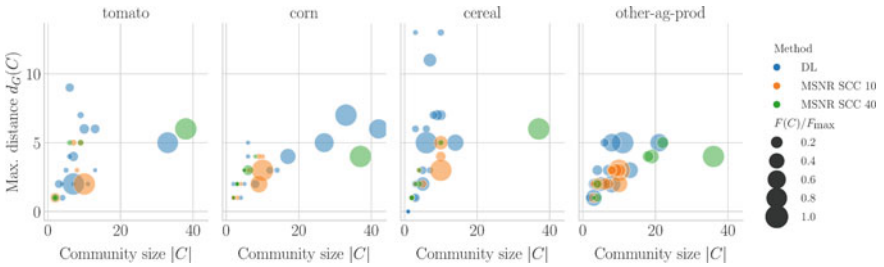


Fig. 1 Comparison of communities obtained using MSNR-CD and DL. Here, each community C is plotted with respect to its size $|C|$, max. distance $d_G(C)$, and the total flow within the community $F(C)$ normalized by $F_{\max}$, the maximum value of $F(\cdot)$ across all communities. We fix $x = 0.5$ and number of bins for Poisson rates = 16

and `other-ag-prod`. We denote each country by its ISO Alpha-3 codes, while each FAF zone has its state code prefixed to a numeric ID. Properties of networks are provided in Table 1.

Experiment design and implementation. We considered the following values of the diffusion probability parameter x: 0.1, 0.5, and 0.9 to study how the ranking of edges and, in turn, the community structure is affected by variations in transmission probability. For the max community size n_{SCC}, we consider these values: 10, 20, 30, and 40. The values considered for the number of bins for discretizing the weights were 2, 4, and 16. We then analyze the resulting communities in terms of both structural and dynamical properties. Since the method has a stochastic component, we replicate the experiment for up to 100 times, and analyze the similarity between the replicate results. The MSNR-CD framework is implemented using C++ and all the analysis was performed using Python 3.8. All experiments in this thesis were performed on an HPC system that runs Linux *x*86_64 operating system with a memory of 100 GB.

Comparison with modularity-based approach. In Fig. 1, we compare the structure of the communities resulting from MSNR and DL. Firstly, most of the communities (and all of the top communities) obtained using DL do not satisfy SC property. Secondly, in both `tomato` and `corn`, many communities have large $d_G(C)$ relative to their size $|C|$. On the other hand, we observe that with MSNR, the communities, by design, satisfy SC property, and have comparatively smaller $d_G(C)$ relative to their size. For $n_{\text{SCC}} = 40$, MSNR also captures a large community with high total flow $F(C)$. We compared the weighted modularity values (see Sect. 3 for the definition) of the community sets found by MSNR-CD and DL (Table 1). For all networks, DL generated community sets have significantly higher modularity values than those of MSNR-CD. This is expected as DL iteratively maximizes the modularity, in contrast to our reliability-based method. As we varied the n_{SCC}, the modularity was lowest at $n_{\text{SCC}} = 40$. However, in the case of the FAF networks, `cereal` and `other-ag-prod`, the modularity values of MSNR-CD are comparable to that of DL.

Table 1 Network properties and the modularity values of community sets found by MSNR-CD and DL. The fourth column corresponds to total traded volume (10^6 tonnes). The fifth and sixth columns correspond to maximum in degree and out degree respectively. The last two columns correspond to community size n_{SCC}

Network	Nodes	Edges	Vol.	Max. in	Max. out	DL	MSNR-CD	
							10	40
tomato	150	770	7.6	26	56	0.62	0.24	0.07
corn	183	1482	180	33	109	0.47	0.10	0.05
cereal	113	533	1.3	25	21	0.82	0.51	0.25
other-ag-prod	132	1067	1.1	36	36	0.80	0.66	0.57

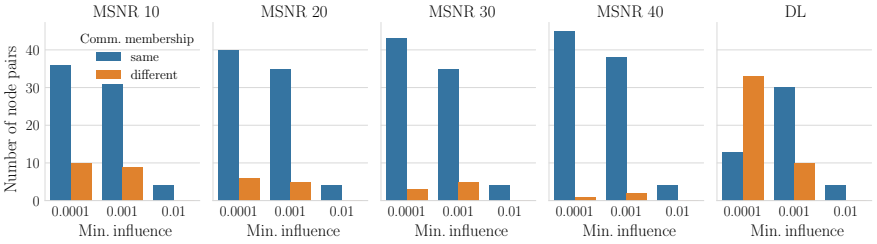


Fig. 2 Comparison of minimum influence $p_{\min}(u, v)$ for node pairs (u, v) across different community detection methods. The node pairs are categorized based on whether they belong to the same community or not. For MSNR-CD, we fix $x = 0.5$ and number of bins for Poisson rates = 16. To avoid clutter, only $p_{\min}(u, v) > 0$ are plotted. However, greater the x, greater the number of non-zero $p_{\min}$ values. But the trend remains the same. These are representative results for the tomato network. We have omitted remaining networks due to space constraint

Community and dynamics. In Fig. 2, we have plotted minimum influence $p_{\min}$ for node pairs with non-zero $p_{\min}$. In a good community partition, it is expected that node pairs belonging to the same community have relatively higher $p_{\min}$ than pairs where nodes belong to different communities. We compare DL with MSNR-CD for $n_{\text{SCC}} = 40$ as the larger communities in the latter are comparable in size with those obtained using DL. In the case of DL, the number of node pairs with nodes belonging to different communities and having large $p_{\min}$ is considerably higher than that in MSNR-CD. From a dynamics perspective, the DL approach fails to group together several mutually influential node pairs. Lastly, we compare the quality of communities for different values of n_{SCC}. When the community sizes are restricted (like $n_{\text{SCC}} = 10$), many mutually influential node pairs are assigned different communities. However, this happens only for node pairs for which $p_{\min}$ is small.

Comparison of communities across commodities. We observe from Fig. 3 that in tomato, the communities are often spatially contiguous, e.g., in Europe, Asia, and the Americas. In corn however, communities can be geographically diverse, e.g., the community formed by Americas, India, and Southeast Asia or the one formed by Australia and Southern Africa. In the FAF networks, we see strong geographic contiguity. Also, the community structure differs across networks.

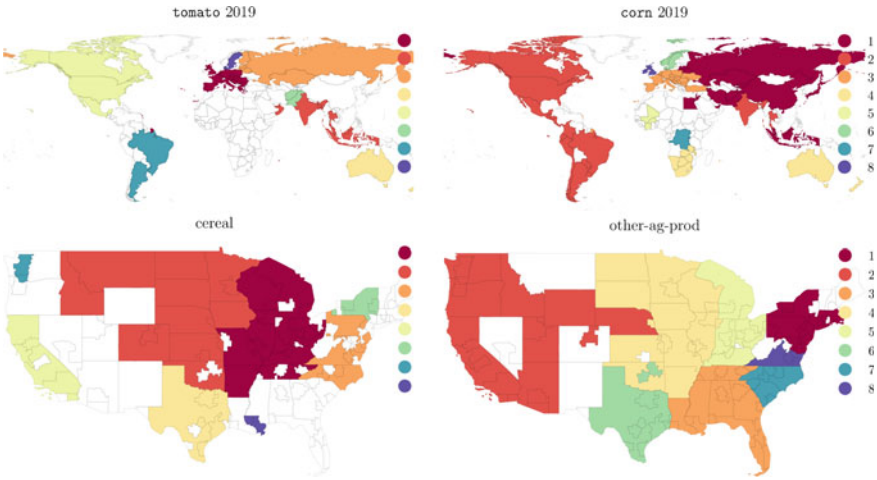


Fig. 3 Geographic contiguity in community structure, or the lack thereof: top eight communities (in order of size) in `tomato`, `corn`, `cereal` and `other-ag-prod` are shown for $x = 0.5$, $n_{\mathrm{SCC}} = 20$ and number of bins = 16

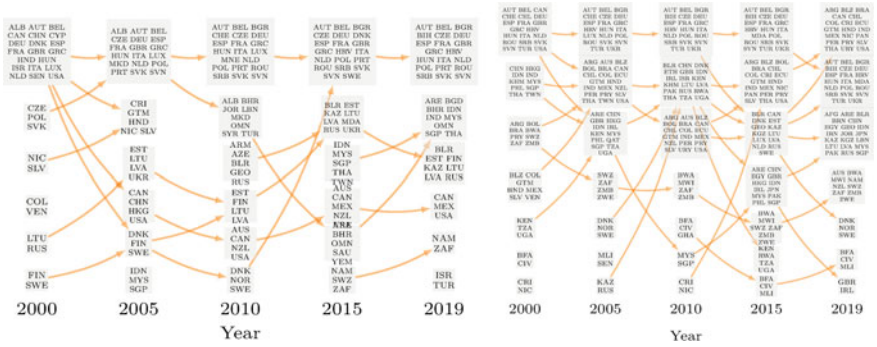


Fig. 4 Evolution of community structure over time in **a** `tomato`, **b** `corn`. The MSNR-CD parameters are $x = 0.5$, $n_{\mathrm{SCC}} = 20$ and number of bins = 16

Evolution of community structure over time. We compared communities found in `tomato` and `corn` across the years in the range 2000–2019 at 5 year intervals (Fig. 4). New communities emerge while others disintegrate through the years. In `tomato`, early communities were often pairs of countries, which coalesced into significant groups by the year 2019, e.g., in recent years, Russia and Lithuania are part of a larger community showing intensification of trade. Similarly, in `corn`, the general trend is of an increase in the number of large-sized communities.

Hierarchy in communities. In Fig. 5, we compare the top communities obtained for different values of n_{SCC} in `corn` and `other-ag-prod`. We observe hierarchical clustering. The larger communities obtained for large values of n_{SCC} (starting

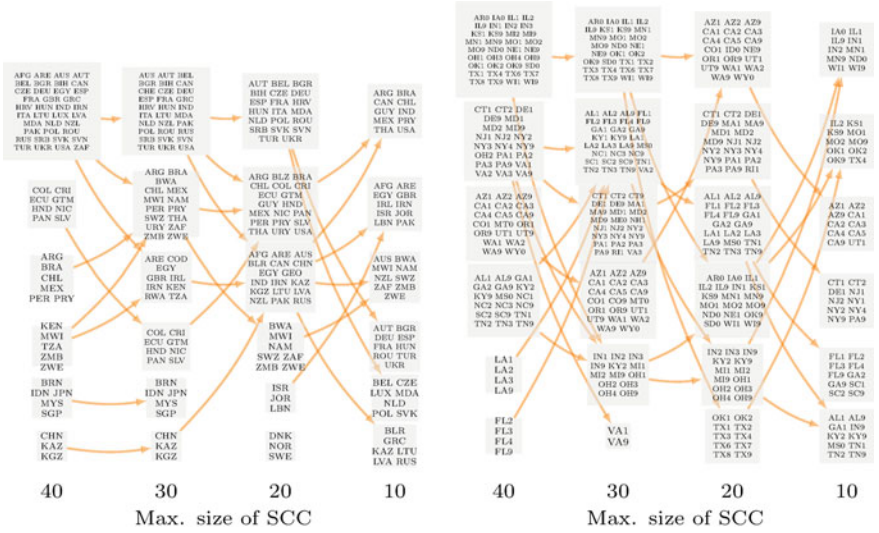


Fig. 5 Change in community structure with decreasing size of n_{SCC} for **a** `corn`, **b** `other-ag-prod` , keeping $x = 0.5$ and number of bins for Poisson rate = 16. The remaining networks are omitted due to space constraints

with 40) progressively break into smaller stronger communities when n_{SCC} is lowered. In `other-ag-prod`, the few large communities turn into many medium sized ones, accompanied by numerous exchanges of members. Not surprisingly, communities that were relatively small to begin with remain relatively unaltered with decreasing n_{SCC}. We observe the same hierarchical patterns in `tomato` and `cereal` too thus demonstrating the effectiveness of our methods in discovering strong sub-communities.

Sensitivity to parameters. The ranking of edges in the MSNR-CD algorithm can depend on the diffusion probability, and, in turn, can affect the community structure. The edge probability depends on the phenomenon being studied and can vary widely; some species are more invasive than others, for example. In Fig. 6a, we observe that the community structure is stable for almost all communities. Plot (b) shows that the community structure hardly varies with the number of bins for Poisson rates. We observe generally a power-law relationship in edge weights. For most of the edges, since their weights are very small, discretizing has the effect of increasing their weight. Lesser the number of bins, the greater the increase in weight. For very few edges (less than 5%), the effect is opposite, their weight decreases with the number of bins. Despite this, the community partition does not seem to be changing.

Stability of the resulting communities. Due to the stochastic nature of MSNR-CD, we replicated the community detection experiment up to 100 times. To find how similar the obtained community sets were, we computed the Rand index for every pair of results. In `tomato` and `cereal`, we found that MSNR-CD produced the

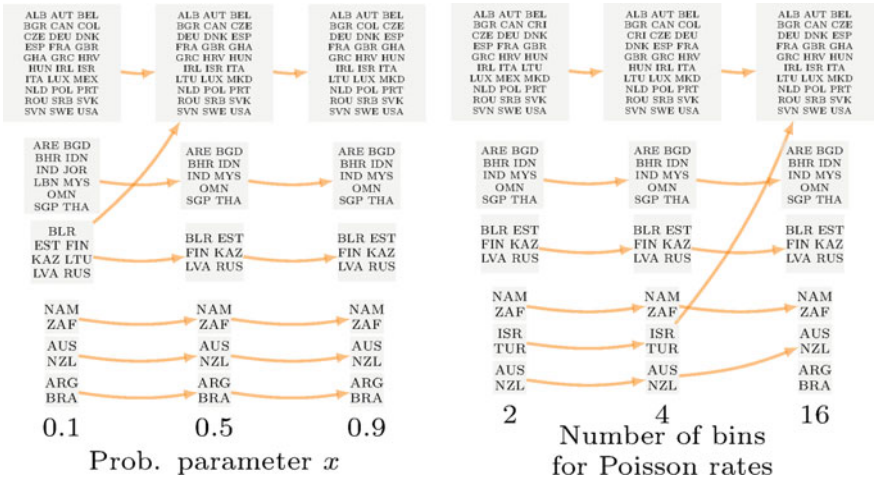


Fig. 6 Changes in community structure with varying MSNR-CD parameters: **a** probability parameter x and **b** number of bins for Poisson rates. Both plots are for `tomato`. In plot (**a**), number of bins = 16, and in plot (**b**), $x = 0.5$. In both, we set $n_{\text{SCC}} = 30$

exact same communities on every run. In `corn` and `other-ag-prod`, there were only two different community structures found among all the runs with Rand index of 0.919 and 0.97 respectively, indicating their high similarity.

6 Future Work

In this work, we developed an approach to discover strong communities in directed edge-weighted networks with respect to both network structure and dynamics. An important direction of work to explore in this regard is the effect of introducing edge costs and budget. Some interactions might be more difficult to remove than others, thus potentially leading to very different communities. In the context of commodity flow networks, it is important to consider aggregated flows of commodities as it is common to transport many commodities together (mixed freight). Our generic framework based on Monte-Carlo and perturbation techniques can be used to discover communities in multilayer networks with complex diffusion processes.

Acknowledgements This work was supported in part by the United States Agency for International Development under the Cooperative Agreement no. AID-OAA-L-15-00001, Feed the Future Innovation Laboratory for Integrated Pest Management, AgAID grant no. 2021-67021-35344 from the USDA NIFA, grant no. 2019-67021-29933 from the USDA NIFA, UVA Strategic Investment Fund SIF160, NSF Expeditions in Computing Grant CCF-1918656, and OAC-1916805 (CINES). We thank the reviewers for providing valuable suggestions for revising the paper.

References

1. Berahmand, K., Bouyer, A., Vasighi, M.: Community detection in complex networks by detecting and expanding core nodes through extended local similarity of nodes. IEEE Trans. Comput. Social Syst. **5**(4), 1021–1033 (2018)
2. Birnbaum, Z.W.: On the importance of different components in a multicomponent system. In: Krishnaiah, P.R. (ed.) Multivariate analysis-II. Proceedings of the 2nd International Symposium on Multivariate Analysis, pp. 581–592. Academic Press, New York (1969)
3. Chen, Y.: Trade, food security, and human rights: the rules for international trade in agricultural products and the evolving world food crisis. Routledge (2016)
4. ComTrade. Import and export (2021). http://comtrade.un.org/db/
5. Domb, C.: Order-disorder statistics. ii. a two-dimensional model. Proc. R. Soc. London Ser. A Math. Phys. Sci. **199**(1057), 199–221 (1949)
6. Dugué, N., Perez, A.: Directed Louvain: maximizing modularity in directed networks. Ph.D. thesis, Université d'Orléans (2015)
7. Ercsey-Ravasz, M., Toroczkai, Z., Lakner, Z., Baranyi, J.: Complexity of the international agro-food trade network and its impact on food safety. PloS One **7**(5), e37810 (2012)
8. Eubank, S., Nath, M., Ren, Y., Adiga, A.: Perturbative methods for mostly monotonic probabilistic satisfiability problems (2022). arXiv:2206.03550
9. FAF. Freight Analysis Framework (FAF) version 5 (2022). https://faf.ornl.gov/faf5/
10. FAO. Production and trade (2021). http://www.fao.orgfaostatendata
11. Fortunato, S.: Community detection in graphs. Phys. Rep. **486**(3–5), 75–174 (2010)
12. Gephart, J.A., Pace, M.L.: Structure and evolution of the global seafood trade network. Environ. Res. Lett. **10**(12), 125014 (2015)
13. Ghosh, R., Teng, S., Lerman, K., Yan, X.: The interplay between dynamics and networks: centrality, communities, and Cheeger inequality. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1406–1415 (2014)
14. Gilligan, C.A.: Sustainable agriculture and plant diseases: an epidemiological perspective. Philos. Trans. R. Soc. B Biol. Sci. **363**(1492), 741–759 (2008)
15. Gilligan, C.A., Gubbins, S., Simons, S.A.: Analysis and fitting of an sir model with host response to infection load for a plant disease. Philos. Trans. R. Soc. London Ser. B Biol. Sci. **352**(1351), 353–364 (1997)
16. Harenberg, Steve, Bello, Gonzalo, Gjeltema, La., Ranshous, Stephen, Harlalka, Jitendra, Seay, Ramona, Padmanabhan, Kanchana, Samatova, Nagiza: Community detection in large-scale networks: a survey and empirical evaluation. Wiley Interdiscip. Rev. Comput. Stat. **6**(6), 426–439 (2014)
17. Hulme,. P.E.: Trade, transport and trouble: managing invasive species pathways in an era of globalization. J. Appl. Ecol. **46**(1), 10–18 (2009)
18. Leicht, E.A., Newman, M.E.J.: Community structure in directed networks. Phys. Rev. Lett. **100**(11), 118703 (2008)
19. Lin, X., Dang, Q., Konar, M.: A network analysis of food flows within the United States of America. Environ. Sci. Technol. **48**(10), 5439–5447 (2014)
20. Malliaros, F.D., Vazirgiannis, M.: Clustering and community detection in directed networks: a survey. Phys. Rep. **533**(4), 95–142 (2013)
21. Moore, E.F., Shannon, C.E.: Reliable circuits using less reliable relays. J. Franklin Inst. **262**(3), 191–208 (1956)
22. Nath, M., Venkatramanan, S., Kaperick, B., Eubank, B., Marathe, M.V., Marathe, A., Adiga, A.: Using network reliability to understand international food trade dynamics. In: International Conference on Complex Networks and their Applications, pp. 524–535. Springer (2018)
23. Newman, M.E.J.: Modularity and community structure in networks. Proc. Nat. Acad. Sci. **103**(23), 8577–8582 (2006)
24. Palmer WR, Zheng, T.: Spectral clustering for directed networks. In: Benito, R.M., Cherifi, C., Cherifi, H., Moro, E., Rocha, L.M., Sales-Pardo, M. (eds.), Complex Networks & Their Applications IX, pp. 87–99. Springer International Publishing, Cham (2021)

25. Roth, Dan: On the hardness of approximate reasoning. Artif. Intel. **82**(1), 273–302 (1996)
26. Sutrave S., Scoglio, C., Isard, S.A., Shawn Hutchinson, J.M., Garrett, K.M.: Identifying highly connected counties compensates for resource limitations when evaluating national spread of an invasive pathogen. PLoS One **7**(6), e37793 (2012)
27. United States Census Bureau. Commodity Flow Survey (2017). https://www.census.gov/programs-surveys/cfs.html
28. Valiant, L.G.: The complexity of enumeration and reliability problems. SIAM J. Comput. **8**(3), 410–421 (1979)
29. Wang, X., Liu, G., Li, J., Nees, J.P.: Locating structural centers: a density-based clustering method for community detection. PLOS One **12**(1), 1–23 (2017)
30. Zhang, Y., Adhikari, B., Jan, S.T.K., Aditya Prakash, B.: Meike: influence-based communities in networks. In: Proceedings of the 2017 SIAM International Conference on Data Mining, pp. 318–326. SIAM (2017)

Structural Network Measures

Winner Does Not Take All: Contrasting Centrality in Adversarial Networks

Anthony Bonato, Joey Kapusin, and Jiajie Yuan

Abstract In adversarial networks, edges correspond to negative interactions such as competition or dominance. We introduce a new type of node called a low-key leader in adversarial networks, distinguished by contrasting the centrality measures of CON score and PageRank. We present a novel hypothesis that low-key leaders are ubiquitous in adversarial networks and provide evidence by considering data from real-world networks, including dominance networks in 172 animal populations, trading networks between G20 nations, and Bitcoin trust networks. We introduce a random graph model that generates directed graphs with low-key leaders.

1 Introduction

Adversarial networks, where edges capture competition, dominance, or enmity, are gaining prominence in the study of complex networks. Negative interactions are critically important to the study of social networks and more broadly, real-world complex networks, and are often hidden drivers of link formation. Adversarial networks appear throughout network science, and examples range from negatively correlated stocks in market graphs [2], trade deficit between nations [12], the spatial location of cities as a model to predict the rise of conflicts and violence [9], and animal predation networks and food webs [13]. Even in the highly cited Zachary Karate club network [21], the negative interaction between the administrator and instructor was the impetus for the split of the club participants into two communities. Adversarial networks may be directed or undirected, and we focus on the directed case in the present paper.

The *Dynamic Competition Hypothesis* (or DCH) was introduced in [5] and provides a quantitative framework for the structure of evolving adversarial or competition

Supported by an NSERC Discovery Grant.

A. Bonato (✉) · J. Kapusin · J. Yuan
Toronto Metropolitan University, Toronto, ON, Canada
e-mail: abonato@ryerson.ca

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_22

networks. The DCH posits that leaders in adversarial networks exhibit high closeness, low in-degree, and high out-degree. Leaders also possess high common out-neighbor (or CON) scores, which measures shared competition versus other nodes; see Sect. 2 for a definition and discussion of CON scores.

We focus on the role of competing notions of centrality in adversarial networks. Centrality measures are used to identify certain key nodes within complex networks. There are several methods to measure centrality, such as degree distribution, PageRank, closeness, and betweenness. Our focus will be on the novel detection and analysis of certain nodes in adversarial networks, measured by comparing CON scores and PageRank; the latter measure is an established tool for determining influential nodes in a network.

To motivate our discussion of low-key leaders, we consider the popular social game television franchise Survivor, where contestants progressively eliminate each other by voting until only one remains. In the 35th season of the American social game show Survivor, Ben Driebergen won over finalists Chrissy Hofbeck and Ryan Ulrich [17]. While Ryan and Chrissy played a strategic game throughout the reality show competition by forging alliances and voting out key competitors, Ben won in part based on his finding multiple immunity idols and the sympathy he garnered from the jury as a veteran. Although Ben won the game, more low-key players like Ryan were instrumental in shaping the underlying adversarial, co-voting network.

In Sect. 2, we identify a low-key leader as a node which has a relatively high CON score, but low PageRank; intuitively, a low-key leader is highly likely to affect link evolution while remaining less visible in the network. This notion is analogous to so-called silent or quiet leaders in management positions in companies, who may be more diplomatic, introverted, but remain influential; see [8]. Low-key leaders appear to be ubiquitous in adversarial networks, and we support this hypothesis in Sect. 3 with data from three distinct sources: dominance networks in 172 distinct animal populations, trading networks between G20 nations, and Bitcoin trust networks. A new random graph model is introduced in Sect. 3, with the aim of synthetically generating low-key leaders in scale-free directed graphs. The concluding section contains several directions for future research.

We consider directed graphs (or *digraphs*) with multiple directed edges in the paper. Additional background on graph theory and complex networks may be found in the book [20].

2 Low-Key Leaders

An approach taken in [5, 6] in the detection of leaders in adversarial networks is the common out-neighbour score (or CON score). For nodes u, v, w in a graph G, we define w to be a *common out-neighbor* of u and v if (u, w) and (v, w) are two directed edges in G. We let $\mathrm{CON}(u, v)$ be the number of common out-neighbour of distinct nodes u and v, and define

$$\mathrm{CON}(u) = \sum_{v \in V(G)} \mathrm{CON}(u, v).$$

A high CON score for a node indicates it shares many of the same adversaries with other nodes, and hence, is more in sync with how links evolve in the network. A low CON score indicates the opposite trait, where the node is less of a driver of link evolution.

PageRank centrality is based on the stationary distribution of a random walk on the network that periodically teleports to a node chosen uniformly at random. For a formal definition of PageRank, see [4]. In adversarial networks, we compute the PageRank of nodes on the reversed-edge network, where we change the orientation of the directed edges. Hence, if a node in the network has many out-edges, they will more likely have higher PageRank in the reversed-edge network.

We define a *low-key leader* (or *LKL*) in an adversarial networks as a node whose CON score and PageRank are negatively correlated, with high CON score and low PageRank. Recall that, according to the DCH, leaders in a network are nodes that exhibit high closeness, high CON score, low in-degree, and high out-degree. In contrast, low-key leaders have less centrality due to their low PageRank but remain influential actors in the network owing to their high CON score.

The definition of low-key leader given in the previous paragraph is more heuristic, as having low or high scores is subject to interpretation. To make the definition more precise, we consider the following approach. While CON scores are integers, PageRank consists of probabilities in [0, 1]. To compare the difference between the two scores to validate the presence of LKLs, we re-scale both scores by using the *unity-based normalization*, defined as follows. Suppose we are given real numbers $X_1, X_2, \ldots, X_n$, with minimum $X_{\min}$ and maximum $X_{\max}$. For $1 \leq i \leq n$, define

$$X_{i,\mathrm{norm}} = \frac{X_i - X_{\min}}{X_{\max} - X_{\min}}.$$

Such scaling measure is used to set all values $X_{i,\mathrm{norm}} \in [0, 1]$; note that we apply this normalization also to PageRank (whose values are already in [0, 1]) for consistency.

Suppose that for a set of nodes x_i, where $1 \leq i \leq n$, the CON score and PageRank of x_i are denoted by CON_i and PR_i, respectively. Define

$$\varepsilon_i = \mathrm{CON}_{i,\mathrm{norm}} - \mathrm{PR}_{i,\mathrm{norm}}.$$

Note that $\varepsilon \in [-1, 1]$. We abuse notation and refer to ε_i as simply ε. We consider a node to be a low-key leader if it has the maximum value of ε, and $\varepsilon > 0.5$. We refer to ε as the *low-key leader strength* of a node.

We hypothesize that adversarial networks typically contain at least one low-key leader. Note that the assertion is on the presence of influential nodes within adversarial networks; no other data is required other than the presence of negative ties. We provide evidence for the hypothesis in real-world, adversarial networks in the next section.

3 Data and Methods

To validate our hypothesis on low-key leaders, we consider three types of adversarial networks: dominance networks in animal groups, Bitcoin trust networks, and the trading networks between nations. In the interest of space, we refer the reader to https://github.com/jkapusin/Low-Key-Leaders for complete data sets, as well as CON, PageRank, and low-key leader strengths of nodes for each network.

3.1 Dominance Networks

We first consider dominance networks, where directed edges correspond to some form of dominant-subordinate relations between members of an animal population. The animal social dominance data set, compiled by Shizuka and McDonald [16] and available in the Dryad Digital Repository, contains 172 distinct dominance networks of an animal group. Networks are represented as weighted adjacency matrices, and each entry in a given matrix corresponds to the number of times that the animal in the row is dominated by the one in the column.

We first consider more closely the Bonanni2007-2 data set, which was randomly chosen. The Bonanni2007-2 data set contains information on dominance behaviour of mongrel dogs living in a free-ranging or semi-free-ranging state. Directed edges correspond to aggressive signals such as lunging, biting, or snarling between the dogs. Each individual dog is identified using a three letter code. See Fig. 1 for a visualization of this network and see [3] for more discussion.

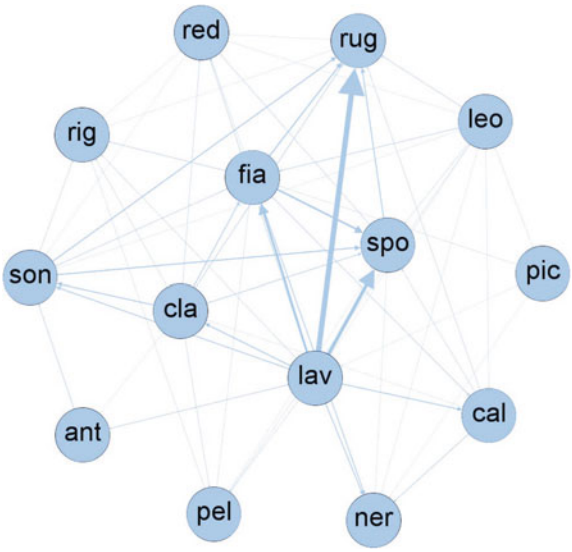


Fig. 1 The Bonanni2007-2 dominance network

ID	CON	PR	ε
leo	488	0.0393	0.9206
cla	446	0.1702	0.4560
fia	396	0.1398	0.4239
son	378	0.0809	0.5466
red	356	0.0128	0.6856
cal	343	0.0132	0.6540
rig	260	0.0132	0.4596
lav	253	0.3684	-0.5504
pel	218	0.0159	0.3537
spo	193	0.0363	0.2381
ner	165	0.0114	0.2422
rug	140	0.0109	0.1850
pic	97	0.0146	0.0740
ant	61	0.0730	-0.1737

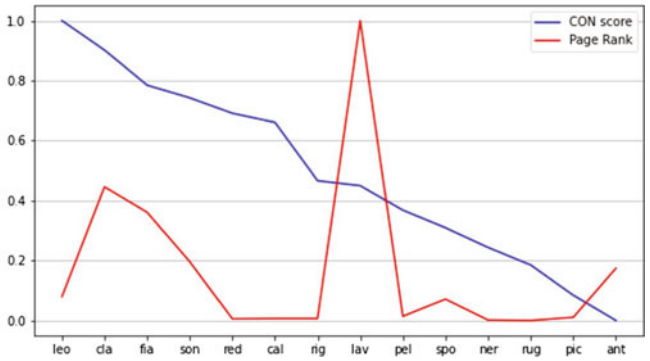


Fig. 2 CON score versus PageRank in the Bonanni2007-2 dominance network. Nodes such as leo, fia, and cla represent a population of mongrel dogs, and edges correspond to an observed dominance behavior. The table lists the CON scores, PageRank (PR), and low-key leader strengths (ε). The histogram depicts normalized CON scores versus PageRank

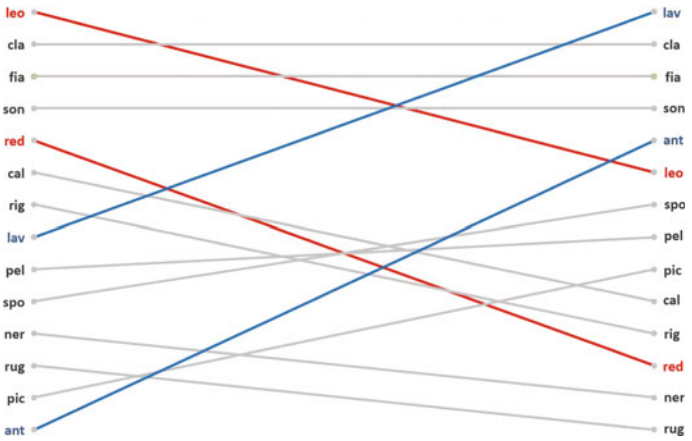


Fig. 3 A Slope Graph to compare the rankings via CON and PageRank in the Bonanni2007-2 dominance network. On the left, the top nodes via CON scores, while on the right, the top nodes via PageRank on the reversed-edge network. Nodes are labeled in grey if the difference in rankings is less than five. Nodes are labeled in red if the CON ranking is at least five places higher than the PageRank, and in blue if the PageRank is at least five places higher than the CON score

See Fig. 2 for a comparison of centrality scores in the Bonanni2007-2 network. We seriate the animals via the difference in their CON score and PageRank from the highest value to the lowest. A slope graph representation of the data is provided in Fig. 3 and compares the rankings with CON scores and PageRank. The dog leo emerges as having the top CON score and highest difference between their CON scores and PageRank, with low-key leader strength 0.9206. We therefore determine that leo is the low-key leader in the Bonanni2007-2 network (Fig. 4).

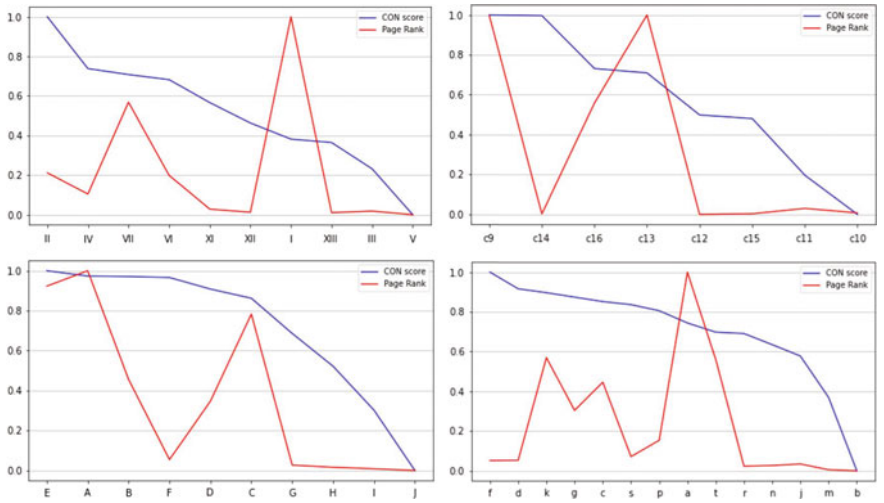


Fig. 4 CON scores versus PageRank in four animal dominance populations. From top left and clockwise, we list the data set name, predicted low-key leader, and relevant citation: Allee1954-3, II, [1]; Poisbleau2005-1a, C14, [14]; deWaal1977-1, f, [7]; and Watt1986-1a, F, [19]

We considered the low-key leader strengths of nodes in all the 172 animal dominance networks and detected LKLs in 155 or 90.12% of them using a low-key leader strength of $\varepsilon = 0.5$. If we let choose ε to be at least 0.4, then 95.35% of them contain a low-key leader. We think the prevalence of LKLs in these dominance networks provides support for our hypothesis. Figure 4 compares CON scores and PageRank in four other animal dominance populations [1, 7, 14, 19]. As referenced at the beginning of the section, see https://github.com/jkapusin/Low-Key-Leaders for a list of all the detected LKLs.

3.2 Trade Networks

UN Comtrade [18] is a statistical database storing international trading information between nations that is organized by the United Nations Statistics Division. There are over 170 nations reporting their annual international trading data in the database.

We extract trading data of the 19 nations within the G20 and Spain from 2019. An edge directed inward to the reporter nation represents importation, while edges directed outwards represent exportation. Hence, nations with larger in-degree than out-degree have larger trading deficits; a trading deficit between nations may be viewed as form of dominance or adversarial relationship. Trading volumes are considered as the weights of the edges in the weighted graphs. See Fig. 5.

Figure 6 is the histogram of the weighted networks of the CON score versus PageRank of the trading networks. Figure 7 is the corresponding slope graph. Among these

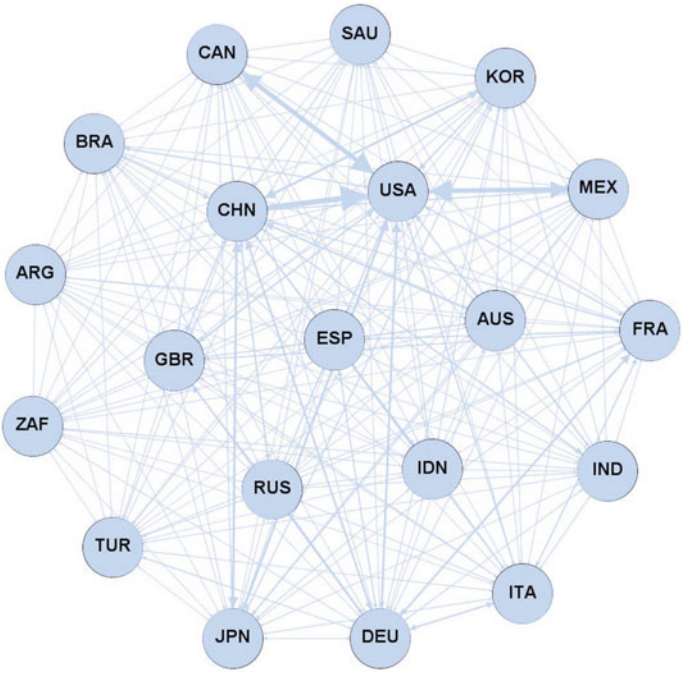


Fig. 5 The G20 trade network

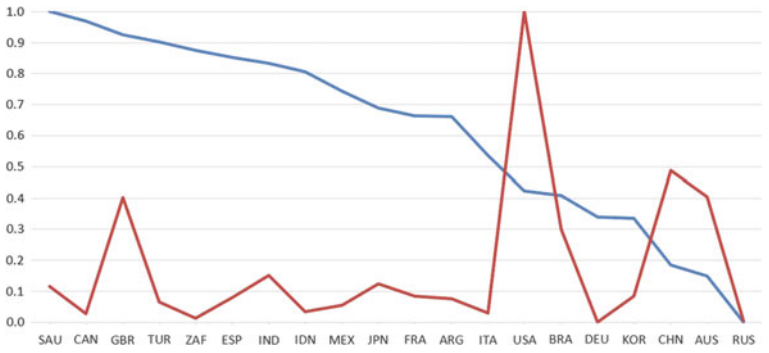


Fig. 6 CON score (blue) versus PageRank (red) for the trade deficit network of G20 nations. Nodes are nations denoted by three-letter codes such as CAN, USA, and CHN

nations, Canada (CAN) emerges as a low-key leader with low-key leader strength 0.9401. These results support that anecdotal view that while Canada does not have the highest trade in the G20, it plays an influential secondary role in shaping international trade dynamics among the higher income nations.

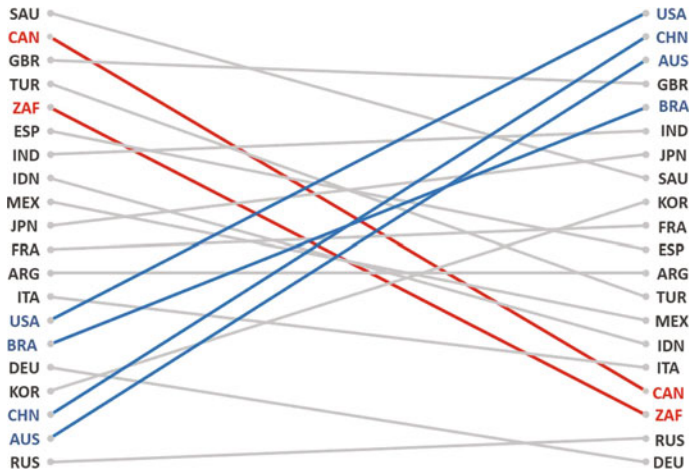


Fig. 7 A Slope Graph to compare the rankings via CON and PageRank in the G20 trade deficit network. On the left, the top nodes via CON scores, while on the right, the top nodes via PageRank. Nodes are labeled in grey if the difference in rankings is less than ten. Nodes are labeled in red if the CON ranking is at least ten places higher than the PageRank, and in blue if the PageRank is at least ten places higher than the CON ranking

3.3 Bitcoin Trust Networks

Our final data set consists of a graph with a much larger number of nodes and edges than the dominance networks and trading networks. Users trading the cryptocurrency Bitcoin may anonymously rate others on their trustworthiness. The members of Bitcoin trust networks rate other members by assigning an integer from -10 (total distrust) to $+10$ (total trust); see [10]. We formed an adversarial network with nodes the users and edges corresponding to negative ratings; for example, if user x rates user y with -2, then we formed a directed edge (x, y). The rating scores are considered as the weights in the weighted network. Data for Bitcoin trust networks was taken from [11].

Figure 8 provides a visualization of the OTC Bitcoin trust network with 5882 nodes and 3563 edges. As the number of users is large, we select the top 340 users by sorting the difference between CON score and PageRank of each user ranked from the highest to the lowest. Users outside this set of 340 had scores at or near zero and so were omitted. The histogram contrasting CON scores and PageRank is given in Fig. 9. The user with ID 3789 emerged as a LKL from our analysis, with low-key leader strength 0.5552.

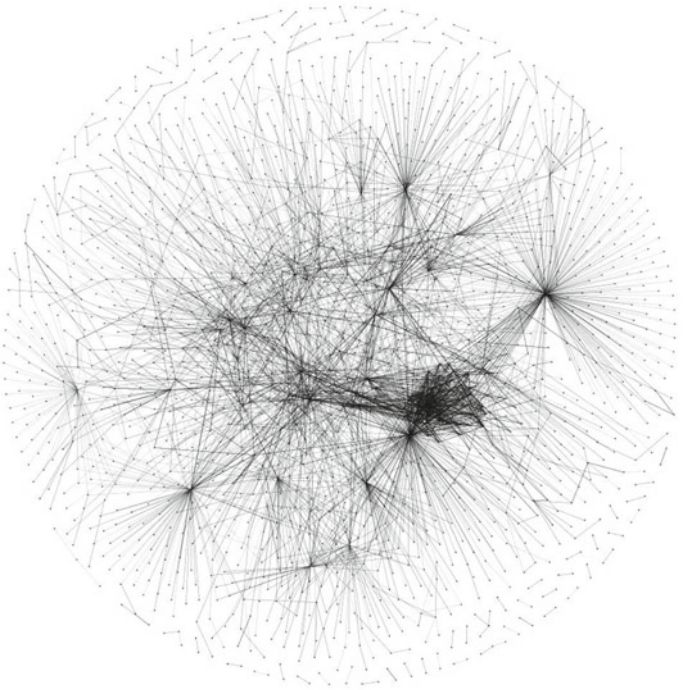


Fig. 8 A visualization of the bitcoin over-the-counter (or OTC) trust network, where nodes are users and directed edges correspond to negative ratings between them

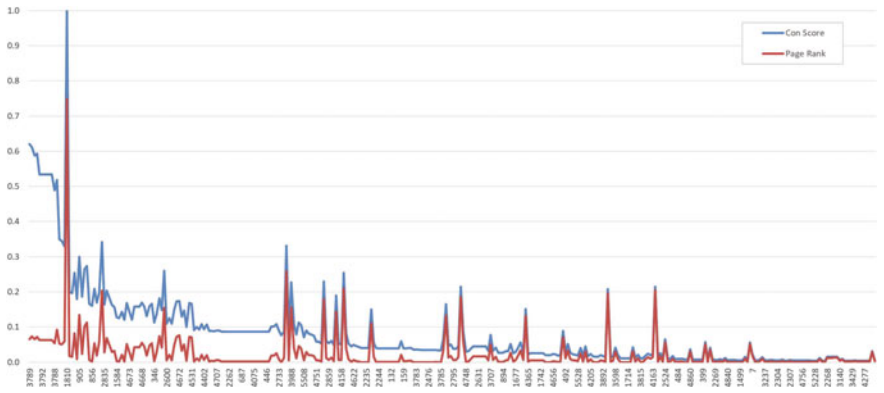


Fig. 9 CON (blue) and PageRank (red) scores in the OTC bitcoin trust network

4 Directed Ranking Model

Many models for complex networks were proposed over the last two decades involving various mechanisms such as preferential attachment and copying; see [4], for example, for an early survey. We introduce a random directed graph generation model based on ranked-based attachment that simulates digraphs containing a low-key leader. In rank-based attachment models, the degree of a node is a function of their predetermined rank. An undirected rank-based attachment model was introduced in [15]. Such models are offline, in the sense that the number of nodes will not change over time.

The *directed ranking model* produces a sequence of digraphs G_n with nodes $V_n = \{1, 2, \ldots, n\}$, where $n \geq 1$ is an integer. The model has a fixed parameter and an adjustable parameter: the order of the digraph $n \in \mathbb{N}^+$ is fixed and the attachment strength α is chosen in $(0, 1)$. Note that n will not change over time. For each $v_i \in V_n$, it receives a label $l(n_i) \in \{1, 2, \ldots, n\}$ chosen uniformly at random. Nodes are ranked based on their labels; that is, we denote l as the label which node receives and r as its corresponding rank. If $l(v_i) < l(v_j)$, then the nodes are ranked $r(v_i) > r(v_j)$. Each node has a unique rank and the node that receives label 1 obtains the highest rank among all, while whichever node receives label n has the lowest rank. For simplicity, we reorder the sequence of nodes and simply let $r(v_i) = i$ for all choices of i.

Edges in the model are added according to the attachment strength α. They are generated by following the random process: for each distinct pair of nodes v_i and v_j, the probability of generating a directed edge (i, j) equals

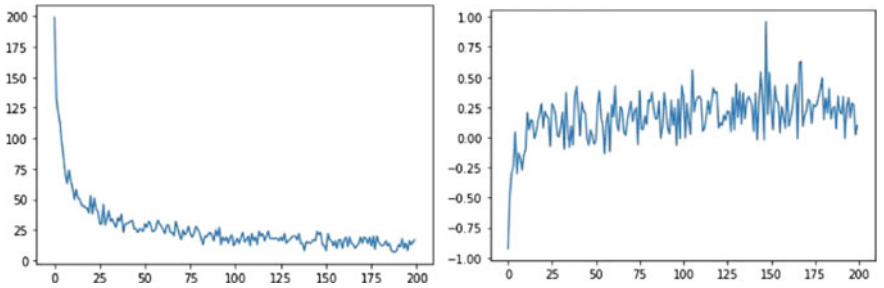
$$\mathbb{P}((i, j) \in E(G_n)) = j^{-\alpha}.$$

We set $\alpha = \frac{1}{2}$ for simplicity. We next uniformly choose one node at random from the existing nodes, say v_m, that we call the *copy node*. Let v_r be the node with the highest out-degree. We deterministically add directed edges (v_m, v_j), for every directed edge (v_r, v_j). The copy node v_m has high out-degree but note that its in-degree remains unchanged. Note that only one copy node is selected.

Note that for a node v_i, its in-degree $\deg^-(v_i)$ is the sum of $n - 1$ independent Bernoulli trials with a predetermined probability based on the ranking scheme. The expected in-degree of a node v_i with strength attachment $\alpha = \frac{1}{2}$ may be expressed as:

$$\begin{aligned}\mathbb{E}(\deg^-(v_i)) &= \sum_{j=1}^{n} i^{-\frac{1}{2}} \\ &= i^{-\frac{1}{2}} \sum_{j=1}^{n} 1 \\ &= ni^{-\frac{1}{2}}.\end{aligned}$$

Therefore, the expected in-degree of v_i is one of n, $\frac{n}{\sqrt{2}}$, $\frac{n}{\sqrt{3}}$, $\ldots$, $\sqrt{n}$.



(a) The in-degree distribution of G_{200}.

(b) ε values of G_{200}.

Fig. 10 Simulation of the directed ranking model with 200 nodes

We consider an example simulating a digraph with $n = 200$ using the directed ranking model. As shown in Fig. 10a, the in-degree distribution possesses a heavy tail. We determined the CON scores and PageRank in our simulated digraph. As depicted in Fig. 10b, the copy node v_{148} is the low-key leader as it has the largest low-key leader strength equaling 0.9354.

5 Discussion and Future Work

We introduced a new type of node in adversarial networks using centrality measures. A low-key leader (or LKL) corresponds to a node that has a relatively high CON score, but relatively low PageRank. We asserted that LKLs typically exist in adversarial networks. To validate the assertion in real-world networks, we analyzed three different types of adversarial networks: animal dominance networks in 172 animal populations, trading networks between G20 nations, and Bitcoin trust networks. The analysis of the contrasting CON scores and PageRank in the three types of data sets supported presence of LKLs in trade networks, Bitcoin trust networks, and in over 90% of the considered animal dominance networks. We introduced the directed ranking model that generates with high probability digraphs that have an expected power law in-degree distribution, and also possess low-key leaders.

While we provided evidence for our hypothesis on LKLs using three different types of networked data sets, we may consider networks in other knowledge domains to further validate their presence. We will consider a more rigorous analysis of the directed ranking model in future work, exploring concentration results on the in-degree distribution, as well as small world and spectral properties. An on-line version of the directed ranking model, where new nodes are introduced over time, will be considered in the full version of the paper. Finally, while we predicted the existence of the low-key leaders in adversarial networks, we do not posit in this work why they exist. The underlying mechanism as to why LKLs appear to be prevalent in adversarial networks remains open.

References

1. Allee, W.C., Dickinson, J.C., Jr.: Dominance and Subordination in the smooth dogfish mustelus canis (Mitchill). Physiol. Zool. **27**, 356–364 (1954)
2. V. Boginski, S. Butenko, P.M. Pardalos, On structural properties of the market graph. In: Innovation in Financial and Economic Networks, pp. 29–45. Edward Elgar Publishers
3. Bonanni, R., Cafazzo, S., Abis, A., Barillari, E., Valsecchi, P., Natoli, E.: Age-graded dominance hierarchies and social tolerance in packs of free-ranging dogs. Behavi. Ecol. **33**, 1004–1020 (2017)
4. Bonato, A.: A course on the web graph. In: American Mathematical Society Graduate Studies Series in Mathematics, Providence, Rhode Island (2008)
5. A. Bonato, N. Eikmeier, D.F. Gleich, R. Malik, Dynamic competition networks: detecting alliances and leaders. In: Proceedings of Algorithms and Models for the Web Graph (WAW'18) (2018)
6. A. Bonato, N. Eikmeier, D.F. Gleich, R. Malik, Centrality in dynamic competition networks. In: Proceedings of the International Conference on Complex Networks and Their Applications (2019)
7. de Waal, F.B.M.: The organization of agonistic relations within two captive groups of Java-monkeys (Macaca fascicularis). Z. Tierpsychol **44**, 225–282 (1977)
8. A.M. Grant, F. Gino, D.A. Hofmann, The hidden advantages of quiet bosses. Harvard Bus. Rev. **88**(12) (2010)
9. Guo, W., Lu, X., Donate, G.M., Johnson, S.: The spatial ecology of war and peace (2022). arXiv:1604.01693
10. Kumar, S., Hooi, B., Makhija, D., Kumar, M., Subrahmanian, V.S., Faloutsos, C.: REV2: fraudulent user prediction in rating platforms. In: 11th ACM International Conference on Web Search and Data Mining (WSDM) (2018)
11. Leskovec, J.: The Stanford large network dataset collection. http://snap.stanford.edu/data/index.html
12. Li, Y., Wu, X., Yang, S.: Social network dominance based on analysis of asymmetry. In: Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (2016)
13. Montoya, J.M., Pimm, S.L., Solé, R.V.: Ecological networks and their fragility. Nature **442**, 259–264 (2006)
14. Poisbleau, M., Fritz, H., Guillemain, M., Lacroix, A.: Testosterone and linear social dominance status in captive male dabbling ducks in winter. Ethology **111**, 493–509 (2005)
15. Prałat, P., Janssen, J.: Rank-based attachment leads to power law graphs. SIAM J. Discrete Math. **24**, 420–440 (2010)
16. Shizuka, D., McDonald, D.B.: Data from: the network motif architecture of dominance hierarchies. Dryad Digital Repository (2015)
17. Survivor wiki, Survivor: Heroes versus Healers versus Hustlers. https://survivor.fandom.com/wiki/Survivor:Heroes versus Healers versus Hustlers
18. UN comtrade data. United Nation (2022). http://comtrade.un.org/
19. Watt, D.J.: Relationship of plumage variability, size and sex to social dominance in Harris' sparrows. Anim. Behav. **34**, 16–27 (1986)
20. West, D.B.: Introduction to Graph Theory, 2nd ed. Prentice Hall (2001)
21. Zachary, W.W.: An information flow model for conflict and fission in small groups. J. Anthropol. Res. **33**, 452–473 (1977)

Reconstructing Degree Distribution and Triangle Counts from Edge-Sampled Graphs

Naomi A. Arnold, Raúl J. Mondragón, and Richard G. Clegg

Abstract Often, due to prohibitively large size or to limits to data collecting APIs, it is not possible to work with a complete network dataset and sampling is required. A type of sampling which is consistent with Twitter API restrictions is uniform edge sampling. In this paper, we propose a methodology for the recovery of two fundamental network properties from an edge-sampled network: the degree distribution and the triangle count (we estimate the totals for the network and the counts associated with each edge). We use a Bayesian approach and show a range of methods for constructing a prior which does not require assumptions about the original network. Our approach is tested on two synthetic and two real datasets with diverse degree and triangle count distributions.

Keywords Network reconstruction · Bayesian statistics · Sampling

1 Introduction

Analysis of complex networks remains a growing area and network data sets are more and more commonly available. However, some data sets are only a sample of the entire network. For very large networks, it may not be possible to work with complete data because of its size. Additionally, APIs can rate-limit the number of queries, meaning that not all nodes and edges are present [15]. A common example is the Twitter stream API which returns a 1% random sample of all tweets in real-time [21]. In the usual Twitter graph formulation where edges constitute 1:1 replies or retweets, this corresponds to uniform edge sampling of the full Twitter reply/retweet graph. Inferring even simple characteristics such as the true number of nodes or edges from a sample can be nontrivial [6, 12].

In this work we present a methodology for recovering the degree sequence and the triangle sequence (per edge) under a uniform edge-sampling scenario where for

N. A. Arnold (✉) · R. J. Mondragón · R. G. Clegg
Queen Mary University of London, London E1 4NS, UK
e-mail: n.a.arnold@qmul.ac.uk

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_23

an undirected graph G, a sample is constructed by uniformly sampling each edge of G with probability p. First, we build on methods by Ganguly et al. [11] who recover the degree distribution from node-sampled networks using a Bayesian approach and we extend this to edge-sampled networks. We address the problem of finding an appropriate prior degree distribution by proposing two different ways to construct a prior. We further extend this Bayesian approach to estimating the edge triangle count (the number of triangles associated with each edge) and the total triangle count.

We find that our Bayesian method outperforms the standard scale-up method at estimating the degree sequence, particularly in small p scenarios where as few as 10% of the edges remain. Moreover, the priors we use do not make any assumptions about the original degree distribution. For estimating the triangle per link count, in 3 out of the 4 network datasets we use, a Poisson prior achieves similar performance to a correct prior.

This paper is structured as follows. First, in Sect. 3 we describe the edge sampling procedure and derive properties of graphs that have been sampled in this way. Then in Sect. 4 we introduce the various estimators used for these properties, with Sect. 5 showing how to construct a prior for the Bayes estimators. Finally in Sect. 6 we present our results on recovering these properties on synthetic and real datasets. We discuss the implications in Sect. 7.

2 Related Work

Sampling of complex networks in general is a well studied problem. One point of interest is how well sampling preserves different properties, such as node rankings in Twitter networks [15], temporal features [1] and scaling properties [14]. These works have aimed also at designing sampling schemes specifically to preserve a given quantity. Other works have used sampling to estimate quantities on graphs that are prohibitively large to work with in their entirety, with a focus on triangle counting [2, 18, 20] or other motifs [5, 13].

Two recent works studied the problem of recovering a network's degree distribution working from a small sample, first posed by Frank [10] in his PhD thesis in 1971. The first by Zhang et al. [24] frames it as an inverse problem involving the vector of observed degree counts and a linear operator representing the sampling scheme. The second by Ganguly et al. [11] uses a range of estimators for individual vertex degrees in node-sampled networks; simple scale-up estimators, risk minimisation estimators and Bayes posterior estimates. Antunes et al. [2] whose work was on sampling methods for estimating the triangle distribution, studied the $n = 1$ sample size problem as restricted access scenario as a case study. Other than this, little attention has been given specifically to these restricted access problems, noted in [24].

A related problem is reconstructing network structure from unreliable or noisy data, such as social networks constructed from reported friendships, which are well known for having missing or spurious edges due to the different interpretations of

"friendship" [23]. Young et al. [23] address this using a Bayesian approach for finding posterior probabilities for an edge's existence given the measurements obtained. Newman [17] use a Bayesian approach involving the empirical false and true positive rates of observing an edge from the data.

3 Properties of Edge Sampled Graphs

Let $G = (V, E)$ be an undirected simple graph with vertex set $V = \{v_1, \ldots, v_N\}$ and edge set $E = \{e_1, \ldots, e_M\}$. Consider a sampling regime where each edge $e_l \in E$ is included in the sampled graph with probability $p \in [0, 1]$, and each vertex $v_i \in V$ is included if any edge incident to it is included. Denote this sampled graph $G' = (V', E')$, where $V' \subseteq V$ and $E' \subseteq E$. Let the sizes of V' and E' be N' and M' respectively. This is known as *incident subgraph sampling* [14].

3.1 Degree

Let k_i, respectively k_i' denote the degree of a node $v_i \in G$ and G' respectively. Then k_i' follows a binomial distribution $k_i' \sim B(k_i, p)$, with conditional probability given by

$$\mathbb{P}(k_i' = k' | k_i = k) = \binom{k}{k'} p^{k'} (1-p)^{k-k'}, \tag{1}$$

with expectation $\mathbb{E}(k_i' | k_i = k) = kp$ and variance $\mathrm{Var}(k_i' | k_i = k) = kp(1-p)$. The probability that node $v_i \in V$ of degree k_i has degree 0 in G' is given by $\mathbb{P}(k_i' = 0) = (1-p)^{k_i}$.

Nodes in G that become isolated as part of the sampling process are invisible to observers of G' and should be considered removed. In this way, let δ_i be the indicator random variable representing the removal of node v_i from G, with probability $(1-p)^{k_i}$.

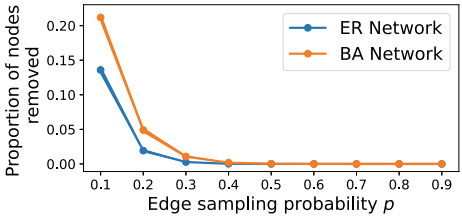


Fig. 1 Proportion of nodes removed in Erdős-Rényi and Barabási-Albert graphs, of size 1000 nodes and 10,000 (ER) and 9900 (BA) edges (see Table 1), using the edge-sampling procedure

Then, the expected number N_0 of removed nodes from G is given by $\mathbb{E}(N_0) = \sum_{i=1}^{N} \mathbb{E}(\delta_i) = \sum_{k\geq 1}(1-p)^k N_k$, where N_k is the number of vertices in G of degree k. This is dependent on the degree distribution; distributions with large numbers of low-degree nodes will experience higher numbers of nodes removed. This brings to mind the friendship paradox [9], where a node incident to a randomly chosen edge will on average have a higher degree than a randomly chosen node. From Fig. 1 we see Barabási-Albert [4] networks experience more node removal than Erdős-Rényi [8] networks.

The variance in the number of nodes removed is given by

$$\begin{aligned}\mathrm{Var}(N_0) &= \mathrm{Var}\left(\sum_{i=1}^{N}\delta_i\right) = \sum_{i=1}^{N}\mathrm{Var}(\delta_i) + \sum_{1\leq i\neq j\leq N}\mathrm{Cov}(\delta_i,\delta_j)\\ &= \sum_{k\geq 0}(1-p)^k\left[1-(1-p)^k\right]N_k\\ &\quad + \sum_{k,k'\geq 0}\left[(1-p^{k+k'-1}-(1-p)^{k+k'}\right]N_{k,k'}\end{aligned}$$

where $N_{k,k'}$ is the number of edges connecting vertices of degree k and k'.

3.2 Triangles

Let T_l be the number of triangles in G which include edge $e_l \in E$. Then the number of triangles in G, denoted by T, is given by

$$T = \frac{1}{3}\sum_{e_l\in E} T_l \tag{2}$$

where the factor of $\frac{1}{3}$ is present because each triangle in the sum is counted three times, once for each link.

Let T_l' be the number of triangles which include edge e_l in the sampled graph G', defining $T_l' = 0$ if $e_l \notin E'$. In the case that edge e_l remains in the sampled network, then each triangle that includes e_l will remain in the sampled network if and only if the other two edges remain; this occurs with probability p^2. There are T_l such triangles, so the number of these which remain in the sampled network is binomially distributed with T_l trials and probability p^2. That is,

$$\mathbb{P}(T_l' = t' | T_l = t, e_l \in E') = \binom{t}{t'} p^{2t'}(1-p^2)^{t-t'}. \tag{3}$$

In the case that e_l does not remain in the sampled network, the following holds:

$$\mathbb{P}(T_l' = t' | T_l = t, e_l \notin E') = \delta_{0,t'} \tag{4}$$

where $\delta_{0,t'}$ is the Kronecker delta function, taking the value of 1 if $t' = 0$ and 0 otherwise (since we defined $T_l' = 0$ if $e_l \notin E'$).

We can use the law of total probability to remove the conditioning on e_l from Eqs. (3) and (4) and find $\mathbb{P}(T_l' = t')$ as follows,

$$\begin{aligned}\mathbb{P}(T_l' = t' | T_l = t) &= \mathbb{P}(T_l' = t' | T_l = t, e_l \in E')\mathbb{P}(e_l \in E') \\ &\quad + \mathbb{P}(T_l' = t' | T_l = t, e_l \notin E')\mathbb{P}(e_l \notin E') \\ &= p\binom{t}{t'} p^{2t'}(1-p^2)^{t-t'} + \delta_{0,t'}(1-p). \end{aligned} \tag{5}$$

Therefore, the conditional probability mass function for T_l' given T_l is given by Eq. (5).

The expected value of T_l' given T_l is given by

$$\begin{aligned}\mathbb{E}(T_l' | T_l = t) &= \sum_{t'=0}^{t} t' \mathbb{P}(T_l' = t' | T_l = t) \\ &= \sum_{t'=0}^{t} t' \left[p\binom{t}{t'} p^{2t'}(1-p^2)^{t-t'} + \delta_{0,t'}(1-p) \right] \\ &= p \sum_{t'=0}^{t} t' \binom{t}{t'} p^{2t'}(1-p^2)^{t-t'} = p \times p^2 t = p^3 t \end{aligned} \tag{6}$$

where Eq. (6) comes from noting that the sum in the lhs precisely evaluates the expected value of a binomial random variable with t trials and probability p^2.

Let T' be a random variable representing the triangle count of G', then

$$\mathbb{E}(T') = \mathbb{E}\left[\frac{1}{3} \sum_{e_l \in E} T_l' \right] = \frac{1}{3} \sum_{e_l \in E} p^3 T_l = p^3 \, T$$

where T' is the triangle count of the sampled network G'.

The variance of T_l' given T_l is then

$$\mathrm{Var}(T_l' | T_l = t) = p^3 t (1 - p^2 + p^2 t - p^3 t). \tag{7}$$

An argument involving computation of the covariances $\mathrm{Cov}(T_j, T_l)$ shows that the variance of the expected total triangle count of G' given the individual triangle counts $T_1, \ldots, T_M$ is given by

$$\mathrm{Var}(T'|T_1,\ldots,T_M) = \frac{1}{9}\Bigg[3p^3(1-p^2)T + (p^3-p^2)\sum_{e_l\in E} T_l^2 + 6T(p^3-p^6) + 8k(p^5-p^6)\Bigg]. \tag{8}$$

where k is the number of triangles which share a link. Full derivations of Eqs. (7) and (8) can be found in the first author's thesis [3] which also contains derivations for wedge counts and clustering coefficient. An expression for this variance conditioned on total triangle count T not edge triangle counts is given in [20].

The number of triangles per node T_i can be obtained from T_{e_ℓ} from $2T_i = \sum_k T_{e_\ell=(i,k)\in E}$ meaning the estimators of the edge-sampled network can be extended to evaluate vertex statistics e.g. local transitivity of the nodes $c_i = \sum_k T_{e_\ell=(i,k)\in E}/(k_i(k_i-1))$.

4 Estimators for the Degree Sequence and Triangle Count

The previous showed how the distribution of a quantity X' in a sampled graph G' could be calculated as a conditional probability $P(X'=x'|X=x)$ given the unsampled measurement $X=x$. This section aims to estimate the true network quantity X given its sampled counterpart X'.

4.1 Method of Moments Estimators

Let X be a random variable associated with a statistic of G and let X' be that statistic on G' with expected value $\mathbb{E}(X') = f(X,p)$. A naive 'scale-up' estimator for X given observed value x' for X' is the solution $\hat{x}$ to the equation $x' = f(\hat{x},p)$, provided a solution exists. Borrowing the terminology from [11], we will refer to these estimators as *method of moments estimators*(MME).

4.1.1 Degree

For a node of degree k_i' in G', the MME for k_i is given by k_i'/p. This is an unbiased estimator with mean $\mathbb{E}\,(k_i'/p) = \frac{1}{p}k_i\,p = k_i$ and variance $\mathrm{Var}(k_i'/p) = \frac{1}{p}k(1-p)$. Nodes with the lowest possible degree (one) in the sampled graph are estimated as having degree $1/p$ in the unsampled graph so as p decreases, the estimation of low-degree nodes becomes poorer.

4.1.2 Triangle Count

The expected triangle count $\mathbb{E}(T_l')$ of edge e_l is $p^3 T_l$. If in addition, e_l remains in G', its expected triangle count is given by $p^2 T_l$. Therefore the MME for T_l is $p^{-3} T_l'$ or $p^{-2} T_l$, without and with the conditioning respectively. Similar to the MME for degree, it provides poor estimates for edges that have a low triangle count, as it disallows any estimates of T_l in the range $(0, 1/p^2)$.

Similarly, an MME estimate proposed by Tsourakakis et al. [20] for the total triangle count of a network is $p^{-3} T'$, which has expected value $\mathbb{E}\,(p^{-3} T') = T$. They found that this estimator has variance $\frac{1}{p^6}\left((p^3 - p^6)T + 2k(p^5 - p^6)\right)$, where k is the number of pairs of triangles which share a link.

4.2 *Bayes Estimator*

This estimator relies on Bayes theorem, giving

$$\mathbb{P}(X = x | X' = x') = \frac{\mathbb{P}(X' = x' | X = x)\mathbb{P}(X = x)}{\mathbb{P}(X' = x')}. \tag{9}$$

$\mathbb{P}(X' = x' | X = x)$ is the *likelihood* which is determined by the edge sampling procedure and is known. $P(X = x)$ is the *prior* function which will be denoted by $\pi(x)$; this is in general not known.

A posterior estimate for X given X' can then be given as the expected value

$$\mathbb{E}\,(X | X' = x') = \frac{\sum_x x \mathbb{P}(X' = x' | X = x)\pi(x)}{\mathbb{P}(X' = x')}. \tag{10}$$

The immediate question arises of how to deal with the prior $\pi(x)$, as this may involve making assumptions about the structure of G. This will be discussed case by case for the degree and triangle count.

4.2.1 Degree

Using the likelihood function for the degree from Eq. (1), a posterior estimate for the degree of node v_i given it has degree k_i' in G' is

$$\mathbb{E}(k_i | k_i' = k') = \frac{\sum_{k=k'}^{\infty} k \binom{k}{k'} (1-p)^k \pi(k)}{\sum_{k=k'}^{\infty} \binom{k}{k'} (1-p)^k \pi(k)} \tag{11}$$

where $\pi(k)$ is a prior for the degree distribution $P(k)$ of G.

4.2.2 Triangle Count

Using the likelihood function from Eq. (3), a posterior estimate for the triangle count of edge e_l in G given it remains in G' is

$$\mathbb{E}(T_l|T_l' = t', e_l \in G') = \frac{\sum_{t=t'}^{\infty} t\binom{t}{t'}(1-p^2)^t \pi(t)}{\sum_{t=0}^{\infty}\binom{t}{t'}(1-p^2)^t \pi(t)} \tag{12}$$

where $\pi(t)$ is a prior for the proportion of edges with triangle count t.[1]

To establish the total triangle count, summing the value of this estimator over the remaining edges in G' and dividing by 3, as in Eq. (2), will provide an underestimate for the total triangle count of G, since there are potentially many missing edges in G'. To mitigate this, we scale this factor up to the estimated number of edges in G. That is, our estimate of the total triangle count becomes

$$\hat{T} = \frac{1}{3p}\sum_{e_l \in G'} \hat{T}_l.$$

5 Constructing a Prior

The Bayes estimators for the degree and triangle per link sequences require the choice of a prior. In this section, we propose methods for constructing priors.

5.1 Degree Distribution

A prior could be obtained from chosen family of distributions such as the Zipf distribution or a power law distribution, but this baked-in assumption may not be desirable. Furthermore, it has been shown that the distribution of a sampled network may not even follow the distribution of the true network [19]. Therefore, we propose two different methods of constructing a prior which do not make assumptions about the degree distribution of the true network.

First, it is possible to estimate the prior using a Monte Carlo method to minimise the ℓ_2 norm of the error. In this approach, we find a degree sequence $\{\kappa_i\}$ which minimises $\min\left(||p\kappa_i - k_i'||_2^2\right)$, with the restrictions that the degree is an integer number and the sum of the degrees is equal to twice the number of links. To do this, we start with $\kappa_i = \lfloor k_i'/p \rfloor$. If the sum $\sum_{v_i \in V'} \kappa_i$ of the estimated degrees is not equal to the estimated number of links $\lfloor 2M'/p \rfloor$ then we increment or decrement the degree of

[1] In experimental runs, the native binomial functions introduced numerical inaccuracies for large powers of $(1-p)$. Therefore, an equivalent evaluation of binomial probabilities using the log-gamma function and laws of logs was used in practice.

nodes chosen uniformly at random until equality holds. Then we rewire links at random for a large number of iterations (15,000 in our case), accepting each proposed rewire if it decreases the ℓ_2 error. If $1/p$ is an integer, then the MME $\kappa_i = k'_i/p$ is a global minimum.

The MME (and hence sometimes the minimisation method) cannot estimate the degree $k_i \approx k'_i/p$ when $k'_i = 0$; that is, the lowest possible estimated degree is $1/p$. If a good estimate for the original number of nodes is known then another prior we for capturing these low degree nodes is constructed by "cascading" links from high degree to low degree nodes. More precisely, as with the minimisation method, we start with an estimated degree sequence $\kappa_i = \lfloor k'_i/p \rfloor$, redistributing links as before if the total estimated degree does not match twice the estimated number of links. Then, we place the nodes in descending order based on their estimated degree, with the knowledge of the original number of nodes in the network being used to append placeholder nodes which would have been removed by the sampling process. Finally, we pick the first occurring node in this list with degree zero, and increment this degree by simultaneously decrementing the degree of the node directly before it. This step is performed iteratively until there are no degree zero nodes. Finally, as a comparison point representing the best possible result achievable with the Bayes method, we use a true prior which is the degree frequencies of the original network as a probability distribution.

5.2 Triangle per Link Distribution

The two methods for prior construction of the degree distribution do not immediately translate to an analogue for triangles, and little is known about the triangle per edge distribution as a starting place for selecting a prior. As an initial approach therefore, we use a Poisson distribution $\mathrm{Po}(\lambda)$ with $\lambda = 3\hat{T}/M'$, the average number of triangles per link in the MME estimator. As with the degree distribution, we include a result with a true prior as a comparison point.

6 Results

To test the capability of our estimators of degree sequence and triangle count, we consider four different starting networks: an Erdős–Rényi $G(N, M)$ network [8] with $N = 1000$ and $M = 10{,}000$, a Barabási–Albert network [4] of approximately the same size and density, a real collaboration network from authors who submitted to the ArXiV high-energy theoretical physics category [16] (henceforth Hep-Th for brevity) and an Internet autonomous systems topology dataset (henceforth AS) [7]. A quick reference of some summary statistics can be found in Table 1. These datasets were chosen to represent a heterogeneous selection of network types. The ER network has a Poisson degree and edge triangle count distribution and an overall low number

Table 1 Original statistics of network datasets used prior to sampling. Shown is the number of nodes N, number of edges M, average node clustering coefficient $\bar{C}$ [22], degree assortativity ρ and average number of triangles per edge $\bar{T}_l$

Dataset	N	M	$\bar{C}$	ρ	$\bar{T}_l$
Erdős-Rényi	1000	10,000	0.019	0.021	0.39
Barabási-Albert	1000	9900	0.063	−0.038	1.91
Cit-Hep-Th collaborations	5835	13,815	0.506	0.185	2.31
AS topology	11,174	23,409	0.296	−0.195	2.55

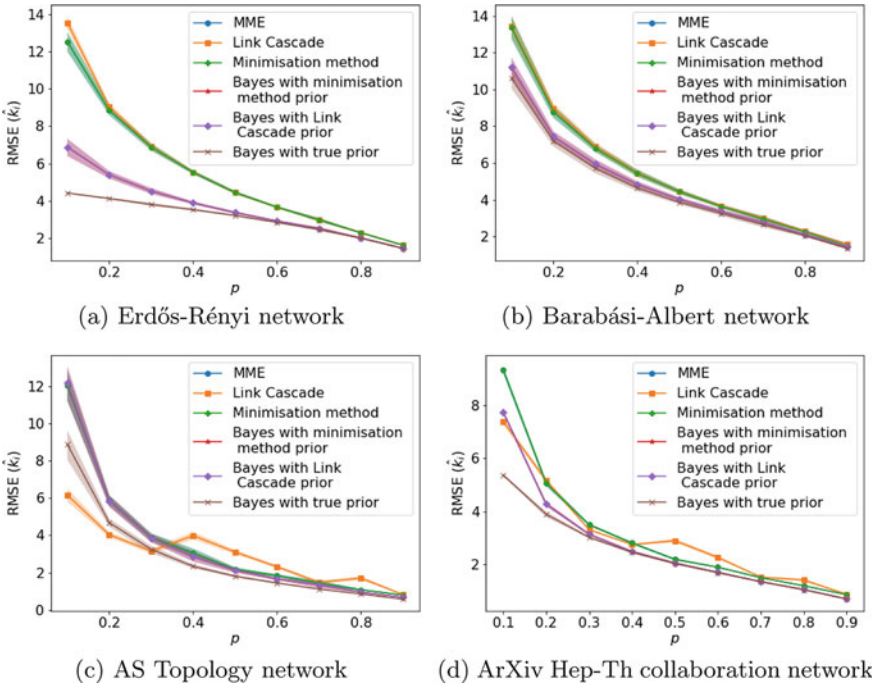


Fig. 2 Error in estimation of the true degree sequence. Each value is averaged over experiments with the shaded error bars representing standard deviation. The MME is overlays the minimisation method in all

of triangles for the density of the network. The BA network has a theoretically power law degree distribution, and a low triangle count for its density. The Hep-Th and AS networks have a heavy-tailed degree distribution but have very different degree correlations and the Hep-Th has a higher clustering than the AS network. For each of these datasets modelled as a graph G, we take an edge-sampled network G' with edge sampling probability p, for $p = 0.1, 0.2, \ldots, 0.9$ and from this, reconstruct the degree sequences, edge triangle counts and total triangle counts using our estimators.

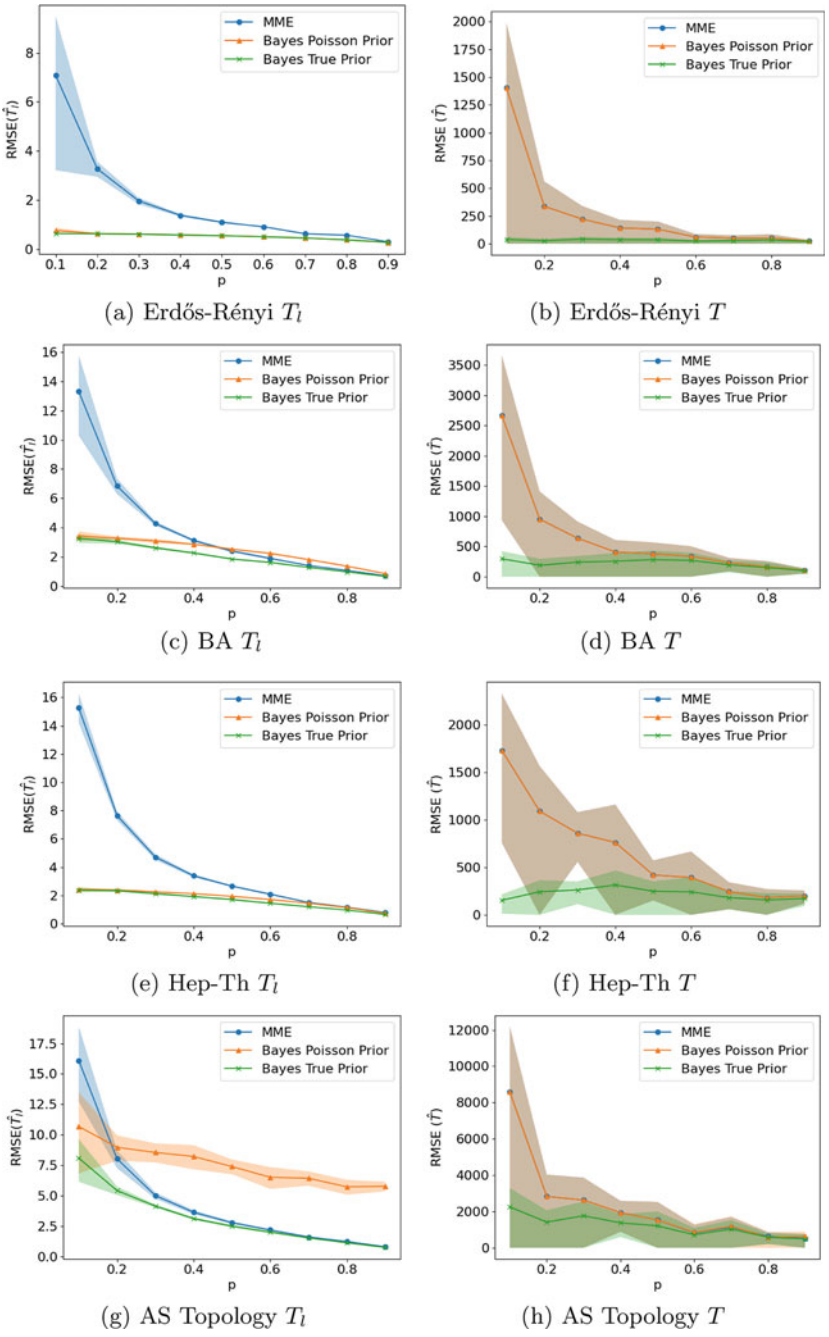


Fig. 3 Error in estimation of the triangles per link sequence using our different approaches, and total triangles. Each value is averaged over experiments with the shaded error bars representing standard deviation. The MME overlays the Bayes estimator with Poisson prior in the right hand column

In the degree distribution experiment, we reconstruct the degree k of nodes in V' using our chosen estimators $\hat{k}$, and compute the root mean squared error of the degree sequences as $\text{RMSE}(\hat{k}) = \sqrt{\frac{1}{N'}\sum_{v_i \in V'}(k_i - \hat{k}_i)^2}$. These results are shown in Fig. 2, showing the mean and s.d. error over 10 experiments. In all but the AS topology network, the Bayes estimator with true prior has the lowest error, though this is included only to show the best possible result that could be obtained with the Bayes method since the true prior is unknowable. The next approaches that do well at this task are the "link cascade" method and the Bayes estimator using the link cascade as a prior. This method assumes knowledge of the number of nodes in G (i.e. the number of nodes pruned by the edge-sampling) so performs better at estimating low degree nodes. This is particularly evident in Fig. 2c, performing better than the Bayes approach with true prior. The Monte Carlo minimisation method on its own in many cases overlays the MME, due to the restriction that the degree sequence is an integer (c.f. Sect. 5).

In the triangle count experiment, we estimate the triangle per edge count $\hat{T}_l$ for edges $e_l \in E'$ and compute the mean squared error as $\text{RMSE}(\hat{\mathbf{T}}) = \left[\frac{1}{M'}\sum_{e_l \in E'}(T_l - \hat{T}_l)^2\right]^{\frac{1}{2}}$. In addition, we estimate the total number of triangles as described in Sect. 4 and calculate the mean squared error over the 10 experiments performed. These are shown in Fig. 3 with the triangle per link error on the left column and total triangle error on the right. In all experiments, the Bayes estimator with Poisson prior overlays the MME for total number of triangles; this is because the λ used in the Poisson distribution is the MME estimate of the average number of triangles per link. However, in all but the AS topology, the Poisson prior improves the estimate of triangles per link especially in the small p scenario. In the AS topology dataset, the Poisson is an inappropriate prior, performing poorly even with large sample sizes.

7 Conclusion

This paper provided methods for recovering statistics from networks sampled via uniform edge sampling such as graphs limited to a sample by the Twitter API. Our results show that our Bayesian estimators perform much better than standard approaches on the degree sequence even when the priors were constructed without knowledge of distributions for the original network. For the triangle count per edge, we showed that while the Bayes estimates do not always improve upon the MME for total triangle counts, they provide a markedly better estimate of triangles per link in the small p scenario. However, an inappropriate choice of prior can lead to a bias even when the sample size is large.

Future work will investigate generalising methods we used for constructing a degree distribution prior for constructing a prior for triangle counts per link. One can also consider other sampling regimes and network properties for which a likelihood can be calculated.

References

1. Ahmed, N.K., Neville, J., Kompella, R.: Network sampling: From static to streaming graphs. ACM Trans. Knowl. Discov. Data (2013)
2. Antunes, N., Guo, T., Pipiras, V.: Sampling methods and estimation of triangle count distributions in large networks. Netw. Sci. (2021)
3. Arnold, N.: Studying evolving complex networks. Ph.D. thesis, Queen Mary University of London (2021)
4. Barabási, A.-L., Albert, R.: Emergence of scaling in random networks. Science (1999)
5. Bhattacharya, B.B., Das, S., Mukherjee, S.: Motif estimation via subgraph sampling: the fourth-moment phenomenon. Annals Stat. **50**(2), 987–1011 (2022)
6. Bianconi, G.: Grand canonical ensembles of sparse networks and Bayesian inference. Entropy (2022)
7. Chen, Q., Chang, H., Govindan, R., Jamin, S.: The origin of power laws in internet topologies revisited. In: Proceedings of IEEE Computing and Communication Societies (2002)
8. Erdős, P., Rényi, A., et al.: On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci. (1960)
9. Feld, S.L.: Why your friends have more friends than you do. Am. J. Sociol. (1991)
10. Frank, O.: Statistical inference in graphs. Ph.D. thesis, Foa Repro Stockholm (1971)
11. Ganguly, A., Kolaczyk, E.D.: Estimation of vertex degrees in a sampled network. In: Asilomar Conference on Signals, Systems, and Computers (2017)
12. Katzir, L., Liberty, E., Somekh, O.: Estimating sizes of social networks via biased sampling. In: Proceedings on International Conference on World Wide Web (2011)
13. Klusowski JM, Wu, J.: Counting motifs with graph sampling. In: Conference on Learning Theory, pp. 1966–2011. PMLR (2018)
14. Leskovec, J., Faloutsos, C.: Sampling from large graphs. In: Proceedings of the International Conference on Knowledge Discovery and Data Mining (2006)
15. Morstatter, F., Pfeffer, J., Liu, H., Carley, K.: Is the sample good enough? comparing data from Twitter's streaming API with Twitter's firehose. In: Proceedings of the International AAAI Conference on Web and Social Media (2013)
16. Newman, M.E.: The structure of scientific collaboration networks. Proceedings of the National Academy of Sciences (2001)
17. Newman, M.E.: Network structure from rich but noisy data. Nat. Phys. **14**(6), 542–545 (2018)
18. Stefani, L.D., Epasto, A., Riondato, M., Upfal, E.: Triest: counting local and global triangles in fully dynamic streams with fixed memory size. ACM Trans. Knowl. Discov. Data (TKDD) (2017)
19. Stumpf, M.P., Wiuf, C., May, R.M.: Subnets of scale-free networks are not scale-free: sampling properties of networks. PNAS (2005)
20. Tsourakakis, C.E., Kang, U., Miller, G.L., Faloutsos, C.: Doulion: counting triangles in massive graphs with a coin. In: Proceedings International Conference on Knowledge Discovery and Data Mining (2009)
21. Twitter: Stream Tweets in real-time: developer documentation (2022). https://developer.twitter.com/en/docs/tutorials/stream-tweets-in-real-time
22. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature (1998)
23. Young, J.-G., Cantwell, G.T., Newman, M.: Bayesian inference of network structure from unreliable data. J. Complex Netw. (2020)
24. Zhang, Y., Kolaczyk, E.D., Spencer, B.D.: Estimating network degree distributions under sampling: an inverse problem, with applications to monitoring social media networks. Annals Appl. Stat. (2015)

Generalizing Homophily to Simplicial Complexes

Arnab Sarker, Natalie Northrup, and Ali Jadbabaie

Abstract Group interactions occur frequently in social settings, yet their properties beyond pairwise relationships in network models remain unexplored. In this work, we study homophily, the nearly ubiquitous phenomena wherein similar individuals are more likely than random to form connections with one another, and define it on simplicial complexes, a generalization of network models that goes beyond dyadic interactions. While some group homophily definitions have been proposed in the literature, we provide theoretical and empirical evidence that prior definitions mostly inherit properties of homophily in pairwise interactions rather than capture the homophily of group dynamics. Hence, we propose a new measure, k-simplicial homophily, which properly identifies homophily in group dynamics. Across 16 empirical networks, k-simplicial homophily provides information uncorrelated with homophily measures on pairwise interactions. Moreover, we show the empirical value of k-simplicial homophily in identifying when metadata on nodes is useful for predicting group interactions, whereas previous measures are uninformative.

Keywords Social network analysis · Homophily · Simplicial complexes

1 Introduction

Group interactions fundamentally differ from interactions between pairs of individuals. When individuals assemble in groups of size three or more, social pressure increases [2], social loafing may occur [16], and joint decisions can become polar-

A. Sarker (✉) · N. Northrup · A. Jadbabaie
Massachusetts Institute of Technology, Cambridge, MA, USA
e-mail: arnabs@mit.edu

N. Northrup
e-mail: natnorth@mit.edu

A. Jadbabaie
e-mail: jadbabai@mit.edu

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_24

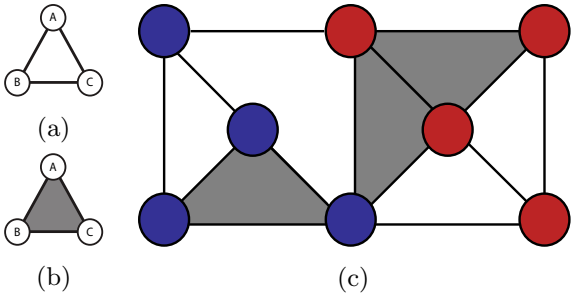


Fig. 1 **a** A closed but not filled triangle, indicating only pairwise interactions (e.g., three separate two-author papers). **b** A closed and filled triangle, indicating a group interaction (e.g., a single paper by three authors, which would additionally indicate that all pairs of authors have co-authored with one another). **c** Example where group homophily is inherited from edge structure. If nodes were randomly labeled, then $1/7$ filled triangles would have nodes of the same type on average (c.f. (3)), suggesting the presence of homophily (2 out of 3 filled triangles have nodes of the same type, and $2/3 > 1/7$). However, the edge structure of the network is such that 4 out of 6 closed (filled or unfilled) triangles have nodes of the same type, which suggests that the homophily of filled triangles is as if they were randomly placed into the underlying edge structure

ized [5]. However, fundamental properties of group interactions in complex networks are not yet fully explored. As such, *higher order* models, which explicitly encode group interactions with data structures such as simplicial complexes and hypergraphs, have received attention in recent literature [3, 4, 23].

In this work, we consider the principle of homophily as it pertains to group interactions. Homophily, the well-known tendency for individuals to form social connections with those similar to themselves, is a core organizing principle of social networks [17, 18]. This notion is nearly ubiquitous, appearing in contexts such as marriage, friendship, information transfer, physical contact, and online social networks [14, 18, 24]. In such networks, social ties are correlated with similarity in age, occupation, religion, and/or each individual's local network structure [6, 18]. Although this empirical ubiquity of homophily makes it valuable in understanding social structure, previous studies have restricted to analysis of graphs, which only encode pairwise interactions between individuals.

Our work builds on group homophily definitions considered recently in the context of hypergraphs, a generalization of graphs that can encode interactions between arbitrarily large groups of individuals [25]. For a particular hypergraph with labeled nodes, prior work considers all hyperedges of fixed size $g \geq 3$, and defines homophily relative to if nodes were labeled at random, which we refer to as a *node baseline*. However, this approach can potentially inherit the dyadic, graph-based notion of homophily rather than that of group interactions. In other words, much of the variation of group homophily scores with a node baseline can be explained by the standard dyadic notion (Fig. 1). This observation goes beyond the provided example: in the 16 empirical datasets of this work, nearly 70% of the variation in group homophily (for groups of size 3) using a node baseline can be explained by homophily scores

defined only on edges. Hence, we introduce a new measure, k-simplicial homophily, which properly isolates homophily due to group dynamics.

Contributions. In Sect. 3, we precisely define k-simplicial homophily as a formal way to account for underlying interactions in a network when establishing the presence of homophily for groups. Rather than model the social network with a hypergraph, we use a simplicial complex which requires additional structure in the network model. We establish theoretically that when k-simplicial homophily is applied to edges, we recover a standard definition of homophily on graphs, suggesting it is a natural generalization of homophily for groups.

Furthermore, contrary to the existing notions of homophily, k-simplicial homophily successfully isolates properties of group dynamics. We provide theoretical evidence of this in Sect. 4, where we introduce the simplicial stochastic block model, a generative network model which allows for homophily in pairwise interactions to be decoupled from that of triadic interactions. We show prior measures can incorrectly conclude the presence of group homophily, whereas k-simplicial homophily identifies group homophily if and only if the formation of triadic interactions depends on node class labels.

We then apply group homophily definitions to empirical data. In 15 out of 16 empirical datasets, we find that homophily scores using k-simplicial homophily are lower than scores computed with the node baseline. Moreover, in 4 of these datasets, we find anti-homophily with respect to k-simplicial homophily, and note that the anti-homophily is justified in each dataset. Importantly, we do not find a significant relationship between edge homophily scores and k-simplicial homophily scores on triangles, suggesting that k-simplicial homophily provides novel insights into group dynamics.

In Sect. 5, we show the utility of the new information provided by k-simplicial homophily in the data-driven application of higher order link prediction. Originally proposed by Benson et al. [4] as a benchmark problem for higher order models and algorithms, higher order link prediction involves using network information up to a certain time t to predict if new group interactions will occur after time t. We find that k-simplicial homophily indicates whether node labels are useful in the prediction task, whereas previous definitions of group homophily are uninformative in determining the utility of node labels.

1.1 Related Work

Group homophily has been considered for data-driven applications such as transductive learning [22] and clustering [15]. In such works, the authors use a homophily parameter as an input into a generative hypergraph model, and homophily is defined relative to a baseline distribution computed using frequencies of node class labels. Here, we instead propose homophily measures which describe existing datasets to aid analysis of group interactions in empirical settings.

A particularly relevant work is Veldt et al. [25], as it aims to broadly define homophily in the context of hypergraphs. Like previous work, the baseline considered by the authors uses randomization of node labels in order to determine if hyperedges in a network are more likely than random to be among nodes of the same type. The author's main focus in the work is understanding the complexity that arises in group homophily due to the fact that different numbers of each category of individuals can be present in a particular group. That is, for the setting considered by the authors where nodes are given one of two labels, a hyperedge of size k can have t members of one group, and $k - t$ members of the other for any $0 \leq t \leq k$. For a fixed k, the authors define a homophily score for each t, and prove impossibility results showing their homophily scores can not be strictly increasing in t and can not be greater than unity for all $t \geq k/2$. In this work, we define similar metrics for homophily which are based on simplicial complexes as opposed to hypergraphs. We also consider a more general setting where three or more class labels are allowed, which helps to avoid impossibility results from prior work and allows for a broader selection of data.

2 Preliminaries

We discuss three data structures, each of which considers a set of nodes V, where $|V| = n$, and a labeling function $C : V \to \{1, \ldots, m\}$, which maps each node to one of $m \geq 2$ classes.

Graphs and Hypergraphs. Graphs and hypergraphs are common models of interactions in complex networks [3]. We consider undirected graphs which consist of a set of nodes V and a set of edges E, where each edge $e \in E$ denotes a pairwise interaction between nodes. Hypergraphs, in contrast, have a set of hyperedges $H \subseteq 2^V$ which are unrestricted in size. Hence, group interactions can be encoded as elements of H, with no additional structure required of H.

Simplicial Complexes. Simplicial complexes provide a way to encode group interactions which requires more structure than hypergraphs. A simplicial complex is a set of simplices $X \subseteq 2^V$, where each element $x \in X$ is referred to as a k-simplex if it contains $k + 1$ different elements of V. In Fig. 1, nodes would then correspond to 0-simplices, edges to 1-simplices, and filled triangles to 2-simplices. Simplicial complexes also have the following structural property:

$$x \in X \implies \sigma \in X, \ \forall \sigma \subseteq x \, .$$

That is, for every simplex x in X, all subsets of x must also be contained in the simplicial complex. In Fig. 1 the network can be modeled as a simplicial complex because for each filled triangle, all edges associated with the triangle are in the network. This simple assumption leads to a rich mathematical theory from algebraic topology [12]. While we will not discuss algebraic topology at length (instead, see [11, 12]), we do utilize the definition of a k-skeleton.

Definition 1 (k-skeleton [11]) For a simplicial complex X, let X^j denote the set of all j-simplices in X, i.e. those elements of X with exactly $j+1$ elements. The k-skeleton of X, denoted $X^{(k)}$, is defined

$$X^{(k)} = \bigcup_{j=0}^{k} X^j \,. \tag{1}$$

As we will see, the k-skeleton accounts for underlying interactions when defining group homophily, as it encodes all interactions of size at most $k+1$. For example, in Fig. 1, we used the 1-skeleton, referred to as the underlying graph, to argue that homophily in triadic interactions (filled triangles) can be inherited from homophily in closed triangles, which are defined by pairwise interactions.

3 Defining Group Homophily

Defining homophily for groups is far more complex than for edges, as there are significantly more options for node labels to be assigned in a group of size $g \geq 3$ than there are for an edge which only contains two nodes. To reduce this complexity, we focus on two types of homophily in this work: one based on the proportion of homogeneous groups in a network, and another which takes into account the number of individuals in a group which share a particular class label.

3.1 Homophily of Homogeneous Groups

In what follows, we refer to a group as homogeneous if all nodes share the same class. We use g to refer to group size in a hypergraph, and k to refer to k-simplices in a simplicial complex, which have size $g = k+1$. For an arbitrary hypergraph H, let H^g represent the hyperedges of size g and $H_h^g \subseteq H^g$ represent the homogeneous hyperedges of size g. The affinity score is then defined

$$a^g(H) = \left|H_h^g\right| \;/\; \left|H^g\right| \,. \tag{2}$$

The following random baseline formalizes notions of higher order homophily from previous literature [15, 22, 25], and can be applied to arbitrary hypergraphs:

$$b_h^g(H) = \sum_{c=1}^{m} \binom{n_c}{g} / \binom{n}{g}, \tag{3}$$

where n_c represents the number of individuals in class c, so $b_h^g(H)$ represents probability that a group of size g in H with random node labels is homogeneous.

Definition 2 (Hypergraph Homophily Score [25]) The hypergraph homophily score $s_h^g(H)$ is defined

$$s_h^g(H) = a^g(H) \,/\, b_h^g(H)\,. \tag{4}$$

The score $s_h^g(H)$ indicates the presence of homophily if $s_h^g(H) > 1$, or anti-homophily if $s_h^g(H) < 1$. The score also coincides with a traditional metric of graph homophily when $g = 2$, which we denote the *graph homophily score* [7].

The second random baseline applies only to simplicial complexes. Let $X^{(\bar{k-1}),k}$ represent the possible k-simplices that may occur in X.[1] The baseline is then

$$b_x^k(X) = a^{k+1}(X^{(\bar{k-1}),k})\,. \tag{5}$$

Intuitively, $b_x^k(X)$ is the probability that a randomly placed k-simplex into the $(k-1)$-skeleton of X is homogeneous. The corresponding homophily score is:

Definition 3 (k-Simplicial Homophily Score) The k-simplicial homophily score $s_x^k(X)$ is

$$s_x^k(X) = a^{k+1}(X) \,/\, b_x^k(X)\,. \tag{6}$$

The primary difference between k-simplicial homophily and hypergraph homophily lies in the definition of the baseline score. In hypergraph homophily, the baseline depends only on the composition of nodes, whereas for k-simplicial homophily, the $(k-1)$-skeleton accounts for the underlying interactions.

Example 1 In Fig. 1, consider the case $k = 2$, such that we are focused on triangles. Then, $X^{(k-1)}$ represents the underlying graph, and $X^{(\bar{k-1}),k}$ represents closed triangles in the underlying graph. Because 4 out of 6 closed triangles are homogeneous, $b_x^2(X) = 4/6$, and similarly because 2 out of 3 filled triangles are homogeneous, $a^3(X) = 2/3$. Therefore, $s_x^2(X) = 1$.

Notably, simplicial homophily and hypergraph homophily coincide when edges are the focus, as both generalize the standard definition of homophily in edges.

Proposition 1 *Let $G = (V, E)$ represent an undirected graph, and let $C : V \to \{1, \ldots, m\}$ represent a labeling of nodes into m classes. Then, the graph homophily score and the k-simplicial homophily score on edges coincide.*

The proof of the claim follows directly from the equivalence of (3) and (5) when applied to edges. Proposition 1 shows that k-simplicial homophily is actually a natural extension of graph-based notions of homophily [7, 20]. However, the two approaches to homophily differ when group size increases beyond 2, as edges in a simplicial complex can impose structure on triads.

[1] Formally, given $X^{(k-1)}$ is the $(k-1)$-skeleton of X, $X^{(\bar{k-1}),k}$ represents the maximal set of k-simplices which could be added to $X^{(k-1)}$ while preserving that $X^{(k-1)} \cup X^{(\bar{k-1}),k}$ is a simplicial complex.

3.2 Homophily in Heterogeneous Groups

While the scores of the previous section conveniently summarize homophily into a single value, they cannot handle heterogeneity in node labels for a group. To handle this distinction, we focus on type-t interactions as defined in [25]. For a class c and group size g, a type-t interaction is an interaction with exactly t members from class c. The type-t affinity score for class c is defined [25]:

$$a_c^g(t; H) = t \times \left|H_c^{t,g}\right| \; / \; \sum_{i=1}^{g} i \times \left|H_{h,c}^{i,g}\right| , \tag{7}$$

where $H_{h,c}^{i,g}$ is the set of type-i hyperedges for class c. The random baselines for the heterogeneous scores are then

$$b_h^g(t; H) = \frac{\binom{n_c-1}{t-1} \times \binom{n-n_c}{g-t}}{\binom{n-1}{g-t}} , \quad \text{and} \quad b_x^k(t; X) = a_c^{k+1}(t; X^{(\bar{k-1}),k}) , \tag{8}$$

where the former can be shown to be the expectation of $a_c^g(t; H)$ when node labels are assigned randomly, and the latter generalizes the randomization scheme of Sect. 3.1. The heterogenous homophily scores can then be defined as follows.

Definition 4 (Heterogeneous Homophily Scores) For a hypergraph H, group size g, and class c, the *heterogenous hypergraph homophily score* is [25]

$$s_{h,c}^g(t; H) = a_c^g(t; H) \; / \; b_h^g(t; H) . \tag{9}$$

For a simplicial complex X, the *heterogeneous k-simplicial homophily score* is

$$s_{x,c}^k(t; H) = a_c^{k+1}(t; X) \; / \; b_x^k(t; H) . \tag{10}$$

These definitions provide additional granularity when understanding homophily. However, we note such definitions are prone to impossibility results: when nodes are divided into two classes, heterogeneous homophily scores can not increase monotonically with the parameter t or exceed unity for all $t > g/2$. [25].

4 Homophily in Network Data

Synthetic Networks In order to show the difference in homophily definitions, we build upon recent models of random simplicial complexes to introduce the *simplicial stochastic block model*, a straightforward generalization of the Δ-ensemble of Kahle [13]. The generative model has the following inputs:

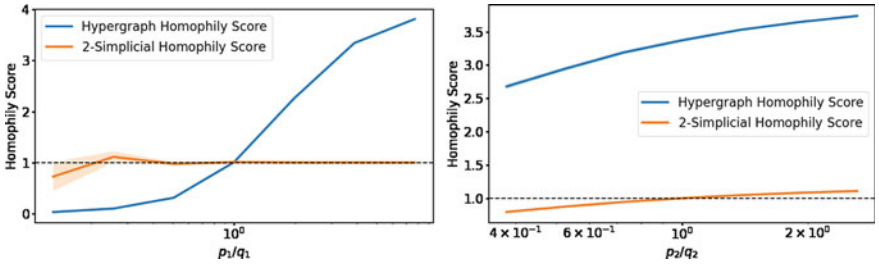


Fig. 2 Triadic homophily in the simplicial stochastic block model. Error bars represent 95% confidence intervals. (Left) Varying p_1/q_1 with $p_2 = q_2 = 0.5$. The hypergraph homophily score defined on triangles inherits the homophily due to edges, whereas the 2-simplicial homophily is near 1. (Right) Varying p_2/q_2 with $p_1 = 4q_1$. The k-simplicial homophily score is larger than 1 if and only if $p_2/q_2 > 1$, and hence correctly captures homophilous group dynamics. In contrast, since $p_1 > q_1$, hypergraph homophily scores are consistently inflated

- $n_1, \ldots, n_m$, the number of nodes in each class for the model.
- p_1 and q_1, the probability of an edge (1-simplex) forming between nodes in the same or different communities, respectively.
- p_2, the probability that a closed triangle consisting of nodes in the same community becomes filled as a 2-simplex.
- q_2, the probability that a closed triangle consisting of nodes in different communities becomes filled.

Each random simplicial complex is then built using a generative process. First, edges form between communities with probabilities p_1 and q_1 as noted above, creating a graph G. Then, for each closed triangle in G, the triangle becomes filled with probability p_2 if all nodes in the closed triangle are of the same community, or with probability q_2 otherwise.

This model can control the presence of homogeneous edges and homogeneous triangles in the network while maintaining the structural requirement of a simplicial complex. p_1 and q_1 determine whether there is homophily in pairwise interactions, and p_2 and q_2 dictate how much homophily occurs in the filled triangles beyond that of the underlying pairwise interactions.

We provide two sets of experimental results on the simplicial stochastic block model, each using two classes of nodes and community sizes of 1000 for each class. In the left of Fig. 2, we set $p_2 = q_2$ which indicates that by construction group formation is not influenced by class labels. k-simplicial homophily detects that $p_2 = q_2$ and reports a value close to 1, whereas the value reported by hypergraph homophily depends on the parameters p_1 and q_1. The hypergraph homophily score in this case is above 1 if and only if $p_1/q_1 > 1$, indicating that hypergraph homophily defined on *triangles* inherits the properties of *edge* homophily prescribed in the model. In contrast, the right figure illustrates that k-simplicial homophily can effectively identify whether group dynamics are homophilous. We let $p_1 > q_1$ and vary the ratio p_2/q_2. The k-simplicial homophily score on triangles is above 1 if and only if p_2/q_2 is above

Table 1 Summary statistics for the 16 datasets used for homophily comparisons, 9 of which are also used for link prediction experiments. Additional details can be found at https://github.com/arnabsarker/SimplicialHomophily

Dataset	Nodes	Classes	Edges	Triangles	Time steps
`cont-village` [21]	46	5	329	610	
`cont-hospital` [10]	81	5	1381	6268	12,605
`cont-workplace-13` [10]	100	5	3915	80,173	20,129
`email-Enron` [4]	148	2	1344	1159	
`cont-workplace-15` [10]	232	12	16,725	329,056	21,536
`cont-primary-school` [10]	241	11	8317	5139	3124
`bills-senate` [8, 9]	297	4	10,555	11,460	4975
`cont-high-school` [10]	326	9	5,818	2,370	8,938
`bills-house` [8, 9]	1495	3	29,959	16,884	4871
`hosp-DAWN` [4]	2558	364	124,155	1,081,440	8
`soc-youtube` [19]	10,513	10	85,134	24,903	
`soc-flickr` [19]	54,104	10	1,231,068	2,692,349	
`coauth-dblp` [1]	105,256	2	316,631	384,549	55
`clicks-trivago` [4]	172,737	160	176,194	116,264	
`soc-livejournal` [19]	259,865	10	329,954	176,547	
`soc-orkut` [19]	399,314	10	1,120,880	17,339	

1, whereas the hypergraph score is consistently above 1 because $p_1 > q_1$. Because the hypergraph score is influenced by both p_1/q_1 and p_2/q_2, it can not decouple their effects.

Empirical Networks To understand the effect of different homophily definitions in empirical networks, we apply the definitions to the 16 publicly available datasets described in Table 1. With the empirical networks, we are able to quantify the difference between the k-simplicial homophily score and the hypergraph homophily score for triadic interactions. Using the definitions from Sect. 3.1, we compute the homogeneous homophily scores for all 16 datasets and display them in Fig. 3. For all but one dataset (`contact-hospital`), the hypergraph homophily score is higher than the k-simplicial homophily score, consistent with synthetic experiments where $p_1 > q_1$. The result is particularly strong for `retail-trivago`, `cont-high-school`, `bills-house`, and `coauth-dblp`, for which k-simplicial homophily suggests anti-homophily in group formation. In `retail-trivago`, which has the strongest tendency for anti-homophily, we posit that travelers headed to a specific destination might look at two hotels to compare cost and amenities, but if a traveler is browsing more than three hotels, they are likely taking a longer trip or have more flexibility for their search. For the remaining three datasets, the tendency for anti-homophily is much smaller, but can still be explained by a desire for diversity in larger group sizes.

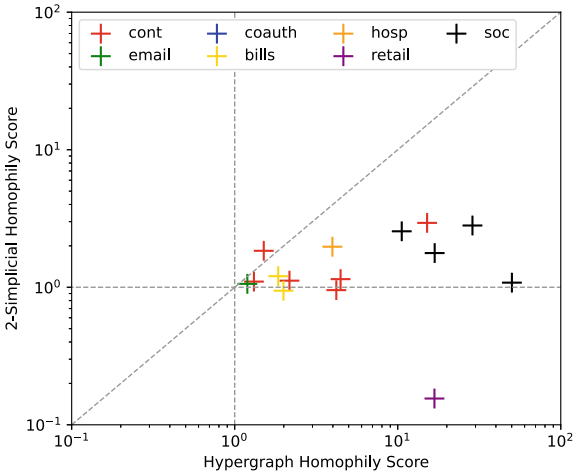


Fig. 3 Scatterplot of global homophily scores with a hypergraph baseline compared to a simplicial complex baseline. In 15 out of 16 datasets, the 2-simplicial homophily score is lower than the hypergraph homophily score, as the hypergraph homophily score inherits properties from edges

In the context of heterogeneous homophily definitions, it appears that pairwise interactions explain much of hypergraph homophily, as suggested in Fig. 4. For nearly all classes and each value of t, the observed metric is closer to the random baseline of k-simplicial homophily than that of hypergraph homophily. That is, the baseline in k-simplicial homophily tends to "flatten" the homophily scores as a function of type t. This intuition is confirmed with homogeneous homophily scores. When using the graph homophily score to predict hypergraph homophily, we find that a simple linear model results in an R^2 value of 0.698 ($p < 0.001$), with a positive coefficient that further indicates that edge homophily positively influences hypergraph homophily. In contrast, the same analysis using graph homophily to explain k-simplicial homophily on triads results in an R^2 value of 0.167 ($p = 0.117$), suggesting that k-simplicial homophily offers distinct insights on group dynamics. In particular, we show that this distinct information is particularly useful in the task of higher order link prediction.

5 Homophily and Higher Order Link Prediction

Higher order link prediction has been introduced as a "benchmark problem to assess models and algorithms that predict higher-order structure" [4]. One is given a partial time series of network data up to a time t, and then is asked to predict if a closed but not filled triangle will become filled the after time t. In the prediction task, we learn two separate logistic regression models on the first 50% of simplices observed in the data, and test the logistic regression model on the remaining 50% of data. The first model ("Without Labels") serves as a baseline and uses the local features

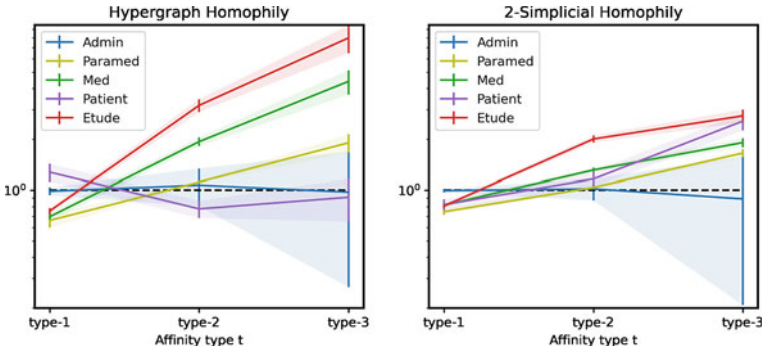


Fig. 4 Homophily scores with heterogenous group compostion for `cont-hospital`. Error bars represent 95% confidence intervals. We find the simplicial complex baseline often results in less extreme values of homophily for the majority of classes in the data and all values of t, suggesting that pairwise interactions account for much variation in hypergraph homophily

described in Benson et al. [4] to predict the binary outcome of whether a particular closed but not filled triangle will become filled. The features of this regression include the frequency with which each tie occurs between each pair of nodes in the closed triangle, the degree of each node (in the traditional graph sense and weighted by the number of simplices each node is in), the number of common neighbors between the nodes, and logarithmic rescalings of all of these factors. The second model ("With Labels") uses the features of the first model and an additional binary indicator feature which is 1 if and only if all nodes in the closed triangle are the same type.

Table 2 Group Formation Prediction Performance. Prediction performance is measured using the AUC-PR and presented relative to a random baseline. Bolded entries indicate a statistically significant larger performance metric via a bootstrapping procedure which also produces confidence intervals. Table is sorted by 2-simplicial homophily score of the training set, and shows that extreme 2-simplicial homophily scores indicate when node labels are useful

Dataset	Prediction Performance		Homophily Score	
	Without Labels	With Labels	2-Simplicial ↓	Hypergraph
`bills-house`	1.12 (±0.016)	**1.18** (±0.031)	0.92	2.01
`coauth-dblp`	1.25 (±0.029)	**1.42** (±0.037)	0.99	1.12
`cont-workplace-13`	**2.36** (±0.019)	2.22 (±0.016)	1.05	1.30
`bills-senate`	**4.74** (±0.257)	3.38 (±0.161)	1.16	1.76
`cont-workplace-15`	1.16 (±0.001)	1.16 (±0.001)	1.17	3.87
`cont-primary-school`	1.08 (±0.000)	1.08 (±0.000)	1.34	2.03
`cont-hospital`	3.38 (±0.028)	**4.46** (±0.038)	1.79	1.56
`hosp-DAWN`	4.48 (±0.001)	**4.50** (±0.001)	2.36	6.82
`cont-high-school`	1.48 (±0.001)	**1.55** (±0.001)	2.84	8.05

The results of the logistic regression are presented in Table 2. We evaluate performance of different features using the area under the precision-recall curve (AUC-PR) and report the score relative to a random baseline, as has been done in the literature [4]. The table is sorted by the 2-simplicial homophily score computed on the training set of data, i.e. the first 50% of simplices which are used to train the logistic regression model. We find that for extreme values of the 2-simplicial homophily score, indicating either homophily or anti-homophily, that the prediction performance increases when homogeneous node labels are used as a regressor. Specifically, the two lowest 2-simplicial homophily scores and the three highest 2-simplicial homophily scores are for datasets where node labels increase predictive performance, whereas the four datasets with moderate scores see no change or decreases in performance. In contrast, when hypergraph homophily scores are sorted, no clear patterns emerge.

6 Conclusions

We proposed a measure for homophily in simplicial complexes, k-simplicial homophily, which isolates the homophily present in group dynamics. The necessity of such a definition was established on synthetic and empirical data, which indicated that prior notions of homophily for arbitrary hypergraphs can inherit homophilous structure from underlying pairwise interactions and miss the effect of group dynamics. k-simplicial homophily applies to groups of arbitrary size, and we provided experimental and theoretical evidence on triadic interactions that k-simplicial homophily provides distinct information from homophily scores on edges. Moreover, we showed the empirical value of k-simplicial homophily, as extreme scores indicate the value of node labels for predicting if group interactions will occur. These techniques ultimately provide a general approach to isolate group dynamics in simplicial complexes, which we believe will be useful in analyzing group interactions in complex networks more broadly.

References

1. Agarwal, S., Mittal, N., Katyal, R., Sureka, A., Correa, D.: Women in computer science research: what is the bibliography data telling us? Acm Sigcas Comput. Soc. **46**(1), 7–19 (2016)
2. Asch, S.E.: Opinions and social pressure. Sci. Am. **193**(5), 31–35 (1955)
3. Battiston, F., Cencetti, G., Iacopini, I., Latora, V., Lucas, M., Patania, A., Young, J.-G., Petri, G.: Networks beyond pairwise interactions: structure and dynamics. Phys. Rep. (2020)
4. Benson, A.R., Abebe, R., Schaub, M.T., Jadbabaie, A., Kleinberg, J.: Simplicial closure and higher-order link prediction. Proc. Nat. Acad. Sci. **115**(48), E11221–E11230 (2018)
5. Brown, R.: Social Psychology. Simon and Schuster (1986)
6. Dong, Y., Johnson, R.A., Xu, J., Chawla, N.V.: Structural diversity and homophily: a study across more than one hundred big networks. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 807–816 (2017)

7. Easley, D., Kleinberg, J., et al.: Networks, Crowds, and Markets, vol. 8. Cambridge University Press Cambridge (2010)
8. Fowler, J.H.: Connecting the congress: a study of cosponsorship networks. Polit. Anal. **14**(4), 456–487 (2006)
9. Fowler, J.H.: Legislative cosponsorship networks in the us house and senate. Soc. Netw. **28**(4), 454–465 (2006)
10. Génois, M., Barrat, A.: Can co-location be used as a proxy for face-to-face contacts? EPJ Data Sci. **7**(1), 11 (2018)
11. Ghrist, R.W.: Elementary Applied Topology, vol. 1. Createspace Seattle (2014)
12. Hatcher, A.: Algebraic Topology (2002)
13. Kahle, Matthew, et al.: Topology of random simplicial complexes: a survey. AMS Contemp. Math **620**, 201–222 (2014)
14. Kossinets, G., Watts, D.J.: Empirical analysis of an evolving social network. Science **311**(5757), 88–90 (2006)
15. Kumar, T., Vaidyanathan, S., Ananthapadmanabhan, H., Parthasarathy, S., Ravindran, B.: Hypergraph clustering by iteratively reweighted modularity maximization. Appl. Netw. Sci. **5**(1), 1–22 (2020)
16. Latané, B., Williams, K., Harkins, S.: Many hands make light the work: the causes and consequences of social loafing. J. Personal. Soc. Psychol. **37**(6), 822 (1979)
17. Lazarsfeld, P.F., Merton, R.K., et al.: Friendship as a social process: a substantive and methodological analysis. Freedom Control Modern Soc. **18**(1), 18–66 (1954)
18. McPherson, M., Smith-Lovin, L., Cook, J.M.: Birds of a feather: homophily in social networks. Ann. Rev. Sociol. **27**(1), 415–444 (2001)
19. Mislove, A., Marcon, M., Gummadi, K.P., Druschel, P., Bhattacharjee, B.: Measurement and analysis of online social networks. In: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, pp. 29–42 (2007)
20. Newman, M.E.J.: Assortative mixing in networks. Phys. Rev. Lett. **89**(20), 208701 (2002)
21. Ozella, L., Paolotti, D., Lichand, G., Rodríguez, J.P., Haenni, S., Phuka, J., Leal-Neto, O.B., Cattuto, C.: Using wearable proximity sensors to characterize social contact patterns in a village of rural Malawi. EPJ Data Sci. **10**(1), 46 (2021)
22. Satchidanand, S.N., Ananthapadmanaban, H., Ravindran, B.: Extended discriminative random walk: a hypergraph approach to multi-view multi-relational transductive learning. In: Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)
23. Schaub, M.T., Benson, A.R., Horn, P., Lippner, G., Jadbabaie, A.: Random walks on simplicial complexes and the normalized hodge 1-laplacian. SIAM Rev. **62**(2), 353–391 (2020)
24. Stehlé, J., Voirin, N., Barrat, A., Cattuto, C., Isella, L., Pinton, J.-F., Quaggiotto, M., Van den Broeck, W., Régis, C., Lina, B., et al.: High-resolution measurements of face-to-face contact patterns in a primary school. PloS One **6**(8), e23176 (2011)
25. Veldt, N., Benson, A.R., Kleinberg, J.: Higher-order homophily is combinatorially impossible (2021). arXiv:2103.11818

Statistical Network Similarity

Pierre Miasnikof, Alexander Y. Shestopaloff, Cristián Bravo, and Yuri Lawryshyn

Abstract Graph isomorphism is a problem for which there is no known polynomial-time solution. The more general problem of computing graph similarity metrics, graph edit distance or maximum common subgraph, is NP-hard. Nevertheless, assessing (dis)similarity between two or more networks is a key task in many areas, such as image recognition, biology, chemistry, computer and social networks. In this article, we offer a statistical answer to the following questions: (a) "*Are networks* G_1 and G_2 *similar?*", (b) *"How different are the networks* G_1 and G_2?" and (c) "*Is* G_3 *more similar to* G_1 *or* G_2?". Our comparisons begin with the transformation of each graph into an all-pairs distance matrix. Our node-node distance, Jaccard distance, has been shown to offer an accurate reflection of the graph's connectivity structure. We then model these distances as probability distributions. Finally, we use well-established statistical tools to gauge the (dis)similarities in terms of probability distribution (dis)similarity. This comparison procedure aims to detect (dis)similarities in connectivity structure and community structure in particular, not in easily observable graph characteristics, such as degrees, edge counts or density. We validate our hypothesis that graphs can be meaningfully summarized and compared via their node-node distance distributions, using several synthetic and real-world graphs. Empirical results demonstrate its validity and the accuracy of our comparison technique.

Note on terminology: *For the sake of compactness, the work in this article focuses exclusively on simple graphs. We only consider unweighted, undirected graphs with no self-loops or multiple edges. Throughout this article, the terms graph and network*

P. Miasnikof (✉) · Y. Lawryshyn
University of Toronto, Toronto, ON, Canada
e-mail: p.miasnikof@mail.utoronto.ca

A. Y. Shestopaloff
Queen Mary University of London, London, UK

Memorial University of Newfoundland, St. John's, NL, Canada

C. Bravo
University of Western Ontario, London, ON, Canada

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_25

are used interchangeably. Similarly, the terms vertex and node and the terms edge, arc, link and connection are used as synonyms.

1 Introduction

Graph isomorphism is a problem for which there is no known polynomial-time solution. The more general problem of computing graph similarity metrics, graph edit distance or maximum common subgraph, is NP-hard. Nevertheless, assessing network (dis)similarity is a key task in many areas, such as image recognition, biology, chemistry, computer and social networks. In this article, we offer a statistical answer to the following questions: (a) "*Are networks G_1 and G_2 similar?*", (b) "*How different are the networks* G_1 and G_2?" and (c) "*Is G_3 more similar to* G_1 or G_2?".

We obtain these answers by first converting networks (graphs) into an all pairs distances matrix. To achieve this transformation, we use Jaccard distance instead of the typically used shortest-path or the also common random walk-based distances (e.g., commute, resistance, ...). Previous work has highlighted the shortcomings of shortest-path [1] and random walk-based distances [16, 17, 22]. The advantages of Jaccard distance, especially its relation to connectivity structure, have also been demonstrated [4, 19, 20].

Our comparison technique is focused on comparing each network's connectivity structure and community structure in particular, not on easily observable graph characteristics. We argue that changes in connectivity may be indicative of critical network event occurrences, which makes structural conectivity-based (dis)similarity worthy of investigation. For example, the presence of denser subgraphs may indicate a loss of connection to the broader network and the appearance of bottlenecks, in a physical or computer network. They can also be an indicator of malicious activity, especially of the multi-party coordinated variety [24, 27, 28].

As described later in this article, Jaccard distance also has a probabilistic interpretation. On the basis of this interpretation, we then compare networks as probability distributions of distances, using well-established statistical techniques. As stated earlier, our comparisons are not restricted to a few key statistical or graph characteristics, such as mean degree or density. Instead, our conversion to a distance matrix and interpretation of these distances as a probability distribution captures each graph's entire structure.

This probabilistic approach and associated statistical tests are the major contribution of this work. Converting graphs to probability distributions not only allows the use of well-established statistical tools, it offers objective significance metrics. It also opens the door to similarity comparisons based on subsampling. While space restrictions do not allow us to explore this avenue here, our initial investigations in this area show promise. For now, this avenue is left for future work. Our future work will explore similarity comparisons through statistical subsampling in great detail.

2 Previous Work

The comparison of static graphs and the study of temporal graphs are overlapping topics. Indeed, the study of temporal graphs naturally includes comparisons of snapshots of time-evolving graphs. In the past, several authors have highlighted the need to study graph similarity and their evolution over time. These authors have illustrated their claims using various areas of application, areas as varied as image recognition [3], network robustness and resilience [14], mobile telephony [7, 14, 25] and public transportation [18]. Notably, graph comparisons and temporal graphs remain current topics of inquiry [6, 12, 26].

A full review of the graph similarity and temporal graphs literature is beyond the scope of this short article. However, we wish to highlight the fact that this article is built upon the foundations of Schieber et al. [23] and the very recent work of Wang et al. [26]. These authors have modeled graphs as probability distributions. We also wish to highlight that our work does not rely on costly embedding computations which have been presented in the recent literature (e.g., [2, 26]).

3 Methods

We model graphs as probability distributions of vertex-vertex distances. We too posit that graphs can be meaningfully summarized and compared on the basis of their node-node distances. Just as others before us, we begin by obtaining the distances between all vertex pairs. However, unlike in previous work, we use Jaccard distances [4, 13, 19, 20]. The main difference between earlier work and ours lies in the choice of node-node distance.

Schieber et al. [23] use shortest path distance. However, previous work has highlighted its shortcomings. For example, Akara-pipattana et al. [1] stated the following: "*While intuitive and visual, this notion of distance is limited in that it does not fully capture the ease or difficulty of reaching point j from point i by navigating the graph edges. It does not say whether there is only one path of minimal length or many such paths, whether these paths can be straightforwardly located, or whether alternative paths are considerably or only slightly longer*". In the past, Chebotarev and Shamis [5] as well as Fouss et al. [8] have also highlighted the unsuitability of shortest-path distance as a similarity measure between vertices. We have also echoed these assertions in recent publications and have demonstrated the superiority of the Jaccard distance as a reflection of graph structure [19, 20].

In their very recent work, Wang et al. [26] use a combination of embedding and Euclidean distance. While they report interesting results, this two-step process appears cumbersome and ill-suited to larger graphs, at first glance. Arguably, embedding graphs into vector space carries a non-trivial computational cost. In this specific case, the authors use the DeepWalk algorithm [21] to obtain their embedding. While the creators of DeepWalk claim their technique is scalable and parallelizable, it simu-

lates incomplete random walks across the network, relies on simplifying assumptions, requires several input parameters and also performs gradient descent optimization. In contrast, Jaccard distance only relies on simple vertex-pair level arithmetic computations, instead of multiple layers of neighborhoods (multiple layers of neighbors' neighbors), and has been shown to offer an accurate reflection of graph structure [19, 20]. Its computation can also be easily performed incrementally or in parallel (not possible with random walk simulations). In addition, several authors have highlighted the breakdown of the random-walk based commute (resistance) distance in the case of larger graphs [16, 17, 22].

We would also like to draw attention to the fact some authors restrict their comparisons to graphs with equal numbers of nodes [9]. Yet, others are interested in the more general case of comparisons between graph with unequal numbers of nodes [15]. Because we compare connectivity through cumulative distributions, the number of nodes in each graph is not relevant. Our technique applies equally to either case.

3.1 Vertex-vertex Jaccard Distance

The Jaccard distance separating two vertices i and j is defined as

$$\zeta_{ij} = 1 - \underbrace{\frac{|a_i \cap a_j|}{|a_i \cup a_j|}}_{s_{ij}} \in [0, 1]\,.$$

Here, a_i (a_j) represents the set of all vertices with which vertex i (j) shares an edge. The ratio s_{ij} is the well known Jaccard similarity. The Jaccard distance (ζ_{ij}) is its complement.

3.1.1 Probabilistic Interpretation of the Jaccard Distance

The Jaccard similarity (s_{ij}) between two nodes i and j can be interpreted probabilistically. Consider all nodes of a network excluding i and j and select at random a node k. The Jaccard similarity is then an estimate of the (conditional) probability that both i and j are connected to k, given that at least one of i and j is connected to k. Mathematically, we express s_{ij} as,

$$s_{ij} = P\left(\left(e_{ik} \wedge e_{jk}\right) \mid \left(e_{ik} \vee e_{jk)}\right)\right)\,,$$

where e_{ij} indicates the existence of an edge between nodes i and j.

The Jaccard distance (ζ_{ij}) is its complement. It can be interpreted as one of these two cases:

(a) the (conditional) probability that i is connected to k, but j is not,

(exclusive) or

(b) the (conditional) probability that j is connected to k, but i is not.

Mathematically, we express it as

$$\zeta_{ij} = 1 - P\left(\left(e_{ik} \wedge e_{jk}\right) \mid \left(e_{ik} \vee e_{jk)}\right)\right) = P\left(\left(e_{ik} \veebar e_{jk}\right) \mid \left(e_{ik} \vee e_{jk)}\right)\right) .$$

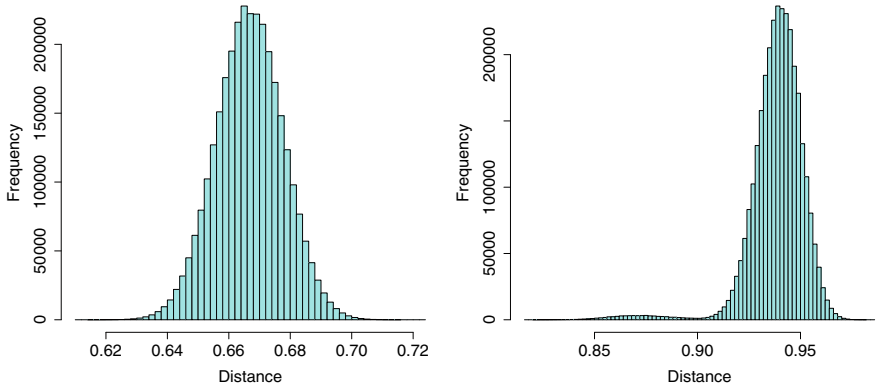
3.1.2 From Graph to Empirical Probability Distribution

Once all distances ζ_{ij} have been obtained, we examine their statistical distribution. On the basis of the probabilistic interpretation of the ζ_{ij} just described, we treat these quantities as random variables. This model allows us to study and compare graphs as empirical probability distributions of node-node distances.

This transformation from graph to probability distribution also opens the door to similarity comparisons based on subsampling. While space restrictions do not allow us to explore this avenue here, our initial investigations in this area show promise. Our future work will explore similarity comparisons through statistical subsampling.

Figure 1 illustrates the interpretation of a graph as a probability distribution. The image on the left shows the distribution of node-node distances for an Erdős-Rényi (ER) graph with an edge probability of $p = 0.5$. The image on the right is of the distribution of distances between nodes of a stochastic block model graph (SBM) of varying cluster sizes and inout edge probabilities of $0.9/0.1$.

The structural differences between these two graphs is immediately obvious. The ER graph's distances are symmetrically distributed about their mean, in a Gaussian-like pattern. In stark contrast, the SBM graph's distances are left-skewed and



(a) ER (b) SBM

Fig. 1 Distances as distributions

bi-modal. The left mode reflects distances between nodes in the same blocks, whereas the right mode reflects distances between nodes not in the same blocks. Naturally, this pattern does not occur under the ER model.

3.2 *Dissimilarity of Probability Distributions*

We compare the networks of interest via the empirical probability distributions of the Jaccard distances between their nodes. To perform these comparisons, we use the Kolmogorov-Smirnov (K-S) distance and the Wasserstein distance of order p. These distances are defined as follows: In a comparison between two networks, let $F_1(x)$ be the empirical cumulative distribution function (CDF) of Jaccard distances for the first network, and $F_2(x)$ the empirical CDF of Jaccard distances for the second network (F^{-1} denotes the inverse CDF). The Kolmogorov-Smirnov (K-S) distance to compare F_1 and F_2 is defined as

$$D = \sup_x |F_1(x) - F_2(x)| \quad (\in [0, 1]) .$$

Meanwhile, the Wasserstein distance of order p between F_1 and F_2 is defined as

$$W_p(F_1, F_2) = \left(\int_0^1 |F_1^{-1}(u) - F_2^{-1}(u)|^p du \right)^{1/p} .$$

(In our experiments, we set the parameter $p = 2$.)

The K-S distance metric D is also a test statistic. In this specific case, it is a test statistic for the two-sample K-S test. The hypotheses of this test are listed below.

- Null hypothesis (H_o): the two samples are drawn from the same distribution
- Alternative hypothesis (H_a): the two samples are drawn from different distributions

The p-values of the K-S test provide an interpretation and validation of the test statistic (distance D). Concretely, they are the measure of the area under the Kolmogorov distribution's probability density curve beyond the point D. This area represents the probability of obtaining a distance of the same or greater magnitude, under the (null) hypothesis that both samples were drawn from the same distribution. Small p-values provide evidence that the maximum vertical distance between the empirical CDFs of two compared graphs is statistically significantly different from zero. These p-values are obtained directly from the Kolmogorov distribution. We compare the D metric against the critical values (cv) of a statistical table or, more conveniently, by using statistical software (e.g., SciPy https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.stats.ks_2samp.html). For example, in a two-sample two-sided test with sample sizes n_1 and n_2, the $\alpha = 0.01$ cv for D is $1.63 \times \sqrt{(n_1 + n_2)/(n_1 n_2)}$. In all our tests, these cv are at most $\sim 10^{-3}$. Hence, the p-value of any D greater than 10^{-3} is less than 0.01.

While the K-S distance is always contained in the interval [0, 1], the Wasserstein distance is not. To make comparisons more meaningful and easier to interpret, we transform the latter, so that it also lies on the same interval. After obtaining the quantity W_p, we perform the following transformation,

$$\tilde{\mathcal{W}}_p = 1 - \exp(-W_p) \quad (\in [0, 1]) \,.$$

In our comparisons, we use the quantity $\tilde{\mathcal{W}}_p$.

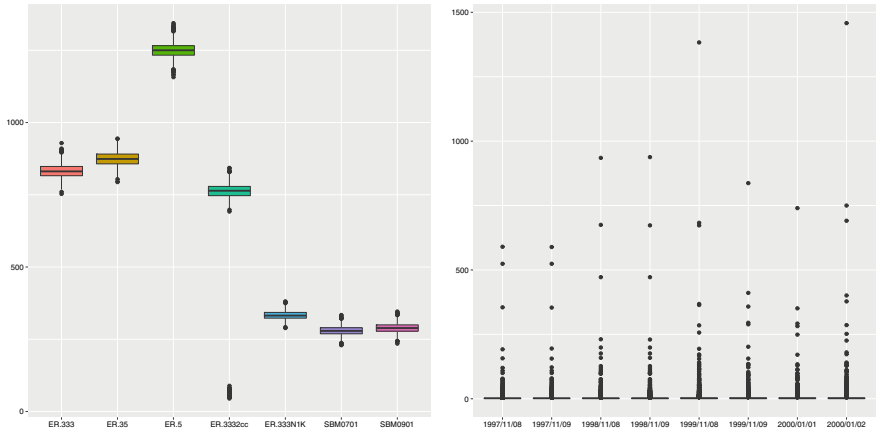
4 Numerical Results

Due to space limitations, we restrict our attention to comparisons using our own technique, only. We reserve comparisons to other similarity techniques for future work. Here, we validate our hypothesis that graphs can be meaningfully summarized and compared via their node-node distance distributions. We begin with validations using several synthetic graphs with known (dis)similarity. To illustrate real-world relevance, we also compare several real-world networks. Key characteristics of our test graphs are reported in Table 1. The columns correspond to

- $|V|$: number of vertices,
- $|E|$: number of edges,
- $\mathcal{K}$: density,

Table 1 Graph characteristics

		$\|V\|$	$\|E\|$	$\mathcal{K}$	$\min(D)$	$\bar{D}$	$\max(D)$	$\|CC\|$
Synthetic	ER.333	2500	1,039,694	0.33	753	831.76	929	1
	ER.35	2500	1,092,408	0.35	794	873.93	944	1
	ER.5	2500	1,562,067	0.50	1,157	1249.65	1344	1
	ER.3332cc	2500	887,948	0.28	45	710.36	843	2
	ER.333N1K	1000	166,417	0.33	289	332.83	381	1
	SBM0701	2495	348,674	0.11	230	279.50	334	1
	SBM0901	2495	360,867	0.12	235	289.27	346	1
Real-world	1997/11/08	3015	5156	0.00	1	3.42	590	1
	1997/11/09	3,011	5150	0.00	1	3.42	589	1
	1998/11/08	4,296	7815	0.00	1	3.64	935	1
	1998/11/09	4,301	7838	0.00	1	3.64	938	1
	1999/11/08	6,127	12,046	0.00	1	3.93	1383	1
	1999/11/09	3,962	7931	0.00	1	4.00	837	1
	2000/01/01	3,570	7033	0.00	1	3.94	740	1
	2000/01/02	6,474	12,572	0.00	1	3.88	1458	1



(a) Synthetic (b) Real-world

Fig. 2 Degree distributions, box-plots

- $\min(D)$: minimum degree,
- $\bar{D}$: mean degree,
- $\max(D)$: maximum degree and
- $|CC|$: number of connected components (Fig. 2).

The generative models used to create the synthetic graphs are listed below. These graphs were generated using the NetworkX library [10].

- ER.333: ER with $n = 2500,\ p = 0.333$
- ER.35: ER with $n = 2500,\ p = 0.35$
- ER.5: ER with $n = 2500,\ p = 0.5$
- ER.3332cc: ER also with $p = 0.333$, but with two connected components ($n_1 = 2300,\ n_2 = 200$)
- ER.333N1K: ER also with $p = 0.333$, but with only $n = 1000$
- SBM0701: stochastic block model, with clusters in range of [37,62] and $p_{in} = 0.7,\ p_{out} = 0.1$
- SBM0901: stochastic block model, with clusters in range of [37,62] and $p_{in} = 0.9,\ p_{out} = 0.1$

Meanwhile, the real-world graphs were obtained from the Harvard Dataverse repository [11]. These data sets are from the University of Oregon's "Route Views Project". Each graph contains a daily snapshot of a set of internet "autonomous systems" and their connections (Tables 2, 3, 4 and 5).

Our results show that our technique based on a transformation from graph to probability distribution and distance measurements with either the Wasserstein and K-S distances between node-node distributions are valid measures of network (dis)similarity. Indeed, these metrics accurately identify network structure changes, even in arguably difficult cases. For example, both metrics accurately detect the disconnection into

Table 2 Wasserstein distances, synthetic graphs

	ER.333	ER.35	ER.5	ER.3332cc	ER.333N1K	SBM0701	SBM0901
ER.333	NA	0.01	0.13	0.07	0.01	0.13	0.13
ER.35	NA	NA	0.11	0.07	0.01	0.14	0.14
ER.5	NA	NA	NA	0.16	0.13	0.24	0.24
ER.3332cc	NA	NA	NA	NA	0.06	0.12	0.12
ER.333N1K	NA	NA	NA	NA	NA	0.13	0.13
SBM0701	NA	NA	NA	NA	NA	NA	0.00
SBM0901	NA	NA	NA	NA	NA	NA	NA

Table 3 Wasserstein distances, real-world graphs

	1997/11/08	1997/11/09	1998/11/08	1998/11/09	1999/11/08	1999/11/09	2000/01/01	2000/01/02
1997/11/08	NA	0.00	0.03	0.03	0.04	0.04	0.04	0.04
1997/11/09	NA	NA	0.03	0.03	0.04	0.04	0.04	0.04
1998/11/08	NA	NA	NA	0.00	0.02	0.03	0.03	0.02
1998/11/09	NA	NA	NA	NA	0.02	0.03	0.03	0.02
1999/11/08	NA	NA	NA	NA	NA	0.02	0.02	0.01
1999/11/09	NA	NA	NA	NA	NA	NA	0.01	0.02
2000/01/01	NA	NA	NA	NA	NA	NA	NA	0.02
2000/01/02	NA	NA	NA	NA	NA	NA	NA	NA

Table 4 K-S distances and (p-values), synthetic graphs (*Note* $\alpha = 0.01$ cv for D is at most $\sim 10^{-3}$)

	ER.333	ER.35	ER.5	ER.3332cc	ER.333N1K	SBM0701	SBM0901
ER.333	NA	0.43 (0.00)	1.00 (0.00)	0.15 (0.00)	0.11 (0.00)	1.00 (0.00)	1.00 (0.00)
ER.35	NA	NA	1.00 (0.00)	0.46 (0.00)	0.38 (0.00)	1.00 (0.00)	1.00 (0.00)
ER.5	NA	NA	NA	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)
ER.3332cc	NA	NA	NA	NA	0.15 (0.00)	0.85 (0.00)	0.85 (0.00)
ER.333N1K	NA	NA	NA	NA	NA	1.00 (0.00)	1.00 (0.00)
SBM0701	NA	NA	NA	NA	NA	NA	0.07 (0.00)
SBM0901	NA	NA	NA	NA	NA	NA	NA

two connected components of the ER graph with probability $p = 0.333$ (ER.333 vs. ER.3332cc).

Our results also confirm that our procedure correctly identifies the structural stability of the internet networks. In fact, our procedure is robust to degree outliers

Table 5 K-S distances and (p-values), real-world graphs (*Note* $\alpha = 0.01$ cv for D is at most $\sim 10^{-3}$)

	1997/11/08	1997/11/09	1998/11/08	1998/11/09	1999/11/08	1999/11/09	2000/01/01	2000/01/02
1997/11/08	NA	0.00 (0.98)	0.01 (0.00)	0.01 (0.00)	0.01 (0.00)	0.02 (0.00)	0.02 (0.00)	0.01 (0.00)
1997/11/09	NA	NA	0.01 (0.00)	0.01 (0.00)	0.01 (0.00)	0.02 (0.00)	0.02 (0.00)	0.01 (0.00)
1998/11/08	NA	NA	NA	0.00 (0.87)	0.01 (0.00)	0.01 (0.00)	0.02 (0.00)	0.01 (0.00)
1998/11/09	NA	NA	NA	NA	0.01 (0.00)	0.01 (0.00)	0.02 (0.00)	0.01 (0.00)
1999/11/08	NA	NA	NA	NA	NA	0.01 (0.00)	0.02 (0.00)	0.00 (0.00)
1999/11/09	NA	NA	NA	NA	NA	NA	0.00 (0.00)	0.01 (0.00)
2000/01/01	NA	NA	NA	NA	NA	NA	NA	0.02 (0.00)
2000/01/02	NA	NA	NA	NA	NA	NA	NA	NA

that are very common in real-world networks. Arguably, while the number of nodes and edges of internet networks do vary, the graph's connectivity structure remains constant. This robustness is reflected in the small distance between the distributions.

Even so, here, we must also acknowledge the limitations of our method. While our network comparison technique is indeed robust to degree outliers, it does correctly detect changes in the number of edges and vertices and classify these networks as significantly different. However, the magnitude of their difference is very small, which is why we highlight robustness. For example, in the comparison between internet networks, our technique correctly identifies the difference between graphs 2000/01/01 and 2000/01/02 as statistically significant ($p = 0.00$). As mentioned earlier, the magnitude of the difference is very low ($D, \tilde{\mathcal{W}} = 0.02$), in spite of a very significant difference in the number of edges and vertices. This low distance variation in response to large node and edge count variations in the distances between CDFs may be considered a limitation. For this reason, we caution against interpreting absolute magnitudes of distances without testing for significance.

Nevertheless, these graphs appear to be rather similar, from a structural point of view. Indeed, both networks, have equal density and a very similar degree distribution. We posit that the magnitude of the difference remains small, although statistically significant, due to the structural similarity of these networks. Meanwhile, in the comparison between the ER graphs with 1000 and 2500 nodes (ER.333 vs. ER.333N1K), our technique did correctly identify a variation in network structure and a greater dissimilarity (distance) between these graphs. While these two graphs share the same edge probability parameter, their degree distributions differ significantly.

Finally, we must also offer a comparison of our technique to arguably simpler to obtain network characteristics, namely density and degree distribution. While density

does indeed offer valuable information about a graph's structure, a comparison of densities is not sufficient to detect a change in structure. For example, the graphs ER.333 and ER.333N1K have identical densities, yet have significantly different degree distributions. Also, a graph's density does not offer any information regarding local connection patterns, such as community structure for example.

Degree distribution also offers very valuable information about a network. Again, a comparison of degree distributions only offers a partial assessment of (dis)similarity. For example, the ER.333 and ER.333N1K graphs have significantly different degree distributions, yet have very similar connectivity patterns. Our two stochastic block model graphs (SBM0701 and SBM0901) have degree distributions that are very similar to the ER.333N1K, yet their connectivity (community) structure is totally different.

5 Conclusion

In this article, we present a statistical graph comparison technique which is based on node-node distances. Our results show that our technique accurately detects differences in graph structure. Future work will focus on statistical comparisons via subsampling, which should offer greater scalability for our comparison technique. Naturally, we will also conduct further tests, using different scenarios and compare our results to those obtained with other similarity techniques.

References

1. Akara-pipattana, P., Chotibut, T., Evnin, O.: Resistance distance distribution in large sparse random graphs (2021). arXiv:2107.12561
2. Bai, Y., Dingand S. Bian, H., Chen, T., Sun, Y., Wang, W.: SimGNN: A Neural Network Approach to Fast Graph Similarity Computation (2018). arXiv:1808.05689
3. Bunke, H.: Graph matching: Theoretical foundations, algorithms, and applications. Proc. Vision Interf. **21** (2000)
4. Camby, E., Caporossi, G.: The extended Jaccard distance in complex networks. Les Cahiers du GERAD G-2017-77 (2017)
5. Chebotarev, P., Shamis, E.: The Matrix-Forest Theorem and Measuring Relations in Small Social Groups. arXiv Mathematics e-prints math/0602070 (2006)
6. Coupette, C., Vreeken, J.: Graph similarity description: how are these graphs similar? In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 185–195. KDD '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3447548.3467257
7. Du, Z., Yang, Y., Gao, C., Huang, L., Huang, Q., Bai, Y.: The temporal network of mobile phone users in Changchun Municipality, Northeast China. Sci. Data **5**, 180228 (2018)
8. Fouss, F., Francoisse, K., Yen, L., Pirotte, A., Saerens, M.: An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification. Neural Netw. **31**, 53–72 (2012). https://www.sciencedirect.com/science/article/pii/S0893608012000822

9. Grohe, M., Rattan, G., Woeginger, G.: Graph Similarity and Approximate Isomorphism (2018). arXiv:1802.08509
10. Hagberg, A., Schult, D., Swart, P.: Exploring network structure, dynamics, and function using network X. In: Varoquaux, G., Vaught, T., Millman, J. (eds.), Proceedings of the 7th Python in Science Conference, pp. 11–15. Pasadena, CA USA (2008)
11. Han, J.: Autonomous systems graphs (2016). https://doi.org/10.7910/DVN/XLGMJR
12. Huang, S., Hitti, Y., Rabusseau, G., Rabbany, R.: Laplacian Change Point Detection for Dynamic Graphs (2020). arXiv:2007.01229
13. Jaccard, P.: Étude de la distribution florale dans une portion des Alpes et du Jura. Bulletin de la Société Vaudoise des Sciences Naturelles **37**, 547–579 (1901)
14. Tang, J., Leontiadis, I., Scellato, S., Nicosia, V., Mascolo, C., M. Musolesi, M., Latora, V.: Applications of temporal graph metrics to real-world networks. In: Temporal Networks, p. 135 (2013)
15. Koutra, D., Parikh, A., Ramdas, A., Xiang, J.: Algorithms for graph similarity and subgraph matching (2011). http://www.cs.cmu.edu/jingx/docs/DBreport.pdf. Accessed on 01 Dec 2015
16. von Luxburg, U., Radl, A., Hein, M.: Getting lost in space: large sample analysis of the resistance distance. In: Lafferty, J.D., Williams, C.K.I., Shawe-Taylor, J., Zemel, R.S., Culotta, A. (eds.), Advances in Neural Information Processing Systems 23, pp. 2622–2630. Curran Associates, Inc. (2010). http://papers.nips.cc/paper/3891-getting-lost-in-space-large-sample-analysis-of-the-resistance-distance.pdf
17. von Luxburg, U., Radl, A., Hein, M.: Hitting and commute times in large random neighborhood graphs. J. Mach. Learn. Res. **15**(52), 1751–1798 (2014). http://jmlr.org/papers/v15/vonluxburg14a.html
18. Maduako, I., Wachowicz, M., Hanson, T.: STVG: an evolutionary graph framework for analyzing fast-evolving networks. J. Big Data **6** (2019)
19. Miasnikof, P., Shestopaloff, A.Y., Pitsoulis, L., Ponomarenko, A.: An empirical comparison of connectivity-based distances on a graph and their computational scalability. J. Complex Netw. **10**(1) (2022). https://doi.org/10.1093/comnet/cnac003
20. Miasnikof, P., Shestopaloff, A.Y., Pitsoulis, L., Ponomarenko, A., Lawryshyn, Y.: Distances on a graph. In: Benito, R.M., Cherifi, C., Cherifi, H., Moro, E., Rocha, L.M., Sales-Pardo, M. (eds.) Complex Networks & Their Applications IX, pp. 189–199. Springer International Publishing, Cham (2021)
21. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: Online Learning of Social Representations (2014). arXiv:1403.6652
22. Ponomarenko, A., Pitsoulis, L., Shamshetdinov, M.: Overlapping community detection in networks based on link partitioning and partitioning around medoids. PLOS One **16**(8), 1–43 (2021). https://doi.org/10.1371/journal.pone.0255717
23. Schieber, T., Carpi, L., Diaz-Guilera, A., Pardalos, P., Masoller, C., Ravetti, M.: Quantification of network structural dissimilarities. Nat. Commun. **8**, 13928 (2017)
24. Shrivastava, N., Majumder, A., Rastogi, R.: In: 2008 IEEE 24th International Conference on Data Engineering, pp. 486–495 (2008)
25. Tang, J., Mascolo, C., Musolesi, M., Latora, V.: Exploiting temporal complex network metrics in mobile malware containment. In: 2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, pp. 1–9 (2011)
26. Wang, Z., Zhan, X.X., Liu, C., Zhang, Z.K.: Quantification of network structural dissimilarities based on network embedding. iScience 104446 (2022). https://www.sciencedirect.com/science/article/pii/S2589004222007179
27. Yan, H., Zhang, Q., Mao, D., Lu, Z., Guo, D., Chen, S.: Anomaly detection of network streams via dense subgraph discovery. In: 2021 International Conference on Computer Communications and Networks (ICCCN), pp. 1–9 (2021)
28. Ying, X., Wu, X., Barbará, D.: Spectrum based fraud detection in social networks. In: 2011 IEEE 27th International Conference on Data Engineering, pp. 912–923 (2011)

Intersection of Random Spanning Trees in Small-World Networks

András London and András Pluhár

Abstract Alon et al. [1] investigated the following 2-player zero-sum game on a connected graph G: *tree player* chooses a spanning tree T of G, while *edge player* chooses an edge e of G. The payoff to the edge player is defined by a function $\text{cost}(T, e)$. It is a natural continuation of their work is to consider the case when both players are a tree player. This also leads us to the problem of intersection of random spanning trees, that is the number of common edges of two spanning trees of G chosen uniformly at random. In this paper we derive a lower bound for the minimum expected intersection and using bootstrap simulations we determine the empirical mean value for synthetic and real networks. Experiments show that for random model networks there is no significant difference between the two value. On the other hand, interestingly, for some real networks the observed empirical mean intersection highly differs from the minimum expected. Our finding may provide a new perspective of investigating real small-world networks and gives some new insights on the structure of them.

Keywords Random spanning trees · Small-world networks

1 Introduction

Given an undirected, connected graph G, a *spanning tree* T of G is a subgraph that is a tree which includes all nodes of G. In case of weighted graphs the concept of *minimum spanning tree*, that is a spanning tree with the minimum possible total edge weight, has an extraordinary importance. Spanning trees play a key role in many applications such as network design including computer networks, telecommunications networks, transportation networks, water supply networks and electrical grids, see e.g. [8];

A. London (✉) · A. Pluhár
Institute of Informatics, University of Szeged, H-6726 Árpád tér 2, Szeged, Hungary
e-mail: london@inf.u-szeged.hu
URL: http://www.inf.u-szeged.hu/london

A. London
Poznań University of Economics and Business, Poznań, Poland

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_26

clustering, see, for instance, single linkage method and applications in finance [10, 11]; or image registration and segmentation [12, 13], just to mention a few without being exhaustive.

The starting point of this paper is the work of Alon et al. [1] where they investigated a zero-sum game played on a connected graph G by two players, namely the *tree player* and the *edge player*. In a turn, tree player chooses a spanning tree T of G, while edge player chooses and edge e of G. The cost $\text{cost}(T, e)$ of edge player defined as follows: if e is an edge of T then $\text{cost}(T, e) = 0$, while if e is not in T, then the cost is the length of the unique cycle (also called fundamental cycle) formed by adding e to T. They derived bounds on $\text{cost}(T, e)$ for various graph classes and pointed out that the game arose in connection with the k-server problem, an online optimization problem, on road networks. In a different scientific community a similar game introduced and analyzed [2], which later referred as a *secure broadcast game* e.g. in [3]. In this game *broadcaster* B, located in a network node wants to broadcast a message to all other nodes. This is accomplished by choosing a spanning tree. The other player, called *eavesdropper* E can observe the transmission along a single link. B wins the game if the spanning tree avoids E's edge, while E wins if the tree includes it. The game has been used as an interpretation of the *p-modulus theory* of graphs, see [3, 4].

A natural continuation of the game theoretic work is to consider another tree player instead of the edge player. In this scenario we are given a connected graph G, two players P_1 and P_2 choose a spanning tree, T_1 and T_2 of G respectively, by not knowing each other's choice. P_1's goal is to maximize the number of common edges of T_1 and T_2, i.e. the intersection of two trees, while P_2's goal is the opposite.

Finding the optimal strategies and the value of this game seems far from being obvious for general graphs. Optimal strategies are often sought as so-called mixed strategies, means that players pick each spanning tree with a certain probability. For some graph classes the optimal strategy for *both* players are taking a uniform random spanning tree. Therefore we arrive to the problem of *intersection of random spanning trees*, that is the number of common edges of two spanning trees that are chosen uniformly at random. Of course this choice is not an optimal strategy of the game in general, but the parameter it provides (the size of the intersection) captures a lot about the structure of a graph. For real small-world networks the size of the mean intersection provides meaningful insights on the structure of them. We think that mainly the modular structure and the number of weak links (i.e. links between communities) [17] drive the value of the parameter since links between communities tend to be part of spanning trees of the network.

Spanning trees in complex networks have been investigated from various perspectives. They considered e.g. as skeletons of the network [15], used for dimension reduction [10] or utilized for efficient visualization of evolving networks [16]. In this work provide another perspective and based on that some new insights on the marco-scale structure of small-world networks. For instance, one expects large intersection if the graph has high Newman modularity, since the edges between the clusters, the weak links are over-represented in the spanning trees. On the other our findings suggest that network heterogeneity, i.e. heterogeneous degree distribution, by itself does not indicate higher spanning tree intersection as minimally expected.

Next we discuss the main concepts and definitions we are dealing with. Then we derive a lower on the expected value of intersection of two random spanning trees by considering a more general problem. We present experimental results on both synthetic network models and real-world networks and provide some potential directions of future work as well.

Throughout this paper, $G = (V, E)$ will be a finite, connected, undirected and unweighted graph with $|V| = n$ and $|E| = m$.

2 Random Spanning Trees

Let $\mathcal{T}_G$ be the set of all spanning trees of a graph G. The cardinally of $\mathcal{T}_G$, by Kirchhoff's matrix tree theorem is explicitly known and can be calculated as the product of the positive eigenvalues of the graph Laplacian divided by n. This allows to consider *random spanning trees* of G chosen randomly from among all spanning trees with equal probability. Another results of Kirchhoff states that the probability of an edge $e \in T$ for a random tree T equals to the effective resistance of that edge, when considering the graph as an electrical network with unit edge conductance. For more details and computation methods see [19]. Besides, there are several good Monte Carlo Markov chain algorithms providing uniform spanning trees that can be used in simulations, see, for instance, Wilson's algorithm [14].

We should note here, that more general random spanning trees can be considered given any probability mass function over $\mathcal{T}_G$. Related problems and results are discussed e.g. in [4]. In this work we are dealing with the case of uniform distribution only.

2.1 Minimum Expected Intersection

In order to provide a lower bound on the expected value of the number of common edges of two random spanning trees of G we consider a more general problem on hypergraphs. We point out that the calculations can be easily done directly for graphs and its random spanning trees.

Definition 1 Let $\mathcal{H} = (X, F)$ be a uniform hypergraph where X denotes the set of nodes, while F denotes the set of hyperedges of $\mathcal{H}$. Let $|X| = \mu$, $|F| = k$ and $|T| = \nu$ for each hyperedge $T \in F$.

Accordingly, we define the expected intersection of two hyperedges of $\mathcal{H}$ as follows:

Definition 2 Let T_1 and T_2 be two hyperedges of $\mathcal{H}$ chosen independently with probability $1/k$. The expected intersection of T_1 and T_2 is $\mathrm{In}(\mathcal{H}) := \mathbb{E}(|T_1 \cap T_2|)$.

The key theorem that works for any hypergraph $\mathcal{H}$ is the following.

Theorem 1 $\mathrm{In}(\mathcal{H}) \geq \nu^2/\mu$.

Proof The degree $d_{\mathcal{H}}(x)$ of a node $x \in X$ is the number of hyperedges contains x. Then $\sum_{x \in X} d_{\mathcal{H}}(x) = k\nu$, while the average degree is $\overline{d}_{\mathcal{H}} = k\nu/\mu$. If we pick two hyperedges randomly, then the probability they both containing a node x having degree $d_{\mathcal{H}}(x)$ is

$$\Pr(x \in T_1 \cap T_2) = \frac{\binom{d_{\mathcal{H}}(x)}{2}}{k^2}$$

and hence the expected number of edges in the intersection is

$$\mathbb{E}(|T_1 \cap T_2|) = \sum_{x \in X} \frac{\binom{d_{\mathcal{H}}(x)}{2}}{k^2} \geq \mu \frac{\binom{\frac{k\nu}{\mu}}{2}}{k^2} = \frac{\nu^2}{\mu},$$

by noting that the expectation is minimized when all node has the same degree.

Observation 1 *Let $p(e) = \Pr(e \in T)$, where $T \in F$ chosen uniformly at random. The exact intersection value is* $\mathrm{In}(\mathcal{H}) = \sum_{e \in X} p^2(e)$.

Now let us define $\mathcal{H} = (X, F)$ as follows. Each node $x \in X$ of $\mathcal{H}$ corresponds to an edge $e \in E$ of G and a hyperedge $T \in F$ corresponds to a spanning tree of G. Hence $\mathcal{H}$ is an $(n-1)$-uniform hypergraph with $|X| = m$ and applying Theorem 1 we get the following lower bound for graphs.

Corollary 1 *Given a connected graph G of n nodes and m edges and $T_1, T_2 \in \mathcal{T}_G$ two random spanning trees. The minimum expected intersection of T_1 and T_2 is* $(n-1)^2/m$.

We note here, that in a special case when G is the union of two spanning trees the expected intersection $(n-1)/2$ can be easily obtained and it equals to the value of the game in the 2-player game on this graph introduced above.

In the next section by performing numerical experiments, we investigate that how the the experimentally observed intersection values differs from the derived minimum expected intersection. In order to provide a simple metric that shows how random spanning trees of a network likely to intersect (compared to the minimum expected intersection) we use the following normalized score:

$$\mathrm{RTI} = \frac{\text{observed mean} - \text{min. expected}}{\text{maximum} - \text{min. expected}} = \frac{\text{observed mean} - \frac{(n-1)^2}{m}}{n - \frac{(n-1)^2}{m}}.$$

RTI takes values between 0 and 1. RTI $= 0$ means that the observed mean intersection equals to the minimum expected intersection, while RTI $= 1$ if and only if the network is a tree. (Note that the maximum possible intersection of two spanning trees is $n-1$, but we used n in the formula to avoid division by zero).

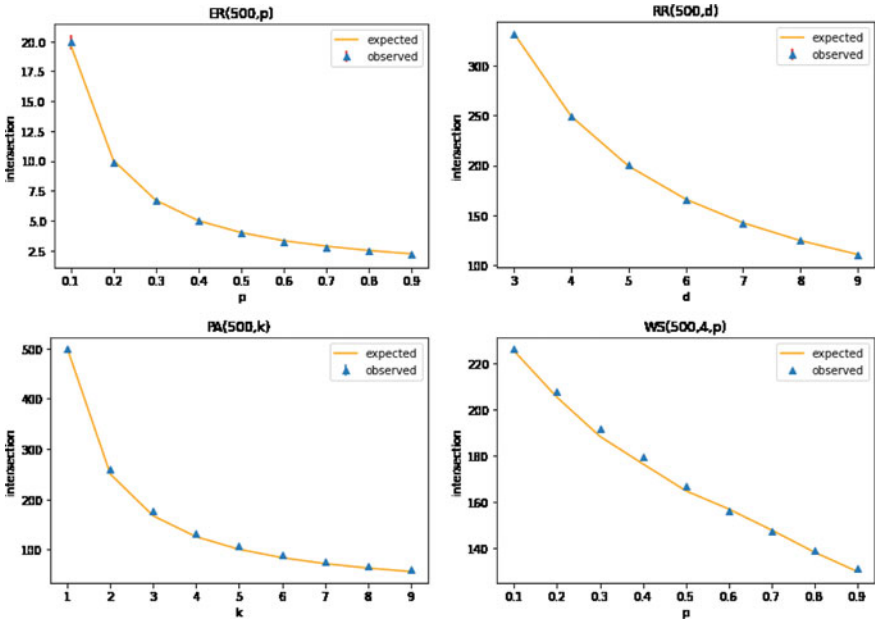


Fig. 1 Experimental results on synthetic networks Erdős-Rényi, random regular, Barabási-Albert and Watts-Strogatz (top-down) with 500 nodes, respectively. Blue triangles (with red lines indication std) shows the observed results based on bootstrap experiments, while orange line indicates the minimum expected intersection

3 Experiments

We performed our experiments with the following setup. For each network we randomly sampled 100 pairs of spanning trees with bootstrap sampling (i.e. a sample may be used multiple times) and calculated the empirical mean intersection and the corresponding standard deviation. This is referred as *observed mean intersection*. The *minimum expected intersection*, according to the findings of the previous section was calculated as $(n-1)^2/m$.

3.1 Experiments on Random Model Networks

Firstly we performed experiments on some synthetic networks including Erdős-Rényi random graphs $G(n, p)$ [5] (i.e. a graph of n nodes with the probability of drawing an edge between any pair of nodes is p) with $n = 100, 200, 500$ and $p = 0.1, 0.2, \ldots, 1$; random regular graphs $RR(n, d)$ (i.e. a random graph of n nodes where each node having degree d) with $n = 100, 200, 500$ and $d = 3, 4, \ldots, 9$; Preferential attachment networks $PA(n, k)$ [6] (i.e. a network of n nodes created

by a process where at each step a new node is added to the network and connects to k already existing nodes) with $n = 100, 200, 500$ and $k = 1, 2, \ldots, 10$; Watts-Strogatz networks $WS(n, k, p)$ [7] (i.e. starting from a 4-regular network of n nodes each edge is rewired with probability p) with $n = 100, 200, 500$, $k = 4$ and $p = 0.1, 0.2, \ldots, 1$. For each parameter we generated 10 graph (e.g. 10 graphs for $G(200, 0.1)$ or $BA(500, 2)$, etc.) and for each graph we sampled 100 pairs of spanning trees. Then sample means and standard deviations were calculated.

The results for $n = 500$ are shown in Fig. 1 and suggest that in case of random networks models the minimum expected intersection equals to the observed (real) intersection obtained by the simulations. Similar conclusions can be drawn when $n = 100$ and $n = 200$. For almost all cases the standard deviation was close to 0 confirming the robustness of the simulations. An interesting observation is that heterogeneous degree distribution (i.e. power-law), by itself does not indicate higher spanning tree intersection as minimally expected. Deeper investigation of the intersection value as the function of the model graph's parameter (like parameter p for $WS(n, k, p)$) could be a topic of another study. For instance, since $m \propto \binom{n}{2} p$ for $G(n, p)$ we expect intersection size $2/p$ that is confirmed by the simulations (Fig. 1 top right).

Here we are more interested in that how the minimum expected value differs from the real one in case of real complex networks. In the next section we present our experiments considering networks having various global structural characteristics.

3.2 Experiments on Real-Networks

To perform experiments on real-world networks we considered the largest connected component in case of disconnected networks and we did not take into account the direction and/or weight of the edges in case of directed and/or weighted networks, respectively.

We compared the RTI score with some metrics that are commonly used as indicators of the small-world structure of a network. These are network density $\rho = m/\binom{n}{2}$, clustering coefficient $cc = 3 \times$ number of triangles$/\binom{n}{3}$ and average shortest path length $\overline{\ell} = 1/\binom{n}{2} \cdot \sum_{i \neq j} \ell_{ij}$, where ℓ_{ij} is the length of the shortest path between nodes i and j.

Table 1 shows the results on 12 networks including social, biological and technological networks as well (the network data is available on websites [20–22]). The network size varies from $n = 15$ (marriage links between Florentine families) to $n =$33,696 (Enron email communication network). The average shortest path length varies between $\overline{\ell} = 2.408$ (Zachary karate club) and $\overline{\ell} = 6.05$ (Arxiv collaboration network on general relativity and quantum computing), except the US power grid technological network with $\overline{\ell} = 19$, while almost all network have relatively high clustering coefficient ensuring a strong small-world property of the examined networks.

Table 1 Network statistics and random spanning tree intersection of some real-world network

network	n	m	ρ	cc	$\overline{\ell}$	min. exp.	obs. mean	RTI
Florentine	15	20	0.190	0.191	2.486	9.8	10.5 (0.888)	0.166
Zachary	34	78	0.140	0.256	2.408	13.961	15.00 (2.291)	0.052
Dolphins	62	159	0.084	0.309	3.357	23.40	28.34 (2.19)	0.131
Adjnoun	112	425	0.068	0.157	2.535	29.00	38.8 (4.3)	0.120
Jazz	198	2742	0.141	0.520	2.235	14.15	23.51 (3.66)	0.051
C-elegans	297	2148	0.049	0.180	2.455	40.79	57.8 (5.5)	0.067
NetSci	379	914	0.013	0.431	6.042	156.328	182.12 (7.98)	0.116
Wiki-Vote	889	2914	0.007	0.127	4.096	270.61	430.47 (9.66)	0.259
Polblogs	1222	16,717	0.022	0.226	2.737	89.18	300 (10.55)	0.186
Arxiv GR-QC	4158	13,428	0.002	0.629	6.053	1287	2102 (21.65)	0.284
Power grid	4941	6594	0.0005	0.103	18.99	3700.87	3921 (16.71)	0.178
Email-Enron	33,696	180,811	0.0003	0.085	4.025	6279.225	16,602 (64.73)	0.376

The highest RTI score was obtained for the Enron email network (RTI = 0.376) where the difference between the empirically observed mean and minimum expected intersection is more than 10,000, i.e. more than 5% of the total number of edges. The Arxiv collaboration network and the Wiki-Vote network also provided high RTI (0.284 and 0.259, resp.). We observed that RTI value larger than 0.1 suggests significant difference between the real tree intersection value and the expected lower bound with respect to the total number of edges in the network. Besides the RTI scores, it is worth to compare the normalized scores obtained by dividing the two intersection values (min. expected and observed, resp.) by the size of a spanning tree of the network ($n - 1$), both taking values between 0 and 1. The difference of the two values shows that how much larger the size of the intersection (in percentages) in reality compared to that is minimally expected on the bases of the spanning tree size. In case of the networks investigated here this value varies between 3% (Zachary) and 30% (Enron) with relatively large values in between, e.g. for Wiki-Vote (18%), PolBlogs (17.2%) and Arxiv-GR-QC (19.6%).

Comparing RTI and Newman modularity scores (calculated by considering the community structure given by the "Leuven" fast greedy method [18]) provides another interesting aspect of random tree intersection. Among the investigated networks the Zachary, the Jazz musician and the C-elegans networks show the lowest RTI score (in the range 0.051–0.067) and the lowest modularity score (in the range 0.369–0.439) as well. On the other hand, Email-Enron and Arxiv GR-QC (having the highest RTI scores) has modularity 0.504 and 0.793. However, NetSci citation network has the highest modularity (0.838), suggest that not just modularity, but it seems also the density correlates with RTI, the former positively, while the latter negatively. Deeper analysis on correlations and relationship with other metrics remains the topic of a future study.

4 Conclusions and Future Work

In this work we investigated that how randomly chosen spanning trees likely to intersect in order to get some new insights on the macro-scale structure of complex networks. At first, we derived a lower bound on the expected value of intersection (i.e. the number of common edges of two randomly chosen spanning trees) as a function of the number of nodes and edges of the network. We compared this value with the real (empirical) intersection value (obtained by simulation experiments) and observed that in case of random model networks there is no significant difference between the two value. On the other hand, more interestingly, experiments show that for some real networks the observed mean intersection highly differs from the minimum expected suggesting the existence of special links likely to appear in most of the spanning trees. For instance, we expect large intersection if the graph has high Newman modularity, since the edges between the clusters, sometimes called weak links, are over-represented in the spanning trees. Moreover if many such links are in the intersection then the connectivity of the network can be more easily destroyed. The edges appears more often in the intersection are more crucial for the connectivity and in this sense, RTI could be thought as a measure for robustness or resilience of the network. We are planning further investigations in this direction. Another possibilities for future work are to derive the exact bounds for random model networks and examine the relation between the tree intersection value and widely-used global structural metrics of complex networks.

Acknowledgements We are thank to the anonymous referees for their useful comments that help to improve the quality of the paper. This work was supported by National Research, Development and Innovation Office-NKFIH Fund No. SNN-135643.

References

1. Alon, N., Karp, R. M., Peleg, D., West, D.: A graph-theoretic game and its application to the k-server problem. SIAM J. Comput. **24**(1), 78–100. (1995). https://doi.org/10.1137/S0097539792224474
2. Gueye, A., Walrand, J.C., Anantharam, V.: Design of network topology in an adversarial environment. In: International Conference on Decision and Game Theory for Security, pp. 1–20. Springer, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17197-0_1
3. Kottegoda, K.: Spanning tree modulus and secure broadcast games. Ph.D. Thesis, Kansas State University (2020). https://hdl.handle.net/2097/40756.24
4. Albin, N., Clemens, J., Hoare, D., Poggi-Corradini, P., Sit, B., Tymochko, S.: Fairest edge usage and minimum expected overlap for random spanning trees. Discrete Math. **344**(5), 112282 (2021). https://doi.org/10.1016/j.disc.2020.112282
5. Erdős, P., Rényi, A.: On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci. **5**(1), 17–60 (1960). http://dx.doi.org/10.1515/9781400841356.38
6. Barabási, A. L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999). http://dx.doi.org/10.1126/science.286.5439.509
7. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature **393**(6684), 440–442 (1998). https://doi.org/10.1038/30918
8. Wu, B.Y., Chao, K.M.: Spanning trees and optimization problems. Chapman and Hall/CRC, New York (2004). https://doi.org/10.1201/9780203497289
9. Gower, J.C., Ross, G.J.: Minimum spanning trees and single linkage cluster analysis. J. Royal Stat. Soc. Ser. C (Appl. Stat.) **18**(1), 54–64 (1969). https://doi.org/10.2307/2346439
10. Mantegna, R.N.: Hierarchical structure in financial markets. Eur. Phys. J. B **11**(1), 193–197 (1999). https://doi.org/10.1007/s100510050929
11. Tola, V., Lillo, F., Gallegati, M., Mantegna, R.N.: Cluster analysis for portfolio optimization. J. Econ. Dyn. Control **32**(1), 235–258 (2008). https://doi.org/10.1016/j.jedc.2007.01.034
12. Ma, B., Hero, A., Gorman, J., Michel, O.: Image registration with minimum spanning tree algorithm. In: Proceedings International conference on Image Processing, vol. 1, pp. 481–484. IEEE (2000). https://doi.org/10.1109/ICIP.2000.901000
13. Xu, Y., Uberbacher, E.C.: 2D image segmentation using minimum spanning trees. Image Vision Comput. **15**(1), 47–57 (1997). https://doi.org/10.1016/S0262-8856(96)01105-5
14. Wilson, D.B.: Generating random spanning trees more quickly than the cover time. In: Proceedings of the 28th Annual ACM Symposium on Theory of Computing, pp. 296–303 (1996). https://doi.org/10.1145/237814.237880
15. Kim, D.H., Noh, J.D., Jeong, H.: Scale-free trees: the skeletons of complex networks. Phys. Rev. E **70**(4), 046126 (2004). https://doi.org/10.1103/PhysRevE.70.046126
16. Chen, C., Morris, S.: Visualizing evolving networks: minimum spanning trees versus pathfinder networks. In: IEEE Symposium on Information Visualization, pp. 67–74. IEEE (2003). https://doi.org/10.1109/INFVIS.2003.1249010
17. Granovetter, M.S.: The strength of weak ties. Am. J. Sociol. **78**(6), 1360–1380. (1973). https://doi.org/10.1109/INFVIS.2003.1249010
18. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. Theory Exp. (10), P10008 (2008). https://doi.org/10.1088/1742-5468/2008/10/P10008
19. Ellens, W., Spieksma, F.M., Van Mieghem, P., Jamakovic, A., Kooij, R.E.: Effective graph resistance. Linear Algebra Appl. **435**(10), 2491–2506 (2011). https://doi.org/10.1016/j.laa.2011.02.024
20. http://www.personal.umich.edu/mejn/netdata/
21. https://networkrepository.com/
22. https://snap.stanford.edu/data/

Node Classification Based on Non-symmetric Dependencies and Graph Neural Networks

Emanuel Dopater and Miloš Kudělka

Abstract One of the interesting tasks in social network analysis is detecting network nodes' roles in their interactions. The first problem is discovering such roles, and the second is detecting the discovered roles in the network. Role detection, i.e., assigning a role to a node, is a classification task. Our paper addresses the second problem and uses three roles (classes) for classification. These roles are based only on the structural properties of the neighborhood of a given node and use the previously published non-symmetric relationship between pairs of nodes for their definition. This paper presents transductive learning experiments using graph neural networks (GNN) to show that excellent results can be obtained even with a relatively small sample size for training the network.

Keywords Complex network · Graph neural network · Non-symmetric dependency · Node prominency

1 Introduction

Today's technologies offer us many different ways to connect with each other. As a whole, these connections form networks of different types, and we are represented as nodes in them and our connections as edges. These edges are not symmetrical and have different strengths (weights). In the world of networks organized in this way, communities emerge, and nodes play different roles within or between them. Our paper focuses on working with the roles that nodes play in networks.

The first area we need to work with roles is "role discovery." Its goal is to use different methods to understand what are the internal similarities between nodes. Then,

E. Dopater (✉) · M. Kudělka
VSB - Technical University of Ostrava, 17. listopadu 2172/15, 708 00 Ostrava-Poruba, Czech Republic
e-mail: emanuel.dopater.st@vsb.cz

M. Kudělka
e-mail: milos.kudelka@vsb.cz

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_27

based on these similarities, nodes can be divided into groups, and each of these groups becomes a role. An overview of the many different methods is provided, for example, by Rossi and Ahmed in [14]. They describe roles as the main node-level connectivity patterns such as star-center/edge nodes, peripheral nodes, near-clique nodes, and bridge nodes that connect different regions of the graph, among many other types of connectivity patterns. In essence, they divide role discovery approaches into graph-based, feature-based, and hybrid. Network embedding provides a network representation in which similar nodes are close together. Pengfei et al. in [7] present a survey on role-based network embeddings and a general framework for understanding role-oriented network embedding and a two-level categorization to better classify existing methods. Alvarez-Gonzalez et al. in [1] look at the problem of finding inductive network embeddings in large networks without domain-dependent node/edge attributes.

The second area is "role detection", where we know the roles and the rules (algorithm) based on which we are able to assign each node of the network to one of the roles. In our approach (see Kudelka et al. [9]), we use an algorithm based on the analysis of non-symmetric relationships around the given node, which works with three different roles. Alternative approaches to role detection based on analysis of structural properties are provided by Ghalmane et al. in [3], and Henderson et al. in [6].

The third area is the classification of network nodes, i.e., assigning roles to nodes whose roles we do not know but with knowledge of roles of other nodes. In this case, we can use various methods from machine learning and artificial intelligence. In this paper, we use transductive learning with graph neural networks (see Hamilton in [5]). Relatively similar work has been done, e.g., by Lizhong Xiao et al. in [16]. In this work, they classified nodes in social networks using traditional approaches such as DeepWalk and logistic regression, and the GNN-based GraphSage classifier. They compared the results of the traditional and GNN-based approaches and determined the superiority of the GNN-based approach.

The text is organized as follows. We first summarize the previously published description of non-symmetric dependency and prominency and add the definition of prominency weight that we need for transductive learning experiments. In the next section, we describe the three datasets that were the subject of our experiments. This is followed by the experiments section and a discussion of the results. In the final section, we summarize the results we have achieved.

2 Dependency and Prominency

Dependency [9] of node x on node y is defined as

$$D(x, y) = \frac{w(x, y) + \sum_{v_i \in CN(x,y)} w(x, v_i) \cdot r(x, v_i, y)}{\sum_{v_j \in N(x)} w(x, v_j)} \tag{1}$$

$$r(x, v_i, y) = \frac{w(v_i, y)}{w(x, v_i) + w(v_i, y)}, \tag{2}$$

where $CN(x, y)$ is set of all common neighbors of x, y, $N(x)$ is a set of all neighbors of node x, $w(x, y)$ is weight of edge between node x, y and $r(x, v_i, y)$ is the coefficient of the dependency of node x on node y via the common neighbor v_i. For the following text, we binarize the dependency so that node x is dependent on node y if $D(x, y) \geq 0.5$. This dependency does not apply mutually. Node y does not have to be dependent on node x (i.e., in general, the dependency relationship is non-symmetric).

The analysis of non-symmetric dependencies is mainly intended for weighted networks. However, this non-symmetry also manifests itself in unweighted networks (see Fig. 1).

Based on the dependency type, we can divide the neighbors of a given node into four groups (some of the groups may be empty). The first group includes those nodes on which the given node is dependent; the second group includes those dependent on the given node. The third group includes the nodes with which the node is mutually dependent, and the fourth group includes nodes mutually independent with the given node. Next, let us define four group properties for a given node as depicted in Fig. 2.

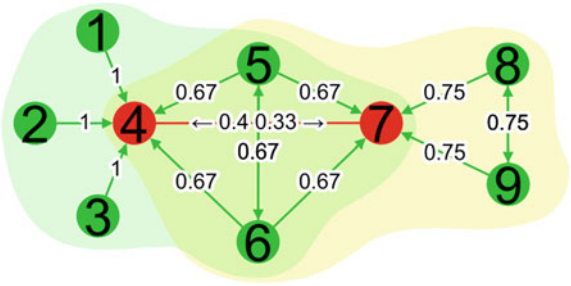


Fig. 1 Dependencies and strongly-prominent nodes [9]

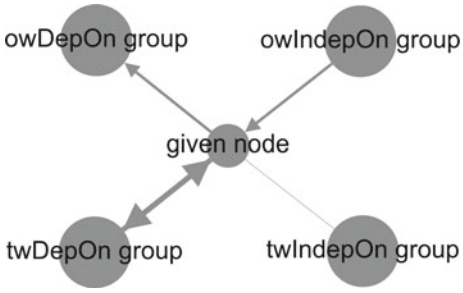


Fig. 2 Four neighbor groups of the given node

owDepOn is the number of neighbors on which a given node is unilaterally dependent.

owIndepOn is the number of neighbors that are unilaterally dependent on the node.

twDepOn is the number of neighbors with which the given node is mutually dependent.

twIndepOn is the number of neighbors with which the given node is mutually independent.

These four properties and neighbor groups allow us to define roles in the network. The roles are based only on the structural properties of the network and the dependencies derived from them. These roles describe the *prominency* of a node:

A strongly prominent node is not dependent on any of its neighbors, and at least one of its neighbors is dependent on it ($owIndepOn > 0 \wedge owDepOn = 0 \wedge twDepOn = 0$), see (see red nodes in Fig. 1).

A weakly prominet node has at least one neighbor that is dependent on it, and the node itself is not dependent on that neighbor ($owIndepOn > 0 \wedge (owDepOn > 0 \vee twDepOn > 0)$).

A non-prominent node is a node that is neither strongly nor weakly prominent, see green nodes in Fig. 1.

Although it is not evident at first glance, it is necessary to explain that the nodes in the same role can differ. This is because they can have substantial differences in degree and clustering coefficient, and similarly, they can have different types of dependencies with their neighbors. Therefore, when learning, as will be shown later, it makes sense to perform a structural analysis on only a small part of the network (or another network) and for the rest to investigate whether the learned information is sufficient to classify the other nodes.

2.1 Prominency Weight

For use in the experiments presented below, let us define the *prominency weight*. This weight for each node determines how independency over dependency prevails for that node, and it may be the case, for example, that a weakly-prominent node may have a much higher weight than a strongly-prominent one.

Further, let

$$pScore = \frac{owIndepOn}{owIndepOn + \frac{1}{2} \cdot (owDepOn + twDepOn)} \tag{3}$$

Table 1 Networks used in experiments

Network	n	m	k_{avg}	k_{max}	CC	Q	$NP\%$	$SP\%$	$WP\%$
condmat	16,726	47,594	5.691	107	0.621	0.874	64.40	13.59	22.01
enron	36,692	183,831	10.020	1383	0.497	0.593	93.93	5.48	0.60
facebook	63,731	817,090	25.642	1098	0.221	0.605	69.46	28.75	1.79

and

$$pWeight = \frac{owIndepOn^2}{owIndepOn + twIndepOn}, \tag{4}$$

then the prominency weight is defined as

$$Prominency\ Weight = pScore \cdot pWeight \tag{5}$$

Note that all four properties related to dependencies in the neighborhood of a given node are used to calculate the prominency weight.

3 Datasets

In our experiments, we work with three networks of three different types (co-authorship, email, and social). These networks differ in their properties and also in the relative frequency of occurrences of each role. These networks are:

cond-mat (condmat) [11] network of coauthorships between scientists posting preprints on the Condensed Matter E-Print Archive.
Available at http://www-personal.umich.edu/~mejn/netdata/

Email-Enron (enron) [17] communication network that covers all the email communication within a dataset of around half a million emails.
Available at https://snap.stanford.edu/data/email-Enron.html

fb-friends (facebook) [15] network of friendship where nodes are users and edges between the users represent friendship relations.
Available at http://networkrepository.com/fb-wosn-friends.php.

The structural properties of these networks are presented in Table 1. For each network, the table lists the number of nodes and edges, average and maximum degree, clustering coefficient and Louvain modularity. This is followed by the percentage of the relative frequency of occurrences of non-prominent, strongly-prominent, and weakly-prominent nodes. Note that in terms of roles, all sets are highly imbalanced, which can strongly affect classification quality.

4 Node Prominency Classification

We aimed to carry out experiments with classifiers based on layers of the graph neural network [5] and predict the prominencies of the nodes using various information about the nodes in three different networks in Table 1. We performed a classification in the manner of transductive learning, where we randomly chose 20% of the nodes intended for training, and the remaining nodes were used for the testing and evaluation of the classifier. In this work, we performed only classification with transductive learning. However, our goal was to examine the effectivity of GNNs to leverage this effectivity on inductive learning tasks in our future research.

Procedure of experiment When performing the experiments, we followed our experimentation scheme, which is shown graphically in Fig. 3. In this subsection, we describe only the basics of our approach.

Data preparation The data preparation part aimed to prepare the data in a suitable form to train the classifier.

Node attributes: In all experiments, we classified the prominency of nodes, so the prominency types were class labels of the nodes. For each network in Table 1, we performed three independent experiments, whereas node features (information about the nodes spread by message passing) were used:

1. Network connectivity—each node had one-dimensional vector containing scalar 1.
2. Node degree and local cluster coefficient, we used standard scaling for this feature.
3. Prominency weight (see Eq. 5), we used standard scaling for this feature.

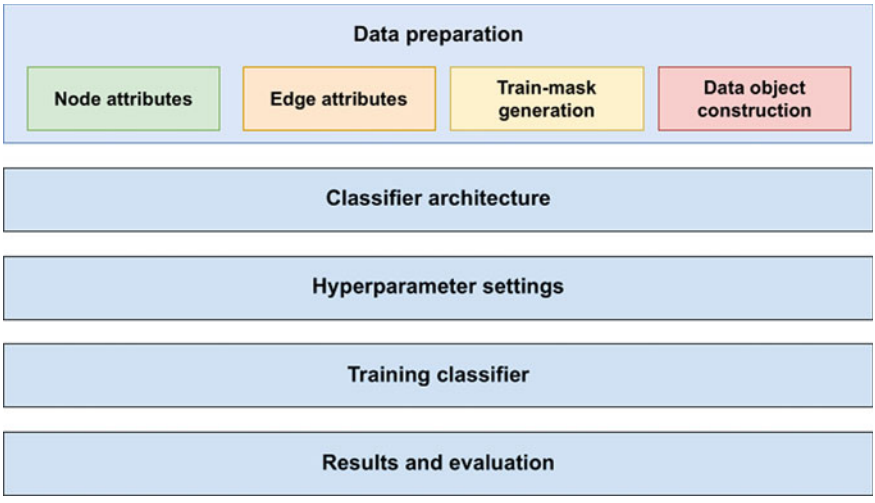


Fig. 3 Visualization of experiment scheme

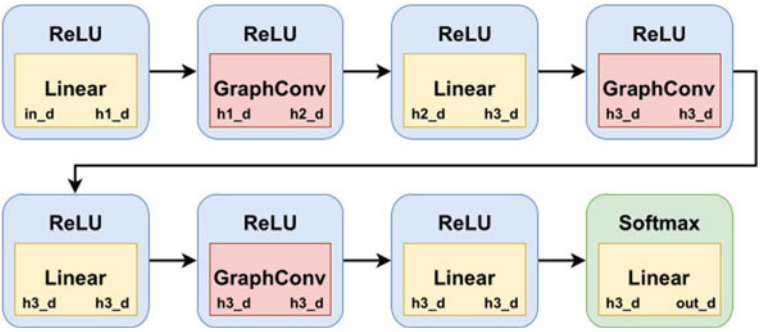


Fig. 4 Node classifier architecture

Edge attributes: We did not use any information on the edges. In this step, we only loaded the adjacency list containing information about the network's connections.

***Train-mask* generation**: A *train-mask* is an array indicating which nodes are intended for training and which are for evaluating the classifier. For training and testing our model, we used five-fold cross-validation. Our *train-mask* was generated on the basis of the cross-validation split in each of these splits.

Data object construction: In the final step of data preparation, we placed all the information prepared about the network into one object that represents the whole network.

Classifier architecture Our classifier architecture, with layers and their dimension and activation functions, is represented graphically in Fig. 4. We tried to perform classifications with several architectures and GNN layers, but the results were similar. We have also determined that three GNN layers are sufficient to extract all relevant information from the neighborhood. Our goal was not to find the best architecture and most suitable GNN layers, so we stayed with architecture in Fig. 4.

In the inner rectangle, there is a neural network layer with its input and output dimensions, and the round rectangle represents the activation function. Arrows represent flow through layers.

Hyperparameter settings Our hyperparameters were set to the recommended and used values in the literature, and we experimentally verified their performance. Our training setting was as follows:

- The number of folds in cross-validation is 5.
- in_d was the same value as the node feature vector size.
- $h1\_d$ was 20.
- $h2\_d$ was 30.
- $h3\_d$ was 40.
- out_d was 3 (which was the number of predicting classes).
- As an optimizer algorithm, we chose *Adam* [8].
- The learning rate of *Adam* was 0.005.

Table 2 F1-score describing the GNN classifier performance for each role

	Non-prominent		Strongly-prominent		Weakly-prominent	
	Mean	SD	Mean	SD	Mean	SD
condmat_C	0.904	0.027	0.678	0.009	0.601	0.027
condmat_DC	0.887	0.059	0.707	0.016	0.597	0.070
condmat_PW	0.987	0.010	0.698	0.025	0.763	0.056
enron_C	0.941	0.027	0.584	0.114	0.083	0.074
enron_DC	0.895	0.091	0.605	0.212	0.130	0.110
enron_PW	0.938	0.050	0.604	0.171	0.148	0.118
facebook_C	0.889	0.009	0.787	0.009	0.225	0.030
facebook_DC	0.953	0.005	0.895	0.008	0.677	0.024
facebook_PW	0.939	0.013	0.862	0.022	0.567	0.058

- The learning rate decay of *Adam* was 0.0005.
- The loss function was the cross-entropy loss function.
- The number of training epochs was 1000.
- We used class weighting due to class imbalances during training. The class weight is a number that indicates a "weight" of the trained node when the optimization algorithm updates the learnable parameters. We computed the class weight simply by equation $class\_weight_i = (1 - \frac{L_i}{D})^2$, where i is index of class, L_i is the number of nodes in the i class and D is the number of all nodes.

Training classifier For training, we used cross-validation with five splits. In each of these splits, we initialized a new instance of a classifier. One fold was used for training and four folds for evaluating the classifier. Before the generation of cross-validation splits, the set of nodes was randomly shuffled.

Results and evaluation For evaluation, we used the trained classifier to predict the prominencies of the other 80% of nodes and compare these predicted values with the real ones. We used the F1-score as a measure of evaluation for individual classes and the balanced accuracy (BA) and Matthew's correlation coefficient (MCC) to evaluate classification as a whole. We used multi-class version of BA and MCC measures (see "Used tools" at page 355). Due to the five-fold cross-validation, we have five different results for each experiment, so we calculated the mean and standard deviation for the measures used. The results of the F1-scores of the experiments are presented in the Table 2, and BA and MCC in the Table 3. In these tables, suffix "C" means that as node features, only network connectivity was used, "DC" means local degree coefficient and node degree, and "PW" means prominency weight.

The results with network connectivity (one-dimensional vectors with scalar 1 are remarkable because GNN can extract significant information to classify the prominency role of the node from its neighborhood structure with relatively high accuracy. Additional information about the nodes can increase the accuracy of the classification.

Table 3 BA and MCC measure describing the multi-class classification performance

	BA		MCC	
	Mean	SD	Mean	SD
condmat_C	0.743	0.011	0.635	0.034
condmat_DC	0.753	0.020	0.632	0.070
condmat_PW	0.823	0.024	0.804	0.040
enron_C	0.689	0.051	0.566	0.100
enron_DC	0.798	0.141	0.507	0.152
enron_PW	0.715	0.078	0.584	0.143
facebook_C	0.674	0.018	0.689	0.013
facebook_DC	0.873	0.012	0.850	0.013
facebook_PW	0.803	0.022	0.807	0.031

Used tools

Here we list the important tools we used to carry out the experiments:

NetworkX [4] is a Python library for network data manipulation. We used it to calculate the local clustering coefficient and node degree.

PyTorch [12] is an extensive Python-based machine learning framework. We used it to build and train the classifier.

PyTorch Geometrix [2] is a library built on top of PyTorch and provides tools and layers to deal with network data in machine learning. We used its implementation of the GraphConv layer [10].

scikit-learn [13] is an extensive Python library with a large number of different tools for machine learning. We used it for evaluation measures.

5 Discussion

The performance and stability of the GNN classifier are both for individual roles (classes) and overall at a very high level. The only exception is the classification of weakly-prominent nodes, especially for the enron network (see Table 2). This is mainly because for the enron network, only 0.60% of the nodes are in this role. Even for the facebook network, which has only 1.79% weakly-prominent nodes, the classification value is low. However, unlike the enron network, the use of features works here, and it does not really matter whether it is a combination of clustering coefficient + degree or prominency weight. For the other roles, the impact of features is much smaller, although in most cases, there is at least a small improvement in the performance of the GNN classifier when they are used.

Table 3 shows that if we use multi-class measures to assess the performance of the GNN classifier as a whole, then the results here are also very good. This is particularly

evident for the condmat and facebook networks, where the multi-class MCC exceeds 0.8 when using prominency weight as a feature (for facebook network, this is also true for the clustering coefficient + degree feature combination). This is even though MCC evaluates the performance of the classifier very strictly in the case of highly imbalanced datasets, which is also the case for our experimental datasets.

6 Conclusion

In this paper, we presented experiments on the classification of network nodes in different roles. The roles are based on the analysis of non-symmetric dependencies in the neighborhood of a given node. For extremely large and time-varying networks, it is important to assume that we have knowledge of only a small part of the network for learning. We used graph neural networks for learning and showed that even with a relatively small sample of network training data, the performance of the GNN classifier is very high. This is despite the fact that these were highly imbalanced datasets.

A limitation of our approach is transductive learning, which works with the same network for training and testing, even though only a small part of it is used for training. Therefore, in future research, we will focus on inductive learning for different types of networks. Based on the results of our experiments, we anticipate that for inductive learning, and thus for finding general rules applicable to unknown situations, it will be necessary to distinguish which type of network is involved (in our case, co-authorship, communication, or social).

Acknowledgements This work is partially supported by SGS, VSB-Technical University of Ostrava, under the grant no. SP2022/77 and Ministry of Health of the Czech Republic under grants no. NU20-06-00269, NU21-06-00370.

References

1. Alvarez-Gonzalez, N., Kaltenbrunner, A., Gómez, V.: Inductive graph embeddings through locality encodings (2020). arXiv:2009.12585
2. Fey, M., Lenssen, J.E.: Fast graph representation learning with pytorch geometric. CoRR abs/1903.02428 (2019). http://arxiv.org/abs/1903.02428
3. Ghalmane, Z., Cherifi, C., Cherifi, H., El Hassouni, M.: Exploring hubs and overlapping nodes interactions in modular complex networks. IEEE Access **8**, 79650–79683 (2020)
4. Hagberg, A.A., Schult, D.A., Swart, P.J.: Exploring network structure, dynamics, and function using networkx. In: Varoquaux, G., Vaught, T., Millman, J. (eds.), Proceedings of the 7th Python in Science Conference, pp. 11–15. Pasadena, CA USA (2008)
5. Hamilton, W.L.: Graph representation learning. Synthesis Lect. Artif. Intell. Mach. Learn. **14**(3), 1–159 (2020)
6. Henderson, K., Gallagher, B., Eliassi-Rad, T., Tong, H., Basu, S., Akoglu, L., Koutra, D., Faloutsos, C., Li, L.: Rolx: structural role extraction & mining in large graphs. In: Proceedings of

the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1231–1239 (2012)
7. Jiao, P., Guo, X., Pan, T., Zhang, W., Pei, Y., Pan, L.: A survey on role-oriented network embedding. IEEE Trans. Big Data (2021)
8. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2017)
9. Kudelka, M., Ochodkova, E., Zehnalova, S., Plesnik, J.: Ego-zones: non-symmetric dependencies reveal network groups with large and dense overlaps. Appl. Netw. Sci. **4**(1), 1–49 (2019)
10. Morris, C., Ritzert, M., Fey, M., Hamilton, W.L., Lenssen, J.E., Rattan, G., Grohe, M.: Weisfeiler and leman go neural: higher-order graph neural networks. CoRR abs/1810.02244 (2018). http://arxiv.org/abs/1810.02244
11. Newman, M.: The structure of scientific collaboration networks. Proc. Nat. Acad. Sci. **98**(2), 404–409 (2001)
12. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: an imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.), Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019). http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
13. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
14. Rossi, R., Ahmed, N.: Role discovery in networks. IEEE Trans. Knowl. Data Eng. **27**(4), 1112–1131 (2014)
15. Viswanath, B., Mislove, A., Cha, M., Gummadi, K.: On the evolution of user interaction in Facebook. In: Proceedings of the 2nd ACM Workshop on Online Social Networks, pp. 37–42 (2009)
16. Xiao, L., Wu, X., Wang, G.: Social network analysis based on graph sage. In: 2019 12th International Symposium on Computational Intelligence and Design (ISCID), vol. 2, pp. 196–199 (2019)
17. Yang, Y., Klimmt, B.: Introducing the enron corpus. In: CEAS Conference (2004)

Mean Hitting Time of Q-subdivision Complex Networks

Pankaj Kumar, Anurag Singh, Ajay K. Sharma, and Hocine Cherifi

Abstract The Mean Hitting Time is a fundamental structural measure of random walks on networks with many applications ranging from epidemic diffusion on networks to fluctuations in stock prices. It measures the mean expected time for a random walker to reach all the source-destination pair nodes in the network. Previous research shows that it scales linearly with the network size for small-world sparse networks. Here, we calculate the Mean Hitting Time for large real-work complex networks and investigate how it scales with the q-subdivision operation used to grow the network. Indeed, this operation is essential in modeling realistic networks with small-world, scale-free and fractal characteristics. We use the Eigenvalues and eigenvectors of the normalized adjacency matrix of the initial network G to calculate the Hitting Time T_{ij} between nodes i and j. We consider two complex real-world networks to analyze the evolution of the Mean Hitting Time as the networks grow with the q-subdivision. Results show that the Mean Hitting Time increases linearly with the value of q. This work provides insight into the design of realistic networks with small Mean Hitting Time.

P. Kumar (✉) · A. Singh · A. K. Sharma
Department of Computer Science and Engineering, National Institute of Technology Delhi, New Delhi, India
e-mail: 212211009@nitdelhi.ac.in

A. Singh
e-mail: anuragsg@nitdelhi.ac.in

H. Cherifi
University of Burgundy, Dijon, France
e-mail: hocine.cherifi@u-bourgogne.fr

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_28

1 Introduction

Random walks on networks are extensively exploited to describe the dynamics of many complex systems in nature and society. Typical applications include information flow in social networks [1–5], mobility patterns [6, 7], image segmentation [8, 9], collaborative recommendation [10, 11], visual saliency [12, 13], community detection [14, 15], and so on. In this context, the Hitting Time is a fundamental structural measure. It is the expected time for a random walker to reach its destination on the network from a starting node. The mean Hitting time is its mean value on all node pairs in the graph. It is denoted by $\bar{T}(G)$. One can use the Mean Hitting time $\bar{T}(G)$ as a measure of the connectedness of the network. The smaller the value of $\bar{T}(G)$, the more connected the network. The Mean Hitting Time $\bar{T}(G)$ indicates the mean search cost in a network [16, 17]. Mean Hitting time $\bar{T}(G)$ is used as global utility of social recommender networks [10].

Real-world Networks are growing every day by some new connection or addition of new nodes. The Mean Hitting time calculation for such dynamic or growing networks is a complex task. Earlier works explored the mean Hitting time for small networks. In this paper, we compute the mean Hitting time for large networks. We use the q-subdivision operation to simulate the network x growth dynamic.

Networks subdivision is an essential operation. It proceeds as follows. For each edge uv, we insert a new node x and replace the edge uv by two edges ux and xv obtaining the subdivision network $S(G)$. The properties of networks sub-division is extensively studied by various researchers. The literature reports many extension of network subdivision such as q-subdivision network ($S_q(G)$) [18] and q-full subdivision network [19]. The $S_q(G)$ is obtain from G by adding $ux_1v, ux_2v, ux_3v, \ldots, ux_qv$ for every edge uv by G [18].

The hierarchical lattice [20] is a complex network model with a unique scale-free fractal topology. It has recently received significant attention. It is generated using the iterative q-subdivision operation. The properties of a traditional subdivision network are well understood. In contrast, $S_q(G)$ properties deserve more investigations.

In Sect. 2, we review previous related work on Mean Hitting Time. Section 3 describes the methodology used to calculate the node-pair Hitting Time of a connected network and its q-subdivision Network. Section 4 presents the calculation of the Mean Hitting Time of a connected network and its q-subdivision network. Section 5 reports the experiments to explore the scale-free properties of the q-subdivision network. We investigate the variation of mean hitting time with an increment of q-value in the q-subdivision network. Section 6 gives the conclusion. We also discuss the limitations of the paper and future work directions.

2 Related Work

This section reviews some related research. A recent work reports the results of an extensive study of a simple connected graph q-subdivision [18]. The authors derive formal expressions of eigenvalues and eigenvectors of normalized adjacency matrix for the q-subdivision. They also report essential results about the two-node Hitting Time, the Kemeny constant for random walks, the two-node resistance distance, and the Kirchhoff index. Considering the scale-free fractal hierarchical lattices, they provide explicit expressions for some of these quantities.

In [21], the authors calculate the Hitting Time of small real-world networks for q-triangulation. q-triangulation network is made up of network G by adding q disjoint path $ux_1v, ux_2v, \ldots, ux_qv$ for every edge uv. Properties of the q-triangulation network are fundamental and extensively used for research purposes. They report results about the variation of Hitting time with increasing value of q in q-triangulation and the calculation of node-pair Hitting time in which one can generate one or two nodes after q-triangulation operations. Query suggestion using hitting time is made on the bipartite undirected graph in which one set of vertices contains queries, the other set of vertices contains URL, and some weight is given to every edge [22].

Among all networks, Complete Networks with N nodes have a minimum mean hitting time $(N - 1)$, which scales linearly with the size of Network [23]. A heterogeneous sparse network can have a low mean hitting time compared to a dense complete network. However, the behavior of heterogeneous sparse networks is similar to the dense complete network, which is instrumental in designing networks, where the search is fast between any pair of nodes [23].

In [24] the authors calculate the Mean Hitting Time of a class of sparse networks using rhombus operation. Results show that the scaling behavior is similar to a complete network. Note that one uses Rhombus operation to maintain the sparsity and growth of the network. Mean Hitting time for a recursive growth tree made up on any arbitrary tree growing due to the implementation of many primitive operation and purpose a series of combinatorial techniques called mapping transformation to exactly determine associative mean hitting time [25].

These related works highlight the importance of hitting time and mean hitting time. Indeed, it is essential to calculate these quantities for a critical q-subdivision network. Different graph operations have been used to grow the network considering small real-world networks. The proposed work uses q-subdivision to expand the network. We rely on two large real-world networks for calculating the mean Hitting time. We also study the variation of mean Hitting time with network growth due to q-subdivision operation. We also explore if the Scale-free property of the real-world network remains after the q-subdivision operation.

3 Methodology

Section 3.1 recalls some important matrices required to calculate the Hitting Time. It is calculated using the eigenvalue and eigenvector of the normalized adjacency matrix and the degree matrix. First, we show how the various network properties, such as the number of nodes, number of edges, etc., of the q-subdivision network relate to the original network. Second, we express the Hitting Time between two nodes of $S_q(G)$ in terms of Hitting Time between two nodes of G. Finally, we derive the equation of the mean Heating Time using the node-pair Hitting Time.

3.1 Network Matrices Notation

Let, G be a connected network with m number of edges and n number of nodes, with $E(G) = \{e_1, e_2, e_3, \ldots, e_m\}$ for edges and $V(G) = \{1, 2, 3, \ldots, n\}$ for nodes. The adjacency matrix of the given network G is denoted by A. It has entry $A(i, j) = 1$ if node i is connected via an edge e in E(G), otherwise $A(i, j) = 0$. Let $\tau(i)$ represent the node's set in network G which are neighbours of node i. node i's degree is represented as $d_i = |\tau(i)| = \sum_{j \in V} A(i, j)$. d_i becomes the $D(i, i)$ of degree matrix D of G and rest entries are zero. The network G incident matrix B is a $n \times m$ matrix with entry $B(i, j)$ equal to 1 if the e_j edge is incident on node i, else $B(i, j)$ equals 0.

Lemma 1 *Suppose G be a n-node simple network with one connected component. If G is bipartite, its incident matrix has rank $rank(B) = n - 1$, otherwise $rank(B) = n$.*

3.2 Random Walk Model on Networks

A network G can be an unbiased discrete-time random walk [26] model on G. If node i has degree d_i, a walker jumps to one of its neighbor nodes with equal probability $1/d_i$, and the walker jumps to a node other than the neighbor node with equal probability $0/di$ or 0. In other words, a walker's probability of moving from the initial node i to the final node j is $A(i, j)/d_i$. The transition probability matrix $T = D^{-1}A$ characterizes a random walker on G can move as it can move in Markov chain [27]. $A(i, j)/d_i$ is the entry in $T(i, j)$.

P, the normalised adjacency matrix of G is defined as $P = D^{1/2}TD^{-1/2} = D^{-1/2}AD^{-1/2}$, and T, the transition probability matrix is similar to it. The entry $P = D^{1/2}TD^{-1/2} = D^{-1/2}AD^{-1/2}$ indicates that P is a symmetric matrix.

Lemma 2 *Suppose G be a non-bipartite network with* 1 *connected component having m edges and n nodes, and eigenvalues of its normalised adjacency matrix P be* $1 = \lambda_1 > \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_n \geq -1$. $\lambda_n = -1$, *if and only if G is bipartite.*

Let the normalized adjacency matrix P has eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ and orthonormal eigenvectors are $v_1, v_2, \ldots, v_n$ corresponding to above eigenvalues, in which $v_i = (v_{i1}, v_{i2}, \ldots, v_{in})^T$. Then

$$v_1 = (\sqrt{d_1/2m}, \sqrt{d_2/2m}, \ldots, \sqrt{d_n/2m}) \tag{1}$$

and

$$\sum_{k=1}^{n} v_{ik} v_{jk} = \sum_{k=1}^{n} v_{ki} v_{kj} = \begin{cases} 1, if\ i = j; \\ 0, otherwise. \end{cases} \tag{2}$$

The Hitting Time [18] is a fundamental quantity in random walks [26]. The first passage time T_{ij} from source node i to final node j is the expected number of steps or jumps a traveler takes to reach the destination node j starting from the source node i. One can use Hitting Times to define or calculate various useful network G quantities. For example, $K(G)$, The Kemeneny's constant is defined as the predicted number of moves a traveler takes, starting at source node i and ending at a randomly chosen destination node from random walks' stationary distribution on G [28].

Theorem 1 *The first-passage time T_{ij} from the initial node i to the final node j of random walks model on a connected network G is given by:*

$$T_{ij} = 2m \sum_{k=2}^{n} \frac{1}{1 - \lambda_k} \left(\frac{v_{kj}^2}{d_j} - \frac{v_{ki} v_{kj}}{\sqrt{d_i d_j}} \right). \tag{3}$$

where the λ_k is kth eigenvalue of normalized adjacency matrix as described in Lemma 2 and v_{ki} or v_{kj} is ith or jth value in the eigenvector v_k corresponding to the eigenvalue λ_k.

3.3 Matrices Notation of Q-subdivision Network

The diagonal degree matrix, the adjacency matrix, the normalized adjacency matrix, the number of edges, and the number of nodes of the $S_q(G)$ are denoted by $\hat{D}$, $\hat{A}$, $\hat{P}$, $\hat{m}$, $\hat{n}$ respectively.

Definition 1 [18] Consider G, which is a connected network. The q-subdivision network of G is represented by $S_q(G)$. To generate $S_q(G)$ from G we have to replace each edge uv with q 2-length disjoint paths: $ux_1v, ux_2v, \ldots ux_qv$. Furthermore, the paths for the various edges are node disjoint.

$\hat{Z}$ is the quantity associated with $S_q(G)$ and Z is the corresponding quantity in G. There are $\hat{m} = 2mq$ edges and $\hat{n} = n + mq$ nodes in $S_q(G)$. The node set $\hat{V} := V \cup V'$ in which V is the old node's set while V' is new node's set comes in $S_q(G)$ [18].

Now, the adjacency matrix $\hat{A}$ [18] of $S_q(G)$ is,

$$\hat{A} = \begin{pmatrix} 0 & B & \cdots & B \\ B^T & 0 & \ldots & 0 \\ . & . & \cdots & . \\ . & . & \cdots & . \\ . & . & \cdots & . \\ B^T & 0 & \ldots & 0 \end{pmatrix}$$

The diagonal degree matrix $\hat{D}$ [18] is defined in terms of G is,

$$\hat{D} = diag\{qD, \underbrace{2I_m, \ldots, 2I_m}_{q}\}$$

The normalized adjacency matrix $\hat{P}$ [18] of $S_q(G)$ is,

$$\hat{P} = \begin{pmatrix} 0 & D^{1/2}B & \ldots & D^{1/2}B \\ D^{1/2}B & 0 & \ldots & 0 \\ . & . & \cdots & . \\ . & . & \cdots & . \\ . & . & \cdots & . \\ D^{1/2}B & 0 & \ldots & 0 \end{pmatrix}$$

3.4 Hitting Time Calculation for Random Walks on Q-subdivision Network

The expression for Hitting time between two nodes in $S_q(G)$ in terms of Hitting time between two vertices in G is derived as follows [18].

Theorem 2 *Let G be a connected network with m edges and n vertices. The q-subdivision network of G with $\hat{V} = V \cap V'$ is $S_q(G)$. Then*

1. *if $i, j \in V$, then $\hat{T_{ij}} = 4T_{ij}$*
2. *if $i \in V', j \in V, \hat{\tau}(i) = \{x, y\}$, then $\hat{T_{ij}} = 1 + 2(T_{xj} + T_{yj})$ $\hat{T_{ji}} = 2mq - 1 + 2(T_{jx} + T_{jy} - (T_{yx} + T_{xy})$*
3. *if $i, j \in V', \hat{\tau}(i) = \{x, y\}, \hat{\tau}(j) = \{z, w\}$, then $\hat{T_{ij}} = 2mq + T_{xz} + T_{yz} + T_{xw} + T_{yw} - (T_{zw} + T_{wz})$*

Here V is a set of old vertices that are present in G, and V' represents the new vertex's set in which new vertices come after the q-subdivision operation on G. $\hat{\tau}(i)$ is the set of neighbour node of node i in $S_q(G)$.

4 Mean Hitting Time for Random Walks on Q-subdivision Network

To our knowledge, the Mean Hitting Time of q-subdivision network of a network G with n nodes has not been derived earlier. Here we give its expression. We aim to investigate how the mean hitting time varies with the q. q is a parameter whose value defines the network's size after applying the q-subdivision operation. The higher the q-value, the larger the size of the network. In the Eq. 4, the Mean Hitting Time of network G is taken from Definition 2. Equation 5 is the expression for the Mean Hitting Time of the q-subdivision network of the network G derived using Eq. 4. Equation 6 rewrite the numerator term of Eq. 5 in terms of the quantities of the actual network G and q.

Definition 2 Mean first passage Time $\bar{T}$ For Random walks modeled on Network of n nodes is the mean of all node pair Hitting Time T_{ij} where $1 \leq i \leq n$ and $1 \leq j \leq n$.

$$\bar{T} = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} T_{ij} \tag{4}$$

The Mean Hitting Time For Random walks on $S_q(G)$ network is $\hat{\bar{T}}$ is given by

$$\hat{\bar{T}} = \frac{1}{(n+mq)^2} \sum_{i=1}^{n+mq} \sum_{j=1}^{n+mq} \hat{T}_{ij} \tag{5}$$

where we can write

$$\begin{aligned} \sum_{i=1}^{n+mq} \sum_{j=1}^{n+mq} \hat{T}_{ij} = & \underbrace{\sum_{i=1}^{n} \sum_{j=1}^{n} 4T_{ij}}_{i,j \in V} + \underbrace{\sum_{i=n+1}^{n+mq} \sum_{j=1}^{n} 1 + 2(T_{xj} + T_{yj})}_{i \in V', j \in V and \tau(i) = \{x,y\}} \\ & + \underbrace{\sum_{i=1}^{n} \sum_{j=n+1}^{n+mq} 2mq - 1 + 2(T_{ix} + T_{iy}) - (T_{yx} + T_{xy})}_{i \in V, j \in V' and \tau(j) = \{x,y\}} \\ & + \underbrace{\sum_{i=n+1}^{n+mq} \sum_{j=n+1}^{n+mq} 2mq + T_{xz} + T_{yz} + T_{xw} + T_{yw} - (T_{zw} + T_{wz})}_{i \in V', j \in V' and \tau(i) = \{x,y\}, \tau(j) = \{z,w\}} \end{aligned} \tag{6}$$

Now in Eq. 6, one value is constant where $i, j \in V$ otherwise, all other values are increasing with an increase in the q-values more rapidly than the value $(n+mq)^2$ in Eq. 4 increases. So with the increasing value of q, in Eq. 4, the increment in the

Table 1 Dataset properties

S. No.	Name of data set	Nodes	Edges
Data set 1	Ego-Facebook	4039	88,234
Data set 2	Feather-lastfm-social	7624	27,806

numerator is higher than the denominator. Hence the average value increase with the growing value of q.

5 Results and Analysis

It is essential to calculate the mean hitting time in real-world networks. Therefore, we consider two real-world network data sets for calculating the mean hitting time with the parameters mentioned in Table 1. We use these data sets because they cover the typical topological properties of numerous real-world networks. The first one, the ego-Facebook network, is a very dense network as compared to the Last Fm user Network. Note that Mean Hitting Time calculation is faster in sparse networks compared to dense networks.

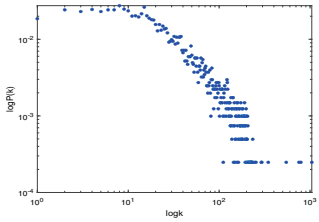
5.1 Scale-Free Property of q-subdivision Network

A ubiquitous property of most real-world networks is that there are very few nodes with very high degrees and many with low degrees. One can think of an Instagram network where celebrities like an actor, cricket player, or some famous people have many followers while regular users have few followers. These few nodes with high degrees are called influencers in social networks and hubs in technological networks. The presence of huge hubs distinguishes scale-free networks from other types of networks. A network having power-law degree distribution is said to be scale-free. We may simply write the degree distribution for an undirected network as follows:

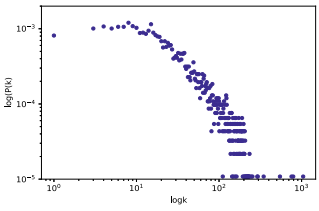
$$P(k) \propto k^{-\gamma}, \tag{7}$$

where, γ denotes an exponent. As the degree k increases, this form of $P(k)$ decays slowly, increasing the chances of finding a node with a very high degree.

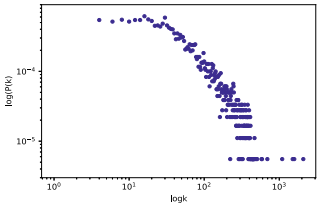
In Fig. 1a, degree distribution is plotted in log-log scale experimentally to check whether the given network is scale-free. k represents the degree, and $P(k)$ represents the frequency of nodes of degree k. There is a larger number of nodes having a low degree. As we move right or increase the degree value, the frequency decreases. So few vertices have a high degree. The long tail observed in Fig. 1a shows that the ego-Facebook network is scale-free. Figure 1b, c are the degree-distribution



(a) Degree distribution of ego-Facebook Network



(b)$S_1(G)$ degree distribution of ego-Facebook network



(c) $S_2(G)$ degree distribution of ego-Facebook network

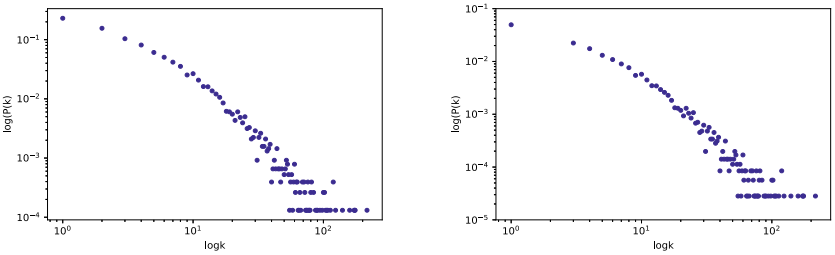
Fig. 1 Degree distribution of q-subdivision network of ego-Facebook network with $q = 0, 1, 2$

of 1-subdivision network and 2-subdivision network respectively. Here the degree-distribution curve in the network is shifted downward and right because one adds new nodes of degree 2, and the degree of old nodes increases. The degree distribution curve in Fig. 1b, c show the same behavior as shown by the degree distribution curve of the ego-Facebook Network. We observe similar long tails in the three figures. Therefore, we can expect that the q-subdivision network of a scale-free network is scale-free. The goodness of fit test results for the power-law distribution confirms this intuition.

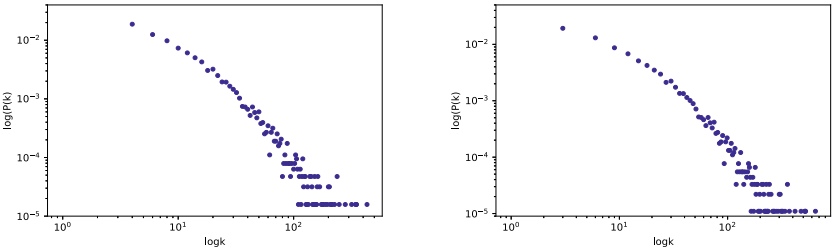
Figure 2a illustrates the degree distribution in log-log scale where k represents the degree and $P(k)$ represents the fraction of nodes of degree k. There is a large number of nodes with a low degree. In contrast, few vertices have a high degree. The long tail in Fig. 2a suggests that the ego-Facebook network is scale-free. Figure 2b–d are the degree-distribution of 1-subdivision network, 2-subdivision network and 3-subdivision network respectively. Here the degree-distribution curve in the network is shifted downward and right because one adds many new nodes of degree 2, and old nodes' degree increases. The degree distribution curve in Fig. 2b–d exhibit the same behavior as the degree distribution of the ego-Facebook Network. One notices the same long tail in the three figures. Goodness-of-fit test with the power-law confirms these results: the q-subdivision network of a scale-free network is scale-free.

5.2 *Variation of Mean Hitting Time of Q-subdivision Network with q*

From Eqs. 5 and 6, one can conclude that in calculating the Mean Hitting Time, the denominator in the Eq. 5 is directly proportional to q^2, and the numerator is propor-



(a) Degree distribution of Last Fm user Network G (b)$S_1(G)$ degree distribution



(c) $S_2(G)$ degree distribution (d) $S_3(G)$ degree distribution

Fig. 2 Degree distribution of q-subdivision network of last Fm user network with $q = 0, 1, 2$

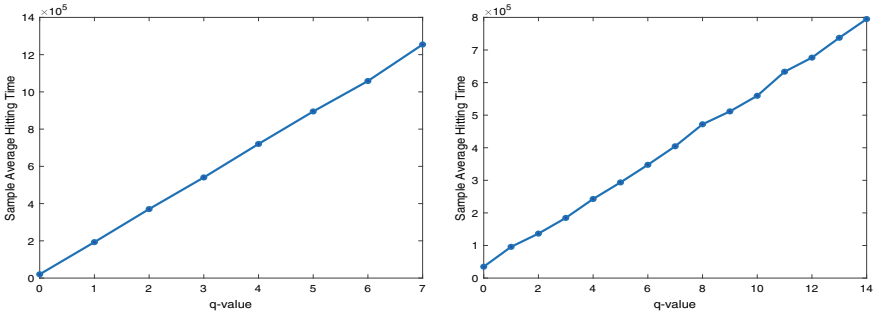


Fig. 3 Mean hitting time of q-subdivision network of network with $q = 0, 1, 2, \ldots$. **a** sample mean Hitting time vs q in ego-Facebook network **b** $S_1(G)$ sample mean hitting time versus q in last Fm user networ

tional to q^3. Hence the mean Hitting Time is proportional to q. The mean Hitting time equation takes the form $\hat{\hat{T}} = c_0 + c_1 q$. So the Hitting Time varies linearly with q. It validates the result obtained from the experiment

As discussed in Sect. 5, Eq. 5, the numerator increases more rapidly than the denominator with an increasing q-value. Hence, the Mean Hitting Time increases with q. The variation of Mean Hitting Time is shown in Fig. 3a, b. In Fig. 3a, we use the ego-Facebook Network with 4039 nodes and 88, 234 edges. We pick a random

sample of 200 pair of nodes and calculate the average Hitting time for that sample. A similar process is repeated for the q-subdivision network. We plot the network Mean Hitting Time versus q-value. As one can see, the mean Hitting time increases linearly with q. Figure 3b shows the variation of Mean Hitting time in the Last Fm user Network with 7624 nodes and 27, 806 edges. The evolution of the Mean Hitting Time versus q exhibits the same behavior as the ego-Facebook Network.

6 Conclusions and Future Work

The Mean Hitting Time varies linearly with q in the q-subdivision network. Real-world networks confirm this behavior. Indeed, Experiments show that the Mean Hitting Time also scales linearly with the network size. To calculate the Mean Hitting Time, one computes the node-pair Hitting Time for the q-subdivision graph. Then, we perform the mean of all possible node-pair Hitting Times. We use the eigenvalues and their corresponding eigenvectors of the normalized adjacency matrix to calculate the node-pair Hitting Time. The main limitation of this work is that the time complexity to calculate the mean Hitting time is very large, about $O(n^3)$ where n is the number of nodes in the network. A second limitation is linked to the calculation of the mean Hitting time of the q-subdivision network. Indeed, we need to make the q-subdivision network to find the neighbor set of new nodes created during the operation. Future work needs to address the time complexity of calculating node-pair hitting time. One solution consists in removing the operation of q-subdivision and writing the equation of Mean Hitting Time in terms of m edges of the network, q value in q-subdivision operation, n nodes of the network, ith and jth node of the network. Further, the work can be done on applying Hitting Time like in the implementation of the EWMA control chart and checking coronavirus alert levels for symmetric COVID-19 cases system [29]. First Passage time has applications in optimal stopping time and bond pricing or cost. Another direction of research is to consider directed networks.

References

1. Rajeh, S., Savonnet, M., Leclercq, E., Cherifi, Hocine: Interplay between hierarchy and centrality in complex networks. IEEE Access **8**, 129717–129742 (2020)
2. Rajeh, S., Savonnet, M., Leclercq, E., Cherifi, Hocine: Characterizing the interactions between classical and community-aware centrality measures in complex networks. Sci. Rep. **11**(1), 1–15 (2021)
3. Ibnoulouafi, A., El Haziti, M., Cherifi, H.: M-centrality: identifying key nodes based on global position and local degree variation. J. Stat. Mech. Theory Exp. **2018**(7), 073407 (2018)
4. Chakraborty, D., Singh, A., Cherifi, H.: Immunization strategies based on the overlapping nodes in networks with community structure. In: International Conference on Computational Social Networks, pp. 62–73. Springer, Cham (2016)

5. Kumar, M., Singh, A., Cherifi, H.: An efficient immunization strategy using overlapping nodes and its neighborhoods. In: Companion Proceedings of the The Web Conference, pp. 1269–1275 (2018)
6. Fasino, D., Tonetto, A., Tudisco, F.: Hitting times for second-order random walks (2021)
7. Mboup, D.D., Cherif, D., Cherifi, H.: Temporal networks based on human mobility models: a comparative analysis with real-world networks. IEEE ACCESS **10**, 5912 (2022)
8. Messadi, M., Cherifi, H., Bessaid, A.: Segmentation and abcd rule extraction for skin tumors classification (2021). arXiv:2106.04372
9. Lasfar, A., Mouline, S., Aboutajdine, D., Cherifi, H.: Content-based retrieval in fractal coded image databases. In: Proceedings 15th International Conference on Pattern Recognition. ICPR-2000, vol. 1, pp. 1031–1034. IEEE (2000)
10. Wong, Felix Ming Fai., Liu, Zhenming, Chiang, Mung: On the efficiency of social recommender networks. IEEE/ACM Trans. Netw. **24**(4), 2512–2524 (2015)
11. Drif, A., Zerrad, H.E., Cherifi, H.: Ensemble variational autoencoders for recommendations Ensvae. IEEE Access **8**, 188335–188351 (2020)
12. Hamidi, M., Chetouani, A., El Haziti, M., El Hassouni, M., Cherifi, H.: Blind robust 3d mesh watermarking based on mesh saliency and wavelet transform for copyright protection. Information **10**(2), 67 (2019)
13. Pastrana-Vidal, R.R., Gicquel, J.-C., Colomes, C., Cherifi, H.: Frame dropping effects on user quality perception. In: Proceedings 5th International WIAMIS (2004)
14. Rosvall, M., Bergstrom, Carl T.: Maps of random walks on complex networks reveal community structure. Proc. Nat. Acad. Sci. **105**(4), 1118–1123 (2008)
15. Cherifi, H., Palla, G., Szymanski, B.K., Xiaoyan, L.: On community structure in complex networks: challenges and opportunities. Appl. Netw. Sci. **4**(1), 117 (2019)
16. Guimerà, R., Díaz-Guilera, A., Vega-Redondo, F., Cabrales, A., Arenas, A.: Optimal network topologies for local search with congestion. Phys. Rev. Lett. **89**(24), 248701 (2002)
17. Feng, M., Hong, Qu., Yi, Z.: Highest degree likelihood search algorithm using a state transition matrix for complex networks. IEEE Trans. Circuits Syst. I Regular Papers **61**(10), 2941–2950 (2014)
18. Zeng, Y., Zhang, Zhongzhi: Spectra, hitting times and resistance distances of q-subdivision graphs. Comput. J. **64**(1), 76–92 (2021)
19. Fiedorowicz, A., Hałuszczak, M.: Acyclic chromatic indices of fully subdivided graphs. Inf. Process. Lett. **112**(13), 557–561 (2012)
20. Zhang, Z.-Z., Zhou, S.-G., Zou, T.: Self-similarity, small-world, scale-free scaling, disassortativity, and robustness in hierarchical lattices. Eur. Phys. J. B **56**(3), 259–271 (2007)
21. Zeng, Y., Zhang, Z.: Hitting times and resistance distances of q-triangulation graphs: accurate results and applications (2018). arXiv:1808.01025
22. Mei, Q., Zhou, D., Church, K.: Query suggestion using hitting time. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, pp. 469–478 (2008)
23. Sheng, Y., Zhang, Z.: Low-mean hitting time for random walks on heterogeneous networks. IEEE Trans. Inf. Theory **65**(11), 6898–6910 (2019)
24. Jing, S., Wang, X., Yao, B.: Mean hitting time for random walks on a class of sparse networks. Entropy **24**(1), 34 (2021)
25. Ma, F., Wang, P.: Mean hitting time on recursive growth tree network (2021). arXiv:2112.04727
26. Xia, F., Liu, J., Nie, H., Yonghao, F., Wan, L., Kong, X.: Random walks: a review of algorithms and applications. IEEE Trans. Emerg. Topics Comput. Intell. **4**(2), 95–107 (2019)
27. Brooks, S., Gelman, A., Jones, G., Meng, X.-L.: Handbook of Markov chain Monte Carlo. CRC press (2011)
28. Hunter, J.J.: The role of Kemeny's constant in properties of Markov chains. Commun. Stat. Theory Methods **43**(7), 1309–1321 (2014)
29. Yupaporn, A., Rapin, S., et al.: Ewma control chart based on its first hitting time and coronavirus alert levels for monitoring symmetric covid-19 cases. Asian Pacific J. Tropical Med. **14**(8), 364 (2021)

Delta Density: Comparison of Different Sized Networks Irrespective of Their Size

Jakub Plesnik, Kristyna Kubikova, and Milos Kudelka

Abstract Two typical characteristics of networks are average degree and density. Both characteristics are related, but using the second one does not provide easily interpretable information when analyzing differently sized networks. This paper deals with the measurement of network density with the possibility of comparing networks of different sizes. We point out the problems of the classical approach and, in response, introduce a new measure called Δ-density. The theoretical background of Δ-density is accompanied by a practical application example. We use five real networks with temporal information in the experiments to analyze the evolution of Δ-density.

Keywords Network density · Average degree · Delta density

1 Introduction

The network analysis quite often leads to non-trivial tasks of network comparison. Techniques for such analysis vary from complex methods, such as network alignment, to easily comparing global properties and summary statistics, such as network density, degree distribution, transitivity, average shortest path length, and others. When research focuses on network structure and primarily its connectivity, only two properties are in play: average degree and network density.

An abundance of papers is focused on the relationship between the size of the network and its density. In their work, Laurienti et al. describe a universal relationship between network size and connection density across various types of systems and identify that as a fractal size-density relationship for self-organized networks [1]. Leskovec et al. studied the temporal evolution of complex networks and described

J. Plesnik (✉) · K. Kubikova · M. Kudelka
Department of Computer Science, Faculty of Electrical Engineering and Computer Science, VSB-Technical University of Ostrava, 17. listopadu 2172/15, 708 00 Ostrava-Poruba, Czech Republic
e-mail: jakub.plesnik@vsb.cz

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_29

Densification Power Law which refers to increased density with network growth [2]. Furthermore, Neli Blagus et al. study the relationship between the size and density of complex real-world networks and their existence in their self-similar scale [3].

From the point of network clustering, Yin et al., with their local closure coefficient, which is defined as the fraction of length-2 paths emanating from the head node that induce a triangle, show that even a small change in the definition can provide very different results from the traditional clustering coefficient [4]. It is important to note that we can consider the clustering coefficient as a measure of the node's neighborhood density. Everything described in this paper about Δ-density can also be applied to the clustering coefficient.

The work of van Wijk et al. shows the difficulty of comparing multiple networks of different sizes and states that either applying a fixed N and k_{avg} or comparing networks with different N and k_{avg} will lead to a certain bias [5]. Related to network comparison, Brigham et al. show that size and density strongly interact with all graph-level measures [6].

In this paper, we propose a new measure called Δ-density. Its aim is to enable a straightforward comparison of network density between networks of different sizes. The paper is structured as follows: In the second section, we introduce datasets used in the third section to state the problem that this paper aims to solve. The fourth section defines a measure called Δ-density, which is applied in the following section as part of the experiments. In the final section, the summary of this paper is provided.

2 Datasets

We used five dynamic networks of different sizes and origins for our experiments. In selected networks, communication, collaboration, and social networks are represented. Each dynamic network has been further split into five subsets/snapshots in time. These subsets were created at the point in time when networks reached 20, 40, 60, 80, and 100% of the total number of nodes. In the process of subset generation, we worked with edges as unweighted and undirected.

Linux-kernel. A communication network of the Linux kernel mailing list, in which each node represents a person identified by their email address, and each directed edge represents a reply from one user to another. The dataset contains data from 2006 to 2013.

coauth-DBLP. A database of scientific publications such as conference papers, journal articles, etc. Each node in the network is a publication, and each edge represents a citation of a publication by another publication. The data used for this paper are from 2018 to 2022.

Facebook (WOSN). An undirected network containing friendship data of Facebook users, where nodes represent the users, and a friendship between two users represents the edge. The dataset contains data from the year 2009.

epinions. A trust network from the online social network of a general consumer review site Epinions.com, where edges represent the trust between the users that are represented by nodes.

Enron. An email Communication network consisting of over a million emails sent within the Enron company between the years 1999 and 2003.

3 Motivation: Analysis of Network Density and Average Degree

This paper proposes a solution to a problem concerning the density comparison of different-sized networks. First, we need to look at the classical approach using network density and average degree and show its usage and some of its pitfalls. It is a fact that two networks of different sizes with the same average degree may have significantly different densities, and two differently sized networks with the same density can significantly differ in average degree.

By restating definitions, we can say that network density is a measure of how many edges between nodes exist compared to how many edges between nodes are possible. The average degree is the average number of edges per node in the network. By the addition of a new edge, we will always increase both average degree and density, but by the addition of a node with a degree significantly lower than the number of nodes in the network, we may still increase the average degree, but we will lower its density.

In Fig. 1a, we can see how the density of the network changes with network growth on the example of five real-world datasets. The density of all networks, except Facebook, is decreasing even though we know from Table 1 that networks are growing in both the number of nodes and edges.

By closer look, we can see one of the issues with the usage of network density for comparison between networks. Networks Epinions and Enron seem to have a similar value of network density, which leads to a question: *Are these two networks*

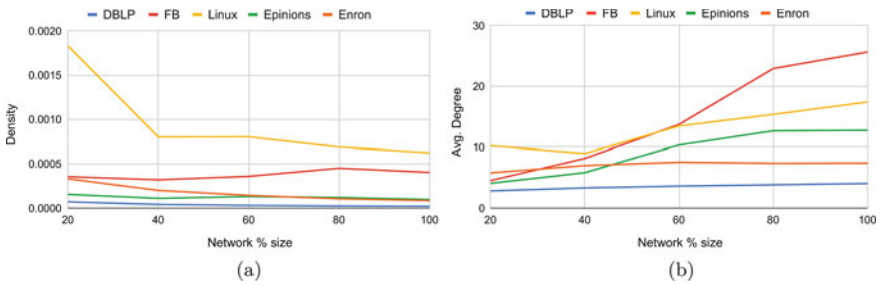


Fig. 1 Evolution of network properties: **a** evolution of network density, **b** evolution of the average degree

Table 1 Properties of analysed datasets

Network	Size %	n	m	k_{avg}	Density
coauth DBLP	20	40,000	56,455	2.82	7.06e−5
	40	80,000	132,858	3.32	4.15e−5
	60	120,001	216,953	3.62	3.01e−5
	80	160,000	306,758	3.83	2.40e−5
	100	200,000	404,560	4.05	2.02e−5
Facebook WOSN	20	12,746	28,770	4.51	3.54e−4
	40	25,492	103,851	8.15	3.20e−4
	60	38,238	262,183	13.71	3.59e−4
	80	50,984	584,355	22.92	4.50e−4
	100	63,731	817,090	25.64	4.02e−4
Linux Kernel	20	5585	28,569	10.23	1.83e−3
	40	11,170	49,987	8.95	8.01e−4
	60	16,755	112,775	13.46	8.03e−4
	80	22,340	171,701	15.37	6.88e−4
	100	27,927	242,974	17.4	6.23e−4
epinions	20	26,365	53,698	4.07	1.55e−4
	40	52,730	153,380	5.82	1.10e−4
	60	79,095	409,304	10.35	1.31e−4
	80	105,460	667,989	12.67	1.20e−4
	100	131,828	841,363	12.76	9.68e−5
Enron	20	17,454	50,450	5.78	3.31e−4
	40	34,908	121,879	6.98	2.00e−4
	60	52,362	196,987	7.52	1.44e−4
	80	69,816	256,799	7.36	1.05e−4
	100	87,273	321,918	7.38	8.45e−5

similarly dense? Looking back to network properties, we can see a disproportion in network sizes. Epinions network is much larger with a significantly higher number of edges. Based on the definition, we could say that these two networks have a similar proportion of their connectivity compared to their maximum potential, but their internal density may not be the same, and if compared, we need to keep this in mind.

In Fig. 1b, we can see the average degree for all networks over time. There is an interesting data point at 80%, where networks Linux($k_{avg} = 15.3$) and Epinions($k_{avg} = 12.6$) have values relatively close to each other. That could suggest that those two networks are similarly dense. If we look for other properties in the Table 1 we can again see a discrepancy in network sizes. Thus we can say that both networks have a similar number of connections per node, but the density may be different due to the network size.

As shown in previous examples, using both average degree and network density in network comparison is not a simple task. Furthermore, we may face a contradiction when both properties are considered together. For many real-world networks, it has

been shown that the average degree is increasing as networks grow [2]. Our data supports this trend, but the network density itself is decreasing. The question is: *Can we meaningfully compare the density of networks of different sizes?*

4 Δ-Density

This paper aims to define a new measure that considers both the density and the average degree in such a way that allows the values of this measure to be comparable for networks of different sizes. Let us further denote this measure as Δ-density.

The value of Δ-density is in the [0; 1] interval. One of the most important factors is an average degree k_{avg} for which we can establish value of Δ-density. For example we can assume that for $k_{avg} = 12$ Δ-density$= 0.9$.

The concept of Δ density is based on the assumption that each existing edge has a greater influence on the determination of Δ density than a missing edge between a pair of nodes. If we consider $1, 2, \ldots, m$ of existing edges in the network and gradually count them into Δ-density, then the previously calculated edge has less effect on Δ-density than the later calculated edge. In this context, let us determine a constant $\delta \geq 0$. Then m existing and gradually counted edges will have values for Δ-density $1 + \delta, 1 + 2 * \delta, \ldots, 1 + m * \delta$. Then the Δ-density for a network with n nodes and m edges is defined as:

$$\Delta(\delta, n, m) = \frac{A(m, \delta)}{(A(m, \delta) + M(n) - m)} \tag{1}$$

where $M(n)$ is the maximum possible number of edges in a network with n nodes and thus $M(n) = \frac{n*(n-1)}{2}$, and $A(m, \delta) = \frac{m*(2+\delta*(1+m))}{2}$ is the sum of the arithmetic series.

As mentioned above, if we add to the network a node with the number of edges corresponding to the average degree, its density decreases. Informally speaking, δ compensates for this decrease in such a way that the lower δ is, the less compensation for the decrease in density it provides, and vice versa.

In determining the δ we need to take into consideration two characteristics, the network size and the relationship between the average degree $k = \frac{2*m}{n}$ and the corresponding expected value of Δ-density Δ_{exp}. Value of Δ_{exp} could be, for example, set to 0.9; this is ideal if we want to compute δ based on a reference network that we consider dense. By doing so, we will get δ which we can use to compute Δ-density. For a reference network, Δ-density will result equal to Δ_{exp}, but we can use the same value of δ to compute Δ-density of other networks, which then will be comparable to each other.

If we modify the formula for calculating the Δ-density we can define a δ function. Calculation of the δ based on the network size n, average degree k_{avg} and the expected value of Δ-density Δ_{exp} is as follows:

Table 2 δ calculated for two values of expected Δ-density and average degree

n	$k_{avg} = 12, \Delta_{exp} = 0.9$	$k_{avg} = 5, \Delta_{exp} = 0.5$
100	0.214	0.142
1000	0.246	0.158
10000	0.250	0.160
100000	0.250	0.160

Table 3 Δ-density calculated with $\delta = 0.16$ for artificial networks of different sizes and average degree

n	$k_{avg} = 2$	$k_{avg} = 5$	$k_{avg} = 9$	$k_{avg} = 14$
100	0.1577	0.5286	0.7876	0.9039
1000	0.1399	0.5029	0.7665	0.8858
10000	0.1381	0.5003	0.7644	0.8870
100000	0.1379	0.5000	0.7642	0.8869
1000000	0.1379	0.5000	0.7642	0.8869

$$\delta(n, k_{avg}, \Delta_{exp}) = \frac{4 * (d * n - d - k_{avg})}{((1 - d) * k_{avg} * (k_{avg} * n + 2))}. \quad (2)$$

In the Table 2 we can see that the computed δ value is stable for artificial networks having thousands of nodes; however, it does not differ much for smaller networks. Further, in the Table 3 we can see a demonstration of the Δ-density values for artificial networks with different average degree and selected $\delta = 0.160$. As shown in Table 2, the value 0.160 corresponds to the expected $Delta$-density of 0.5 with an average degree of 5. We can notice that the values are stable regardless of the network size.

It should also be noted that for $\delta = 0$ the Δ-density becomes ordinary network density. However, even for small δ values, the Δ-density value stabilizes for networks with a size of more than a thousand nodes, as we can see in the example above. That means that by choosing the suitable value of δ, we could create a setting that allows us to calculate Δ-density with stable results regardless of the network size and can be used to compare such networks.

5 Experiments

For the application of Δ-density to compare the density of networks of different sizes, our first step is to choose a suitable value of parameter δ, which will be used for all analyzed networks. The value must be greater than 0 without an upper limit, but in most cases, values in the range of 0–1 are used. Our proposed solution for finding the ideal value of δ is to choose a reference network that we consider dense for

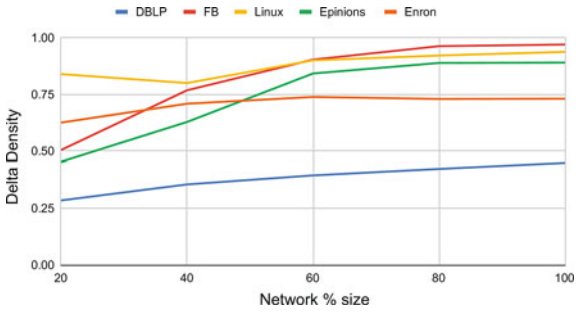


Fig. 2 Evolution of Δ-density as networks grows for parameter $\delta = 0.2$

the analysis. The resulting Δ-density for the given network will create our reference point, and other networks can be compared relatively to this point. Alternatively, since δ is computed only from an average degree and size of the network, we could also use any n and k_{avg} as we see fit.

In our experiments we chose Linux(size 60%) as a reference network. Network's n and k_{avg} from Table 1 is used in Eq. 3 to compute δ value. Function $\delta(n, k, d)$ from Eq. 2. The result δ value is 0.2 after approximation.

$$\delta(16, 755, 13.462, 0.9) = 0.198 \approx 0.2 \tag{3}$$

With our δ value, we can now compute Δ-density for all analyzed networks. Using the same δ will enable comparison where our reference network will have $\Delta - density \approx 0.9$, and will be referenced as dense. Furthermore, we propose to establish a threshold value of Δ-density analytically. If the network has a higher Δ-density than the threshold, we also consider such network as dense. For this experiment, we use a threshold value of 0.85.

In Fig. 2 we can see the computed value of Δ-density for all analyzed networks as they grow. The obvious observation is that our reference network(Linux) is dense from its size of 60% and larger. Furthermore, Facebook Δ-density is above the threshold, thus is dense, for its sizes 60, 80, and 100%. Linux and Facebook are both dense after reaching the size of 60%. Network Epinions is for its size of 60% below the threshold; it can be viewed as dense only for its size of 80 and 100%. Both Enron and DBLP have Δ-density under the threshold, so we see them as not dense, but since DBLP has significantly lower values, we can still say that Enron is denser than DBLP.

5.1 Effect of Parameter δ

In Fig. 3 we can see cases of not ideally selected δ values. This may occur when we do not use a reference network for computation of δ but rather try to blind pick. In plot a, with a too small δ value, the result is almost identical to the plot with the average degree. On the other hand, in plot b, the value is too large, resulting in almost

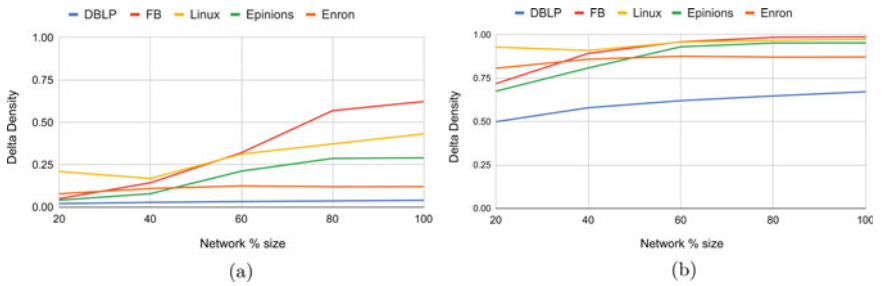


Fig. 3 Δ-density with poorly selected values of parameter δ. For **a** δ is too low, **b** δ is higher than ideal

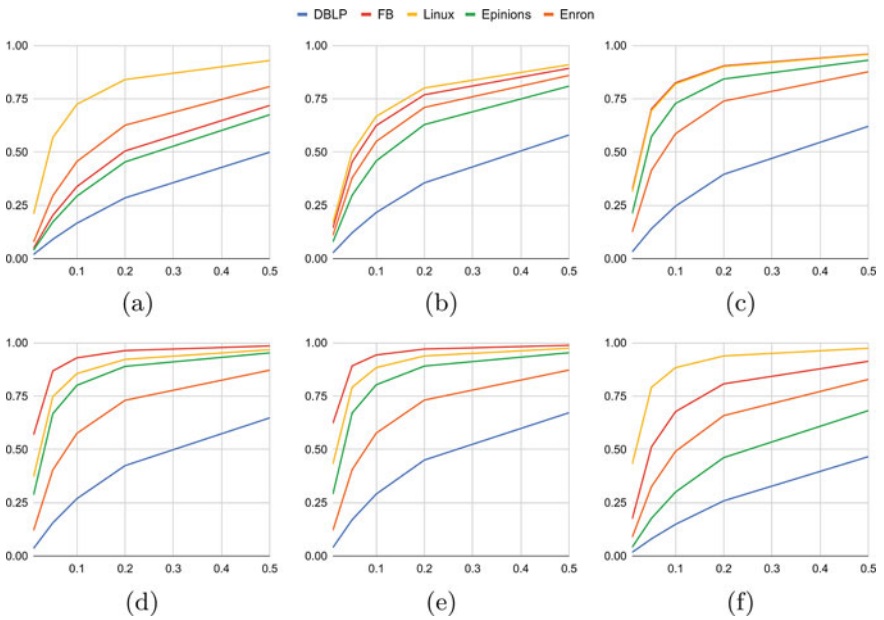


Fig. 4 Relationship between Δ-density (y-axis) and parameter δ (x-axis) presented on multiple sizes of snapshot. Each plot uses following network size **a** = 20%, **b** = 40%, **c** = 60%, **d** = 80%, **e** = 100%. Plot **f** shows snapshots normalizes on the size of the smallest of analyzed datasets

all networks converging towards 1, thus making the result considered dense. In both cases, the analytical value is significantly lowered than in Fig. 2.

In Fig. 4 we can see multiple plots (a,b,c,d,e), each for different network size. These plots show how Δ-density grows with a higher value of parameter δ. As we can see on c, d and e with higher δ values of Δ-density slowly converge towards 1. Networks that are significantly less dense would require a much higher value of parameter δ to also converge towards 1.

For the experiment with Δ-density on networks of the same size, we have created snapshots for each network equal in the number of nodes to the smallest analyzed network. In Fig. 4 *f* we can see Δ-density for a network of the same size for different δ parameter. As we can see overall characteristics of the Δ-density growth remain the same for all networks.

6 Conclusion

In this paper, we have presented a problem of network density comparison between networks of different sizes and introduced a new measure that we call Δ-density. Our proposed solution aims at defining a measure that takes into account both the density and the average degree. Measure definition is supported by application on both artificial and multiple real-world networks.

In our experiments, we showed an application of Δ-density on multiple real-world networks of different sizes and origins. We provide detailed steps on how to use Δ-density and interpreted results. In the same section, we pointed out some of the pitfalls that come with poor parameter selection δ. Furthermore, we have shown a relationship between Δ-density and parameter δ.

Even though Δ-density is novel and has not been tested by many practical applications, we believe that it brings a new point of view by removing bias that can occur when an analysis is based solely on network density or average degree.

Acknowledgements This work is partially supported by SGS, VSB-Technical University of Ostrava, under the grant no. SP2022/77 and Ministry of Health of the Czech Republic under grants no. NU20-06-00269, NU21-06-00370.

References

1. Laurienti, P., Joyce, K., Telesford, Q., Burdette, J., Hayasaka, S.: Universal fractal scaling of self-organized networks. Nat. Precedings **5** (2010)
2. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations (2005)
3. Self-similar scaling of density in complex real-world networks. Phys. A Stat. Mech. Appl. **391**(8), 2794–2802 (2012)
4. Yin, H., Benson, A., Leskovec, J.: The local closure coefficient: a new perspective on network clustering, pp. 303–311 (2019)
5. van Wijk, B., Stam, C., Daffertshofer, A.: Comparing brain networks of different size and connectivity density using graph theory. PloS One **5**, e13701 (2010)
6. Anderson, B., Butts, C., Carley, K., Heinz, H.: The interaction of size and density with graph-level indices Ž . Social Netw. **21**, 239–267 (1999)

Resilence and Robustness of Networks

Robustness of Network Controllability with Respect to Node Removals

Fenghua Wang and Robert Kooij

Abstract Network controllability and its robustness has been widely studied. However, analytical methods to calculate network controllability with respect to node removals are currently lacking. This paper develops methods, based upon generating functions for the in- and out-degree distributions, to approximate the minimum number of driver nodes needed to control directed networks, during random and targeted node removals. By validating the proposed methods on synthetic and real-world networks, we show that our methods work very well in the case of random node removals and reasonably well in the case of targeted node removals, in particular for moderate fractions of attacked nodes.

Keywords Controllability · Complex networks · Node failures · Node attacks

1 Introduction

Network controllability has been investigated for different kinds of networks, like biological networks [1], transportation networks [2] and corruption networks [3]. A network is controllable if the states of nodes can be steered to any expected states in a finite time by imposing external inputs to some of the nodes. Kalman's controllability rank condition is used to judge whether a linear system is controllable or not [4]. However, sometimes we do not know the weighted interactions within the network, which describe the strength with which a node affects other nodes. To overcome the issue, the concept of structural controllability has been proposed [5].

F. Wang (✉)
Delft University of Technology, 2628 CD, Delft, The Netherlands
e-mail: F.Wang-8@tudelft.nl
URL: https://nas.ewi.tudelft.nl/index.php/fenghua-wang

R. Kooij
TNO, Unit ICT, 2595 DA, Den Haag, The Netherlands
e-mail: R.E.Kooij@tudelft.nl
URL: https://nas.ewi.tudelft.nl/index.php/rob-kooij

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_30

The interaction matrix and input matrix of the linear time-invariant system are structural if their elements are independently free parameters or some are fixed zeros. The system is called structurally controllable if it is possible to find values of structural interaction and input matrices to make the system satisfy the usual controllability condition. Besides investigating the necessary and sufficient conditions to make the specific system strong structural controllable [6], another research direction is to find the minimum set of inputs to make the system fully controllable [7]. Liu et al. [8] reduce the structural controllability problem into the optimization problem of finding a set of unmatched nodes in a maximum matching of the network. The nodes where the external input signals are imposed are named driver nodes. The number of unmatched nodes equals the minimum number of driver nodes needed to fully control the network. Note that the results reported in Liu et al. [8] critically depend on the assumption that the direct network has no self-links, i.e. a node's internal state can only be changed upon interaction with neighboring nodes [9]. We will follow this assumption throughout the paper.

Network structural controllability as a generic system property is applied to measure and enhance network robustness. Measuring network robustness is usually done by measuring network performance changes during perturbations imposed upon the network [10]. The widely adopted perturbations in the research of the robustness of network controllability are random node or link removal, which are used as a benchmark compared with other perturbations. Another kind of perturbation deals with targeted attack strategies. For example, attack strategies can relate to network topology features, such as betweenness, degree and closeness. Pu et al. [11] demonstrate that degree-based attacks are more harmful to network controllability compared to random attacks. Lu et al. [12] find that a betweenness-based attack strategy is more harmful than a degree-based attack strategy in most real-world networks. However, Wang et al. [13] find that attacking bridge links, whose removal results in a disconnected network, is an effective way to destroy network controllability. Another kind of targeted attack strategy is based on critical nodes and links. Critical nodes and links are defined through the property that their removal will increase the number of driver nodes [8]. Sun et al. [14] report random attack under the protection of critical links is less efficient than a random link attack, and a targeted attack aiming at critical links is more harmful than a random attack. Lou et al. [15] propose a hierarchical attack removal framework where nodes or links are classified into critical, sub-critical and normal categories. They find that hierarchical attack strategies are more efficient than some metric-based attack strategies such as betweenness- or degree-based strategies in interdependent networks. There is also some research focusing on how to enhance the robustness of network controllability. Giulia et al. [16] show that network controllability is determined by the density of nodes with in-degree and out-degree equal to one or two. Adding links to low degree nodes is beneficial to network controllability. Lou et al. [17] find that multi-loop structures can improve the robustness of network controllability. Zhang et al. [18] investigate different redundant design strategies of interdependent networks. They present that betweenness-based strategy and degree-based strategy for node backup and high degree first strategy for edge backup can optimize robustness of network controllability.

Besides the aforementioned qualitative research on the robustness of network controllability, quantitative research has been conducted. Lu et al. [12] develop the numerical approximations of random node attacks and target node attacks based on degree on Erdös-Rényi (ER) networks. The results fit well when the fraction of nodes is below 20%. Sun et al. [14] explore the closed-form approximation of the number of controllable nodes under random link attacks, targeted attacks and random attacks with protection. Dhiman et al. [19] use machine learning to quantify the minimum fraction of driver nodes under random link attacks and target link attacks, which performs better than the closed-form approximation proposed by Sun et al. [14]. Later, Chen et al. [20] develop analytical approximations for the minimum number of driver nodes during random link removal by using methods based on generating functions.

However, to our knowledge, analytical methods to approximate the network controllability during random and targeted node removal on different kinds of networks are lacking. The framework to calculate the structural controllability of linear systems for directed networks has been proposed by Liu et al. [8]. This paper uses their framework to develop analytical approximations based on degree distributions to calculate the minimum fraction of driver nodes during node removal. We choose two cases for the removal of nodes: random node removal and targeted node removal, based upon the node degrees. In order to validate our methods, we use two types of synthetic networks and four real-world communication networks.

This paper is organized as follows. The second section introduces the networks used in the study for validation. The analytical results for the robustness of network controllability during random node removal are presented in the third section. The fourth section shows the results for the robustness of network controllability for targeted node removal. The final section reports the conclusion and discussion.

2 Network Data

We will validate our theoretical results, which will be derived in the subsequent sections, on two classes of synthetic networks and on a number of real-world networks. In this section, we give details on the used networks.

2.1 Directed Synthetic Networks

We choose two kinds of synthetic networks: Erdös-Rényi (ER) networks and Swarm Signalling networks (SSNs).

We generate a directed ER network on N nodes, by placing a directed link between any pair of nodes, with a given probability p_{ER}. The average number of links for such ER networks satisfies $L = N(N-1)p_{ER}$. In this paper we have used two ER networks, with $N = 50$, $p_{ER} = 0.07$ and $N = 100$, $p_{ER} = 0.04$.

Table 1 Properties of four real-world communication networks

Name	N	L	$< k >$
HinerniaGlobal	55	81	2.95
Syringa	74	74	2.00
Interoute	110	146	2.65
Cogentco	197	243	2.47

The topology for Swarm Signalling Networks (SSNs) that we use was suggested in [21]. The SSN has a regular out-degree, while the in-degree distribution follows a Poisson distribution. To generate SSNs, we need two parameters. One is the number of nodes N, and the other is the out-degree value k. For each node, the node randomly creates k outgoing links to other nodes. In this paper we have used two SSNs, both with $N = 10^4$ and with $k = 2$ and $k = 5$.

2.2 Real-World Networks

The real-world networks used in this study are taken from the Internet Topology Zoo [22], a collection of real-world communication networks. We change those undirected networks into directed networks by using two attributes: source node and target node [14]. The properties of the networks are shown in Table 1, which shows the number of nodes N, the number of links L, and the average total degree $< k >$. The total degree is the sum of the in-degree and the out-degree. Obviously, the average in-degree equals the average out-degree and therefore the average total degree is twice the average in-degree (and hence the average out-degree).

3 Minimum Fraction of Driver Nodes Under Random Node Removals

This section presents how to analytically approximate network controllability in the case of random node removals.

3.1 Analytical Approximation

3.1.1 General Networks

From [8], for directed network $\mathcal{G}(N, L)$ with N nodes and L links, we can determine the minimum number of driver nodes by using generating functions of the in- and

out-degree distributions ($G_{in}(x)$ and $G_{out}(x)$, respectively) and of the excess in- and out-degree distributions ($H_{in}(x)$ and $H_{out}(x)$, respectively). These generating functions are defined as follows:

$$\begin{aligned} G_{in}(x) &= \sum_{k=0}^{\infty} P_{in}(k_{in}) x^{k_{in}}, \; G_{out}(x) = \sum_{k=0}^{\infty} P_{out}(k_{out}) x^{k_{out}}, \\ H_{in}(x) &= \frac{\sum_{k=1}^{\infty} k_{in} P_{in}(k_{in}) x^{k_{in}-1}}{< k_{in} >} = \frac{G'_{in}(x)}{G'_{in}(1)}, \\ H_{out}(x) &= \frac{\sum_{k=1}^{\infty} k_{out} P_{out}(k_{out}) x^{k_{out}-1}}{< k_{out} >} = \frac{G'_{out}(x)}{G'_{out}(1)}, \end{aligned} \tag{1}$$

where k_{in} and k_{out} denote in- and out-degree, respectively, while $P_{in}(\cdot)$ and $P_{out}(\cdot)$ are in- and out-degree probability distribution, respectively. Then the minimum fraction of driver nodes is given by:

$$\begin{aligned} n_d = \frac{1}{2}\{&G_{in}(\omega_2) + G_{in}(1-\omega_1) - 2 + G_{out}(\hat{\omega_2}) + G_{out}(1-\hat{\omega_1}) \\ &+ k[\hat{\omega_1}(1-\omega_2) + \omega_1(1-\hat{\omega_2})]\}, \end{aligned} \tag{2}$$

where ω_1, ω_2, $\hat{\omega}_1$ and $\hat{\omega}_2$ satisfy

$$\omega_1 = H_{out}(\hat{\omega_2}), \omega_2 = 1 - H_{out}(1-\hat{\omega_1}), \hat{\omega_1} = H_{in}(\omega_2), \hat{\omega_2} = 1 - H_{in}(1-\omega_1), \tag{3}$$

and k denotes half of the average degree equal to the average in-degree and the average out-degree, $k = \frac{1}{2} < k > = < k_{in} > = < k_{out} >$.

During the node removal process, the set of driver nodes includes two parts. One is the set containing N_D driver nodes that control the remaining part of the network, and the other set is formed by N_r removed nodes. We assume that each removed node needs to be controlled by an individual driver node. We define the fraction of driver nodes n_D as $n_D = \frac{N_D + N_r}{N}$. After randomly removing a fraction p of nodes in the network, the fraction of driver nodes n_D satisfies

$$n_D = \frac{n_d(1-p)N + pN}{N} = n_d(1-p) + p. \tag{4}$$

Based on the research of Shao et al. [23], the generating function after randomly removing a fraction p nodes corresponds to the original generating function, with the adjusted argument $\bar{x} = p + (1-p)x$. Then the generating functions of in- and out-degree, and the excess in- and out-degree, after randomly removing a fraction p of nodes, are adjusted as follows:

$$\bar{G}_{in}(x) = G_{in}(p + (1-p)x), \quad \bar{G}_{out}(x) = G_{out}(p + (1-p)x),$$
$$\bar{H}_{in}(x) = \frac{\bar{G}'_{in}(x)}{\bar{G}'_{in}(1)}, \quad \bar{H}_{out}(x) = \frac{\bar{G}'_{out}(x)}{\bar{G}'_{out}(1)}. \tag{5}$$

Next, we use Eqs. (2) and (4) to acquire the fraction of minimum number of nodes n_D after randomly removing a fraction p of nodes:

$$\begin{aligned} n_D = \frac{1}{2}(1-p)\{&\bar{G}_{in}(\omega_2) + \bar{G}_{in}(1-\omega_1) - 2 + \bar{G}_{out}(\hat{\omega}_2) + \bar{G}_{out}(1-\hat{\omega}_1) \\ &+ k(1-p)[\hat{\omega}_1(1-\omega_2) + \omega_1(1-\hat{\omega}_2)]\} + p, \end{aligned} \tag{6}$$

where ω_1, ω_2, $\hat{\omega}_1$ and $\hat{\omega}_2$ satisfy

$$\omega_1 = \bar{H}_{out}(\hat{\omega}_2), \quad \omega_2 = 1 - \bar{H}_{out}(1-\hat{\omega}_1), \quad \hat{\omega}_1 = \bar{H}_{in}(\omega_2), \quad \hat{\omega}_2 = 1 - \bar{H}_{in}(1-\omega_1), \tag{7}$$

and k is half of the average degree equal to the average in-degree and the average out-degree, $k = \frac{1}{2} <k> = <k_{in}> = <k_{out}>$.

3.1.2 ER Networks

Both the in-degree distribution $P_{in}(k_{in})$ and the out-degree distribution $P_{out}(k_{out})$ of ER networks follow a Poisson distribution with average degree k [20]. Therefore, the generating functions of in-degree and out-degree are as follows,

$$G_{in}(x) = e^{-k(-x+1)}, G_{out}(x) = e^{-k(-x+1)}. \tag{8}$$

The minimum fraction of driver nodes n_D after a fraction p of nodes is randomly removed in the ER networks can be obtained through Eq. (6) as

$$n_D = p + p\omega_2 - \omega_2 + [1 - p + k(1-p)^2(1-\omega_2)]e^{k(1-p)(\omega_2-1)} \tag{9}$$

where ω_2 satisfies $1 - \omega_2 - e^{-k(1-p)e^{-k(1-p)(1-\omega_2)}} = 0$.

3.1.3 SSNs

In a SSN with the number of nodes N and average in-degree and out-degree equal to k, the in-degree distribution resembles a Poisson distribution with mean value k and the out-degree distribution follows a Dirac delta function. Then the generating functions of in-degree and out-degree distribution can be denoted as follows,

$$G_{in}(x) = e^{-k(-x+1)}, G_{out}(x) = x^k. \tag{10}$$

Based on Eq. (6), the minimum fraction of driver nodes n_D after randomly removing a fraction p of nodes can be calculated by

$$n_D = p + p\omega_2 - \omega_2 + [1 - p + (k-1)(1-p)^2(1-\omega_2)]e^{k(1-p)(\omega_2-1)} \tag{11}$$

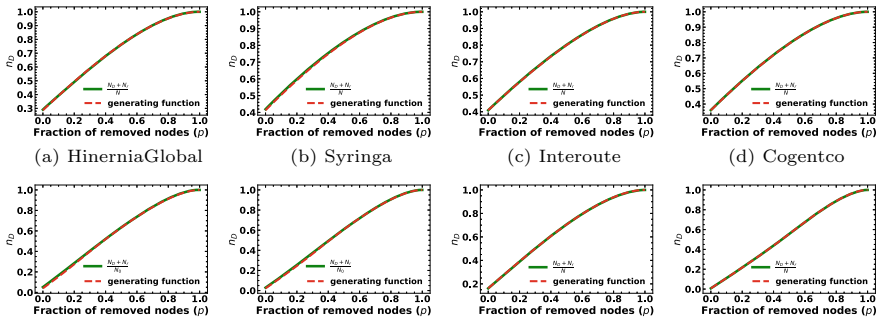
where ω_2 satisfies $1 - \omega_2 - [p + (1-p)(1 - e^{-k(1-p)(1-\omega_2)})]^{k-1} = 0$.

Note that for the real-world networks, the generating functions for the in- and out-degree distributions, can simply be obtained from the histograms of these distributions. We use the relative frequency of degree as the corresponding probability in generating functions.

3.2 Validation

We ran simulations on the various networks described in Sect. 2. Specifically, for each communication network, we do 10,000 realizations, and in each realization, we remove a node randomly at each step until all nodes have been removed. For each kind of synthetic network, we heuristically choose two pairs of parameters: ER networks with $N = 50$, $p = 0.07$ and $N = 100$, $p = 0.04$ and SSNs with $N = 10^4$, $k = 2$ and $N = 10^4$, $k = 5$. In each realization, we generate a synthetic network, given its parameters, and remove nodes one by one randomly. After removing a node, we recalculate the minimum fraction of driver nodes using the maximum matching algorithm. However, as our SSNs have a large number of nodes, we remove 1% of the original number of nodes at each step. We do 10,000 realizations for each synthetic network as well. Then we obtain the average minimum fraction of driver nodes. The green lines in Fig. 1 show the simulation results.

Since we know each network's in-degree and out-degree distributions, we can compute the minimum fraction of driver nodes of a network according to the equations mentioned above for the minimum fraction n_D. The results obtained using the



(a) HinerniaGlobal (b) Syringa (c) Interoute (d) Cogentco

(e) ER(50,0.07) (f) ER(100,0.04) (g) SSN(10^4, 2) (h) SSN(10^4, 5)

Fig. 1 The minimum fraction of driver nodes n_D during random node removal in different kinds of networks. The green lines are calculated by the maximum matching algorithm over 10,000 realizations. The red dashed lines are obtained by the analytical methods

generating functions of the degree distributions are depicted as red dashed lines in Fig. 1.

The results are shown in Fig. 1. As the predicted values in the red lines and the simulated values virtually overlap, we conclude that the analytical approximations for network controllability in the case of random node removals are very accurate. The reason for this is that, after removing a fraction p of the nodes at random, we still have expressions for the generating functions of the in- and out-degree distributions, see Eq. (5).

4 Minimum Fraction of Number of Driver Nodes Under Targeted Node Removals

Degree centrality has been deeply investigated in the context of network robustness [24]. Nodes with a high degree have a large influence on network functioning and might be assumed to have a high probability of being attacked. We will explore how to analytically approximate network controllability during targeted degree-based node removals.

We assume that for node attacks, the probability of attacking a node, is proportional to some power of its degree. Because we consider directed graphs $\mathcal{G}(N, L)$, with node set $\mathcal{N}$, there are three types of node degree: in-degree, out-degree and total degree. In this paper we will only consider node attacks based upon total degree. If we denote the probability of removing node i with total degree k_i as p_i, we have $p_i = \frac{{k_i}^\alpha}{\sum_{j\in\mathcal{N}} {k_j}^\alpha}$. For $\alpha = 0$, each node has the same probability of being removed, hence for this case targeted node removal corresponds to random node removal, as discussed in Sect. 3. If $\alpha > 0$, the node with a larger degree has a higher probability of being removed; when $\alpha < 0$, the node with the smaller degree has a higher probability of being removed. In this section, we focus on analyzing the results with $\alpha > 0$. Specifically, we consider two cases: $\alpha = 1$ and $\alpha = 10$.

4.1 Analytical Approximation

4.1.1 Case: $\alpha = 1$

The main challenge is to obtain expressions for the generating functions for the in- and out-degree distributions after removing a fraction p of the nodes through attacks. In general, it is not possible to obtain the generating function both for the in- and the out-degree distribution, after a fraction p of nodes has been attacked. Therefore we have to come up with a heuristic to deal with this. Here we will map the targeted node attack process (based upon total degree) into a random node attack process. We suppose that the generating functions of in-degree distribution and out-degree distribution change to those corresponding to random node removal, but such that

the total number of links after randomly removing a fraction $\bar{p}$ of nodes is equal to the total number of links after targeted removal of a fraction p of the nodes. As reported in [24], the fraction $\bar{p}$ can be calculated by

$$\bar{p} = 1 - \frac{f G'_\alpha(f)}{< k >}, \tag{12}$$

where $f \equiv G_\alpha^{-1}(1-p)$, $G_\alpha(x) \equiv \sum_k p_k x^{k^\alpha}$ and $< k >$ is the average total degree of the initial network and p_k is the probability of total degree k. If $\alpha = 1$, $G_\alpha(x) \equiv \sum_k p_k x^k$, which is the generating function for the total degree distribution. For ER networks, the generating function of total degree is $G(x) = e^{-<k>(-x+1)}$ and for SSNs, the generating function of total degree is $G(x) = x^{\frac{<k>}{2}} e^{-\frac{<k>}{2}(-x+1)}$.

4.1.2 Case: $\alpha = 10$

The interesting part of parameter α is that when α approaches ∞, the order of removed nodes follows the rank of node degree values in descending order. At each step, the node with the largest degree will be removed. In the simulations, we adopted large values of α, and we found that the results for $\alpha = 10$ are the same as the results for $\alpha = 100$, which means the result for $\alpha = 10$ is representative for the case $\alpha = \infty$.

We want to develop an analytical method to estimate the corresponding network controllability for $\alpha = 10$. We map the fraction p of removed nodes under targeted attacks for $\alpha = 10$ onto the effective proportion $\bar{p}$ of nodes under random node attack. Under the attack strategy to remove the largest degree node at each step, total degree of all removed nodes can be obtained according to the degree distribution after giving the removed fraction p. The effective proportion $\bar{p}$ is the total degree of all removed nodes normalizing by the total degree of all nodes in the initial network, which can be calculated as $\bar{p} = \frac{\sum_{k=k_{max}}^{k=\bar{k}} p_k N k}{N<k>} = \frac{\sum_{k=k_{max}}^{k=\bar{k}} p_k k}{<k>}$, where the largest degree value is denoted as k_{max}, the probability of removed nodes with degree k is denoted as p_k and degree $\bar{k}$ satisfies $\sum_{k=k_{max}}^{k=\bar{k}} p_k = p$. Similarly, except removed probability $p_{\bar{k}}$, other probability p_k is equal to probability $P(k)$ in the generating function. Then the minimum number of driver nodes can be approximated by replacing argument p by $\bar{p}$ in Eqs. (5)–(6).

4.2 Validation

4.1.2 $\alpha = 1$

We choose the same network set to do simulations under targeted node removal, based on total degree. When using the maximum matching algorithm to calculate

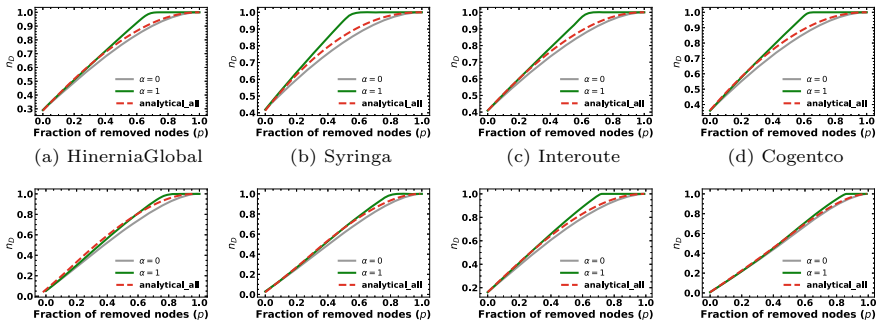
the minimum fraction of the number of driver nodes, we recalculate the fraction value n_D after removing nodes for each kind of targeted attack with $\alpha = 1$. We do 10,000 realizations for each communication network and 1000 realizations for each synthetic network. The simulation results are presented as green lines in Fig. 2. For the analytical method, we employ the effective fraction of removed nodes $\bar{p}$ acquired by Eq. (12). Red lines in Fig. 2 represent the analytical results. We also show the simulation results under random node removals in grey lines in Fig. 2.

We find that the analytical results are a reasonable fit with the simulations, especially for small values of the fraction p of attacked nodes. It indicates that the proposed method of calculating the effective proportion $\bar{p}$ is inaccurate in the late removal stage.

4.2.2 $\alpha = 10$

Analogously, we do the simulations with $\alpha = 10$ under total degree targeted node removal, 10000 realizations for each communication network and 1000 realizations for each synthetic network. The simulation results are shown in the green lines. We present the analytical results in red lines. The simulation results of network controllability under random node attacks are depicted in grey lines. The results with $\alpha = 10$ of total degree target node removal are shown in Fig. 3.

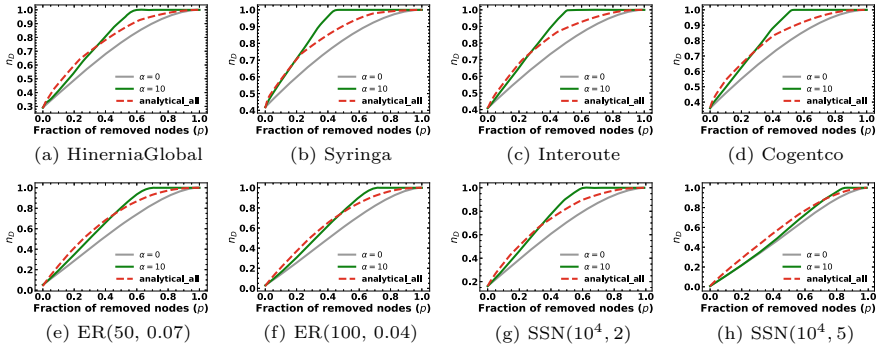
The proposed approaches for the case $\alpha = 10$ can approximate network controllability in a closed-form but do not perfectly fit the simulation results. The analytical result lines are first above the targeted attack lines, then below the targeted attack lines but still above the random attack lines, until the fraction of removed nodes approaches one.



(a) HinerniaGlobal (b) Syringa (c) Interoute (d) Cogentco

(e) ER(50, 0.07) (f) ER(100, 0.04) (g) SSN(10^4, 2) (h) SSN(10^4, 5)

Fig. 2 The minimum fraction of driver nodes n_D during targeted node removal based on the total degree with $\alpha = 1$ in different kinds of networks. The green and grey lines are the average n_D calculated by the maximum matching algorithm over 10,000 realizations of real networks and 1000 realizations of synthetic networks. The grey lines are the results of simulations under random node removal ($\alpha = 0$), and the green lines are the results of removing nodes with probability based on the degree with $\alpha = 1$. The red dashed lines are obtained by the analytical approximation approach



(a) HinerniaGlobal (b) Syringa (c) Interoute (d) Cogentco

(e) ER(50, 0.07) (f) ER(100, 0.04) (g) SSN(10^4, 2) (h) SSN(10^4, 5)

Fig. 3 The minimum fraction of driver nodes n_D during targeted node removal based on the total degree with $\alpha = 10$ in different kinds of networks. The green and grey lines are the average n_D calculated by the maximum matching algorithm over 10,000 realizations of real networks and 1000 realizations of synthetic networks. The grey lines are the results of simulations under random node removal ($\alpha = 0$), and the green lines present the results of removing nodes with probability based on the degree with $\alpha = 10$. The red dashed lines are obtained by the analytical approximation methods

5 Conclusion and Discussion

In this study, we propose analytical methods, based on generating functions, to compute the minimum fraction of the number of driver nodes in directed networks, subject to node removals. We find that the analytical methods fit simulation results very well for random node removals. Moreover, we develop analytical methods for two cases during targeted node removal based on different degrees. One is the probability of a removed node in proportion to the degree, and the other is that a node with the largest degree tends to be removed. We find that the proposed analytical methods for targeted node removal fit the simulation results reasonably well, in particular for small values of the fraction of removed nodes.

In the future, we aim to extend our results by also considering node attacks, based on the in-degree or the out-degree of nodes, and localized node attacks, as in [24]. Also, we would like to validate our results on a larger set of networks, both synthetic and real-world networks, such as scale-free networks, small-world networks and power grids.

References

1. Wu, L., Li, M., Wang, J.-X., Wu, F.-X.: Controllability and its applications to biological networks. J. Comput. Sci. Technol. **34**(1), 16–34 (2019)
2. Rinaldi, M.: Controllability of transportation networks. Transp. Res. Part B Methodol. **118**, 381–406 (2018). [Online] Available: https://www.sciencedirect.com/science/article/pii/S0191261518301930

3. Solimine, P.C.: Network controllability metrics for corruption research. In: Corruption Networks, pp. 29–50. Springer (2021)
4. Kalman, R.E.: Mathematical description of linear dynamical systems. J. Society Ind. Appl. Math. Ser. A Control **1**(2), 152–192 (1963) [Online]. Available: https://doi.org/10.1137/0301010
5. Lin, C.-T.: Structural controllability. IEEE Trans. Autom. Control **19**(3), 201–208 (1974)
6. Jia, J., van Waarde, H.J., Trentelman, H.L., Camlibel, M.K.: A unifying framework for strong structural controllability. IEEE Trans. Autom. Control **66**(1), 391–398 (2021)
7. Olshevsky, A.: Minimal controllability problems. IEEE Trans. Control Netw. Syst. **1**(3), 249–258 (2014)
8. Liu, Y.-Y., Slotine, J.-J., Barabási, A.-L.: Controllability of complex networks. Nature **473**(7346), 167–173 (2011)
9. Cowan, N.J., Chastain, E.J., Vilhena, D.A., Freudenberg, J.S., Bergstrom, C.T.: Nodal dynamics, not degree distributions, determine the structural controllability of complex networks. PloS ONE **7**(6), 1–5 (2012)
10. Van Mieghem, P., Doerr, C., Wang, H., Hernandez, J.M., Hutchison, D., Karaliopoulos, M., Kooij, R.: A framework for computing topological network robustness. Delft Univ. Technol. Rep. **20101218**, 1–15 (2010)
11. Pu, C.-L., Pei, W.-J., Michaelson, A.: Robustness analysis of network controllability. Phys. A Stat. Mech. Appl. **391**(18), 4420–4425 (2012) [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378437112003135
12. Lu, Z.-M., Li, X.-F.: Attack vulnerability of network controllability. PLOS ONE **11**(9), 1–27 (2016) [Online]. Available: https://doi.org/10.1371/journal.pone.0162289
13. Wang, L., Zhao, G., Kong, Z., Zhao, Y.: Controllability and optimization of complex networks based on bridges. Complexity, pp. 1–10 (2020) [Online]. Available: https://ideas.repec.org/a/hin/complx/6695026.html
14. Sun, P., Kooij, R.E., Van Mieghem, P.: Reachability-based robustness of controllability in sparse communication networks. IEEE Trans. Netw. Serv. Manage. **18**(3), 2764–2775 (2021)
15. Lou, Y., Wang, L., Chen, G.: A framework of hierarchical attacks to network controllability. Commun. Nonlinear Sci. Numer. Simul. **98**, 105780 (2021)
16. Menichetti, G., Dall'Asta, L., Bianconi, G.: Network controllability is determined by the density of low in-degree and out-degree nodes. Phys. Rev. Lett. **113**, 078701 (2014) [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.113.078701
17. Lou, Y., Yang, D., Wang, L., Tang, C.-B., Chen, G.: Controllability robustness of Henneberg-growth complex networks. IEEE Access **10**, 5103–5114 (2022)
18. Zhang, Z., Yin, Y., Zhang, X., Liu, L.: Optimization of robustness of interdependent network controllability by redundant design. PLOS ONE **13**(2), 1–17 (2018) [Online]. Available: https://doi.org/10.1371/journal.pone.0192874
19. Dhiman, A., Sun, P., Kooij, R.: Using machine learning to quantify the robustness of network controllability. In: Machine Learning for Networking, pp. 19–39. Springer International Publishing (2021) [Online]. Available: https://doi.org/10.1007/978-3-030-70866-5_2
20. Chen, A., Sun, P., Kooij, R.E.: The recoverability of network controllability. In: 2021 5th International Conference on System Reliability and Safety (ICSRS), pp. 198–208. IEEE (2021)
21. Komareji, M., Bouffanais, R.: Resilience and controllability of dynamic collective behaviors. PLOS ONE **8**(12), 1–15 (2013) [Online]. Available: https://doi.org/10.1371/journal.pone.0082578
22. Knight, S., Nguyen, H.X., Falkner, N., Bowden, R., Roughan, M.: The internet topology zoo. IEEE J. Sel. Areas Commun. **29**(9), 1765–1775 (2011) [Online]. Available: https://networks.skewed.de/net/internet_top_pop
23. Shao, J., Buldyrev, S.V., Braunstein, L.A., Havlin, S., Stanley, H.E.: Structure of shells in complex networks. Phys. Rev. E **80**, 036105 (2009) [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.80.036105
24. Kenett, D.Y., Gao, J., Huang, X., Shao, S., Vodenska, I., Buldyrev, S.V., Paul, G., Stanley, H.E., Havlin, S.: Network of interdependent networks: overview of theory and applications. Netw. Netw. Last Frontier Complex. 3–36 (2014)

Optimal Network Robustness in Continuously Changing Degree Distributions

Masaki Chujyo and Yukio Hayashi

Abstract Realization of highly tolerant networks against malicious attacks is an important issue, since many real-world networks are extremely vulnerable to attacks. Thus, we investigate the optimal robustness of connectivity against attacks on networks in changing degree distribution ranging from power-law to exponential or narrower ones. It is numerically found that the smaller variances of degree distributions lead to higher robustness in this range. Our results will provide important insights toward optimal robustness against attacks in changing degree distributions.

Keywords Robustness against attacks · Continuously changing degree distributions · Variance of degree distribution · Feedback vertex set

1 Introduction

In our modern society, the realization of robust systems against malicious attacks is an important issue. Unfortunately, it has been found that many real-world systems of social, technological, and biological networks commonly have power-law degree distributions, and such networks are extremely vulnerable to targeted attacks [1]. Thus, there are several studies for improving network robustness of connectivity against attacks. In particular, networks with higher degree-degree correlations [13] are known to be more robust against attacks [14, 15]. Such structures with positive degree-degree correlations are called onion-like structures and are known to be robust. Based on increasing the degree-degree correlations, some rewiring methods have been proposed for improving the robustness [17, 18].

M. Chujyo (✉) · Y. Hayashi
Japan Advanced Institute of Science and Technology, 1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan
e-mail: mchujyo@jaist.ac.jp

Y. Hayashi
e-mail: yhayashi@jaist.ac.jp

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_31

Recently, a new strong relation between the robustness and loops in networks is beginning to get attention. The robustness is deeply related to a feedback vertex set (FVS), which is a minimum set of necessary nodes for loops [7]. In other words, a network is easily fragmented after it becomes loopless. This relation is also supported by an asymptotically equivalence of network dismantling and decycling for a random network whose second moment does not diverge [2]. Network dismantling is a minimum set of nodes whose removal makes it a smaller size of connected components, while network decycling is a minimum set of nodes whose removal makes it loopless. Through numerical simulations, the network with a large size of FVS is more robust [6]. Furthermore, based on increasing the size of FVS, two types of rewiring methods have been proposed with and without preserving degrees for improving the robustness [5]. The rewiring methods without preserving degrees significantly increase both the robustness and the size of FVS, even when the degree-degree correlations is negative. In particular, it is found that the rewiring methods without preserving degrees tend to decrease the gap of the maximum and minimum degrees in all tested networks. These results suggest that the robustness and the size of FVS significantly increase as the gap of degrees decreases, equivalently as the variance of degree distribution becomes smaller. Thus, we focus on the variance of degree distributions for improving the robustness of connectivity against attacks.

On the other hand, there are few studies about the robustness against attacks in changing degree distributions. One of them is a study of the robustness in networks with specific degree distributions as power-law and exponential ones. Scale-free networks with power-law degree distributions are more vulnerable to attacks than networks with exponential distributions [1]. In addition, a growing network (GN) model [8–10] generates networks with continuously changing degree distributions between power-law and exponential ones. However, the robustness between them is still unknown exactly. Another of them has investigated the robustness in a special class of networks with multimodal distributions including power-law ones [16]. It is known that bimodal networks with only two types of degrees in this class are the most robust in the meaning of maximizing the sum of two critical thresholds of whole fragmentation by random failures and malicious attacks. This means that the robustness against the sum of attacks and failures increases as the variance of degree distribution decreases. Thus, we focus on the variances of degree distributions to investigate the optimal robustness against attacks in continuously changing degree distributions ranging from power-law to exponential or narrower ones.

2 Continuously Changing Degree Distributions

We introduce a slightly modified growing network (GN) model [8–10] and an inverse preferential attachment (IPA) model [11]. The modified GN model and the IPA model generate various networks with continuously changing degree distributions by a parameter. In each of the following models, a new node is added and connects by m links to existing nodes at each time step. The minimum degree of generated

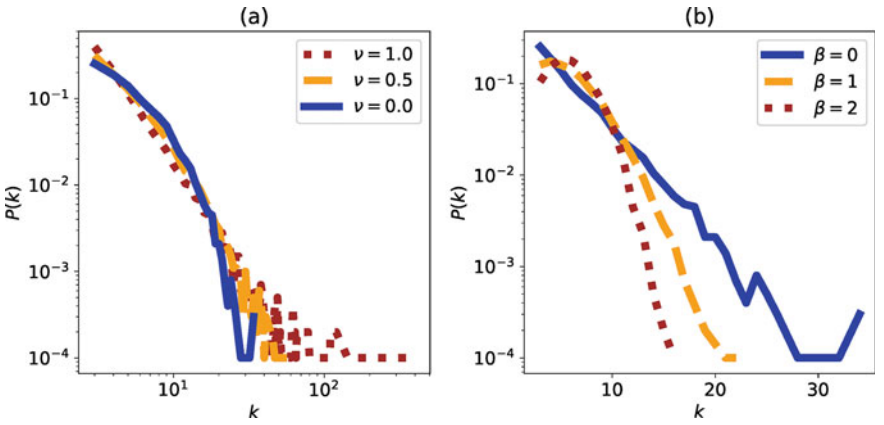


Fig. 1 Degree distributions by **a** the modified GN model for $\nu = 0$, 0.5, and 1 and **b** the IPA model for $\beta = 0$, 1, and 2, in networks with $N = 10{,}000$ nodes and $m = 3$. In both **a** and **b**, the gap between the maximum degree and the minimum degree m becomes smaller as the parameter ν decreases or β increases. **a** Brown dotted and blue solid lines show power-law ($\nu = 1$) and exponential distributions ($\nu = 0$). Orange dashed line shows a degree distribution for $\nu = 0.5$ **b** Blue solid line shows an exponential distribution at $\beta = \nu = 0$. Orange dashed and brown dotted lines show degree distributions for $\beta = 1$ and 2

networks is a constant m and the average degree becomes almost $2m$ for a larger number of nodes. The initial configuration is set as a complete graph with 7 nodes.

In the original GN model, a new node connects $m = 1$ link to an existing node i with the connection probability proportional to k_i^{ν}, $\nu \geq 0$, where k_i denotes a degree of node i. In the original GN model for $m = 1$, it is analytically derived that the degree distributions change from power-law ($\nu = 1$) to power-law with exponential cutoff ($0 < \nu < 1$) or exponential ones ($\nu = 0$) [8–10]. However, networks generated by the original GN model for $m = 1$ are vulnerable because they are random trees, regardless of the degree distribution. Thus, we extend the original GN model to $m \geq 2$ and distinguish it as a modified GN model. We numerically show degree distributions of networks generated by the modified GN model for $m = 3$ in Fig. 1a. The degree distributions change in the same way from power-law to exponential ones in the original GN model. In Fig. 1a with a log-log scale, brown dotted ($\nu = 1$) and blue solid ($\nu = 0$) lines show power-law and exponential degree distributions, which are derived in the original GN model. For $0 \leq \nu \leq 1$, the maximum degree becomes smaller as ν decreases. In addition, Fig. 2a shows the variances σ^2 of degree distributions versus the parameter ν. The variances become smaller as ν decreases. For $\nu > 1$, many links concentrate to a few hub nodes in a star-like structure, however we do not consider it because the robustness against attacks to the centers is obviously weak.

For generating narrower degree distributions than exponential ones, we introduce the IPA model, in which the connection probability from a new node to an existing node i is proportional to $k_i^{-\beta}$, $\beta \geq 0$ [11]. In the IPA model, the degree distributions

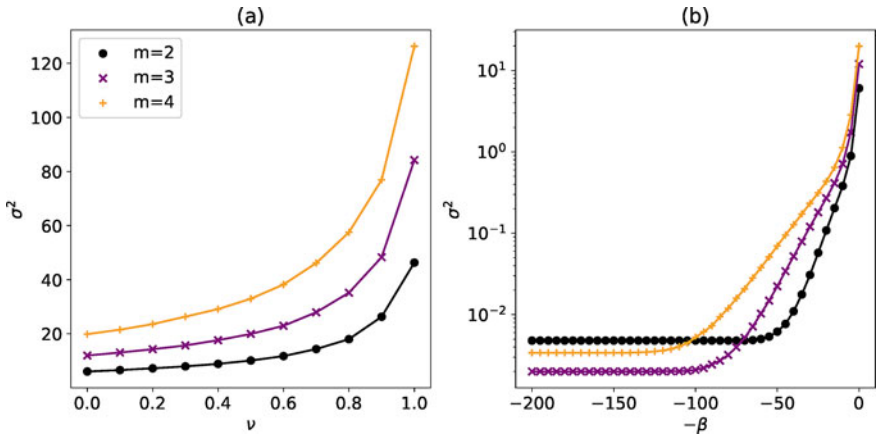


Fig. 2 Variances σ^2 of degree distributions versus (a) the parameter ν in the modified GN model and (b) the parameter β in the IPA model for $N = 10{,}000$ nodes. Black, purple, and orange lines show the results for $m = 2$, 3, and 4, respectively. For each value of m, σ^2 decreases as β increases or ν decreases. Note that the connection probability to a node i with degree k_i is proportional to k_i^{ν} or $k_i^{-\beta}$

continuously change from exponential distributions ($\beta = 0$) to narrower ones as β increases, as shown in Fig. 1b. Figure 1b shows that the maximum degree decreases as β increases. In addition, Fig. 2b shows the variances σ^2 of degree distributions decrease as β increases. However, the variances are converging to constant values ≥ 0. The minimum variances are 0.0048, 0.002, and 0.0034 for $m = 2$, 3, and 4, respectively. The variances cannot become zero even for $\beta \to \infty$, because these networks must have some nodes with degrees smaller than the average degree of $2m$ [11]. Note that lines in Fig. 2 are connected at $\nu = 0$ and $\beta = 0$, since the connection probability functions k^{ν} and $k^{-\beta}$ are continuous as $\nu \to 0$ and $\beta \to 0$.

By combining the modified GN model and the IPA model, we can generate various networks with continuously changing degree distributions ranging from power-law, power-law with exponential cutoff, exponential, and narrower ones. Although the networks generated by these models have specific structures, we focus in particular on effects of degree distribution on robustness against attacks. In particular, networks generated by the IPA model tend to have a chain-like structure [11] shown in Fig. 3a. In Sects. 3 and 4, for investigating the pure effect of degree distributions on the robustness, we use random networks with degree sequences of networks generated by these models. We apply a configuration model [3, 4] to the networks and remove such special structures. Figure 3b illustrates a network after applying a configuration model, which does not have the chain-like structure.

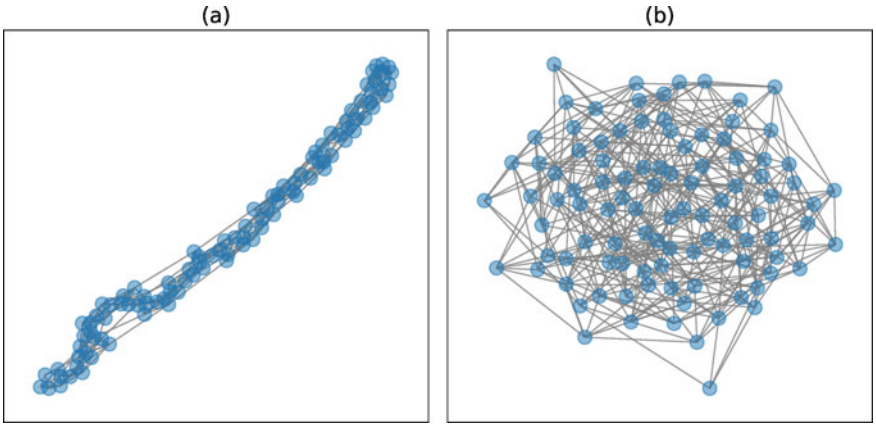


Fig. 3 Visualization examples of networks for $N = 200$ nodes and $m = 4$. They are generated by **a** the IPA model for $\beta = 200$ and **b** the corresponding a configuration model with the same degree distribution

3 Effect of Continuous Changes of Degree Distributions on Robustness

In continuously changing degree distributions, we consider the robustness index [14] against two types of attacks: typical high degree adaptive (degree-based) attacks [1] and more powerful belief propagation (BP) attacks based on network decycling [12]. The robustness index against degree-based and BP attacks denote R_{hub} and R_{bp}, respectively. The robustness index is defined as $R_{\mathrm{hub}} \stackrel{\mathrm{def}}{=} \sum_{q=1}^{N} S(q)/N$, where N is the number of nodes, q is the number of removed nodes by the degree-based attacks, and $S(q)$ is the fraction of nodes in the largest connected components [14]. In the same formula, R_{bp} is defined as the case where nodes are removed by BP attacks. In Sects. 3 and 4, we show the averaged results for 100 configuration models with degree sequences of networks generated by the modified GN model and the IPA model.

First, we show the results for the modified GN model. Table 1 shows the values of R_{hub} and R_{bp} in a configuration model for networks generated by the modified GN model for $N = 10{,}000$ nodes, $m = 2$, 3, and 4, and $\nu = 0$, 0.5, and 1. Remember that ν is a parameter in the connection probability proportional to k^{ν} in the modified GN model. In Table 1, R_{hub} and R_{bp} increase as ν decreases. Thus, in the range from power-law to exponential distributions, a smaller variance of degree distributions leads to higher robustness against both degree-based and BP attacks. Note that R_{bp} is slightly smaller than R_{hub} in comparison with them for the same m and ν.

Next, we show the results for the IPA model. Figure 4a, b show R_{hub} and R_{bp} versus the variances σ^2 of degree distributions in a configuration model for networks generated by the IPA model for $N = 10{,}000$ nodes, $m = 3$, and $\beta = 0$, 10, ..., 200. In Fig. 4a, b, R_{hub} and R_{bp} increase as the variance σ^2 decreases. The green and blue

Table 1 Robustness index in a configuration model for networks generated by the modified GN model for $N = 10{,}000$

	R_{hub}			R_{bp}		
m	$\nu = 0$	0.5	1	$\nu = 0$	0.5	1
2	0.1709	0.1439	0.09887	0.1614	0.1354	0.09213
3	0.2564	0.2316	0.1895	0.2475	0.2234	0.1804
4	0.3078	0.2866	0.2516	0.3000	0.2787	0.2424

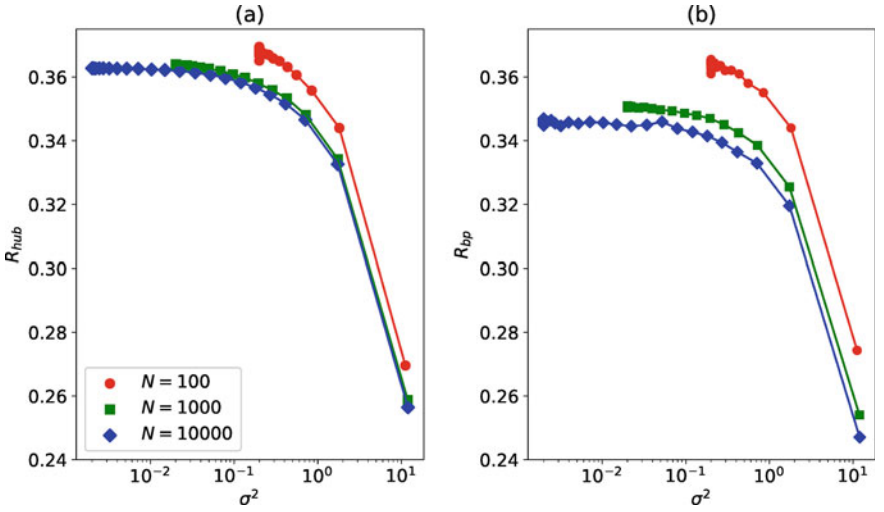


Fig. 4 Robustness against **a** typical degree-based attacks and **b** more powerful BP attacks versus the variances σ^2 of degree distributions in a configuration model by the IPA model for $m = 3$ and $\beta = 0, 10, ..., 200$. Red, green, and blue lines with circle, square, and diamond marks indicate the results for $N = 100$, 1000, and 10,000. In both **a** and **b**, the robustness index becomes higher as the variance is smaller

lines with diamond and square marks in Fig. 4a, b shows that both R_{hub} and R_{bp} are almost unchanged as the variance σ^2 becomes smaller than 10^{-2}. This is because the networks are approaching to regular graphs but not complete regular graphs even for $\beta \to \infty$ [11]. Also, in the range from exponential to narrower distributions, a smaller variance of degree distribution leads to higher robustness against both degree-based and BP attacks.

By combining the above results for the modified GN model and the IPA model, we can find that the robustness increases as the variance decreases in the two ranges from power-law to exponential degree distributions and from exponential to narrower ones. Furthermore, it is suggested that random regular graphs with zero variance seem to have the highest robustness against attacks. Note that the random 2-regular graph is unlikely to be fully connected, so even a random regular graph must have more than the average degree of 3 to be robust.

Table 2 Correlation coefficients of the robustness index and the rate of FVS in a configuration model for networks generated by the modified GN and IPA models for $N = 10{,}000$

	Modified GN model		IPA model	
m	Corr (R_{hub}, FVS)	Corr (R_{bp}, FVS)	Corr (R_{hub}, FVS)	Corr (R_{bp}, FVS)
2	0.9999	0.9986	0.9987	0.9988
3	0.9999	0.9997	0.9997	0.9990
4	0.9999	0.9994	0.9997	0.9988

4 Relation of Robustness and FVS in Changing Degree Distributions

It has been suggested that networks become more robust, when they have a larger rate of FVS by loop enhancement [5, 6]. Thus, we investigate the relation between the robustness and the rate of FVS in a configuration model for networks generated by the modified GN model and the IPA model. However, since to obtain the FVS is NP-hard in a combinatorial optimization problem [7], we apply an approximate method by a message-passing algorithm for estimating the FVS [19].

We show the relation for the modified GN model. Figure 5a, b show R_{hub} and R_{bp} versus the estimated rate of FVS in a configuration model of the modified GN model for $N = 10{,}000$ nodes and $\nu = 0, 0.1, ..., 1$. Black circle, purple star, and orange hexagon points show the results for $m = 2$, 3, and 4, respectively. In Fig. 5a, R_{hub} increases as the rate of FVS increases. This is also supported by their correlation coefficients, which are higher than 0.99 as shown in Table 2. Furthermore, both R_{hub} and the rate of FVS increase as the variances of degree distributions decrease as shown by color gradations in Fig. 5a. The color gradation of points is darker as the variance decreases. Similarly, R_{bp} is strongly correlated with the rate of FVS as shown in Fig. 5b and Table 2. Therefore, in degree distributions generated by the modified GN model, the robustness against both attacks and the rate of FVS increase as the variance of degree distributions decreases. Note that the values of R_{hub} and R_{bp} are almost same as shown in Figs. 5a, b.

For degree distributions by the IPA model, similar results are obtained as them by the modified GN model. Figure 6a, b show R_{hub} and R_{bp} versus the estimated rate of FVS in a configuration model for networks generated by the IPA model for $N = 10{,}000$ nodes and $\beta = 0, 10, ..., 200$. Even in narrower degree distributions by the IPA model, both R_{hub} and R_{bp} increase as the rate of FVS increases, as shown in Fig. 6a, b. This is also supported by their correlation coefficients, which are higher than 0.99 as shown in Table 2. However, the value of R_{bp} is slightly lower than that of R_{hub} in Fig. 6a, b. Furthermore, color gradation in Fig. 6a, b shows that both R_{hub} and R_{bp} increase as the variance decreases, and that rate of FVS also increases.

By combining the results for the modified GN model and the IPA model, we can find that the strong correlation of the robustness and the rate of FVS holds in the two

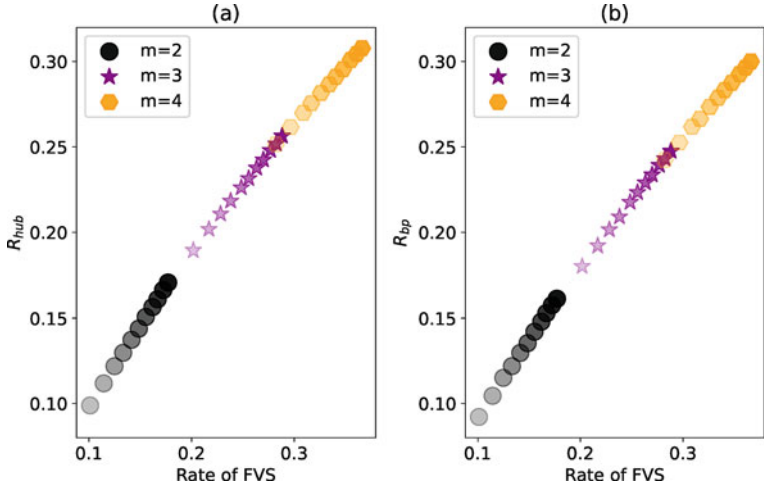


Fig. 5 Robustness index versus the rate of FVS in a configuration model for networks generated by the modified GN model for $\nu = 0, 0.1, ..., 1$ and $N = 10{,}000$. The y-axis shows the robustness index against **a** typical degree-based attacks and **b** more powerful BP attacks. Black circle, purple star, and orange hexagon points show the results for $m = 2, 3$, and 4, respectively. Color gradation of points are darker as the variances of degree distributions are smaller. In both **a** and **b**, the robustness increases as the variance decreases, and the rate of FVS also increases

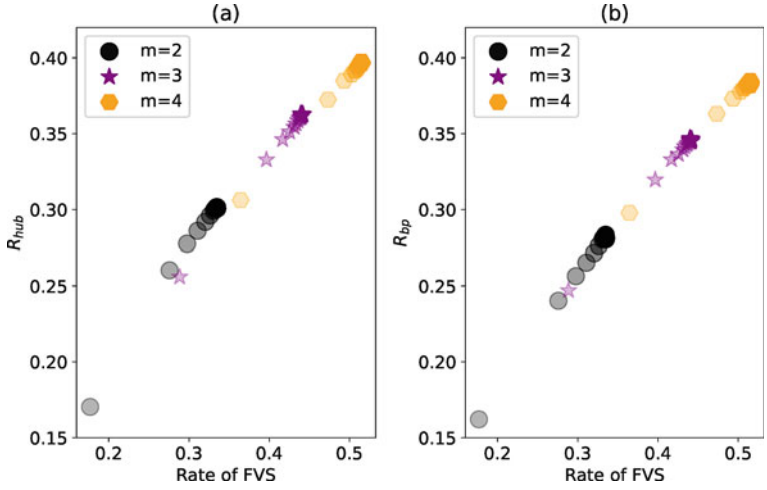


Fig. 6 Robustness index versus the rate of FVS in a configuration model for networks generated by the IPA model for $N = 10{,}000$ and $\beta = 0, 10, ..., 200$. The y-axis shows the robustness index against **a** typical degree-based attacks and **b** more powerful BP attacks. Black circle, purple star, and orange hexagon points show the results for $m = 2$, 3, and 4, respectively. Color gradation of points are darker as the variances of degree distributions are smaller. In both **a** and **b**, the robustness increases as the variance decreases, and the rate of FVS also increases

ranges from power-law to exponential degree distributions and from exponential to narrower ones. Moreover, both the robustness and rate of FVS become higher as the variance of degree distributions decreases in these ranges.

5 Effect of Chain-Like Structure Generated by the IPA Model on Robustness

We discuss the effect of chain-like structure on the robustness in networks generated by the IPA model, in comparing them before and after applying a configuration model. Remember that chain-like structure with a large diameter is generated by the IPA model especially for a large β as shown in Fig. 3a. The reason for generating chain-like structure is as follows [11]. For $k^{-\beta}$-attachment for a very large β, at the current time step t, a new node connects to the existing nodes added at $t-1$, $t-2$, ..., $t-(m-1)$ whose degrees are m, $m-1$, ..., $2m-2$, because the increase of degree is only one at each time step in prohibiting multi-links. The remaining one link connects to an existing node with a degree $2m-1$. In addition, by assuming that a new node connects to the oldest node with a degree $2m-1$, the diameter of a generated network is estimated by $N/\Delta t$, $\Delta t = m(m+3)/2$ [11]. Here, this diameter of $\mathcal{O}(N)$ means the existing of chain-like structure.

As β increases, the variance of degree distributions becomes smaller and chain-like structure is emerged simultaneously by the IPA model. However, for the robustness, there is a trade-off between decreasing variance of degree distributions and emergence of chain-like structure. The emergence of chain-like structure makes a network less robust since it is easily fragmented by removing nodes, whereas decreasing the variance makes it more robust. Therefore, these two opposite effects work on the robustness in the IPA model.

We confirm this trade-off for networks generated by the IPA model before randomizing by a configuration model. We show the results of R_{hub} before and after applying the configuration model in Fig. 7. In Fig. 7, solid lines show R_{hub} on a configuration model of the IPA model for $\beta = 0, 10, ..., 200$ and $m = 2, 3$, and 4. This case shown by solid lines reflects the pure effect of degree distributions on the robustness, since chain-like structure is removed by applying a configuration model. In solid lines of Fig. 7, R_{hub} increases as β increases in corresponding smaller σ^2. On the other hand, dotted lines in Fig. 7 show R_{hub} on networks by the IPA model before applying a configuration model. Each dotted line of R_{hub} moves up and down as β increases because of the trade-off of two effects: emergence of chain-like structure and making the variance smaller. By the trade-off, R_{hub} takes a local minimum value for $m = 2$ and 3 in Fig. 7a, b. For $m = 2$, the local minimum of R_{hub} is given at $\beta = 20$, 45, and 55 for $N = 100$, 1000, and 10,000 nodes, respectively. For $m = 3$, the local minimum of R_{hub} is given at $\beta = 35$, 60, and 85 for $N = 100$, 1000, and 10,000 nodes, respectively. However, there are no such local minimum values for

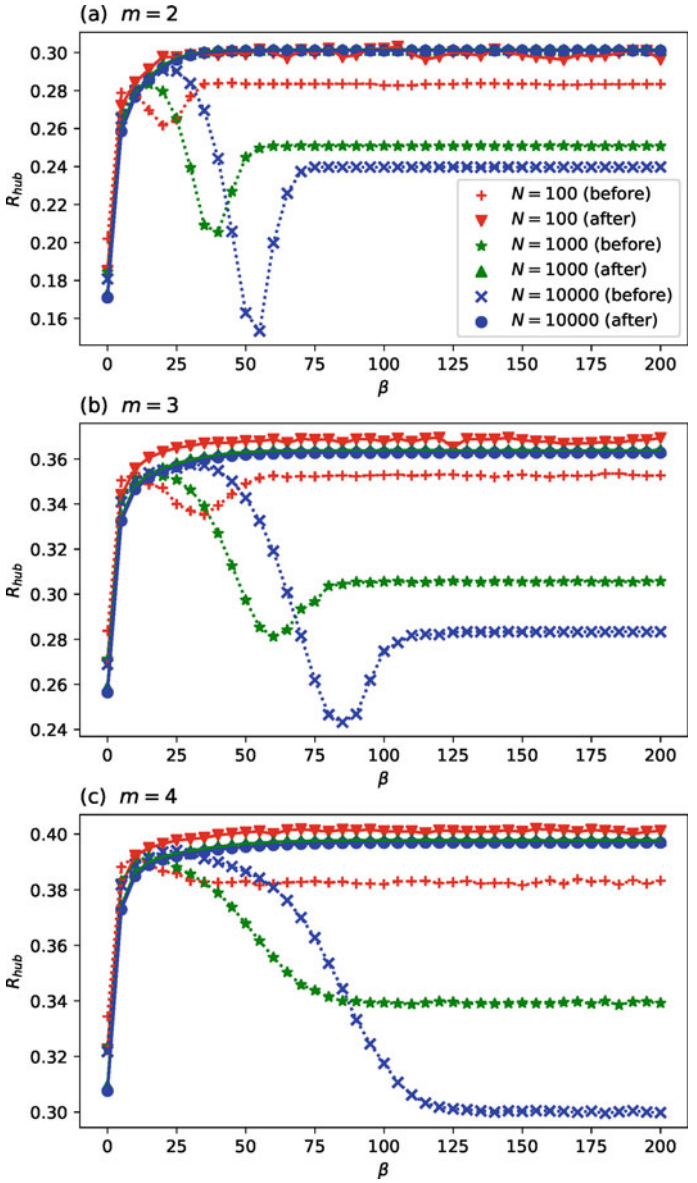


Fig. 7 Robustness index against degree-based attacks versus the parameter β in the IPA model. Red, green, and blue lines are the results for $N = 100$, 1000, and 10,000. The dotted and solid lines indicate the results before and after applying a configuration model for **a** $m = 2$, **b** $m = 3$, and **c** $m = 4$

$m = 4$ in Fig. 7c. The reason is still unknown in revealing the detail mechanism. In addition, the values of R_{hub} are convergent for $\beta > 125$, because the attachment becomes always connecting to nodes with the minimum degree.

6 Conclusion

We investigate the optimal robustness in continuously changing degree distributions ranging from power-law, power-law with exponential cutoff, exponential, and narrower ones generated by the modified GN model and the IPA model. By applying a configuration model in order to investigate the pure effect of degree distributions on the robustness against the typical degree-based attacks and more powerful BP attacks, we obtain that the robustness against both attacks and the rate of FVS increase as the variance of degree distributions decreases. These results suggest that networks with the minimum variance have the optimal robustness against attacks.

Acknowledgements This research is supported in part by JSPS KAKENHI Grant Number JP.21H03425.

References

1. Albert, R., Jeong, H., Barabási, A.L.: Error and attack tolerance of complex networks. Nature **406**(6794), 378–382 (2000)
2. Braunstein, A., Dall'Asta, L., Semerjian, G., Zdeborová, L.: Network dismantling. Proc. Nat. Acad. Sci. **113**(44), 12368–12373 (2016). https://www.pnas.org/content/113/44/12368
3. Callaway, D.S., Hopcroft, J.E., Kleinberg, J.M., Newman, M.E.J., Strogatz, S.H.: Are randomly grown graphs really random? Phys. Rev. E **64**(4), 041902 (Sep 2001). https://link.aps.org/doi/10.1103/PhysRevE.64.041902
4. Catanzaro, M., Boguñá, M., Pastor-Satorras, R.: Generation of uncorrelated random scale-free networks. Phys. Rev. E **71**(2), 027103 (Feb 2005). https://link.aps.org/doi/10.1103/PhysRevE.71.027103
5. Chujyo, M., Hayashi, Y.: A loop enhancement strategy for network robustness. Appl. Netw. Sci. **6**(1), 1–13 (2021)
6. Hayashi, Y., Uchiyama, N.: Onion-like networks are both robust and resilient. Sci. Rep. **8**(1), 1–13 (2018)
7. Karp, R.M.: Reducibility among combinatorial problems. In: Complexity of Computer Computations, pp. 85–103. Springer (1972)
8. Krapivsky, P.L., Redner, S.: Organization of growing random networks. Phys. Rev. E **63**(6), 066123 (May 2001). https://link.aps.org/doi/10.1103/PhysRevE.63.066123
9. Krapivsky, P.L., Redner, S.: A statistical physics perspective on web growth. Comput. Netw. **39**(3), 261–276 (2002). https://www.sciencedirect.com/science/article/pii/S1389128602002128
10. Krapivsky, P.L., Redner, S., Leyvraz, F.: Connectivity of growing random networks. Phys. Rev. Lett. **85**(21), 4629–4632 (Nov 2000). https://link.aps.org/doi/10.1103/PhysRevLett.85.4629
11. Liao, F., Hayashi, Y.: Emergence of robust and efficient networks in a family of attachment models. Phys. A: Stat. Mech. Appl. **599**, 127427 (2022). https://www.sciencedirect.com/science/article/pii/S0378437122003168

12. Mugisha, S., Zhou, H.J.: Identifying optimal targets of network attack by belief propagation. Phys. Rev. E **94**(1), 012305 (Jul 2016). https://link.aps.org/doi/10.1103/PhysRevE.94.012305
13. Newman, M.E.J.: Assortative mixing in networks. Phys. Rev. Lett. **89**(20), 208701 (Oct 2002). https://link.aps.org/doi/10.1103/PhysRevLett.89.208701
14. Schneider, C.M., Moreira, A.A., Andrade, J.S., Havlin, S., Herrmann, H.J.: Mitigation of malicious attacks on networks. Proc. Nat Acad. Sci. **108**(10), 3838–3841 (2011). https://www.pnas.org/content/108/10/3838
15. Tanizawa, T., Havlin, S., Stanley, H.E.: Robustness of onionlike correlated networks against targeted attacks. Phys. Rev. E **85**(4), 046109 (Apr 2012). https://link.aps.org/doi/10.1103/PhysRevE.85.046109
16. Tanizawa, T., Paul, G., Havlin, S., Stanley, H.E.: Optimization of the robustness of multimodal networks. Phys. Rev. E **74**(1), 016125 (Jul 2006). https://link.aps.org/doi/10.1103/PhysRevE.74.016125
17. Wu, Z.X., Holme, P.: Onion structure and network robustness. Phys. Rev. E **84**(2), 026106 (Aug 2011). https://link.aps.org/doi/10.1103/PhysRevE.84.026106
18. Xulvi-Brunet, R., Sokolov, I.M.: Reshuffling scale-free networks: from random to assortative. Phys. Rev. E **70**(6), 066102 (Dec 2004). https://link.aps.org/doi/10.1103/PhysRevE.70.066102
19. Zhou, H.J.: Spin glass approach to the feedback vertex set problem. Eur. Phys. J. B **86**(11), 1–9 (2013)

Investments in Robustness of Complex Systems: Algorithm Design

Van-Sy Mai, Richard J. La, and Abdella Battou

Abstract We study the problem of determining suitable investments in improving the robustness of complex systems comprising many component systems with an aim of minimizing the (time) average costs to system operators. The problem is formulated as an optimization problem that is nonconvex and challenging to solve for large systems. We propose two approaches to finding a good solution to the optimization problem: the first approach is based on a gradient method and finds a local optimizer. The second approach makes use of a convex relaxation of the original problem and provides both a lower bound on the optimal value and a feasible point. The lower bound can be used to bound the optimality gap of the solutions obtained by our methods. We provide numerical results to demonstrate the effectiveness of the proposed approaches.

Keywords Complex systems · Optimization · Resilience · Robustness

1 Introduction

With increasing complexity, modern engineering systems, such as information and communication networks and power systems, consist of many (component) systems that depend on each other to deliver their services. This interdependence among systems makes it possible for a local failure or infection of a system by malware

V.-S. Mai · A. Battou
NIST, Gaithersburg, MD 20899, USA
e-mail: vansy.mai@nist.gov

A. Battou
e-mail: abdella.battou@nist.gov

R. J. La (✉)
NIST and University of Maryland, College Park, MD 20742, USA
e-mail: richard.la@nist.gov
URL: http://www.ece.umd.edu/hyongla

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_32

to spread to other systems. From this viewpoint, it is clear that sound investments in robustness of the complex system should consider the interdependence among comprising systems. A similar issue arises also in the problem of managing the spread of an infectious disease via social contacts.

The problem of optimizing the investments in robustness of complex systems or the mitigation of disease spread has been studied extensively. In [1–4], researchers adopted a game theoretic formulation to study the problem of security investments with distributed agents. In another line of research more closely related to our study, researchers examined optimal strategies using vaccines/immunization (prevention) [5], antidotes or curing rates (recovery) [6–8] or a combination of both preventive and recovery measures [9, 10]. However, these studies do not take into account dynamics; they focus on either the expected costs stemming from single or cascading failures/infections [3, 4, 11] or the exponential decay rate to the disease-free state as a key performance metric. When systems experience random failures over time, the exponential decay rate adopted in the previous studies is no longer a suitable performance metric.

In a recent study, Mai et al. [12, 13] investigated the problem of minimizing the (time) average costs of a system operator while accounting for dynamics, where the costs include both (security) investments and economic losses incurred following failures or infections. However, the authors considered investments only in resilience, but not in recovery. In this paper, we extend this study and consider investments in both resilience and recovery. It turns out that incorporating two different types of investments complicates the optimization problem for determining optimal investments significantly. This is due to additional *coupling* terms that are introduced in the new model, which were not present in [12, 13]. This leads to a highly nonconvex optimization problem that is difficult to solve in general. However, we show that, under a technical condition, we can formulate a convex relaxation that provides a lower bound on the optimal value *and* a good feasible solution for the original problem (Theorem 4), whose optimality gap can be bounded. In addition, we show that a gradient-based method also produces a good-quality solution.

Notation and Terminology — Let $\mathbb{R}$ and $\mathbb{R}_+$ denote the set of real numbers and nonnegative real numbers, respectively. For a matrix $A = [a_{i,j}]$, let $a_{i,j}$ denote its (i, j) element and A^{T} its transpose. We use boldface letters to denote (column) vectors, e.g., $\mathbf{x} = [x_1, ..., x_n]^{\mathrm{T}}$, $\mathbf{0} = [0, ..., 0]^{\mathrm{T}}$, and $\mathbf{1} = [1, ..., 1]^{\mathrm{T}}$. For any two vectors $\mathbf{x}$ and $\mathbf{y}$ of the same dimension, $\mathbf{x} \circ \mathbf{y}$, $\frac{\mathbf{x}}{\mathbf{y}}$, and $\mathbf{x}^{\mathbf{y}}$ are their element-wise product, division, and exponentiation, respectively. For $\mathbf{x} \in \mathbb{R}^n$, $\mathtt{diag}(\mathbf{x}) \in \mathbb{R}^{n \times n}$ denotes the diagonal matrix with diagonal elements $x_1, \ldots, x_n$.

A directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consists of a set of nodes $\mathcal{V}$ and a set of directed edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. A directed path in $\mathcal{G}$ is a sequence of directed edges in the form $\big((i_1, i_2), (i_2, i_3), ..., (i_{k-1}, i_k)\big)$. The graph $\mathcal{G}$ is strongly connected if there is a directed path from each node to any other node.

The rest of the paper is organized as follows: Sect. 2 describes the setup and the problem formulation, including the optimization problem. Our proposed approaches

are described in Sect. 3, followed by numerical studies in Sect. 4. We conclude in Sect. 5.

2 Model and Formulation

The complex system under consideration consists of N ($N \gg 1$) systems, and we denote the set of (component) systems by $\mathcal{A} = \{1, \ldots, N\}$. Each system in a subset $\mathcal{A}_R$ ($\subset \mathcal{A}$) experiences random failures. The frequency with which a system $i \in \mathcal{A}_R$ suffers random failures depends on the amount of investments in its *resilience*, which we denote by s_i^p; when system i invests s_i^p in improving its *resilience*, it experiences random failures according to a Poisson process with failure rate $\lambda_i(s_i^p)$. Here, we assume that $\lambda_i(s_i^p) = \bar{\lambda}_i \times q_i(s_i^p)$, where $\bar{\lambda}_i \geq 0$ is the failure rate when no investment is made in its resilience, and $q_i : \mathbb{R}_+ \to [0, 1]$ is a decreasing function and quantifies how the resilience of system i improves with its investment in resilience. We assume $\bar{\lambda}_j = 0$ for every system $j \in \mathcal{A} \setminus \mathcal{A}_R =: \mathcal{A}_R^c$.

In addition to random failures, systems also experience *secondary* failures brought on by the failures of other systems due to interdependence among systems. The rate at which the failure of system i causes that of another system j depends on system j's resilience and is equal to $\xi_{i,j} \times q_j(s_j^p)$, where $\xi_{i,j} \in \mathbb{R}_+$. Thus, even systems in $\mathcal{A}_R^c$ can experience secondary failures. Note that this failure transmission rate depends on system j's investment in resilience. When $\xi_{i,j} > 0$, we say that system i supports system j or system j depends on system i. We adopt the convention $\xi_{i,i} = 0$ for all $i \in \mathcal{A}$.

When system i suffers a failure, the *recovery time* required to repair the system and put it back in service depends on the amount of investment in *recovery*, which we denote by s_i^r; when system i invests s_i^r in *recovery*, the recovery times are given by independent and identically distributed exponential random variables with parameter $\delta_i(s_i^r)$. We assume that $\delta_i : \mathbb{R}_+ \to (0, \infty)$ is strictly increasing. Furthermore, the recovery times of different systems are assumed to be mutually independent.

In addition to the investments in the resilience and recovery of systems, the system operator also incurs other costs; when system i fails, it can cause economic losses, e.g., some servers in system i may need to be taken offline for inspection and repair and remain inaccessible during the period to other systems that depend on the servers. We call the economic losses *failure costs*. To model the failure costs, we assume that the failure of system i causes economic losses of c_i per unit time. Define $\mathbf{c} = (c_i : i \in \mathcal{A})$ to be the failure cost vector.

From our discussion, we can define a following dependence graph $\mathcal{G} = (\mathcal{A}, \mathcal{E})$, where $\mathcal{E} := \{(i, j) \mid \xi_{i,j} > 0, \ i, j \in \mathcal{A}\}$. Let $B = [b_{i,j} : i, j \in \mathcal{A}]$ be an $N \times N$ matrix that describes the failure transmission rates among systems, where the element $b_{i,j}$ is equal to $\xi_{j,i}$. We assume that B is irreducible. Note that B is irreducible if and only if the dependence graph $\mathcal{G}$ is strongly connected.

2.1 Model

We adopt the well-known Susceptible-Infected-Susceptible (SIS) model to capture the evolution of the state of each system: if a system i is up and running at time $t \in \mathbb{R}_+$, we say that the system is at 'susceptible' state. If the system is being repaired following a failure at time t, we say that the system is 'infected'. Let $\mathbf{p}(t)$ be a vector, whose i-th element is the probability that system i is at 'infected' state at time $t \in \mathbb{R}_+$. The dynamics of $\mathbf{p}(t)$ is approximated using the following (Markov) differential equations, starting with $\mathbf{p}(0)$ at $t = 0$:

$$\dot{\mathbf{p}}(t) = (\mathbf{1} - \mathbf{p}(t)) \circ \left(\bar{\boldsymbol{\lambda}} + B\mathbf{p}(t)\right) \circ \mathbf{q}(\mathbf{s}) - \boldsymbol{\delta}(\mathbf{s}) \circ \mathbf{p}(t), \quad t \in \mathbb{R}_+, \tag{1}$$

where $\mathbf{s} = \left(\mathbf{s}_i = (s_i^p, s_i^r) : i \in \mathcal{A}\right)$, $\bar{\boldsymbol{\lambda}} = \left(\bar{\lambda}_i : i \in \mathcal{A}\right)$, $\mathbf{q}(\mathbf{s}) = \left(q_i(s_i^p) : i \in \mathcal{A}\right)$ and $\boldsymbol{\delta}(\mathbf{s}) = \left(\delta_i(s_i^r) : i \in \mathcal{A}\right)$.

Suppose that, for each fixed investment profile $\mathbf{s} = (\mathbf{s}_i : i \in \mathcal{A})$ in $\mathbb{R}_+^{2N}$, the system state $\mathbf{p}(t)$ converges to a stable equilibrium (the existence and uniqueness of such an equilibrium will be addressed shortly), which we denote by $\bar{\mathbf{p}}(\mathbf{s})$. From (1), it is clear that $\bar{\mathbf{p}}(\mathbf{s})$ is a solution to the following equation.

$$\mathbf{g}^{\mathbf{s}}(\mathbf{p}) := (\mathbf{1} - \mathbf{p}) \circ \left(\bar{\boldsymbol{\lambda}} + B\mathbf{p}\right) \circ \mathbf{q}(\mathbf{s}) - \boldsymbol{\delta}(\mathbf{s}) \circ \mathbf{p} = \mathbf{0} \tag{2}$$

We are interested in solving the following problem:

$$[\mathbf{P0}] \qquad \min_{\mathbf{s} \geq \mathbf{0}} \quad F(\mathbf{s}) := w(\mathbf{s}) + \mathbf{c}^{\mathrm{T}} \bar{\mathbf{p}}(\mathbf{s}), \tag{3}$$

where $w : \mathbb{R}_+^{2N} \to \mathbb{R}_+$ is the cost function that quantifies the investment costs. The second term in (3) is the average failure costs that the system operator suffers due to the failures of systems. Although we do not impose any constraints on $\mathbf{s}$ (other than nonnegativity), our analysis can be extended to handle constraints, e.g., $\mathbf{1}^{\mathrm{T}}\mathbf{s} \leq s_{\mathrm{bgt}}$, where s_{bgt} is the available budget for investments.

3 Main Analysis

In order to make progress, we introduce following assumptions.

Assumption 1 Suppose that $\mathcal{G}$ is strongly connected and the following holds:

A1-a. At least one system experiences random failures with a positive rate, i.e., $\mathcal{A}_R \neq \emptyset$ and $\bar{\lambda}_i > 0$ for all $i \in \mathcal{A}_R$.

A1-b. For each $i \in \mathcal{A}$, the function $q_i(s_i^p) = (1 + \kappa_i s_i^p)^{-\alpha_i}$, where κ_i and α_i are some positive constants.

A1-c. For each $i \in \mathcal{A}$, the recovery rate of system i is given by $\delta_i(s_i^r) = \theta_i(1 + \zeta_i \cdot s_i^r)^{\beta_i} > 0$, where β_i, θ_i and ζ_i are some positive constants.

A1-d. The cost of investments is equal to $w(\mathbf{s}) = \mathbf{1}^{\mathrm{T}}\mathbf{s}^{p} + \mathbf{1}^{\mathrm{T}}\mathbf{s}^{r}$.

Assumption A1-b can be viewed as a form of the law of diminishing returns with increasing investments in resilience, where the shape is determined by the parameters α_i and κ_i. Similarly, since the mean recovery time is inversely proportional to the recovery rate, the law of diminishing returns may suggest $\beta_i < 1$ in Assumption A1-c. Larger values of α_i and κ_i (resp. β_i, θ_i, and ζ_i) indicate higher effectiveness of available tools for improving resilience (resp. expediting recovery) and, thus, greater benefits from investments in resilience and recovery.

The following theorem states that, for fixed investments $\mathbf{s} \in \mathbb{R}^{2N}_{+}$, there is a unique equilibrium of the differential system described by (1); see, e.g., [14].

Theorem 1 *Suppose that Assumption 1 holds. Then, for fixed investments* $\mathbf{s} \in \mathbb{R}^{2N}_{+}$, *there is a unique equilibrium* $\bar{\mathbf{p}}(\mathbf{s}) \in (0, 1)^N$ *that satisfies* (2).

Theorem 1 asserts that the unique equilibrium of (1) satisfying (2) is strictly positive. Hence, under Assumption 1, we can rewrite the constraints in (2) as

$$(\mathbf{p}^{-1} - \mathbf{1}) \circ (\bar{\boldsymbol{\lambda}} + B\mathbf{p}) = \boldsymbol{\theta} \circ (\mathbf{1} + \boldsymbol{\kappa} \circ \mathbf{s}^{p})^{\boldsymbol{\alpha}} \circ (\mathbf{1} + \boldsymbol{\zeta} \circ \mathbf{s}^{r})^{\boldsymbol{\beta}}, \tag{4}$$

where $\mathbf{p}^{-1} = (p_i^{-1} : i \in \mathcal{A})$. Based on this observation, we reformulate our original problem [P0] in (3) as the following equivalent problem:

$$\begin{aligned} [\mathbf{P1}] \qquad \min_{\mathbf{s},\mathbf{p}} \quad & w(\mathbf{s}) + \mathbf{c}^{\mathrm{T}}\mathbf{p} =: f(\mathbf{s}, \mathbf{p}) \\ \text{s.t.} \quad & (4),\ \mathbf{p} \in (0, 1]^N,\ \mathbf{s} \geq \mathbf{0} \end{aligned}$$

The main difficulty in solving this problem lies with the constraint in (4). Specifically, the constraint in (4) involves bilinear terms that are not only nonconvex, but also known to be difficult to handle. For this reason, although the objective function is linear in optimization variables $\mathbf{s}$ and $\mathbf{p}$, problem [P1] is nonconvex.

In view of the observation that [P1] is nonconvex, finding an optimal point for a large system (with $N \gg 1$) is in general challenging. In this section, we will discuss how we can obtain a feasible point $(\mathbf{s}', \mathbf{p}')$ to [P1] along with a lower bound f_{lb} on the optimal value f^* of [P1] so that we can bound the gap $f(\mathbf{s}', \mathbf{p}') - f^*$ using $f(\mathbf{s}', \mathbf{p}') - f_{lb}$. In order to find a lower bound on f^*, under a technical assumption, we formulate a convex relaxation of [P1], which can be solved efficiently. We find a feasible point to [P1] in two different ways; in the first method, we use a gradient-based method to find a local minimizer of [P1]. In the second method, we use an optimal point to the aforementioned convex relaxation to construct a feasible point and show that it solves [P1] under certain conditions.

In order to cope with the difficulty caused by constraint in (4), we first rewrite the constraint as following two constraints using an auxiliary variable $\boldsymbol{\phi} \geq \mathbf{1}$.

$$(\mathbf{p}^{-1} - \mathbf{1}) \circ (\bar{\boldsymbol{\lambda}} + B\mathbf{p}) = \boldsymbol{\theta} \circ \boldsymbol{\phi} \tag{5a}$$

$$\boldsymbol{\phi} = (\mathbf{1} + \boldsymbol{\kappa} \circ \mathbf{s}^{p})^{\boldsymbol{\alpha}} \circ (\mathbf{1} + \boldsymbol{\zeta} \circ \mathbf{s}^{r})^{\boldsymbol{\beta}} \tag{5b}$$

Notice that constraint (5a) involves only optimization variables $\mathbf{p}$, whereas constraint (5b) has optimization variables $\mathbf{s}$. This observation will be exploited in our algorithm design below. Here, we will briefly illustrate the usefulness of this structure. Consider the following subproblems for fixed $\boldsymbol{\phi} \geq \mathbf{1}$:

$$\begin{aligned} [\mathbf{SP1}] \qquad & \min_{\mathbf{p}} \quad \mathbf{c}^{\mathrm{T}}\mathbf{p} =: g(\mathbf{p}) \\ & \text{s.t.} \quad \text{(5a)}, \ \mathbf{p} \in (0,1]^N \end{aligned}$$

$$\begin{aligned} [\mathbf{SP2}] \qquad & \min_{\mathbf{s}} \quad w(\mathbf{s}) \\ & \text{s.t.} \quad \text{(5b)}, \ \mathbf{s} \geq \mathbf{0} \end{aligned}$$

Here, in view of Theorem 1, [SP1] is simply the problem of finding the equilibrium failure probability $\bar{\mathbf{p}}$ for fixed $\boldsymbol{\theta} \circ \boldsymbol{\phi}$ corresponding to some investment profile $\mathbf{s}$. Unfortunately, there appears to be no closed-form solution to this problem. One can, however, resort to a numerical method instead as shown below.

Theorem 2 *([13]) Suppose $\bar{\boldsymbol{\lambda}} \gneq \mathbf{0}$, $\boldsymbol{\theta} > \mathbf{0}$, and B is irreducible. Then, the iteration*

$$\mathbf{p}_{k+1} = \frac{\bar{\boldsymbol{\lambda}} + B\mathbf{p}_k}{\bar{\boldsymbol{\lambda}} + B\mathbf{p}_k + \boldsymbol{\theta} \circ \boldsymbol{\phi}}, \ k \in \mathbb{N}, \tag{6}$$

converges to a unique equilibrium $\hat{\mathbf{p}}$ when starting with any $\mathbf{p}_0$ such that $\hat{\mathbf{p}} \leq \mathbf{p}_0 \leq \mathbf{1}$. Moreover, the convergence is exponential with some rate $\rho_0 < 1 - \min_{i \in \mathcal{A}} \hat{p}_i$.

Next, let us consider problem [SP2], which amounts to finding optimal investments for a given $\boldsymbol{\phi}$, or equivalently, for a given failure probability $\hat{\mathbf{p}}$. Unlike [SP1], problem [SP2] can be solved analytically as follows:

Theorem 3 *For each $\boldsymbol{\phi} \geq \mathbf{1}$, the solution to* [SP2] *is given by*

$$\hat{s}_i^r(\boldsymbol{\phi}) = \zeta_i^{-1} \max\left\{0, \tau_i \phi_i^{1/(\alpha_i+\beta_i)} - 1\right\} \ \text{with } \tau_i := \left(\frac{\beta_i \zeta_i}{\alpha_i \kappa_i}\right)^{\frac{\alpha_i}{\alpha_i+\beta_i}}$$

$$\hat{s}_i^p(\boldsymbol{\phi}) = \kappa_i^{-1}\Big(\phi_i^{1/\alpha_i} \max\left\{1, \tau_i \phi_i^{1/(\alpha_i+\beta_i)}\right\}^{-\beta_i/\alpha_i} - 1\Big).$$

The proof of this theorem is straightforward and is omitted here.

Denote the optimal values of [SP1] and [SP2] for fixed $\boldsymbol{\phi} \geq \mathbf{1}$ by $g^*(\boldsymbol{\phi})$ and $w^*(\boldsymbol{\phi})$, respectively. Then, in principle, we can solve problem [P1] by solving

$$\min_{\boldsymbol{\phi} \geq \mathbf{1}} \quad g^*(\boldsymbol{\phi}) + w^*(\boldsymbol{\phi}). \tag{7}$$

Unfortunately, this is in general not a convex problem due to the implicit function $g^*(\boldsymbol{\phi})$ and possible nonconvexity of $\hat{s}_i^p(\boldsymbol{\phi})$ and $\hat{s}_i^r(\boldsymbol{\phi})$ (which is the case when $\alpha_i +$

$\beta_i > 1$) as described in Theorems 2 and 3, respectively. As a result, we resort to a numerical method that can find a (local) optimizer of the problem. Among different methods for solving nonconvex problems, we consider next in Sect. 3.1 a gradient type algorithm due to its simplicity and scalability. Later, in Sect. 3.2 we will show that when $\alpha_i + \beta_i \leq 1$, we can obtain practical convex relaxation of the original problem.

3.1 Gradient Method

First, it is tempting to use a gradient method to solve the problem in (7). However, note that w^* is nonsmooth and possibly nonconvex since $\hat{s}_i^r(\boldsymbol{\phi})$ is nonsmooth and nonconvex. Thus, directly solving this problem using gradient methods is known to be difficult. As a result, we will use [P0], which is of higher dimension than (7) but smooth with simple constraints and hence is easier to apply gradient methods to.

To this end, we show how to compute gradient $\nabla F(\mathbf{s})$ efficiently. Note that

$$\nabla F(\mathbf{s}) = \nabla w(\mathbf{s}) + \nabla g(\mathbf{p}(\boldsymbol{\phi}(\mathbf{s}))) = \mathbf{1} + J_{\mathbf{p}}(\mathbf{s})^{\mathrm{T}}\mathbf{c}, \tag{8}$$

where $J_{\mathbf{p}}(\mathbf{s}) = \left[\partial p_i(\mathbf{s})/\partial s_j\right]$ is the Jacobian matrix. Thus, the bulk of computation lies in evaluating $J_{\mathbf{p}}(\mathbf{s})$. By applying the chain rule, we obtain

$$J_{\mathbf{p}}(\mathbf{s}) = J_{\mathbf{p}}(\boldsymbol{\phi}) J_{\boldsymbol{\phi}}(\mathbf{s}). \tag{9}$$

Here, $J_{\boldsymbol{\phi}}(\mathbf{s})$ can be evaluated easily from (5a, 5b), i.e.,

$$J_{\boldsymbol{\phi}}(\mathbf{s}) = \left[\mathtt{diag}\left(\tfrac{\boldsymbol{\phi}\circ\boldsymbol{\alpha}\circ\boldsymbol{\kappa}}{\mathbf{1}+\boldsymbol{\kappa}\circ\mathbf{s}^{\rho}}\right), \mathtt{diag}\left(\tfrac{\boldsymbol{\phi}\circ\boldsymbol{\beta}\circ\boldsymbol{\zeta}}{\mathbf{1}+\boldsymbol{\zeta}\circ\mathbf{s}^{r}}\right)\right]. \tag{10}$$

The matrix $J_{\mathbf{p}}(\boldsymbol{\phi})$ can be computed by totally differentiating (5a) with respect to $\boldsymbol{\phi}$; this is similar to the approach of [13]. In fact,

$$M(\boldsymbol{\phi}) J_{\mathbf{p}}(\boldsymbol{\phi}) = -\mathtt{diag}(\boldsymbol{\theta} \circ \bar{\mathbf{p}}(\mathbf{s})) \tag{11}$$

where $M(\boldsymbol{\phi}) = \mathtt{diag}(\boldsymbol{\theta} \circ \boldsymbol{\phi} + \bar{\boldsymbol{\lambda}} + B\bar{\mathbf{p}}(\mathbf{s})) - \mathtt{diag}(\mathbf{1} - \bar{\mathbf{p}}(\mathbf{s}))B$, and $\bar{\mathbf{p}}(\mathbf{s}) = \hat{\mathbf{p}}(\boldsymbol{\phi}(\mathbf{s}))$, which can be computed efficiently using the fixed point iteration in Theorem 2. We can show that $M(\boldsymbol{\phi})$ is a nonsingular M-matrix. Thus, from (11) we get

$$J_{\mathbf{p}}(\boldsymbol{\phi}) = -M(\boldsymbol{\phi})^{-1}\mathtt{diag}(\boldsymbol{\theta} \circ \bar{\mathbf{p}}(\mathbf{s})). \tag{12}$$

Substituting (10) and (12) in (9) and using it in (8), we obtain

$$\nabla F(\mathbf{s}) = \mathbf{1} - J_{\boldsymbol{\phi}}(\mathbf{s})^{\mathrm{T}}\mathtt{diag}(\boldsymbol{\theta} \circ \bar{\mathbf{p}}(\mathbf{s}))\mathbf{z}, \quad \text{where } \mathbf{z} := M(\boldsymbol{\phi})^{-\mathrm{T}}\mathbf{c}.$$

As a result, we can now use a projected gradient method for solving [P0]; see [13] for the detailed algorithm as well as an efficient and scalable approach for computing $\mathbf{z}$ without matrix inversions for large systems (by exploiting the structure of $M(\boldsymbol{\phi})$ and using the power method).

3.2 Convex Relaxation

We will introduce relaxations to the constraints in (5). First, to relax (5a), we use the same approach used in [13]. Specifically, define

$$\mathbf{y} := -\ln \mathbf{p},\ \ \mathbf{t} := \bar{\boldsymbol{\lambda}} \circ e^{\mathbf{y}},\ \ U := \mathtt{diag}(e^{\mathbf{y}}) B \mathtt{diag}(e^{-\mathbf{y}}). \tag{13}$$

Using these new variables, (5a) can be rewritten as

$$\mathbf{t} + U\mathbf{1} = \bar{\boldsymbol{\lambda}} + B\mathbf{p} + \boldsymbol{\theta} \circ \boldsymbol{\phi}, \tag{14}$$

which is linear in the variables $\mathbf{t}$, $\mathbf{p}$, $\boldsymbol{\phi}$ and U. Next, we relax the nonconvex equality constraints in (13) with the following convex inequality constraints:

$$e^{-\mathbf{y}} \le \mathbf{p} \le \mathbf{1}, \quad \bar{\boldsymbol{\lambda}} \circ e^{\mathbf{y}} \le \mathbf{t}, \quad \mathtt{diag}(e^{\mathbf{y}}) B \mathtt{diag}(e^{-\mathbf{y}}) \le U \tag{15}$$

We can express these inequality constraints as a following set of at most $2N + |\mathcal{E}|$ exponential cone constraints:

$$(p_i, 1, -y_i) \in \mathcal{K}_{\exp} \text{ for all } i \in \mathcal{A} \tag{16a}$$

$$(t_i, 1, y_i + \log \bar{\lambda}_i) \in \mathcal{K}_{\exp} \text{ for all } i \in \mathcal{A}_R \tag{16b}$$

$$(u_{ij}, 1, y_i - y_j + \log b_{ij}) \in \mathcal{K}_{\exp} \text{ for all } (i, j) \in \mathcal{E}, \tag{16c}$$

where $\mathcal{K}_{\exp} := \mathrm{cl}(\{(x_1, x_2, x_3) \mid x_1 \ge x_2 e^{x_3/x_2}, x_2 > 0\})$. These constraints in (16) (as well as those in (19) below) can be handled efficiently by conic optimization solvers, e.g., MOSEK [15].[1]

We now consider the constraint (5b). Note that since we aim to minimize the investment costs from $\mathbf{s}^p$ and $\mathbf{s}^r$ and the right-hand side of (5b) is strictly increasing in each element, at an optimal point the constraint will be active, allowing us to replace the equality with the inequality, i.e.,

$$\boldsymbol{\phi} \le (\mathbf{1} + \boldsymbol{\kappa} \circ \mathbf{s}^p)^{\boldsymbol{\alpha}} \circ (\mathbf{1} + \boldsymbol{\zeta} \circ \mathbf{s}^r)^{\boldsymbol{\beta}}. \tag{17}$$

In general this is not a convex constraint because of the product term on the right-hand side; however, it can be recast as convex constraints when $\boldsymbol{\alpha} + \boldsymbol{\beta} \le \mathbf{1}$. To see

[1] Any mention of commercial products is for information only; it does not imply a recommendation or endorsement by NIST.

this, let us first use a change of variable to rewrite the right-hand side of (17): for each $i \in \mathcal{A}$, define

$$\eta_i = 1 + \kappa_i s_i^p \quad \text{and} \quad \vartheta_i = 1 + \zeta_i s_i^r. \tag{18}$$

From their relations, we have $s_i^p(\eta_i) := (\eta_i - 1)/\kappa_i$ and $s_i^r(\vartheta_i) := (\vartheta_i - 1)/\zeta_i$. Note that η_i and ϑ_i are linear in s_i^p and s_i^r, respectively, and vice versa. With a little abuse of notation, we denote $(s_i^p(\eta_i) : i \in \mathcal{A})$ and $(s_i^r(\vartheta_i) : i \in \mathcal{A})$ by $\mathbf{s}^p(\boldsymbol{\eta})$ and $\mathbf{s}^r(\boldsymbol{\vartheta})$, respectively. In order to rewrite constraint (17) as conic constraints, we need the following assumption.

Assumption 2 We assume $\boldsymbol{\alpha} + \boldsymbol{\beta} \leq \mathbf{1}$.

This assumption implies that the (marginal) rates of both the increase in recovery rates and the decrease in failure rates slow down relatively quickly with increasing investments. In other words, the available tools are not very effective and, as a result, the failure rates do not diminish quickly and the recovery rates do not improve rapidly with increasing investments in resilience and recovery, respectively. Under Assumption 2, we can express the constraint in (17) as the following conic constraints:

$$\begin{cases} (\eta_i, \vartheta_i, \phi_i) \in \mathcal{P}_3^{\alpha_i, 1-\alpha_i} & \text{if } \alpha_i + \beta_i = 1, \\ (\eta_i, \vartheta_i, 1, \phi_i) \in \mathcal{P}_4^{\alpha_i, \beta_i, 1-\alpha_i-\beta_i} & \text{if } \alpha_i + \beta_i < 1, \end{cases} \quad i \in \mathcal{A}, \tag{19}$$

where $\mathcal{P}_n^{a_1,\ldots,a_m} = \left\{ \mathbf{x} \in \mathbb{R}^n \;\middle|\; \prod_{i=1}^m x_i^{a_i} \geq \sqrt{\sum_{j=m+1}^n x_j^2},\ x_1, \ldots, x_m \geq 0 \right\}$, $m < n$, is an n-dimensional power cone ($n \geq 3$), which is convex. Clearly, the above power cone constraints require Assumption 2. When this assumption does not hold, one must resort to other techniques to obtain a convex relaxation of (17).

Based on these new constraints (14)–(19), we obtain the following convex relaxation of [P1]: define $\tilde{w}(\boldsymbol{\eta}, \boldsymbol{\vartheta}) := w(\mathbf{s}^p(\boldsymbol{\eta}), \mathbf{s}^r(\boldsymbol{\vartheta})) = \sum_{i=1}^N (s_i^p(\eta_i) + s_i^r(\vartheta_i))$.

$$\begin{aligned} [\mathbf{CR}] \qquad \min_{\mathbf{p},\mathbf{t},\mathbf{y},\boldsymbol{\eta},\boldsymbol{\vartheta},\boldsymbol{\phi},U} \quad & \tilde{w}(\boldsymbol{\eta}, \boldsymbol{\vartheta}) + \mathbf{c}^{\mathrm{T}}\mathbf{p} \\ \text{s.t.} \quad & (14),\ (16),\ (19), \\ & \mathbf{p} \in (0, 1]^N,\ \boldsymbol{\eta} \geq \mathbf{1},\ \boldsymbol{\vartheta} \geq \mathbf{1},\ \mathbf{y} \geq \mathbf{0} \end{aligned}$$

Suppose that $\mathbf{x}_{\mathrm{R}}^+ := (\mathbf{p}^+, \mathbf{t}^+, \mathbf{y}^+, \boldsymbol{\eta}^+, \boldsymbol{\vartheta}^+, \boldsymbol{\phi}^+, U^+)$ is an optimal point of [CR], and let $\mathbf{s}^+ = (\mathbf{s}^p(\boldsymbol{\eta}^+), \mathbf{s}^r(\boldsymbol{\vartheta}^+))$. Define

$$\mathbf{p}' = e^{-\mathbf{y}^+},\ \boldsymbol{\phi}' = \boldsymbol{\phi}^+ + \mathtt{diag}(\boldsymbol{\theta}^{-1}) B(\mathbf{p}^+ - \mathbf{p}'),\ \text{and } \mathbf{s}' = \hat{\mathbf{s}}(\boldsymbol{\phi}'), \tag{20}$$

where $\hat{\mathbf{s}}(\boldsymbol{\phi}')$ is an optimal point of [SP2] for $\boldsymbol{\phi}'$, which was given in Theorem 3. The above relaxation provides both an upper bound and a lower bound on the optimal cost as shown below.

Theorem 4 *Suppose that f^* is the optimal value of [P1]. Then, $(\mathbf{s}', \mathbf{p}')$ is a feasible point of [P1], and we have*

$$f(\mathbf{s}^+, \mathbf{p}^+) \leq f^* \leq f(\mathbf{s}', \mathbf{p}'). \tag{21}$$

Moreover, the last two constraints of (15) are active at $\mathbf{x}^+$, i.e.,

$$\mathbf{t}^+ = \bar{\boldsymbol{\lambda}} \circ e^{\mathbf{y}^+} \text{ and } U^+ = \mathtt{diag}(e^{\mathbf{y}^+}) B \mathtt{diag}(e^{-\mathbf{y}^+}). \tag{22}$$

This result shows that the tightness of our relaxation can be judged via the gap $f(\mathbf{s}', \mathbf{p}') - f(\mathbf{s}^+, \mathbf{p}^+)$. Note that in view of Theorem 4, $w(\mathbf{s})$ can be expressed as $w(\hat{\mathbf{s}}(\boldsymbol{\phi})) =: \hat{w}(\boldsymbol{\phi})$, which is a convex function of $\boldsymbol{\phi}$ under Assumption 2. Thus,

$$\begin{aligned} f(\mathbf{s}', \mathbf{p}') - f(\mathbf{s}^+, \mathbf{p}^+) &= \hat{w}(\boldsymbol{\phi}') - \hat{w}(\boldsymbol{\phi}^+) + \mathbf{c}^{\mathrm{T}}(\mathbf{p}' - \mathbf{p}^+) \\ &\leq \nabla \hat{w}(\boldsymbol{\phi}')^{\mathrm{T}}(\boldsymbol{\phi}' - \boldsymbol{\phi}^+) + \mathbf{c}^{\mathrm{T}}(\mathbf{p}' - \mathbf{p}^+) && \text{(convexity of } \hat{w}) \\ &= \left(\mathbf{c} - B^{\mathrm{T}} \nabla \hat{w}(\boldsymbol{\phi}')\right)^{\mathrm{T}}(\mathbf{p}' - \mathbf{p}^+), && \text{(from (20))} \end{aligned}$$

where $\nabla \hat{w}(\boldsymbol{\phi}')$ is the gradient of $\hat{w}$ at $\boldsymbol{\phi}'$ (or any subgradient if non-differentiable). Since $\mathbf{p}' \leq \mathbf{p}^+$, the above bound suggests that the gap $f(\mathbf{s}', \mathbf{p}') - f(\mathbf{s}^+, \mathbf{p}^+)$ is likely to be small when the failure cost vector $\mathbf{c}$ is sufficiently large (which is often the case in practice). Clearly, this gap is 0 when $\mathbf{c} \geq B^{\mathrm{T}} \nabla \hat{w}(\boldsymbol{\phi}')$, i.e., $(\mathbf{s}', \mathbf{p}')$ is a global optimal solution to the original problem; this is the case, for example, when $\boldsymbol{\alpha} + \boldsymbol{\beta} = \mathbf{1}$ and $\mathbf{c}$ is sufficiently large (independent of $\boldsymbol{\phi}'$). Moreover, when the convex relaxation is not tight, we can obtain an improved solution using the gradient method in Sect. 3.1 with $(\mathbf{s}', \mathbf{p}')$ as a starting point.

Finally, we note that when either $(\alpha_i, \beta_i) = (0, 1)$ or $(\alpha_i, \beta_i) = (1, 0)$ for all $i \in \mathcal{A}$, both (5b) and (17) are simple affine constraints and can be used directly in our relaxed problem without the need to transform them into conic constraints provided in (19); a similar approach for the special case with $(\boldsymbol{\alpha}, \boldsymbol{\beta}) = (\mathbf{0}, \mathbf{1})$ can be found in [13].

4 Numerical Results

In this section, we provide some numerical results to demonstrate the usefulness and efficacy of our proposed approaches. Our studies are carried out in MATLAB (version 9.5) on a laptop with 8GB RAM and a 2.4 GHz Intel Core i5 processor. For our numerical studies, we generate a set of strongly connected scale-free networks with the power law parameter for node degrees set to 1.5, and the minimum and maximum node degrees equal to 2 and $\lceil 3 \log N \rceil$, respectively, in order to ensure network connectivity with high probability.

For all considered networks, we fix $\theta_i = 1$ and $\alpha_i = \beta_i = 0.5$ for all $i \in \mathcal{A}$. The failure transmission rates $b_{j,i}$, $(j, i) \in \mathcal{E}$, and the parameters κ_i and ζ_i are selected

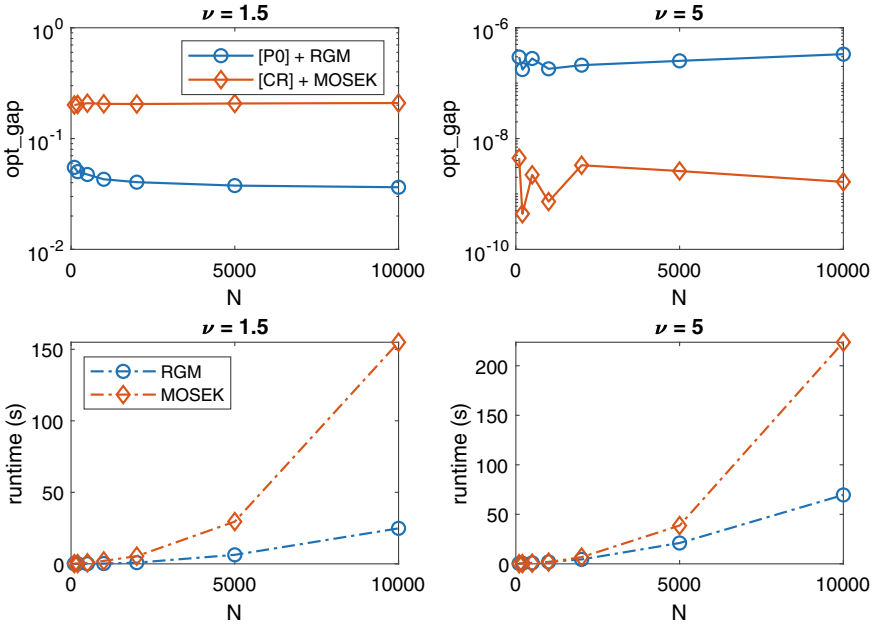


Fig. 1 Comparison between RGM and MOSEK package in two cases $\nu = 1.5$ and $\nu = 5$; here ν is the parameter associated with the cost vector $\mathbf{c} = \nu B^{\mathrm{T}}\mathbf{1}$

uniformly at random in [0.01, 1], [1, 1.5], and [0.5, 1], respectively. We choose failure cost vector $\mathbf{c} = \nu B^{\mathrm{T}}\mathbf{1}$ with a varying parameter $\nu > 0$ to reflect an observation that nodes that support more neighbors should, on the average, cause larger economic losses. In each considered network, we assign positive failure rates of $\bar{\lambda}_i = 0.1$ to 20% of the nodes (sampled without replacement).

We solve the relaxed problem [CR] using MOSEK package and define the relative optimality gap at $(\mathbf{s}', \mathbf{p}')$ as $\mathsf{opt\_gap} = \left|1 - \frac{f(\mathbf{s}^+, \mathbf{p}^+)}{f(\mathbf{s}', \mathbf{p}')}\right|$. We use the Reduced Gradient Method (RGM) in [13] to find a local optimizer $\tilde{\mathbf{s}}$ of [P0] with initial point $\mathbf{s}^{(0)} = \mathbf{0}$. This gives us $F(\tilde{\mathbf{s}})$, which is an upper bound on the optimal cost, and the relative optimality gap for RGM's solution is given by $\mathsf{opt\_gap} = \left|1 - \frac{f(\mathbf{s}^+, \mathbf{p}^+)}{F(\tilde{\mathbf{s}})}\right|$. The results (averaged over 5 runs) are shown in Fig. 1.

As we can see, when ν is small with smaller failure costs, the relaxation is not exact, and the RGM yields a solution with a smaller optimality gap (less than 5.5% for $\nu = 1.5$). When ν is large, the relaxation becomes tight, and both approaches provide (nearly) optimal solutions. In this case, MOSEK yields a slightly better solution since it uses an interior-point method. In terms of runtimes, RGM outperforms MOSEK, especially for large networks. The above results suggest that RGM not only is scalable, but also can find a good solution to the original problem, if not (nearly) optimal.

5 Conclusion

We studied the problem of determining suitable investments in improving the robustness of complex systems. Unlike in previous studies, we considered investments in both resilience and recovery, while taking into account dynamics. The problem of minimizing the average costs of a system operator is formulated as an optimization problem, which is shown to be nonconvex. We then proposed two approaches to determining (nearly optimal) investments based on a gradient-based method and a convex relaxation. The effectiveness of the proposed approaches are demonstrated using numerical studies.

References

1. Hota, A.R., Sundaram, S.: Interdependent security games on networks under behavioral probability weighting. IEEE Control Netw. Syst. **5**(1), 262–273 (2018)
2. Khalili, M.M., Zhang, X., Liu, M.: Incentivizing effort in interdependent security games using resource pooling. In: Proceedings of NetEcon (2019)
3. La, R.J.: Interdependent security with strategic agents and global cascades. IEEE/ACM Trans. Netw. **24**(3), 1378–1391 (2016)
4. Lelarge, M., Bolot, J.: A local mean field analysis of security investments in networks. In: Processing of International Workshop on Economics of Networked Systems, pp. 25–30 (2008)
5. Cohen, R., Havlin, S., Ben-Avraham, D.: Efficient immunization strategies for computer networks and populations. Phys. Rev. Lett. **91**(247901), (Dec 2003)
6. Borgs, C., Chayes, J., Ganesh, A., Saberi, A.: How to distributed antidote to control epidemics. Random Struct. Algorithms **37**(2), 204–222 (Sept 2010)
7. Mai, V.S., Battou, A., Mills, K.: Distributed algorithm for suppressing epidemic spread in networks. IEEE Contr. Syst. Lett. **2**(3), 555–560 (2018)
8. Ottaviano, S., De Pellegrini, F., Bonaccorsi, S., Van Mieghem, P.: Optimal curing policy for epidemic spreading over a community network with heterogeneous population. J. Complex Networks **6**(6), (Oct 2018)
9. Nowzari, C., Preciado, V.M., Pappas, G.J.: Optimal resource allocation for control of networked epidemic models. IEEE Control Netw. Syst. **4**(2):159–169 (June 2017)
10. Preciado, V.M., Zargham, M., Enyioha, C., Jadbabaie, A., Pappas, G.J.: Optimal resource allocation for network protection against spreading processes. IEEE Control Netw. Syst. **1**(1), 99–108 (2014)
11. Kunreuther, H., Heal, G.: Interdependent security. J. Risk Uncertainty **26**(2/3), 231–249 (2003)
12. Mai, V.-S., La, R.J., Battou, A.: Optimal cybersecurity investments for SIS model. In: Proceeding of IEEE Globecom (2020)
13. Mai, V.-S., La, R.J., Battou, A.: Optimal cybersecurity investments in large networks using SIS model: algorithm design. IEEE/ACM Trans. Netw. **29**(6), 2453–2466 (2021)
14. Khanafer, A., Başar, T., Gharesifard, B.: Stability of epidemic models over directed graphs: a positive systems approach. Automatica **74**, 126–134 (2016)
15. MOSEK ApS.: The MOSEK optim. toolbox for MATLAB manual. Version 9.0. (2019). http://docs.mosek.com/9.0/toolbox/index.html

Incremental Computation of Effective Graph Resistance for Improving Robustness of Complex Networks: A Comparative Study

Clara Pizzuti and Annalisa Socievole

Abstract Real-world infrastructures are often subject to random failures or intentional attacks that can significantly impact their robustness and hence many processes taking place on them. In this paper, we focus on the robustness of complex networks by proposing a link addition strategy for improving the network robustness. The approach exploits the incremental computation of the Moore-Penrose pseudoinverse matrix to efficiently compute the effective graph resistance when a new link is added to the network. Experiments on both real-world and synthetic data sets show that the strategy provides a good trade-off between a low percentage error of effective graph resistance with respect to the exhaustive search and the simulation time needed to obtain the optimal link.

Keywords Network robustness · Graph spectra · Effective resistance · Moore-Penrose pseudoinverse · Genetic algorithm

1 Introduction

During the last decade, the pervasiveness of networks as a representation model of real-world organizations has gained a central role in the comprehension and control of such systems. The Internet, electric power grid, airline, railway transportation systems, and communication systems in general consist of objects interconnected among them. The daily use of such infrastructures by societies, individuals, and organizations worldwide has stimulated the study of the capability of networks to react to failures. The correct functioning of the network in case of either natural or intentional malfunctioning is a main problem because of the side effects a disruption could have in maintaining services.

C. Pizzuti (✉) · A. Socievole
National Research Council of Italy (CNR), Institute for High Performance Computing and Networking (ICAR), Via Pietro Bucci, 8-9C,87036 Rende, CS, Italy
e-mail: clara.pizzuti@icar.cnr.it

A. Socievole
e-mail: annalisa.socievole@icar.cnr.it

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_33

Though network robustness has been studied in different contexts, a shared definition is that it is a measure of the capability of the network of maintaining good performances after an attack [1, 2]. Several robustness measures have been proposed in the last years. Freitas et al. [5] classify robustness measures into three categories: measures based on *graph connectivity*, such as diameter or average distance, measures based on *adjacency matrix spectrum*, such as spectral radius or gap, and those based on *Laplacian matrix spectrum*. In this last category, *algebraic connectivity* and *effective graph resistance* are two popular measures extensively studied in the literature. The algebraic connectivity has been proposed by Fielder [4] and it is defined as the second smallest eigenvalue of the Laplacian matrix of a graph. Fielder showed that the larger the algebraic connectivity, the more difficult to disconnect the graph. The *effective graph resistance* R_G of a graph G, proposed by Ellens et al. [3], is a measure that views a graph as an electric circuit. R_G is related to the algebraic connectivity, which provides upper and lower bounds to R_G, and to random walks. In fact, R_G is proportional to the time expected to reach any vertex j starting from a vertex i, averaged over all the pairs of nodes. These connections, and the fact that R_G decreases when edges are added to G, make the effective graph resistance a good measure to evaluate the robustness of a network.

In [9] a method based on Genetic Algorithms [7], named *RobGA*, which finds the edge that maximally improves the effective graph resistance has been proposed. The main drawback of the approach is that the effective graph resistance must be recomputed every time an edge is chosen as a candidate solution. Thus, the computing time when large values of population size are used can be rather high.

In this paper, we propose to modify *RobGA* by introducing an incremental computation of R_G each time a new edge is added to G. The modified approach, named *RobGA*$\{L^+\}$, is compared with *RobGA* and other four strategies of link addition, proposed by Wang et al. [12]. Experiments on both real-world and synthetic data sets show that *RobGA*$\{L^+\}$ provides a good trade-off between a low percentage error of effective graph resistance with respect to the exhaustive search and low simulation time.

The paper is organized as follows. In the next section, the concept of effective graph resistance is recalled and the problem we tackle in the paper is defined. In Sect. 3, the *RobGA* method is briefly described and the efficient computation of R_G is introduced. Section 4 describes the strategies used to choose a link. In Sect. 5, the methods are executed on both real-world and synthetic networks, and a comparison of the results obtained by applying each strategy is reported. Finally, Sect. 6 concludes the paper and discusses the advantages and disadvantages of the new proposed methodology.

2 Effective Graph Resistance

Let $G = (V, E)$ be an undirected and connected graph without self-loops modeling a network, where V is the set of n nodes constituting the network, and E the set of m links connecting pair nodes of V.

The adjacency matrix A of G is an $n \times n$ symmetric matrix with elements $a_{ij} = 1$ if there is an edge between nodes i and j, 0 otherwise. Let $\Delta = diag(d_i)$ be the $n \times n$ diagonal degree matrix, where $d_i = \sum_{j=1}^{n} a_{ij}$, the $Laplacian\ L$ of G is defined as the $n \times n$ symmetric matrix $L = \Delta - A$. Thus $L_{ij} = d_i$ if $i = j$, $L_{ij} = -1$ if $(i, j) \in E$, and $L_{ij} = 0$ otherwise.

Δ and A are symmetric and have real eigenvalues, thus the Laplacian L is positive semi-definite, with nonnegative eigenvalues, the smallest eigenvalue being $\lambda_1 = 0$, the remaining eigenvalues all positive and can be ordered as $0 = \lambda_1 \leq \lambda_2 \leq \ldots \lambda_n$. The set of eigenvalues $\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ is called the spectrum of L. The second smallest eigenvalue λ_2 is called the *algebraic connectivity*, known also as the Fielder vector [4]. Since any Laplacian L has a zero eigenvalue, its rank is at most $n - 1$, thus the inverse matrix does not exist. However, the *Moore-Penrose pseudoinverse* of L, denoted as L^+, can act as the inverse matrix. In particular, for any connected graph, the eigen-decomposition of the pseudoinverse L^+ has the same set of orthogonal eigenvectors of L and the eigenvalues of L^+ are the reciprocal of the positive eigenvalues of L , i.e. $\lambda_1^+ = 0$, and $\lambda_i^+ = \frac{1}{\lambda_i}$, $i = 2, \ldots, n$.

The *effective graph resistance* R_G is a measure derived from the field of electric circuit analysis [3] where an edge e_{ij} with weight w_{ij} corresponds to an electrical resistance $\omega_{ij} = w_{ij}^{-1}$ Ohm. The *effective resistance* R_{ij} between two nodes i and j is the electrical resistance measured between i and j.

Ranjan et al. [10] showed that R_{ij} can be expressed in terms of the elements of L^+ as:

$$R_{ij} = l_{ii}^+ + l_{jj}^+ - l_{ij}^+ - l_{ji}^+ \tag{1}$$

Given R_{ij}, *the effective graph resistance* is defined as the sum of the effective resistances between all possible pairs of nodes in the graph G and has been shown by Klein and Randic [8] that it can be computed using the inverse non-zero eigenvalues of L as

$$R_G = n \sum_{k=2}^{n} \frac{1}{\lambda_k} \tag{2}$$

As robustness measure, the effective graph resistance is deemed a good measure by several studies. Ghosh et al. [6] showed that it measures the closeness of two nodes and how well a graph is connected. Another interesting property is that its value decreases when edges are added to a graph [3]. This agrees with the intuitive concept of robustness that complete graphs are more robust than trees or path graphs. There exists also an analogy between random walks and effective graph resistance [3, 11]. The smaller the resistance between two nodes i and j, the smaller the expected duration time of a random walk from i to j and back. Short random walks and the existence of many paths between nodes are an indication of robustness since node or edge failures can be easily recovered by another path. A smaller value of R_G means that the network is more robust.

The effective graph resistance strictly decreases when a link is added to the network and increases when a link is removed [12]. Thus, a strategy to improve robustness can be the addition of a link that diminishes the effective graph resistance. This strategy, as outlined in [12], is appropriate when new infrastructural connections are planned to increase network efficiency. A challenging task is to detect the link, among all the possible ones, whose addition maximally decreases the effective graph resistance [12].

In this paper we consider the problem of enhancing the robustness of a network by adding a link. Let $G + \{e_{ij}\}$ denote the graph G when the link $e_{ij} = (i, j)$ is added to G.

Problem (link addition): given an undirected graph $G = (V, E)$ with n nodes and m links, let $E_c = \{(i, j) \mid (i, j) \notin E\}$ be the set of m_c possible new links. Find the link $e_{ij} = (i, j) \in E_c$ such that

$$R_{G+\{e_{ij}\}} \leq R_{G+\{e_{kl}\}} \tag{3}$$

for any other edge $e_{kl} = (k, l) \in E_c$.

Several strategies of edge addition have been proposed and evaluated to improve R_G and thus make a network more robust [12]. In [9] the Genetic Algorithm *RobGA* that optimizes the effective graph resistance has been proposed and compared with the heuristics evaluated in [12]. The approach showed very good performance with respect to such heuristics. However, the computation of the effective graph resistance every time the link to add to the network is chosen is computationally expensive. In the following, we propose a variation to *RobGA*, named *RobGA*$\{L^+\}$, that exploits an approximate computation of R_G when a new link is added to G leveraging on the approach proposed by Ranjan et al. [10] for the incremental computation of the pseudoinverse Laplacian matrix.

3 *RobGA*$\{L^+\}$: Incremental Computation of the Effective Graph Resistance

In this section, the algorithm *RobGA* is first summarized, and then the incremental computation of the effective graph resistance when a link is added to G is described.

RobGA is a Genetic Algorithm using a population of individuals consisting of a vector of 2 genes corresponding to an edge $(i, j) \notin E$, i.e. an edge which has been chosen to be added to the graph G and does not already exist between the nodes i and j. The crossover operator, for each individual I, generates a child e_3 from two parents $e_1 = (i, j)$ and $e_2 = (k, l)$, by combining e_1 and e_2 such that the edge obtained by combining the four nodes $\{i, j, k, l\}$ does not exist. Finally, the mutation operator, given an individual (i, j), disconnects i from j and connects it to another node k chosen at random among the nodes which are not its neighbors. *RobGA* initializes the population at random with not already existing edges. Then,

for a number of generations, for each individual e of the population, the Laplacian of $G + \{e\}$ is computed and the fitness function $R_{G+\{e\}}$ is evaluated. A new population of individuals is created by applying the variation operators. Finally, the individual e minimizing $R_{G+\{e\}}$ is returned as solution to the problem.

In [9] it is highlighted that the computational complexity of the algorithm is dominated by the fitness computation which is performed $T \times P$ times, where T is the number of generations and P the population size. R_G needs the computation of the eigenvalues of the adjacency matrix of G, which is of the order $O(n^3)$, thus the complexity of *RobGA* is $O(T \times P \times n^3)$, i.e. $O(n^3)$. It is worth pointing out that, when running the method, the constant $T \times P$ sensibly increases the computing time when high values of population size are used, which is often the case for large networks in order to obtain good results. Thus avoiding to recompute R_G from scratch could be a good objective to pursue.

A solution to this problem can be obtained by exploiting the equation (1) and the approximate computation of the pseudoinverse L^+ proposed by Ranjan et al. [10]. These authors showed how to compute the pseudoinverse of the Laplacian of the graph $G + \{e_{ij}\}$ when a new edge $e_{ij} \notin E$ is added to G (Theorem 3 in [10]). Let l^+_{ij} and $l^{+(1)}_{ij}$ denote the elements of the Moore-Penrose pseudoinverse of Laplacians L^+ and $L^{+(1)}$ of G and the updated $G + \{e_{ij}\}$ when en edge is added to G, respectively. Then

$$l^{+(1)}_{ij} = l^+_{ij} - \frac{(l^+_{ji} - l^+_{ij})(l^+_{ij} - l^+_{ji})}{\omega_{ij} + R_{ij}} \tag{4}$$

where R_{ij} is the effective resistance of nodes i and j in the graph G.

4 Strategies for Link Addition

This section describes a set of strategies for selecting a link whose addition would minimize the effective graph resistance. We also consider the *exhaustive search* for comparison, which is the best strategy since it checks all the possible links and computing the corresponding effective graph resistance is able to find the link optimizing it. Its computational cost, however, is high having a complexity $O(n^5)$. Alternative and lower-cost strategies that are able to determine the link to remove have been proposed by Wu et al. [12]. In the following, we describe these strategies and use them for performance comparison.

- S_1 (**semi-random**): the link $e(i, j) \in E_c$ has i with the minimum degree and j is randomly chosen. The complexity of the strategy is $O(n^2 - n + m_c + 1)$, where $O(n(n-1))$ is required for computing the degree of all the nodes, $O(m_c)$ for finding the node having the lowest degree and $O(1)$ for choosing a random node.
- S_2 (**degree product**): the link $e(i, j)$ is chosen among the links in E_c having the minimum degree product $d_i d_j$. This strategy has complexity $O(n^2 - n + 2m_c)$,

where $O(n(n-1))$ is for computing the degree of all the nodes, $O(m_c)$ for computing $d_i d_j$ and $O(m_c)$ for finding the minimum $d_i d_j$ product.

- S_3 (**Fiedler vector**): the link $e(i, j)$ is chosen using the Fiedler vector y corresponding to the eigenvector of the second smallest eigenvalue of the Laplacian matrix of G. Nodes i and j are chosen as the couple having the maximum difference $|y_i - y_j|$, where y_i and y_j are the components i and j of the Fiedler vector, respectively. The complexity of S_3 is $O(n^3 + 2m_c)$, where $O(n^3)$ is for computing the Fiedler vector, $O(m_c)$ for computing $|y_i - y_j|$ for the m_c possible new node pairs and $O(m_c)$ for finding the maximum of the difference $|y_i - y_j|$.
- S_4 (**effective resistance**): the link $e(i, j)$ is chosen among the links in E having the highest effective resistance R_{ij}. The effective resistance R_{ij} between nodes i and j is computed as in equation (1). The complexity of this strategy is $O(n^3 + 4m_c)$, where $O(n^3)$ is for computing the Moore-Penrose pseudoinverse L^+, $O(3m_c)$ for computing R_{ij} for the m_c unconnected node pairs and $O(m_c)$ for finding the maximum value of effective resistance.

5 Experimental Evaluation

In this section, *RobGA* and *RobGA*$\{L^+\}$ are compared with the four strategies of link addition presented in [12]. The strategies have been implemented using MATLAB R2020a. We used the Genetic Algorithm solver implemented in the Global Optimization Toolbox for *RobGA* and *RobGA*$\{L^+\}$ by setting crossover fraction 0.9, mutation rate 0.2, maximum number of generations 300 and population size 100. In addition to the aforementioned four strategies, we also consider the *exhaustive search* in order to include the worst case scenario in which the search of the optimal link explores all the possible new links. In the following subsections, we describe the tested complex networks, the performance indexes and the obtained results.

5.1 Networks

The complex networks on which focuses this paper are both real-world and synthetically generated. Tables 1 and 2 summarize their features. Real-world networks are the following:

- **Internet Backbones**. From the Internet Topology Zoo[1] repository, we selected 5 representative networks, *Bell South*, *ASNET-AM*, *ITC Deltacom*, *ION*, and *US Carrier*. Each of these networks has a backbone topology where each node is a BGP (Border Gateway Protocol) router. An attack as blackholing or traffic redirection to other destinations on such networks would severely degrade their performance.

[1] http://www.topology-zoo.org.

Table 1 Topological features of real-world networks: network acronym (ID), number of nodes (N), number of links (L), average degree ($< k >$), average clustering coefficient ($< C >$), and density (D)

Network	ID	N	L	$< k >$	$< C >$	D
Bell South	BS	51	66	1.294	0.081	0.052
ASNET-AM	AA	65	77	1.184	0.063	0.037
ITC Deltacom	ITC	113	161	1.425	0.053	0.025
ION	ION	125	146	1.168	0.006	0.019
US Carrier	USC	158	189	1.196	0.002	0.015
Ego 3980	3980	44	138	3.136	0.227	0.072
Ego 686	686	168	1656	9.8572	0.266	0.059
Ego 3437	3437	532	4812	9.045	0.272	0.017
US Power Grid	USPG	4941	6594	2.669	0.103	5.403e−04

Table 2 Topological features of synthetic networks

Network type	Network ID	N	L	$< k >$	$< C >$	D
Erdős-Rényi	ER_128	128	627.3	5.23	0.054	0.041
	ER_256	256	1423.2	11.117	0.064	0.043
	ER_512	512	3240.2	12.64	0.01	0.024
	ER_1024	1024	6310.2	12.222	0.004	0.011
Watts-Strogatz	WS_128	128	384	6	0.109	0.047
	WS_256	256	768	6	0.104	0.023
	WS_512	512	2560	10	0.036	0.019
	WS_1024	1024	5120	10	0.039	0.009
Bárabasi-Albert	BA_128	128	253.4	3.954	0.132	0.031
	BA_256	256	510.2	3.984	0.129	0.015
	BA_512	512	1529.7	5.996	0.017	0.011
	BA_1024	1024	3065.2	5.986	0.012	0.005

– **Facebook Ego Networks**. We also considered 3 Facebook ego networks, *Ego 3980*, *Ego 686*, and *Ego 3437* where each node represents a Facebook user and a link between them a friendship. These social networks are often subject to fake news spreading and profile hacking, just to provide some examples. For each network, we do not consider the whole topology but only the *largest connected component* (LLC) since some nodes (8 nodes in *Ego 3980*, 2 nodes in *Ego 686* and *Ego 3437*) are disconnected from the main component.
– **US Power Grid**. We finally considered the Western States Power Grid[2], a larger network where transformers, substations and generators are the nodes, and the high-voltage transmission lines are the links. These networks are vulnerable to cascading failures and blackouts.

[2] http://konect.uni-koblenz.de/networks/opsahl-powergrid.

We also consider the following synthetic networks:

- **Erdős-Rényi**. These random networks are generated from a set of n nodes by randomly assigning a link between two nodes with a probability p_c, also called *link density*. When $p_c = 2\ln(n)/n$, the graph is connected. These synthetic networks are used to model peer-to-peer networks, ad-hoc networks and collaboration networks.
- **Watts-Strogatz**. These networks, also named *small-world* networks, are generated from a ring lattice of N nodes and k links for each node where each link is randomly rewired with probability p. Here we set $k = 6$ and $p = 0.5$ for the 128-nodes and 256-nodes networks, and $k = 10$ and $p = 0.5$ for the 512-nodes and 1024-nodes networks. These networks well model contact networks such as Bluetooth or Wi-Fi encounters networks.
- **Bárabasi-Albert**. These networks follow a power law degree distribution and are characterized by the *preferential attachment* feature for which nodes tend to link with high degree nodes. They are generated from n_0 initial nodes. Then, at every time step t, a new node with $n_k \leq n_0$ links is connected to the n_k existing nodes with a probability $p = d_i/2m_t$, where d_i is the degree of node i and m_t is the total number of links at time t. Here, we set $n_0 = 5$ and $n_k = 2$ when the network size is 128 and 256, and $n_0 = 10$ and $n_k = 3$ for the other sizes.

5.2 Performance Measures

To evaluate the improvement of our proposed strategy over S1, S2, S3, S4 and *RobGA*, we adopt the following measures built on the effective graph resistance and the simulation time, respectively.

- *Percentage Error*: the effective graph resistance percentage error between a link addition strategy and the exhaustive search of the optimal link to add, where $R^*_{G+\{e\}}$ denotes the best value obtained by the exhaustive search and $R^{S_x}_{G+\{e\}}$ is the effective graph resistance value obtained by the chosen link addition strategy S_x.

$$\Delta R_G = \left| \frac{R^{S_x}_{G+\{e\}} - R^*_{G+\{e\}}}{R^*_{G+\{e\}}} \right| * 100 \tag{5}$$

- *Speedup*: the ratio between the simulation time $t^{S_x}_{sim}$ needed by a link addition strategy S_x and the time $t^{RobGA\{L^+\}}_{sim}$ needed by $RobGA\{L^+\}$.

$$Speedup = \frac{t^{S_x}_{sim}}{t^{RobGA\{L^+\}}_{sim}} \tag{6}$$

If the speedup is greater than 1, $RobGA\{L^+\}$ is faster than the other strategy. More specifically, if the speedup factor is n, then $RobGA\{L^+\}$ has an $n\text{-}fold$ speedup.

5.3 Results

In a first experiment, we computed the effective graph resistance for the strategies S_1, S_2, S_3, S_4, *RobGA* and *RobGA*$\{L^+\}$. Then, we measured the percentage error ΔR_G for each strategy with respect to the exhaustive search, which tests all the possible missing links to find a candidate link $e(i, j)$ minimizing $R_{G+\{e\}}$. In particular, for the genetic algorithms *RobGA* and *RobGA*$\{L^+\}$ we considered 10 runs for each network and provide an average percentage error. We omit the results of the standard deviation of ΔR_G as it is very small. Table 3 illustrates the performance of the strategies for the real-world networks. On the Internet backbones, the percentage error of *RobGA* is the lowest, achieving even 0 on *Bell South*, *ASNET-AM* and *ION*. This last result indicates that *RobGA* finds the same link of the exhaustive search, as can be observed in Table 5 where we report the links added by the several strategies corresponding to the best ΔR_G value. On *Bell South*, for example, the link added by both the exhaustive search and *RobGA* is [43 47]. For *RobGA*$\{L^+\}$ we also report both the average ΔR_G value over the 10 networks runs and its minimum value. It is worth pointing out that *RobGA*$\{L^+\}$ matches the exhaustive search on the first 3 backbones achieving 0 as minimum ΔR_G. While on the other two networks, *ION* and *US Carrier*, *RobGA*$\{L^+\}$ is outperformed by *RobGA* which has the lowest percentage error, *RobGA*$\{L^+\}$ has significantly lower values of effective graph resistance if compared to strategies $S1$, $S2$, $S3$ and $S4$.

On the Facebook networks, the percentage error of *RobGA* is again the lowest. More specifically, on network 3980, the strategies $S4$, *RobGA* and *RobGA*$\{L^+\}$ by adding link [36 42] equal the performance of exhaustive search. The other strategies, $S1$, $S2$ and $S3$ have higher error values. The approximation of *RobGA*$\{L^+\}$ over network 686 works well, being the second best performing strategy and having a minimum error of 0.021 and an average value of 0.282. Finally, on the largest Facebook network (3437) *RobGA* has an average error of 0.077, followed by *RobGA*$\{L^+\}$ with no error in the best case corresponding to the addition of link [440 503], and with an error value of 0.419 which is the second best in the strategies ranking, followed by $S4$, $S3$, $S2$ and $S1$. Note that *RobGA*$\{L^+\}$ achieves a minimum error value of 0.021 by adding link [440 503] having in common node 440 with link [243 440] added by the exhaustive search.

For the US Power Grid, the number of possible links to add is $1.2 * 10^7$. Therefore, the strategies were evaluated by comparing only the effective graph resistance because of the high computational complexity of the exhaustive search. Also for $S4$ the computational time was too high and we do not consider it in the comparison. Table 4 presents the effective graph resistance values obtained on the US Power Grid. The leading strategies are again *RobGA* and *RobGA*$\{L^+\}$ with *RobGA* being the top performing among the other strategies (Table 5).

Table 6 illustrates the strategy comparison over synthetically generated Erdős-Rényi, Watts-Strogatz, and Bárabasi-Albert networks with 128 nodes. For each network, we generated 10 samples and run *RobGA*$\{L^+\}$ and *RobGA* 10 times. *RobGA*

Table 3 Comparison of percentage error ΔR_G between effective graph resistances in the augmented network for S_1, S_2, S_3, S_4 heuristics, *RobGA* and *RobGA*$\{L^+\}$ over real-world networks. For *RobGA*$\{L^+\}$, the minimum and the average $\Delta R_{G+\{e\}}$ values are reported

ID	$\Delta R^{S_1}_{G+\{e\}}$	$\Delta R^{S_2}_{G+\{e\}}$	$\Delta R^{S_3}_{G+\{e\}}$	$\Delta R^{S_4}_{G+\{e\}}$	$\Delta R^{RobGA}_{G+\{e\}}$	$\Delta R^{RobGA\{L^+\}}_{G+\{e\}}$
BS	8.918	4.493	0.76	0	0	{0, 0.401}
AA	7.472	6.97	1.914	2.565	0	{0, 1.152}
ITC	15.062	10.591	1.235	1.231	0.184	{0, 0.985}
ION	9.484	7.669	1.749	3.834	0	{0.255, 0.292}
USC	19.869	15.765	5.692	10.453	0.159	{0.396, 0.7}
3980	8.938	3.018	0.386	0	0	{0, 0.77}
686	7.26	0.847	0.435	0.826	0	{0.021, 0.282}
3437	2.186	1.567	0.635	0.522	0.077	{0, 0.419}

Table 4 Comparison of effective graph resistance in the original network (R_G) and in the augmented network resulting from the various strategies over USPG

ID	R_G	$R^*_{G+\{e\}}$	$R^{S_1}_{G+\{e\}}$	$R^{S_2}_{G+\{e\}}$	$R^{S_3}_{G+\{e\}}$	$R^{S_4}_{G+\{e\}}$	$R^{RobGA}_{G+\{e\}}$	$R^{RobGA\{L^+\}}_{G+\{e\}}$
USPG	6.377e+07	–	6.242e+07	6.314 e+07	6.173e+07	–	6.105e+07	6.107e+07

Table 5 Links added by the several strategies over real-world networks for the best ΔR_G value

ID	e^*	e^{S_1}	e^{S_2}	e^{S_3}	e^{S_4}	e^{RobGA}	$e^{RobGA\{L^+\}}$
BS	[43 47]	[32 25]	[14 24]	[3 6]	[3 6]	[43 47]	[43 47]
AA	[32 37]	[23 3]	[27 52]	[15 33]	[7 36]	[32 37]	[32 37]
ITC	[34 59]	[48 19]	[40 80]	[40 109]	[40 109]	[34 59]	[34 59]
ION	[4 55]	[30 20]	[7 72]	[5 55]	[54 103]	[4 55]	[104 55]
USC	[80 93]	[78 67]	[56 72]	[116 148]	[41 48]	[79 77]	[79 78]
3980	[36 42]	[37 5]	[4 39]	[23 42]	[36 42]	[36 42]	[36 42]
686	[26 62]	[140 145]	[62 89]	[62 164]	[88 153]	[26 62]	[7 62]
3437	[243 440]	[388 165]	[410 434]	[366 477]	[440 430]	[440 503]	[440 503]
USPG	–	[2847 3401]	[1853 3148]	[799 4463]	–	[3925 1776]	[4432 2747]

Table 6 Comparison of average percentage error ΔR_G over synthetic networks with 128 nodes

ID	$\Delta R^{S_1}_{G+\{e\}}$	$\Delta R^{S_2}_{G+\{e\}}$	$\Delta R^{S_3}_{G+\{e\}}$	$\Delta R^{S_4}_{G+\{e\}}$	$\Delta R^{RobGA}_{G+\{e\}}$	$\Delta R^{RobGA\{L^+\}}_{G+\{e\}}$
ER	1.028	0.029	0.044	0.093	0.004	0.103
WS	0.694	0.055	0.083	0.038	0.027	0.122
BA	0.607	0.219	0.064	0.012	0	0.0426

performs the best over all the datasets having very small ΔR_G average values. Its approximation, *RobGA*$\{L^+\}$, performs the third best over the Bárabasi-Albert networks while on Erdős-Rényi and Watts-Strogatz networks, the ranking of the strategies changes. On Erdős-Rényi networks, for example, the strategies ranking is *RobGA*, $S2$, $S3$, $S4$, *RobGA*$\{L^+\}$, $S1$. On Watts-Strogatz networks, the second best is $S4$, followed by $S2$, $S3$, *RobGA*$\{L^+\}$ and $S1$. Overall, the top performing strategy remains *RobGA*. The second best strategy is mainly *RobGA*$\{L^+\}$ and in some scenarios $S4$.

Despite the lower performance of *RobGA*$\{L^+\}$ over *RobGA*, which is negligible over real-world networks and Bárabasi-Albert networks, and more significant over Erdős-Rényi and Watts-Strogatz networks, its computational time is much lower, especially when compared to $S4$. In the second experiment, we focused on analyzing the simulation time of *RobGA*$\{L^+\}$ compared to *RobGA* and $S4$ by considering synthetic networks of increasing sizes: 128, 256, 512 and 1024. Table 7 illustrates the results in terms of speedup. Overall, *RobGA*$\{L^+\}$ is clearly faster than its contestant strategies. As the number of nodes increases, the speedup of *RobGA*$\{L^+\}$ becomes more significant, especially over $S4$ whose simulation time can be very high. On Watts-Strogatz networks with 1024 nodes, for example, *RobGA*$\{L^+\}$ is ten times faster than *RobGA* and even 1000 times faster than $S4$. On large networks, $S4$ is not easily applicable, it can be chosen if parallel computation can be run. Compared to *RobGA*, *RobGA*$\{L^+\}$ provides a good trade-off between a low percentage error of effective graph resistance and low simulation time. Especially on large networks where the simulation time of the exhaustive search or $S4$ is too high, the *RobGA*$\{L^+\}$ strategy would be a preferable choice. When investing on infrastructure networks, for example, where the addition of a link is costly and a strategy close to the exhaustive search is a requirement, *RobGA*$\{L^+\}$ would provide a computationally efficient solution.

6 Conclusion

This paper focused on the enhancement of network robustness by evaluating the impact a single link addition has on the resulting effective graph resistance of the

Table 7 Speedup of *RobGA*$\{L^+\}$ over *RobGA* and S4 on synthetic networks

Network type	Strategy	$N = 128$	$N = 256$	$N = 512$	$N = 1024$
Erdős-Rényi	*RobGA*	1.984	5.381	7.612	10.261
	S4	1.985	101.993	401.451	1097.735
Watts-Strogatz	*RobGA*	1.805	3.53	5.076	9.708
	S4	1.311	97.859	216.135	885.474
Bárabasi-Albert	*RobGA*	1.781	2.535	7.347	7.141
	S4	2.933	59.4267	379.817	795.38

network. We presented a comparative analysis of 6 different single-link addition strategies with the aim of evaluating their degree of robustness improvement and the speedup over the exhaustive search. The results on both real-world and synthetic networks indicate that the methods based on Genetic Algorithms are the best performing. It is worth pointing out that, though *RobGA* is still the best performing approach, the modified version *RobGA*$\{L^+\}$ is more competitive in terms of computing time, and finds solutions which are very close to the exact one.

RobGA$\{L^+\}$ can be selected as candidate strategy since it is able to provide a good trade-off between a low percentage error of the effective graph resistance with respect to the exhaustive search and low simulation time. Especially on large and sparse networks where the simulation time of the exhaustive search or similar strategies, like S_4, is too high, the *RobGA*$\{L^+\}$ strategy would be a preferable choice.

References

1. Beygelzimer, Alina, Grinstein, Geoffrey, Linsker, Ralph, Rish, Irina: Improving network robustness. Phys. A: Stat. Mech. Appl. **357**(3–4), 593–612 (2005)
2. Ellens, W., Kooij, R.E.: Graph measures and network robustness. arXiv:1311.5064
3. Ellens, W., Spieksm, F.M., Van Mieghem, P., Jamakovic, A., Kooij, R.E.: Effective graph resistance. Linear Algebra Appl. **435**(10), 2491–2506 (2011)
4. Fiedler, Miroslav: Algebraic connectivity of graphs. Czech. Math. J. **23**(2), 298–305 (1973)
5. Freitas, S., Yang, D., Kumar, S., Tong, H., Chau, D.H.: Graph vulnerability and robustness: a survey. In: IEEE Transactions on Knowledge and Data Engineering. https://doi.org/10.1109/TKDE.2022.3163672 (2022)
6. Ghosh, A., Boyd, S., Saberi, A.: Minimizing effective resistance of a graph. SIAM Rev. **50**(1), 37–66 (2008)
7. Goldberg, D.E.: Genetic Algorithms in Search. Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc, Boston, MA, USA (1989)
8. Klein, D.J., Randić, M.: Resistance distance. J. Math. Chem. **12**, 81–95 (1993)
9. Pizzuti, C., Socievole, A.: A genetic algorithm for improving robustness of complex networks. In: Tsoukalas, L.H., Grégoire, É., Alamaniotis, M. (eds.) IEEE 30th International Conference on Tools with Artificial Intelligence, ICTAI 2018, 5–7 November 2018, Volos, Greece, pp. 514–521. IEEE (2018)

10. Ranjan, G., Zhang, Z.L., Boley, D.: Incremental computation of pseudo-inverse of laplacian. In: Combinatorial Optimization and Applications. COCOA, pp. 730–749. Springer International Publishing, Switzerland (2014)
11. Tizghadam, A., Leon-Garcia, A.: On robust traffic engineering in core networks. In: IEEE Global Telecommunications Conference, GLOBECOM, vol. 2008, pp. 1–6 (2008)
12. Wang, X., Pournaras, E., Kooij, R.E., Van Mieghem, P.: Improving robustness of complex networks via the effective graph resistance. Eur. Phys. J. B **87**(9):221 (2014)

Analysis on the Effects of Graph Perturbations on Centrality Metrics

Lucia Cavallaro, Pasquale De Meo, Keyvan Golalipour, Xiaoyang Liu, Giacomo Fiumara, Andrea Tagarelli, and Antonio Liotta

Abstract Graph robustness upon node failures state-of-art is huge. However, not enough is known on the effects of centrality metrics ranking after graph perturbations. To fill this gap, our aim is to quantify how much small graph perturbations will affect the centrality metrics. Thus, we considered two type of probabilistic failure models (i.e., Uniform and Best Connected), a fraction τ of nodes under attack, with $0 < \tau \leq 1$, and three popular centrality metrics (i.e., Degree, the Eigenvector and the Katz centrality). We discovered that in the Uniform model the amount of change in the adjacency matrix due to a perturbation is not significantly affected when τ is small even with a quite high failure probability (i.e., $p \leq 85\%$) and that the Eigenvector centrality is the most susceptible metric to deformation respect to the others herein analysed; whereas, in the Best Connected model, the amount of perturbation is proportional to τ.

Keywords Network science · Graph robustness · Centrality metrics · Spectral radius · Probabilistic failure

L. Cavallaro (✉) · A. Liotta
Free University of Bozen-Bolzano, Faculty of Computer Science, Bolzano, Italy
e-mail: lucia.cavallaro@unibz.it

P. De Meo
University of Messina, Department of Ancient and Modern Civilizations, Messina, Italy

K. Golalipour
Islamic Azad University, Department of Computer Engineering, Sari, Iran

X. Liu
Chongqing University of Technology, Chongqing, China

G. Fiumara
University of Messina, Department of Mathematics and Computer Science, Physics and Natural Science, Messina, Italy

A. Tagarelli
University of Calabria, Department of Computer Engineering, Modelling, Electronics, and Systems Engineering, Cosenza, Italy

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_34

1 Introduction

Node centrality metrics have been extensively studied by researchers from a broad range of disciplines such as Computer Science, Sociology, Economics and Life Sciences to identify the most important elements in complex systems [1, 2]. Nodes with high centrality are those nodes whose removal quickly causes the fragmentation of a complex system into many independent subsystems [2]. In case of human societies, nodes with high centrality often contribute the most to the spread of diseases or new ideas [3].

Popular examples of centrality metrics are the *Degree* (which counts the number of neighbours of a node i) and the *Eigenvector centrality* (which assumes a node i is important if it is linked to nodes which are, in their turn, important). A further example of centrality function is the *Katz centrality* that is defined as the sum of the contributions associated with all walks[1] starting from a node i.

The computation of node centrality heavily depends on the accurate knowledge of the *topology* of the graph G; i.e., on the correct specification of the nodes and edges in G. To this end, the simplest object to describe the graph topology is the *adjacency matrix*, $\mathbf{A} \in \mathbb{R}^{|N|\times|N|}$, namely a square matrix such that the entry $\mathbf{A}_{ij}$ equals one if and only if nodes i and j are connected by an edge, 0 otherwise.

Malicious external agents as well as the malfunctioning of some components of the system associated with G might cause the deactivation of some nodes/edges and, thus, an alteration of the topology of G. Such an alteration compromises the efficiency of the system represented by G. In the worst cases, the loss in connectivity can stop the functioning of the whole system. It is well known, for instance, that the failure of some routers on the Internet could cause the interruption of communications on a global scale [4, 5]. According to the terminology introduced in [5], nodes in a graph are *occupied* (or not) if the physical elements to which they correspond are functioning (or not); the probability of occupation of each node can be uniform or it can depend on other parameters such as the node degree [5].

Previous research works [1, 6] investigated how the connectivity of G would change if a fraction of its nodes were deleted and such a problem is known as *site percolation problem*; other authors, instead, focused on the the *bond percolation problem*, namely how graph robustness depends on the corruption of some of its edges [7].

Despite the prolific research activity in the field of graph robustness upon node/edge failures, not enough is known on the consequences that the deactivation of some nodes along with their incident edges would have on node centrality. In addition, approaches considered so far assume that *all nodes* in G can fail; however, such an assumption is unrealistic [8].

In this paper, we consider two probabilistic node failure models which have been introduced in [8], that is the *Uniform* model, which assume that each node can fail

[1] A walk of length k in a graph is a sequence $\langle i_0, i_1, \ldots, i_{k-1}\rangle$ of nodes such that pair of consecutive nodes in the sequence are connected by an edge.

with probability p, being p a constant, and the *Best Connected* (BC) model, which assume that a node can survive an attack with probability proportional to its degree. We also assume that a fraction τ (with $0 < \tau \leq 1$) of nodes is under attack.

In the scenario above, we are concerned with two main questions: $\mathbf{RQ}_1$ Under which conditions can we classify a perturbation as "small" in the *Uniform* and in the *Best Connected* (BC) models, respectively? $\mathbf{RQ}_2$ How does the norm of the centrality vector vary in the Uniform and in the BC models?

To address our research questions we considered four real dataset on which we applied the two probabilistic failure models by varying the fraction of targeted nodes τ and the failure probability p. Our results unveiled to be consistent on what asserted by Albert et al. [9] when τ is small (e.g., $\tau = 0.1$). Indeed, [9] proved that if we remove a small fraction of nodes from an ER graph, some topological properties of the graph (such as the size of the largest connected component or the diameter) tend to vary little. Instead, for high values of τ (e.g., $\tau = 1$), then the role of p became crucial in the perturbation analysis; indeed, we highlighted a decrease in ψ (i.e., the metric that quantify the amount of change in the adjacency matrix $\mathbf{A}$ due to the application of a perturbation) that gets more and more clear as p gets large.

The paper is organised as follows: Sect. 2 briefly summarises the state-of-art, in Sect. 3 we provide some background materials on graphs and the node centrality metrics herein considered. In Sect. 4 we define the strategy followed to perform our experiments as well as we describe the metrics herein used to quantify the amount of perturbation. Section 5 illustrates the experiments we performed to study how variation in $||\Delta\mathbf{A}||$ impact on the the Degree, the Eigenvector and the Katz centrality. Lastly, in Sect. 6 we draw our conclusions and illustrate our future research plans.

2 Related Works

One of the early approaches to analysing how alterations in graph topology affect the ranking generated by a centrality metric is due to Costenbader and Valente [10]. The authors taken random samples from an input directed graph and they varied the proportion of sampled nodes; specifically, they started by sampling 80% of the available nodes and they gradually decreased the sampling proportion by 10%. The process above stopped if the sampled network contained less than 10% of the input nodes. At each sampling level, Costenbader and Valente [10] computed how the centrality in the original graph and in the sampled graph were correlated. The authors found that some centrality metrics such as the in-degree and the eigenvector centrality in the original and sampled graph were highly correlated; for other centrality metrics such as the out-degree), they observed a quicker decline in average correlation as function of the sampling rate.

A nice extension of the work done by Costenbader and Valente is due to Borgatti et al. [11], who studied whether some centrality metrics can be regarded as robust if random errors occurred in the graph topology. The authors generated random graphs of different size and density and they considered four types of errors, namely, edge

deletion, node deletion, edge addition, and node addition. The main results of the study proposed in [11] show that the accuracy of centrality measures declines in a predictable function as function of the amount of error.

The approaches above assume that graph topology is full specified and that some sampling task has been applied on it.

A different perspective has been considered by Diesner et al. [12]; here, the authors consider social networks constructed from records of social interactions. Potential ambiguities of social entities may greatly affect the network construction process: for instance, nodes associated with the same string could be wrongly merged despite they correspond to distinct individuals. Diesner et al. [12] investigated the robustness of some centrality metrics such as the in-degree and they found that some graph statistics are heavily influenced from incorrect data but the process of detecting the most important nodes was robust to disambiguation flaws. Such a result implies that highly central individuals will still continue occupying a prominent ranking if we heavily corrupt input data. In line with Diesner et al. [12], Mishra et al. [13] studied to what extent flawed author name disambiguation can lead to wrong conclusions about gender bias in science.

3 Background

In this section we provide some background materials that will be extensively used throughout the paper.

We define a graph G as a pair $G = \langle N, E \rangle$ in which N is the *set of nodes* and $E \subseteq N \times N$ is the set of edges. Herein, we conducted our experiments on undirected and unweighted graphs, which means that for each edge $\langle i, j \rangle \in E$, we have $\langle j, i \rangle \in E$ (i.e., undirected graphs) and that the edges have all the same cost (i.e., unweighted graphs). We define the *order* of a graph as the number $n = |N|$ of its nodes and the *size* of a graph as the number $m = |E|$ of its edges.

We say that a graph is *sparse* (resp., *dense*) if $m = O(n)$ (resp., $m = O(n^2)$).

Given a node $i \in N$, we define the *neighbourhood* $N(i)$ of i as the set of nodes linked to i, namely $N(i) = \{j \in N : \langle i, j \rangle \in E\}$. The *degree* d_i of i is equal to the number of neighbours of i, namely $d_i = |N(i)|$.

A *walk* of length k (being k a non-negative integer) is an ordered sequence of nodes $\langle i_0, i_1, \ldots, i_k \rangle$ such that consecutive nodes in the sequence are tied by an edge. We use the term *path* for walks that do not have repeated vertices. A walk is *closed* if it starts and ends at the same node.

Each unweighted graph G of order n is associated with an $n \times n$ matrix $\mathbf{A}$ called *adjacency matrix* such that $\mathbf{A}_{ij} = 1$ if and only if $\langle i, j \rangle \in E$, 0 otherwise. The adjacency matrix is relevant to describe many graph properties: for instance, the matrix $\mathbf{A}^2$ where $\mathbf{A}^2_{ij} = \sum_{r=1}^{n} \mathbf{A}_{ir}\mathbf{A}_{rj}$, gives the number of walks of length two going from i to j. By induction, for any positive integer k, the matrix $\mathbf{A}^k$ will give the number

of closed (resp., distinct) walks of length k between any two nodes i and j if $i = j$ (resp., if $i \neq j$) [14].

The adjacency matrix of any undirected graph is *symmetric* and, hence, all its eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ are real. The largest eigenvalue λ_1 of $\mathbf{A}$ is also called its *principal eigenvalue* or *spectral radius* of G. Moreover, the corresponding eigenvectors $\mathbf{e}_1, \ldots, \mathbf{e}_n$ will form an orthonormal basis in $\mathbb{R}^n$ [15]. Eigenpairs $\langle \lambda_i, \mathbf{e}_i \rangle$ are formed by the eigenvalue λ_i and its associated eigenvector $\mathbf{e}_i$.

We define the *centrality of a node* as a function $f_\Theta : N \to \mathbb{R}^+$ which takes as input a node $i \in N$ along with an (optional) set of parameters Θ and it returns a non-negative real number as output. The centrality $f_\Theta(i)$ of a node i assesses the "importance" of i within G; because we can interpret the notion of importance in a number of ways, we have consequently many definitions of centrality. In particular, in this paper we focus on three popular centrality metrics, that is *Degree, Eigenvector and Katz centrality*.

The *Degree Centrality* of a node coincides with the degree of that node. If we use the vector $\mathbf{d} \in \mathbb{R}^N$ to store the degree centrality of the nodes in G and we denote as $\mathbf{1} \in \mathbb{R}^n$ the vectors with all entries equal to one, then the degree centrality $\mathbf{d}$ can be computed as follows:

$$\mathbf{d} = \mathbf{A} \times \mathbf{1} \tag{1}$$

A further, interesting centrality metric is the *eigenvector centrality* [1]. Unlike the degree, the eigenvector centrality of a node i does not depend on the number of neighbours of i but *on the importance of these neighbours*. Specifically, the eigenvector centrality can be recursively computed through the following equation:

$$\mathbf{A} \times \mathbf{e} = \lambda \mathbf{e} \tag{2}$$

Here $\mathbf{e} \in \mathbb{R}^n$ is the vector storing the eigenvector centrality of nodes in G. If we assume that the graph G is undirected and connected, we can take the largest eigenvalue λ_1 and the corresponding eigenvector $\mathbf{e}_1$ (also known as *principal eigenvector*); by the Perron-Frobenius theorem [16] we have that all the components of $\mathbf{e}_1$ are positive and, thus, we can interpret the i-th component of $\mathbf{e}_1$ as the eigenvector centrality of $\mathbf{A}$ itself.

The Katz centrality [1] of a node i counts all walks beginning at i; each walk of length k is associated with a weight equal to β^k, being the parameter β called the *attenuation factor* [17]. We can introduce a vector $\mathbf{k} \in \mathbb{R}^n$ which stores the Katz centrality of the node i in its i-th component; the vector $\mathbf{k}$ is defined as follows:

$$\mathbf{k} = \left(\mathbf{I} + \beta\mathbf{A} + \beta^2\mathbf{A}^2 + \ldots\right) \times \mathbf{1} = \left(\sum_{k=0}^{+\infty} \beta^k \mathbf{A}^k\right) \times \mathbf{1} \tag{3}$$

being $\mathbf{I} \in \mathbb{R}^{n \times n}$ the identity matrix. If we assume that $\beta < \frac{1}{\lambda_1}$ then the series $\sum_{k=0}^{+\infty} \beta^k \mathbf{A}^k$ (often called *Neuman series*) is convergent and its sum equals to the the inverse of the matrix $\mathbf{I} - \beta\mathbf{A}$ [17]:

$$\mathbf{k} = \left(\sum_{k=0}^{+\infty} \beta^k \mathbf{A}^k \right) \times \mathbf{1} = (\mathbf{I} - \beta\mathbf{A})^{-1} \times \mathbf{1} \tag{4}$$

4 Methods

In this section we describe the metrics herein used to quantify the amount of change in the adjacency matrix $\mathbf{A}$ due to the application of a perturbation.

We define a *perturbation* as the action of making inactive a fraction τ (with $0 < \tau \leq 1$) of the nodes of a graph $G = \langle N, E \rangle$. Let $N' \subseteq N$ be a subset of nodes of size $|N|' = \lceil \tau \times |N| \rceil$ and we assume nodes in N' are under attack.

We assume that each node in N' is associated with a *failure probability* p_i and we considered two options for modelling p_i namely: (i) *Uniform*, in which p_i is a constant and (ii) *Best Connected* (BC), in which p_i is proportional to the degree of i [8]. Next, let $\mathbf{p} \in \mathbb{R}^{|N|}$ be a vector such that the i-th component of $\mathbf{p}$ equals the failure probability p_i.

A perturbation transforms the adjacency matrix $\mathbf{A}$ of G into a new matrix $\tilde{\mathbf{A}} = \mathbf{A} - \Delta\mathbf{A}$.

Hence, the problem we wish to address can be summarised by the following research questions:

RQ$_1$ Under which conditions can we classify a perturbation as "small" in the Uniform and in the BC models, respectively?

RQ$_2$ How does the norm of the centrality vector vary in the Uniform and in the BC models?

To quantify the amount of change associated with a perturbation we consider the following parameter ψ:

$$\psi = \frac{||\Delta\mathbf{A}||_F}{||\mathbf{A}||_F} \tag{5}$$

where $||\cdot||_F$ is the Frobenius norm [18].

We also computed the deformation of centrality metrics. Specifically, let $f_\Theta(\mathbf{A})$ (resp., $f_\Theta(\tilde{\mathbf{A}})$) be the vector containing the centrality scores computed via the function $f_\Theta(\cdot)$ on the input (resp., modified) adjacency matrix $\mathbf{A}$ (resp., $\tilde{\mathbf{A}}$). Here f denotes an arbitrary centrality metric while Θ is an optional set of parameter from which the calculation of $f(\cdot)$ depends on. To capture the deformation of the centrality vector we introduce the parameter ζ:

$$\zeta = \frac{||f_\Theta(\tilde{\mathbf{A}}) - f_\Theta(\mathbf{A})||}{||f_\Theta(\mathbf{A})||} \tag{6}$$

5 Experiments

In this section we report the experiments we carried out to validate our model to answer our research questions.

5.1 Datasets

We ran our evaluation on four real datastes, namely: (i) *Twitch-PT* [19] (1, 912 nodes and 31, 299 edges), a social network of Twitch users collected in Spring 2018. Twitch is a video live streaming service that provides services including, but not limited to video game live streaming and e-sports competitions broadcasts. Nodes are Twitch users located in Portugal and edges are mutual follower relationships between them. (ii) *Twitch-EN* [19] (7, 126 nodes and 35, 324 edges) is a dataset having the same structure and meaning of *Twitch-PT*, but its nodes correspond to Twitch users from UK. (iii) *AstroPH* [20] (18, 771 nodes and 198, 050 edges), a graph recording scientific collaborations between authors who submitted papers to the Astro-Physics category in the e-print arXiv service. Here nodes are associated with authors and there is an edge from nodes i and j if and only if authors i and j wrote a paper together. *Cond-Mat* [20] (30, 460 nodes and 120, 029 edges), a collaboration network depicting scientific collaborations between authors who submitted papers to the Condense Matter category in ArXiv. Nodes and edges have the same meaning of the ones in *AstroPH*.

5.2 When a Perturbation is Small

In this section we describe the first experiment devoted to answer $\mathbf{RQ}_1$, namely we wish to ascertain under which circumstances a perturbation on a graph G and, consequently, on its adjacency matrix $\mathbf{A}$ can be regarded as small. To do so we measured how the parameter ψ in Equation (5) varies as function of p and τ in the Uniform model and as function of τ in the BC Model.

We first focus on the Uniform model and, due to space limitations, we report in Figure 1a and b our results for $\tau = 0.1$ and $\tau = 1.0$. The Figures indicate that ψ is independent on the dataset under scrutiny.

In fact, the parameter ψ depends on the norm of $\Delta\mathbf{A}$ that depends only on the product $\tau \times p$ as well as on the graph order n. The dependence of $||\Delta\mathbf{A}||_F$ on n of nodes is absorbed by the denominator of ψ.

If $\tau = 0.1$, then ψ is constant for all values of p less than 0.85; *vice versa*, a steep decrease in ψ occurs if p is bigger than 0.85. In other words, if τ is small, the number of nodes which can actually fail is a small fraction of the entire node set and, thus,

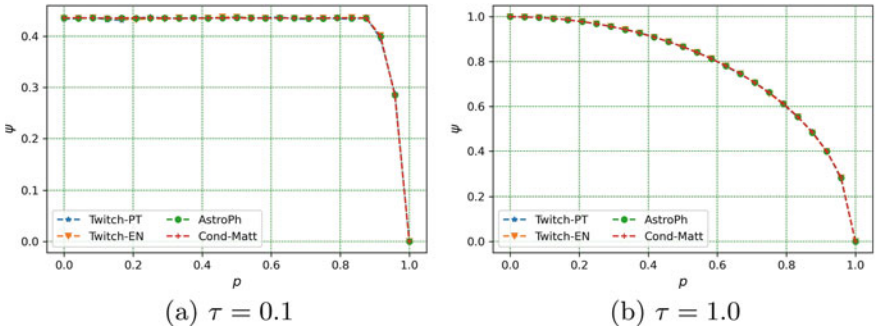


Fig. 1 Variation of ψ as function of p in the Uniform model for real graphs under investigations if **a** $\tau = 0.1$, **b** $\tau = 1.0$

the failure of these nodes does not significantly affect the norm of the perturbation matrix (which answer to $\mathbf{RQ}_2$).

Such an experimental behaviour extends previous work by Albert et al. [9] on the robustness of random graphs upon random node removal; in particular, Albert et al. [9] proved that if we remove a small fraction of nodes from an Erdős-Rényi (ER) graph, some topological properties of the graph (such as the size of the largest connected component or the diameter) tend to vary little. The high level of resilience of ER graphs is still true for $||\Delta\mathbf{A}||_F$.

If $\tau = 1$ all nodes can potentially fail, and, thus, the parameter p plays a key role in the variation of ψ; specifically, we highlight a decrease in ψ which gets more and more clear as p gets large.

In the BC strategy (see Fig. 2) we observe that if $\tau \to 1$, then we have a bigger chance of selecting high-degree nodes. The removal of high-degree nodes clearly generates a bigger increase in $||\Delta\mathbf{A}||_F$. This explains why ψ increases as τ increases. Unlike the Uniform model, the topology of the input graph in the BC model affects the value of ψ, which always increases in an almost linear fashion but the rate of growth of ψ differs from one dataset to another.

5.3 *How the Centrality Metrics Vary in Probabilistic Failure Models*

In this section we study how ζ varies as function of τ; in our tests, τ increased up to 0.2.

In the Uniform model we fixed $p = 0.1$. We also fixed $\alpha = 0.5$ in the calculation of the Katz coefficient.

From Figures 3 and 4 we observe that both the Degree and the Eigenvector centrality increase in a linear fashion as τ increase. Specifically, in the Uniform model, the ζ associated with the Eigenvector Centrality grows faster than the ζ correspond-

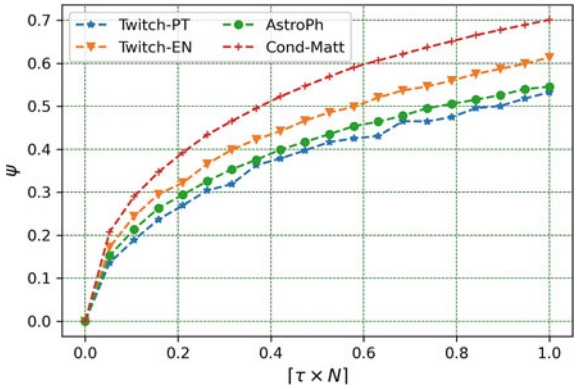


Fig. 2 Variation of ψ as function of τ in the BC model for the real graphs considered in our tests

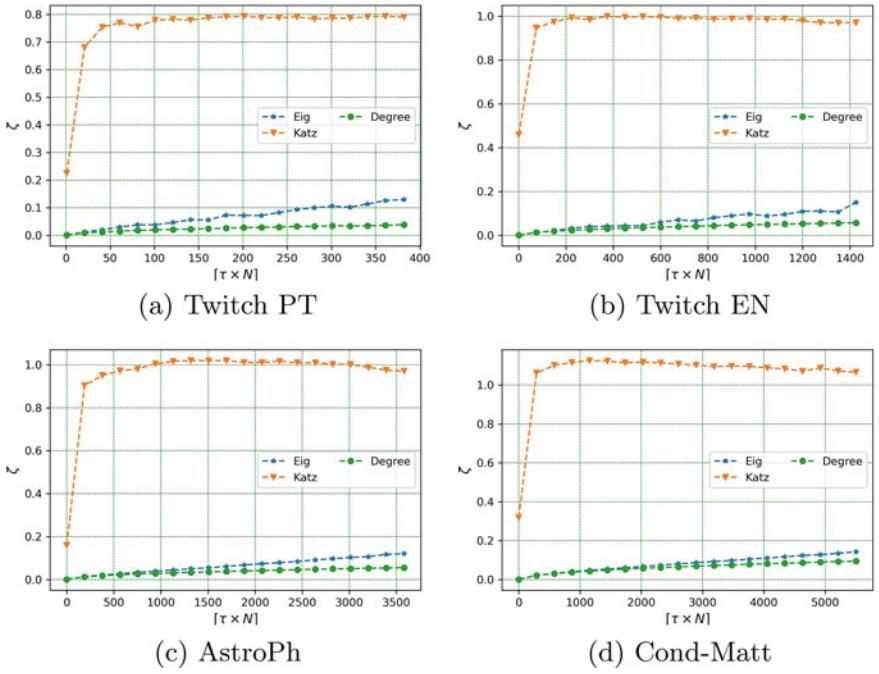


Fig. 3 Variation of ζ as function of τ in the uniform model for real graphs

ing to the Degree. An opposite trend emerges in the BC model: here, ζ grows faster in case of the Degree than in the Eigenvector Centrality.

From the analyses conducted with Katz Centrality resulted that small values of τ are sufficient to generate a sharp increase in ζ; however, the observed values of ζ tend to quickly stabilise.

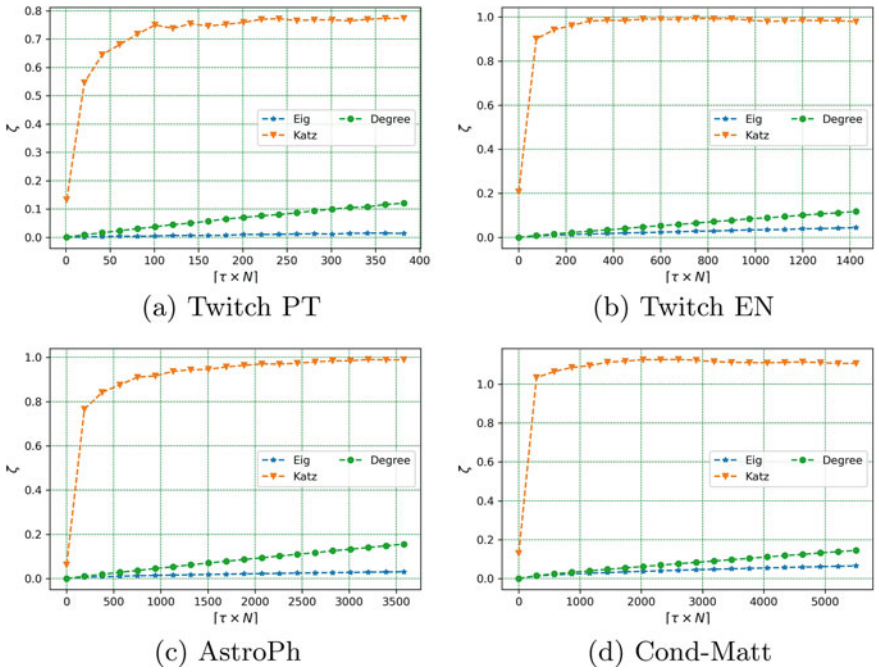


Fig. 4 Variation of ζ as function of τ in the BC model for real graphs

6 Conclusions

In this paper, we considered two probabilistic node failure models [8], called *Uniform* and *Best Connected*, and we simulated the removal of nodes (and corresponding edges) from a graph in both of them to investigate whether and to what extent small perturbations in a graph will affect the centrality metrics. As evaluation metrics we defined ψ (which quantifies the amount of change in the adjacency matrix due to the application of a perturbation) and ζ (which evaluate the deformation of centrality metrics).

The analyses unveiled that in *Uniform* model, ψ was constant if $\tau = 0.1 \forall p \leq 0.85$, whereas a steep decrease of ψ was detected for $p > 0.85$. Instead, for $\tau = 1$, ψ depends on p: the larger p, the lower ψ is. In terms of centrality metrics, ζ grown faster in Eigenvector Centrality than in Degree.

This means that the amount of perturbation is not significantly affected when a small fraction of nodes is targeted even with a quite high failure probability (i.e., $p \leq 85\%$) and that the Eigenvector centrality is the most susceptible metric to deformation respect to the other herein analysed.

On the other side, when the *Best Connected* model was considered, ψ grows proportionally to τ, whereas ζ had an opposite trend respect to what encountered in the

Uniform model; indeed, ζ grown faster in the Degree rather than in the Eigenvector Centrality.

Thus, when the most connected nodes are the most resilient within the network an higher fraction of targeted nodes can let to pick and successfully remove those strong nodes; as a consequence, the amount of perturbation is higher. In addition, the most affected centrality metric is the Degree as the nodes resilience was established to be proportional to such a metric.

Our next research goal consists of studying the *continuity* of the same centrality measures herein discussed (i.e., Degree, the Eigenvector Centrality and the Katz Centrality) if some nodes in a graph fail as well as expanding the range of pool of centrality metrics to study. Specifically, given their importance as tools for the analysis of complex systems, we plan to include centrality metrics such as the betweenness and the closeness. Unfortunately, the computation of betweenness/closeness relies on the calculation of all pairs shortest paths in a graph and, consequently, betweenness and closeness are hard to compute on graphs of even modest size. Recently, however, [21] introduced scalable Deep Learning methods to predict betweenness/closeness in large graphs; these methods were able to achieve accurate results even in large graphs and, thus, our goal is to apply the methods we developed in this paper with the centrality prediction techniques illustrated in [21] to study the continuity of betweenness and closeness.

References

1. Newman, M.: Networks: An Introduction. Oxford University Press (2010)
2. Lü, L., Chen, D., Ren, X., Zhang, Q., Zhang, Y., Zhou, T.: Vital nodes identification in complex networks. Phys. Rep. **650**, 1–63 (2016)
3. Kang, C., Kraus, S., Molinaro, C., Spezzano, F., Subrahmanian, V.: Diffusion centrality: a paradigm to maximize spread in social networks. Artif. Intell. **239**, 70–96 (2016)
4. Albert, R., Jeong, H., Barabási, A.: Error and attack tolerance of complex networks. Nature **406**(6794), 378–382 (2000)
5. Callaway, D.S., Newman, M.E.J., Strogatz, S.H., Watts, D.J.: Network robustness and fragility: percolation on random graphs. Phys. Rev. Lett. **85**(25), 5468 (2000)
6. Li, M., Liu, R., Lü, L., Hu, M., Xu, S., Zhang, Y.: Percolation on complex networks: theory and application. Phys. Rep. **907**, 1–68 (2021)
7. Moore, C., Newman, M.: Exact solution of site and bond percolation on small-world networks. Phys. Rev. E **62**(5), 7059 (2000)
8. Cavallaro, L., Costantini, S., De Meo, P., Liotta, A., Stilo, G.: Network connectivity under a probabilistic node failure model. In: IEEE Transactions on Network Science and Engineering, pp. 1–1 (2022)
9. Barabási, A., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)
10. Costenbader, E., Valente, T.W.: The stability of centrality measures when networks are sampled. Soc. Netw. **25**(4), 283–307 (2003)
11. Borgatti, S.P., Carley, M.M., Krackhardt, D.: On the robustness of centrality measures under conditions of imperfect data. Soc. Netw. **28**(2), 124–136 (2006)
12. Diesner, J., Evans, C.S., Kim, J.: Impact of entity disambiguation errors on social network properties. In: Ninth International AAAI Conference on Web and Social Media (2015)

13. Mishra, S., Fegley, B.D., Diesner, J., Torvik, V.I.: Self-citation is the hallmark of productive authors, of any gender. PloS One **13**(9), e0195773 (2018)
14. Cvetković, D., Rowlinson, P., Simic, S.: Eigenspaces of Graphs, vol. 66. Cambridge University Press (1997)
15. Strang, G.: Introduction to Linear Algebra, vol. 3. Cambridge Press Wellesley, Wellesley, MA (1993)
16. Langville, A., Meyer, C.: Google's PageRank and Beyond. Princeton university press (2011)
17. Benzi, M., Klymko, C.: On the limiting behavior of parameter-dependent network centrality measures. SIAM J. Matrix Anal. Appl. **36**(2), 686–706 (2015)
18. Franklin, J.: Matrix Theory. Courier Corporation (2012)
19. Rozemberczki, B., Allen, C., Sarkar, R.: Multi-scale attributed node embedding. J. Complex Netw. **9**(2):cnab014 (2021)
20. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graph evolution: densification and shrinking diameters. ACM Trans. Knowl. Disc. Data (TKDD) **1**(1):2–es (2007)
21. Grando, F., Zambenedetti Granville, L., Lamb, L.C.: Machine learning in network centrality measures: tutorial and outlook. ACM Comput. Surv. **51**(5):102:1–102:32 (2019)

Robustness of Preferential-Attachment Graphs: Shifting the Baseline

Rouzbeh Hasheminezhad and Ulrik Brandes

Abstract The widely used characterization of scale-free networks as "robust-yet-fragile" originates primarily from experiments on instances generated by preferential attachment. According to this characterization, scale-free networks are more robust against random failures but more fragile against targeted attacks when compared to random networks of the same size. Here, we consider a more appropriate baseline by requiring that the random networks match not only the size but also the inherent minimum degree of preferential-attachment networks they are compared with. Under this more equitable condition, we can (1) prove that random networks are almost surely robust against any vertex removal strategy and (2) show through extensive experiments that scale-free networks generated by preferential attachment are not particularly robust against random failures.

Keywords Robustness · Scale-free networks · Preferential attachment

1 Introduction

The class of scale-free networks is of particular interest in network science. While preferential attachment models can generate only a vanishing fraction of all scale-free networks [17], many claims about scale-free networks arise from experiments on preferential-attachment instances. One such claim, originating from [1], is that scale-free networks have a "robust-yet-fragile" nature, i.e., compared to random networks of the same size, they are more robust against random failures but more fragile against targeted attacks [9].

Common models for preferential attachment, including the one used in the experiments of [1], yield instances with a constant average degree and a minimum degree

R. Hasheminezhad (✉) · U. Brandes
Social Networks Lab, ETH Zürich, Zürich, Switzerland
e-mail: shashemi@ethz.ch

U. Brandes
e-mail: ubrandes@ethz.ch

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_35

of half that value. Because of their constant average degree, however, random networks of the same size are very likely to contain isolated vertices [10]. We prove, as a first contribution, that the robustness and connectivity of such random networks changes significantly if they are required to have a minimum degree of at least k for any constant $k \geq 3$. With this in mind, it seems more appropriate and natural to compare the robustness of preferential-attachment instances with random networks of the same size *and the same minimum degree*. To the best of our knowledge, this has not yet been done. We aim to fill this gap with our second contribution through an extensive suite of experiments. Our experiments show, to the affirmative, that scale-free instances generated by preferential attachment are consistently more fragile than size-matching random networks whose minimum degree is at least as large. To put it more bluntly: We find that in an equitable setting, graphs created by preferential attachment do not exhibit a "robust-yet-fragile" nature. Our formal contributions can be summarized as follows:

1. For any constant $k \geq 3$, almost all graphs with a constant average degree and a minimum degree of at least k are connected and provably robust against any vertex removal strategy.
2. Under targeted attacks and random failures, scale-free networks generated by preferential attachment are more vulnerable than random networks of the same size whose minimum degree is at least as large.

2 Preliminaries

We use $\mathbb{N}$ to denote the set of positive integers. When we say that a statement holds for large enough $n \in \mathbb{N}$, there exists a constant $n_0 \in \mathbb{N}$ such that the statement holds for all n that are larger. We say that a sequence of events $\mathcal{A}_n$ holds almost surely if $\lim_{n\to\infty} \mathsf{Pr}[\mathcal{A}_n] = 1$.

2.1 Graphs and Degree Sequences

In this paper, we consider only simple undirected graphs and use the terms graph and network interchangeably. A graph $G = (V, E)$ consists of a set of vertices V and a set of edges $E \subseteq \binom{V}{2}$. If $\{u, w\} \in E$, then u and w are said to be adjacent. A graph is called complete if each vertex is adjacent to all other vertices. The degree, $\deg_G(v)$, of a vertex v is the number of vertices in G adjacent to v. If $v_1, \cdots, v_n$ is an ordering of V where $\deg(v_1) \geq \cdots \geq \deg(v_n)$, then $D(G) = (\deg(v_1), \ldots, \deg(v_n))$ is the degree sequence of G.

The subgraph of a graph $G = (V, E)$ induced by $V' \subseteq V$ is $G[V'] = (V', E')$, where $E' = \big\{\{u, w\} \in E | u, w \in V'\big\}$. Given a positive integer k, the k-core of a

graph G is the inclusion-maximal induced subgraph, where all vertex degrees are at least k. The k-core of a graph is unique and can be determined efficiently [5].

The reachability relation is defined as the reflexive and transitive closure of the adjacency relation. The connected components of a graph are its subgraphs induced by the equivalence classes of the reachability relation. A graph is called connected if it consists of a single connected component. The largest connected component, or LCC for short, is the one with the largest number of vertices.

2.2 Network Robustness

The invariance of a network structural property when the elements of the network are removed is referred to as the robustness of that network [14]. We focus only on the removal of vertices and consider the number of vertices in the largest connected component as the structural property of interest.

Given a connected graph $G = (V, E)$ and the sequence of vertices $B = (b_1, b_2, \ldots, b_T)$ in order of their removal, we can quantify the robustness of G by

$$R_G(B) = \frac{1}{T} \sum_{t=1}^{T} \frac{|\mathrm{LCC}\,(G[V \setminus \{b_1, \ldots, b_t\}])\,|}{|\mathrm{LCC}\,(G)\,|}.$$

This robustness score originally proposed in [12] is a generalization of the score used in [18], where in the latter score, B is a permutation of V. Note that the above score captures the relative size of the largest connected component and the rate at which it shrinks when the vertices are removed. The most commonly considered vertex removal strategies in the literature are random failures and targeted attacks. In random failures, vertices are removed uniformly at random. In targeted attacks, the vertices with the highest initial degree are removed first.

If the strategy is clear from the context and we accept random variation due to vertex selection and tie-breaking rules, we can parameterize the robustness score by the fraction β of removed vertices rather than by the precise sequence. Note that $R_G(\beta)$ is upper-bounded by $1 - \beta/2 + o(1)$ for any strategy [11].

A graph invariant closely related to robustness is its (vertex) isoperimetric number, defined as $h(G) = \min_{\emptyset \neq S \subset V, |S| \leq \frac{|V|}{2}} \left\{ \frac{|\partial S|}{|S|} \right\}$, where ∂S is the subset of vertices in $V \setminus S$ that are adjacent to at least one vertex in S. A graph G is called an α-(vertex) expander for some constant $\alpha > 0$, if $h(G) \geq \alpha$. Intuitively, this means that many vertices need to be removed to disconnect a sizable subset of vertices from the rest.

2.3 Network Models

The set of simple graphs with n vertices and m edges is denoted by $G(n, m)$, and $G(n, m, k)$ is the subset of graphs in $G(n, m)$ that have minimum degree at least k. The models $\mathsf{G}(n, m)$ and $\mathsf{G}(n, m, k)$ consist of the uniform distribution on $G(n, m)$ and $G(n, m, k)$, respectively.

Scale-free networks are those networks in which the fraction of vertices with degree k is approximately proportional to $k^{-\gamma}$ for some $\gamma > 1$. Since its popularization by Barabási and Albert [3], preferential attachment has been the most widely used mechanism for generating scale-free networks.

Although there are several instantiations of the same general idea, here we adhere to the approach used in the original robustness experiments of [1]. This generative preferential-attachment model $\mathsf{PA}(n, k)$ starts from a complete graph with $2k + 1$ vertices and successively adds $n - (2k + 1)$ vertices. Each newly added vertex is made adjacent to k distinct vertices, drawn without replacement from the pool of existing vertices and with probability proportional to their current degree. All graphs generated in this way are connected, have a minimum degree of k, $m = kn$ edges (an average degree of $2k$), and are thus elements of $G(n, nk, k)$. Eliminating the vertices of a graph generated from $\mathsf{PA}(n, k)$ in reverse order of construction shows that its k-core is the entire graph, with the corresponding seed graph being the only higher-degree core nested in the k-core.

To assess their relative robustness, preferential attachment graphs are often compared with random size-matching graphs drawn from $\mathsf{G}(n, m)$. If the number of edges is bounded linearly and away from one, i.e., $m = kn$ edges for some constant $k > 1$, random graphs are not likely to be connected but almost surely have a giant component [10, 16]. This is also acknowledged in the robustness experiments of Albert et al. [1], where rejection sampling is used to find a graph with a large enough largest connected component.

So, while size and connectivity are largely held constant in the experiments, the minimum-degree property of preferential-attachment graphs has not been considered. We will show in the next section that this can be expected to have a major influence on what can reasonably be considered robust.

3 Theory

Under the condition that the minimum degree is at least k for a constant $k \geq 3$, we show in Theorem 1 that almost all graphs with a constant average degree are not only connected but also provably robust against any vertex removal strategy.

The significant change in connectivity and robustness of the sparse random networks described above by constraining their minimum degree implies that the conclusions of studies on the robustness of networks such as [1], in which scale-free instances with constant average degree are pitted against size-matching random net-

works, could also change significantly in a fairer framework if the size-matching random networks satisfy the additional requirement of having a minimum degree at least as large as that of the scale-free instances compared with them. This observation is one of the main motivations for our experiments in Sect. 4.

Theorem 1 *Let $\epsilon \in (0, 1)$ be any constant. Furthermore, let G be a graph drawn from $\mathcal{G}(n, m, k)$, where $2m = cn$ and $c \geq k \geq 3$ for some constants c, k. If n is large enough, then almost surely G is connected and $R_G(B) = 1 - o(1)$ for any vertex sequence B with $|B| \leq n^{\epsilon}$.*

Proof The proof of Lemma 2.3.5 in [15] and Lemma 2.2 in [6] imply that it suffices to show that $G = (V, E)$ is almost surely an expander, i.e., there exists a constant $\alpha > 0$ for which G is an α-expander. Let ξ be the event that G is not an expander. In the following, we will show that $\mathsf{Pr}[\xi] \in o(1)$ and hence almost surely G is an expander.

Let $\mathcal{D} = \{D(G) : G \in G(n, m, k)\}$ and $\tilde{\mathcal{D}}$ be a subset of degree sequences in $\mathcal{D}$ for which the maximum degree is at most $2\,(\log n)^2$, where $\log n$ is the natural logarithm of n. We can write $\mathsf{Pr}[\xi]$ as

$$\underbrace{\sum_{d \in \mathcal{D} \setminus \tilde{\mathcal{D}}} \mathsf{Pr}\left[\xi | D(G) = d\right] \mathsf{Pr}\left[D(G) = d\right]}_{:=A} + \underbrace{\sum_{d \in \tilde{\mathcal{D}}} \mathsf{Pr}\left[\xi | D(G) = d\right] \mathsf{Pr}\left[D(G) = d\right]}_{:=B}.$$

It is sufficient to show that $A \in o(1)$ and $B \in o(1)$. First, we show the former. Note that

$$A := \sum_{d \in \mathcal{D} \setminus \tilde{\mathcal{D}}} \underbrace{\mathsf{Pr}\left[\xi | D(G) = d\right]}_{\leq 1} \mathsf{Pr}\left[D(G) = d\right] \leq \sum_{d \in \mathcal{D} \setminus \tilde{\mathcal{D}}} \mathsf{Pr}\left[D(G) = d\right].$$

The right-hand side of the above inequality is the probability that the maximum degree of G is greater than $2(\log n)^2$. Based on Lemma 12 in [8], the probability of the aforementioned event is asymptotic to zero, under our assumptions. This implies $A \in o(1)$. It remains to show $B \in o(1)$. Note that

$$\begin{aligned} B &:= \sum_{d \in \tilde{\mathcal{D}}} \mathsf{Pr}\left[\xi | D(G) = d\right] \mathsf{Pr}\left[D(G) = d\right] \\ &\leq \max_{d \in \tilde{\mathcal{D}}} \mathsf{Pr}\left[\xi | D(G) = d\right] \underbrace{\sum_{d \in \tilde{\mathcal{D}}} \mathsf{Pr}\left[D(G) = d\right]}_{\leq 1} \leq \max_{d \in \tilde{\mathcal{D}}} \mathsf{Pr}\left[\xi | D(G) = d\right]. \end{aligned}$$

In [7] the authors showed that a uniformly sampled graph conditioned on having a given degree sequence of size n in which the elements are not less than 3 and not greater than $n^{0.02}$ is almost surely an expander. This implies that the term on the right-hand side of the above inequality is asymptotic to zero; hence $B \in o(1)$, which completes the proof. □

4 Experiments

The conclusions in [1] stemmed mainly from experiments on synthetic scale-free networks generated from $\mathsf{PA}(n, k)$. We use the same type of synthetic networks to compare the robustness of preferential-attachment networks with size-matching random networks and random networks of the same size whose minimum degree is at least as large. To this end, we generate 100 networks using the model $\mathsf{PA}(n, k)$, where $n = 10{,}000$, $k = 3$ and draw an equal number of networks from $\mathsf{G}(n, m)$ and $\mathsf{G}(n, m, k)$, respectively, where $m = nk$.[1] For each of the 300 networks, we then compute their robustness scores under random failures and targeted attacks where the fraction of removed vertices is $\beta \in \{0.05, 0.1, 0.2\}$.

The results shown in Fig. 1 confirm that preferential-attachment networks are "robust-yet-fragile" when compared to random graphs of the same size, i.e., they are more fragile against targeted attacks but more robust against random failures. However, such networks are more vulnerable against both, targeted attacks and random failures, than size-matching random networks constrained to have at least the same minimum degree. We corroborate this finding in the following two sections under varying conditions.

Note that the minimum degree of graphs appears to be a crucial property for their robustness. This is suggested asymptotically by Theorem 1 and evidenced empirically, as the robustness scores of graphs drawn from $\mathsf{G}(n, m, k)$ are very close to the theoretical upper bounds indicated by dashed lines. This is further investigated in Sect. 4.3.

4.1 Sensitivity to the Choice of the Fraction of Removed Vertices

In this section, we evaluate the sensitivity of our observed patterns in Fig. 1 to the choice of the proportion of removed vertices. For this purpose, we repeat the procedure to create Fig. 1 but instead of choosing moderate proportions of removed vertices $\beta \in \{0.05, 0.1, 0.2\}$, we choose relatively higher proportions by increasing each previously considered proportion by 0.2, i.e., we consider $\beta \in \{0.25, 0.3, 0.4\}$. The result is shown in Fig. 2, from which we observe that the claimed patterns based on Fig. 1 still hold in general. The main difference is that the robustness of the random networks drawn from $\mathsf{G}(n, m, k)$ becomes less optimal when the proportion of removed vertices increases noticeably.

[1] We construct $\mathsf{PA}(n, k)$ and $\mathsf{G}(n, m)$ graphs using linear-time algorithms [4], rejecting instances with less than 96% of their vertices in the LCC. Under mild assumptions, for $\tilde{n}$ and $\tilde{m}$ given in [13], the k-core of a graph drawn from $\mathsf{G}(\tilde{n}, \tilde{m})$ is likely to have n vertices and m edges; in which case it is uniformly distributed on $G(n, m, k)$ [2]. This observation allows us to efficiently draw from $\mathsf{G}(n, m, k)$ by rejection sampling.

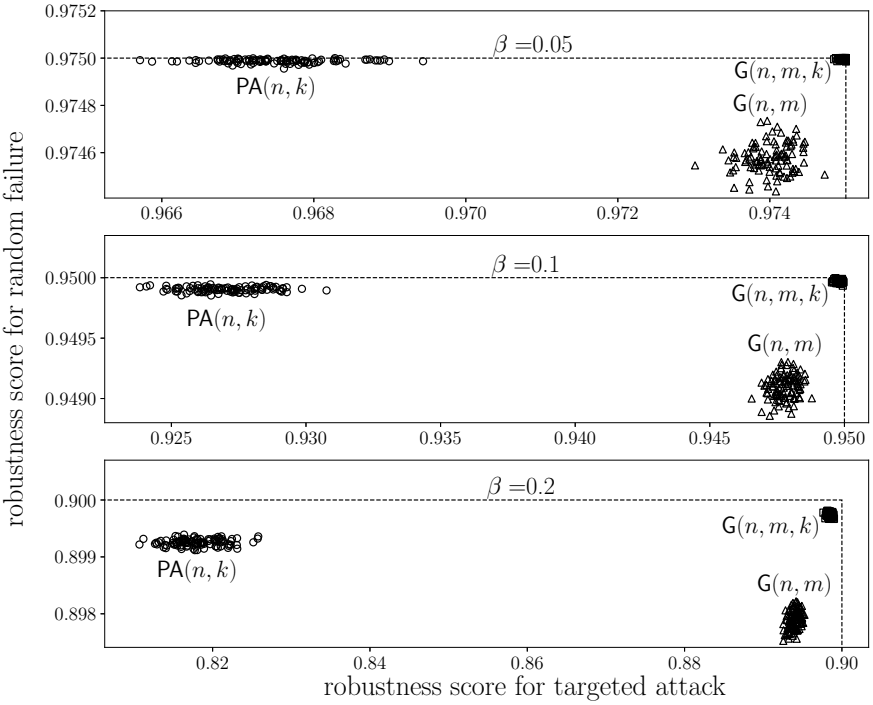


Fig. 1 Robustness of networks generated from $\mathsf{PA}(n, k)$ compared to networks drawn from $\mathsf{G}(n, m)$ and $\mathsf{G}(n, m, k)$ after 5%, 10%, and 20% of vertices were removed in targeted attacks or random failures, where $n = 10{,}000$, $m = 30{,}000$, and $k = 3$. The dashed lines represent the upper bounds for the robustness score as given in Sect. 2.2

4.2 Sensitivity to the Choice of Parameters in the PA Model

In this section, we evaluate the sensitivity of our inferred patterns based on Fig. 1, to the variation of parameters n, k in the underlying preferential attachment model $\mathsf{PA}(n, k)$. For this purpose, we vary $k \in \{3, 4, 5\}$ for fixed $n = 10{,}000$ and $n \in \{1000, 10{,}000, 20{,}000\}$ for fixed $k = 3$, all else being equal and precisely as in the setting presented at the beginning of Sect. 4. For each pair of n, k considered, we use z-scores to compare the networks generated from $\mathsf{PA}(n, k)$ with random networks drawn from $\mathsf{G}(n, m)$ and random networks drawn from $\mathsf{G}(n, m, k)$, where $m = nk$.[2] The comparison refers to their expected robustness score when $\beta \in \{0.05, 0.1, 0.2\}$ fraction of vertices are removed under targeted attacks or random failures. The obtained z-scores are visualized in Fig. 3.

[2] Given a group X and a reference group Y, both of size N, with respective means μ_X, μ_Y and standard deviations σ_X, σ_Y, we compute the corresponding z-score as $\sqrt{N}(\mu_X - \mu_Y)/(\sqrt{\sigma_X^2 + \sigma_Y^2})$.

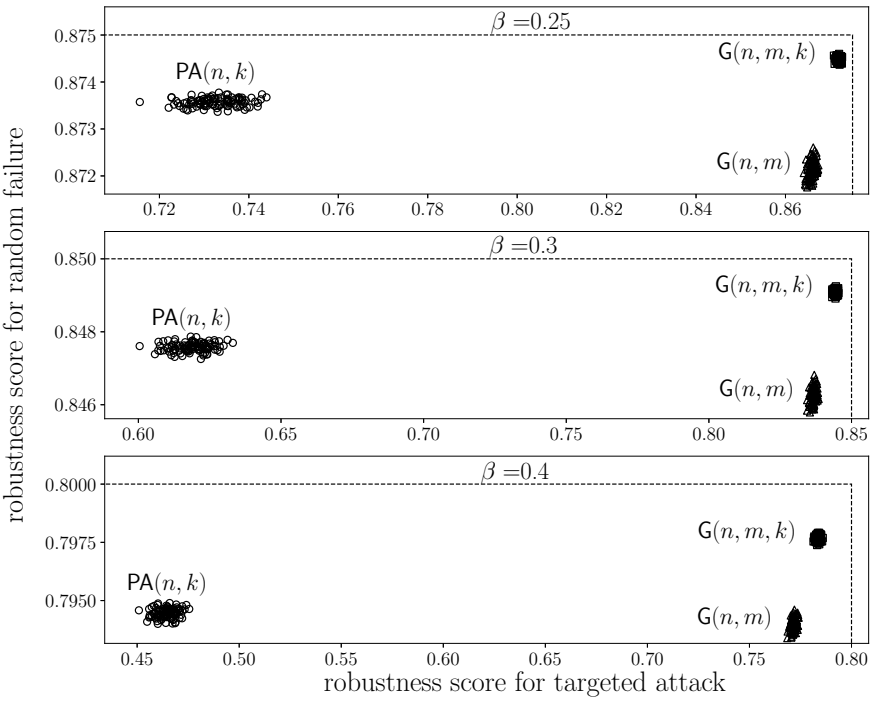


Fig. 2 Robustness of networks generated from $\mathsf{PA}(n, k)$ compared to networks drawn from $\mathsf{G}(n, m)$ and $\mathsf{G}(n, m, k)$ after 25%, 30%, and 40% of vertices were removed in targeted attacks or random failures, where $n = 10{,}000$, $m = 30{,}000$, and $k = 3$. The dashed lines represent the upper bounds for the robustness score as given in Sect. 2.2

Our results suggest that networks generated by using preferential attachment are consistently more robust against random failures but more fragile against targeted attacks when compared to random networks of the same size. This is underscored by the fact that the points corresponding to the latter networks are located in the fourth quadrants in Fig. 3. However, we note that preferential-attachment networks are always more vulnerable to targeted attacks and random failures compared to random networks of the same size whose minimum degree is at least as large. This is underscored by the fact that the points corresponding to the latter networks are located in the first quadrants in Fig. 3. We note that our claimed patterns generally hold even when n or k are varied; however, the significance of the trends we assert increases as n increases, while it decreases as k increases. In other words, the patterns we discuss are most evident for more massive and sparser networks, which, are generally of greater relevance.

Its positive and negative values represent a tendency of elements in X to reach values above and below the reference mean μ_Y, respectively.

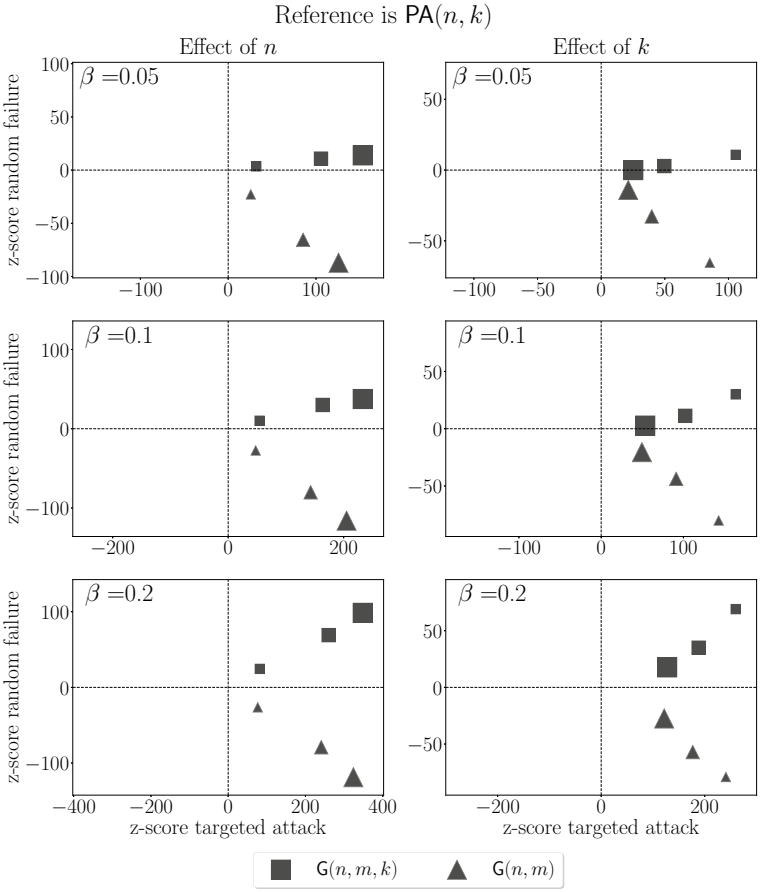


Fig. 3 In the column on the left, the larger marker sizes correspond to the larger values of $n \in \{1000, 10{,}000, 20{,}000\}$ for fixed $k = 3$. In the column on the right, the larger marker sizes correspond to the larger values of $k \in \{3, 4, 5\}$ for fixed $n = 10{,}000$

4.3 Consistency of Near-Optimal Robustness

In our experiments, we have used networks drawn from $\mathsf{G}(n, m, k)$, where $m = nk$. For $n = 10{,}000$, $k = 3$, we have seen in Figs. 1 and 2 that these networks exhibit near-optimal robustness against targeted attacks and random failures when only a moderate fraction of their vertices are removed. In this section, we evaluate the sensitivity of this pattern to variations in n, k, and β. To this end, we consider nine combinations of n, k with $n \in \{1000, 10{,}000, 20{,}000\}$ and $k \in \{3, 4, 5\}$. Then we draw 100 networks from $\mathsf{G}(n, m, k)$ and compute their average robustness score normalized to the maximum

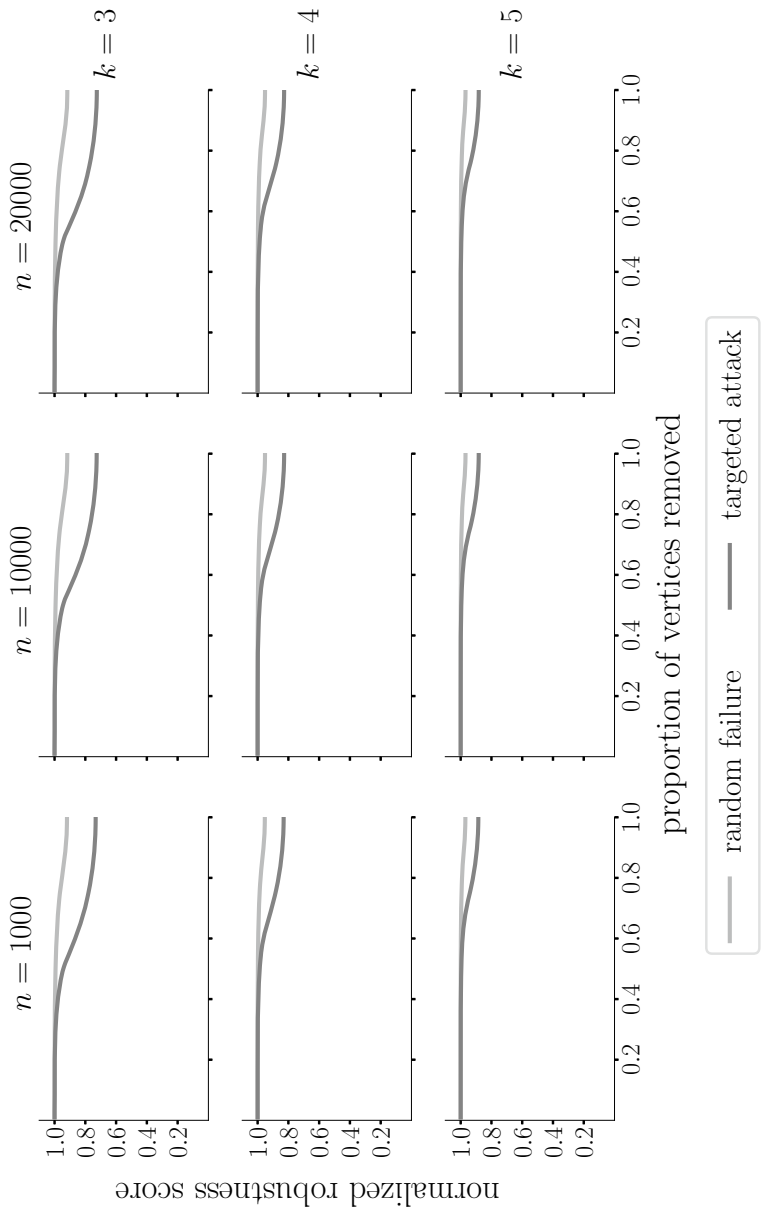


Fig. 4 The expected robustness score of networks drawn from $\mathsf{G}(n, m, k)$ normalized by the maximum achievable robustness score when $\beta \in \{0, 0.01, \ldots, 0.99, 1\}$ fraction of the vertices are removed by targeted attacks or random failures. Here, we consider $m = nk$ for each fixed pair of $n \in \{1000, 10{,}000, 20{,}000\}$, $k \in \{3, 4, 5\}$

achievable robustness score when $\beta \in \{0, 0.01, \ldots, 0.99, 1\}$ portion of the vertices are removed under targeted attacks or random failures.[3] The result is shown in Fig. 4.

When no more than 40% of vertices are removed by targeted attacks or random failures, Fig. 4 illustrates the consistent near-optimal robustness of networks drawn from $\mathsf{G}(n, m, k)$ in the case $m = nk$ for a constant $k \geq 3$. Moreover, we note that this 40% threshold does not depend appreciably on n but increases with k. For example, when $k = 5$, we can observe across different n that the robustness score does not noticeably deviate from its optimal value as long as no more than 60% of vertices are removed in targeted attacks or random failures. From the discussions here, we can conclude that for a constant $k \geq 3$ and $m = nk$, the near-optimal robustness of networks drawn from $\mathsf{G}(n, m, k)$ is consistent as long as the fraction of vertices removed by targeted attacks or random failures is not too large.

5 Conclusions

We have shown that, for any constant $k \geq 3$, almost all graphs in which the number of edges is linear in the number of vertices (i.e., the average degree is upper-bounded by a constant) and the minimum degree is at least k, are connected and provably robust against any vertex removal strategy. Motivated by this new theoretical result, we have shown experimentally that the dictum "robust-yet-fragile" is not a suitable characterization of preferential-attachment networks, let alone scale-free networks in general, because it stems from a poorly chosen baseline. It appears that the previously assessed robustness is largely due to their constant minimum degree, rather than their skewed degree distribution.

Although the characterization of scale-free networks as "robust-yet-fragile" originates mainly from experiments with preferential-attachment instances [1, 9], only a vanishing fraction of scale-free networks can be generated by preferential attachment [17]. While a straightforward extension of our work would consider the robustness of more general classes of scale-free networks, it will also be interesting to study properties other than robustness.

References

1. Albert, R., Jeong, H., Barabási, A.L.: Error and attack tolerance of complex networks. Nature **406**, 378–382 (2000)
2. Anastos, M., Frieze, A.: Hamilton cycles in random graphs with minimum degree at least 3: an improved analysis. Random Struct. Algorithms **57**(4), 865–878 (2020)
3. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)

[3] Note that for any connected graph, the maximum achievable robustness score after removing β portion of the vertices is $1 - \beta/2 + o(1)$, as given in Sect. 2.2.

4. Batagelj, V., Brandes, U.: Efficient generation of large random networks. Phys. Rev. E **71**(3), 036113 (2005)
5. Batagelj, V., Zaveršnik, M.: Fast algorithms for determining (generalized) core groups in social networks. Advan. Data Anal. Classif. **5**(2), 129–145 (2011)
6. Ben-Shimon, S., Krivelevich, M.: Vertex percolation on expander graphs. Eur. J. Comb. **30**(2), 339–350 (2009)
7. Benjamini, I., Kozma, G., Wormald, N.: The mixing time of the giant component of a random graph. Random Struct. Algorithms **45**(3), 383–407 (2014)
8. Bollobás, B., Fenner, T.I., Frieze, A.M.: Hamilton cycles in random graphs of minimal degree at least k. In: Baker, A., Bollobás, B., Hajnal, A. (eds.) A Tribute to Paul Erdős, pp. 59–96. Cambridge University Press (1990)
9. Doyle, J.C., Alderson, D.L., Li, L., Low, S., Roughan, M., Shalunov, S., Tanaka, R., Willinger, W.: The "robust yet fragile" nature of the internet. Proc. Nat. Acad. Sci. (PNAS) **102**(41), 14497–14502 (2005)
10. Erdős, P., Rényi, A.: On random graphs I. Publicationes Mathematicae Debrecen **6**, 290–297 (1959)
11. Hasheminezhad, R., Boudourides, M., Brandes, U.: Scale-free networks need not be fragile. In: Proceedings of the 2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. pp. 332–339. IEEE (2020)
12. Hasheminezhad, R., Brandes, U.: Constructing provably robust scale-free networks. In: Ribeiro, P., Silva, F., Mendes, J.F., Laureano, R. (eds.) Network Science. pp. 126–139. Springer International Publishing (2022)
13. Janson, S., Luczak, M.J.: Asymptotic normality of the k-core in random graphs. Ann. Appl. Probab. **18**(3), 1085–1137 (2008)
14. Klau, G.W., Weiskircher, R.: Robustness and resilience. In: Brandes, U., Erlebach, T. (eds.) Network Analysis, Lecture Notes in Computer Science (LNCS), vol. 3418, pp. 417–437. Springer (2005)
15. Lountzi, A.: Expander Graphs and Explicit Constructions. Master's thesis, Uppsala University, Algebra and Geometry (2015)
16. Molloy, M., Reed, B.: The size of the giant component of a random graph with a given degree sequence. Comb., Probab. Comput. **7**(3), 295–305 (1998)
17. Petersen, C., Rotbart, N., Simonsen, J.G., Wulff-Nilsen, C.: Near optimal adjacency labeling schemes for power-law graphs. In: Chatzigiannakis, I., Mitzenmacher, M., Rabani, Y., Sangiorgi, D. (eds.) 43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016). Leibniz International Proceedings in Informatics (LIPIcs), vol. 55, pp. 133:1–133:15. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik (2016)
18. Schneider, C.M., Moreira, A.A., Andrade, J.S., Jr., Havlin, S., Herrmann, H.J.: Mitigation of malicious attacks on networks. Proc. Nat. Acad. Sci. (PNAS) **108**(10), 3838–3841 (2011)

The Vertex-Edge Separator Transformation Problem in Network-Dismantling

Xiao-Long Ren

Abstract In complex networks, network-dismantling aims at finding an optimal set of nodes (or edges) such that the removal of the set from the network will lead to the disintegration of the network, that is, the size of the giant/largest connected component is not bigger than a specific threshold (for instance, 1% of the original network size). Existing algorithms addressing this topic can be divided into two closely related but different categories: vertex separator-oriented algorithms and edge separator-oriented algorithms. There has been a lot of research on these two categories, respectively. However, to the best of our knowledge, less attention has been paid to the relation between the vertex separator and edge separator. In this paper, we studied the separator transformation (ST) problem between the separator of the vertexes and edges. We approximated the transformation from edge separator to vertex separator using Vertex Cover algorithm, while approximated the transformation from vertex separator to edge separator using an Explosive Percolation (EP) approach. Moreover, we further analyzed the results of the vertex-edge separator transformation through the explosive percolation method in detail. The transformation problem in network-dismantling opens up a new direction for understanding the role of the vital nodes set and edges set as well as the vulnerability of complex systems.

Keywords Network-dismantling · Graph partition · Vertex separator · Edge separator · Vertex cover · Explosive percolation

X.-L. Ren (✉)
Yangtze Delta Region Institute (Huzhou), University of Electronic Science and Technology of China, Huzhou 313001, P. R. China
e-mail: renxiaolong@csj.uestc.edu.cn

Computational Social Science, ETH Zürich, Zurich 8092, Switzerland

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_36

1 Introduction

Numerous types of complex systems, such as the Internet, social media, power grids, transport systems, world trading systems, et al., can be represented as networks (or graphs), composed of a set of nodes and a set of edges [1, 2]. Because of the heterogeneous nature of these complex networks (systems), different nodes and edges usually play different roles [3–5]. In most real networks, a small part of nodes and edges are critical for their structure and function. Examples include communication networks that affect the spreading of messages and behaviors [4, 6, 7], the role of superspreaders for the propagation of rumors or diseases [8, 9], drivers to control a network [10–12], or vital nodes to maintain the connectivity and flow of a complex network [5, 13, 14], et al.

Among these hot topics, one fundamental challenge in the complex networks is to identify an optimal set of nodes (or edges), whose removal would dismantle the network such that the size of the giant connected component (GCC) is at most C, known as C-dismantling problem [15–18]. The removed set is referred to as node separators (or edge separators). The removal (or deactivation, or immunization) of the separators from the complex networks, usually a very small set, will induce some fundamental changes to the structure and the function of the networks.

The network-dismantling problem has wide applications in a broad spectrum of social-economic scenarios. Finding the most efficient network-dismantling strategy at minimum overall costs belongs to the NP-hard class [16, 17], which means that we cannot find the optimal vertex or edge separator with minimum removal cost in a nondeterministic polynomial time. This indicates that there doesn't exist an efficient algorithm that can find the exact optimal solution for large-scale networks. Taking a step back, many researchers proposed algorithms to approximate the optimal solution of the network-dismantling problem. Some widely used algorithms are based on network centrality [19–21], spreading and cascading theory [22, 23], spin-glass and optimal percolation theory [16, 24, 25], as well as the linear programming [26], semidefinate programming [27], spectral partitioning [28, 29] and even the deep reinforcement learning techniques [18, 30].

In terms of objects removed, the existing algorithms in this direction can be roughly divided into two closely related but different categories: vertex (node) separator algorithms and edge (link) separator algorithms. There has been a lot of research on these two categories recently. A summary of the progress on the network-dismantling problem can be found in Refs. [31–35].

However, to the best of our knowledge, less attention has been paid to the relation between the vertex separator and edge separator. Only recently have people approximated the vertex separator based on edge sets (not edge separators) using the weighted vertex cover approach [17, 36]. In this article, we focused on the transformation problem between the separator of the vertices and edges. The separator-transformation problem includes two sub-problem: (1) the transformation from edge separator to vertex separator, and (2) the transformation from vertex separator to edge separator. In this article we introduce the algorithm of the Vertex Cover to approximate the

transformation from edge separator to vertex separator, and then, approximate the transformation from vertex separator to edge separator using an Explosive Percolation (EP) approach in detail.

2 The Transformation of the Dismantling Set of Nodes and Edges

2.1 Sub-Problem 1: Transformation From Edge Separator to Vertex Separator

In this subsection, we will introduce the first sub-problem of the separator-transformation in network-dismantling, going from the edge separator to the vertex separator. For a network $G = (V, E)$, given an edge separator S_e, how can we get the approximate optimal vertex separator S_v based on S_e? One straightforward but ineffective approach is to randomly remove one of the endpoints of each edge in set S_e. The size of the vertex set found in this way roughly equals to the size of S_e. In this situation, the reinsertion fine-tuning technique [9, 37] can be applied to get a better node-removal set.

A further idea is to find a set of nodes that can cover all the edges in S_e. This is the famous Vertex Cover problem in graph theory. A vertex cover of S_e is a set of vertices that includes at least one endpoint of every edge in S_e. The problem of finding a minimum vertex cover of a set of edges is a typical example of an NP-hard optimization problem. If we study this separator-transformation in generalized network-dismantling [17], which considers non-unit removal cost, this correspond to the Weighted Vertex Cover problem. In the GND algorithm [17], we adopted a linear time 2-approximation algorithm [38] to find the node removal set with almost least removal cost. The pseudocode of the weighted vertex cover algorithm can be find in the Supplementary Information (SI) of ref. [17]. We will not discuss this in depth here.

Please note that the Vertex Cover problem in a bipartite graph can be done optimally via maximum matching according to the König's theorem [39, 40]. In this sub-problem, however, the edge separator structure may not necessarily be translated to an exact bipartite graph, for some edges may only appear on one side of the bipartite graph. But this bipartite approach will undoubtedly prompt the solving of this sub-problem. We are continuing to work on a fix for this issue.

2.2 Sub-Problem 2: Transformation From Vertex Separator to Edge Separator

In this subsection, we will address the second sub-problem of the separator-transformation, going from the vertex separator to the edge separator. For a network

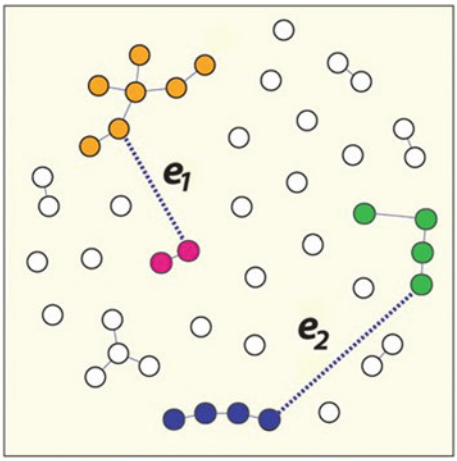


Fig. 1 Illustration of the explosive percolation process in network evolution comes from [41]. Under the sum rule of the explosive percolation, among total $K = 2$ samples, the edge (e_2) that minimizes the sum of the cluster sizes will be selected to reinsert to the network

$G = (V, E)$, given a vertex separator S_v, how can we get the optimal edge separator S_e according to S_v?

Remove all the vertices in S_v actually means removing all the edges connected with the vertices in S_v. In terms of edge removal, this kind of operation removes too much edges. This, however, can help us to narrow down the search scope in the process of find the suitable set of edges. That is to say, we only need to find a subset of the edges that connected with any vertices in S_v, whose removal can break up the original network and make the GCC size of the remaining network smaller than a specified threshold C.

Let's make above analysis in a reverse way. For a network $G = (V, E)$ and a vertex separator S_v, the remaining network G^* is the network after removing the vertex separator S_v from G. The size of the largest cluster of G^* is smaller than C. Usually G^* is composed of plenty of disconnected clusters. According to the above discussion, from the perspective of edge removal-based dismantling, too many edges were removed from G^*, as some of them were removed unnecessarily. One straightforward method to deal with this situation is to reinsert the unnecessarily removed edges into the remaining network. In our approach, the process of adding edges is similar to the **sum rule-based explosive percolation (EP) process** in complex networks [41], that is, in every time step, among a total of K randomly selected edges, the edge minimizing the sum of the sizes of the clusters it merges will be selected to reinsert to the network (see Fig. 1).

Inspired by the sum rule-based explosive percolation in the network evolution process, we propose the following Explosive Percolation-based Transformation Algorithm to solve the problem of transformation from vertex separator to edge separator:

Explosive Percolation-based Transformation Algorithm
Input: Network G, vertex separator S_v, sample size K

Output: Edge separator S_e
Method:

1. Get the remaining network G^* by removing S_v from G. Denote the set of the removed edges as R_e.
2. Randomly select K edges from R_e so that for each edge, the sum of the sizes of the two clusters it connected with is smaller than the dismantling threshold C.
3. Select the edge minimizing the sum of the sizes of the two clusters among the K samples in step 2, and reinsert it into G^*. Update R_e, G^* and the size of the clusters in G^*.
4. Repeat step 2 and step 3 until there is no more edge that can be reinserted. All the remaining edges of R_e compose the edge separator S_e. Output S_e.

From the above algorithm, the node separator can be transformed into an edge separator which needs much less removal cost to dismantle the network when the edge removal is feasible. In the following section, we will analyze the performance of the above transformation from node separator to edge separator in detail.

3 Results

Before starting the detailed discussion, we will introduce several popular dismantling algorithms used in this paper. Betweenness [42] is a classical vertex ranking algorithm based on shortest paths. The Betweenness centrality of a vertex is the number of the shortest paths that go through the vertex. Betweenness is widely used to solve the network dismantling problems [33]. BPD stands for belief-propagation decimation algorithm [24], which dismantles a network by finding and deleting the approximate solution set of the feedback vertex set (FVS) problem to break all the loops in the networks. BPD then breaks big trees until only small trees are left. Lastly, BPD optimizes the vertex separator by restoring some nodes to the graph. Along similar lines, the Min-Sum algorithm [16] breaks all the loops in the 2-core of the network by Min-Sum message passing [43], following with tree breaking and greedy reintroduction of cycles. CoreHD [21] is a simple and fast decycling-based method that dismantles networks by removing the node with the highest degree from the remaining 2-core of the network progressively until there is no 2-core structure remaining. Finally, CoreHD algorithm ends up with tree breaking and vertex reinsertion. GNDR is short for Generalized Network Dismantling with Reinsertion algorithm, which is based on the spectral properties of a node-weighted Laplacian operator [17, 36]. GNDR allows the removal cost of networks to take arbitrary non-negative real values. A detailed description of these algorithms can be found in Chap. 2 of ref. [44]. Here we mainly focus on the performance of solutions to the transformation problems.

Firstly, we compare the performance of the node removal-based dismantling algorithm and its EP-based transformation. In Fig. 2, taking the US Airport network as an example, we set the dismantling threshold as 50% of the original size of the network to get a better visualization. In the upper subfigure, the red solid line corresponds

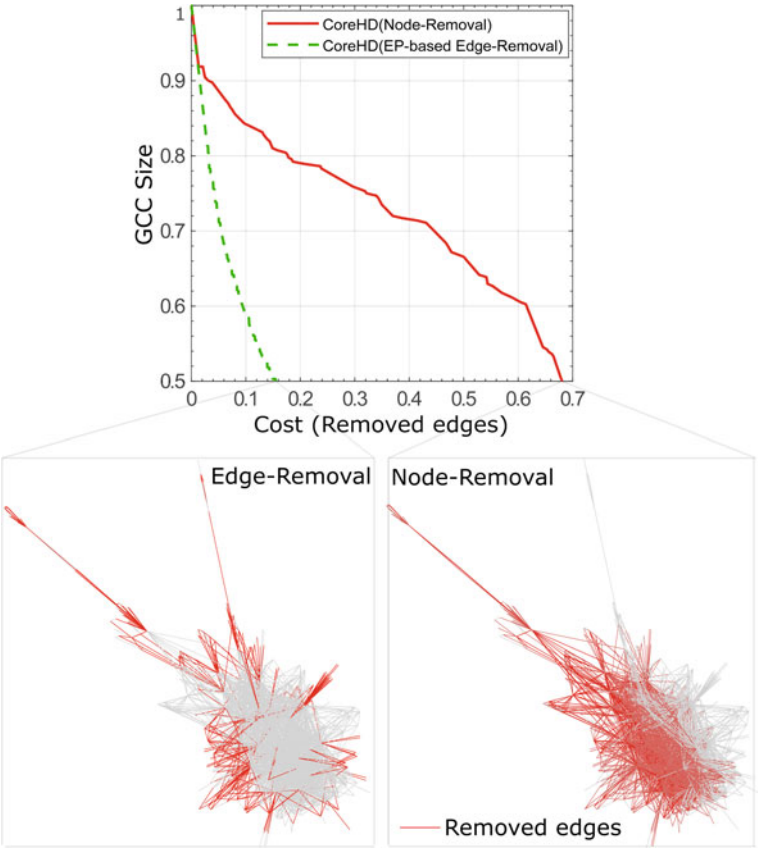


Fig. 2 Comparison of a vertex(node) separator-oriented algorithm and its explosive percolation-based edge separator-oriented counterpart, the CoreHD example. To get a better visualization, we set the threshold of the GCC of the dismantling algorithm to be 50% and then colored the removed edges with red. It is obvious that to reach the same dismantling target size, the edge-based removal saves a lot of removal costs

to the stander CoreHD algorithm [21]. The green dashed line is the result of its EP-based edge separator. In the two subfigures below, the removed edges are represented in red.

Furthermore, we compare the classical, node removal-based algorithms with their edge removal results produced by using the EP-based transformation approach. In Fig. 3, taking the Powergrid network as an example, we compared the CoreHD algorithm [21], BPD algorithm [24], Min-Sum algorithm [16], and GNDR algorithms [17] (red solid lines) with their EP-based edge separators (green dashed lines), respectively. We set the dismantling threshold C at the 1% of the original size of the network. From both the ratio of the removed cost and the area under the curve (AUC) (see the definition in [44]) perspective, the removal costs of the EP-based edge separators

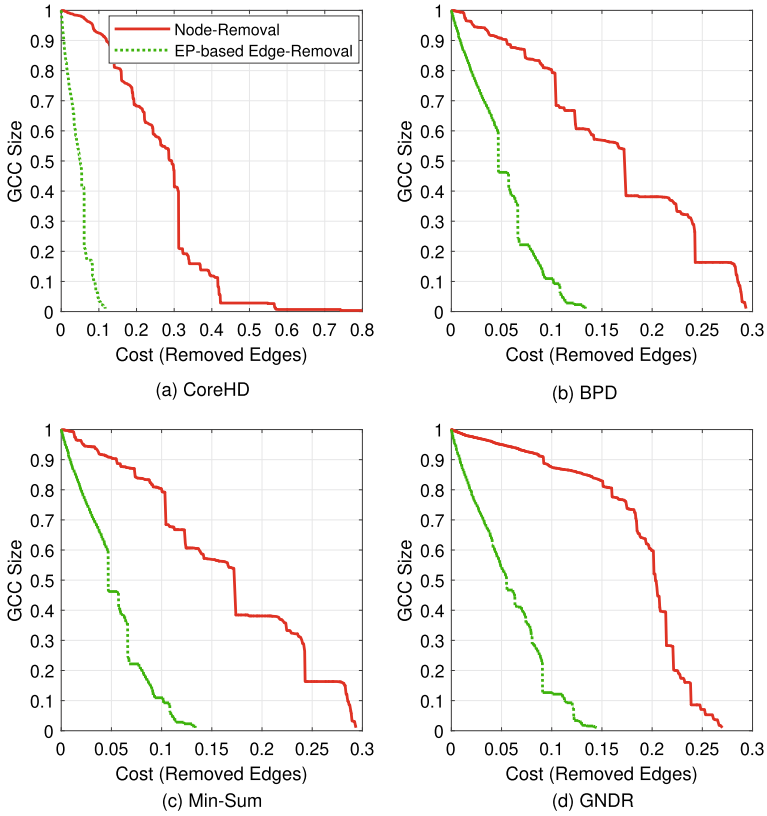


Fig. 3 Performance of the edge-removal strategies transformed from vertex-separator using the explosive percolation approach defined in Sect. 2.2. The vertex-separators are produced by CoreHD, BPD, Min-Sum, and GNDR algorithm, respectively. After the transformation, the overall removal costs have been greatly reduced

have been greatly reduced. This is result of identifying and removing necessary edges by the EP-based edge separator, instead of removing all the edges connected to the node separator.

From above two figures, we can easily come to this conclusion: Comparing with the classical node removal-based dismantling algorithm, the EP-based transformation algorithm can efficiently obtain the corresponding edge separators, which can save a lot of removal costs while reaching the same dismantling threshold.

In Fig. 4, we analyse the impact of the sample size K on the performance of the EP-based transformation algorithm. In the sum rule-based explosive percolation, with larger sample size K of the randomly selected edges, the algorithm can find the better edge separator. Fig. 4 verifies this point. But larger K also requires more computational time. Thus, for all the other result in this paper, we set $K = 10$.

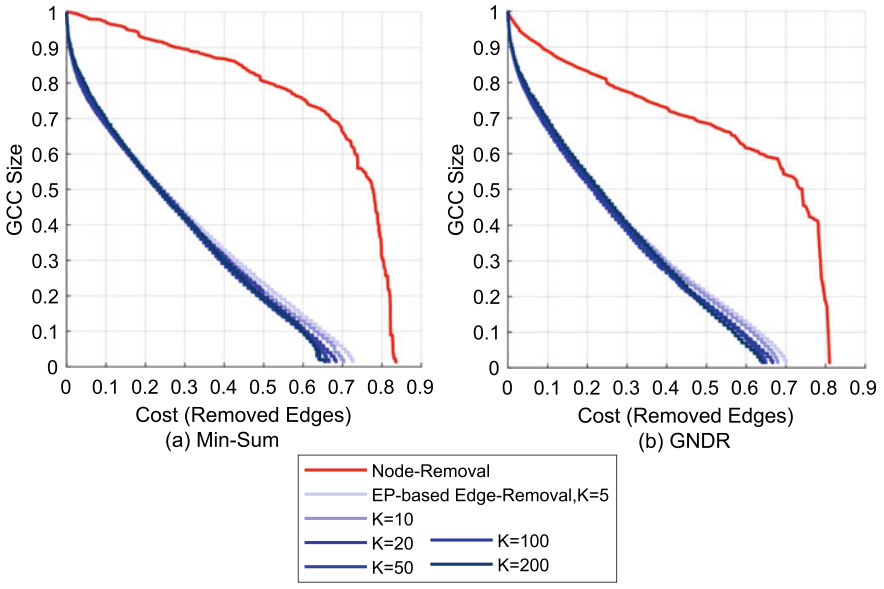


Fig. 4 Performance of the proposed explosive percolation-based approach with the different values of sampling size K, the Petster network example. It is straightforward that larger K leads to better performance, but requiring more computational time

Finally, in Fig. 5, we compare the original Node Betweenness, original Edge Betweenness, and the EP-based edge separator of the Node Betweenness algorithm. Taking the Petster network as an example, we set the dismantling threshold C at 1% of the network size. This result shows that the EP-based edge removal method is much better than the original Edge Betweenness algorithm in both indicators — the ratio of the removed cost and the AUC.

4 Conclusion

In this paper, we firstly studied the transformation problem between the vertices and edges separator and approximated the transformation going from the edge separator to the vertex separator using a Vertex Cover algorithm, and approximated the transformation from the vertex separator to the edge separator using an Explosive Percolation approach. Then, we further analyzed the results of the vertex-edge separator transformation through the explosive percolation method in detail. The transformation problem in network-dismantling opens up a new direction for understanding the role of the set of vital nodes and edges [5] as well as the vulnerability [45, 46] of complex systems.

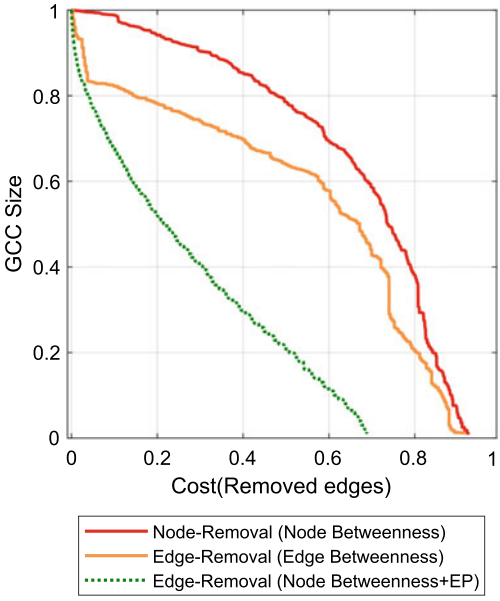


Fig. 5 Comparison of the betweenness-based dismantling algorithms and its variants. Red line: Removed nodes according to Node Betweenness Centrality. Yellow line: Removed edges according to Edge Betweenness Centrality. Green dash line: Removed edges according to the result of the transformation from the node separator (the red line).

Acknowledgements The author would like to thank Prof. Dirk Helbing and Dr. Nino Antulov-Fantulin from ETH Zurich for their feedback and suggestions on this study. The author would like to thank the anonymous reviewers for their comments. This work was supported by the Postdoctoral Research Fund of China (No. 2022M710620), Natural Science Foundation of Sichuan Province (23NSFSC3624), and Science and Technology Foundation of Huzhou (No. 2021YZ12).

References

1. Barabási, A.-L.: Network Science. Cambridge University Press (2016)
2. Newman, M.E.J.: Networks. Oxford University Press (2018)
3. Moreno, Y., Pastor-Satorras, R., Vespignani, A.: Epidemic outbreaks in complex heterogeneous networks. Eur. Phys. J. B-Condens. Matter Complex Syst. **26**(4), 521–529 (2002)
4. Gladwell, M.: The Tipping Point: How Little Things can Make a Big Difference. Little, Brown (2006)
5. Lü, L., Chen, D., Ren, X.-L., Zhang, Q.-M., Zhang, Y.-C., Zhou, T.: Vital nodes identification in complex networks. Phys. Rep. **650**, 1–63 (2016)
6. Jiang, J., Huang, Z.-G., Seager, T.P., Lin, W., Grebogi, C., Hastings, A., Lai, Y.-C.: Predicting tipping points in mutualistic networks through dimension reduction. Proc. Nat. Acad. Sci. U.S.A. **115**(4), E639–E647 (2018)
7. Liu, X., Li, D., Ma, M., Szymanski, B.K., Stanley, H.E., Gao, J.: Network resilience. Phys. Rep. **971**, 1–108 (2022)
8. Kitsak, M., Gallos, L.K., Havlin, S., Liljeros, F., Muchnik, L., Stanley, H.E., Makse, H.A.: Identification of influential spreaders in complex networks. Nat. Phys. **6**(11), 888–893 (2010)
9. Morone, F., Makse, H.A.: Influence maximization in complex networks through optimal percolation. Nature **524**(7563), 65–68 (2015)

10. Liu, Y.-Y., Barabási, A.-L.: Control principles of complex systems. Rev. Mod. Phys. **88**(3), 035006 (2016)
11. Amani, A.M., Jalili, M., Yu, X., Stone, L.: Controllability of complex networks: choosing the best driver set. Phys. Rev. E **98**(3), 030302 (2018)
12. Zhang, Y., Garas, A., Schweitzer, F.: Control contribution identifies top driver nodes in complex networks. Advan. Complex Syst. **22**(07n08), 1950014 (2019)
13. Del Ferraro, G., Moreno, A., Min, B., Morone, F., Pérez-Ramírez, Ú., Pérez-Cervera, L., Parra, L.C., Holodny, A., Canals, S., Makse, H.A.: Finding influential nodes for integration in brain networks using optimal percolation theory. Nat. Commun. **9**(1), 2274 (2018)
14. Tahmassebi, A., Amani, A.M., Pinker-Domenig, K., Meyer-Baese, A.: Determining disease evolution driver nodes in dementia networks. In: Medical Imaging 2018: Biomedical Applications in Molecular, Structural, and Functional Imaging, vol. 10578, p. 1057829. International Society for Optics and Photonics (2018)
15. Janson, S., Thomson, A.: Dismantling sparse random graphs. Comb. Probab. Comput. **17**(2), 259–264 (2008)
16. Braunstein, A., Dall'Asta, L., Semerjian, G., Zdeborová, L.: Network dismantling. Proc. Nat. Acad. Sci. U.S.A. **113**(44), 12368–12373 (2016)
17. Ren, X.-L., Gleinig, N., Helbing, D., Antulov-Fantulin, N.: Generalized network dismantling. Proc. Nat. Acad. Sci. U.S.A. **116**(14), 6554–6559 (2019)
18. Fan, C., Zeng, L., Sun, Y., Liu, Y.-Y.: Finding key players in complex networks through deep reinforcement learning. In: Nature Machine Intelligence, pp. 1–8 (2020)
19. Albert, R., Jeong, H., Barabási, A.-L.: Error and attack tolerance of complex networks. Nature **406**(6794), 378–382 (2000)
20. Brandes, U.: On variants of shortest-path betweenness centrality and their generic computation. Soc. Netw. **30**(2), 136–145 (2008)
21. Zdeborová, L., Zhang, P., Zhou, H.-J.: Fast and simple decycling and dismantling of networks. Sci. Rep. **6**, 37954 (2016)
22. Motter, A.E., Lai, Y.-C.: Cascade-based attacks on complex networks. Phys. Rev. E **66**(6), 065102 (2002)
23. Altarelli, F., Braunstein, A., Dall'Asta, L., Wakeling, J.R., Zecchina, R.: Containing epidemic outbreaks by message-passing techniques. Phys. Rev. X **4**(2), 21024 (2014)
24. Mugisha, S., Zhou, H.-J.: Identifying optimal targets of network attack by belief propagation. Phys. Rev. E **94**, 012305 (2016)
25. Qin, S.-M., Ren, X.-L., Lü, L.: Efficient network dismantling via node explosive percolation. Commun. Theor. Phys. **71**(6), 764 (2019)
26. Leighton, T., Rao, S.: Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. J. ACM **46**(6), 787–832 (1999)
27. Arora, S., Rao, S., Vazirani, U.: Expander flows, geometric embeddings and graph partitioning. J. ACM (JACM) **56**(2), 1–37 (2009)
28. Fiedler, M.: Algebraic connectivity of graphs. Czech. Math. J. **23**(2), 298–305 (1973)
29. Guattery, S., Miller, G.L.: On the quality of spectral separators. SIAM J. Matrix Anal. Appl. **19**(3), 701–719 (1998)
30. Grassia, M., De Domenico, M., Mangioni, G.: Machine learning dismantling and early-warning signals of disintegration in complex systems. Nat. Commun. **12**(1), 1–10 (2021)
31. Bichot, C.-E., Siarry, P.: Graph Partitioning. Wiley (2013)
32. Buluç, A., Meyerhenke, H., Safro, I., Sanders, P., Schulz, C.: Recent advances in graph partitioning. In: Algorithm Engineering, pp. 117–158 (2016)
33. Wandelt, S., Sun, X., Feng, D., Zanin, M., Havlin, S.: A comparative analysis of approaches to network-dismantling. Sci. Rep. **8**(1), 1–15 (2018)
34. Pei, S., Wang, J., Morone, F., Makse, H.A.: Influencer identification in dynamical complex systems. J. Complex Netw. **8**(2) (2019)
35. Bellingeri, M., Bevacqua, D., Scotognella, F., Alfieri, R., Nguyen, Q., Montepietra, D., Cassi, D.: Link and node removal in real social networks: a review. Front. Phys. **8**, 228 (2020)

36. Ren, X.-L., Antulov-Fantulin, N.: Ensemble approach for generalized network dismantling. In: International Conference on Complex Networks and Their Applications, pp. 783–793. Springer (2019)
37. Fan, C., Zeng, L., Feng, Y., Xiu, B., Huang, J., Liu, Z.: Revisiting the power of reinsertion for optimal targets of network attack. J. Cloud Comput. **9**(24), 1–13 (2020)
38. Bar-Yehuda, R., Even, S.: A linear-time approximation algorithm for the weighted vertex cover problem. J. Algorithms **2**(2), 198–203 (1981)
39. König, D.: Gráfok és mátrixok. Matematikai és Fizikai Lapok **38**(1031), 116–119 (1931)
40. Pothen, A., Simon, H.D., Liou, K.-P.: Partitioning sparse matrices with eigenvectors of graphs. SIAM J. Matrix Anal. Appl. **11**(3), 430–452 (1990)
41. Achlioptas, D., D'Souza, R.M., Spencer, J.: Explosive percolation in random networks. Science **323**(5920), 1453–1455 (2009)
42. Brandes, U.: A faster algorithm for betweenness centrality. J. Math. Sociol. **25**(2), 163–177 (2001)
43. Altarelli, F., Braunstein, A., Dall'Asta, L., Zecchina, R.: Optimizing spread dynamics on graphs by message passing. J. Stat. Mech.: Theory Exp. **2013**(09), P09011 (2013)
44. Ren, X.-L.: The dismantling problem in complex networks and its applications. Ph.D. thesis. ETH Zurich (2021)
45. Holme, P., Kim, B.J., Yoon, C.N., Han, S.K.: Attack vulnerability of complex networks. Phys. Rev. E **65**, 056109 (2002)
46. Trajanovski, S., Scellato, S., Leontiadis, I.: Error and attack vulnerability of temporal networks. Phys. Rev. E **85**, 066105 (2012)

Network Analysis

Gig Economy and Social Network Analysis: Topology of Inferred Network

Gustavo Pilatti, Flavio L. Pinheiro, and Alessandra Montini

Abstract Unparalleled advances in information technology have resulted in the virtualization of the workplace, as well as in a surge in non-traditional work arrangements based on short-term contracts ("gigs"). Work that is done remotely through online platforms may be hidden by technology. Thus, how could we possibly access information about the social network of workers? What are the business and social values hidden in workers' underlying informal social networks? Here, we propose applying methods from complex network analysis to have an overview of data from a Brazilian food-delivery company. We present the steps used to make the inference of a social network that relates delivery men according to their co-location patterns. Hence, the obtained network offers a valuable framework to explore, in the future, questions related to the role of informal social networks in the spreading of innovations and the coordination of behaviors and business strategies.

Keywords Informal social network · Complex network · Gig economy

1 Introduction

In many practical and theoretical fields, the capacity to infer and characterize social interactions and their impact is extremely useful. Indeed, the common premise of network science is that the structure of the network plays an important role in the emergent behavior of the networked system [7]. Often, the social network is not directly observed and must be inferred from human digital traces (e.g., emails, chats,

G. Pilatti (✉) · A. Montini
Universidade de São Paulo, Sao Paulo, Brazil
e-mail: pilatti@usp.br

A. Montini
e-mail: amontini@usp.br

F. L. Pinheiro
Universidade Nova de Lisboa, Lisbon, Portugal
e-mail: fpinheiro@novaims.unl.pt

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_37

meetings) or surveyed data [3, 12]. But there is a gap in mapping the relations in the offline environment when there exists an absence of information available. And this problem is clear in the gig economy, in which it is difficult to gather digital data from the participants and, thereby, build their social network and come up with all the possible insights this information could bring to both workers and platforms.

Gig economy workers usually perform a standard task determined by an online platform under a temporary and flexible work arrangement. While it may be seen as an easy-to-perform job or as a commodity task, valuable knowledge is generated with experience. As the information available to traditional industries is not available in this context, it is not possible to perform a social network analysis. Our goal is to understand, infer and analyze the social patterns of the gig economy workers using network analysis as a framework that could, in future work, help them learn and improve both their well-being and performance.

Past works have used network analysis to study gig economy workers [5, 9]. However, there is at present no single research about their social network. We aim to fill this gap and, as such, contribute to the technical analysis of social networks and also to the social discussion about this new type of job.

Here, we look into the Brazilian food delivery industry and the delivery men's work. In this study, we performed a natural experiment where the gathered information was easily and accurately recorded in the database of a Food Delivery company. In particular, the data includes the geolocation and working status of couriers (i.e., delivery man) and was captured in real-time from the delivery man's mobile device. Thus, studying this information provides a good starting point towards understanding how work happens outside the offices, at the same time some measures have been taken to guarantee the anonymity of the workers.

This research will be developed to describe and characterize the network attributes that are more relevant and characterize its topology. To accomplish this goal, there will be the necessary to study the statistical significance of the connection between the couriers, avoiding spurious factors.

2 Data and Methods

We conducted our study and gathered the data from a popular online food and groceries delivery system (platform) to empirically understand the gig workers' relations. More specifically, this online platform is a mobile application where registered customers choose their food or groceries from listed merchants, and the deliveries are made by couriers that create a profile. Our focus will be on these gig workers, the couriers.

The dataset used in this study contains information on the geolocation of 226 delivery men for a period of 14 d in February 2022. The data was collected from the company's data lake and is from one city with approximately 260k residents (2020), with a platform coverage of 73% of the population.

First we captured all the telemetry data from the delivery men in this period. This data consists of their geolocation every 15 s and their status (allocated in some delivery route, paused, or idle). The original dataset contains more than 837k observations. The geolocation establishes the latitude and longitude considering 1-m precision. Since, GPS from mobile phone devices can have variable accuracy, ranging from 1 to 12 m [1], we grouped the data to obtain an accuracy of 10 m precision and also in 10 min intervals. This way, we are able to keep the privacy of the data. Finally, using their location and time, we matched delivery men according to whether they were co-located or not (i.e., shared the same location).

During the period of two weeks, we captured 3276 events in which groups of 2 to 4 delivery men were co-located. The remaining records (events with a single delivery man) were discarded. Each record of this data panel means that "*Driver i*" was present in a determined latitude and longitude at the same time frame as the "*Driver j*" was also in the same location. Moreover, none of the drivers were allocated in a planned delivery (in other words, we looked for "on route" status as "idle"). Based on the platform operational data, the drivers are idle, on average, 45% of their online time. Our assumption at this point is that being "free" (not allocated to a route), in the same place, at the same time can indicate the potential for social interaction. As an example, we can observe in Fig. 1 the meeting points during these two weeks analyzed. Also, we call attention (blue dots) to the meeting points where high-performance couriers meet (lower fraud costs and high punctuality in deliveries). It is worth highlighting that gig workers only use an app to work, and all other activities done outside this app, like conversations in the messages app, are not recorded or stored by the app, which creates a huge difficulty to map interaction among them, especially virtual interaction.

2.1 Network Statistics

The below network statistics were chosen to describe the delivery men network broadly. The undirected graph itself stands for a static delivery man-delivery man network. Edges represent a courier's relation to another courier and are weighted by the recurrence of the encounter. The nodes are the couriers themselves.

- Average degree (k) is defined as the number of contact points with whom each delivery man *i* was together at the same location and time. It can be associated with the cohesion of the network and may be used as a proxy for individual social capital [10]. Our network has an average degree of 20.58, with 1310 being the sum of unique edges and 226 being the sum of nodes (or delivery men), the distribution is referenced in Fig. 2. This value of k is lower than that observed in ordinary organizations (with workers that have jobs inside the offices with official communication channels such as emails and chats). This difference is supposed to exist, given that we are inferring an informal network [7].
- Average shortest path length (L) is the shortest connection path between two delivery men, averaged over all pairs of delivery men. The literature usually associates

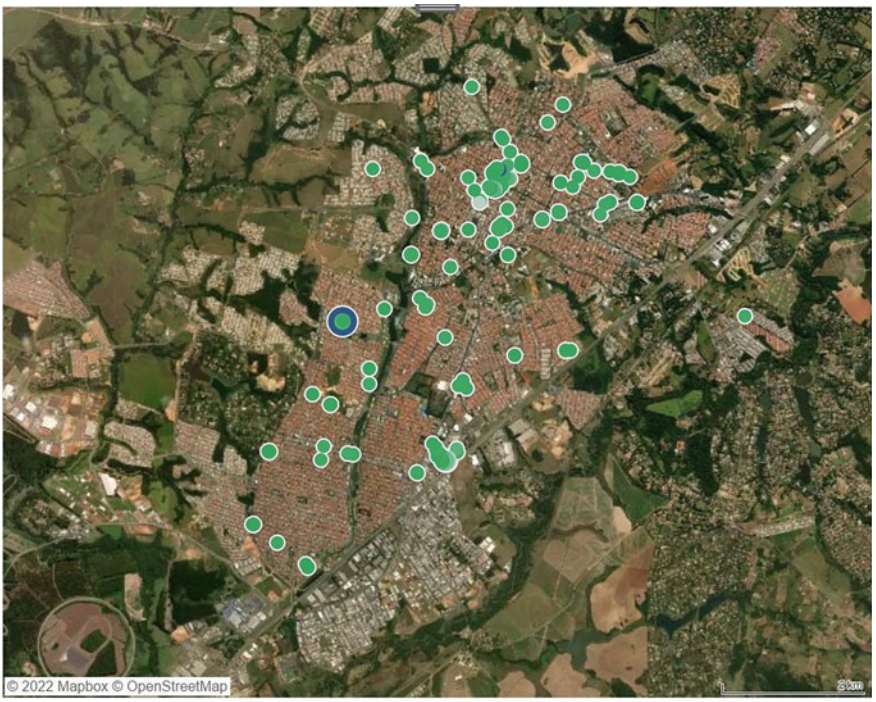


Fig. 1 Geographical location where delivery men met during the analyzed period. The size of the dot denotes the amount of interactions recorded in that place and the blue color denotes delivery men with higher performance in the platform (green denotes lower performance)

these statistics with the ability of an idea or innovation to be spread in the network [7]. These authors bring us an average L of 3.17, and our L is 3.27 for our biggest component (which is represented in Fig. 3. We can infer that this statistic is similar to what is found in organizational networks, even though we are capturing just one way of communication (offline), which can potentially lead to the under-evaluation of the connections.

- Degree assortativity (A) of our network has a value of -0.0044. This statistic quantifies the tendency of each node to be connected to nodes with similar properties in a complex network. Also, the literature shows a correlation between higher assortativity and cooperation among the players of the network [13]. What is possible to infer from a low value of assortativity is that the network is internally connected [15] and it may be considering aleatory drivers encounters. Specifically in this case, what is possible to deduce is that most couriers have a low degree and are connected to drivers that are hubs in the network (as can be seen in Fig. 3), meaning that most meetings occur between drivers with very different degrees.
- Clustering coefficient (C) in our network has a value of 0.2899 and exceeds the expected value for random graphs [14], being similar to organization networks

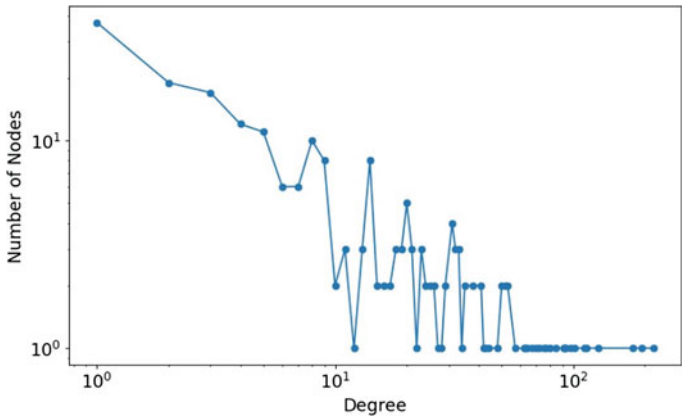


Fig. 2 Degree distribution of the original network using log-log values, where we can identify a concentration of nodes with lower degrees, what can be inferred as occasional encounters among delivery men

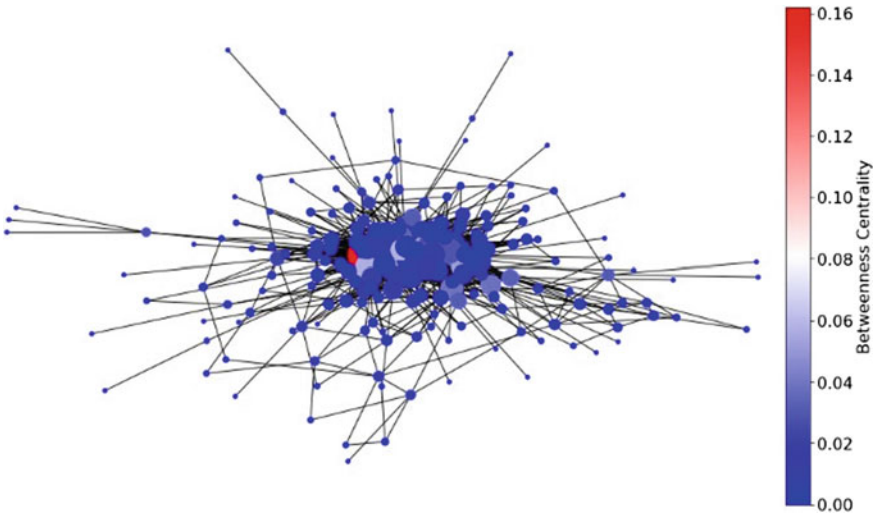


Fig. 3 The betweenness centrality is represented in the color of the above graph and the size of the nodes represents their degrees (min:1 max:20)

[7]. This statistic is a measure of density, showing that if the couriers all know each other, we will have a high clustering coefficient. If the couriers don't know each other, then we are going to have a low clustering coefficient. In organizational literature, this statistic is associated with the idea of the existence of open communication channels among the players.

2.2 Network Significance

Considering that the interaction between the nodes influences the importance of the nodes, we may define a statistically significant connection as a connection where the probability of finding a delivery man that connects two other delivery men is larger than what we would expect based on the prevalence of these delivery men alone: P(i, j) > P(i) P(j). Our main idea is the importance of nodes being linked to their relative position in the network and the correlations with each other. The correlation is calculated using the standard method of ϕ correlation [2, 8].The steps followed to find the statistically significant values are as follows.

The first step, as described above, was to construct an undirected weighted graph, where each node is a delivery man and each edge is the simultaneous presence of both delivery men in the same place, at the same time. This built graph can be represented as a network G developed from N transactions and can be denoted by $G = (D, E)$, where D is a set of d nodes and E is a set of edges, thus, we may represent $E \subseteq D \times D$. With this definition, it is possible to create a $d \times d$ symmetric matrix $A_G = (a_{ij})_{((ij)\in D\times D)}$ that is called the adjacency matrix of G. In this matrix, every $a_{ij} > 0$ if $(i, j) \in E$ is an edge of G, and $a_{ij} = 0$ if $(i, j) \notin E$, or, in other words, this edge does not exist. In addition, $a_{ij} = 0$ if i=j.

Then, given the above adjacency matrix G, we may calculate the ϕ correlation, that we call PCC (from Phi-Correlation Coefficient). The PCC_{ij} is calculated as:

$$PCC_{ij} = \frac{((a_{ij} * N) - (a_i * a_j))}{\sqrt{((a_i * a_j)(N - a_i)(N - a_j))}}, \text{ where:}$$

a_{ij} is the count of transactions containing both i and j nodes, a_i is the count of transactions containing i, and a_j is the count of transactions containing j, N is the count of transactions over the entire network. Each pair of nodes will result in a connection or an edge with a PCC value [11].

2.3 Bootstrap

The bootstrap method is a statistical technique popularized by Bradley Efron in 1979. The goal is to perform simulations (shuffles) on the data and create new, possible datasets. With the simulation we can estimate the distribution of a value, making possible the calculation of statistics that are themselves computed from the same data. In our case, we use a nonparametric bootstrap to estimate the PCC variation, enabling us to calculate the significance of the PCC with the estimated confidence intervals (CI). The nonparametric method is a robust alternative to classic methods for statistical inference. But, we cannot test the null hypothesis of an edge correlation being different from zero, we can only use the bootstrap to show the accuracy of the edge correlation estimates and to compare edges to other edges [4].

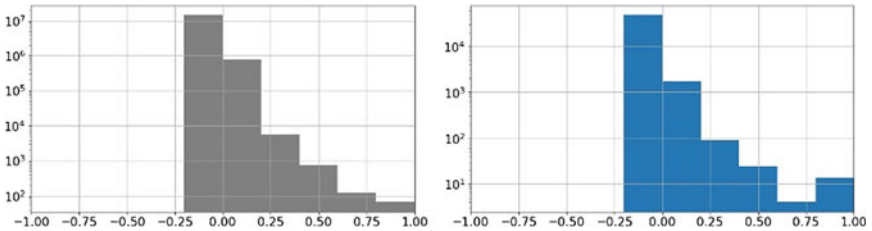


Fig. 4 The gray (left) histogram stands for the distribution of PCCs for the 300 bootstrapped samples and the blue (right) histogram stands for the distribution of PCCs in the original dataset

Since parametric methods assume that the observed data comes from some known distribution with unknown parameters, which are estimated thanks to this data, we choose the nonparametric method. Non-parametric bootstrapping fits better for the situation of this study since we do not know beforehand the PCCs distribution.

First, we performed the resampling, in other words, a sample with a replacement of the same size as the original network. To do this, we got the list with all connections (3276), and kept one node of each connection fixed while sorting the others. This way we got datasets of similar size (since resulting connections among the same driver were not considered). If N_B is the number of bootstrap samples to achieve a CI with a minimum significance level of α, it is necessary at least $N_B = \frac{2}{\alpha}$ samples. We set $\alpha = 0.05$ and overreached N_B value with a total of 300 samples.

Following, for each edge in each one of the 300 generated samples, the PCC was calculated the same way as in the original dataset. By the end, we had a similar PCC distribution, as it is possible to infer from Fig. 4.

Finally, using the percentile method, it is possible to construct nonparametric confidence intervals by calculating percentiles from the bootstrap estimates. As we previously set $\alpha = 0.05$ and we want to identify for each node if the original PCC is inside the confidence interval, the critical value was set in the 95th percentile. In sequence, the original value was compared with the interval between the minimal PCC from the bootstrap data and the 95th percentile.

After accomplishing these steps, we could finally approach a clean network without spurious connections, which we go deeper into the statistics and their meaning in the next section.

3 Results

This research started with a courier-courier undirect network inferred from the geolocation of these workers. Since it was inferred, as a first step it is necessary to verify if the connections found were by chance or not. Assessing the statistical significance of our network allows us to apply all the social network knowledge generated in the last few decades. When the steps mentioned above are applied, the edges with

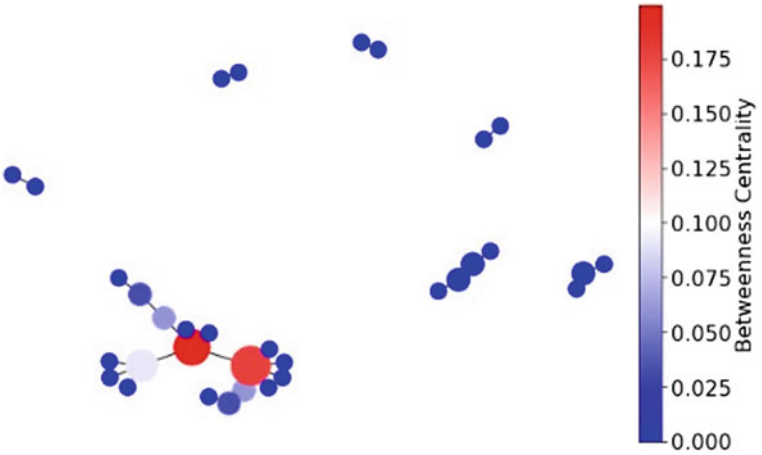


Fig. 5 The final network, after we made the exclusions mentioned, we arrived in a network with 33 nodes and 26 unique edges

significant PCC are identified, and we can filter the positive PCC. This way, a huge reduction in the coarse network, as it was supposed to be. The premise behind the network construction tells that if at least two couriers are idle in the same spot at the same time, it will possibly lead to social interaction. This premise is not always true, but the data showed us that it is possible to find some useful patterns, separating the noise from the signal.

From the original network, a smaller and significant network rises, with 26 significant connections (unique edges) among 33 delivery men, as it is possible to identify in Fig. 5. This final network showed stability in the assortativity coefficient (−0,1373), with low degree nodes connecting to other low degree nodes, an increase in the averaged the shortest path of the biggest component to 4.69, and a decrease in the average degree, to 2.54.

Our first contribution is to illustrate how we could possibly create a social network using geolocation and time data in the context of the gig economy, where we lack information from the workers and where it was possible to infer a significant informal social network among the delivery men from the studied city, separating from noise data.

Our second contribution comes up when we apply a well-known method, the nonparametric bootstrap, in a new situation that generates potential data for new studies.

Future assessments of temporal changes in the structure of the network, as well as replicating the method in other cities can identify hidden patterns of offline communication and may help predict, for example, fraud events or highlight the most central delivery men in the network (influential node), which can help the propagation of information since these workers do not have an official channel between them and the app platforms. Also, calculating the stability of centrality indices, sub-

setting the bootstrap datasets, using, for example, correlation stability coefficient (CS-coefficient) can bring new information on edge significance in the offline communication scenario.

References

1. Bakuła, M.; Uradziński, M.; Krasuski, K.: Performance of DGPS smartphone positioning with the use of P(L1) versus P(L5) pseudorange measurements. Remote Sens, **14**, 929 (2019). https://doi.org/10.3390/rs14040929
2. Candia, C., Encarnação, S., Pinheiro, F.L.: The higher education space: connecting degree programs from individuals choices. EPJ Data Sci. **8**(39) (2019). https://doi.org/10.1140/epjds/s13688-019-0218-4
3. Das, K., Samanta, S., Pal, M.: Study on centrality measures in social networks: a survey. Soc. Netw. Anal. Min. **8**(13) (2018). https://doi.org/10.1007/s13278-018-0493-2
4. Epskamp, S., Borsboom, D., Fried, E.I.: Estimating psychological networks and their accuracy: a tutorial paper. Behav. Res. **50**, 195-212 (2018). https://doi.org/10.3758/s13428-017-0862-1
5. Huang, K., Yao, J., Yin, M.: Understanding the skill provision in gig economy from a network perspective: a case study of fiverr. Proc. ACM Human-Comput. Interact. **3**(CSCW), 1-23 (2019)
6. Jabagi, N., Audebrand, L.K., Croteau, A.-M., Marsan, J. (2018). Connecting with Gig-workers: an organizational identification perspective. In: Proceedings of the European Group of Organization Studies (EGOS) 34th Colloquium
7. Jacobs, A.Z., Watts, D.J.: A large-scale comparative study of informal social networks in firms. Manag. Sci. **67**(9), 5489–5509 (2021)
8. Kalgotra, P., Sharda, R., Luse, A.: Which similarity measure to use in network analysis: impact of sample size on phi correlation coefficient and Ochiai index. Int. J. Inf. Manage. **55** (2020). https://doi.org/10.1016/j.ijinfomgt.2020.102229
9. Kinder, E., Jarrahi, M.H., Sutherland, W.: Gig platforms, tensions, alliances and ecosystems: an actor-network perspective. Proc. ACM Human-Comput. Interact. **3**(CSCW), 1-26 (2019)
10. Reagans, R., McEvily, B.: Network structure and knowledge transfer: the effects of cohesion and range. Adm. Sci. Q. **48**(2), 240–267 (2003)
11. Ronen, S., Gonçalves, B., Hu, K.Z., Vespignani, A., Pinker, S., Hidalgo, C.A.: Links that speak: the global language network and its association with global fame. Proc. Nat. Acad. Sci. U.S.A. **111**(52), E5616–E5622 (2014)
12. Wagner, C.J., González-Howard, M.: Studying discourse as social interaction: the potential of social network analysis for discourse studies. Educ. Res. **47**(6), 375–383 (2018)
13. Wang, J., Suri, S., Watts, D.J.: Cooperation and assortativity with dynamic partner updating. Proc. Nat. Acad. Sci. **109**(36), 14363–14368 (2012)
14. Watts, D.J., Strogatz, S.H.: Collective dynamics of small-world networks. Nature **393**(6684), 440–442 (1998)
15. Xulvi-Brunet, R., Sokolov, I.M.: Changing correlations in networks: assortativity and dissortativity. Acta. Phys. Pol. B **36**(5), 1431–1455 (2005)

Understanding Sectoral Integration in Energy Systems Through Complex Network Analysis

Andrea Diaz, Stephane Tchung-Ming, Ana Diaz Vazquez, Nicoleta Anca Matei, Esperanza Moreno Cruz, and Florian Fosse

Abstract This paper studies the concept of sectoral integration of energy systems from a network perspective. In the energy arena, the transition towards a cleaner use of energy has led to a series of changes in how we consume energy, the energy vectors we use to satisfy our needs and, in general, the configuration of our energy systems. These developments add complexity to our systems as their production and consumption configuration evolve. The concept of sectoral integration is recent and does not yet have a commonly agreed-upon definition nor a consistent measuring approach. We show that network analysis can be used to explore this evolution, allowing quantifying the degree of integration of existing systems. By using a stylized model, we propose a series of global and local measures, focusing on different parts of the energy system and allowing measuring system integration quantitatively. We then illustrate the developed measures by analysing the evolution of two European countries' energy systems over the recent past (1990–2019).

The views expressed are purely those of the authors and may not in any circumstances be regarded as stating an official position of the European Commission.

A. Diaz (✉) · S. Tchung-Ming · A. Diaz Vazquez · N. A. Matei · F. Fosse
European Commission, Joint Research Centre (JRC), Seville, Spain
e-mail: Andrea.DIAZ-RINCON@ec.europa.eu

S. Tchung-Ming
e-mail: Stephane.TCHUNG-MING@ec.europa.eu

A. Diaz Vazquez
e-mail: ana.diaz-vazquez@ec.europa.eu

N. A. Matei
e-mail: Nicoleta-Anca.MATEI@ec.europa.eu

F. Fosse
e-mail: Florian.FOSSE@ec.europa.eu

E. Moreno Cruz
Seville, Spain
e-mail: Esperanza.MORENO-CRUZ@ext.ec.europa.eu

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_38

Keywords Energy systems · Network theory · Sectoral integration · Renewables · Electrification of demand

1 Introduction

Recently, sectoral integration has been identified as a promising trend to speed up the transition towards low-carbon, efficient energy systems. This form of "energy system metabolism" refers to "coupling the energy consuming sectors–buildings, transport, and industry–with the power producing sector through smart infrastructure, increasing the penetration of renewable energy and thus decarbonizing the economy" [6]; other transformation sectors are also concerned. Research projects dealing with the subject have flourished over the last few years, e.g. under the umbrella of the Smart Energy Systems ERA-NET program in Europe. Key enabling technologies/pathways seem to be synthetic fuels (power-to-X), fuel cells, batteries, waste heat and, of course, smart management systems.

The main claims associated with sectoral integration relate to its potential to make the energy transition faster, less GHG intensive and more cost-effective, and to absorb larger amounts of variable electricity [6]. This topic has emerged as crucial in the energy modelling community. Existing studies focus on specific technologies or pathways, like electric vehicles [13], hydrogen [5], storage [14] or, more generally, poly-generation pathways [3], electricity [2], transport [8] or even cover the whole energy system [4]. The benefits of system integration, concerning e.g. the penetration of non-dispatchable energy sources, are highlighted throughout all the studies. However, there is a need for a more systematic analysis of the consequences of integration at sectoral and system levels and an in-depth analysis of the policy implications. The forthcoming technological revolution may require policy and R&D support, e.g. in transport [6], the role of infrastructures (energy, road, telecommunications etc.) and networks –particularly, the rigidities imposed by their physical limits.

One relevant and innovative way to fill these gaps in the domain of quantitative analysis could be to acknowledge the complex nature of energy systems. We cannot deny that all these structural and technological changes bring complexity to our energy systems. Complexity methods can help address the energy policy challenges ahead [1, 9, 10], especially when understanding the evolution of energy systems.

This paper proposes a methodology to analyse energy systems from a network perspective, a framework that can be used to assess the sectoral integration of energy systems. The study is structured as follows. A review of the sectoral integration concept is given in the upcoming section, followed by the presentation of the methodology developed. An application to two actual energy systems is discussed thereafter. The last section concludes while offering some potential directions for future extensions.

2 Sectoral Integration: A Definition

There is not yet a commonly agreed-upon definition of sectoral integration in the literature. For some, the concept is limited to the integration of hydrogen as an energy carrier [5] or, more generally, the (direct or indirect) electrification of demand [11, 15]. Whereas for others, the notion is related to the connection of supply infrastructures, such as the gas and electricity networks [12] or even the interconnection of the demand sectors.

In this paper, we follow the proposal of the EU Strategy for Energy System Integration [7] primarily because it covers a wide variety of aspects critical to any effort to address system integration. According to this proposal, ***energy system integration*** *involves the coordinated planning and operation of the energy system "as a whole", across multiple energy carriers, infrastructures and consumption sectors by creating stronger links between them with the objective of delivering low-carbon, reliable and resource-efficient energy services at the least possible cost for society.*

An integrated system should therefore encompass at least four complementary and mutually reinforcing concepts [7]:

- A cleaner power system, with more direct electrification of end-use sectors such as industry, heating of buildings and transport
- A cleaner fuel system (including clean hydrogen) for hard-to-electrify sectors like heavy industry or transport
- A 'multi-directional' system in which consumers play an active role in energy supply
- A more efficient and "circular" system, where waste energy is captured and re-used

From this definition, we can conclude that, in order to study sectoral integration, we first need a holistic view of the energy system that allows tracing the relationships between the different components from the supply and demand sides. Second, we need an approach that enables us to quantify the interconnection between the different components of the system at both the local and global levels. With this in mind, we look at the energy systems as networks where different producing and consuming activities are linked through energy flows with a direction and a weight. By using network analysis techniques, we formulate rules to measure the various aspects of sectoral integration from the network perspective while focusing at the same time on some crucial components. Note that this approach considers the networks as an abstract representation of the whole energy system of a country/region and not as the representation of the physical networks composing them, such as the electricity grids or gas infrastructures. For instance, when following this approach, electricity production is seen as a node within the network whose inputs are the different sources used to produce electricity in a given country, irrespectively of the actual configuration of the electricity production industry or infrastructure in that country.

3 Methodology

The following measures consider the energy system as a directed and weighted graph (G) defined by a set of n nodes $V(G)$ representing energy production, transformation, distribution and consumption activities, and m edges $E(G)$ representing the energy flows between those activities. Each node uses in-energy flows and releases out-energy flows in different proportions depending on its role within the system. For instance, the in-energy flows of the electricity production node correspond to the different resources used for centralized power generation (e.g. coal, gas, wind). In contrast, the out-energy flow of this node corresponds to the carrier electricity, which is centralised in a node. This node can be seen as a hub to gather electricity coming from different electricity production activities (electricity production nodes) and its dispatch to other nodes where it is used, such as the demand sectors (e.g. buildings and transport). In this sense, the node *electricity* can be seen as representing the transmission and distribution of electricity. This can be transposed to other carriers, such as liquid fuels (e.g. gasoline, diesel), whose node can be seen as the distribution to storage facilities for final consumption.

For every edge, a weight w_{ij} is assigned, representing the total energy flow of a carrier from origin i to destination j. All energy flows are represented in the same unit (e.g., toe—tonne of oil equivalent) regardless of the type of energy carrier they correspond to. Throughout the paper, we use the terms energy carrier, energy product, energy flow or, simply, fuel interchangeably.

The set of nodes $V(G)$ is further differentiated into two subsets: supply (SUP) and demand (DMD). The nodes on the supply side ($v_i \in SUP$) can be either energy carries or transformation activities. As already said, the nodes representing energy carriers can be seen as points where an energy carrier is centralised for distribution. In contrast, transformation nodes convert a specific energy carrier into another (e.g. wind or hydro into electricity). The nodes on the demand side ($v_i \in DMD$) correspond to all the energy-consuming sectors or transformation activities carried out by consumers (e.g. electricity generation by households or industry). Figure 1 illustrates the different network components.

For clarity in representation, in Fig. 1 we differentiate the shape of the nodes according to their type. Hexagons represent products, diamonds represent transformation activities, and circles represent demand sectors. Products can flow in and out transformation and demand activities, whereas relationships between products are not allowed. In the remaining part of this section, we propose a series of measures meant to assess sectoral integration taking into account the concepts described by the EU Strategy for Energy System Integration [7] presented above.

Starting from the demand perspective, the aspect of consumers playing an active role in the energy supply can take two forms. First, the consumers that send energy to the supply side, which we define as **vertical integration of demand**. This is, for instance, the case of end-use sectors producing energy to cover their demand needs and sending any surplus to the distribution network (e.g. electricity, gas). Second, the consumers that exchange energy with other consuming sectors, which we call

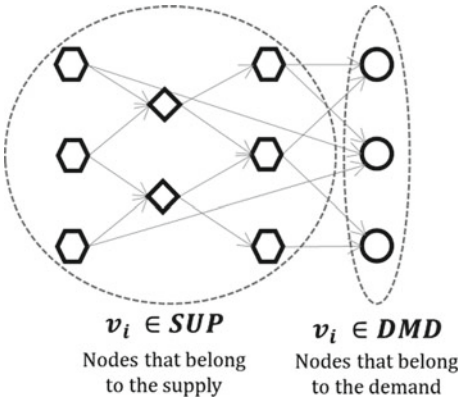


Fig. 1 Network representation with differentiation in two subsets: supply and demand

horizontal integration of demand. To measure the degree of *vertical integration of demand* (VI_{DMD}), we consider the energy flows from each demand node to the supply nodes as a share of all the energy in- and out-flowing of that demand node for each carrier. Keeping the notation w_{ij} as the energy flow of a carrier from origin i to destination j, we define:

$$VI_{DMD} = \sum_{i \in DMD} \frac{\sum_{j \in SUP} w_{ij}}{\sum_{j \in G} \left(w_{ji} + w_{ij}\right)}$$

To measure the degree of *horizontal integration of demand* (HI_{DMD}), we account for the energy flows of each specific carrier from the demand to other demand nodes as a share of all the energy in- and out-flowing of that demand node for each particular carrier, as follows:

$$HI_{DMD} = \sum_{i \in DMD} \frac{\sum_{j \in DMD} w_{ij}}{\sum_{j \in G} \left(w_{ji} + w_{ij}\right)}$$

Finally, to measure the degree of *direct electrification of demand* (DE_{DMD}), we take into account the sum of the share of electricity (elc) in-flowing to each demand node compared to the rest of energy carriers used, as follows:

$$DE_{DMD} = \sum_{i \in DMD} \frac{\sum_{j \in elc} w_{ji}}{\sum_{j \in G} w_{ji}}$$

It would be interesting to multiply this index by a coefficient that measures the share of renewables in electricity production.[1] However, in our case, this share would only consider the mix of the domestic production without accounting for electricity coming from other origins (i.e. imports of electricity).

4 Application to Actual Energy Systems

In order to illustrate the developed approach, we analyse the evolution of the energy systems of two European countries over 30 years (1990–2019): Luxembourg and France. The dataset employed corresponds to the yearly energy balance of each country published by Eurostat.[2]

Among the 40 countries available in the Eurostat database, we chose to concentrate on these two contrasting cases (Luxembourg and France) to provide a sufficiently wide application with significant differences in terms of network size and configuration. We are currently preparing a public application that includes the database's complete set of countries.

The energy balance accounts for the complete chain of supply, transformation and consumption of energy. The data allow tracing the relative contribution (flow) of each energy carrier (fuel, product) in the system, which ultimately shows the relationships between the different activities within the system. Beyond their reliability, these data allow a comparison between the countries on the same basis.

The energy balance takes the form of a matrix where columns represent all the different energy sources or products and rows represent all the different flows. This matrix is translated into a network (Figs. 2 and 3) where nodes represent activities such as energy supply, energy transformation (grey diamonds), energy distribution of the different products (hexagons) and energy consumption (blue circles). The colours of the distribution nodes (also called the product nodes, indistinctively) depend on the type of carrier represented (e.g. brown for fossil fuels, green for renewables and red for electricity). For the sake of simplicity, nodes representing energy supply (imports, exports and primary production) are not explicitly displayed in the representation, although they are accounted for in the network metrics. Edges represent the flow of energy products between those activities, with edge colours inherited from the product type. The size of the nodes is proportional to their degree relative to the network. For readability, only the nodes with the highest degree are displayed with their label in the most complex networks.

After a first glance at Figs. 2 and 3, we note significant differences between the energy systems of the two countries. Throughout the entire period, the energy system of Luxembourg is smaller, composed of a lower amount of nodes (activities)

[1] Such an indicator can take the form of $\theta_{RNW}^{ELC} = \frac{\sum_{j \in RNW}(w_{ji})}{\sum_{j \in G} w_{ji}} \forall i \in elc$, which accounts for the share of renewables (RNW) in electricity production (ELC) within the system.

[2] Available at: https://ec.europa.eu/eurostat/web/energy/data/energy-balances.

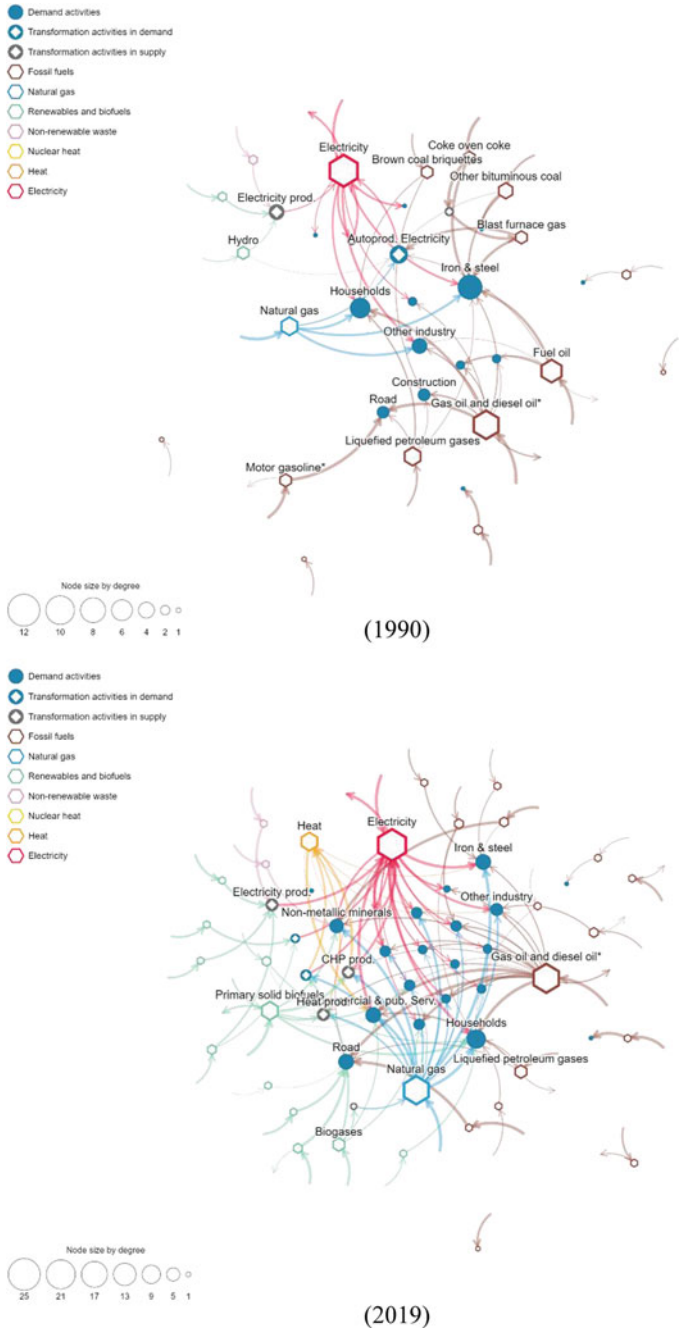


Fig. 2 Luxembourg: energy system network 1990 and 2019

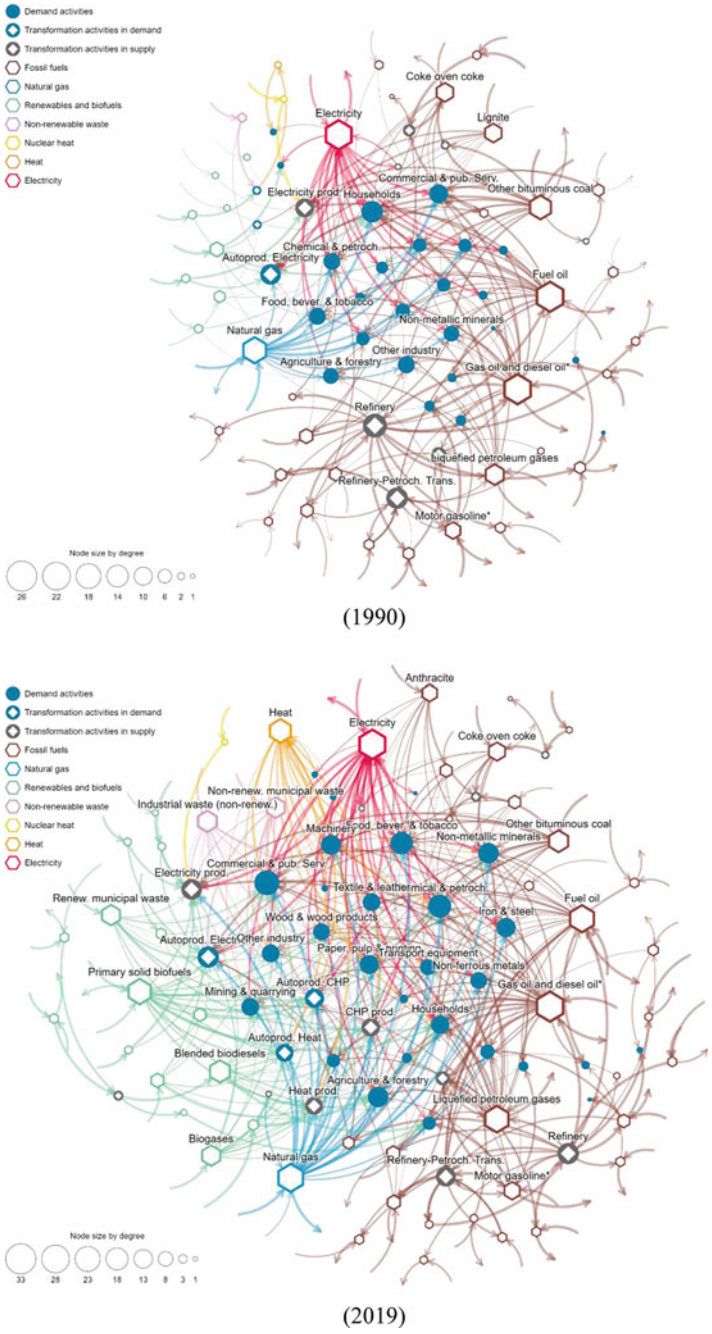


Fig. 3 France: energy system network 1990 and 2019

and edges (flows of energy), compared to the French system, even though quite a significant evolution can be observed in both countries.

The energy system of Luxembourg (Fig. 2) accounted 56 nodes and an average degree of 2.5 in 1990 and 97 nodes and an average degree of 3.0 in 2019. The significant increase in the number of nodes (73%) is mainly explained by the growth in the number of energy products consumed and their supply (38 nodes in 1990 against 69 in 2019). The new energy vectors/sources are basically renewable sources such as biofuels (under different forms: solid, liquid and gaseous), wind, solar (photovoltaic and thermal) and ambient heat (for heat pumps). Note that, although the significant increase in the number of nodes, the average degree only grows by 23%, revealing that the increase in connectivity does not follow the same pace of growth of the network.

When looking at specific nodes (Fig. 2), it is interesting to note that the electricity node appears more connected in 2019 with respect to 1990, with an out-degree evolving from 8 to 19 over the observed period. Indeed, as the electrification of demand has developed, electricity passed from being used by only six demand sectors initially to 17 demand sectors in 2019 (other uses of electricity are exports and other transformation activities). A similar effect can be noticed for natural gas, whose out-degree evolves from only four (of which three demand sectors) in 1990 to 18 (of which 15 demand sectors) in 2019.

On the other hand, France (Fig. 3) accounted 135 nodes and an average degree of 4.0 in 1990, and 162 nodes and an average degree of 5.8 in 2019. In contrast to the Luxembourg case, in France, the growth of the number of nodes (20%) is less significant than the growth of the average degree (45%). This fact suggests an emphasis in the network's connectivity rather than the number of activities performed within the system. We can distinguish therefore two distinct patterns of evolution between the two countries. The number of products consumed in France increases with the introduction of, mainly, biofuels (under different forms), wind and solar photovoltaic.

Compared to Luxembourg, the increase of connectivity of the electricity node is less important (with an out-degree going from 20 to 26 in the analysed period), mainly because the French energy system exhibited a strong use of electricity in final demand from early on. Electricity goes from being used in 18 demand sectors in 1990 to 22 sectors in 2019 (other uses of electricity are exports and other transformation activities). An interesting perspective however would be to have a look at the penetration rate of electricity per sector (taking into account the weights). Finally, a perhaps more appealing energy carrier to study is heat. In 1990, heat was only used by one demand sector whereas 15 end-use sectors were using this carrier in 2019.

A slightly different perspective can be gained when looking at the evolution over time of the basic metrics of the energy networks of both countries (i.e. number of nodes and average degree), normalized to one with respect to their 1990's value (see Fig. 4).[3]

[3] Other common methods of normalization could also be considered in order to control for the size of the country and the magnitude of its energy system, for example, normalizing with respect to the

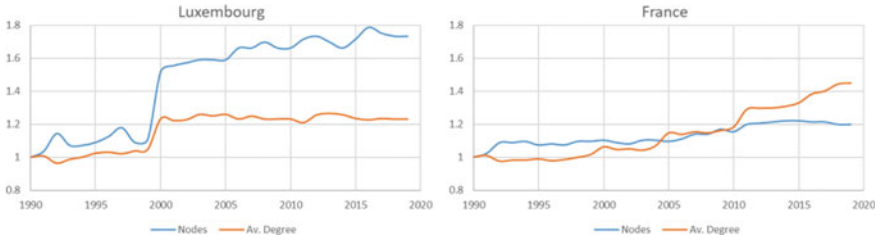


Fig. 4 Evolution of basic network indicators

Over the last three decades, France has seen a relatively constant increase in the number of activities/energy carriers and average degree regarding the values registered in 1990. It can be observed that a trend of higher growth of connectivity compared to number of nodes has arisen in the last decade. Luxembourg, on the other hand, presents a more irregular evolution over time with a rather stagnant average degree in the last two decades despite the growth of the number of nodes.

Regarding the indicators of sectoral integration derived in the previous section and given the dataset employed, from the three measures we developed for the demand side, only the degree of electrification of demand can be thoroughly calculated. Concerning the aspects of vertical and horizontal integration of demand, it is important to point out that these are relatively recent concepts, and the energy balance does not trace, as such, these types of energy exchanges. An alternative approach has been used to compute a slight variation of the measure for the vertical integration of demand (a proxy), while the horizontal integration could not be calculated altogether. This highlights the limitation of the data available in the energy balances and points to the need for traditional energy statistics to evolve in order to be able to capture the current and foreseen system transformations properly. Current statistics are not any more aligned with the needs arising from the energy transition.

Figure 5 details the evolution of the electrification of demand for both countries considered in this analysis. As expected, France exhibits a higher electrification index than Luxembourg, given the strong use of electricity in final demand since early on in this country. Nonetheless, it can be observed that the index has more than doubled over the last three decades for both countries, it increases from 5.9 to 11.5 for France and from 3.3 to 6.9 for Luxembourg. Nevertheless, this indicator is limited to assessing electricity use in final demand and can be improved in two aspects. First, by considering the share of renewables used for electricity generation. Second, by accounting for the share of domestic electricity production. Research into introducing these points is already underway.

Concerning the vertical integration of demand, as already stated, the energy balance does not provide the amount of energy that each demand sector sends to the network. The energy balance traces, however, some energy-producing activities

population of the country or the average final energy consumption of the country or, more generally, of the European Union.

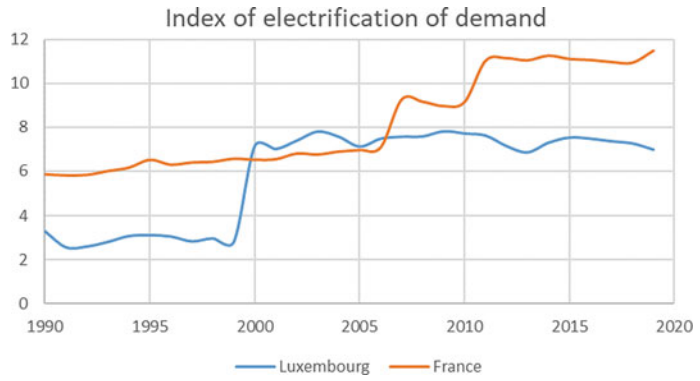


Fig. 5 Evolution over time of the electrification of demand

outside the energy sector under the category of "Auto-producers". These refer to industrial establishments that produce electricity and heat as "secondary products" (not as their principal activity). Whereas much of the produced energy is used within the unit, some is also sold to users outside. The energy balance reports this last component, which can give an idea of the degree to which demand plays an active role in supply. We use, therefore, this category as a proxy to calculate the vertical integration of industry as a whole (Fig. 6).

From Fig. 6 we can note that the level of vertical integration of industry regarding electricity "supply" (blue line, left axis) has varied over time with a slightly decreasing tendency. In any case, the level of 20% of electricity supply by the industry (as a whole), is not exceeded in both cases. Something, however, encouraging from Fig. 6 is that the share of renewables in that electricity "auto-production" (orange line, right axis) has been increasing over time, reaching in Luxembourg 60% of the mix and 40% in France. This suggests that the industry is in general using more clean sources to satisfy its energy needs, being therefore on the right path required to reach the decarbonisation of this sector.

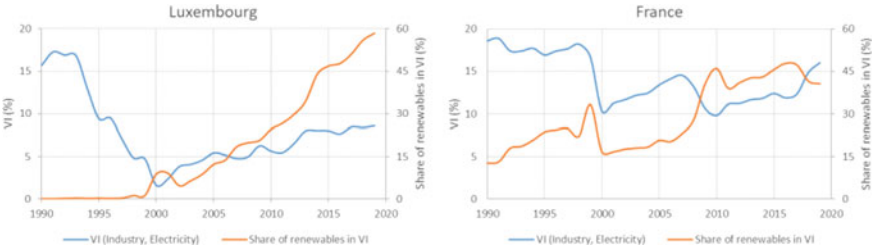


Fig. 6 Evolution over time of the vertical integration of demand (industry, electricity)

5 Conclusion

This paper proposes an approach to analysing energy systems from a network perspective. The methodology can be used to represent energy-system data as networks of interconnected activities. The resulting visual tool allows a straightforward comparison and analysis of energy systems as well as a holistic and synthetic view of them. It also allows inferring crucial qualities of the energy systems, such as the electrification of demand or the use of renewable sources, which are helpful for the analysis of sectoral integration and, in a more general manner, the analysis of energy systems.

The application, with data from Luxembourg and France, reveals some progress in electrification and the use of renewables by demand. It also shows the different development patterns followed by both systems, for instance, in the configuration, connectivity and evolution of the system and, in general, in the evolution of sectoral integration indicators. However, we have to acknowledge that the use of historical data limits the study as the concept of sectoral integration is only starting to develop. The use of these data reveals the need to rethink the statistical accounting of energy systems presented by the energy balances. The way these balances are constructed needs to evolve together with the evolutions of the systems. Only in this way, they will continue to be relevant for the analyses in the sector.

Our study ultimately provides the framework for a new way to analyse energy systems. Various extensions and future research developments are foreseen starting from this initial analysis. Firstly, to further our research, we intend to extend the analyses to the complete set of countries available in the Eurostat database. A more extensive comparison might provide new elements to understand energy systems' evolution and challenges. Secondly, after identifying key indicators to quantitatively measure the sectoral integration of energy systems, their role in determining the evolution of greenhouse gas emissions in Europe could be studied. Thirdly, future pathways of the energy systems could be explored to further assess the development of sectoral integration and complexity indicators under different long-term European energy policy scenarios.

Acknowledgements We thank the constructive comments of three anonymous reviewers, which substantially improved the quality of the paper. The work was carried out within the framework of the JRC exploratory research project **SIACES**—Sectoral Integration Assessment of Complex Energy Systems (31180-2021), which benefited from the financial support of the JRC Scientific Development Unit.

References

1. Bale, C.S.E., Varga, L., Foxon, T.J.: Energy and complexity: new ways forward. Appl. Energy **138**, 150–159 (2015)
2. Bellocchi, S., Manno, M., Noussan, M., Prina, M.G., Vellini, M.: Electrification of transport and residential heating sectors in support of renewable penetration: scenarios for the Italian energy system. Energy **196**, 117062 (2020)

3. Boblenz, K., Frank, V., Meyer, B.: Energy system analysis for evaluation of sector coupling technologies. Fuel **254**, 115658 (2019)
4. Brown, T., Schlachtberger, D., Kies, A., Schramm, S., Greiner, M.: Synergies of sector coupling and transmission reinforcement in a cost-optimised, highly renewable European energy system. Energy **160**, 720–739 (2018)
5. De Vita, A., Capros, P., Evangelopoulou, S., Kannavou, M., Siskos, P., Zazias, G., Boeve, S., et al.: Sectoral integration—long-term perspective in the EU energy system (2018)
6. European Commission: The role of sectoral integration in the clean energy transition. World Future Energy Summit. Abu Dhabi (2019)
7. European Commission: Powering a climate-neutral economy: an EU strategy for energy system integration (2020)
8. Han, S., Kim, J.: Optimization-based integration and analysis of a complex renewable energy system for the transportation sector. Chem. Eng. Res. Des. **128**, 1–14 (2017)
9. Hidalgo, C.A., Hausmann, R.: The building blocks of economic complexity. Proc. Nat. Acad. Sci. **106**(26), 10570–10575 (2009)
10. Labanca, N., Ruzzenenti, F., Fath, B.D., Bauwens, T., Bertoldi, P., Shove, E., Allen, T., et al.: Complex Systems and Social Practices in Energy Transitions. In: Labanca, N. (ed.). Springer, Cham (2017)
11. Robinius, M., Otto, A., Euser, P., Welder, L., Syranidis, K., Ryberg, D.S., Grube, T., Markewitz, P., Peters, R., Stolten, D.: Linking the power and transport sectors—part 1: the principle of sector coupling. Energies **10**, 956 (2017)
12. Schaber, K., Steinke, F., Hamacher, T.: Managing temporary oversupply from renewables efficiently: electricity storage versus energy sector coupling in Germany. Paper presented at the international energy workshop, Paris (2013)
13. Sterchele, P., Kersten, K., Palzer, A., Hentschel, J., Henning, H.-M.: Assessment of flexible electric vehicle charging in a sector coupling energy system model—modelling approach and case study. Appl. Energy **258**, 114101 (2020)
14. Victoria, M., Zhu, K., Brown, T., Andresen, G.B., Greiner, M.: The role of storage technologies throughout the decarbonisation of the sector-coupled European energy system. Energy Convers. Manage. **201**, 111977 (2019)
15. Wietschel, M., Held, A., Pfluger, B., Ragwitz, M.: Energy integration across electricity, heating & cooling and the transport sector—sector coupling. Working paper sustainability and innovation, No. S08/2020, Fraunhofer-Institut für System- und Innovationsforschung ISI, Karlsruhe (2020)

An Analysis of Bitcoin Dust Through Authenticated Queries

Matteo Loporchio, Anna Bernasconi, Damiano Di Francesco Maesa, and Laura Ricci

Abstract Dust refers to the amounts of cryptocurrency that are smaller than the fees required to spend them in a transaction. Due to its "economically irrational" nature, dust is often used to achieve some external side effect, rather than exchanging value. In this paper we study this phenomenon by conducting an analysis of dust creation and consumption in the Bitcoin blockchain. We do so by exploiting a new method that allows resource-constrained nodes to retrieve blockchain data by sending authenticated queries to possibly untrusted but more powerful nodes. We validate the method effectiveness experimentally and then analyze the collected data. Results show that a large amount of dust can be traced back to on-chain betting services.

Keywords Authenticated data structures · Bitcoin · Blockchain · Network analysis

1 Introduction

Blockchain technology has the potential to transform several social and economic activities by providing new ways of organizing business processes and handling information without the need for trusted third-party entities. However, its large-scale adoption demands the solution of several challenging problems. For instance, the huge size of current blockchain ledgers is a big factor hindering their widespread

M. Loporchio (✉) · A. Bernasconi · D. Di Francesco Maesa · L. Ricci
Department of Computer Science, University of Pisa, Largo Bruno Pontecorvo 3, 56127 Pisa, Italy
e-mail: matteo.loporchio@phd.unipi.it

A. Bernasconi
e-mail: anna.bernasconi@unipi.it

D. Di Francesco Maesa
e-mail: damiano.difrancesco@unipi.it

L. Ricci
e-mail: laura.ricci@unipi.it

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_39

adoption. Indeed, while it was initially sustainable for ordinary computing nodes, the current size of blockchains like Bitcoin [9] and Ethereum [13] has reached hundreds of gigabytes, and in some cases, crossed 1 terabyte.

Taking Bitcoin as a reference, one solution in this regard is to assign the task of synchronizing the entire blockchain to *full nodes*, namely nodes with large computational, storage, and bandwidth capabilities, therefore able to store a copy of the entire ledger. On the other hand, *light nodes* with constrained resources may retrieve transactions of interest only by submitting queries to full nodes. Despite its simplicity, this solution requires light nodes to trust full nodes and thus recreates the main problem that blockchains aim to solve: the need for trusted intermediaries.

To overcome this, untrusted full nodes may authenticate the query results by sending a cryptographic proof of integrity, usually referred to as *verification object* (VO). To construct such proofs, several *authenticated data structures* [10] like *Merkle trees* [8] have been proposed over the years, yet current solutions are generally limited to simple queries (e.g., retrieving all transactions involving a specific address). Nonetheless, in several scenarios it might be necessary to retrieve data satisfying more complex conditions.

In this paper, we focus on the problem of detecting Bitcoin *dust*, namely tiny amounts of value that are often left unspent inside an address because they are smaller than the fee required to spend them in a transaction. In the Bitcoin network, dust is particularly relevant since it is frequently related to transactions produced by specific on-chain services, as will be discussed in our experimental analysis, or even *dust attacks* attempting to break users' pseudonymity [2]. To enable efficient retrieval and analysis of specific Bitcoin transactions, such as dust ones, in this paper we propose a method based on *range queries* that extends our previous work presented in [6] and allows us to filter out all amounts that fall below the so-called *dust limit* [11]. More precisely, we propose to construct a *Merkle interval tree* (i.e., a specialized version of a *Merkle R-tree* [16]) from the input and output amounts of Bitcoin transactions. As use case, we simulate a scenario where a Bitcoin light node submits a range query to a full node to retrieve all Bitcoin dust. We briefly introduce the algorithms for fetching transaction inputs and outputs within a given range, and for verifying the integrity of the results. These algorithms are employed by full and light nodes to fetch and verify results, respectively. We also present a rich set of analysis of the returned transactions aimed at discovering patterns behind dust creation and consumption. Our analysis reveals that a significant part of all Bitcoin dust has been generated by *Satoshi Dice* [3], a popular blockchain-based gambling game launched in April 2012.

Related work Blockchain query authentication has already been covered from a number of different perspectives in the literature. For instance, Nakamoto's *Simplified Payment Verification* (SPV) [9] can be considered a primitive form of query autentication, as it enables light nodes to verify that a transaction has been included in the blockchain without downloading the entire ledger, but it does not protect against results omission. More advanced solutions like *vChain* [15] or the *GCA*^{2}*-tree* [17] rely on *set accumulators* [5] for authenticating Boolean and aggregate range queries,

respectively. Concerning Bitcoin dust, Delgado-Segura et al. [4] include dust outputs in their analysis of the unprofitable outputs contained in the set of Bitcoin unspent outputs. Finally, the authors of [12] proposed a strategy for identifying and preventing the creation of large amounts of dust transactions in the Bitcoin blockchain.

2 Background

In this section we introduce the notation and theoretical concepts used throughout the paper.

Intervals We refer to *intervals* as sets of consecutive integers between a lower and an upper bound. Given $l, u \in \mathbb{Z}$, we will represent an interval as $[l, u] = \{x \in \mathbb{Z} \mid l \leq x \leq u\}$. Moreover, given two intervals $I_1 = [l_1, u_1]$ and $I_2 = [l_2, u_2]$ we will denote by $I_1 \sqcup I_2 = [\min\{l_1, l_2\}, \max\{u_1, u_2\}]$ the minimum-width interval enclosing both I_1 and I_2.

Authenticated data structures Authenticated Data Structures (ADSs) [10] incorporate cryptographic information (e.g., a digest computed with a *cryptographic hash function* [7]) for guaranteeing the integrity of the data they are built upon. Among the most prominent examples, we cite the binary Merkle tree [8], which is widely employed in Bitcoin and other blockchain systems for ensuring the authenticity of transactions stored inside a block. More advanced ADSs combine Merkle trees with existing data structures to enable efficient data retrieval. In this regard, we recall Merkle R-trees [16], already mentioned in Sect. 1, for managing spatial data and cite Ethereum's *modified Merkle Patricia tree* [13], which is based on *Patricia tries* for representing the state of Ethereum accounts.

Bitcoin blockchain The Bitcoin blockchain is an ordered list of blocks, each comprising a header and a content payload. The header contains all the data needed to make the chain immutable, e.g., a cryptographic hash pointer to the previous block and a Merkle Tree root of the content payload. The content section contains a list of transactions, namely redistribution of funds between entities. Each transaction can be associated with a set of general information (e.g., the transaction hash, the block it is part of, or the fees paid), zero or more inputs, and one or more outputs. Each output can be seen as a couple $(amount, recipient)$, where $recipient$ specifies who and under which conditions the $amount$ can be redeemed, i.e., spent by another transaction. The simplest example of redeem condition can be requiring a digital signature associated to an address (i.e., an encoding of a cryptographic hash of a public key) receiving the amount, but arbitrarily complex conditions can be specified. Inputs of transactions are, instead, pointers to previously created transaction outputs. All outputs redeemed by a transaction are completely consumed and their held value is redistributed among the newly created outputs (or spent as voluntary fee).

3 Efficient Information Retrieval in Bitcoin

In this section we detail our approach for authenticating range queries on the Bitcoin blockchain.

Data model In our model, light nodes issue *intra-block* range queries, meaning that they are interested in retrieving data from a single block of the chain. We assume that a block contains a set of transactions $T = \{t_1, \ldots, t_n\}$ and that the light node specifies a query interval $[l, u]$ to retrieve all transaction outputs (or inputs) whose amount is between l and u (both included). Our goal is to guarantee two fundamental properties for the query results delivered by full nodes, namely *authenticity* and *completeness* [14]. Results satisfy the former if and only if they are returned without any modification with respect to the original data stored on the blockchain and the latter if and only if all objects satisfying the interrogation are returned, without any omission.

System model Fig. 1 illustrates the proposed system model for our query authentication protocol based on [6]. In this scenario, the miner that constructs a block also builds a tree-shaped authenticated data structure on the set of transactions in that block. The cryptographic hash of the ADS root is then embedded in the block header, using a specific `indexHash` field. The full node receiving a range query $Q = [l, u]$ from the light node uses the data structure to fetch transactions from the corresponding block and builds a verification object which is then sent back to the requesting node. This task is accomplished with a suitable *query algorithm* that traverses the ADS to retrieve relevant information and build the corresponding proof. The light node, in turn, runs a *verification algorithm* that takes the VO as input, extracts the matching transactions and reconstructs the ADS root. This reconstructed value is then compared against the one that has been incorporated in the header by the miners. If these two do not coincide, then it means that either some matching transaction has not been returned or data have been modified with respect to their original version, thus revealing the malicious behavior of the full node.

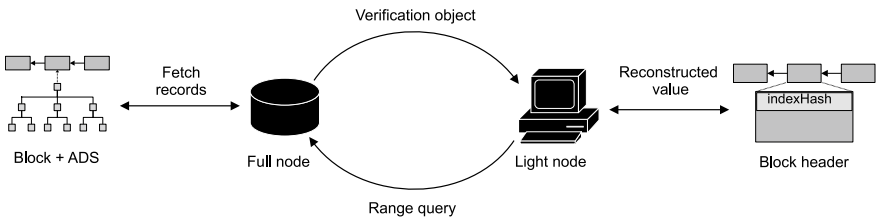


Fig. 1 The blockchain query authentication process, based on [6]

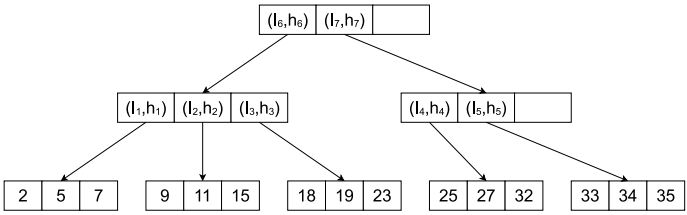


Fig. 2 An example of a Merkle interval tree for a set of $m = 15$ values with page capacity $c = 3$

3.1 Merkle Interval Tree

To authenticate unidimensional range queries on the Bitcoin blockchain, we propose to employ a specialized version of the Merkle R-tree [16], which we will refer to as the *Merkle interval tree*. In this section, we provide an overview of the algorithms for constructing and querying the Merkle interval tree, as well as for verifying the results delivered by full nodes.

The diagram of Fig. 2 illustrates the structure of a Merkle interval tree with page capacity $c = 3$ for a set of $m = 11$ values. Here, each leaf node ℓ stores at most c values from the set and is associated with a cryptographic digest h_ℓ and an interval I_ℓ. The digest is computed with a cryptographic hash function $\mathcal{H}$ and the endpoints of I_ℓ coincide with the minimum and maximum value in ℓ. For instance, the interval of the leftmost leaf is $I_1 = [2, 7]$ and its digest h_1 is obtained by hashing the binary representations of its elements, namely 2, 5, and 7. Conversely, internal nodes store at most c entries $e_i = (I_i, h_i)$ that describe their children. Taking the root as a reference, the interval I_6 of the first entry is obtained as $I_1 \sqcup I_2 \sqcup I_3$ from the intervals in the first child, while the corresponding digest is computed as $h_6 = \mathcal{H}(I_1|h_1|I_2|h_2|I_3|h_3)$, where | denotes the string concatenation operator. On the other hand, for the second entry in the root we have that $I_7 = I_4 \sqcup I_5 = [25, 32] \sqcup [33, 35] = [25, 35]$ and $h_7 = \mathcal{H}(I_4|h_4|I_5|h_5)$.

Construction algorithm To build a Merkle interval tree from a set of m values, miners run the *packed algorithm* discussed in [6]. The key ideas behind the method are the following:

(1) first, the set of input values is sorted;
(2) then it is partitioned in $\lceil m/c \rceil$ groups of c consecutive values, each inducing a new leaf node;
(3) nodes are iteratively merged in groups of c until only one is left, namely the root.

Concerning its complexity, we showed in [6] that if the capacity c is constant, then this procedure runs in $O(m \log m)$ time.

Query algorithm To query the Merkle interval tree, full nodes employ the algorithm described in [16] for Merkle R-trees, which outputs a VO with both matching records and authenticated information. Given a range query $Q = [l, u]$, the full node starts from the tree root and proceeds recursively. When it reaches an internal node

z, the algorithm scans each entry $e_i = (I_i, h_i)$ and compares the interval I_i of the i-th child with $[l, u]$. If these intersect, then the subtree rooted in e_i is immediately explored with a recursive call. On the other hand, if their intersection is empty the corresponding child node is *pruned*, since it does not contain matching transactions. As soon as a leaf is reached, all the underlying records are added to the VO.

Verification algorithm The light node, in turn, can run the verification algorithm from [16] to check the authenticity and completeness of the received results. To this aim, the algorithm takes the initial query and the VO as inputs and uses them to reconstruct the root node of the Merkle interval tree. The hash of the reconstructed node is then compared against the `indexHash` field in the block header. If the values coincide, then the proposed query authentication procedure guarantees both authenticity and completeness of the results delivered by full nodes, as formally proved in [6].

4 Bitcoin Dust

In Bitcoin, the term *dust* refers to the tiny amounts of value that are smaller than the minimum fee required to spend them in a transaction. As detailed in [11], an amount is considered dust if it is lower than the threshold of 546 satoshi[1]. In the Bitcoin blockchain, dust may originate on different occasions. For instance, the popular blockchain-based gambling service *Satoshi Dice* [3] used to send back 1 satoshi to losing players to notify they had lost their bet. This scenario will be further investigated in Sect. 5, where we will present our results. Another example are provably unspendable outputs with zero or dust amounts that can be generated by creating transactions with the special `OP_RETURN` redeeming instruction, whose goal is to store arbitrary data on the blockchain [1].

Even if we will not cover this aspect in our analysis, dust can also be linked to malicious behaviors. In this regard, we mention *dust attacks*[2], during which an adversary sends tiny amounts of bitcoin to many different addresses with the goal of deanonymizing the receivers and breaking their privacy. More precisely, the attacker hopes that the users (or their wallet software) will eventually aggregate these amounts as inputs to a larger transaction. Since it is uncommon for input addresses to belong to different users, this strategy may allow the attacker to link all of them to the same identity.

For the rest of this paper, we will say that a transaction is *dust-creating* (resp. *dust-consuming*) if and only if it has at least one dust output (resp. input). Furthermore, an address is a dust *receiver* (resp. a *sender*) if and only if it is associated with one of the dust outputs (resp. inputs) of a transaction.

[1] A satoshi represents the smallest bitcoin denomination, namely 10^{-8} bitcoins.

[2] These are also referred to as *forced address reuse attacks* [2].

5 Experimental Results

This section is devoted to the presentation and discussion of our experimental results. Our contribution in this regard is twofold:

(1) first, we evaluate the algorithms of Sect. 3.1 by performing range queries on real data from the Bitcoin blockchain;
(2) secondly, we analyze the data collected in the previous step with the goal of better understanding the use of dust amounts in Bitcoin.

The data set used for our experiments includes all $N_{txs} = 245{,}410{,}083$ transactions in the first $N_b = 479{,}969$ blocks of the Bitcoin blockchain, thus covering the time period between January 3rd, 2009 and August 10th, 2017. The code has been written in Java and Python and is publicly available at https://github.com/mloporchio/BTXA. Our implementation has been tested on a full node running Ubuntu Linux, with an 8-core Intel Xeon 5218 CPU @ 2.3 GHz and 256 GB of RAM, and an Apple Macbook as light node with a dual-core Intel i7 CPU @ 1.7 GHz and 8 GB of RAM.

5.1 Tree Construction

We evaluate the algorithms described in Sect. 3.1 using the Bitcoin data set. For each block, we build a Merkle interval tree on all transaction outputs using the corresponding amount as search key. We then fix a range $Q = [1, 545]$ with the intent of retrieving all non zero dust outputs, query the data structure, and finally verify the results using the algorithms described in Sect. 3.1. This procedure is repeated for different values of the page capacity and the corresponding execution times, expressed in milliseconds, are shown in Table 1. For each capacity value, we report the tree construction, query, and verification times, computed as the average over all blocks in our data set.

Table 1 Average execution times for the Merkle interval tree construction, query, and verification methods

Capacity	Construction	Query	Verification
4	1.467	0.001	0.017
8	1.173	0.001	0.025
16	1.034	0.001	0.059
32	0.980	0.002	0.177
64	0.961	0.006	0.456
128	0.948	0.007	0.552
256	0.942	0.007	0.553

Results are expressed in milliseconds

Our tree-based approach is compared with the naive solution which scans all transactions of each block and includes only the matching transaction outputs in the final result. In this case, the average query execution time, computed over all N_b blocks in the data set, is equal to 0.052 ms. By comparing this result with the query times of Table 1, the tree-based approach appears to be always an order of magnitude faster, independently of the page capacity. On the other hand, we remark that there is no construction cost in the naive solution. However, in our solution, even if the construction cost is much higher than the query cost, it is only paid once by the miner which builds the ADS and each full node. In fact, once constructed, Merkle interval trees can be employed for answering any number of queries and hence the cost of their construction can be considered as amortized on all subsequent interrogations issued by light nodes.

The previously described experiments on dust outputs are repeated for retrieving dust inputs, and the obtained results, both transaction outputs and inputs, constitute the data sets we use in the rest of this section. More precisely, we refer to D_{out} (resp. D_{in}) as the set including all dust outputs (resp. inputs). Each output in D_{out} is represented as a tuple containing:

(1) the transaction timestamp;
(2) the block and transaction identifiers;
(3) the destination address;
(4) the amount in satoshi;
(5) the output offset, i.e., its position among all outputs of the same transaction.

Similarly, a dust input in D_{in} consists of all fields from 1 to 4 (in this case, however, the address represents the payment source) as well as the identifier and offset of the transaction where it has been created.

5.2 Transaction Analysis

The data sets D_{in} and D_{out} obtained from Sect. 5.1 contain $N_{in} = 2{,}569{,}846$ dust inputs and $N_{out} = 4{,}400{,}757$ dust outputs, respectively. Using them we can deduce that 1,705,560 transactions (nearly 0.7% of the total number N_{txs}) are creating dust, while only 429,544 (nearly 0.2% of N_{txs}) are consuming it. As a result, each dust-creating transaction has 2.58 dust outputs on average, while dust-consuming have 5.98 dust inputs. We then evaluate the frequencies of dust outputs and inputs in dust-creating and dust-consuming transactions, respectively. In this regard, the first two plots (from left to right) of Fig. 3 show that approximately 10^6 dust-creating transactions have exactly one dust output. Similarly, most dust-consuming transactions (i.e., between 10^5 and 10^6) have only one input. More generally, we can observe that transactions with a high number of dust inputs and outputs appear to be less common. From the second plot, we can also observe that the relationship between the number of dust inputs n_i and the number of transactions with n_i dust inputs can be accurately described by a power law. Finally, in the third (resp. fourth)

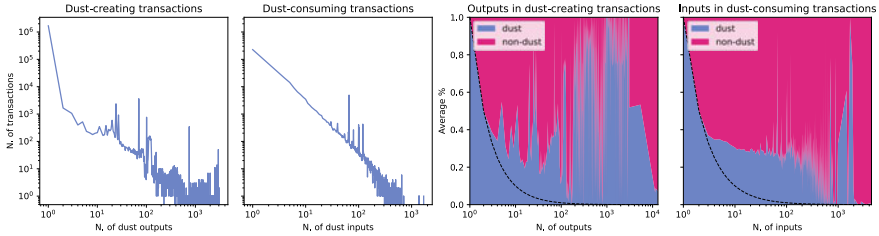


Fig. 3 Properties of dust-creating and dust-consuming transactions

plot of Fig. 3 we reported the average percentages of dust and non-dust outputs (resp. inputs) in dust-creating (resp. dust-consuming) transactions. The dashed lines in both charts represent the minimum percentages for dust outputs and inputs. For instance, since in the third (resp. fourth) plot we are only considering dust-creating (resp. dust-consuming) transactions, in a transaction with k outputs (resp. inputs) the percentage of dust outputs (resp. inputs) will always be at least $1/k$. Except for the obvious cases of one output and one input, in both scenarios the subdivision does not seem to follow any regular pattern, although non-dust amounts seem to be generally prevailing among the inputs and in transactions with less than 10^2 outputs.

5.3 Address Analysis

To identify the top dust senders and receivers, we examine the most frequent addresses in D_{in} and D_{out}. We select the top 5 addresses for each category and gather our results in Table 2. Concerning the top senders, we can observe that all addresses are related to Satoshi Dice, the gambling game already mentioned in Sect. 4. All Satoshi Dice addresses are easily recognizable, since they have been generated with mnemonic `1dice` prefixes, and their winning odds (denoted by W) and winning multiplier (denoted by M) are known in advance to the players. As discussed in Sect. 4, this result can be easily explained since Satoshi Dice addresses send back to the bettor a small fraction of the original wager in case of loss. It is also interesting to notice that the activity of the top receivers can be linked to this betting game too. For each of them, Table 2 contains the number of transactions having at least one Satoshi Dice input address. For 4 addresses out of 5, at least 75% of the received transactions were sent by a known Satoshi Dice address. This leads us to believe that their owners are gamblers, with the only exception of `18d3HV2bm94UyY4a9DrPfoZ17sXuiDQq2B`. Indeed, this address has received dust in 8 099 different transactions and 8 098 of them are actually *coinbase* transactions, used by miners for collecting the block reward. This fact suggests that the address may belong to a member of a mining pool.

Table 2 The top 5 dust sender and receiver addresses

	Address	TXs	Notes
Senders	1dice8EMZmqKvrGE4Qc9bUFf9PX3xaYDp	374,464	Satoshi Dice ($W = 48.8281\%$, $M = 2.004\times$)
	1dice97ECuByXAvqXpaYzSaQuPVvrtmz6	177,201	Satoshi Dice ($W = 50.0000\%$, $M = 1.957\times$)
	1dice6YgEVBf88erBFra9BHf6ZMoyvG88	127,790	Satoshi Dice ($W = 12.2070\%$, $M = 8.000\times$)
	1dice7fUkz5h4z2wPc1wLMPWgB5mDwKDx	123,005	Satoshi Dice ($W = 24.4141\%$, $M = 4.003\times$)
	1dice1e6pdhLzzWQq7yMidf6j8eAg7pkY	93,724	Satoshi Dice ($W = 0.0015\%$, $M = 64000\times$)
Receivers	1PEDJAibfNetJzM289oXsW1qLAgjYDjLgN	14,337	10,758 from SD (75%) – Satoshi Dice gambler
	15tcsumFPsrw2p9Egmk7wGszFJVxpw7UiD	11,131	11,126 from SD (100%) – Satoshi Dice gambler
	18d3HV2bm94UyY4a9DrPfoZ17sXuiDQq2B	8099	0 from SD (0%) – Mining pool member
	14z1fVwxMG71WcijX9J9te8G1wyp7tVqdz	7628	7626 from SD (100%) – Satoshi Dice gambler
	1FE1CDgkqzMSFXFreXrET7hvhEfCP9QabY	3739	3739 from SD (100%) – Satoshi Dice gambler

Table 3 Dust output classification

	No. of outputs	Percentage (%)
Unspent	1,830,911	41.604
NOD	2,420,707	55.007
OD	149,139	3.389
Total	4,400,757	100.000

5.4 Output Analysis

As detailed in Table 3, we classify the transaction outputs in D_{out} into three distinct groups:

(1) *unspent* outputs, still left in the corresponding addresses;
(2) *Not Only Dust* (NOD), spent in combination with at least one non-dust output;
(3) *Only Dust* (OD), spent exclusively in combination with other dust outputs.

From Table 3 we can notice that most outputs (i.e., nearly 55%) belong to the NOD category, while nearly 42% of them are still unspent. However, since our analysis has only taken into account blocks up to August 10th, 2017, these outputs could have been spent at a later time. Instead, the remaining 3% is made up of all OD outputs, spent only in combination with other dust outputs. The role of NOD and OD outputs has been further examined with a temporal analysis, presented in the following paragraph.

Temporal analysis The temporal analysis we have conducted on the dust outputs has a dual purpose:

(1) finding the average number of blocks between the output creation and consumption (i.e., their expenditure);
(2) discovering patterns in the consumption itself.

Concerning their consumption, we have discovered that, on average, dust outputs get consumed after 25,165.33 blocks, while non-dust outputs only last for 3207.36 blocks. Our findings are summarized by the plots of Fig. 4, where we report the number of dust and non-dust outputs that have been spent after a given number of blocks. The mean values of the two distributions suggest that dust amounts tend to go unnoticed and get spent after a longer period of time. Taking 10 min as the average inter-block time[3], this results in a period of approximately 175 d for dust outputs and 22 d for non-dust amounts. This gap is probably due to the fact that users typically wait until they have collected a sufficient number of dust outputs before they can aggregate them into a single transaction.

To discover meaningful patterns behind dust consumption, we first associate each output with the timestamp of the transaction where it is spent and then count the

[3] The inter-block time is the time that passes between the creation of two consecutive blocks.

Fig. 4 Output duration

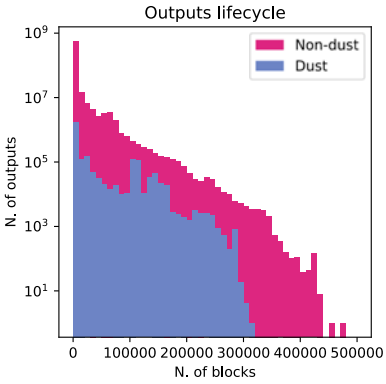


Table 4 Top 5 addresses for NOD output consumption

Address	NOD outputs	From SD	Percentage (%)	Description
1PEDJAibfNetJzM289oXsW1qLAgjYDjLgN	12,502	10,758	86	Satoshi Dice gambler (see Table 2)
14z1fVwxMG71WcijX9J9te8G1wyp7tVqdz	7628	7626	100	Satoshi Dice gambler (see Table 2)
18d3HV2bm94UyY4a9DrPfoZ17sXuiDQq2B	7288	0	0	Mining pool member (see Table 2)
1GmREU2gwcvQHRQFgwHvbD4dyL8iryCPMY	3614	3270	90	Satoshi Dice gambler
1dES7RLppoYc8mLQedwUoJMZZ9RnuCP5f	3526	3526	100	Satoshi Dice gambler

number of consumed outputs on a yearly basis. The results are summarized by the leftmost plot of Fig. 5, where each year is divided in quarters for better readability. We can notice that the consumption of NOD outputs has started raising during the second quarter of 2011, reaching its peak in 2013. Other minor peaks can also be observed during 2014 and 2015. To identify the main causes of NOD aggregations, we analyze, once again, the most frequent addresses. The top 5 addresses in terms of NOD output consumption are listed in Table 4. Interestingly enough, the first three have already been identified in Table 2 and we have already linked them to mining activity and Satoshi Dice. The remaining two addresses can also be associated with this popular gambling game. Indeed, as reported in Table 4, at least 90% of their NOD outputs have been sent by a Satoshi Dice address, thus confirming our intuition. Moreover, from the NOD histogram of Fig. 5, we notice that there has been a sudden increase of aggregations during the second quarter of 2012. As stated in [3], this increase may coincide with the launch of the betting game, which took place on April 24th, 2012.

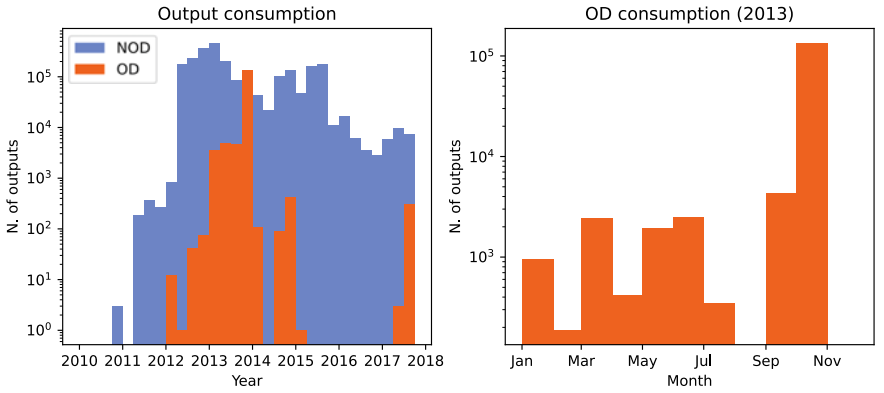


Fig. 5 Temporal analysis of dust output consumption

Regarding OD aggregations, we notice that 2013 has been the most prolific year. In fact, the rightmost histogram of Fig. 5 shows a peak with more than 10^5 consumed outputs during the month of October. We have further investigated this phenomenon by examining the most frequent addresses and found out that `1JwSSubhmg6iPtRjtyqhUYYH7bZg3Lfy1T` has spent 134,693 outputs. This is about 90% of the total number of OD outputs, which, as reported in Table 3, is equal to 149,139. The interesting fact about this address is that its private key has been compromised[4], which allows anyone to redeem bitcoins as soon as they are sent to it.

6 Conclusions and Future Work

In this paper we have conducted an analysis of Bitcoin dust creation and consumption. Our data were collected from the Bitcoin blockchain using a new methodology for authenticated information retrieval. The proposed approach has been evaluated experimentally, showing its advantage over a naive direct scan of the chain. On the other hand, the dust analysis results show that a significant part of these tiny bitcoin amounts can be traced back to a single service, namely Satoshi Dice, a popular blockchain-based betting game launched in 2012. The information we have collected can be further exploited during future work, e.g., to identify dust based deanonymization attacks and to study their impact on the whole network.

[4] Source: https://privatekeys.pw/address/bitcoin/1JwSSubhmg6iPtRjtyqhUYYH7bZg3Lfy1T.

References

1. Bartoletti, M., Pompianu, L.: An analysis of Bitcoin OP_RETURN metadata. In: Financial Cryptography and Data Security. pp. 218–230. Springer International Publishing, Cham (2017)
2. Bitcoin Wiki.: Privacy—forced address reuse. https://en.bitcoin.it/wiki/Privacy#Forced_address_reuse. Accessed 12 June 2022
3. Bitcoin Wiki.: Satoshi Dice. https://en.bitcoin.it/wiki/Satoshi_Dice. Accessed 12 June 2022
4. Delgado-Segura, S., Pérez-Solà, C., Navarro-Arribas, G., Herrera-Joancomartí, J.: Analysis of the Bitcoin UTXO set. In: Financial Cryptography and Data Security. pp. 78–91. Springer Berlin Heidelberg, Berlin, Heidelberg (2019)
5. Kumar, A., Lafourcade, P., Lauradoux, C.: Performances of cryptographic accumulators. In: IEEE 39th Conference on Local Computer Networks, LCN 2014, Edmonton, AB, Canada, 8–11 Sept 2014. pp. 366–369. IEEE Computer Society (2014)
6. Loporchio, M., Bernasconi, A., Di Francesco Maesa, D., Ricci, L.: Authenticating spatial queries on blockchain systems. IEEE Access **9**, 163363–163378 (2021)
7. Menezes, A., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press (1996)
8. Merkle, R.C.: Protocols for public key cryptosystems. In: Proceedings of the 1980 IEEE Symposium on Security and Privacy, Oakland, California, USA, April 14–16, 1980. pp. 122–134. IEEE Computer Society (1980)
9. Nakamoto, S.: Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf (2008). Accessed 12 June 2022
10. Tamassia, R.: Authenticated data structures. In: Algorithms — ESA 2003, 11th Annual European Symposium, Budapest, Hungary, 16–19 Sept 2003. Proceedings. Lecture Notes in Computer Science, vol. 2832, pp. 2–5. Springer (2003)
11. The Bitcoin Core developers: Bitcoin Dust Limit. https://github.com/bitcoin/bitcoin/blob/c536dfbcb00fb15963bf5d507b7017c241718bf6/src/policy/policy.cpp. Accessed 12 June 2022
12. Wang, Y., Yang, J., Li, T., Zhu, F., Zhou, X.: Anti-dust: a method for identifying and preventing blockchain's dust attacks. In: 2018 International Conference on Information Systems and Computer Aided Education (ICISCAE). pp. 274–280 (2018)
13. Wood, G., et al.: Ethereum: a secure decentralised generalised transaction ledger. Ethereum Proj. Yellow Paper **151**(2014), 1–32 (2014)
14. Xie, M., Wang, H., Yin, J., Meng, X.: Providing freshness guarantees for outsourced databases. In: EDBT 2008, 11th International Conference on Extending Database Technology, Nantes, France, 25–29 March 2008, Proceedings. ACM International Conference Proceeding Series, vol. 261, pp. 323–332. ACM (2008)
15. Xu, C., Zhang, C., Xu, J.: vChain: enabling verifiable boolean range queries over blockchain databases. In: Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, 30 June – 5 July 2019. pp. 141–158. ACM (2019)
16. Yang, Y., Papadopoulos, S., Papadias, D., Kollios, G.: Authenticated indexing for outsourced spatial databases. VLDB J. **18**(3), 631–648 (2009)
17. Zhu, Y., Zhang, Z., Jin, C., Zhou, A.: Enabling generic verifiable aggregate query on blockchain systems. In: 26th IEEE International Conference on Parallel and Distributed Systems, ICPADS 2020, Hong Kong, 2–4 Dec 2020. pp. 456–465. IEEE (2020)

Optimal Bond Percolation in Networks by a Fast-Decycling Framework

Leilei Wu and Xiao-Long Ren

Abstract Keeping a physical distance and creating social bubbles are popular measures that have been implemented to prevent infection and slow transmission of COVID-19. Such measures aim to reduce the risk of infection by decreasing the interactions among social networks. This, theoretically, corresponds to the optimal bond percolation (OBP) problem in networks, which is the problem of finding the minimum set of edges whose removal or deactivation from a network would dismantle it into isolated sub-components at most size C. To solve the OBP problem, we proposed a fast-decycling framework composed of three stages: (1) recursively removes influential edges from the 2-core of the network, (2) breaks large trees, and (3) reinserts the unnecessarily removed edges through an explosive percolation process. The proposed approaches perform better than existing OBP algorithms on real-world networks. Our results shed light on the faster design of a more practical social distancing and social bubble policy.

Keywords Network-dismantling · Optimal bond percolation · Robustness · Explosive percolation

1 Introduction

The process of globalization creates many opportunities, but sometimes also brings side effects that may cause damage to our societies [1, 2]. One recent example is the quick global contagion of COVID-19 [3], which has infected more than 5.3 bil-

L. Wu · X.-L. Ren (✉)
Yangtze Delta Region Institute (Huzhou), University of Electronic Science and Technology of China, Huzhou 313001, People's Republic of China
e-mail: renxiaolong@csj.uestc.edu.cn

L. Wu
Department of Physics, University of Fribourg, Fribourg 1700, Switzerland

Alibaba Business School, Hangzhou Normal University, Hangzhou 311121, People's Republic of China

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_40

lion people and killed more than 6.3 million people worldwide as of June 2022 [4]. Some measures have been implemented to prevent infection and slow transmission of COVID-19, such as keeping a physical distance and creating social bubbles [5, 6]. Such measures are intended to reduce the risk of infection by decreasing the interactions among social networks. This process, theoretically, corresponds to the optimal bond percolation in complex networks [7, 8]. Optimal bond percolation (OBP) is the problem of finding the minimum set of edges whose removal or deactivation from a network would dismantle it into isolated sub-components, i.e., social bubbles, at most size C.

The study of optimal site/bond percolation in networks by removing nodes or edges has been a long time [9–16]. For the wide applications in a broad social-economic scenario, there has been a lot of research on a similar problem under the name of network-dismantling [17–19], influence maximization [20, 21], or network immunization [22, 23]. Solutions to the OBP problem are also a theoretical fundamental of the strategies to cope with the epidemic, such as social distancing and population immunization when the resource is limited. However, finding the exact minimal set of nodes or edges for general networks is an NP-hard problem [12, 17] which means that we cannot find the exact smallest set in nondeterministic polynomial time. Many researchers approximated this problem by different methods of linear programming [24], semidefinite programming [25, 26], spectral partitioning [11, 13, 18], and deep reinforcement learning techniques [27]. Some of them achieved performance that is close to the theoretically optimal solution [17, 27–29]. These algorithms, however, are global and need to compute certain equations of the whole system repeatedly to obtain the solutions.

Inspired by the decycling approach to solving the optimal site percolation problem in ref. [17, 30], we proposed a 2-core-based fast-decycling framework to address the optimal bond percolation problem. This is based on the truth that all the loops of a network are contained in its 2-core structure. Thus, decycling the 2-core structure means breaking all the loops in the networks, and only trees remain after the decycling process. Then we introduced two categories of algorithms based on this framework: localized (decentralized) algorithms (CoreHS) and globalized algorithms (CoreHB and CoreHCI). To put it simply, the 2-core-based framework is composed of three stages: (1) recursively removes edges of the highest importance from the 2-core structure of the network, (2) recursively breaks the trees in the remaining network [27], and (3) reinserts the unnecessarily removed edges into the networks through the explosive percolation [31] process. See the illustration and a toy example in Fig. 1. In Sect. 3, results show that these 2-core-based approaches perform better than existing simple OBP algorithms and are as good as the state-of-art algorithms when applied on real-world networks.

2 Materials and Methods

This section describes some empirical network data sets used in this study and some classical algorithms we used as ingredients when the proposed framework was imple-

mented. Then, several indicators to evaluate the performances of algorithms were introduced. Further, we depicted the proposed 2-core-based framework for optimal bond percolation problem in detail. At last, two categories of algorithms implemented based on the framework were explained.

2.1 Data Sets

In this study, we mainly focus on some popular used social networks and infrastructure networks to show the performance of the proposed algorithms and baseline algorithms. (a) Petster Network [16]. This is an interest-based social network with 2,000 users and 16,098 undirected friendships among all the users on the website hamsterster.com. The raw data set was downloaded from KONECT [32] (http://konect.unikoblenz.de, last visited 2017). (b) Corruption Network. This data set comes from well-documented political corruption scandals over two decades in Brazil [33]. The giant/largest connected component (GCC) of the corruption network contains 309 nodes and 3,281 interactions [18]. (c) Crime Network. This is a network with 754 nodes representing people obtained from a set of criminal records [32]. If two people have committed a crime together, they will share a link. There are 2,127 links in this data set [19]. (d) USAir Network. This network represents the US air transportation system, consisting of 332 airports and 2126 airlines nationwide [34]. (e) PowerGrid Network. This is an undirected Power Grid network of the Western States in the United States of America, in which edges are power supply lines and nodes are generators or transformators or substations. The original network data contains 4,941 nodes and 6,594 edges and can be downloaded from KONECT [32] (http://konect.cc/networks/opsahl-powergrid/).

2.2 Classical Algorithms

In this subsection, some classical algorithms involved in this research will be introduced briefly. These algorithms will be used as ingredients or baseline algorithm when the proposed framework was implemented later.

(1) Bond percolation. Bond percolation was firstly introduced by Broadbent and Hammersley in 1957 [35]. In the study of network-dismantling problems, bond percolation is a random process of uniformly removing or recovering edges from networks. This random method is usually used as a baseline to show the performance of different edge-removal algorithms.
(2) Explosive percolation [31]. It is the process that recovering edges from networks according to some rules rather than randomly. For example, in Stage 3 of the 2-core-based link removal framework of OBP, the sum-rule-based explosive

percolation minimizes the sum of the sizes of clusters when recovering one edge from K randomly selected candidates at each step.

(3) K-core decomposition [36, 37]. K-core is the largest subgraph composed of nodes with remaining degrees of at least k after all the nodes with degrees smaller than k was removed progressively. K-core is an efficient method for identifying influential nodes in large-scale networks [38]. Recently, the Generalized k-core [39, 40] is also studied intensively in the topics of network robustness and stability.

(4) CoreHD [30]. CoreHD is a simple and extremely fast network dismantling algorithm that progressively removes nodes with the highest degree from the 2-core of the network. This method is based on the idea that all the loops of a network are contained in its 2-core structure [17]. Thus, decycling the 2-core structure means breaking all the loops in the networks, and only trees remain after the decycling process.

(5) Collective Influence (CI) [20]. The CI algorithm identifying a minimal set of influencers (nodes) which if immunized would prevent the diffusion of epidemic. Each nodes is evaluated by counting the degree of the neighbors belonging to the frontier of its neighboring ball with radius l. CI algorithm removes nodes with the highest CI value progressively and reinserts the unnecessarily removed nodes at the end.

2.3 Evaluation of the Solutions to the OBP Problem

A solution to the OBP problem is usually a set (or list) of links. After the set of links was removed from the network, the GCC size of the remaining network is not greater than C. Usually, there are several ways [27, 41] to test the performance of different algorithms that address the OBP problem. One most common and straightforward way is the proportion of the links that should be removed such that the size of the GCC is at most C. This indicator reflects the capability of algorithms to break the connectivity of networks. In practice, however, one scalar is usually not enough to reflect the performances of the algorithms during the whole removal process. Thus, the curve of GCC size versus the link removal proportion is further employed to evaluate the solutions to the OBP problem. Every solution corresponds to one curve. The lower the curve, the better the solution. See the example shown in Fig. 3. What is more, the area under the curve (AUC) puts more weight on the removal effects throughout the process. See the example shown in Fig. 2. In addition to these indicators, some research also considered the effect of the removal on the dynamics of the networks [42, 43], such as the spreading process, synchronization, the evolution of cooperation, etc. These will be the direction we are working towards.

2.4 *The 2-core-based Framework and Algorithms to Approximate the OBP Problem*

The 2-core of a network can be constructed by iteratively deleting nodes with degrees smaller or equal to one, i.e., isolated nodes or leaves. Actually, 2-core is a subgraph composed of the nodes with a (remaining) degree of at least two. See the toy example in the first row of Fig. 1. Intuitively we can find that all the loops appear only in the 2-core of the network. Thus, to dismantle the network into small pieces, one just needs to break all the loops, i.e., break its 2-core structure iteratively [17]. After this decomposition process, the network (graph) has only some trees left. In order to achieve the goal of optimal site/bond percolation problem, all we need to do is just break the trees larger than C. Inspired by the above thoughts, Zdeborová et al. [30] proposed the CoreHD algorithm to address the optimal **site** percolation problem (aka, network-dismantling problem) by iteratively removing node with highest degree in the 2-core. Following this approach, we proposed a similar framework to approximate the optimal bond percolation problem in this study by iteratively removing most influential link in the 2-core. Further, we implemented algorithms that are localized (decentralized) or globalized. The framework is described below:

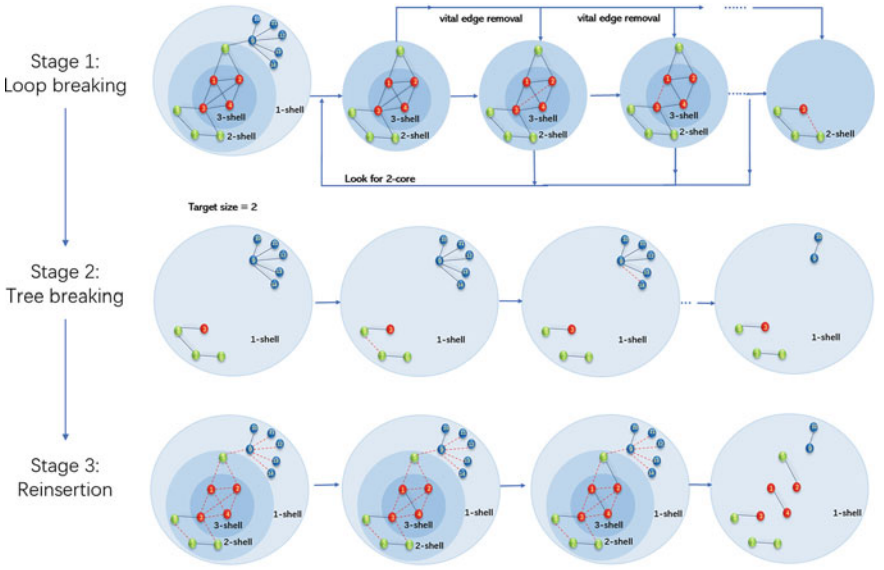


Fig. 1 An overview of the proposed 2-core-based fast-decycling framework that can approximate the optimal bond percolation problem in networks. The framework includes three stages. (1) Loop breaking: Recursively removes most influential edges from the remaining 2-core of the network. (2) Tree breaking: Recursively dismantles the largest tree in the remaining network. (3) Link reinsertion: Restores the unnecessarily removed edges into the networks through the explosive percolation [31] process. The reinsertion of the links should not lead the GCC size of the network larger than C

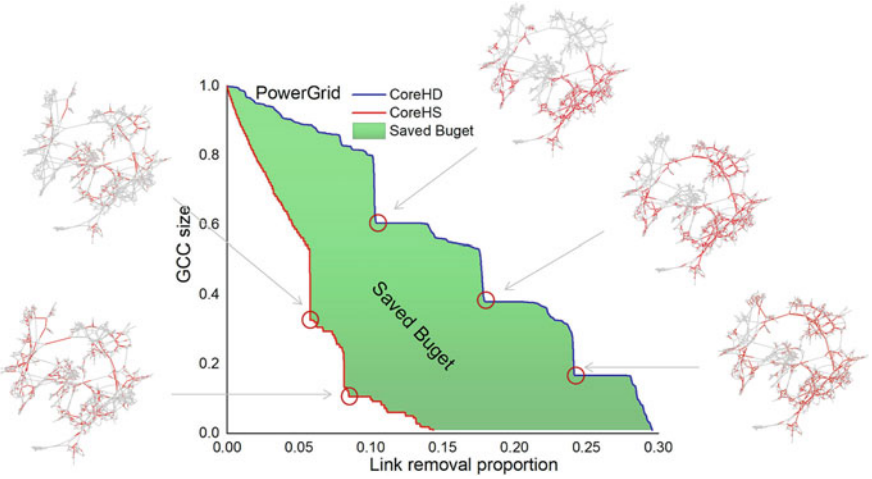


Fig. 2 Performance of the CoreHD algorithm and CoreHS algorithm on the OBP problem, the PowerGrid example. The X-axis is the removed proportion of links from the networks, and Y-axis is the size of the GCC after edge removal. The red lines in the small visualizations are the links that were removed from the network. We can easily see which part of the links was removed when the dismantling threshold was set as a different value

The 2-core-based link removal framework
Input: Adjacency matrix of a network A
Output: A network with GCC smaller than C

- Stage 1 (a). Getting the 2-core structure of the network by iteratively deleting nodes a degree equal to zero or one, i.e., isolated nodes and leaves, until all the nodes have a degree of at least two.
- Stage 1 (b). Iteratively **finding and removing the most influential link** in the remaining 2-core structure and updating the 2-core of the network according to (a). Repeating this step until no nodes belongs to the 2-core.
- Stage 2. Iteratively searching and breaking the largest tree by removing the most central link, until the sizes of all the trees are at most C.
- Stage 3. Restoring the unnecessarily removed edges into the network through the sum-rule-based explosive percolation process [31]. The reinsertion of the links should not lead the GCC size of the remaining network larger than C.

Based on this framework above, we can define some localized (decentralized) or globalized algorithms by choosing different implementation methods of "finding and removing the most influential link" in Stage 1 (b). For example, the famous core-based high degree (CoreHD) algorithm [30] is a typical localized nodal removal approach. As a link removal counterpart of CoreHD, core-based high strength (CoreHS) algorithm has been studied in detail in this research. The strength of a link is defined as the product of the degrees of its two endpoints. High strength indicates that CoreHS always finds and removes the most influential link with the highest strength. Accord-

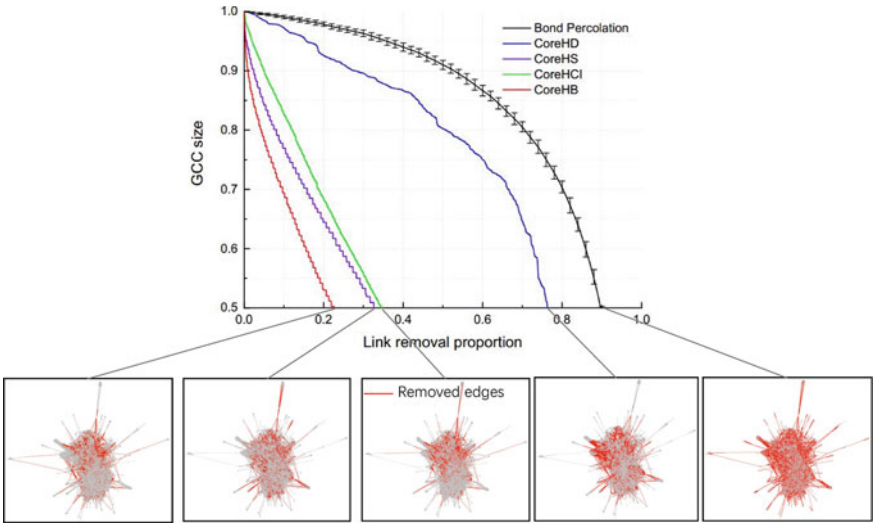


Fig. 3 Comparison of the CoreHD, CoreHS, CoreHCI, CoreHB algorithm with the baseline method bond percolation, the petster network example. The red lines in the small visualizations are the links that was removed from the network at the moment when GCC size is reach the 50% of the original size

ing to the similar way, we also detailed studied the core-based high Collective Influence (CoreHCI) algorithm and core-based high Betweenness (CoreHB) algorithm. Obviously, these two algorithms are globalized algorithms. The localized (decentralized) algorithms are usually much faster and are easily used in the situation when global information is difficult to obtain. The globalized algorithms are generally good at approximating the minimum removal set but time-consuming.

Besides these implementation methods above, the simple bond percolation [44] process (that is, removing edges randomly) is used as a baseline to test the evaluation of all the methods.

Let's add here **how to find the most central link in a tree** in Stage 2. This is different with the process to identify the most influential link in Stage 1(b). To find the central link and break the largest tree in the remaining networks, we initially set one unit resource on every node of the tree. Then all the leaves will be removed, and the number of their resources will be recorded in the first step. At the same time, the resources of the removed leaves will be transferred to their only neighbor. Repeating the above process until all the nodes in the tree are removed. At this moment, every node corresponds to a resource value. Based on this, the centrality of a link in the original tree can be computed by the product of the resource values of its two endpoints. At last, we can quickly get the central link with the largest centrality and break the tree by removing it. This process is similar to the approach in Ref. [27], but their approach is designed to break the tree by removing central nodes.

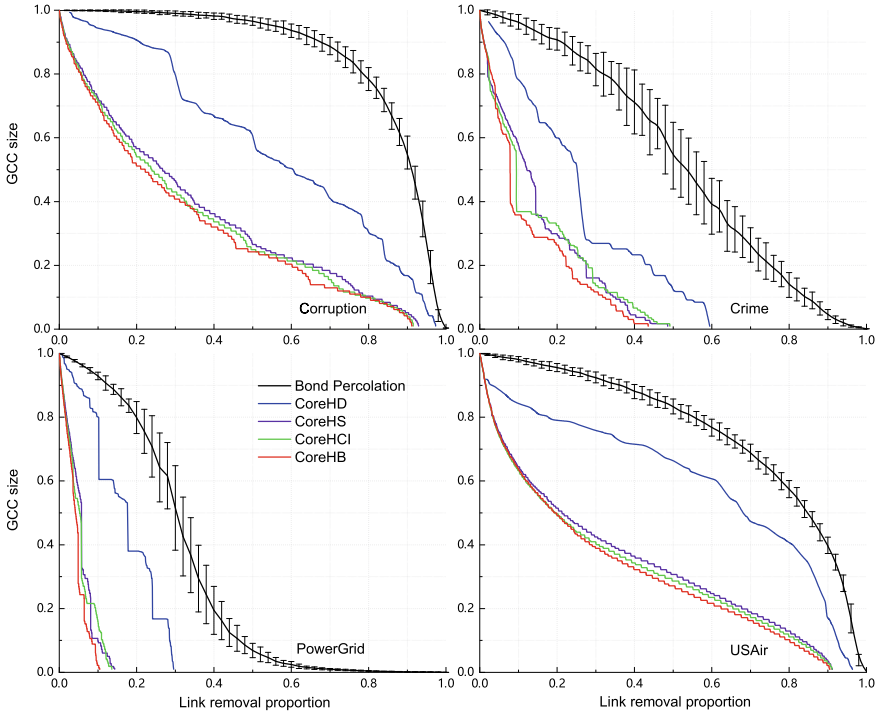


Fig. 4 Performance of the introduced two categories of algorithms based on the 2-core-based framework: localized (decentralized) algorithms (CoreHD and CoreHS) and globalized algorithms (CoreHCI and CoreHB). The X-axis is the removed proportion of links from the networks, and Y-axis is the size of the GCC after edge removal. Four empirical networks were investigated in this figure: corruption network, crime network, PowerGrid network, and USAir network

After breaking all the loops and big trees in Stage 1 and 2, there would be many unnecessarily removed edges from a final perspective. One needs to restore as many edges as possible to approximate the OBP problem. The process of recovering edges one by one is similar to the bond percolation. To slow down the growth of the sizes of the clusters, the sum-rule-based explosive percolation will be applied which minimizes the sum of the sizes of clusters when recovering one edge from K randomly selected candidates at each step [19, 31].

3 Result and Discussions

In this section, we compared the performance of the core-based algorithms to approximate the OBP problem. Firstly, let's see the comparison of CoreHD and CoreHS algorithm in Fig. 2. These are the results of the PowerGrid network example. The

X-axis is the removed proportion of links from the networks, and Y-axis is the size of the GCC after edge removal. The lower the curve, the better the performance of the algorithm. Because the CoreHD is designed for optimal site percolation problems, i.e., nodal removal problems, its performance is not as good as CoreHS. In this figure, the red lines in the small visualizations are the links that was removed from the network. We can easily see which part of the links were removed when the dismantling threshold was set as a different value (Fig. 3). Then, let's compare all the algorithms we mentioned above in the same network, the Petster network, and visualize all the links that were removed at the point when the size of GCC is at most 50% of the original network. We can see that in this empirical network, CoreHB performs the best, and CoreHS and CoreHCI also have a good performance. They are also much better than the CoreHD and the benchmark algorithms.

At last, we detailed investigated the performance of the localized (decentralized) algorithms (CoreHD and CoreHS) and globalized algorithms (CoreHCI and CoreHB) in Fig. 4. The X-axis is the removed proportion of links from the networks, and Y-axis is the size of the GCC after edge removal. Four empirical networks were investigated in this figure: Corruption Network, Crime Network, PowerGrid Network, and USAir network. Because the computation process of CoreHS is extremely similar to CoreHD, the computational time of CoreHS is also fast. The 2-core structure can be obtained with $O(N$ leaf-removal operations. After removing one edge, we only needs to update the 2-core structure with $O(1)$ operations on average in sparse networks. We can conclude that the localized (decentralized) algorithms are usually much faster and are easily used in situation when global information is difficult to obtain. The globalized algorithms are usually good at approximating the minimum removal set.

4 Conclusion

In this paper, we proposed a 2-core-based fast-decycling framework to address the optimal bond percolation problem. The 2-core-based framework is composed of three stages: (1) Loop breaking: Recursively removes most influential edges from the remaining 2-core of the network. (2) Tree breaking: Recursively dismantles the largest tree in the remaining network. (3) Link reinsertion: Restores the unnecessarily removed edges into the networks through the explosive percolation process. We introduced two categories of algorithms based on this framework: localized (decentralized) algorithms (CoreHS) and globalized algorithms (CoreHCI and CoreHB). The localized (decentralized) algorithms are usually much faster and are easily used in the situation when global information is difficult to obtain. The globalized algorithms are generally good at approximating the minimum removal set but time-consuming.

Acknowledgements This work was supported by the Postdoctoral Research Fund of China (No. 2022M710620), and Science and Technology Foundation of Huzhou (No. 2021YZ12).

References

1. Helbing, D.: Globally networked risks and how to respond. Nature **497**(7447), 51–59 (2013)
2. Alvarez-Rodriguez, U., Battiston, F., de Arruda, G.F., Moreno, Y., Perc, M., Latora, V.: Evolutionary dynamics of higher-order interactions in social networks. Nat. Human Behav. pp. 1–10 (2021)
3. Acuto, M.: Covid-19: lessons for an urban (izing) world. One Earth **2**(4), 317–319 (2020)
4. World Health Organization.: Who coronavirus (covid-19) dashboard. https://covid19.who.int. [Online; Accessed 20 June 2022]. Globally, as of 5:24 pm CEST, 17 June 2022, there have been 535,863,950 confirmed cases of COVID-19, including 6,314,972 deaths, reported to WHO. As of 15 June 2022, a total of 11,902,271,619 vaccine doses have been administered (2022)
5. Phillips, N., et al.: The coronavirus is here to stay-here's what that means. Nature **590**(7846), 382–384 (2021)
6. Lokhande, T., Yang, X., Xie, Y., Cook, K., Liang, J., LaBelle, S., Meyers, C.: Gis-based classroom management system to support covid-19 social distance planning. Comput. Urban Sci. **2**(1), 1–12 (2022)
7. Ren, X.-L.: The dismantling problem in complex networks and its applications. Ph.D. thesis, ETH Zurich (2021)
8. Del Ferraro, G., Moreno, A., Min, B., Morone, F., Pérez-Ramírez, Ú., Pérez-Cervera, L., Parra, L.C., Holodny, A., Canals, S., Makse, H.A.: Finding influential nodes for integration in brain networks using optimal percolation theory. Nat. Commun. **9**(1), 2274 (2018)
9. Fiedler, M.: Algebraic connectivity of graphs. Czech. Math. J. **23**(2), 298–305 (1973)
10. Lipton, R., Rose, D., Tarjan, R.: Generalized nested dissection. SIAM J. Numer. Anal. **16**(2), 346–358 (1979)
11. Pothen, A., Simon, H.D., Liou, K.-P.: Partitioning sparse matrices with eigenvectors of graphs. SIAM J. Matrix Anal. Appl. **11**(3), 430–452 (1990)
12. Bui, T.N., Jones, C.: Finding good approximate vertex and edge partitions is NP-hard. Inf. Process. Lett. **42**(3), 153–159 (1992)
13. Guattery, S., Miller, G.L.: On the quality of spectral separators. SIAM J. Matrix Anal. Appl. **19**(3), 701–719 (1998)
14. Requião da Cunha, B., González-Avella, J.C., Gonçalves, S.: Fast fragmentation of networks using module-based attacks. PLoS ONE **10**(11), 1–15 (2015)
15. Tian, L., Bashan, A., Shi, D.-N., Liu, Y.-Y.: Articulation points in complex networks. Nat. Commun. **8**(1), 1–9 (2017)
16. Ren, X.-L., Gleinig, N., Tolić, D., Antulov-Fantulin, N.: Underestimated cost of targeted attacks on complex networks. Complexity **2018** (2018)
17. Braunstein, A., Dall'Asta, L., Semerjian, G., Zdeborová, L.: Network dismantling. Proc. Natl. Acad. Sci. U.S.A. **113**(44), 12368–12373 (2016)
18. Ren, X.-L., Gleinig, N., Helbing, D., Antulov-Fantulin, N.: Generalized network dismantling. Proc. Natl. Acad. Sci. U.S.A. **116**(14), 6554–6559 (2019)
19. Ren, X.-L.: Network dismantling: from vertex separator to edge separator using explosive percolation. (Res. Gate) preprint (2020)
20. Morone, F., Makse, H.A.: Influence maximization in complex networks through optimal percolation. Nature **524**(7563), 65–68 (2015)
21. Chen, W., Wang, C., Wang, Y.: Scalable influence maximization for prevalent viral marketing in large-scale social networks. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1029–1038. ACM (2010)
22. Chen, Y., Paul, G., Havlin, S., Liljeros, F., Stanley, H.E.: Finding a better immunization strategy. Phys. Rev. Lett. **101**(5), 58701 (2008)
23. Clusella, P., Grassberger, P., Pérez-Reche, F.J., Politi, A.: Immunization and targeted destruction of networks using explosive percolation. Phys. Rev. Lett. **117**, 208301 (2016)
24. Leighton, T., Rao, S.: Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. J. ACM **46**(6), 787–832 (1999)

25. Arora, S., Rao, S., Vazirani, U.: Expander flows, geometric embeddings and graph partitioning. In: Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing — STOC 04. ACM Press (2004)
26. Feige, U., Hajiaghayi, M., Lee, J.R.: Improved approximation algorithms for minimum weight vertex separators. SIAM J. Comput. **38**, 629–657 (2008)
27. Fan, C., Zeng, L., Sun, Y., Liu, Y.-Y.: Finding key players in complex networks through deep reinforcement learning. Nat. Mach. Intell. 1–8 (2020)
28. Mugisha, S., Zhou, H.-J.: Identifying optimal targets of network attack by belief propagation. Phys. Rev. E **94**, 012305 (2016)
29. Ren, X.-L., Antulov-Fantulin, N.: Ensemble approach for generalized network dismantling. In: International Conference on Complex Networks and Their Applications, pp. 783–793. Springer (2019)
30. Zdeborová, L., Zhang, P., Zhou, H.-J.: Fast and simple decycling and dismantling of networks. Sci. Rep. **6**, 37954 (2016)
31. Achlioptas, D., D'Souza, R.M., Spencer, J.: Explosive percolation in random networks. Science **323**(5920), 1453–1455 (2009)
32. Kunegis, J.: The Koblenz network collection. In: Proceedings of the International Web Observatory Workshop, pp. 1343–1350 (2013)
33. Ribeiro, H.V., Alves, L.G.A., Martins, A.F., Lenzi, E.K., Perc, M.: The dynamical structure of political corruption networks. J. Complex Netw. **6**, 989–1003 (2018)
34. Xu, Z., Harriss, R.: Exploring the structure of the us intercity passenger air transportation network: a weighted complex network approach. GeoJournal **73**(2), 87–102 (2008)
35. Broadbent, S.R., Hammersley, J.M.: Percolation processes: I. Crystals and mazes. In: Mathematical Proceedings of the Cambridge Philosophical Society, vol. 53, pp. 629–641. Cambridge University Press (1957)
36. Seidman, S.B.: Network structure and minimum degree. Soc. Netw. **5**(3), 269–287 (1983)
37. Dorogovtsev, S.N., Goltsev, A.V., Mendes, J.F.F.: K-core organization of complex networks. Phys. Rev. Lett. **96**(4), 040601 (2006)
38. Kitsak, M., Gallos, L.K., Havlin, S., Liljeros, F., Muchnik, L., Stanley, H.E., Makse, H.A.: Identification of influential spreaders in complex networks. Nat. Phys. **6**(11), 888–893 (2010)
39. Shang, Y.: Attack robustness and stability of generalized k-cores. New J. Phys. **21**(9), 093013 (2019)
40. Shang, Y.: Generalized k-cores of networks under attack with limited knowledge. Chaos, Solitons Fractals **152**, 111305 (2021)
41. Wandelt, S., Sun, X., Feng, D., Zanin, M., Havlin, S.: A comparative analysis of approaches to network-dismantling. Sci. Rep. **8**(1), 1–15 (2018)
42. Liu, X., Li, D., Ma, M., Szymanski, B.K., Stanley, H.E., Gao, J.: Network resilience. Phys. Rep. **971**, 1–108 (2022)
43. Cohen, R., Havlin, S.: Complex networks: structure, robustness and function. Cambridge University Press (2010)
44. Callaway, D.S., Newman, M.E.J., Strogatz, S.H., Watts, D.J.: Network robustness and fragility: percolation on random graphs. Phys. Rev. Lett. **85**(25), 5468–5471 (2000)

Motif Discovery in Complex Networks

Integrating Temporal Graphs via Dual Networks: Dense Graph Discovery

Riccardo Dondi, Pietro Hiram Guzzi, and Mohammad Mehdi Hosseinzadeh

Abstract Interactions among objects are usually modelled using graphs. Nevertheless, these relations may change over time and there exist different kind of relations among object that need to be integrated. We introduce a new network model, called temporal dual networks, to deal with interactions that changes over time and to integrate information coming from two different networks. We consider a fundamental problem in graph mining, that is finding densest subgraphs on this new model. We propose an approach based on both network alignment and dynamic programming. Given two temporal graphs, we obtain a dual temporal graph via alignment and then we look for densest subgraphs in the obtained graph. We present a dynamic programming algorithm to solve the problem in polynomial time. Since this algorithm is not applicable even to medium size network, we present a heuristic that is based on (1) constraining the dynamic programming to consider only bounded temporal graphs and (2) a local search procedure. We show that our method is able to return optimal or near optimal solution even for temporal graphs having 10,000 vertices and 10,000 timestamps.

1 Introduction

Novel network models have been introduced in graph theory and graph mining to extend the classic graph model in order to represent properties of complex systems. For example, temporal information about interactions are represented in temporal

R. Dondi · M. M. Hosseinzadeh (✉)
Università degli Studi di Bergamo, Bergamo, Italy
e-mail: m.hosseinzadeh@unibg.it

R. Dondi
e-mail: riccardo.dondi@unibg.it

P. H. Guzzi
Magna Graecia University, Catanzaro, Italy
e-mail: hguzzi@unicz.it

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_41

graphs [5, 14, 16], while integration of different kinds of relationships is considered in dual graphs [6, 17] and network of networks [10].

In this paper we introduce a new network model, called temporal dual network, in order to integrate interactions that come from two different networks (as in dual networks) and that change over time (as in temporal graphs). The new model can be useful to analyze the evolution of networks, in particular their cohesive parts. For example, consider the case where we want to analyze a community an author belongs to. The idea is to consider two networks, a co-authorship network (conceptual network) and a network based on research interest (physical network). The community an author belongs to may be not static, but dynamic, as she/he may have new coauthors or may strengthen the relations with an existing author (by publishing more papers, for example) or again a relation may be weaken over time. On the other hand, an author may change her/his research interests over time. For these reason, considering only static graphs is not enough to represent these dynamics, but we have to consider how networks/communities change over time. Here we consider two temporal networks (a conceptual and a physical temporal networks) that represents different information, for example a conceptual temporal network represents a co-authorship temporal network; a physical network research interests.

Another example of application is the analysis of social networks to understand the preferences of a users, as it may change some interests over time and this may be inferred from the context she/he considers on a platform and from new relations she/he establishes.

In this paper, we consider a fundamental problem in graph mining: the identification of dense subgraphs in the context of temporal dual networks. The identification of cohesive subgraphs is a fundamental problem in graph mining, since it is related to the identification of cohesive groups [4, 7, 8, 12]. An analysis of the evolution of motifs in temporal networks has been proposed in [1] and the identification of dense subgraphs has been recently considered for temporal networks [2, 5, 16]. We propose a problem for the identification of k densest subgraphs that are temporal disjoint in a temporal dual networks and we design a heuristic for it. This method is based on (1) computing an alignment graph of the conceptual and physical graph and (2) finding k densest subgraphs in the alignment graph. For this second step, we devise two algorithms: an exact dynamic programming algorithm, which is applicable only for small datasets and a heuristic. This heuristic is based on solving a constrained version of the problem via dynamic programming and then applying a local search approach. We present an experimental evaluation of these algorithms on synthetic datasets, generated varying the number of timestamps (from 70 to 10,000 and the number of nodes from 70 to 10,000.

The paper is organized as follows. First, in Sect. 2 we give the definitions that will be useful in the remaining part of the paper and we present the dual temporal graph model. Then, in Sect. 3 we present the algorithmic contributions of the paper, while in Sect. 4 we will present an experimental evaluation of our heuristic on synthetic datasets. Finally, we conclude the paper with some open problems.

2 Definitions

In this section, we start by giving the definitions of temporal graphs and dual graphs, and we introduce the temporal dual graph model. Then we present the formal definition of the problem we are interested into, that is finding k densest subgraphs that are active into disjoint intervals.

We start by introducing a discrete time domain over which is defined a temporal graph and a temporal dual graph.

Definition 1 A discrete time domain $\mathcal{T} = [0, 1, ..., t_{\max}] \subseteq \mathbb{N}$, where each $t \in \mathcal{T}$ is called a timestamp. We define an interval $T = [t_i, t_j]$ over $\mathcal{T}$, with $t_i, t_j \in \mathcal{T}$ and $t_i < t_j$, consists of the timestamp between t_i and t_j.

Two intervals are *disjoint* if they do not share any timestamp. Next, we can present the definition of temporal graph. Notice that in the model we consider the set of nodes is not changing over time.

Definition 2 $G = (V, \mathcal{T}, E)$ is a temporal graph, where V is a set of nodes, and $E \subseteq V \times V \times \mathcal{T}$ is a set of temporal edges.

Given a temporal graph $G = (V, \mathcal{T}, E)$ and a temporal interval T, we define $G[T] = (V, E[T])$ as the *active graph* of G in interval T, where $E[T]$ is the set of active edges at interval T, defined as: $E[T] = \{\{u, v, t\} | \{u, v, t\} \in E \wedge t \in T\}$.

A similar definition of active edges can be given for active edges at timestamp $t \in \mathcal{T}$: $E[t] = \{\{u, v, t\} | \{u, v, t\} \in E\}$.

We can now define the concept of episodes, which represent the temporal subgraphs we will look for.

Definition 3 Let $G = (V, \mathcal{T}, E)$ be a temporal graph, an episode is defined as a pair (T, W) where T is an interval over $\mathcal{T}$ and W is a subgraph of $G[T]$, that is $W = (V_W, E_W)$, where $V_W \subseteq V$ and $E_W \subseteq E[T] \cap (V_W \times V_W)$.

Given a weighted temporal graph $G = (V, \mathcal{T}, E)$, an interval I over $\mathcal{T}$ and an edge $(u, v) \in E$, then the *average weight* of (u, v) in E, denoted by $w_I(u, v)$, is defined as follows:

$$w_I(u, v) = \frac{\sum_{t \in I} w(u, v, t)}{\sqrt{|I|}}$$

where $w(u, v, t)$ is the weight of edge (u, v) at time t. We divide by $\sqrt{|I|}$ and not by $|I|$, since in the latter case this may lead to dense subgraphs defined in a single timestamp.[1]

The weighted density of G in a interval I, denoted by $w - dens(G, I)$, is defined as follows:

$$w - dens(G, I) = \frac{\sum_{(u,v) \in E} w_I(u, v)}{|V|}.$$

[1] Here we use $\sqrt{|I|}$ but other sublinear functions can be considered as well.

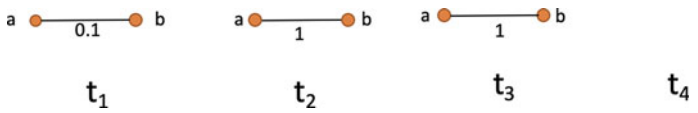


Fig. 1 A temporal weighted network with two nodes and four timestamps [1, 2, 3, 4]

Notice that the fact that the temporal graph is weighted changes some of the properties of episodes with respect to unweighted graphs. For example, while, as discussed in [16], the density of episodes in unweighted graphs is a monotone non decreasing function, this property does not hold in the weighted case, as it can be seen in the following example (Fig. 1), $w_{[2,3]}(u, v) = 1$, while $w_{[1,3]}(u, v) = 0.7$, $w_{[1,4]}(u, v) = 0.525$.

Now, we introduce the definition of dual graph.

Definition 4 $G = (V, E_c, E_p, w_c)$ is a dual graph, where V is a set of nodes, and $G_c = (V, E_c, w_c)$, $G_p = (V, E_p)$ are two graphs defined over the same set of nodes V such that:

- $G_c = (V, E_c, w_c)$ is a weighted graph, called conceptual graph
- $G_p = (V, E_p)$ is an unweighted graph, called physical graph.

Now, we are able to introduce the definition of Temporal Dual Graph.

Definition 5 $G = (V, \mathcal{T}, E_c, E_p, w_c)$ is a Temporal Dual Graph (TDG), where

- V is a set of nodes
- $G_c = (V, \mathcal{T}, E_c, w_c)$ is a weighted temporal graph, called conceptual temporal graph
- $G_p = (V, \mathcal{T}, E_p)$ is an unweighted temporal graph, called physical temporal graph.

Now, we are able to define a temporal densest common subgraph.

Definition 6 Temporal Common Subgraph.
Given a temporal dual graph $G = (V, \mathcal{T}, E_c, E_p, w_c)$, a temporal common subgraph in $G = (V, \mathcal{T}, E_c, E_p, w_c)$ is a pair (W, T) where $T \in \mathcal{T}$ is a temporal interval and $W \subseteq V$ such that:

- $G_p[W, T]$ is connected
- the weighted density of (W, T), denoted by $w - dens(W, T)$, is equal to $dens(G_c[W, T])$ (that is the density in the conceptual temporal graph).

We define the first problem we are interested into.

Problem 1 k-Densest-Episodes *in a Temporal Dual Graph*
Input*: A temporal dual graph $G = (V, \mathcal{T}, E_c, E_p, w_c)$, a positive integer $k \in \mathbb{N}$.*
Output*: A set S of k temporal common subgraphs $S = \{(I_j, W_j) : 1 \leq j \leq k\}$, where $\{I_j : 1 \leq j \leq k\}$ is a set of disjoint intervals, such that $\sum_{j=1}^{k} w - dens(W_j, I_j)$ is maximized.*

The k-Densest-Episodes problem is NP-hard, since, given a static dual graph (hence a temporal graph with a time domain consisting of a single timestamp), it is NP-hard to find a densest common subgraph [17].

In order to solve the problem, we consider the following alignment approach: (1) we first align the conceptual temporal graph and the physical temporal graph and we obtain a temporal alignment graph; (2) then we find a set of k episodes in the temporal alignment graph.

2.1 Graph Alignment Approach

Here we describe the alignment approach we propose to solve the k-Densest-Episodes problem on Temporal Dual Graphs. Graph alignment has already been considered to deal with dual graphs [11, 15]. Here we extend the definition to temporal dual graphs, by essentially defining an alignment for each timestamp t.

Definition 7 Given two input graphs, a weighted graph $G_a = (V_a, E_a, w_a)$ and an unweighted graph $G_b = (V_b, E_b)$, where E_a is a set of weighted edges and E_b is a set of unweighted edges, a graph alignment of G_a and G_b is formally defined as a mapping A from $V_a \rightarrow V_b$.

In particular, here we consider a variant of alignment called *local alignment* which is defined as a partial injective mapping A from V_a to V_b. In our case, the mapping (hence the alignment) of two graphs is implicitly defined by their identifiers, that is two corresponding vertices in the networks have the same identifier both in V_a and in V_b. The output of the alignment is a new weighted graph $G_{al} = (V_{al}, E_{al})$, called *alignment graph* and defined as follows.

Definition 8 Given a weighted graph $G_a = (V_a, E_a, w_a)$ and an unweighted graph $G_b = (V_b, E_b)$, an alignment graph $G_{al} = (V_{al}, E_{al}, w_{al})$, between G_a and G_b is defined as follows:

- The vertex set $V_{al} = \{c_i : (v_{ai}, v_{bi}) \in I\}$
- The edge set E_{al} is defined based on two possible cases: *match, and mismatch* and depends on a parameter δ. For each set $\{c_i, c_j\}$ of two vertices $c_i, c_j \in V_{al}$ corresponding to pairs (v_{ai}, v_{bi}),(v_{aj}, v_{bj}), respectively, then:

1. If both $(v_{ai}, v_{aj}) \in E_a$, and $(v_{bi}, v_{bj}) \in E_b$, then $(c_i, c_j) \in E_{al}$ with weight $w_{al}(c_i, c_j) = w_a(v_{ai}, v_{aj})$
2. If $(v_{ai}, v_{aj}) \in E_a$, and $(v_{bi}, v_{bj}) \notin E_b$, where v_{bi}, v_{bj} are at distance lower than δ in G_b, then $(c_i, c_j) \in E_{al}$ with weight $w_{al}(c_i, w_j)$ defined as the average weight of the edges of the path connecting v_{bi}, v_{bj} in G_b (**mismatch 1 case**)
3. If $(v_{ai}, v_{aj}) \in E_a$, and $(v_{bi}, v_{bj}) \notin E_b$, where v_{bi}, v_{bj} are at distance at least δ in G_b, then $(c_i, c_j) \notin E_{al}$ (**mismatch 2 case**)
4. If $(v_{ai}, v_{aj}) \notin E_a$, then $(c_i, c_j) \notin E_{al}$.

Fig. 2 The possible cases of the graph alignment

The output of the alignment is a new graph $G_{al} = (V_{al}, E_{al})$, called *alignment graph*. Figure 2 presents the three possible cases, where we draw G_a with blue vertices/edges and G_b with red vertices/edges.

Definition 9 Timestamp Alignment Graph. Given a temporal dual graph $G = (V, \mathcal{T}, E_c, E_p, w_c)$, for each timestamp $t \in \mathcal{T}$, a Timestamp Alignment Graph $G_A[t]$ is an alignment graph of the conceptual graph $G_c[t]$ and the physical graph $G_p[t]$ of the same timestamp t. A temporal alignment graph $G_A = (V, \mathcal{T}, E_A, w_A)$ is a collection of timestamp alignment graphs, one for each timestamp

$$G_A = \bigcup_{t=0}^{t_{max}} G_A[t]$$

2.2 Finding Episodes in the Alignment Graphs

Once the temporal alignment graph is computed, we consider the problem of finding a set of (weighted) episodes in it, as defined in the following problem.

Problem 2 k-Densest-Alignment-Episodes
Input: *A temporal alignment graph* $G_A = (V, \mathcal{T}, E_A, w_A)$, *a positive natural* $k \in \mathbb{N}$.
Output: *A set* S *of* k *temporal densest subgraphs* $S = \{(I_j, W_j) : 1 \leq j \leq k\}$, *where* $\{I_j : 1 \leq j \leq k\}$ *is a set of disjoint intervals, such that* $\sum_{j=1}^{k} w - dens(W_j, I_j)$ *is maximized.*

We consider also a variant of the problem, called k-ℓ-Densest-Alignment-Episodes, where there episodes are constrained to happen in a bounded interval.

Problem 3 k-ℓ-Densest-Alignment-Episodes
Input: *A temporal alignment graph* $G_A = (V, \mathcal{T}, E_A, w_A)$, *two positive natural* $\ell, k \in \mathbb{N}$, *with* $\ell \leq t_{max}$.
Output: *A set* S *of* k *temporal densest subgraphs* $S = \{(I_j, W_j)\} : 1 \leq j \leq k\}$, *where* $\{I_j : 1 \leq j \leq k\}$ *is a set of disjoint intervals each one of length at most* ℓ, *such that*

$\sum_{j=1}^{k} w - dens(W_j, I_j)$ *is maximized.*

We will show in Sect. 3 that, unlike the k-Densest-Episodes problem, k-Densest-Alignment-Episodes and k-ℓ-Densest-Alignment-Episodes can be solved in polynomial time.

2.3 The Densest Subgraph Problem

The approach we propose for solving the k-Densest-Alignment-Episodes and the k-ℓ-Densest-Alignment-Episodes problem is based on the computation of a solution of the Densest Subgraph problem on static (weighted) graphs. Given a graph the Densest Subgraph problem asks for a subgraph of maximum weighted density. The problem can be solved in polynomial-time [9] with Goldberg's algorithm, that is based on a reduction to a series of min-cut computation. The time complexity of the Goldberg's algorithm is $O(mn \log n)$ (also in $O(n^3)$ time for unweighted graphs [13]). Furthermore, the Densest Subgraph problem can be approximated within factor $\frac{1}{2}$ by a greedy algorithm of time complexity $O(n + m)$ for unweighted graphs and $O(m + n \log n)$ for weighted graphs [3]. In what follows, we denote by $t_{densest}$ the time required to compute a densest subgraph in a static graphs.

3 Algorithms for K-Densest-Alignment-Episodes and k-l-Densest-Alignment-Episodes

In this section we start by presenting dynamic programming polynomial-time algorithms for k-Densest-Alignment-Episodes (DP) and k-ℓ-Densest-Alignment-Episodes (L-DP).

We introduce the DP algorithm for k-Densest-Alignment-Episodes. Given an alignment graph G_A over the time interval $[1, j]$, with $j \leq t_{\max}$, we define the function $D[j, h]$, with $h \leq k$, as follows:
$D(j, h)$ = The density of h densest episodes in $G_A[1, j]$.

Given two timestamps i and j, with $1 \leq i \leq j \leq t_{\max}$, we denote by $Dens(G_A[i, j])$ the density of a densest subgraph in $G_A[i, j]$. Assume that $Dens(G_A[i, j])$ have already been computed for all values $1 \leq i \leq j \leq t_{\max}$, function $D(j, h)$ can be computed as follows:

$$D(j,h) = \begin{cases} \max \begin{cases} \max_{2 \leq i \leq j} D(i, h-1) + Dens(G_A[i+1, j]) \\ D(j-1, h) \end{cases} & \text{if } h \geq 2 \\ \max \begin{cases} \max_{1 \leq i \leq j} Dens(G_A[i, j]) \\ D(j-1, 1) \end{cases} & \text{if } h = 1 \\ -\infty & \text{if } j = 1 \text{ and } h \geq 2. \end{cases} \quad (1)$$

Lemma 1 *$D(j, h) = q$ if and only if there exist h episodes in $G_A[1, j]$ of overall density q.*

The previous lemma leads to the following result.

Theorem 1 k-Densest-Alignment-Episodes *can be solved in $O(t_{\max}^2\ k\ t_{densest})$ time.*

Next we present the L-DP algorithm for k-ℓ-Densest-Alignment-Episodes. Similarly to k-Densest-Alignment-Episodes, given an alignment graph G_A over the time interval $[1, j]$, with $j \leq t_{\max}$, we define the function $D_c[j, h]$, with $h \leq k$, as follows:
$D_c(j, h)$ = {The density of h densest constrained episodes in $G_A[1, j]$}.

Assume that given an alignment graph $G_A[i, j]$, $Dens(G_A[i + 1, j])$ denotes a constrained densest subgraph in $G_A[i, j]$.

The recurrence to compute $D_c(j, h)$, for each $j \in \{1, 2, \ldots, t_{\max}\}$ is defined as follows:

For $h \geq 2$:

$$D_c(j, h) = \max_{2 \leq i \leq j} \left\{ D_c(i - 1, h - 1) + Dens(G[i, j]) \text{ with } j - i + 1 \leq \ell \right. \tag{2}$$

For $h = 1$:

$$D_c(j, 1) = \max_{1 \leq i \leq j} \begin{cases} Dens(G[i, j]) \text{ with } j - i + 1 \leq \ell \\ D_c(j - 1, 1) \end{cases} \tag{3}$$

Finally, for if $j = 1$ and $h \geq 2$, $D_c(j, 1) = -\infty$.

Similarly to k-Densest-Alignment-Episodes, we can prove the following result.

Theorem 2 k-ℓ-Densest-Alignment-Episodes *can be solved in $O(t_{\max}\ \ell k\ t_{densest})$.*

3.1 A Heuristic for K-Densest-Alignment-Episodes

The time complexity of the dynamic programming to solve k-Densest-Alignment-Episodes makes it non practical even for medium size temporal graphs. Hence we propose a heuristic for k-Densest-Alignment-Episodes, which consists of two phases:

1. The L-DP algorithm for solving k-ℓ-Densest-Alignment-Episodes, with $\ell = \log_2(t_{\max})$, hence having time complexity $O(t_{\max} \log_2 t_{\max}\ k\ t_{densest})$
2. A local search procedure called LocExt that, starting from a solution returned in the first phase, aims at improving its density

Next, we describe the LocExt phase. LocExt starts from a solution S of k-ℓ-Densest-Alignment-Episodes and applies a procedure to possibly improve its density. Notice that an interval I of $\mathcal{T}$ is said to be *uncovered* by a solution S if there is no subgraph of S that contains a timestamp in I. LocExt looks for an improvement of S by greedily applying the following procedure:

- It considers two temporal subgraphs (I_j, W_j) and (I_{j+1}, W_{j+1}) in S and merge them in a temporal graph (I', W') (notice that there is no episode defined in an interval between I_j and I_{j+1})
- It applies the dynamic programming algorithm described in Sect. 3 for k-Densest-Alignment-Episodes to an uncovered interval in $\mathcal{T}$; let (I_c, W_c) be the subgraph computed
- If it holds that $dens(I_c, W_c)+dens(I', W') > dens(I_j, W_j) + dens(I_{j+1}, W_{j+1})$, then it replaces (I_j, W_j) and (I_{j+1}, W_{j+1}) with (I_c, W_c) and (I', W').

4 Experimental Analysis

In this section, we describe the synthetic datasets that we have used in the experimental analysis.

Datasets The synthetic temporal graphs consist of k planted communities (which are cliques) and a background graph. The k planted communities are defined in non-overlapping intervals and are defined on disjoint sets of vertices. Moreover, the weight of each edge of the planted communities is defined equal to 10. The background graph includes all vertices from planted communities over the discrete time domain $\mathcal{T}$ and it is generated based on Erdős-Rényi model with parameter $p = 1/|V|$, $p = 3/|V|$, and $p = 5/|V|$. The edges of the background graph are uniformly distributed on the timestamps of the time domain. The weight of each edge of the background graph is randomly generated in interval [0, 4].

Three groups of synthetic networks called *Synthetic-small*, *Synthetic1* and *Synthetic2* are generated. For each group, we vary the time domain, the number of communities and the number of vertices/edges of background graph and communities.

The *Synthetic-small* dataset is generated for comparison with optimal solution. In each graph of *Synthetic-small*, the background graph contains 70 vertices over a time domain of 70 timestamps with k equal to 4; each community has 12 vertices. In each graph of *Synthetic1*, the background graph contains 1000 vertices over a time domain of 1000 timestamps with k equal to 20; each community has 25 vertices. Finally, in each graph of *Synthetic2*, the background graph contains 10,000 vertices over a time domain of 10,000 timestamp, while k is equal to 40; each community has 50 vertices.

Outcome We report density, running time, quality of the solution intervals and communities, averaged over 100 examples for each value of p. In particular, we calculate

Table 1 Density, running time and quality of the interval and subgraphs solutions (F-measure) on *Synthetic-small* dataset with parameter $p = 1/|V|$, $p = 3/|V|$, and $p = 5/|V|$ for two phases (L-DP and LocExt) of the heuristic and DP. Running time is in seconds and the results are averaged over 100 examples for each value of p

			Intervals	Subgraphs
$p = 1/\|V\|$	Time	Density	F-measure	F-measure
DP	304.45	127.08	1	1
L-DP	0.37	89.90	0.52	0.84
LocExt	0.06	94.98	0.65	0.95
$p = 3/\|V\|$	Time	Density	F-measure	F-measure
DP	339.90	127.12	1	1
L-DP	0.44	89.96	0.50	0.80
LocExt	0.09	96.69	0.66	0.94
$p = 5/\|V\|$	Time	Density	F-measure	F-measure
DP	392.76	127.18	1	1
L-DP	0.49	90.05	0.52	0.83
LocExt	0.07	95.13	0.64	0.93

the F-measure[2] to evaluate the accuracy of the heuristic to find the planted communities and intervals. In the experimental analysis we report the results of the two phases of the heuristic, L-DP for k-ℓ-Densest-Alignment-Episodes and LocExt. Table 1 illustrates solutions and running time of DP as the optimal dynamic programming algorithm for k-Densest-Alignment-Episodes and our heuristic.

Due to the time complexity of DP, we consider the comparison only on the *Synthetic-small* dataset. The results in Table 1 show that the heuristic is able to compute solutions of density approximately 75% of the optimal density for all value p. For the interval quality (that is the timestamps in the planted intervals that are correctly identified by our heuristic), the average F-measure is between 64 and 66%. For the subgraphs quality (that is the nodes in the planted community that are correctly identified by our heuristic), the average F-measure is between 93 and 95%.

On the *Synthetic-small* dataset LocExt is able to improve the detected solution of the first phase of the heuristic (L-DP) in a reasonable time. In the worst case (for $p = 3/|V|$), LocExt needs 20% of the L-DP running time. LocExt improves the density of L-DP at least 5.6% (for $p = 5/|V|$), and at most 7.5% (for $p = 3/|V|$). For interval and subgraph quality, LocExt improves F-measure respect to the solution of L-DP. More precisely, the F-measure of the interval quality is improved at least 23% (for $p = 5/|V|$), and at most 32% (for $p = 3/|V|$); the F-measure of subgraph quality is improved at least 12% (for $p = 5/|V|$), and at most 17.5% (for $p = 3/|V|$). As for the running time on the *Synthetic-small*, the heuristic in the worst case is 641 times (for $p = 3/|V|$) and in the best case 708 times (for $p = 1/|V|$) faster than DP.

[2] F-measure is the fraction of recall and precision.

Table 2 Density, running time and quality of the interval and subgraphs solutions (F-measure) on *Synthetic1* dataset with parameter $p = 1/|V|$, $p = 3/|V|$, and $p = 5/|V|$ for two phases (L-DP and LocExt) of the heuristic. Running time is in seconds and the results are averaged over 100 examples for each value of p

			Intervals	Subgraphs
$p = 1/\|V\|$	Time	Density	F-measure	F-measure
L-DP	24.97	607.17	0.36	0.66
LocExt	10.35	685.4	0.54	0.77
$p = 3/\|V\|$	Time	Density	F-measure	F-measure
L-DP	24.99	607.17	0.38	0.70
LocExt	9.38	689.51	0.57	0.82
$p = 5/\|V\|$	Time	Density	F-measure	F-measure
L-DP	32.89	607.18	0.41	0.75
LocExt	13.70	709.89	0.63	0.89

Table 3 Density, running time and quality of the interval and subgraphs solutions (F-measure) on *Synthetic2* dataset with parameter $p = 1/|V|$, $p = 3/|V|$, and $p = 5/|V|$ for two phases (L-DP and LocExt) of the heuristic. Running time is in seconds and the results are averaged over 100 examples for each value of p

			Intervals	Subgraphs
$p = 1/\|V\|$	Time	Density	F-measure	F-measure
L-DP	738.07	1413.38	0.18	0.45
LocExt	491.28	1772.62	0.41	0.57
$p = 3/\|V\|$	Time	Density	F-measure	F-measure
L-DP	843.30	1413.38	0.18	0.47
LocExt	521.10	1755.78	0.40	0.57
$p = 5/\|V\|$	Time	Density	F-measure	F-measure
L-DP	960.78	1413.38	0.21	0.54
LocExt	382.40	1733.23	0.41	0.64

Table 2 reports the results of the heuristic on *Synthetic1* dataset. On this dataset LocExt improves density of detected solution of the L-DP at least 12.9% (for $p = 1/|V|$) and at most 16.9% (for $p = 5/|V|$). For interval and subgraph quality, LocExt is able to improve the quality of the solution returns by L-DP for k-ℓ-Densest-Alignment-Episodes by increasing the F-measure of interval solution at least 50% (for both $p = 1/|V|$ and $p = 3/|V|$) and at most 54% (for $p = 5/|V|$), and F-measure of subgraph solution at least 16.7% (for $p = 1/|V|$) and at most 18.7% (for $p = 5/|V|$). As for the running time on the *Synthetic1*, the LocExt needs in the worst case 42% (for $p = 5/|V|$) and in the best case 37% (for $p = 3/|V|$) of the running time of L-DP for k-ℓ-Densest-Alignment-Episodes.

Finally, Table 3 shows the results of the heuristic on *Synthetic2*. On this larger size dataset, LocExt improves density of the solutions computed by the L-DP for k-ℓ-Densest-Alignment-Episodes at least 22.6% (for $p = 5/|V|$) and at most 25.4% (for $p = 1/|V|$). For interval and subgraph quality, LocExt improves the F-measure of interval solution at least 95% (for $p = 5/|V|$) and at most 128% (for $p = 1/|V|$), and F-measure of subgraph solution at least 18% (for $p = 5/|V|$) and at most 27% (for $p = 1/|V|$). Table 3 shows the running time of the heuristic on the *Synthetic2*. The LocExt improves the solution with running time in the worst case 67% (for $p = 1/|V|$) and in the best case 40% (for $p = 5/|V|$) of the L-DP for k-ℓ-Densest-Alignment-Episodes.

5 Conclusion

We have introduced a new network model, called temporal dual networks, that integrates the temporal network and dual network frameworks. We have presented a heuristic for the problem and an experimental evaluation on synthetic temporal graphs. Future works include the application of the temporal dual network model to real datasets.

References

1. Braha, D., Bar-Yam, Y.: Time-dependent complex networks: dynamic centrality, dynamic motifs, and cycles of social interactions. In: Adaptive Networks, pp. 39–50. Springer (2009)
2. Castelli, M., Dondi, R., Hosseinzadeh, M.M.: Genetic algorithms for finding episodes in temporal networks. In: Cristani, M., Toro, C., Zanni-Merk, C., Howlett, R.J., Jain, L.C. (eds.), Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 24th International Conference KES-2020, Virtual Event, 16–18 September 2020. Procedia Computer Science, vol. 176, pp. 215–224. Elsevier (2020)
3. Charikar, M.: Greedy approximation algorithms for finding dense components in a graph. In: Approximation Algorithms for Combinatorial Optimization, Third International Workshop, APPROX 2000, Proceedings, pp. 84–95 (2000)
4. Chen, J., Saad, Y.: Dense subgraph extraction with application to community detection. IEEE Trans. Knowl. Data Eng. **24**(7), 1216–1230 (2010)
5. Dondi, R., Hosseinzadeh, M.M.: Dense sub-networks discovery in temporal networks. SN Comput. Sci. **2**(3), 1–11 (2021)
6. Dondi, R., Hosseinzadeh, M.M., Guzzi, P.H.: A novel algorithm for finding top-k weighted overlapping densest connected subgraphs in dual networks. Appl. Netw. Sci. **6**(1), 40 (2021)
7. Dondi, R., Hosseinzadeh, M.M., Mauri, G., Zoppis, I.: Top-k overlapping densest subgraphs: approximation algorithms and computational complexity. J. Comb. Optim. **41**(1), 80–104 (2021)
8. Galbrun, E., Gionis, A., Tatti, N.: Top-k overlapping densest subgraphs. Data Min. Knowl. Discov. **30**(5), 1134–1165 (2016)
9. Goldberg, A.V.: Finding a Maximum Density Subgraph. Tech. rep, Berkeley, CA, USA (1984)
10. Gu, S., Jiang, M., Guzzi, P.H., Milenković, T.: Modeling multi-scale data via a network of networks. Bioinformatics **38**(9), 2544–2553 (2022)

11. Guzzi, P.H., Milenković, T.: Survey of local and global biological network alignment: the need to reconcile the two sides of the same coin. Briefings Bioinform. **19**(3), 472–481 (2018)
12. Hosseinzadeh, M.M.: Dense subgraphs in biological networks. In: International Conference on Current Trends in Theory and Practice of Informatics, pp. 711–719. Springer (2020)
13. Kawase, Y., Miyauchi, A.: The densest subgraph problem with a convex/concave size function. Algorithmica **80**(12), 3461–3480 (2018)
14. Kempe, D., Kleinberg, J., Kumar, A.: Connectivity and inference problems for temporal networks. J. Comput. Syst. Sci. **64**(4), 820–842 (2002)
15. Milano, M., Milenković, T., Cannataro, M., Guzzi, P.H.: L-hetnetaligner: a novel algorithm for local alignment of heterogeneous biological networks. Sci. Rep. **10**(1), 1–20 (2020)
16. Rozenshtein, P., Gionis, A.: Mining temporal networks. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 3225–3226. ACM (2019)
17. Wu, Y., Zhu, X., Li, L., Fan, W., Jin, R., Zhang, X.: Mining Dual Networks—Models, Algorithms, and Applications. TKDD (2016)

Exploring and Mining Attributed Sequences of Interactions

Tiphaine Viard, Henry Soldano, and Guillaume Santini

Abstract We consider entities interacting over time: individuals meeting, customers buying products, etc., each entity being labeled with some information that may depend on time, and possibly extracted from the interaction nature. Capturing the dynamics as well as the structure of these interactions is of crucial importance for analysis. We are interested here in mining sequences of such interactions. For that purpose, we define core closed patterns in this context and introduce algorithms to enumerate them on a labeled stream graph. We run experiments on two real-world datasets, one representing interactions among students and the other representing citations between authors.

1 Introduction

We consider mining connected data with the following view: part of the data consists in attributes values reporting information about objects, while the remaining part of the data reports information about how objects are related. We search then for attribute patterns i.e. sentences expressing constraints on the attributes' values. Various previous work on graphs (see Sect. 2.1) confront the patterns to the connected structure, i.e. consider poorly connected objects as poorly relevant to the knowledge to extract. Then, the mining process enumerates and selects both attribute patterns and the dense subgraphs associated with them. The purpose of this article is to extend

T. Viard (✉)
i3, LTCI, Télécom Paris, Institut Polytechnique de Paris, Palaiseau, France
e-mail: tiphaine.viard@telecom-paris.fr

T. Viard · H. Soldano · G. Santini
LIPN, Université Sorbonne Paris Nord, Villetaneuse, France

H. Soldano
NukkAI, Paris, France

ISYEB, Muséum National d'Histoire Naturelle de Paris, Paris, France

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_42

one such methodology, namely the *core closed pattern* methodology, in order to mine temporal interaction data.

Modelling data with a structural component over time has been done in multiple ways, and in particular recently, by considering interaction data: the connected data is then a sequence of triplets (t, u, v) indicating that nodes u and v interacted at time t (see Sect. 2.2). They may represent, for instance, the interactions between scientists attending a conference, interactions between students, interactions on the web, among others. Enriching such connection data with attributes describing individuals allows to extract knowledge related to them and the way these individuals are connected in time. The individuals' descriptions may themselves be time-dependent.

The present work focus on extending core closed pattern methodology to attributed stream graphs, a process which is facilitated by the fact that the notion of graph cores, which core closed pattern mining heavily relies on, has a natural counterpart in stream graphs. We develop our contributions as follows: after discussing related work in Sect. 2, we present the core closed pattern formalism to mine connected data in Sect. 3. In Sect. 4, we present the stream graph formalism to model interactions over time, and show how to adapt the mining methodology to stream graphs. We then present algorithms, in Sect. 5, and apply them to closed pattern mining on two real-world datasets, in Sect. 6. Finally, we conclude and present some tracks for future work in Sect. 7.

2 Related Work

2.1 FCA and Closed Pattern Mining on Graphs

A recent review on mining and finding dense subgroups within attributed graphs [1] discusses a variety of approaches, algorithms and programs addressing this task. Among them, various works such as [18] or [19] define the subgraph properties that are suitable both from formal and application standpoints. The latter introduced *core closed pattern mining* whose definitions and results necessary for our purpose are presented in Sect. 3.

Closed pattern mining (CPM) is strongly related to Formal concept analysis (FCA) [24] and search for concepts ordered according to the general-to-specific ordering. Each concept is made of a *support set* together with the corresponding *closed pattern*, i.e. the most specific pattern that occurs in this support set. CPM focuses on the efficient enumeration of closed patterns in large datasets [25].

Core closed pattern mining is a variant in which the *support set* of a pattern is reduced to its *core support set* i.e. the *core* of the subgraph induced by the original support set. For instance, the k-core of a graph proposed by Seidman [17] induces the unique maximal subgraph whose nodes all have degree at least k. Core definitions, as generalized in [2], always rely, as k-cores, on some topological property. In [19] it is shown that when reducing support sets to core support sets we may still define closed patterns (see Sect. 3). Adapting efficient enumeration algorithms from closed pattern

mining [14] has allowed to apply the methodology to bipartite [21] and directed [22] networks.

2.2 Stream Graphs and Modelling of Interactions Over Time

Modelling data that has a structural component over time has been done in multiple ways, typically through different variants of dynamic graphs. In this setting, one typically has a sequence of graphs $\{G_i\}$ and a time frame Δ, and for all i, E_i contains all the interactions that happened between times $i\Delta$ and $(i+1)\Delta$. There are multiple variants, for example in which the graph only grows in time [7], or in which multiple concurrent values of Δ are considered [12]. The main drawback of these approaches is the loss of temporal information induced by this aggregation [5].

Recently, a few models take a different perspective, where aggregating is not necessary. The sequences of interactions are then modelled as temporal networks [9], time-varying graphs [6] or stream graphs [11], depending on the research goals and the scientific community. All these approaches consider a sequence of (t, u, v) indicating that nodes u and v interacted at time t. Temporal networks has large bodies of work around diffusion and temporal causality [8] and temporal motif mining [3]; time-varying graphs focuses on reachability and elaborating algorithmic complexity classes [4] while stream graphs focus on extending the notions used for large-graph analysis [11] and applying them to real-world scenarios.

3 Closed Pattern Mining

In this section we report definitions and results required to introduce our attributed stream graph mining methodology. Except regarding the third item of Proposition 1 and considering some rewriting, they are extracted from [21]. To be self-contained, let us first recall closure and interior operator definitions: Let S be an ordered set and $f : S \to S$ a self map such that for any $x, y \in S$, f is *monotone*, i.e. $x \leq y$ implies $f(x) \leq f(y)$ and *idempotent*, i.e. $f(f(x)) = f(x)$. Then if $f(x) \geq x$, f is called a *closure operator* while if $f(x) \leq x$, i.e. f is *intensive*, f is called an *interior operator*.

In closed pattern mining, a pattern q belongs to a lattice L with partial order $\geq$ where $q \geq q'$ means that q is more specific than q', i.e. whenever pattern q occurs in some object o, pattern q' also occurs in o. Consider then a set of objects V, each object v has a description $d(v)$ in L representing the most specific pattern in which it occurs. Pattern q *support set*, $\text{ext}(q)$ is then the set of its occurrences in V. Applying an interior operator p to $\text{ext}(q)$ results in reducing the support set of q into its so-called *core support set* $p \circ \text{ext}(q)$. The most specific pattern with core support set $p \circ \text{ext}(q)$ is then unique, is called a *core closed pattern* and denoted by $f(q)$.

Computing the core closed pattern $f(q)$ relies on an intersection operator *int* such that $int(X)$ returns the most specific pattern occurring in all objects in X. We

obtain then the core closed pattern $f(q)$ as $f(q) = \text{int} \circ p \circ \text{ext}(q)$ and f is a closure operator. In the closed itemset mining setting objects are described as itemsets i.e. subsets of a set of items I and the intersection operator simply is the set theoretic intersection operator $\cap$ as exemplified below:

Example 1 Let us consider L as the powerset 2^I where $I = abcd$, together with the object set $V = 123$, with descriptions $d(1) = abd$, $d(2) = acd$, $d(3) = abc$. The void pattern $\emptyset$ has then support set 123. Consider then the interior operator p such that $\forall X \subseteq V$, $p(X) = X \setminus 3$. We have $p(123) = 12$ and the core closed pattern $f(\emptyset)$ is then $\text{int}(12) = abd \cap acd = ad$.

3.1 Core Closed Pattern Mining

The *core* of a network is the largest vertex set of a graph that induce a (core) subgraph whose vertices all satisfy some topological property P. This idea may be extended to particular cores, called *multi-cores* which are vertex subset tuples. To obtain a core definition, and therefore associated interior operators P has to fulfill some conditions. We summarize results leading to corresponding core closed patterns in what follows.

Let V be a set, $\mathbf{X} = (X_1, \ldots, X_k)$ be a subset tuple, and v be an element of some X_i, which we simply rewrite as $v \in \mathbf{X}$. We show that P monotonicity results in defining interior operator on 2^V.

Proposition 1 *Whenever P is monotone, i.e. . for any tuple X and any $v \in \mathbf{X}$, $P(v, \mathbf{X})$ and $\mathbf{X}' \supseteq \mathbf{X}$ implies $P(v, \mathbf{X}')$, then:*

1. *there is a greatest subset tuple $\mathbf{C} \subseteq X$ where $P(v, \mathbf{C})$ holds for all $v \in \mathbf{C}$*
2. *p_b defined as $p_b(\mathbf{X}) = \mathbf{C}$ is a an interior operator*
3. *p defined on 2^V as $p(X) = p_b(X, \ldots, X)$ is an interior operator on 2^V.*

Proof Items 1 and 2 have been established, in a slightly different form, in [21]. Regarding item 3 we need to prove three properties. The proofs straightforwardly follow from the truth of the corresponding properties of the interior operator p_b. For instance to prove that p is monotone, i.e. $X \subseteq X'$ implies $p(X) \subseteq p(X')$, we remark that $X \subseteq X'$ means $(X, X) \subseteq (X', X')$. As p_b is an interior operator this implies $p_b(X, X) \subseteq p_b(X', X')$ and it follows that $p(X) \subseteq p(X')$. Idempotency and intensivity are proved in the very same way.

When considering core definitions on a graph, as the k-core definition, the tuple is a singleton, and we have $p(X)$ as the largest vertex subset that induces a subgraph in which all vertices have degree at least k. Regarding multi-cores, let G be a directed graph, the $h - a$ BHA bi-core property states that in the subgraph $G(X_1, X_2)$ induced by the directed edges from X_1 towards X_2, if v is in X_1 it has outdegree at least h and if v is in X_2 it has indegree at least a. The corresponding h-a hub-authority (HA) core $p(X)$ was first defined in [22].[1]

[1] *Hub* and *authority* terminology refers to the notions introduced by Kleinberg [10].

3.2 Exhibiting Patterns of Interest

When selecting individual patterns from a pattern set Q, according to various interestingness criteria, the resulting pattern subset contain patterns very similar to other patterns. There are various *pattern set selection* ways of reducing size and redundancy of a pattern set [15, 20]. In our experiments we will use the $g\beta$ *pattern set selection* algorithm first defined and applied to core closed patterns in [20]. It consists in maximizing in the selected pattern set Q_β the sum of the values of a pattern interestingness measure g under the constraint that two patterns q and q' in Q_β have to be at distance $\sigma(q, q')$ at least β. The interestingness measure g as well as the distance measure σ, are $g\beta$ parameters.

4 Stream Graphs

Stream graphs [11] model interactions over time by generalizing many useful notions from complex and social networks analysis. We denote a *stream graph* by the tuple $S = (T, V, W, E)$, where T is a time interval, V a set of nodes. $W \subseteq T \times V$ denotes the presence times of nodes, such that $(t, v) \in W$ means that node v is "active" at time t, and finally, $E \subseteq T \times V \otimes V$ denotes interactions, such that $(t, uv) \in E$ means that nodes u and v interacted at time t. When useful, we write $u \times \Delta$ and $uv \times \Delta$ to state that for all t in Δ, node u is active and u and v interact. We consider streams either as undirected, where node pairs are unordered and without loop, or as directed. Bipartite streams are made of directed interactions from a vertex set $\top$ to a vertex set $\bot$. Figure 1a, b depict toy stream graphs.

We say that $S' = (T', V', W', E')$ is a substream of S if and only if $T' \subseteq T$, $V' \subseteq V$, $W' \subseteq W$ and $E' \subseteq E$. We denote this by $S' \subseteq S$. We denote by $S(W')$ the substream graph induced by a time-node vertex subset $W' \subseteq W$, and whose interaction subset $E_{W'}$ contains interaction between time-nodes of W'.

Finally, let us define $G_S = (V_S, E_S)$ the graph induced by S, with $V_S = \{u : \exists (t, uv) \in E, t \in T, v \in V\}$ and $E_S = \{uv : \exists (t, uv) \in E, t \in T\}$. In other words, nodes and edges belong to V_S and E_S if and only if there exist some time t such that (t, uv) belongs to E. The adaptation to the directed case is straightforward.

For any node $v \in V$, we denote its *neighbourhood at time t* by $\mathcal{N}_t(v) = \{(t, u) : \exists (t, uv) \in E, u \in V\}$ the set of (t, u) that interact with node v at time t. We further denote the *degree* of v at time t by $d_t(v) = |\mathcal{N}_t(v)|$. For example, in Fig. 1(left), node b at time 2 interacts with nodes a and d, and so $\mathcal{N}_2(b) = \{a, d\}$, and $d_2(b) = 2$.

When the stream graph is directed the *outneighbourhood at time t* of node v, $\mathcal{N}_t^+(v)$, contains time-nodes such that there exists a directed edge (t, uv) in E and its *outdegree at time t* $d_t^+(v)$ is the size of its outneighbourhood.The *inneighbourhood at time t* and *indegree at time t* of a node are defined in the same way. We also denote

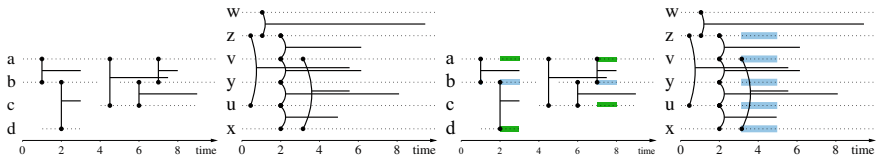


Fig. 1 Two stream graphs and their cores. From left to right: **a** Undirected stream graph S_1 with time-nodes $W = \{a \times [0, 10], b \times [0, 4] \cup [5, 10], \dots\}$ and interactions $E = \{ab \times [1, 3] \cup [7, 8], bd \times [2, 3], \dots\}$. **b** Bipartite stream graph S_2 displaying interactions between nodes in $\top = \{u, v, w\}$ and nodes in $\bot = \{x, y, z\}$. **c** The 2-star-satellite core of S_1, with stars $b \times [2, 3] \cup [7, 8]$ (in blue), and satellites $\{a \times [1, 3] \cup [7, 8], c \times [7, 8], d \times [2, 3]\}$ (in green). **d** The 2, 2-BHA-core of S_2 (in blue)

by $S(W_1, W_2)$ the substream graph of a directed stream graph S induced by two time-node subsets W_1 and W_2 of W and whose interaction subset E_{W_1,W_2} is made of the interactions in E from W_1 to W_2.

5 Pattern Enumeration in Stream Graphs

5.1 Core Operators and Calculation

We introduce two core operators that will be used in our experiments. The k-Star-Satellite core operator selects in an induced substream graph $S(W')$ high degree time-nodes together with their neighbours (see Fig. 1c):

Definition 1 (k-Star-Satellite) Let S be an undirected stream graph and $k \in \mathbb{N}$, the k-star-satellite property $\mathrm{P}((t, v), W')$ holds iff in the induced substream graph $S(W')$ either $d_t(v) \geq k$ or there exists $(t, v') \in \mathcal{N}_t(v)$ such that $d_t(v') \geq k$.

The h-a HA core operator (see Fig. 1d) is a counterpart in directed stream graphs of the h-a HA core operator in directed graphs discussed in Sect. 3.1.

Definition 2 (h-a BHA) Let S be a directed stream graph and $h, a \in \mathbb{N}$, the h-a BHA property $P_b((t, v), W_1, W_2)$ holds iff in the induced substream graph $S(W_1, W_2)$, if (v, t) is in W_1 then $d_t^+(v) \geq h$ and if (v, t) is in W_2 then $d_t^-(v) \geq a$.

The h-a HA core of $G(X)$ is then obtained as $p(X) = H \cup A$ where $p_b(X, X) = (H, A)$ is the h-a BHA bi-core of the induced substream graph $G(X, X)$.

For both definitions, we need to prove that the associate properties are monotone properties (see Sect. 3.1).

Theorem 1 *Definitions 1 and 2 are core properties.*

Proof Let us start with the k-Star-Satellite Property 1. We are interesting in proving that this property is monotonous. Suppose that there exists a substream $S' = (T', V', W', E')$, $S' \subseteq S$ such that for all elements $(t, v) \in W'$, Property 1 holds. In other words, there are enough interactions in E' such that node v at time t either has at least k neighbours (and is a star), or is a neighbour of such a node (and is a satellite).

Let us show that there is no stream $R = (T_R, V_R, W_R, E_R)$, $R \supset S'$ such that the property is false. Suppose that such a stream R exists. Then, there exists elements of W' that are not in W_R. Since the core properties defined both involves degrees, this can only mean that there are interactions in E' that are not in E_R, which in turns means that $R \not\supset S'$. This proves monotonicity of the k-Star-Satellite property. An identical argument holds for Definition 2. □

Generic algorithms to compute cores are detailed in [21]. Such an algorithm considers an object subset X and starts a first pass in which it remove all objects from X that do not satisfy the core property $P(x, X)$, resulting in a new X'. A new pass is then started removing objects that do not satisfy $P(x, X')$, and the process is repeated until a fixed point $C = p(X)$ is reached. For some properties, such as the k-star-satellite property, a single pass reaches the fixed point.

In stream graphs, time is modelled as continuous, and testing for all (t, v) a core property would both (i) require some sort of discretization, (ii) result in redundant computations. Note that the property is usually valid for all instants t on a number of intervals of time. For instance, in Fig. 1, (t, b) is a 2-star for all $t \in [1, 2.5]$. We obtain better algorithms by directly attempting to find the maximal such intervals. They use a data structure to represent the stream graph as a temporal adjacency table $\mathcal{D}_S$: for each node $u \in V$, we store a list $\mathcal{D}_S(u)$ of triplets (t, v, e), sorted in increasing time order, indicating that node u started or stopped interacting with node v at time t. The algorithm computing the k-star-satellite core is detailed in technical note [23].

5.2 Pattern Enumeration and Pattern Set Selection

Let us now discuss the pattern enumeration of all frequent core closed patterns, i.e. with core support set at least s when patterns are itemsets. The algorithm starts with the closure q_0 of the empty pattern $\emptyset$ and associated core support set X. Then, for all the items x we build the pattern $q_0 \cup \{x\}$ and compute its core support set in the stream, the associated core closed pattern q_x and recursively all frequent core closed patterns greater than q_x. Maintaining a list EL of prohibited items results in each pattern being only enumerated once. The algorithm is similar to the one defined by [19] to mine attributed graphs.

Algorithm 1 Pattern enumeration algorithm on time-node set W

```
1: X ← p(W);
2: q_0 ← int(X)
3: EL ← [];
4: ENUM(q_0, X, EL)
5: function ENUM(q, X, EL)
6:     print(q, X)
7:     for x ∈ I \ q do
8:         q_x ← q ∪ {x}
9:         X_x ← p(ext(q_x) ∩ X)
10:        if |X_x| ≥ s then
11:            q_x ← int(X_x)
12:            if q_x ∩ EL = ∅ then
13:                ENUM(q_x, X_x, EL)
14:                EL ← EL ∪ {x}
15:            end if
16:        end if
17:    end for
18: end function
```

The time complexity of Algorithm 1 runs in $\mathcal{O}(2^m \cdot (l^2 m + l^2 + lm + l\gamma))$ time, where $m = |E|$ is the number of interactions, $l = |I|$ the size of the attributes language, and γ the complexity of computing the core property. We provide a detailed breakdown in a technical report [23]. This upper bound is rarely reached in practice : the choice of an adequate core property ensures that there are far less than 2^m patterns.

Notice that there is a correspondence between our patterns and the ones defined for (static) graphs in [19]. Indeed, saying that pattern q has support set X within W is equivalent to saying that for any t, q has support set $X_t = \{v \in V \mid (t, v) \in X\}$ within V. In the experimental section we will consider core closed patterns in the stream graph and their static counterpart.

6 Experiments

We performed our experiments using two data sets, one of individual contacts between high school students (HS-327), and another of article citations extracted from the Association of Computer Linguistics Anthology website. Both datasets are publicly available, and all the code for the following experiments is available online.[2] We did not apply any minimum support constraint on our experiments, though core definitions do imply such constraints (for instance á 4-star satellite core is of size at least 5). The pattern distance in the $g\beta$ selection process (see Sect. 3) for both experiments is as follows: let l_i, l_j be two patterns and let W_i, W_j be their core support sets, the Jaccard distance $\sigma(l_i, l_j)$ is defined as $\sigma(l_i, l_j) = 1 - \mathcal{J}(l_i, l_j)$ where

[2] https://github.com/TiphaineV/pattern-mining.

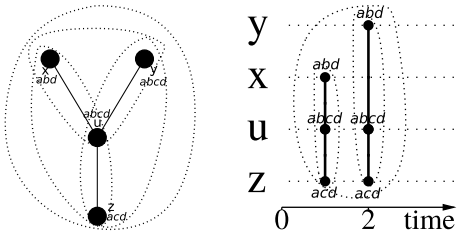


Fig. 2 2-star-satellites on a stream and its induced graph. There are 4 closed patterns on the graph, but 3 in the stream, as closed pattern $(ab, \{u, x, y\})$ cannot exist in time, since u never interacts with x and y at the same time

$\mathcal{J}(l_i, l_j) = \frac{|W_i \cap W_j|}{|W_i \cup W_j|}$. $\sigma(l_i, l_j)$ is equal to 0 whenever $W_i = W_j$ and to 1 if W_i and W_j have no element in common. As a g interestingness measure we consider the core support set size.

Contacts between individuals. HS-327 is a dataset constructed from the results of a study of social interactions of 327 French students conducted in 2013 [13]. The initial dataset provides us with the stream of contacts over 5 days between the students, which amounts to 33,806 temporal interactions. The dataset also contains, for each student u, their class, their gender, and three lists of friends: one is the students u has met (self-report), another is the students that u has declared as friends (self-report), and finally, the friends u has on Facebook. We express each temporal interaction between a pair of nodes as a union of consecutive intervals of the form $[t_{i,j} - 20\text{sec}, t_{i,j}]$.

Academic paper citing. The ACL dataset is extracted from the Association of Computer Linguistics (ACL) anthology, which regroups research papers in Computational Linguistics. We have built a directed temporal citation network in which interaction $ab, [t, t-1]$ means "author a cites author b in some paper published in year t". The dataset relies on articles published between 1979 and 2008, i.e. over a span of 29 years. The dataset contains 250,000 interactions between about 8000 authors.

Each author is described as the set of keywords extracted from the abstracts' content of the all articles they authored during the 1979–2008 period, and therefore authors descriptions do not depend on time. These descriptions are obtained using the CSO ontology, as described in [16, 26]. This leads to a total of 2500 attributes.

6.1 Results

HS-327. We use the k-star-satellite property, and present in Table 1 some numerical results depending on the value of k and the selection parameter β. Notice that rapidly (when $k \geq 5$), there are no more patterns to enumerate other than the empty pattern. This is due to the temporal nature of the data, that spreads out interactions as compared to a static graph.

In the selected patterns, we capture generic patterns, that spread in time, as well as more specific patterns related to particular time intervals. This allows us to study the interactions at multiple time scales.

Table 1 Number of enumerated core closed patterns ($\beta = 0.0$) and corresponding enumeration runtimes in two datasets. Number of $g\beta$-selected core closed patterns is reported for β values ranging from 0.2 to 0.4

Dataset		β					Runtime mns
	k	0.0	0.2	0.4	0.6	0.8	
HS-327	3	620	362	221	125	76	16
HS-327	4	99	75	52	40	31	9
ACL	15, 15	1664	406	175	56	12	90

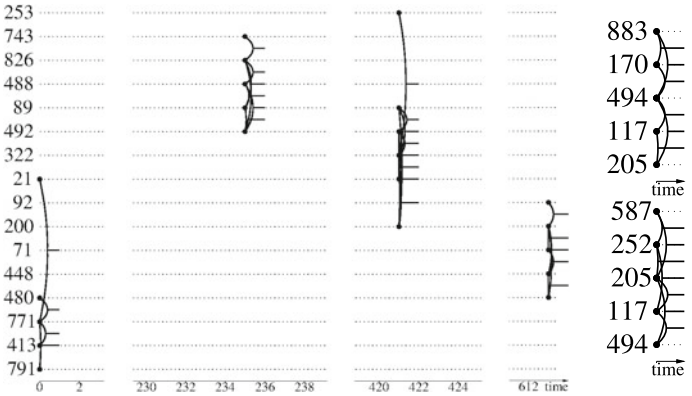


Fig. 3 Cores of some closed patterns on the **HS-327** dataset, selected with $\beta = 0.8$. *Left*: one general pattern spread in time (blanks represent time periods with no interactions) and displaying a large number of students. *Right*: two more specific patterns, involving less students over a shorter time span

We noticed that many patterns contain the gender of the students (either G_M or G_F), reinforcing claims that students tends to regroup in non-mixed gender groups. When considering 4-star satellites, Fig. 3left displays pattern G_M, C_PC, involving 16 male students of the PC class (Physics and Chemistry specialty) whose interactions spread over time. At the contrary, Fig. 3right-top displays a rather specific pattern involving both male and female students. This pattern[3] represent both Facebook friendships (F_xx) and self-declared friendships (D_xx) of students belonging to the same 2BIO3 class (Biology specialty).

Let us compare patterns resulting from mining the stream graph to those obtained from the associated static graph, enumerated by implementing the code from [20]. In the static graph uv is an edge whenever there is some time t such that (t, uv) belongs to the stream graph. Note that when using k-star-satellite cores to mine both graphs, the core closed patterns in the stream graph also are core closed patterns in the static graph. Indeed, if node u has k neighbours at a time $t \in T$, then u has also

[3] namely D_894, F_265, D_205, F_170, F_425, F_871, F_1, D_1, D_883, F_883, F_205,C_2BIO3, F_272, F_106.

natural languages, semantics, syntactics, syntactic structure
linguistics, machine translations, syntactics, syntactic structure
bilingual, correlation analysis, machine translations, translation process
correlation analysis, learning, parsing algorithm, syntactic analysis, syntactics, syntactic structure
correlation analysis, machine translations, statistical machine translation, syntactic structure
correlation analysis, machine translations, phrase-based statistical machine translation, statistical machine translation, translation models

Fig. 4 The 6 closed patterns selected with $\beta > 0.9$ in the ACL dataset

k neighbours in the static graph while it is possible for u to have k neighbours in the static graph, each related to u at different times (see the small example in Fig. 2).

When mining the static graph, we obtain 11,600 4-star-satellite closed patterns, to be compared to the 99 closed patterns obtained from the stream graph. Notice that when considering the time of interaction many of the static patterns do not have any real grounding. We argue then that stream patterns are fewer but of higher relevance.

ACL. We discuss now the 15-15 BHA core closed patterns enumeration and selection on the ACL dataset. In total, 1664 closed patterns were enumerated in less than 90 min. Before pattern set selection we retain only patterns made of at least 4 keywords. The patterns help us highlight different subfields of the ACL Anthology. For instance, closed patterns with interactions around year 1990 involves often keywords `syntactics` or `context-free`, while `learning, natural_language_processing` appear mostly in interactions after 2005. Also, few (16) researchers are active over more than 14 years and it is interesting to look at the closed patterns in which core support sets they appear over time. For most of these researchers, the terms `parsing` and `natural language processing` appear in closed patterns occurring around 2003, even though author `Lynette Hirschman` has `natural language understanding` in closed patterns in which she appears since 1991.

Focusing on the 6 patterns selected with $\beta \geq 0.9$ gives other insights. These patterns, by definition mutually highly dissimilar, are displayed in Fig. 4. They highlight different sub-areas of research of the Association for Computer Linguistics. The fact that keywords occur even in various 0.9-selected closed patterns likely comes from the fact that the scope of the ACL itself regroups researchers on similar topics of research, possibly at different times. As such, even the most dissimilar patterns, regarding interaction time or involved authors retain some conceptual similarity. As an example, `correlation analysis, machine translations` appear in three patterns (lines 3, 5, 6) among the 6 selected patterns.

7 Conclusion

In this article, we strengthen the existing bridges between closed pattern mining and real-world linked data. We show that beyond graphs, these methods can be adapted to streams of interactions, in order to mine relevant patterns from large real-world streams whose time-nodes are labelled by attribute values.

We define cores for undirected and directed stream graphs and show that they satisfy the necessary conditions for closed pattern enumeration. We also discuss how efficiently integrating the temporal dimension by considering interactions on time intervals, alleviates the algorithmic challenges. Our experiments exemplify the kind of information that can be mined on such attributed stream graphs, to be compared to information extracted from static graphs.

Beyond that, we are interested in extending these methods on other static and dynamic linked structures, as hypergraphs or knowledge graphs.

References

1. Atzmueller, M., Günnemann, S., Zimmermann, A.: Mining communities and their descriptions on attributed graphs: a survey. Data Mining Knowl. Disc (2021)
2. Batagelj, V., Zaversnik, M.: Fast algorithms for determining (generalized) core groups in social networks. Adv. Data Anal. Classification (2011)
3. Braha, D., Bar-Yam, Y.: Time-dependent complex networks: dynamic centrality, dynamic motifs, and cycles of social interactions. In: Adaptive Networks, pp. 39–50. Springer (2009)
4. Braud-Santoni, N., Dubois, S., Kaaouachi, M.H., Petit, F.: The next 700 impossibility results in time-varying graphs. Int. J. Netw. Comput. (2016)
5. Caceres, R.S., Berger-Wolf, T.: Temporal scale of dynamic networks. In: Temporal Networks, pp. 65–94. Springer (2013)
6. Casteigts, A., Flocchini, P., Quattrociocchi, W., Santoro, N.: Time-varying graphs and dynamic networks. Int. J. Parallel Emerg. Distrib. Syst. **27**(5), 387–408 (2012)
7. George, B., Kim, S.: Time aggregated graph: a model for spatio-temporal networks. In: Spatio-temporal Netw. 7–24. Springer (2013)
8. Holme, P.: Modern temporal network theory: a colloquium. EPJ B **88**(9) (2015)
9. Holme, P., Saramäki, J.: Temporal networks. Phys. Rep. **519**(3), 97–125 (2012)
10. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. J. ACM (JACM) **46**(5), 604–632 (1999)
11. Latapy, M., Viard, T., Magnien, C.: Stream graphs and link streams for the modeling of interactions over time. Social Netw. Anal. Mining **8**(1), 61 (2018)
12. Léo, Y., Crespelle, C., Fleury, E.: Non-altering time scales for aggregation of dynamic networks into series of graphs. Comput. Netw. **148**, 108–119 (2019)
13. Mastrandrea, R., Fournet, J., Barrat, A.: Contact patterns in a high school: a comparison between data collected using wearable sensors, contact diaries and friendship surveys. PLOS ONE (2015)
14. Negrevergne, B., Termier, A., Rousset, M.C., Méhaut, J.F.: Para miner: a generic pattern mining algorithm for multi-core architectures. Data Mining Knowl. Discov. (2014)
15. Ouali, A., Zimmermann, A., Loudni, S., Lebbah, Y., Crémilleux, B., Boizumault, P., Loukil, L.: Integer linear programming for pattern set mining; with an application to tiling. In: PAKDD 2017, Jeju, South Korea, May 23–26, 2017 (2017)

16. Salatino, A., Thanapalasingam, T., Mannocci, A., Osborne, F., Motta, E.: The computer science ontology: a large-scale taxonomy of research areas. In: International Semantic Web Conference (2), pp. 187–205. LNCS (2018)
17. Seidman, S.B.: Network structure and minimum degree. Social Netw. **5** (1983)
18. Silva, A., Meira, W., Jr., Zaki, M.J.: Mining attribute-structure correlated patterns in large attributed graphs. Proc. VLDB Endow. **5**(5), 466–477 (2012)
19. Soldano, H., Santini, G.: Graph abstraction for closed pattern mining in attributed networks. In: ECAI, vol. 263 (2014)
20. Soldano, H., Santini, G., Bouthinon, D.: Attributed graph pattern set selection under a distance constraint. In: Complex Networks, 7th ed., pp. 228–241 (2019)
21. Soldano, H., Santini, G., Bouthinon, D., Bary, S., Lazega, E.: Bi-pattern mining of attributed networks. Appl. Netw. Sci. **4**(1), 37 (2019)
22. Soldano, H., Santini, G., Bouthinon, D., Lazega, E.: Hub-authority cores and attributed directed network mining. In: ICTAI (2017)
23. Viard, T., Soldano, H., Santini, G.: Exploring and mining attributed sequences of interactions (2021). arXiv:2107.13329
24. Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: International Conference on Formal Concept Analysis (2009)
25. Zaki, M.J., Hsiao, C.: CHARM: an efficient algorithm for closed itemset mining. In: SDM, pp. 457–473. SIAM (2002)
26. Zevio, S., Santini, G., Soldano, H., Zargayouna, H., Charnois, T.: A combination of semantic annotation and graph mining for expert finding in scholarly data. In: GEM Workshop at ECML PKDD (2020)

Air Transport Network: A Comparison of Statistical Backbone Filtering Techniques

Ali Yassin, Hocine Cherifi, Hamida Seba, and Olivier Togni

Abstract The big break in data collection tools of large-scale networks from biological, social, and technological domains expands the challenge of their visualization and processing. Numerous structural and statistical backbone extraction techniques aim to reduce the network's size while preserving its gist. Here, we perform an experimental comparison of seven main statistical methods in an air transportation case study. Correlations analysis shows that Marginal Likelihood Filter (MLF), Locally Adaptive Network Sparsification Filter (LANS), and Disparity Filter are biased toward high weighted edges. We compare the extracted backbones using four indicators: the size of the largest component, the number of nodes, edges, and the total weight. Results show that techniques based on a binomial distribution null model (MLF and Noise Corrected Filter) tend to retain many edges. Conversely, Disparity Filter, Polya Urn Filter, LANS Filter, and Global Statistical Significance Filter (GLOSS) are pretty aggressive in filtering edges. The ECM Filter lies between these two behaviors. These results may guide users in selecting appropriate techniques for their applications.

Keywords Complex networks · Backbone filtering techniques · Network compression · Graph summarization · Sparsification

A. Yassin (✉) · H. Cherifi · O. Togni
Laboratoire d'Informatique de Bourgogne, University of Bourgogne, Franche-Comté, Dijon, France
e-mail: aliyassin4@hotmail.com; aliyassin@etu.u-bourgogne.fr

H. Cherifi
e-mail: hocine.cherifi@u-bourgogne.fr

O. Togni
e-mail: olivier.togniu@u-bourgogne.fr

H. Seba
University of Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, 69622 Villeurbanne, France
e-mail: hamida.seba@univ-lyon1.fr

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_43

1 Introduction

In recent decades, networks have become an ideal tool for analyzing complex systems [1–5]. A network can represent any complex system as a collection of nodes connected with edges representing binary interactions. Typical analysis include community detection [6, 7], influential nodes identification [8–10], network formation [11, 12]. However, in many systems, we can measure the magnitude of these interactions. In these cases, we refer to those networks as weighted networks [13].

Large-scale networks make the performance of processing and visualization algorithms challenging. A naive approach for reducing the network size is setting a global threshold on the weights of the edge. However, it destroys the heterogeneity of the distribution of edge weights. Consequently, various backbone extraction techniques have been proposed to reduce the network size while preserving the highest amount of "information." One can classify current backbone extraction techniques into structural and statistical methods. Structural techniques [14–18] filter edges and nodes according to a criterion allowing the latent structure of the network to emerge. Statistical techniques [19–25] assess the significance of an edge through a statistical test, eliminating the least significant edges.

A previous study [26] compared six backbone extraction techniques in the Southeast Asian intercity air transport network [27]. The study includes five structural methods (global weight thresholding, k-core decomposition, minimum spanning tree, primary linkage analysis, multiple linkage analysis), and one statistical (disparity filter) [19, 28–31]. They compare the extracted backbones in terms of their geographical and topological structures, pointing out the potential of each technique in the light of different transport research applications. For instance, they recommend k-core decomposition to analyze the best-connected core for multiplex networks. Primary linkage analysis is well-suited for functional/nodal regions analysis and highlighting hub-and-spoke structures. Finally, they recommend the Disparity Filter for analyzing the overall topological and spatial information for large-scale networks and possible hidden structures.

Neal et al. [32] compared five statistical bipartite backbone extraction techniques: fixed fill model, fixed row model, fixed column model, fixed degree sequence model, and stochastic degree sequence model [33–35]. They used synthetic bipartite networks and two real networks: the Globalization and World Cities (GaWC) and a co-authorship network. The comparative evaluation concerns accuracy, speed, statistical power, similarity, and community structure. Although there is no single winner, they recommend the stochastic degree sequence model for extracting the backbone of most bipartite projections.

In recent decades, there has been considerable interest in analyzing transportation and urban networks [36–38]. Network backbone extraction techniques allow faster analysis and more synthetic visualization. Furthermore, it helps distill the network's critical spatial and topological structures quickly. To our knowledge, there are no prior comparative analyses including prominent statistical backbone extraction techniques in weighted transportation networks. This study fills the gap.

First, we examine the relation between edge weights and the p-values obtained from each backbone extraction technique. Then we extract the backbones for a classical significance level $\alpha = 0.05$ and compare them using four indicators: the size of the components, the number of nodes, edges, and the total weight. Finally, we investigate the evolution of the four indicators in each technique for different significance levels. The main contributions of the paper summarize as follows:

- We perform a comparative analysis of seven statistical backbone extraction techniques on the worldwide air transportation network.
- We explore the relationship between edge weights and the techniques' parameters.
- We evaluate the techniques for different significance levels using four indicators: the size of the components, the number of nodes, edges, and the total weight.

The rest of the paper is organized as follows. Section 2 introduces the backbone extraction techniques under test. Section 3 presents the data and methods. Section 4 the results of the experimental results. Section 5 discusses the main findings. Finally, we make some concluding remarks in Sect. 6.

2 Backbone Extraction Techniques

This section briefly introduces the statistical backbone extraction techniques under evaluation. Although not exhaustive, it covers a large variety of solutions with various degrees of sophistication. We refer readers to the original papers exposing the methods for a detailed description. Except for the Locally Adaptive Network Sparsification Filter (LANS), all these techniques share a common framework. They preserve edges with significant deviations according to a null model for the local or global assignment of weights to edges. The LANS Filter does not use a null model. It compares the observed weight to the empirical distribution of weights.

2.1 Disparity Filter [19]

This very popular technique assumes that the normalized weights of a node's edges follow a uniform distribution. Comparisons of the observed normalized edge weights to this null model allow filtering out edges at a desired significance level α. The backbone retains only statistically significant edges compatible with the null model. Since we define a null model for each node, an edge weight can be significant from the viewpoint of one of its nodes and not the other.

2.2 Polya Urn Filter [20]

In the Pólya Urn [39], observing a particular event increases the probability of further observing it. Similarly, one can assume that edge weights are due to an aggregation process of nodes interacting with each other over time. Based on this assumption, the authors define a null model for each edge from the viewpoint of its incident nodes. Then it calculates the probability that a node distributes its strength (summation of weights) at random through a Pólya process. A reinforcement parameter a governs this mechanism. The higher the value of the reinforcement mechanism a, the larger the weight must be for the edge to be considered significant. Note that the disparity filter is a special case of the Polya Urn Filter.

2.3 Marginal Likelihood Filter [21]

While the Disparity Filter and Polya Urn Filter assess the significance of an edge in the light of each node it connects independently, the Marginal Likelihood Filter considers the two nodes the edge connects. An integer-weighted edge is treated as multiple unit edges. The null model assumes that each unit edge randomly chooses two nodes, which results in a binomial distribution. In other words, it calculates the probability of drawing at least w unit edges from the strength of the network (summation of all weights) with probability proportional to both nodes' strengths.

2.4 Noise Corrected Filter [22]

Similar to the Marginal Likelihood Filter, it assumes edge weights are drawn from a binomial distribution. However, it estimates the probability of observing a weight connecting two nodes using a Bayesian framework. This framework enables us to generate posterior variances for all edges. This posterior variance allows us to create a confidence interval for each edge weight. Finally, we remove an edge if its weight is less than δ standard deviations stronger than the expectation, where δ is the only parameter of the algorithm. It also provides a direct approximation through Binomial distribution similar to the Marginal Likelihood Filter.

2.5 Global Statistical Significance Filter (GLOSS) [24]

The GLOSS filter assumes that one cannot assess edges independently of the overall network topology. Therefore, it defines a global null model for evaluating the significance of an edge. However, it makes no assumptions about the distribution

of weights. Instead, it employs the empirical distribution. The model is a network with the same topology as the original network and edge weights randomly drawn from the empirical weight distribution. In other words, it estimates the probability of observing an edge weight between two given nodes considering the nodes' observed degrees and strengths as constraints.

2.6 Locally Adaptive Network Sparsification Filter (LANS) [25]

Like the GLOSS Filter, it makes no assumptions about the underlying weight distribution. Instead, it employs the empirical cumulative density function to judge statistical significance. Thus, from the viewpoint of edge incident nodes, it calculates the probability of choosing an edge at random with a weight at least equal to the observed weight.

2.7 ECM Filter [23]

Using the Enhanced Configuration Model of network reconstruction employs a null model based on the canonical maximum-entropy ensemble of weighted networks having the same degree and strength distribution as the real network.

3 Data and Method

This section presents the dataset under test, the measures used, and the methodology of the comparative analysis.

3.1 Data

We perform a comparative analysis of the backbone extraction techniques on the Worldwide Air Transportation Network [40, 41]. In this network, A node is an airport. There is an edge between two airports if a direct flight connects them. The weight of an edge indicates the number of flights offered by different companies flying between 17 May and 22 May 2018. Table 1 lists the main global topological properties of the network.

Table 1 The topological features of the Worldwide Air Transportation Networks

# Nodes	# Edges	Average degree	Density	Diameter
2734	16,665	12.191	0.004	12

3.2 Methods

To evaluate the performance of the backbone extraction techniques, we perform three experiments.

First, we examine the relationship between the p-values obtained from the backbone extraction techniques and the weights of the edges. The goal is to evaluate if the techniques are not biased against low weights. Indeed, giving the same importance to small and high weights would preserve the hierarchies at all scale of weights and provides a better representation of the network. Then we compute the correlation between the sample weights and the p values to quantify this relation. We use Pearson Correlation. It measures the linear correlation and relation between two variables. It is given by:

$$\rho(X, Y) = \frac{\sum_{i=1}^{n} (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n} (X_i - \bar{X})^2 \sum_{i=1}^{n} (Y_i - \bar{Y})^2}} \tag{1}$$

where n is the sample size, $\bar{X} = \frac{\sum_i^n X_i}{n}$ is the sample mean of variable X, and $\bar{Y} = \frac{\sum_i^n Y_i}{n}$ n is the sample mean of variable Y. Pearson's correlation values are in the $[-1, +1]$ range. The greater the absolute value of the correlation coefficient, the stronger the relationship. Extreme values indicate a perfect linear relationship. A value of 0 implies no linear dependency between the variables.

Second, we extract the backbone for a fixed classical significance level $\alpha = 0.05$. The nodes, edges, and weights represent the information in any network. We expect the extracted backbones to preserve the highest possible information in one giant component. For evaluation, we use the following four indicators: the size of the components, the number of nodes, edges, and the total weight of each backbone.

Finally, we investigate the consistency of the previous experiment by comparing the backbone extraction techniques using different significance levels.

4 Experimental Results

This section reports the results of the three experiments used to compare the backbone extraction techniques. Note that the Polya Urn Filter requires a parameter. We set this parameter to 1.0119, following a fine-tuning process. Additionally, for methods assessing the significance of an edge from each node alone, we consider an edge significant if it is valid for at least one node.

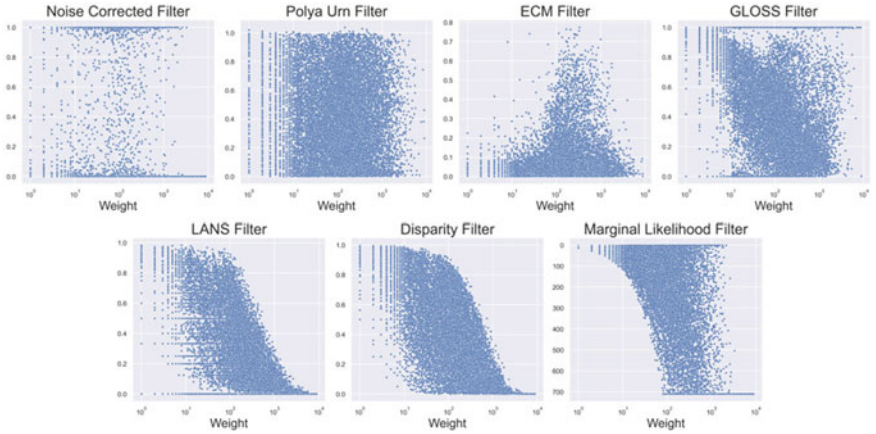


Fig. 1 The weights and p-values of the edges for the backbone extraction techniques under evaluation

4.1 Correlation of the Backbone Extraction Techniques

Before looking at the extracted backbones and their properties, we examine the computed p-values by each backbone extraction technique. The backbone extraction techniques address the limitation of the naive filtering approach, which sets a cut-off on the weights. Thus, a good backbone extraction technique should give equal importance to edges with different weight scales. Figure 1 shows a scatterplot for the weights and p-values corresponding to the backbone extraction techniques. In the top row, one can see that the Noise Corrected Filter, Polya Urn Filter, ECM Filter, and GLOSS Filter give importance to small and high weights. Indeed, they assign small and high p-values for all scales of weights. In contrast, the bottom row shows a narrow distribution of p-values at the extremes. Indeed, the small and high weights at the extremes are associated with high and low p-values, respectively. We compute the Pearson Correlation between the sample weights and p-values of each backbone extraction technique to validate these findings. Table 2 reports these results. It shows no critical correlation between the p-values and the weights for all the techniques. We do not see any correlation between the weight and p-values of the Noise Corrected Filter, Polya Urn Filter, ECM Filter, and GLOSS Filter. However, we notice a low correlation ($0.3 < \rho < 0.5$) between the weights and p-values of the Marginal Likelihood Filter, Disparity Filter, and LANS Filter.

Table 2 Pearson correlation $\rho(X, Y)$ between X the weights and Y the p-values associated with the edges using different backbone extraction techniques: noise corrected filter (NC), polya urn filter (PU), ECM filter (ECM), GLOSS filter (GLOSS), LANS filter (LANS), disparity filter (D), and marginal likelihood filter (ML)

NC	PU	ECM	GLOSS	LANS	D	ML
0.01	0.01	0.08	0.13	0.37	0.48	0.51

4.2 Comparing the Backbone Extraction Techniques for the Same Significance Level

Ideally, a sound filtering technique should preserve the highest amount of information while filtering as many connections as possible and avoiding the breakup of the system. Thus, we consider four indicators to evaluate the backbone extraction techniques: the size of the components, number of nodes, edges, and the total weight. In this experiment, we extract all the backbones using a significance level $\alpha = 0.05$ classically used. Table 3 reports the size of the most significant connected component and the number of components with a size less than 5% for each backbone extraction technique. The Marginal Likelihood Filter, Noise Corrected Filter, and ECM Filter extract a single giant component containing all the backbone nodes. In contrast, the other techniques extract a giant component and multiple small-size components (less than 5% of nodes). However, one can notice that the size of the giant component is greater than 80%. For LANS Filter, Disparity Filter, and Polya Urn Filter retain while it is around 50% for the GLOSS Filter.

Table 4 reports the percentage of isolated nodes, preserved edges, and weights for each backbone extraction technique. One can see that the Marginal Likelihood Filter, Noise Corrected Filter, ECM Filter, and LANS Filter preserve all the nodes of the original network. In comparison, the other techniques isolate more than 60% of the nodes. We observe two typical behavior related to the percentage of edges preserved. In a so-called conservative category, extracted backbones keep a high fraction of

Table 3 Size of the largest connected component and number of components with a size less than 5% for each filtering technique in the worldwide air transportation network

Filter	% LCC	# C < 5%
Marginal likelihood	100	–
Noise corrected	100	–
ECM	100	–
LANS	94	36
Disparity	91.1	19
Polya urn	85	62
GLOSS	50.9	102

Table 4 Percentage of isolated nodes, preserved edges, and weights by each backbone extraction technique in the worldwide air transportation network

Filter	% Isolated nodes	% Edges	% Weights
Marginal likelihood	0	90.9	93.4
Noise corrected	0	84.3	85.9
ECM	0	68.4	57.5
LANS	0	20.9	40.3
Disparity	70.5	9.8	39.5
Polya urn	62.4	9.6	10.6
GLOSS	69.6	5.1	6.4

edges (greater than 80%). This category regroups the Marginal Likelihood Filter and Noise Corrected Filter. In contrast, the second behavior is very aggressive in filtering edges. In that case, the extraction technique removes more than 80% of the edges. Except for the ECM Filter that lies between these two behaviors and preserves 68% of the edges, the other methods embrace this behavior. Finally, the last column shows that the Marginal Likelihood Filter and the Noise Corrected Filter preserve a high percentage (85%) of the total weights. Indeed, it is due to the large fraction held edges. In contrast, the Polya Urn and the GLOSS Filter preserve less than 10% of the weights. At the same time, the other techniques keep between 40 and 60% of the total weights.

4.3 Comparing the Backbone Extraction Techniques for Different Significance Levels

To further analyze the previous experiment's results, we extend the investigation to the evolution of the four indicators while varying the significance levels. Figure 2 illustrates the results of this experiment. We observe a strong filtering regime corresponding to $\alpha < 10^{-2}$. The top left panel depicts the evolution of the edge proportion preserved. One can consider three typical behaviors. In the first category, the backbones maintain a steady percentage of edges across all the range of the significance levels, even in the strong filtering regime. The Marginal Likelihood Filter and Noise Corrected Filter belong to this category preserving around 85% of the edges. The second category includes the ECM Filter. In that case, the percentage of edges increases monotonically with the significance levels at a low rate within the strong filtering regime and at a high rate after that. The third category is characterized by a steady percentage of edges in the strong filtering regime followed by an exponential increase. Polya Urn Filter, Disparity Filter, GLOSS Filter, and LANS Filter belong to this category. However, in the strong filtering regime, LANS Filter preserves a consistent percentage of edges (around 18%) while the others hardly keep any edge.

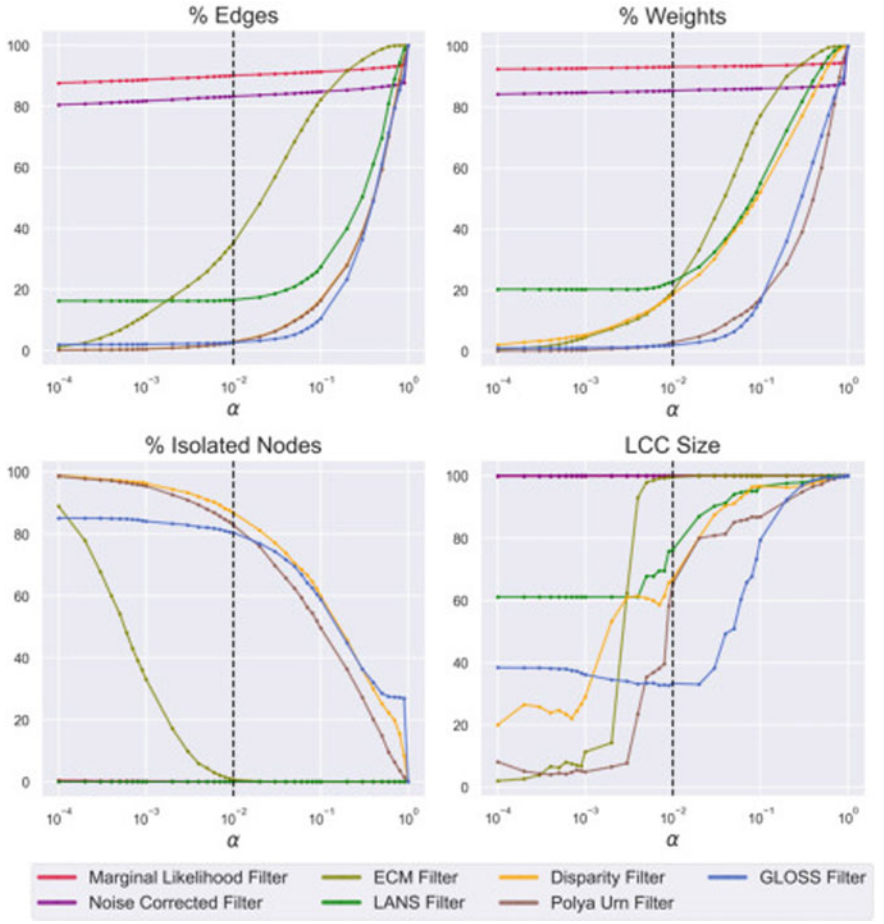


Fig. 2 The percentage of edges, weights, isolated nodes, and the largest connected component size in the extracted backbone using different backbone extraction techniques in the worldwide air transportation network as a function of different significance levels α

The top right panel reports the evolution of weights. Overall, the curves are pretty similar. Consequently, we can consider the same categories and members, except for the Disparity Filter, which joins the ECM filter in the second category. In the strong regime $\alpha < 10^{-2}$, the Disparity Filter preserves the same fraction of weights as the ECM Filter, although it holds a tiny portion of edges. While the ECM Filter keeps a more significant amount of edges, reaching 40% for $\alpha = 10^{-2}$. It shows how the Disparity Filter prioritizes high weights while the ECM Filter preserves weights with different scales. For $\alpha > 10^{-2}$, the Disparity Filter retains the same percentage of weights as the LANS Filter, validating that it prioritizes high weights.

The bottom left panel reports the evolution of the fraction of isolated nodes. The Marginal Likelihood Filter and Noise Corrected Filter do not separate any node from the network because they hardly filter any edges. The LANS Filter also preserves all the nodes even when filtering around 80% of the edge. In contrast, except for the ECM Filter, all the other methods isolate a significant portion of nodes in the strong regime. Indeed, the percentage of isolated nodes decays until there are no more isolated nodes as we reach $\alpha = 10^{-2}$. After that, the percentage of isolated nodes decreases with the increase of the significance level. It illustrates how the techniques fail to preserve all the nodes while filtering edges.

Unlike the Marginal Likelihood Filter and Noise Corrected Filter, other filters do not retain one giant component, as shown in the last panel. The LANS Filter and GLOSS Filter maintain a giant component with a fixed size in the strong regime. Its size increases gradually to form a unique component only when adding all the edges. In contrast, we notice the emergence of a giant component in the Disparity Filter, Polya Urn Filter, and ECM Filter as we approach the boundaries of the strong regime. After that, the ECM Filter stops isolating nodes while the others stop only when maintaining all the edges.

5 Discussion

This study compares seven statistical backbone extraction techniques in the Worldwide Air Transportation network. First, to answer if the backbone extraction techniques give the same importance to small and high edge weights, we examine the relationship between the edge weights and p-values obtained from the backbone extraction techniques. The lower the correlation between the weights and p-values, the better the method in preserving edges with different scales of weights. All techniques maintain all types of weights. However, we notice that the Marginal Likelihood Filter, Disparity Filter, and LANS Filter give more importance to edges with high weights. Indeed, edges with high weight are more likely to be significant, and edges with small weight are more likely to be non-significant. These findings are reported in the last column of Table 5.

We compare the extracted backbones for a significance level $\alpha = 0.05$ using four indicators of performance (the size of the components, number of nodes, edges, and total weight). Recall that a significance level of 0.05 indicates a 5% risk of concluding that an edge is significant when it is not. Results show that the Marginal Likelihood and Noise Corrected Filter based on a binomial distribution null model retain many edges. Consequently, they generate almost complete networks. They have a conservative behavior preserving all the nodes in a giant component with a high percentage of weights.

In contrast, except ECM Filter, the other techniques are very aggressive in filtering edges for this significance level. Indeed, they isolate many nodes, losing many weights. The ECM Filter lies in between these two extremes.

Table 5 Properties of each backbone extraction technique: noise corrected filter (NC), polya urn filter (PU), ECM filter (ECM), GLOSS filter (GLOSS), LANS filter (LANS), disparity filter (D), and marginal likelihood filter (ML)

	Technique behavior	Retains a LCC	Isolate nodes	Prioritized weight scale
ML	Conservative	Yes	No	High
NC	Conservative	Yes	No	Small and high
ECM	Flexible	$\alpha > 10^{-2}$	$\alpha < 10^{-2}$	Small and high
LANS	Aggressive	Yes	No	High
D	Aggressive	$\alpha > 10^{-2}$	Yes	High
PU	Aggressive	$\alpha > 10^{-2}$	Yes	Small and high
GLOSS	Aggressive	$\alpha > 10^{-2}$	Yes	Small and high

We study the evolution of the four indicators with significance levels. Results show that we can distinguish a strong regime corresponding to $\alpha < 10^{-2}$ and a weaker regime corresponding to $\alpha > 10^{-2}$. Table 5 reports the properties of each backbone extraction technique. We can classify the techniques into conservative, flexible, and aggressive according to their ability to filter edges across all significance levels. Conservative techniques do not isolate nodes and always retain a large component. The Flexible technique is dependent on the significance level. It isolates nodes for $\alpha < 10^{-2}$ and retains a large component for $\alpha > 10^{-2}$. The Aggressive techniques always isolate nodes regarding the significance levels and retain a large component only in the weak regime except for the LANS Filter. It always retains a large component.

6 Conclusion

Reducing the size of a network has always attracted scientists due to its many applications. This work compares the statistical backbone extraction techniques in the worldwide air transportation network. Results show that the Marginal Likelihood Filter, Disparity Filter, and LANS Filter give more importance to high-weight edges. The other techniques emphasize both small and high-weighted edges. We show that filters based on a binomial distribution are conservative techniques. Indeed, they always retain a high proportion of links and nodes. Other filters beside the ECM Filter are aggressive in removing edges for reasonable significance levels $10^{-2} \leq \alpha \leq 0.05$. This behavior translates to a high proportion of isolated nodes. The ECM Filter is more flexible. It lies between these two extreme behaviors. Future work will consider an extensive investigation of various types of networks (synthetic and real-world) to consolidate these findings.

Acknowledgements This material is based upon work supported by the Agence Nationale de Recherche under grant ANR-20-CE23-0002.

References

1. Vespignani, A.: Twenty years of network science (2018)
2. Rital, S., Bretto, A., Cherifi, H., Aboutajdine, D.: A combinatorial edge detection algorithm on noisy images. In: International Symposium on VIPromCom Video/Image Processing and Multimedia Communications, pp. 351–355. IEEE (2002)
3. Messadi, M., Cherifi, H., Bessaid, A.: Segmentation and abcd rule extraction for skin tumors classification (2021). arXiv:2106.04372
4. Lasfar, A., Mouline, S., Aboutajdine, D., Cherifi, H.: Content-based retrieval in fractal coded image databases. In: Proceedings 15th International Conference on Pattern Recognition. ICPR-2000, vol. 1, pp. 1031–1034. IEEE (2000)
5. Pastrana-Vidal, R.R., Gicquel, J.-C., Colomes, C., Cherifi, H.: Frame dropping effects on user quality perception. In: Proceedings of 5th International WIAMIS (2004)
6. Fortunato, S., Hric, D.: Community detection in networks: a user guide. Community detection in networks: a user guide. Phys. Rep. **659**, 1–44 (2016)
7. Cherifi, H., Palla, G., Szymanski, B.K., Lu, X.: On community structure in complex networks: challenges and opportunities. Appl. Netw. Sci. **4**(1), 1–35 (2019)
8. Ibnoulouafi, A., El Haziti, M., Cherifi, H.: M-centrality: identifying key nodes based on global position and local degree variation. J. Stat. Mech. Theory Exp. (7), 073407 (2018)
9. Kumar, M., Singh, A., Cherifi, H.: An efficient immunization strategy using overlapping nodes and its neighborhoods. Companion Proc. Web Conf. **2018**, 1269–1275 (2018)
10. Chakraborty, D., Singh, A., Cherifi, H.: Immunization strategies based on the overlapping nodes in networks with community structure. In: International Conference on Computational Social Networks, pp. 62–73. Springer, Cham (2016)
11. Orman, K., Labatut, V., Cherifi, H.: An empirical study of the relation between community structure and transitivity. In: Complex Networks, pp. 99–110. Springer, Berlin, Heidelberg (2013)
12. Orman, G.K., Labatut, V., Cherifi, H.: Towards realistic artificial benchmark for community detection algorithms evaluation (2013). arXiv:1308.0577
13. Barrat, A., Barthélemy, M., Pastor-Satorras, R., Vespignani, A.: The architecture of complex weighted networks. Proc. Nat. Acad. Sci. **101**(11), 3747–3752 (2004)
14. Grady, D., Thiemann, C., Brockmann, D.: Robust classification of salient links in complex networks. Nat. Commun. **3**, 864 (2012)
15. Simas, T., Correia, R.T., Rocha, L.M.: The distance backbone of complex networks. J. Complex Netw. **9** (2021)
16. Rajeh, S., Savonnet, M., Leclercq, E., Cherifi, H.: Modularity-based backbone extraction in weighted complex networks (2022)
17. Ghalmane, Z., Cherifi, C., Cherifi, H., El Hassouni, M.: Extracting backbones in weighted modular complex networks. Sci. Rep. **11**, 12 (2021)
18. Ghalmane, Z., Cherifi, C., Cherifi, H., El Hassouni, M.: Extracting modular-based backbones in weighted networks. Inf. Sci. **576**, 454–474 (2021)
19. Serrano, M.A., Boguna, M., Vespignani, A.: Extracting the multiscale backbone of complex weighted networks. Proc. Nat. Acad. Sci. **106**, 6483–6488 (2009)
20. Marcaccioli, R., Livan, G.: A pólya urn approach to information filtering in complex networks. Nat. Commun. **10**, 745 (2019)
21. Dianati, N.: Unwinding the hairball graph: Pruning algorithms for weighted complex networks. Phys. Rev. E **93** (2016)
22. Coscia, M., Neffke, F.M.H.: Network backboning with noisy data, pp. 425–436. IEEE (2017)
23. Gemmetto, V., Cardillo, A., Garlaschelli, D.: Irreducible network backbones: unbiased graph filtering via maximum entropy (2017)
24. Radicchi, F., Ramasco, J.J., Fortunato, S.: Information filtering in complex weighted networks. Phys. Rev. E **83**, 046101 (2011)
25. Foti, N.J., Hughes, J.M., Rockmore, D.N.: Nonparametric sparsification of complex multiscale networks. PLoS ONE **6**, e16431 (2011)

26. Dai, L., Derudder, B., Liu, X.: Transport network backbone extraction: a comparison of techniques. J. Transp. Geogr. **69**, 271–281 (2018)
27. Dai, L., Derudder, B., Liu, X.: The evolving structure of the southeast Asian air transport network through the lens of complex networks, 1979–2012. J. Transp. Geogr. **68**, 04 (2018)
28. Alvarez-Hamelin, J., Dall'Asta, L., Barrat, A., Vespignani, A.: Large scale networks fingerprinting and visualization using the k-core decomposition, **18**, 11 (2005)
29. J Jr. On the shortest spanning subtree of a graph and the traveling salesman problem. Proc. Am. Math. Soc. **7**, 48–50 (1956)
30. Nystuen, J., Dacey, M.: A graph theory interpretation of nodal regions. Papers Region. Sci. Assoc. **7**, 01 (2005)
31. Rushton, G., Haggett, P., Cliff, A., Frey, A.: Locational analysis in human geography. Geogr. Rev. **70**, 112 (1980)
32. Neal, Z.P., Domagalski, R., Sagan, B.: Comparing models for extracting the backbone of bipartite projections (2021)
33. Zweig, K., Kaufmann, M.: A systematic approach to the one-mode projection of bipartite graphs. Social Netw. Analys. Mining **1**, 187–218 (2011)
34. Saracco, F., Straka, M., Clemente, R.D., Gabrielli, A., Caldarelli, G., Squartini, T.: Inferring monopartite projections of bipartite networks: an entropy-based approach. New J. Phys. **19**, 053022 (2017)
35. Tumminello, M., Micciche, S., Lillo, F., Piilo, J., Mantegna, R.: Statistically validated networks in bipartite complex systems. PloS One **6**, e17994 (2011)
36. Ducruet, C., Rozenblat, C., Zaidi, F.: Ports in multi-level maritime networks: evidence from the Atlantic (1996–2006). J. Transp. Geogr. **18**, 508–518 (2010)
37. O'Kelly, M.: Global airline networks: comparative nodal access measures. Spatial Econ. Anal. pp. 1–23 (2016)
38. Liu, X., Derudder, B., Kang, W.: Measuring polycentric urban development in china: an intercity transportation network perspective. Region. Stud. **50**, 03 (2015)
39. Haigh, J.: Polya urn models. J. R. Stat. Soc. Ser. A **172**, 942 (2009)
40. Alves, L., Aleta, A., Rodrigues, F., Moreno, Y., Amaral, L.: Centrality anomalies in complex networks as a result of model over-simplification. New J. Phys. **22**, 01 (2020)
41. Diop, I.M., Cherifi, C., Diallo, C., Cherifi, H.: Revealing the component structure of the world air transportation network. Appl. Netw. Sci. **6**(1), 1–50 (2021)

Towards the Concept of Spatial Network Motifs

José Ferreira, Alberto Barbosa, and Pedro Ribeiro

Abstract Many complex systems exist in the physical world and therefore can be modeled by networks in which their nodes and edges are embedded in space. However, classical network motifs only use purely topological information and disregard other features. In this paper we introduce a novel and general subgraph abstraction that incorporates spatial information, therefore enriching its characterization power. Moreover, we describe and implement a method to compute and count our spatial subgraphs in any given network. We also provide initial experimental results by using our methodology to produce spatial fingerprints of real road networks, showcasing its discrimination power and how it captures more than just simple topology.

Keywords Spatial networks · Subgraphs · Network motifs

1 Introduction

Complex networks are a very powerful abstraction of real-world systems that allow us to analyze the underlying interactions [11]. Many of these systems have a correspondence to the physical world, such as transportation networks (e.g. road, train or subway), power grids or brain networks. Their components are therefore embedded in space and topology alone does not capture all the relevant information [1]. Being

J. Ferreira · A. Barbosa · P. Ribeiro (✉)
Departamento de Ciência de Computadores, Faculdade de Ciências,
Universidade do Porto, R. Campo Alegre s/n, 4169-007 Porto, Portugal
e-mail: pribeiro@dcc.fc.up.pt

J. Ferreira
e-mail: jcff@fc.up.pt

A. Barbosa
e-mail: alberto.barbosa@fc.up.pt

A. Barbosa · P. Ribeiro
CRACS & INESC-TEC, Porto, Portugal

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_44

able to understand and analyze these spatial networks is therefore a crucial task with multidisciplinary applicability [2, 3].

Subgraphs can be seen as the building blocks of networks and they are the core of rich characterization concepts such as network motifs [9] or graphlet degree distributions [13]. Despite extensions to incorporate dimensions such as weight [4], time [12], color [15] or multiple layers [16], to the best of our knowledge there is no general and widespread subgraph abstraction that incorporates the spatial dimension. We should note that for specific domains there has been some related work, such as in football, where passing networks between different regions of the playing field have been created [10], but these remain specialized and restricted to their own field of study.

In this paper we try precisely to aim towards a general concept of spatial motifs able to characterize networks from any domain. Our first contribution (Sect. 2) is a novel subgraph abstraction that incorporates spatial information in a way that is general enough to incorporate several spatial dimensions (e.g. 2D or 3D) and granularities (e.g. large macroscale vs small microscale regions). The key idea is to automatically create a spatial partition of the subgraph bounding and to color the nodes according to the region they are on. Our second contribution (Sect. 3) is an initial methodology and fully functional framework to detect and count these spatial motifs, based on enumerating subgraph occurrences and then computing their spatial and topological type. Our third and last contribution (Sect. 4) is a proof of concept experimental section, in which we analyze several real word road networks, showing that unlike purely topological motifs, we can distinguish between grid and non grid-like layouts.

2 A Novel Concept of Spatial Motifs

There are several possible ways for expressing spatial properties. For instance, the distance between nodes can be used as edge weight [7], but this would not take into account the relative position of the nodes. Another option would be to use angles between nodes, hence losing the distance information. Our approach relies on first creating a *bounding box* around the found subgraph using the nodes spatial location, and then partitioning this box into regular-sized regions, thus taking into account both the relative position and the distance between nodes.

For the sake of simplicity and given the space constraints, we will mainly focus on a 2D example divided into 2×2 quadrants, but as explained later, our approach is general and extends naturally to higher dimensions. The creation of the bounding box is straight-forward: we require a set of coordinates for each node on the input, and for each found subgraph, we calculate the maximum and the minimum of both the x and the y values, which gives us the limits of our box. Then, we simply calculate the relative position of each node when referring to the center of the bounding box, assigning a quadrant to the node on the form of a color. An example can be seen in

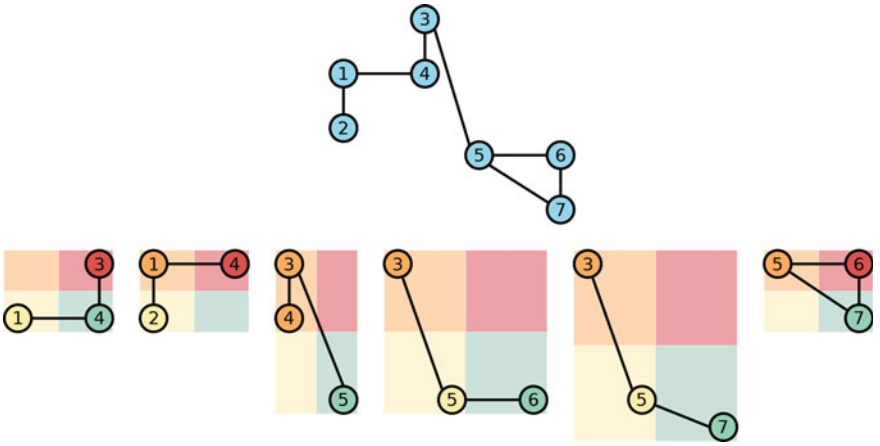


Fig. 1 Example of subgraphs with spatial coloring by 2D quadrants

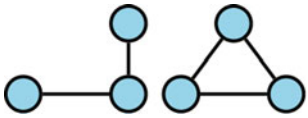


Fig. 2 Chain (type *A*) and triangle (type *B*) topological subgraphs

Fig. 1, where the original spatial network is given above, in the blue nodes, and all its three node spatial subgraph occurrences are given below.

Taking into account the spatial dimension of the nodes on the previous figure, we can enumerate five different subgraph types, being the fourth ({3, 5, 6}) and the fifth occurrence ({3, 5, 7}) of the same type: they both have three nodes in the same quadrants (one orange, one yellow and one green) and the same connections (one orange-yellow edge and another yellow-green edge).

By contrast, if only purely topological properties were used, there would exist only two subgraph types, as depicted in Fig. 2, with the first five occurrences being a chain of three nodes and the last one (({5, 6, 7}) being a triangle.

The above example already illustrates how much richer our spatial representation is, but we would like to emphasize how general our conceptual approach is. From a scale point of view, it naturally extends to higher numbers of nodes (just consider more nodes in each subgraph). From a topological point of view, it is also able to organically integrate features such as direction (just consider that when distinguishing between different isomorphic types). From a granularity point of view, we can also consider any regular division. Here we exemplified with 2×2 quadrants, but we could use any $n \times n$ partition, depending on what we want to measure (and moreover we could even use on the same analysis subgraph occurrences at different n sizes to create a richer set of features). Finally, our approach also naturally extends to higher dimensions (for instance, in a 3D space one could use $2 \times 2 \times 2$ octants as the equivalent of 2D quadrants).

3 Finding and Counting Spatial Motifs

In this section we explain our methodology for finding and counting the occurrences of spatial motifs as defined in the previous section. The motivation for counting will become clearer on Sect. 4, but essentially by computing subgraph frequencies we are able to obtain numerical features characterizing the underlying network. Counting subgraphs is therefore a core network analysis primitive. A fully detailed survey on how to count purely topological motifs can be seen in [14], including approximate and parallel approaches.

Our proposed initial approach has two steps: (i) we first enumerate all subgraph occurrences of a given size k, obtaining sets of k connected nodes; (ii) for each occurrence we identify its spatial type by producing a canonical labeling that is unique to each colored isomorphic type. A fully functional implementation is available at github.[1]

3.1 Enumerating Subgraph Occurrences

In order to enumerate the occurrences of subgraphs with k nodes, we opted to use ESU [17], a general purpose subgraph enumeration algorithm capable of finding each occurrence only once, avoiding symmetries. In short, this is done by starting from a single node and expanding from there, using only vertices that have an index (label at the original graph) greater than that of the original node and that can be neighbors of a newly added vertex but not of any other one previously added. In Fig. 3 we illustrate this process with a small example for $k = 3$ and an original network of six nodes. Inside each tree node box we indicate two node sets: first the current subgraph being enumerated ($V_{subgraph}$) and secondly the set of nodes which can expand it ($V_{extension}$).

The root of the search tree is a starting point to evaluate the subgraph. It's children, on the second level, correspond essentially to one branch per node, with the extension sets being their immediate neighbors with a larger index than the node itself. For instance, the second branch contains $V_{subgraph} = \{2\}$ and $V_{extension} = \{3\}$ (3 is a neighbor of 2 and 1 is not considered since $1 < 2$ and the subgraph with $\{1, 2\}$ would be already considered in the first branch). This process continues in the following tree levels: we add 3 to $V_{subgraph}$ since it has two neighbors that meet the requirements, those are added to $V_{extension}$, resulting in $V_{subgraph} = \{2, 3\}$ and $V_{extension} = \{4, 5\}$. Now we have two possible branches, $V_{subgraph} = 2, 3, 4$ and $V_{subgraph} = 2, 3, 5$. In both these cases we have $|V_{subgraph}| = 3$ and we have reach the desired node set size.

After doing this to every single node we end up with the subgraphs of size 3 represented on the leafs of the tree. The required conditions for a node to be added to $V_{extension}$ make sure that no subgraph is found twice.

[1] We will make available a github link to the source code if the paper is accepted.

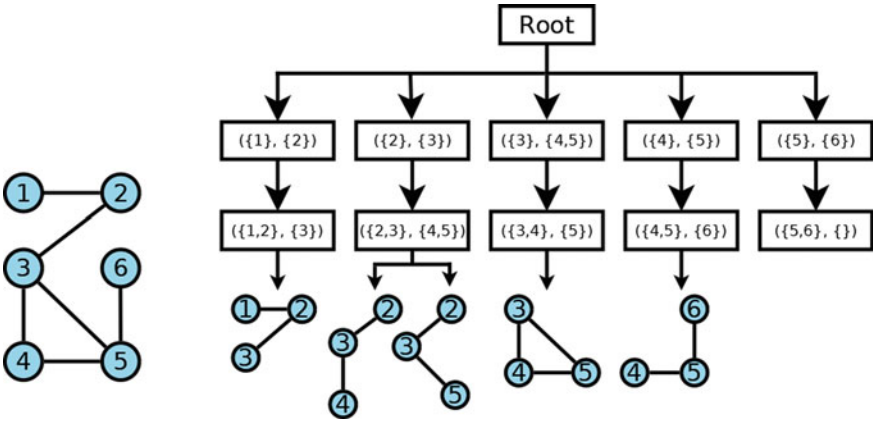


Fig. 3 Example of an ESU search tree for k-subgraph enumeration with $k = 3$

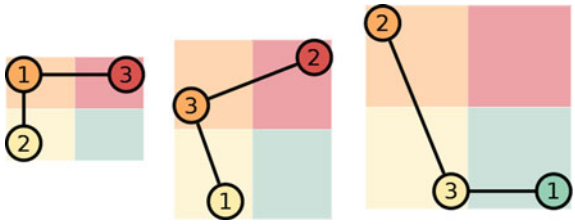


Fig. 4 The first two subgraphs are of the same type and should have the same canonical labeling; the third one should have a different labeling

3.2 Subgraph Types and Canonical Labeling

After having the node sets that correspond to each subgraph occurrence, we still need to discover the spatial type of each one, so that we can increment its frequency. For instance, as we could observe in Fig. 1 that the fourth and fifth subgraphs belong to the same type.

In our approach, we first determine the bounding box of each occurrence by computing the minimum and maximum values of each spatial dimension. We then partition the box into the desired number of regions and we "color" the nodes according to the region in which each one falls, effectively obtaining what could be considered a colored subgraph [15]. Afterwards, we compute a canonical labeling such that two subgraph occurrences will have the same labeling if and only if they correspond to the same (colored) isomorphic type.

Figure 4 illustrates the need for a canonical labeling that takes node colors into account. From a purely topological point of view, all three subgraphs are chains and therefore indistinguishable. However, when incorporating spatial information, this is not the case. We want the first and second subgraphs to have the same label, as they have the same colored topological properties: one node in each quadrant except the fourth one (represented by the colors orange, red and yellow), and two

connections (an orange-red edge and orange-yellow edge). The third subgraph has different spatial properties that correspond to different colored nodes and edges.

In general, even without colors, computing canonical labelings is a very hard computational task, closely related to the graph isomorphism problem [5]. We therefore resorted to `nauty` [8], a third-party and very efficient set of procedures to determine the automorphism group of a vertex-colored graph. Since `nauty` has built-in support for colored nodes, a call to the default method with the required arguments and the quadrant as color is enough to give us the canonical label. To achieve a labeling using colors, `nauty` requires the colors to be given in some order, and the edge labels will be returned in the order the colors were provided, that is, first the edges with the first color, then the ones with the second color, and so on.

4 Experimental Results

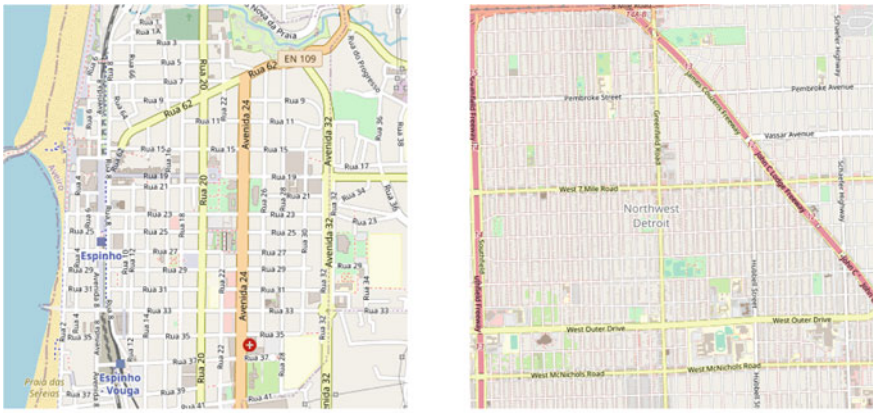
In order to test our implementation and to showcase the applicability of our proposed subgraph abstraction, we now provide a proof of concept. We use real world road networks, which can be considered one of the quintessential spatial network examples to which everyone can relate to. We selected two cities with "grid-like" street layouts (Espinho, Portugal and Detroit, USA) and two with "non grid-like" layouts (Porto, Portugal and Oxford, UK), aiming to distinguish between these two layout groups using our proposed approach.

Our original source of raw street data was OpenStreetMap (OSM) [6], which is a collaborative project that aims to provide a free editable geographic database of the entire world. In Fig. 5, we show an image of the approximate area used for each of the cities mentioned above, taken from OSM, which clearly indicates the nature of the road layouts that are being analyzed.

From the raw data we create a network in which the nodes are true road intersections and edges represent roads between them (this implied the creation of an automated script that given a geographical bounding box will extract all OSM features from it, which are further processed and simplified to create the desired intersection network). In this network, we fix the subgraph size to $k = 3$ and count the number of occurrences of each spatial subgraph. Figure 6 exemplifies this process for a small portion of a map, illustrating the extracted network and all its occurring subgraph occurrences, some of them belonging to the same spatial type. For instance, $\{1, 2, 3\}$ and $\{3, 4, 5\}$ are of the same type, and the same can be said for $\{1, 2, 7\}$ and $\{3, 4, 6\}$.

To better understand and visualize the differences in subgraph occurrences, we opted to further divide spatial types into *classes* (families of subgraphs), that corresponds to the four 90 °C rotations of the same simple type. Figure 7 illustrates this concept and one class of subgraphs.

Figure 8 illustrates representatives of the six most frequent classes of subgraphs that we found in the studied road networks (the frequency of the other possible classes is residual and their very low relative frequency does not impact the conclusions of our analysis).



(a) Espinho (b) (Northwest) Detroit

(c) Porto (d) Oxford

Fig. 5 Layout of the four cities used as input

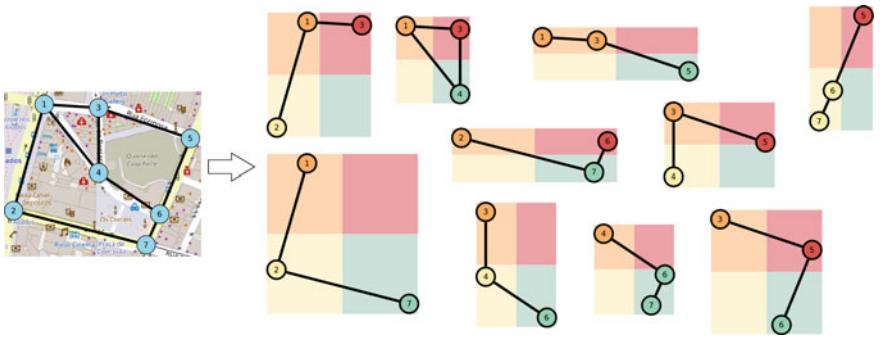


Fig. 6 The network corresponding to a map and its subgraph enumeration

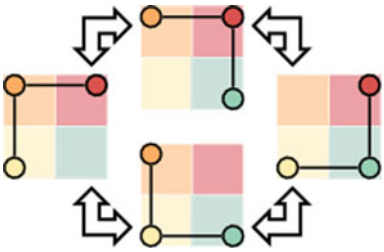


Fig. 7 Subgraph class 1 and its four spatial subgraph types, corresponding to 90 °C rotations

(a) Class 1 (b) Class 2 (c) Class 3 (d) Class 4 (e) Class 5 (f) Class 6

Fig. 8 Representatives of the most frequent classes of subgraphs considered

(a) Espinho (b) Detroit

Fig. 9 Top-4 of subgraphs with most occurrences in the two grid-like cities. Note that all the subgraphs are of class 1 (as defined in Fig. 8)

4.1 Results for "Grid-Like" Street Layouts

Figure 9 represents the top-4 (in order, from left to right) of the subgraphs with most occurrences in Espinho and Detroit. The results are as expected and capture the grid-like nature of the layout. Even if the exact order of simple types is not exactly the same, this top-4 corresponds to class 1 subgraphs, whose representation resembles a right angle, that is, where each node is in a different quadrant and the subgraph is a chain that connects nodes in consecutive quadrants.

The difference in frequency from the 5th to the 4th most common subgraph is noticeable, particularly in the case of Espinho. In both cases, subgraph types from the 5th to the 8th positions are of class 2. Tables 1 and 2 give more detail on the results, showing the relative frequency (percentage of total occurrences) of the 10 most common subgraph types and their associated class. In total, there were 22 different subgraph types found for Espinho and 32 for Detroit.

Table 1 Spatial subgraph frequencies in Espinho

Rank	Subgraph type	Relative frequency
1	1	0.178258
2	1	0.167738
3	1	0.161894
4	1	0.160140
5	2	0.089421
6	2	0.080070
7	2	0.046756
8	2	0.044418
9	4	0.010520
10	3	0.009351
Top-10 total	–	0.948567

Table 2 Spatial subgraph frequencies in Detroit

Rank	Subgraph type	Relative frequency
1	1	0.139369
2	1	0.138028
3	1	0.135329
4	1	0.134110
5	2	0.108859
6	2	0.097870
7	2	0.086585
8	2	0.083102
9	3	0.019452
10	3	0.019208
Top-10 total	–	0.961913

4.2 Results for "Non Grid-Like" Street Layouts

On the other hand, if the city does not have a well defined grid layout, we can observe that the most frequent subgraphs are very different, as can be seen in Fig. 10. In fact, the top-4 for these two cities does not have any subgraph type in common with the grid-like cities. Here, the topological chain type of subgraph is still the most common, which means that if only the topological information of the subgraphs was used, conclusions with this level of detail would not be possible, but in this case instead of having each node in a different quadrant, two nodes share a quadrant, there is a connection between them, and one of them connects to a node in the opposite quadrant, with the entire top-4 of most frequent subgraphs being of the same class.

(a) Porto (b) Oxford

Fig. 10 Top-4 of subgraphs with most occurrences in the two non grid-like cities. Note that all the subgraphs are of class 2 (as defined in Fig. 8)

Table 3 Spatial subgraph frequencies in Porto

Rank	Subgraph type	Relative frequency
1	2	0.133904
2	2	0.131815
3	2	0.118655
4	2	0.118446
5	1	0.081888
6	1	0.080426
7	1	0.079590
8	1	0.075621
9	3	0.023605
10	3	0.022143
Top-10 total	–	0.866095

As on the previous section, we give detailed results of the top-10 most common subgraph types in Tables 3 and 4. Again there is a noticeable increase in frequency from the 5th to the 4th most common subgraph, and the same class appears from rank 5 to rank 8. In total, there were 28 different subgraphs found for both Porto and Oxford.

4.3 Comparison Between Cities

In Fig. 11 we can observe a bar chart of the relative frequency of each subgraph class per city, with the usage of their spatial properties.

If we remove the spatial component from the subgraphs, we are left with only two types of subgraphs: chains and triangles. This means that spatial classes 1–4 will be of the single topological type *A* (chain), and classes 5 and 6 will of the the topological type *B* (triangle). Using the same data as the previous plot but ignoring the spatial properties results in the bar chart depicted in Fig. 12.

We can observe from Fig. 11 that the distribution of the subgraph classes clearly shows a predominance of class 1 in the two grid-like cities, whilst class 2 is more common in the two cities without this layout, which allows us to easily distinguish

Table 4 Spatial Subgraph frequencies in Oxford

Rank	Subgraph type	Relative frequency
1	2	0.149768
2	2	0.148675
3	2	0.133916
4	2	0.121618
5	1	0.074064
6	1	0.071331
7	1	0.070511
8	1	0.066684
9	4	0.024050
10	4	0.022683
Top-10 total	–	0.883302

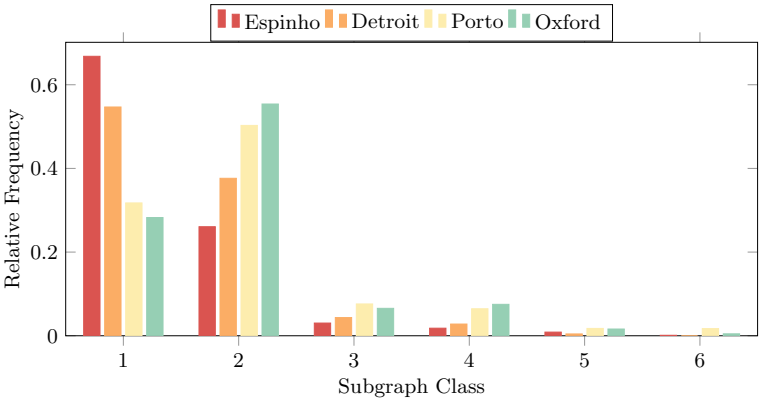
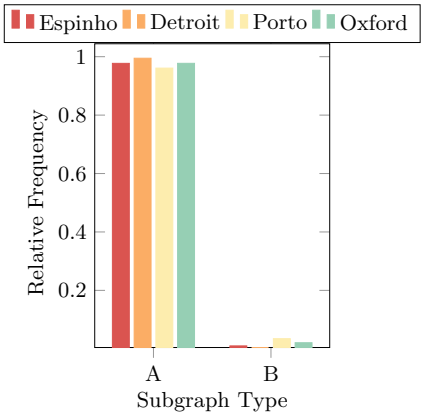


Fig. 11 Spatial subgraph fingerprint of each of the studied cities

Fig. 12 Purely topological subgraph fingerprint of each of the studied studies



them, using only this distribution. Conversely, using the data from Fig. 12 it is not possible to make that distinction, as all cities show a clear dominance of type *A* subgraphs with around the same difference in frequency when compared to type *B*, which is really uncommon in street networks. It is also interesting to note that the cities without a grid layout have a bigger frequency of other types of subgraphs other than classes 1 and 2, even if though those types are still by far the most relevant.

As a final note, we would like to remark that using a normal laptop the subgraph counting and labeling phase takes less than a minute to compute even in the largest considered network (Detroit, with 16,029 nodes and 24,773 edges). A potential drawback of our proposed strategy is that increasing the size k of the subgraph will inevitably lead to an exponential growth of the number of subgraph occurrences and hence on the execution time. However, in this paper we were mainly concerned with proving that the concept could be useful and there are still many improvements that can be made regarding efficiency.

5 Conclusions and Future Work

In this paper we present a set of contributions aiming to incorporate spatial properties into subgraph analysis. We first offer a novel abstraction that relies on a bounding box and regular spatial partitions to attribute node colors that describe the relative position of nodes within the subgraph. We then describe an implementation of a framework capable of discovering and counting these spatial subgraphs, based on enumerating occurrences and then discovering their type using a specialized canonical labeling mechanism. Finally, we provide a proof of concept experiment using real life data in which we show that our approach is able to go beyond classical topological motifs, capturing enough information to distinguish between different road network layouts.

We believe these are promising results that could lead into new insight on the characterization and comparison of network with spatial information. Our end goal is to be able to provide a universal spatial concept of network motifs that can be generally applicable to networks of any domain.

In order to further extend our work, we intend to study the incorporation of higher dimensional data, such as 3D brain networks, and we want to make an extensive evaluation of the role of the granularity in the information gained, by carefully analysing what happens when we use different amounts and sizes of spatial partitions. Furthermore, we want to study how changing the point of reference would impact the results (e.g. what happens to the patterns when we make arbitrary rotations?) and we intend to explore different symmetries and subgraph families that could provide classes that are invariant to spatial transformations (e.g. mirror symmetry). We also want to understand how we can assess statistical significance of the subgraph frequencies, by studying what could be appropriate spatial null models.

Finally, we want to improve the efficiency, not only by improving the exact counting computation, but also by trading accuracy for speed (e.g. using sampling) or using parallelism (e.g. using several threads in multicore machines).

Acknowledgements This work is partially financed financed by National Funds through the Portuguese funding agency, FCT—Fundação para a Ciência e a Tecnologia within project LA/P/0063/2020 and grant SFRH/BD/136525/2018. We would also like to thank the reviewers for the insightful comments.

References

1. Barthélemy, M.: Spatial networks. Phys. Rep. **499**(1–3), 1–101 (2011)
2. Barthelemy, M.: Morphogenesis of spatial networks. Springer (2018)
3. Boguñá, M., Krioukov, D., Almagro, P., Serrano, M.Á.: Small worlds and clustering in spatial networks. Phys. Rev. Res. **2**(2), 023040 (2020)
4. Choobdar, S., Ribeiro, P., Silva, F.: Motif mining in weighted networks. In: 2012 IEEE 12th International Conference on Data Mining Workshops, pp. 210–217. IEEE (2012)
5. Grohe, M., Schweitzer, P.: The graph isomorphism problem. Commun. ACM **63**(11), 128–134 (2020)
6. Haklay, M., Weber, P.: Openstreetmap: user-generated street maps. IEEE Pervasive Comput. **7**(4), 12–18 (2008)
7. Hasan, S., Schneider, C.M., Ukkusuri, S.V., González, M.C.: Spatiotemporal patterns of urban human mobility. J. Stat. Phys. **151**(1), 304–318 (2013)
8. McKay, B.D., Piperno, A.: Practical graph isomorphism, ii. J. Symbol. Comput. **60**, 94–112 (2014)
9. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., Alon, U.: Network motifs: simple building blocks of complex networks. Science **298**(5594), 824–827 (2002)
10. Narizuka, T., Yamamoto, K., Yamazaki, Y.: Statistical properties of position-dependent ball-passing networks in football games. Phys. A Stat. Mech. Appl. **412**, 157–168 (2014)
11. Newman, M.E.: The structure and function of complex networks. SIAM Rev. **45**(2), 167–256 (2003)
12. Paranjape, A., Benson, A.R., Leskovec, J.: Motifs in temporal networks. In: Proceedings of the tenth ACM International Conference on Web Search and Data Mining, pp. 601–610 (2017)
13. Pržulj, N.: Biological network comparison using graphlet degree distribution. Bioinformatics **23**(2), e177–e183 (2007)
14. Ribeiro, P., Paredes, P., Silva, M.E., Aparicio, D., Silva, F.: A survey on subgraph counting: concepts, algorithms, and applications to network motifs and graphlets. ACM Comput. Surv. (CSUR) **54**(2), 1–36 (2021)
15. Ribeiro, P., Silva, F.: Discovering colored network motifs. In: Complex Networks V, pp. 107–118. Springer (2014)
16. Sallmen, S., Nurmi, T., Kivelä, M.: Graphlets in multilayer networks. J. Complex Netw. **10**(2), cnac005 (2022)
17. Wernicke, S.: Efficient detection of network motifs. IEEE/ACM Trans. Comput. Biol. Bioinform. **3**(4), 347–359 (2006)

Improving the Characterization and Comparison of Football Players with Spatial Flow Motifs

Alberto Barbosa, Pedro Ribeiro, and Inês Dutra

Abstract Association Football is probably the world's most popular sport. Being able to characterise and compare football players is therefore a very important and impactful task. In this work we introduce spatial flow motifs as an extension of previous work on this problem, by incorporating both temporal and spatial information into the network analysis of football data. Our approach considers passing sequences and the role of the player in those sequences, complemented with the physical position of the field where the passes occurred. We provide experimental results of our proposed methodology on real-life event data from the Italian League, showing we can more accurately identify players when compared to using purely topological data.

Keywords Sports analytics · Subgraphs · Network motifs · Spatial data

1 Introduction

Association football, also known as soccer or simply football, is probably the world's most popular sport. It is therefore of no surprise that there has been an ever growing interest on collecting and analysing football data in order to inform players, coaches and management staff, trying to gain a competitive edge. Examples of related com-

A. Barbosa · P. Ribeiro (✉) · I. Dutra
Faculdade de Ciências, Departamento de Ciência de Computadores,
Universidade do Porto, R. Campo Alegre s/n, 4169-007 Porto, Portugal
e-mail: pribeiro@dcc.fc.up.pt

A. Barbosa
e-mail: alberto.barbosa@fc.up.pt

I. Dutra
e-mail: dutra@fc.up.pt

A. Barbosa · P. Ribeiro
CRACS & INESC-TEC, Porto, Portugal

I. Dutra
CINTESIS, Porto, Portugal

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_45

puter science research include a vast array of topics, such as team behaviour visualisation [16], talent discovery [3], injury forecasting [12], result prediction [11] or transfer market analysis [6].

In this work our focus is on providing a similarity metric for comparing football players in what concerns their role in the dynamics and passing behavior of the team. This is already useful as a rich characterization tool and could be further applied for instance to come up with suggestions of similar players in other teams that could potentially be good transfer targets.

Our main contribution is the concept of spatial flow motifs and a novel hybrid similarity metric, that incorporates both when and where passes occur in the game. We partition the football field into regions and we use temporal data to construct passing sequences that can be seen as small subgraphs where nodes are classified according to the region of the corresponding passing event. We also present experimental results on real life event data from the Italian League, showcasing how these spatial features can enrich and complement purely topological information, achieving a higher accuracy on the player comparison task.

2 Related Work

The amount of research work related to football analytics is too vast to be included in this paper [13, 14]. Here, we will mainly focus on research that delved into studying motif based patterns in passing networks.

A passing network can be seen as a graph where the nodes represent players and directed edges represent successful passes between two players. Milo et al. defined network motifs as "patterns of interconnections occurring in complex networks at numbers that are significantly higher than those in randomised networks" [7]. Later, Gyarmati et al. [4] defined flow motifs. Considering a passing sequence, a flow motif is a subsequence of the passes where labels represent distinct players without identity. In the context of this paper, all motifs are flow motifs and we will use both terms interchangeably. We will next make a short description of previous research on this topic.

The application of network motif methodology to football data has been a recent research topic among the fields of network science and sports analytics.

Using network motif methodology, Bekkers and Dabadghao [2] identified unique play styles for teams and players. Peña et al. [10] also applied network motif and clustering techniques to football data from the Premier League, La Liga and Champions League, concluding that Xavi Hernandez was the outlier in their analysis. Wiig et al. [15] use centrality measurements and PageRank to identify key passers and/or recipients in football teams.

Håland et al. [5] modelled sequences of passes as flow motifs and concluded that no connection between the ranking of a team and its distribution of flow motifs was clear.

Regarding team behaviour, Gyarmati et al. [4] present a quantitative method to characterise the passing behaviours of football teams, concluding that some unique styles of play do not consist of uncountable random passes but instead are finely structured.

In a previous work [1], we studied player similarity based on the topology of the passing networks. Flow motifs were extracted from sequences of passes that involved each player and the conclusion was that taking into account the specific position of the player in a motif, i.e. the orbit that represents a given player in a motif, was better in comparing players than just comparing the number and type of motifs they were a part of. Also, we provided a way to objectively measure the performance of the models generated, which is a great complement to the almost uniquely visual analysis that is made to evaluate player similarity algorithms.

Even though some work has been done in exploring the spatial dimension of football games [8], to the best of our knowledge there is no framework that entails the characterisation of the passing behaviour of a player through the study of spatial flow motifs.

3 Data Description

The data used in this work was retrieved from a public data set containing spatio-temporal events in association football [9]. We will be using event data from the 2017/2018 season of the top tier Italian league.

Events in the data were transformed into sequences of passes between different players. Players that did not participate in, at least, 80% of the games in the 2017/2018 season were not considered in our analysis. We excluded these players to guarantee that we had the most consistent and complete data as possible regarding the players we analysed.

4 Methodology

After pre-processing the data, the first step of our method was to extract spatial flow motifs from the generated passing sequences, as it is explained in this section. Our code is available at https://github.com/BertoBoss/SpAn.

In the context of this work, we will compute and count "flow motifs", as defined in [4], even when referring to them simply as motifs. We note however that flow motifs are not "network motifs" as classically defined in [7] since they do not take into account the statistical significance of each of the sequences extracted. This is mainly due to the lack of an appropriate null model for football spatial motifs, which could be an interesting future line of work.

We start by overlapping a grid over the space in which the events occurred. This grid will divide the space in m equal parts lengthwise and in n equal parts heightwise.

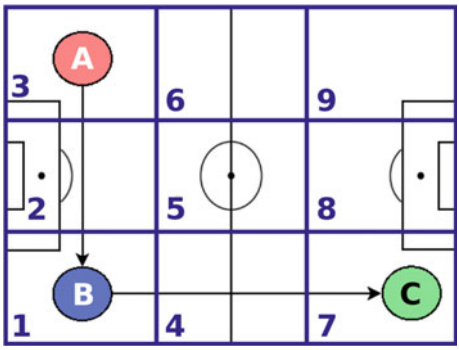


Fig. 1 Grid representing a passing sequence involving three players. First pass goes from player A to player B from cell 3 to cell 1. The second pass goes from player B to player C from cell 1 to cell 7. This sequence represents a 3A.1B.7C spatial flow motif

So there will be $m \times n$ different rectangles in which an event can occur. In this work, we will only consider the case when $m = n = 3$, so there will always be $n^2 = 9$ different rectangles. All different cells originated by the divisions of the grid will be numbered in ascending order bottom-up from left to right, the smallest number being in the bottom cell of the leftmost column and the highest number in the top cell of the rightmost column.

The insertion of each sequence of passes in the grid is done by inserting each pass according to the coordinates in which it started, i.e., the coordinates of the origin of the pass. The only exception is the last point in the sequence that represents where the last pass ended. This will yield a grid where each cell contains a non-negative number of points and each point represents where a given pass was made, except for the last one that represents where the final pass ended. The points are inserted by order of occurrence in the game they happened and preserving that order is important, since we are studying the evolution of a passing sequence between members of the same team. We do so by storing such information in an auxiliary list for each grid we generate. An example of a grid filled with a size 3 passing sequence is presented in Fig. 1.

Each of these generated grids now represents a spatial flow motif. For simplicity purposes, we decided to generate a string that represents each of these motifs in an unique way, so that each of these grids can be easily and uniquely identified by it. Such string needed to embody three crucial characteristics of these motifs: the topology, the relative order of occurrence of each pass and the space on the pitch where it occurred. So we transform a sequence into a string of the kind $K_1C_1.K_2C_2.\ldots.K_N.C_N$, where each K_i is an integer representing the cell of the grid in which the pass it represents occurred, C_i is a character representing the player that made that pass, consistent with the definition of a flow motif, and the order in which the passes occurred is preserved by the order of appearance on the string from left to right. For the passing sequence in Fig. 1, the corresponding string would be **3A.1B.7C**. For simplicity sake, whenever we want to make a reference to a single flow motif, we will use the string method and only present the visual aid if absolutely necessary.

For each player in our data set, we transformed every passing sequence he participated in into a spatial flow motif, using the methodology described above.

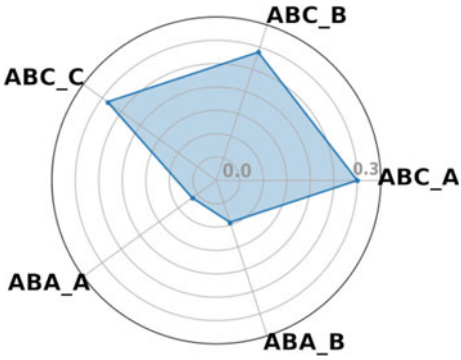


Fig. 2 Fingerprint of Jorginho passing behaviour with topological flow motifs

According to [1], the orbit a player occupies in each flow motif is very important in characterising the passing style of a player. In order to encapsulate that concept, we also consider the specific position a given player occupied in the passing sequence. Using the example in Fig. 1 and extending the concept of spatial flow motifs to include orbits, we would say that player A participated in a ABC_A motif topologically and in a $3A.1B.7C.A$ spatial flow motif. A similar extension can be made with respect to nodes B and C.

The spatial flow motif concept as we present it here can be seen as a good complementary analysis to the purely topological flow motif analysis. Now, we will not only be able to see which types of passing plays a player is involved in more often, but also the areas on the pitch in which those plays tend to occur.

We will now exemplify how the inclusion of spatial data in motif extraction can give us better insight regarding the passing behaviour of each player. Unfortunately, due to the quantity of players analysed, it is not possible to perform this analysis for each player in our data set. Nonetheless, we will briefly analyse the passing behaviour of Jorginho, not only topologically, but also spatially.

Regarding topology, we present in Fig. 2 a plot representing the relative frequency of the participations of Jorginho in different types of passing sequences. We can see that the player tends to be more involved in ABC types of plays (involving three different players) rather than in ABA (involving only two different players). Moreover, we can see that Jorginho seems to have a somewhat balanced participation among ABC plays, participating in similar quantities in all three possible roles of the play. This seems to go along with the intuition that a midfielder like Jorginho, being a playmaker, is involved in all stages of the offensive game of his team. Also, we can see that Jorginho, when participating in ABA motifs, tends to occupy the B position more often, which means that he tends to be the one that receives the pass from some player A and then passes the ball to that same player A.

Although the study of purely topological flow motifs proved to be a good way to characterise the passing behaviour of a player [1], more information can still be extracted from the same data if we look not only at the passing sequences' topology, but also at the spatial information encapsulated in that data, specially in football, where space is a very important aspect of the game.

To do that, we decided to study the distribution of spatial flow motifs grouped by their topology. We will consider a 3×3 division of the pitch, as presented in Fig. 1. From Table 1, we can actually perform a more detailed analysis of the passing behaviours of Jorginho, when compared to the analysis of the frequency of each topological flow motif.

First, one can see that the majority of the highlighted values are on cells that represent the midfield area (cells 4, 5 and 6). Given that Jorginho is a midfielder, it is not surprising to see that a big part of the events he will be involved in will take place on central areas of the pitch. This is however an intuition that we can only confirm when incorporating the space dimension of the data, since it is not possible to do so by only analysing the topological motifs as in Fig. 2.

Another important thing to notice is that the relative frequency of C in cells 7, 8 and 9 is lower in ABC_C flow motifs. This can be, at least partially, explained because he is a player that does not step into advanced areas of the pitch that much, given that he is not a forward, so when he is the last player on the passing sequence, that sequence tends to end in not so advanced areas of the pitch, like on cells 4, 5 or 6. This seems to be opposite to the behaviour when Jorginho occupies the A or B positions in ABC motifs, where position C tends to have higher frequency of appearance in cells 7, 8 or 9.

A noticeable aspect of the data in Table 1 is the fact that Jorginho apparently tends to impose some progression on the pitch with his passing. This becomes apparent by the fact that, when he makes a pass, the next player on the sequence tends to have a higher frequency of appearance in more advanced areas than Jorginho himself. For example, in ABC_B motifs, player A tends to be in cell 7 in higher frequencies than Jorginho (player B), which then passes the ball to a player C that, again, has

Table 1 Frequency of each player (A, B or C) on each cell on a 3×3 grid according to the different topological motif they participate in. Values are normalised by sub-column, meaning that, for example, in ABC_A motifs, player A (Jorginho) occupies a position in cell 1 0.051% of the times he participates in an ABC motif occupying position A. Highest values are on bold. Sub-columns representing Jorginho have a highlighted background

	ABC_A			ABC_B			ABC_C		
Cell	A	B	C	A	B	C	A	B	C
1	0.051	0.061	0.054	0.089	0.045	0.051	0.086	0.082	0.047
2	0.055	0.048	0.048	0.077	0.055	0.038	0.073	0.072	0.043
3	0.033	0.042	0.037	0.058	0.027	0.028	0.064	0.061	0.030
4	**0.299**	**0.251**	**0.238**	**0.245**	**0.307**	**0.220**	**0.244**	**0.264**	**0.297**
5	**0.287**	**0.189**	**0.141**	**0.142**	**0.279**	**0.197**	**0.128**	**0.131**	**0.279**
6	**0.158**	**0.133**	**0.129**	**0.160**	**0.159**	**0.112**	**0.148**	**0.157**	**0.150**
7	0.053	**0.155**	**0.176**	**0.126**	0.058	**0.197**	**0.147**	**0.123**	0.065
8	0.040	0.057	0.079	0.035	0.042	0.085	0.023	0.032	0.048
9	0.024	0.066	0.098	0.069	0.028	0.072	0.088	0.077	0.041

higher frequency in cell 7. This seems to indicate a forward passing bias by Jorginho, meaning that the players that receive the ball from him are often in more advanced positions of the pitch than himself.

On the most defensive areas of the pitch, we can see that the only significant change between the three types of spatial flow motifs is related to the position Jorginho occupies on the passing sequence: when he occupies a given position p, the frequencies of that position on more retreated areas of the pitch are lower for p and higher for the other positions. For example, when Jorginho is on position B on ABC_B motifs, the frequencies of a player being on cells 1, 2 or 3 and on the B position are lower than when he occupies either position A or position C.

It also stands out that there seems to be a slight bias for these plays to winding themselves more on the right side of the pitch than on the left. That becomes more evident when comparing the values of the frequencies on cell 7 (right attacking side) to the values in cell 9 (left attacking side) for ABC_A, ABC_B and ABC_C flow motifs. This can either represent a team behaviour that somehow favours attacking plays to happen on the right side of the pitch when Jorginho is involved in them or this can represent a bias imposed by Jorginho to force the team to play through the right, probably also influenced by the position he occupies on the pitch and the fact that it is more likely that he will pass the ball to players near him. It can also be both, since often players positions and characteristics are highly correlated to the team macro behaviour.

All this domain specific knowledge regarding a single player can be acquired only when we combine the topological information with the spatial information contained in the raw event data.

5 Results

With the analysis of the spatial flow motifs Jorginho is involved in, we intend to show that incorporating an extra layer of spatial information in flow motif counting can result in better and more accurate knowledge extraction from event data. However, when we want to objectively measure if new useful information can be extracted from the processing of spatial data, we need to setup an experimental environment that allows us to take valid conclusions about our methodology.

In an ideal world, we would be able to not only check if the addition of spatial information results in a good complement to the purely topological information that flow motifs naturally provide and have a way to measure how good such potential complement is.

We decided to build a similar experimental environment as the one on [1], but we have adapted it to also account for spatial flow motifs.

We first separate the matches in two different sets: one corresponding to the first half of the season and the other corresponding to the second half of the season (we also experimented a division into odd and even match weeks, but no relevant differences were found).

The idea behind this division is to be able to see how similar a player is to himself in different parts of the same season. Given that there is no ground truth in this domain, we believe that a good way to validate our approach is to exploit the idea that a player, of course with some exceptions, must have similar passing behaviour during the course of the same season. To achieve that, we designed a distance metric to measure the distance between two players, incorporating not only the topological difference between the different motifs that a player was involved in, but also the spatial dimension of the data.

After computing all the spatial flow motifs for every player on the dataset, we calculate the distance between each pair of players. The distance metric has two components: one incorporates the topological component of the flow motifs and the other incorporates the spatial dimension of the flow motifs.

The purely topological distance between two players is defined in Eq. 1, where $D_{top}(A, B)$ represents the distance between players A and B according to the purely topological flow motifs that A and B participate in. M is the set of all flow motifs of size 3 and A_m and B_m represent the normalised frequency (between 0 and 1) of player A and B on motif m, respectively.

$$D_{top}(A, B) = \sqrt{\sum_{m \in M} (A_m - B_m)^2} \tag{1}$$

A big part of adding the spatial dimension is the ability to measure how distant two different sequences are in the pitch. We use cell centroids in order to compute the distance between two different spatial flow motifs. The centroid of a cell k in a grid is a point whose coordinates are $(min_x + max_x/2, min_y + max_y/2)$, where min_x (min_y) is the minimum value of x (y) that a point in a cell k can have and max_x (max_y) is the maximum value of x (y) that a point in a cell k can have. Then we calculate the distance between two motifs m and n by calculating the Euclidean distance between the centroid of the cell in which the first pass occurred in motif m and the centroid of the cell in which the first pass occurred in motif n. We then add it to the distance between the centroid of the cell in which the second pass occurred in motif m and the centroid of the cell in which the second pass occurred in motif n, and so on, until the motif is fully processed. In the context of this paper, we will call this distance $D_{centroids}(m, n)$.

The component of our metric that encapsulates the spatial information of the flow motifs is given in Eq. 2. M represents the set of all flow motifs of size 3, M_k is a set of motifs that are topologically equivalent between themselves, m and n are two topologically equivalent spatial flow motifs, $D_{centroids}(m, n)$ calculates the Euclidean distance of the centroids of the cells in which the passing sequences occurred, $f_A(m)$ represents the frequency of player A in motif m and $f_B(n)$ represents the frequency of player B in motif n.

$$D_{space}(A, B) = \sum_{M_k \in M} \sum_{m,n \in M_k} D_{centroids}(m, n) * f_A(m) * f_B(n) \tag{2}$$

It is important to notice that Eq. 2 is a distance metric thought to complement a merely topological setting, by trying to encapsulate some spatial information that would otherwise be discarded by a purely topological approach. Since both $0 \leq f_A(m) \leq 1$ and $0 \leq f_B(n) \leq 1$, this allows for the weight of the value of $D_{centroids}(m, n)$ to be, in some sense, proportional to the frequency of occurrence of motifs m and n in players A and B, respectively.

One could argue that a different approach, like using $f_A(m) - f_B(n)$, would encapsulate better the idea that "the higher the difference between the frequencies, the higher the weight the distance would have". Even though the intuition is correct, using the difference between the frequencies of each motif when complementing purely topological motif analysis would not take into account the individual frequencies of $f_A(m)$ and $f_B(n)$ for players A and B, respectively. For example, whether $f_A(m) = f_B(n) = 0.1$ or $f_A(m) = f_B(n) = 0.9$ the value of $f_A(m) - f_B(n)$ would be 0. However, using $f_A(m) * f_B(n)$, the second assignment would result in a higher value, that better mirrors the fact that m and n are highly frequent motifs for players A and B, respectively.

The final distance metric is a weighted mean of the two components we approached, as in Eq. 3, where α is a constant between 0 and 1, influencing the weight of each component in the final distance metric.

$$Dist(A, B) = \alpha \times D_{space}(A, B) + (1 - \alpha) \times D_{top}(A, B) \tag{3}$$

Now that we have a distance metric that incorporates both topological and spatial information regarding flow motifs, we can use it to calculate the distance between players.

Let H_1 be the set of games that took place in the first half of the season and H_2 the set of games that occurred in the second half of the season. Also, let A_{H_i} be the set of spatial flow motifs that represent player A and were extracted from the set of games in H_i.

Our task will then be to calculate the distance $Dist(A, B)$ between every A_{H_1} and every B_{H_2}, i.e., we want to calculate the distance between each pair of players A and B such that the spatial flow motifs regarding player A occurred in games in the first half of the season and the spatial flow motifs regarding player B occurred in games belonging to the second half of the season.

Let L_A be a list of players such that the position j each player B occupies in the L_A represents that B is the j-th least distant player to A, according to the distance metric in Eq. 3.

So, for each player P_1 in A_{H_1}, our job is to compute L_{P_1} such that each player P_2 in L_{P_1} belongs to A_{H_2}. When all L_{P_i} are calculated for every player P_i in A_{H_1}, we can see how well our method characterises a player, in the sense that we just need to measure, for every player P_i, the position P_i occupies in L_{P_i}. The lower the position, the better, since we want P_i to have similar passing behaviour to himself.

In [1], an arbitrary threshold was defined, stating that cases where P_i occupied a position $j \leq 10$ it would be considered a positive case, and a negative otherwise.

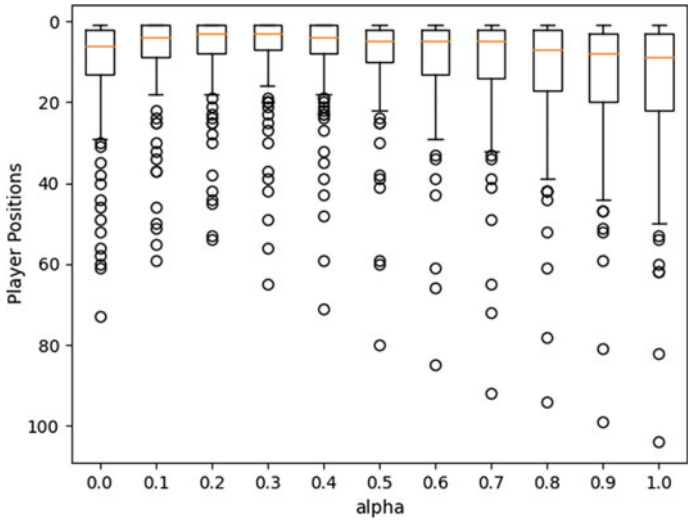


Fig. 3 Boxplot of different distributions generated by the variations in the constant α in $Dist(A, B)$. α varies from 0 to 1 in increments of 0.1

All positive and negative cases were counted and the evaluation model was simply given by the percentage of positive cases that the model got right.

In this work we decided to extend that idea to a more continuous analysis of the distribution of the positions that each P_i occupies in L_{P_i}. A boxplot of those distributions is presented in Fig. 3. Note that in that plot, the values of the constant α range from 0 to 1 in increments of 0.1, with the plot referring to $\alpha = 0$ is the one on the left and each successive plot represents a $+0.1$ increment.

Analysing the boxplot in Fig. 3, it is noticeable that, for some values of α, the distribution is more skewed toward smaller position values than on others. Those values are $0.1 \leq \alpha \leq 0.6$. The mean values for each of the distributions presented in Table 2 confirm that on those values, the mean and standard deviation of the distributions are smaller when compared to $\alpha = 0$, that represents a distribution based on a merely topological distance metric.

The mean value in the distributions presented in 3 represents the average position a player P_i is in L_{P_i}. This means that, for example, with $\alpha = 0.2$, any player P_i is on average the 7th most similar player to himself, which is a really good improvement when compared to the 11th position, on average, that yields from $\alpha = 0$. It is also worth noting that, for $\alpha \geq 0.7$, the distributions seem to exhibit a worst behaviour than considering only the topological nature of the data. That seems to indicate that, on its own, the metric we designed to complement a merely topological study, is not better at characterising the passing behaviour of football players than a classical topologically reliant distance metric.

In Fig. 4, we can see in more detail the curves representing the distributions for four different values of α. We can clearly see curves representing $\alpha = 0.2$ and $\alpha = 0.3$

Table 2 Mean and standard deviation values for each distribution in Fig. 3, according to the values of α

α	Mean	Standard deviation
0	10.98	14.46
0.1	7.78	11.47
0.2	7.15	10.46
0.3	7.26	10.88
0.4	7.84	11.29
0.5	8.80	12.17
0.6	9.88	12.90
0.7	11.13	14.13
0.8	12.4	14.95
0.9	13.71	15.91
1	15.32	17.31

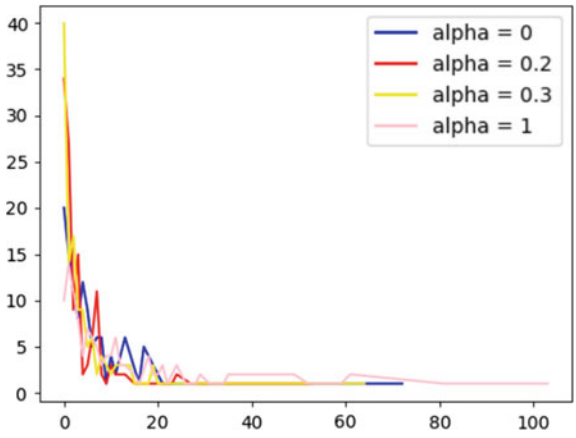


Fig. 4 Four different curve plots showcasing the different distributions caused by changing the value of α in the $Dist(A, B)$ distance metric

have a much more compact look in the smaller position values, specially when compared to the distribution when $\alpha = 1$. Another thing we notice is that with $\alpha = 0.2$ and $\alpha = 0.3$ we have smaller ranges of values in positions and their concentration is higher in smaller position numbers.

6 Conclusions and Future Work

In this work we extended the concept of flow motif to incorporate the spatial dimension of football event data.

We were able to improve the characterisation of the passing behaviour of a player by encapsulating the spatial nature of the data. The distributions that represented the similarities between a given player in different halves of the season were proof of that increase in the capability to characterise a football player passing behaviour, for some values of α. In the future work, we aim at improving the way we count spatial flow motifs, since this approach has proven to be too much time consuming. This can be done either through the conceiving of a parallel algorithm or through improvements on the way we calculate the distance metric (through the exclusion of not relevant spatial flow motifs).

It also seems that it is possible to generalise this approach in order for it to be applied in different domains. In a more formal note, we are building graphs where the nodes are coloured based on the position they occupy in a two dimensional grid and the directed edges incorporate a time dimension in the sense that if node v has an outgoing edge that connects him to node u, then the event that represents node v happened before node u. This means that in a spatio-temporal domain, similar methodology may be applied to count k sized subgraphs (flow motifs) of those spatio-temporal graphs.

Acknowledgements This work is partially financed financed by National Funds through the Portuguese funding agency, FCT—Fundação para a Ciência e a Tecnologia within project LA/P/0063/2020 and grant SFRH/BD/136525/2018. Also, we would like to thank the reviewers for their thoughtful insight.

References

1. Barbosa, A., Ribeiro, P., Dutra, I.: Similarity of football players using passing sequences. In: Brefeld, U., Davis, J., Van Haaren, J., Zimmermann, A. (eds.) Machine Learning and Data Mining for Sports Analytics, pp. 51–61. Springer International Publishing, Cham (2022)
2. Bekkers, J., Dabadghao, S.: Flow motifs in soccer: what can passing behavior tell us? J. Sports Anal. **5**(4), 299–311 (2019)
3. Fenner, J.S., Iga, J., Unnithan, V.: The evaluation of small-sided games as a talent identification tool in highly trained prepubertal soccer players. J. Sports Sci. **34**(20), 1983–1990 (2016)
4. Gyarmati, L., Kwak, H., Rodriguez, P.: Searching for a unique style in soccer (2014). arXiv:1409.0308
5. Håland, E.M., Wiig, A.S., Hvattum, L.M., Stålhane, M.: Evaluating the effectiveness of different network flow motifs in association football. J. Quant. Anal. Sports **16**, 311–323 (2020)
6. Matesanz, D., Holzmayer, F., Torgler, B., Schmidt, S.L., Ortega, G.J.: Transfer market activities and sportive performance in European first football leagues: a dynamic network approach. PLoS ONE **13** (2018)
7. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., Alon, U.: Network motifs: simple building blocks of complex networks. Science **298**(5594), 824–827 (2002)
8. Narizuka, T., Yamamoto, K., Yamazaki, Y.: Statistical properties of position-dependent ball-passing networks in football games. Phys. A Stat. Mech. Appl. **412**, 157–168 (2014)
9. Pappalardo, L., Cintia, P., Rossi, A., Massucco, E., Ferragina, P., Pedreschi, D., Giannotti, F.: A public data set of spatio-temporal match events in soccer competitions. Sci. Data **6**(1), 1–15 (2019)
10. Peña, J.L., Navarro, R.S.: Who can replace xavi? a passing motif analysis of football players (2015). arXiv:1506.07768

11. Razali, N., Mustapha, A., Utama, S., Din, R.: A review on football match outcome prediction using Bayesian networks. J. Phys. Conf. Seri. **1020**, 012004. IOP Publishing (2018)
12. Rossi, A., Pappalardo, L., Cintia, P., Iaia, F.M., Fernández, J., Medina, D.: Effective injury forecasting in soccer with gps training data and machine learning. PLoS ONE **13** (2018)
13. Thakkar, P., Shah, M.: An assessment of football through the lens of data science. Annals Data Sci. **8**(4), 823–836 (2021)
14. Tuyls, K., Omidshafiei, S., Muller, P., et al.: Game plan: what AI can do for football, and what football can do for AI. J. AI Res. **71**, 41–88 (2021)
15. Wiig, A.S., Håland, E.M., Stålhane, M., Hvattum, L.M.: Analyzing passing networks in association football based on the difficulty, risk, and potential of passes. Int. J. Comput. Sci. Sport **18**, 44–68 (2019)
16. Wu, Y., Xie, X., Wang, J., Deng, D., Liang, H., Zhang, H., Cheng, S., Chen, W.: Forvizor: visualizing spatio-temporal team formations in soccer. IEEE Trans. Visual. Comput. Graph. **25**, 65–75 (2019)

Dynamics on/of Networks

Bayesian Approach to Uncertainty Visualization of Heterogeneous Behaviors in Modeling Networked Anagram Games

Xueying Liu, Zhihao Hu, Xinwei Deng, and Chris J. Kuhlman

Abstract Heterogeneous player behaviors are commonly observed in games. It is important to quantify and visualize these heterogeneities in order to understand collective behaviors. Our work focuses on developing a Bayesian approach for uncertainty visualization in a model of networked anagram games. In these games, team members collectively form as many words as possible by sharing letters with their neighbors in a network. Heterogeneous player behaviors include great differences in numbers of words formed and the amount of cooperation among networked neighbors. Our Bayesian approach provides meaningful insights for inferring worst, average, and best player performance within behavioral clusters, overcoming previous model shortcomings. These inferences are integrated into a simulation framework to understand the implications of model uncertainty and players' heterogeneous behaviors.

Keywords Agent-based simulation · Interpretable inference · Models of heterogeneous behaviors · Networked data · Uncertainty visualization

X. Liu (✉) · Z. Hu · X. Deng
Virginia Tech, Blacksburg, VA 24061, USA
e-mail: xliu96@vt.edu

Z. Hu
e-mail: huzhihao@vt.edu

X. Deng
e-mail: xdeng@vt.edu

C. J. Kuhlman
University of Virginia, Charlottesville, VA 22904, USA
e-mail: hugo3751@gmail.com

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_46

1 Introduction

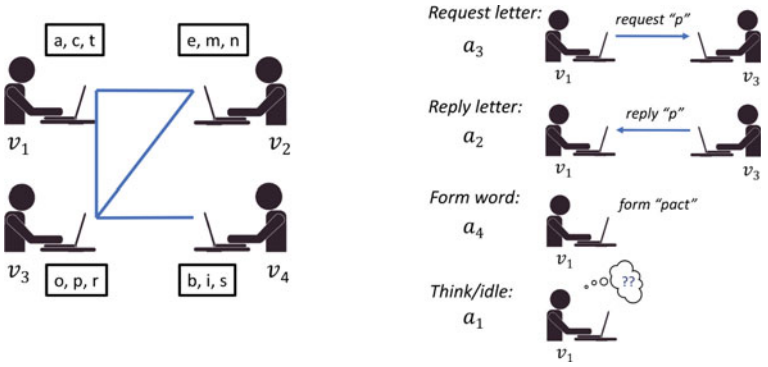
1.1 Background and Motivation

There are many variants of anagram games. Most involve either the unscrambling of letters to form a single unique word or finding as many words as possible from a collection of letters. Anagram games involving single individuals have been studied for over 50 years. As an early example from the 1960s, anagram games were used as priming activities to study anxiety [16], i.e., anagram games are played in a way to induce player anxiety. Also dating from the 1960s, these games have been studied in their own right, e.g., to assess the effects of letter rearrangement and word frequency on player performance [6].

Evaluation of group anagram games, where players cooperate to form words, is a much more recent phenomenon. A ground-breaking work in [2] used in-person group anagram games to prime people to form collective identity. The work in [1] performed *online* group anagram games by imposing a *network* on game players to control their interactions. Our focus is to model the games in [1], which we now overview.

The experimental networked group anagram game (NGrAG) setup is shown in Fig. 1b. Remote human subjects play the game through web browsers. Each player is provided three initial letters and over a 5-min game duration, players try to form as many words as possible *as a team*. Players split evenly the total earnings from the game, which is based on the number of words the team forms. Players cooperate by sharing their available letters with their distance-1 neighbors. When a player v_l shares a letter, she retains a copy of the letter; this is to motivate (mutual) assistance among players. Also, once a letter is acquired, it can be used any number of times in one word and can be used in any number of words (i.e., there is no mechanism or action by which a player loses a letter). Words must be at least three letters. A player is free to take any of the actions in Fig. 1b, any number of times and in any order. See [1] for additional details.

The network has at least three roles in NGrAGs, and these are intertwined with game player behavior (i.e., action) models. First, the network determines the number of neighbors (i.e., degree) of an (ego) player. Section 2.1 states how ego game player data are partitioned by degree in developing behavior models. Second, the letters assigned to those neighbors, along with those of the ego player, determine the words an ego player can form. Third, the behaviors of the neighbor nodes are influenced by their degrees—as for the ego player, in the first point—and hence these neighbors' interactions with the ego (e.g., requesting letters) are dependent on their local network structure.



(a) online game configuration (b) 4 possible player actions

Fig. 1 **a** Illustrative networked group anagram game (NGrAG) with four remote players (v_1 through v_4) and four communication channels (in blue). Players participate through their web browsers. A player's initially assigned letters are in boxes. **b** Four actions that may be taken by any player, at any time during the 5-min NGrAG. Actions can be repeated by a player any number of times. The action vector a is $a = (a_1, a_2, a_3, a_4)$, with a_i given in the graphic

1.2 Novelty and Contributions

Modeling of NGrAGs is an interesting and challenging task. A Bayesian modeling approach is described in [13]. Here, we focus on characterizing variability in player performance through Bayesian uncertainty visualization. Our contributions follow.

First, because of using posterior samples without asymptomatic distributions of model parameters (as in [9]), the proposed Bayesian uncertainty visualization can greatly alleviate the data scarcity issue in model estimation. Consequently, the obtained posterior samples of model parameters avoid extreme values that cause some transition probabilities π_{ij} to be 0 or 1. See Sects. 2 and 4 and Fig. 2.

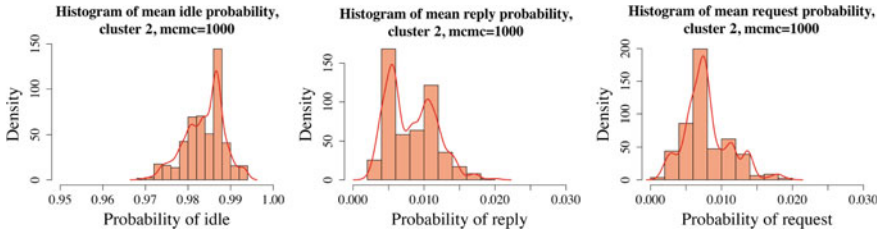


Fig. 2 The three histograms are for group 1, cluster 2, where the initial state is request (a_3). M-H algorithm is applied to draw 1000 $\boldsymbol{B}$ matrix samples after 1000 burn-in. The first histogram is for the probability of transitioning to idle (π_1), the second one is for the probability of transitioning to reply (π_2), and the third one is for the probability of request (π_3). The probability of forming words (π_4) is 0 because there is no forming words action a_4 in the training data. These data demonstrate that the extreme value problem is largely alleviated

Second, the proposed Bayesian uncertainty visualization is a first work to appropriately visualize the uncertainties in data and models for NGrAGs. Different from previous work [9], the proposed method emphasizes the visualization of uncertainty in a comprehensive manner using a two-dimensional bubble plot described in Sect. 3. Such a plot can reflect uncertainties of a player's activeness (i.e., non-idle actions). Moreover, the location, width, and height of bubbles represent the mean and standard deviations of the probabilities inferred from the posterior samples of model parameters. Plots of results from our experiments and analyses are in Sect. 4.

Third, uncertainty visualization enhances agent-based modeling and simulation (ABMS) of these games. We can naturally identify, interpret, and model worst, average, and best categories of player performance, even within one cluster of player behavior. We embed these interpretable inferences within a simulation platform. We demonstrate these effects by simulating NGrAGs that go beyond the conditions for which experiments were conducted (Sect. 5). The network of each simulated game is fixed, consistent with the experiments being modeled.

Our last contribution is broader in scope. The proposed Bayesian uncertainty visualization greatly enhances the explainability of uncertainty quantification for NGrAGs. For a complex system such as a NGrAG, in contrast to "one-shot" games where a game player only makes one binary yes/no decision in a game, quantifying uncertainty for model and data needs to be properly visualized in order to gain meaningful insights. It is precisely this need that motivated this work, which is an outgrowth of the work in [9]. The proposed method can be a good exemplar to achieve such an objective, and can be applied for visualizing uncertainty in other networked applications. Specific works are provided in Related Work Sect. 1.3.

1.3 Related Work

Modeling of network games and data. There are multiple works [9, 13] related to the proposed method. In [13], a Bayesian model of human behavior in anagram games is presented. However, that work does not address methods to identify worst, average, and best behaviors within a Bayesian context. A process of identifying best, average, and worst behaviors within behavioral clusters is presented in [9]. However, that anagram model for determining a player's next action in a game is based on asymptotic normal distributions for the primary behavioral matrix $\boldsymbol{B}$ (presented below), rather than on posterior distributions, as we do here. Most importantly, neither of those works address uncertainty visualization, as in this work. Other games incorporate multiple player actions over time, e.g., [11, 17]. These games, like ours, use fixed networks. Other types of network models, for other phenomena, use evolving networks, e.g., [5].

Bayesian visualization and uncertainty visualization. Visualization is a vital tool for data analysis to describe uncertainties in data [3, 4]. The effective visualization of uncertainty is commonly recognized as a challenging task [10, 14]. Potter et al. [15]

presented a summary of the state-of-the-art techniques in uncertainty visualization, including comparison techniques, attribute modification, and image discontinuity. Gabry et al. [7] illustrated the role of visualization in exploratory data analysis in the context of a Bayesian workflow. House et al. [8] developed Bayesian visual analytics (BaVA) to justify Bayesian sequential update of parameters. Our work aims to visualize uncertainty in a Bayesian framework to effectively and accurately identify the uncertainty in the data and heterogeneous behaviors of players.

2 State Transition Model and Extension

2.1 State Transition Model

Agent-based models (ABMs) for the NGrAG represent the game as a discrete-time stochastic process. That is, at each time step, a player can transition to one of the four states (actions a_1, a_2, a_3, and a_4); see Fig. 1b. In our previous work [13], a Bayesian clustering-based UQ framework is developed as follows. Based on statistical analysis of the game data, we first partitioned the players into two groups: those with less than three neighbors (group $g = 1$) and those with three or more neighbors (group $g = 2$). Then we defined two variables x_e (for engagement) and x_w (for forming words), where x_e is the sum of the number of requests and the number of replies that a player sends and x_w is the number of words a player forms in a game. Based on these two standardized variables, we applied the Dirichlet process (DP)-based Bayesian clustering approach [12] with a specific penalty parameter λ ($\lambda = 2.5$) such that when a point is farther than λ away from every existing cluster center, a new cluster will be formed with this point in it. In this way, we further partitioned the players in the same group into four clusters where those within the same cluster have similar activity levels in the game. For data in each cluster, player behaviors in a game are modeled using the multinomial logistic regression with four predictors shown in Table 1:

$$\pi_{ij} = \exp(\mathbf{z}^T \boldsymbol{\beta}_j^{(i)}) / \sum_{m=1}^{l} \exp(\mathbf{z}^T \boldsymbol{\beta}_m^{(i)}), \quad j = 1, \ldots, l, \tag{1}$$

where

- $l = 4$ since we consider four actions a_1, a_2, a_3, and a_4.
- π_{ij} is the probability of the player, who took action a_i at time t, taking action a_j at time $t + 1$.
- $\mathbf{z} = (1, Z_B(t), Z_L(t), Z_W(t), Z_C(t))^T$ is the predictor vector; Table 1.
- $\boldsymbol{\beta}_j^{(i)} = (\beta_{j1}^{(i)}, \ldots, \beta_{j5}^{(i)})^T$ is the regression coefficient parameter vector.

For a given action a_i at time t, the parameters can be expressed as a matrix $\boldsymbol{B}^{(i)} = (\boldsymbol{\beta}_1^{(i)}, \ldots, \boldsymbol{\beta}_l^{(i)})_{l\times(l+1)}^T$ for $i = 1, \ldots, 4$. Thus, from the game network struc-

Table 1 The four temporal variables of players in the NGrAG and model

Variable	Description
$Z_B(t)$	Size of buffer of letter requests that player v has yet to reply to at time t
$Z_L(t)$	Number of letters that v has available to use at t to form words
$Z_W(t)$	Number of valid words that v has formed up to t
$Z_C(t)$	Number of consecutive time steps that v has taken the same action

ture (which determines a node's degree and hence its group g) and the performance cluster c determined for a node/player ($c = 1$, 2, 3, or 4), a particular model based on Eq. (1), with parameter matrix $\boldsymbol{B}^{(i)}$, is assigned to a game player to predict the probability of next actions.

2.2 Motivation for Model Extension

With the Bayesian approach, we can obtain the posterior distribution for the parameter matrix $\boldsymbol{B}^{(i)}$. One then can quantify the uncertainty of parameters by conducting posterior inference. Markov chain Monte Carlo (MCMC) methods are commonly used to obtain samples from the posterior distribution. Posterior inference can then be conducted empirically. In order to quantify the heterogeneous behavior of players *within* a cluster, our strategy is to identify the parameter matrices that generate the most active behavior, the least active behavior, and the average behavior in terms of probability of being non-idle. Integrating these different levels of performance into the simulation of NGrAG will better capture the heterogeneity among players because we can assign players these different behaviors. These considerations lead to the new uncertainty visualization method in Sect. 3.

3 Bayesian Uncertainty Visualization Method

This section details the proposed Bayesian uncertainty visualization method. The goals are to visualize the uncertainty within and between clusters and to identify the heterogeneous (i.e., worst, average, and best) behaviors of players within each cluster.

For each observation in the training data, one can obtain the corresponding predictor vector $\mathbf{z}$. To directly identify the activity level, we transform the parameter matrix $\boldsymbol{B}^{(i)}$ to a probability vector. In each cluster, players with the same initial action a_i share the same parameter matrix $\boldsymbol{B}^{(i)}$. Thus for these players, without loss of generality, we omit i in $\boldsymbol{B}^{(i)}$ and π_{ij} to get a parameter matrix $\boldsymbol{B}$ and a probability vector $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_4)$ containing the probabilities of the next action using Eq. (1).

Therefore, for a $\boldsymbol{B}$ matrix, a training data set of n observations that have same initial action can generate n probability vectors. The mean of these probability vectors and the corresponding standard error can be obtained. Given a sequence of $\boldsymbol{B}$ matrices, we can compute a sequence of mean probability vectors and their standard errors. To visualize the uncertainty among these mean probability vectors, we create a bubble plot where the center of each bubble represents the mean probability vector for a parameter matrix, with the width to be $2 \times SE(\bar{\pi}_4^r)$ and the height to be $2 \times SE(\bar{\pi}_1^r)$. Using such a plot, it is easy to quantify the activity levels within a cluster and identify the worst, average, and best behaviors. The probability of forming words (π_4) in the probability vector represents the players' ability to form words and the probability of not being idle ($1 - \pi_1$) indicates the players' level of activity in the game—a small to-idle probability π_1 suggests a high activity level.

We summarize the proposed method of uncertainty quantification within a cluster as follows. First, we use Metropolis-Hasting (M-H) algorithm to get R random samples of $\boldsymbol{B}_r$ ($r = 1, \ldots, R$) from the posterior distribution after a burn-in period (taken to be 1000). Second, for each $\boldsymbol{B}_r$, we apply the size n training data to Eq. (1) to produce n probability vectors $\hat{\boldsymbol{\pi}}^{r,l} = (\hat{\pi}_1^{r,l}, \hat{\pi}_2^{r,l}, \hat{\pi}_3^{r,l}, \hat{\pi}_4^{r,l})$, $l = 1, \ldots, n$. Then the mean probability vector and its standard error are calculated:

$$\bar{\boldsymbol{\pi}}^r = \frac{1}{n} \sum_{l=1}^{n} \hat{\boldsymbol{\pi}}^{r,l} = (\bar{\pi}_1^r, \bar{\pi}_2^r, \bar{\pi}_3^r, \bar{\pi}_4^r)^T,$$

$$SE(\bar{\boldsymbol{\pi}}^r) = \frac{1}{n} \sqrt{\sum_{l=1}^{n} (\hat{\boldsymbol{\pi}}^{r,l} - \bar{\boldsymbol{\pi}}^r)^2} = (SE(\bar{\pi}_1^r), SE(\bar{\pi}_2^r), SE(\bar{\pi}_3^r), SE(\bar{\pi}_4^r))^T.$$

Third, one can draw a bubble plot of $1 - \bar{\pi}_1$ against $\bar{\pi}_4$, with one bubble for each $\boldsymbol{B}_r$, $r = 1, \ldots, R$. Each bubble is an ellipse centered at the mean probability $(\bar{\pi}_4^r, 1 - \bar{\pi}_1^r)$ with width $2 \times SE(\bar{\pi}_4^r)$ and height $2 \times SE(\bar{\pi}_1^r)$. Fourth, note that a low mean to-idle probability ($\bar{\pi}_1$) suggests a high engagement level, and a player with a high mean to-idle probability is less active. Accordingly, we select the $\boldsymbol{B}_r$ matrix with the maximum $\bar{\pi}_1^r$ as the matrix of the worst behavior, and the one with the minimum $\bar{\pi}_1^r$ as the matrix of the best behavior. The $\boldsymbol{B}_r$ matrix of the average behavior is one that produces the mean of $\bar{\pi}_1^r$, $r = 1, \ldots, R$.

A key advantage of this proposed method is that we can visually analyze the uncertainty among data. In the bubble plot, it is easy to find the best and the worst behavior and view the heterogeneous behaviors within each cluster. Moreover, the size of the bubble can help us visually detect the variability among the observations. One can also quantitatively compare the activity ranges of clusters with players that have the same number of neighbors to further discover the differences between clusters within the same group ($g = 1$ or 2). Note that this visualized uncertainty quantification was not contained in the previous work [9].

Another advantage is that our Bayesian method alleviates the extreme value problem caused by data scarcity in the previous model [9]. When the size of the training data in each category is unbalanced (e.g., 556 observations have final state idle while only 4 observations have final state reply (a_2) and request (a_3) in group $g = 1$ cluster $c = 2$ with initial state a_3), the asymptotic normal distribution of $\boldsymbol{B}$ would have a very large variance. Thus, the estimated parameter in $\boldsymbol{B}$ can be unexpectedly large and cause an extreme value in the probability vector $\boldsymbol{\pi}$ and an infinite loop in state transitions in the ABM. However, the memorylessness property of MCMC can avoid this problem since every sample is only generated based on the previous one. For this reason, the Bayesian approach avoids the extreme scenarios of players' actions.

4 Visualization of Heterogeneous Behaviors

This section investigates uncertainties within clusters, heterogeneous behaviors, and differences in activity levels between clusters, using the game data and the models of Sects. 2 and 3. Under the Bayesian setting, for each initial state in a cluster, 1000 samples of $\boldsymbol{B}$ matrices are drawn using the M-H algorithm after 1000 burn-in. Figure 2 reports the histograms of probabilities for the aforementioned group 1, cluster 2 with initial state being request (a_3). It is seen that the Bayesian uncertainty quantification methods can alleviate extreme value problems (by producing probabilities away from 0 and 1) caused by data scarcity.

The bubble plots for group 1, cluster 1 and cluster 4, with the initial state being idle (a_1) are presented in Fig. 3a, b, respectively. Clearly, there is uncertainty within the clusters. Moreover, the size of the bubble reflects the variability in the observations and the color reflects the replications. The darker the bubble, the more samples have this transition probability. It is seen in each plot that samples are more gathered at the maximize-a-posterior (MAP) estimation (the blue bubble) and the standard error of to-word (transition) probability is larger than that of to-idle probability in most cases.

Figure 4 contains bubble plots of mean probability for initial state idle (a_1) in the top row, and for initial state reply (a_2) in the bottom row. When the initial state is idle, Fig. 4a, b show that four clusters are well separated and the activity level is ascending, supporting the rationality of clustering players by behavior. It is also seen that group 2 is more active than group 1 with a larger probability of forming words and being non-idle, on a per cluster basis. This corresponds to our assumption that players with more neighbors will be more active in the NGrAG. When the initial state is reply (second row of plots), there are no data points of forming words in the training data of group 1. Thus, the mean probabilities of forming words and the corresponding standard errors will be zero. Consequently, we compare the activity level only based on the probability of being non-idle as shown in Fig. 4c.

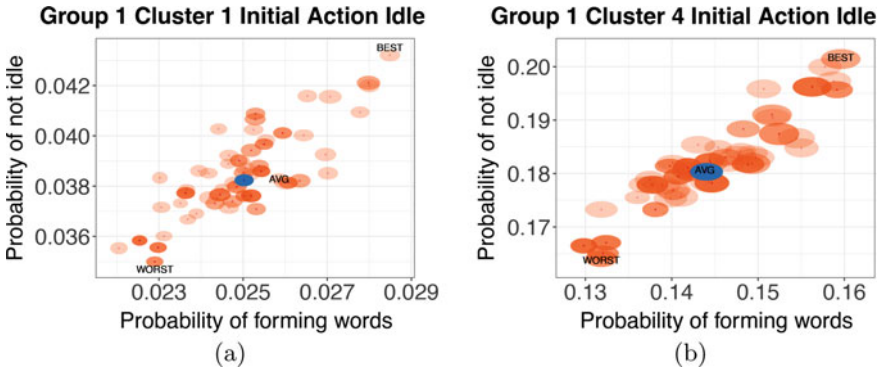


Fig. 3 **a** Bubble plot of group 1, cluster 1, where initial state is idle (a_1). **b** Bubble plot of group 1, cluster 4, where initial state is idle (a_1). Each bubble is a ellipse centered at the mean probability $(\bar{\pi}_4^r, 1 - \bar{\pi}_1^r)$ with width $2 \times SE(\bar{\pi}_4^r)$ and height $2 \times SE(\bar{\pi}_1^r)$, where $SE(\bar{\pi}_4^r)$ and $SE(\bar{\pi}_1^r)$ are standard errors of mean to-word probability and mean to-idle probability, respectively. The blue bubbles are the MAP results. The worst, average, and best performance bubbles are noted

5 Agent-Based Simulations of Networked Anagram Games and Results

In this section, we build ABMs and run them in a software framework to simulate the NGrAG. For particular input conditions and models, we provide results for individual players (also referred to as nodes or agents) and for aggregated totals over all players.

5.1 Simulation Process

ABMs are designed and constructed from the models of Sects. 2 and 3. Inputs to simulations are as follows. The network of Fig. 5 represents the possible interactions among the seven game players. It contains players in groups $g = 1$ and 2. Each player is provided four letters, and the letters are purposely specified to enable players to form words, e.g., one player v_2 is given letters $\{i, l, m, n\}$ and neighboring players are given complementary letters, e.g., v_3 is assigned letters $\{o, p, r, s\}$. Owing to space considerations, we examine two clusters: cluster $c = 3$ for $g = 1$ and cluster $c = 2$ for $g = 2$. With these clusters, we then execute the worst, average, and best behavior models that are produced and evaluated in Sects. 3 and 4. Note that our results illustrate differences among worst, average, and best models, but the results shown are not the largest differences that exist across all $[g, c]$ pairs. This is to emphasize that the differences that we observe from the simulations are pervasive across all model parameters; an expanded version will address the full range of results.

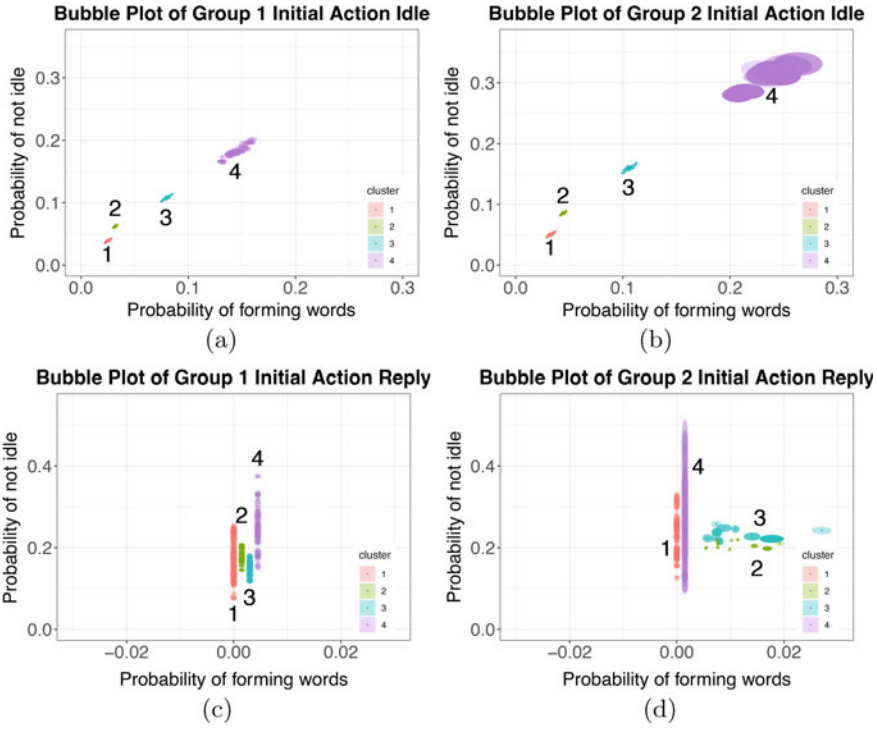


Fig. 4 **a** Bubble plot for group 1 where initial action is idle (a_1). The bubbles for cluster 1 and cluster 4 are shown in Fig. 3a, b, respectively. **b** Bubble plot for group 2 where initial action is idle (a_1). **c** Bubble plot for group 1 where initial action is reply (a_2). Note that the probabilities of forming words are 0 for all clusters. We assign different values for bubbles in different clusters to avoid overlapping. **d** Bubble plot for group 2 where initial action is reply (a_2). Note that the probabilities of forming words are 0 for cluster 1 and cluster 4. We assign a different value for bubbles in cluster 4 to avoid overlapping

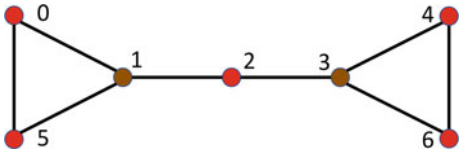


Fig. 5 Seven node (agent) game network on which simulations are run. Red (resp., brown) nodes are low (resp., high) degree nodes of degree $d = 2$ (resp., $d = 3$). Therefore, red (resp., brown) nodes are in group $g = 1$ (resp., $g = 2$)

One simulation is comprised of 100 iterations or runs. Each iteration is a complete simulation of one NGrAG, from time $t = 0$ to $t = 300$ seconds where players request letters from neighbors, reply to neighbor letter requests, and form words, as in the experiments. Our time step is one second, justified by the fact that players do not take

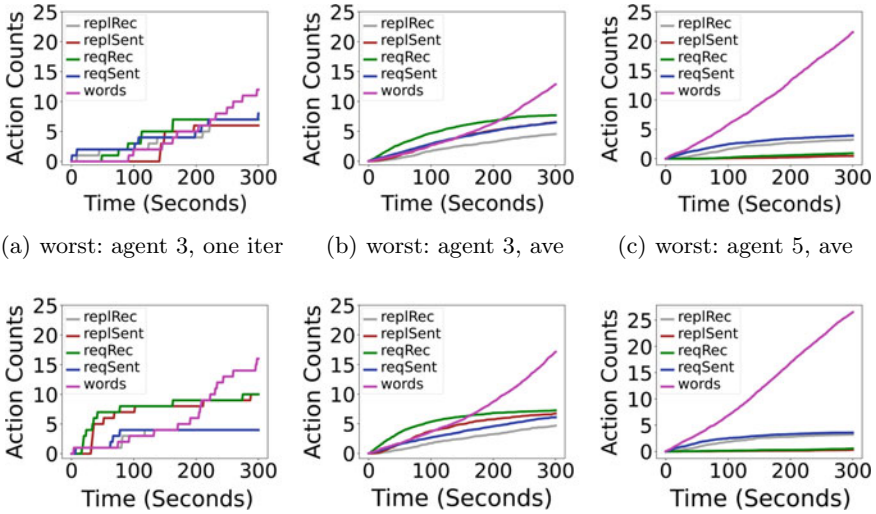
successive actions among request letter, reply to letter request, and form word within one second in the online experiments. Indeed, actions at time steps are mostly idle or thinking. The difference among iterations within a simulation is the stochasticity of the models in producing probabilities of actions π_{ij} of players at each t. Consequently, when we refer to "average" results below, we mean time point-wise averages across the 100 iterations, unless specified otherwise. Finally, we use the "worst," "average," and "best" models in the results below. In a simulation, all players use the same model, so that, for example, if we state that one player is represented by the best model, then all players in that simulation were assigned the best model.

5.2 Visualization of Simulation Results

Figure 6 provides results for a degree $d = 3$ player (agent 3) and a $d = 2$ player (agent 5) from the game setup in Fig. 5. The $[g, c]$ values are given in the caption of Fig. 6. Data in the first row of plots were generated with the worst behavior models for the respective $[g, c]$ pairs. In Fig. 6a, the time histories of actions are given for one of the 100 iterations: number of replies received (rerplRec), of replies sent (replSent), of requests received (reqRec), of requests sent (reqSent), and of words formed (words). The stair-stepped nature of the curves is due to the fact that these curves are from one iteration and so the plotted actions are discrete. In Fig. 6b, the curves correspond to the same action histories, but are smoother because they represent the time point-wise average of all 100 iterations. These first two plots are for agent 3. Figure 6c depicts corresponding average data for player 5. Note that a player with fewer neighbors (player 5) forms more words than a player with a greater number of neighbors (player 3). This is because player 5's behavior is from cluster 3 of group 1, while player 3's behavior is from cluster 2 of group 2; e.g., see Fig. 4a, b and the x-axis values for the two clusters. This again demonstrates the efficacy of identifying heterogeneous behaviors of players.

Figures 6d through f provide the corresponding plots to those in the first row, but now the results are for the best model. In Fig. 6e for player 3, the number of words is greater than that for the worst model, although the numbers of sharing actions are about the same. This same comparison holds for player 5 in Fig. 6 versus Fig. 6c.

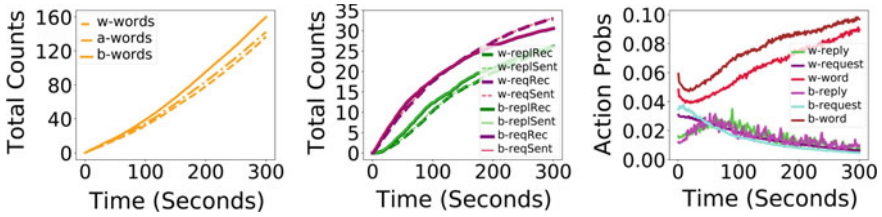
Figure 7 contains aggregate data over all seven game players. Time histories of the total number of words formed for the worst, average, and best models are given in Fig. 7a; the numbers of words increase in this order of the models. Figure 7b provides similar data for the sharing actions of requests and replies, but now only for the worst (dashed curves) and best (solid curves) models. Now, the worst and best model results overlap, with no clear-cut better behavior. This is partially a consequence of the fact that numbers of requests and replies are bounded by the number of a player's neighbors and the number of letters per player. Figure 7c shows the time point-wise *average* probability over all players of taking each action. These probabilities reflect the action counts in the previous two plots.



(a) worst: agent 3, one iter (b) worst: agent 3, ave (c) worst: agent 5, ave

(d) best: agent 3, one iter (e) best: agent 3, ave (f) best: agent 5, ave

Fig. 6 Results of anagram simulations with seven players. Data in plots **a** through **c** are for the *worst* behavior model. The curves are game time histories of counts of actions over the 300 s game. **a** Action histories for agent 3 in one iteration, **b** average action histories for agent 3, and **c** average action histories for agent 5. Data in plots **d** through **f** are the respective plots for the *best* behavior model



(a) sum of words formed (b) sum of requests & replies (c) action probabilities

Fig. 7 Aggregate simulation results across all seven nodes (agents) in a NGrAG. **a** Sum of words formed by all players in time for worst (w-words), average (a-words), and best (b-words) behavior models. **b** Sum of requests and replies across all players. Dashed curves correspond to worst behavior model (prefix "w-" in legend) and solid curves correspond to best behavior model (prefix "b-" in legend). **c** Average action probabilities across all seven players in a game, in time, for replying to letter requests (reply), requesting letters (request), and forming words (word)

6 Summary

This work presents a Bayesian uncertainty visualization method of complicated multi-player game data. Our step-by-step procedures have been applied to a networked group anagram game, where players cooperate to share letters and form words. These visualizations can effectively assist in assessing model uncertainties,

and in improving the interpretable inference of player behaviors. Software modules of these models are used to simulate the game for conditions beyond the experiments.

References

1. Cedeno, V., Hu, Z., et al.: Networked experiments and modeling for producing collective identity in a group of human subjects using an iterative abduction framework. Soc. Netw. Anal. Min. (SNAM), 43 (2020)
2. Charness, G., Cobo-Reyes, R., et al.: Identities, selection, and contributions in a public-goods game. Games Econ. Beh. (2014)
3. Chen, C.H., Härdle, W.K., Unwin, A.: Handbook of data visualization. Springer Science & Business Media (2007)
4. Cook, D., Lee, E.K., Majumder, M.: Data visualization and statistical graphics in big data analysis. Annual Rev. Stat. Appl. **3**, 133–159 (2016)
5. Dankulov, M.M., Melnik, R., Tadić, B.: The dynamics of meaningful social interactions and the emergence of collective knowledge. Sci. Rep. **5**, 12197-1–12197-10 (2015)
6. Dominowski, R.L.: Anagram solving as a function of letter moves. J. Verbal Learn. Verbal Beh. **5**, 107–111 (1966)
7. Gabry, J., Simpson, D., Vehtari, A., Betancourt, M., Gelman, A.: Visualization in Bayesian workflow. J. R. Stat. Soc. Ser. A (Stat. Soc.) **182**(2), 389–402 (2019)
8. House, L., Leman, S., Han, C.: Bayesian visual analytics: Bava. Stat. Anal. Data Mining ASA Data Sci J **8**(1), 1–13 (2015)
9. Hu, Z., Deng, X., Kuhlman, C.J.: Versatile uncertainty quantification of contrastive behaviors for modeling networked anagram games. In: Complex Networks (2021)
10. Johnson, C.R., Sanderson, A.R.: A next step: visualizing errors and uncertainty. IEEE Comput. Graph. Appl. **23**(5), 6–10 (2003)
11. Kearns, M., Suri, S., Montfort, N.: An experimental study of the coloring problem on human subject networks. Science **313**(5788), 824–827 (2006)
12. Kulis, B., Jordan, M.: Revisiting k-means: New algorithms via Bayesian nonparametrics. In: ICML (2012)
13. Liu, X., Hu, Z., Deng, X., Kuhlman, C.J.: A Bayesian uncertainty quantification approach for agent-based modeling of networked anagram games. In: WSC (2022)
14. Pang, A.T., Wittenbrink, C.M., Lodha, S.K., et al.: Approaches to uncertainty visualization. Visual Comput. **13**(8), 370–390 (1997)
15. Potter, K., Rosen, P., Johnson, C.R.: From quantification to visualization: a taxonomy of uncertainty visualization approaches. In: IFIP Working Conference on Uncertainty Quantification, pp. 226–249. Springer (2011)
16. Russell, D.G., Sarason, I.G.: Test anxiety, sex, and experimental conditions in relation to anagram solution. J. Personal. Social Psychol. 493–496 (1965)
17. Unakafov, A.M., Schultze, T., et al.: Emergence and suppression of cooperation by action visibility in transparent games. PLoS Comput. Biol. **16**(1), e1007588 (2020)

Understanding the Inter-Enterprise Competitive Relationship Based on the Link Prediction Method: Experience from Z-Park

Jiayue Yang, Lizhi Xing, and Guoqiang Liang

Abstract Integrating complex network theory, link prediction theory, and related research on industrial competition relationship, this paper proposes the theoretically analytical framework of the competitive relationship among Z-Park high-tech enterprises. By constructing a link prediction model, we reveal the internal dynamics that affect the evolution of the competitive network of enterprises, seek the best index reflecting the network formation mechanism, and apply it to the prediction of potential competitive associations.

Keywords Competitive relationship · ENMON model · Link prediction · Structural similarity index · Evolutionary mechanism

1 Introduction

The industrial complex network is a complex system that is constantly evolving. The dynamic changes of the network are mainly manifested in the entry and exit of enterprises in the market, the emergence or disappearance of competition and cooperation between enterprises, and so on. For promoting regional economic development, it is not only necessary to understand the topology characteristics of the network, but also pay attention to the interaction trend between nodes at the micro-level, then formulate the industrial layout adjustment and structure upgrading strategies, which happens to coincide with the problem solved by link prediction.

Link prediction includes both the prediction of unknown links and the prediction of future links, so it has extensive practical application value and important theoretical guiding significance. The research results of Liu [1] and Liu [2] have proved the effectiveness of link prediction in explaining the structure as well as evolution

J. Yang · L. Xing · G. Liang (✉)
College of Economics and Management, Beijing University of Technology, Beijing, China
e-mail: lianggq@bjut.edu.cn

J. Yang
e-mail: koken@bjut.edu.cn

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_47

mechanism of network, and the factors affecting the network evolution. According to the accuracy of link prediction results, we can discover effective indices to measure the importance of nodes or suitable properties to characterize nodes. Therefore, link prediction provides a scientific quantitative method for analyzing the network evolution mechanism.

Link prediction has been proved to be applicable and effective in explaining network structure characteristics [3–5] and analyzing network evolution mechanism [6–12], and has become a powerful tool for revealing value transmission and fluctuation in complex systems. However, the practice of link prediction method in the research of industrial organizations, especially the research direction of enterprise competition network is still blank. Therefore, based on complex industrial network, this paper will apply link prediction to the study of evolution problems in enterprise competition network.

2 Data and Modelling

This paper selects Zhongguancun Science Park (Z-Park), known as "China's Silicon Valley", as the research object. The changes of Z-Park's industrial structure over the past few decades reflect the direction of China's industrial structure upgrading.

2.1 Data Description

According to the Z-Park Management Committee's "Z-Park High-tech Enterprise Directory" (hereinafter referred to as the "Enterprise Directory"), as of November 2021, there were 23,034 high-tech enterprises in the Z-Park. Specifically, Table 1 shows the distribution of parks and technical fields to which high-tech enterprises belong.

Table 1 The parks and technical fields of high-tech enterprises in Z-park

Category	Category details
Affiliated park	Dongcheng, Xicheng, Chaoyang, Haidian, Fengtai, Shijingshan, Mentougou, Fangshan, Tongzhou, Shunyi, Daxing-Yizhuang, Changping, Pinggu, Huairou, Miyun, Yanqing
Technical field	Electronics and information, advanced manufacturing technology, new energy and energy-efficient technologies, environmental protection, new materials and application, bioengineering and new medicine, modern agriculture and breeding of new animal and plant varieties, aerospace technology, marine engineering, nuclear application, other

Data source "Beijing Municipal Science and Technology Commission, Administrative Commission of Zhongguancun Science Park" official website

According to the above-mentioned enterprise directory, we collected the basic information and development status of high-tech enterprises in Z-Park through tianyancha.com. On tianyancha.com, the "company background" column contains the company's establishment time, and the "company development" column contains the company's "competitive product information" item, which describes the information of companies with the same or similar products and services as the given company.

Before using Python software to obtain the competitive product data of Z-Park's high-tech enterprises, we first obtained the historical names of the renamed companies in the enterprise directory. Through sorting, it is found that on the tianyancha.com, 22,955 of the 23,034 high-tech enterprises in Z-Park can be obtained the company's establishment time and competitive product information, and the availability rate is as high as 99.657%. This can prove the effectiveness and accuracy of the data acquisition strategy.

The statistical results of the year of establishment of the enterprise (see Fig. 1) show that the earliest high-tech enterprise in Z-Park—Beijing BBEF electronics group Co., Ltd. was established in 1950, but in the following 30 years, the growth rate of high-tech enterprises was very slow. Until 1980, the growth of the number of enterprises began to show an upward trend, entered a stage of rapid growth after 2014, and reached a peak in 2016, with an increase of 2,230 enterprises compared to the previous year. After 2016, the growth of high-tech enterprises still maintained a high speed, but the number of new enterprises gradually declined. Affected by the outbreak of the new crown epidemic in 2020 and the global economic downturn, the number of new enterprises in 2021 has become the largest in the past 24 years. The lowest value, only increased by 171.

From the park level (see Fig. 2), Haidian Park, Chaoyang Park, and Changping Park have the highest concentration of high-tech enterprises, contributing half or

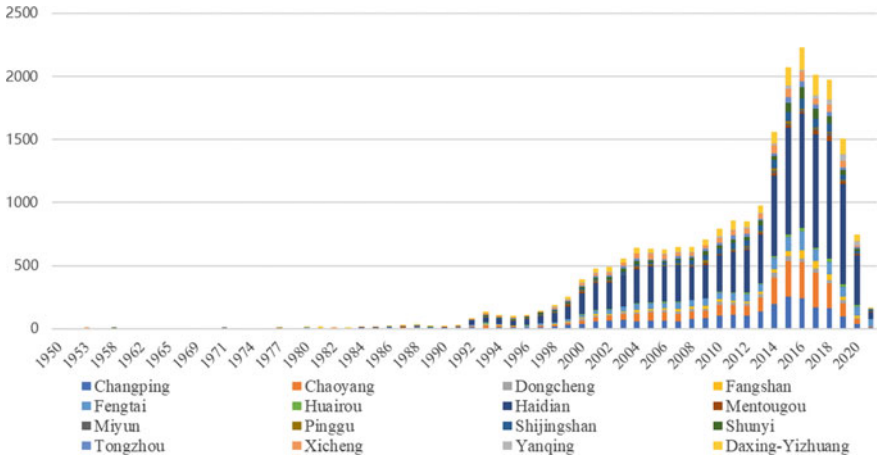


Fig. 1 The number of new enterprises in each park in different years

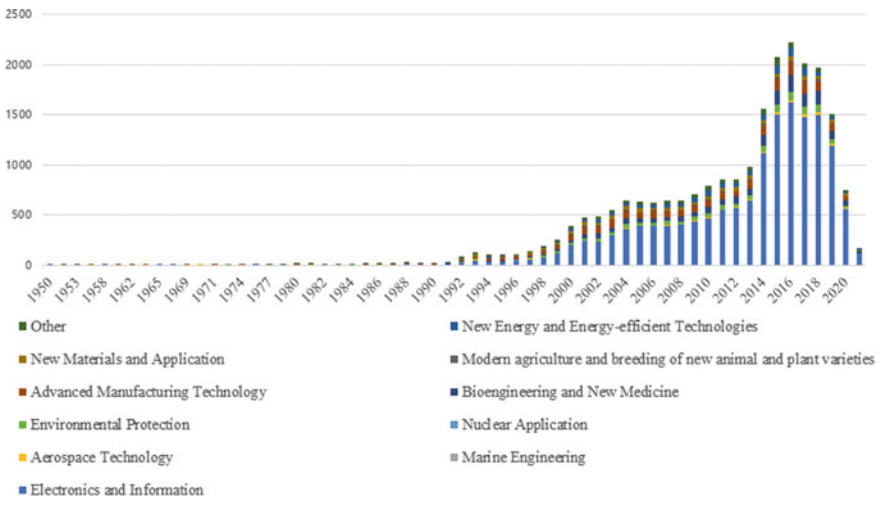


Fig. 2 The number of new enterprises in each technology field in different years

more of the new enterprises. Among them, in 2019 and 2020, the number of new enterprises in Haidian District accounted for more than half of all new enterprises in the Z-Park. The growth of Fengtai Park and Daxing-Yizhuang Park in the early stage was not significant, and the expansion strength of the two began to emerge after 2014. From the perspective of the technical field, the Electronics and Information field covers more than 60% of new enterprises and is an important support for high-tech enterprises, followed by Bioengineering and New Medicine, and Advanced Manufacturing Technology. Taking 2019 as an example, The newly added enterprises of the three accounted for 89.302% of all the newly added enterprises in the Z-Park.

2.2 ENMON Model

This paper endeavors to construct an Enterprises Niche Markets Overlap Network (ENMON) based on the analysis paradigm of graph theory. The network sets innovative entities in a specific area as nodes, the niche overlapping relationship (that is, providing similar products and services) between them as edges, forming a weightless undirected graph $G = (V, E)$ consisting of node set $\overleftrightarrow{V}$ and edge set $\overleftrightarrow{E}$. The specific construction principles are as follows [13]:

(1) Determine the node set of the network $\overleftrightarrow{V} = \{v_i\}$. v_i is composed of all the enterprises in Z-Park and their competing enterprises, where $i \in \{1, 2, \cdots, n\}$ and $n \leq 23034$, the number of nodes is recorded as $\overleftrightarrow{n} = \left|\overleftrightarrow{V}\right|$. Among them, this paper regards the main body that does not compete with any other enterprise as an isolated node and will not be reserved in the model.

(2) Determine the edge set of the network $\overleftrightarrow{E} = \{e_{ij}\}$. e_{ij} represents the competitive relationship between innovative entities in the park. As the relationship is bidirectional, e_{ij} and e_{ji} bear the same meaning. In particular, if the competing enterprises of the Z-Park's enterprises are located outside the park, the competitive relationship will not be considered.
(3) Determine the competitive relationship correlation matrix $A = \{a_{ij}\}$. Since the number of overlapping products or services between enterprises cannot be obtained on the tianyancha.com, that is, the intensity of competition between enterprises cannot be quantitatively described, so it is a_{ij} only used to characterize the existence of a competitive relationship. Specifically, if enterprise i and enterprise j are competing enterprises, then $a_{ij} = 1$, otherwise $a_{ij} = 0$.
(4) Build an enterprise competition network model. Taking the competition relationship matrix as the adjacency matrix, if the element of the matrix between two nodes is 1, there is an undirected edge e_{ij} between v_i and v_j, otherwise there is no connecting edge between the two.

According to the description of the data above, the establishment of the company spans from 1950 to 2021. Due to the small number of enterprises in Z-Park before 1980, there is almost no competition between enterprises. Compared with 2020, the number of enterprises in 2021 increases insignificantly. Therefore, this paper takes 1980 as the starting year and 2020 as the ending year, and divides the original network into 9 sub-networks with a 5-year time window to carry out a series of analyses on the structural evolution of the enterprise competition network.

By establishing a network model, it is not difficult to find that in the sub-network before 2000, the number of enterprises in the park was limited and the distribution areas were scattered, and the competitive network pattern was not yet obvious, while in the sub-network after 2005, the scale of nodes increased rapidly, the overall connectivity of the network is strong, and a large number of clump structures appear. Therefore, this paper takes 2000 and 2005 as examples (see Fig. 3) to show the topology of the network model, and uses the color and size of the node to distinguish the degree of the node, and the enterprise serial number as the node name.

3 Methodology

When making link prediction based on the similarity of nodes, it is commonly assumed that the higher the degree of similarity between two nodes, the greater the possibility of establishing new connections between them. The degree of similarity between two nodes can be described in various ways, such as node attribute information, node historical behavior information, and network structure information. Among them, the node attribute information brings higher prediction accuracy, but it is accompanied by the difficulty of acquisition. This is because the behaviors such as information concealment of nodes will lead to false information or even information loss. Moreover, large-scale data is often mixed with a lot of noise, and

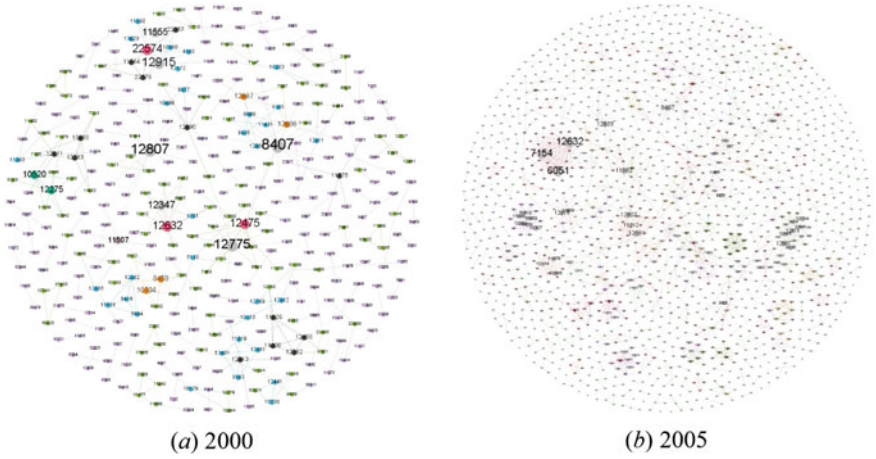


Fig. 3 ENMON model

high time costs wasted to determine which information is useful. Compared with it, network structure information has advantages in terms of availability, accuracy, and effectiveness. Accordingly, this paper takes this kind of indices as the selection range.

The similarity defined based on network structural information is called structural similarity. When the similarity index can essentially capture the structural characteristics or reflect the internal evolution mechanism of the network, it will show higher prediction accuracy. For link prediction, this section selected six indices from three types of existing structural similarity indices, namely local similarity indices, global similarity indices, and quasi-global similarity indices. The reason for choosing them is that they perform well in a large number of existing studies and are suitable for enterprise competition network models with large nodes [13]. The six indices are shown in Table 2.

4 Results and Discussion

To ensure the feasibility of the prediction and the comparability of the results, the sample selected in this paper for link prediction is the ENMON models of five years in 2000, 2005, 2010, 2015, and 2020, they are all typical unweighted and undirected networks. When using the six structural similarity indices for link prediction, it is necessary to divide the existing edge set in the network into a test set and a training set. The random sampling method can ensure that the probability of each edge being selected into the test set is equal, thus it is used as the data set division method in this paper. At the same time, the division ratio is set to 1:9, that is, 10% of the connected edges are selected as the test set, and the remaining 90% of the

Table 2 Structural similarity indices

Index	Equation	Description
Common neighbors (CN)	$S_{ij} = \lvert\Gamma(i) \cap \Gamma(j)\rvert$	Two of the non-connected nodes tend to be connected if they have a lot of common neighbors
Adamic-Adar (AA)	$S_{ij} = \sum_{x \in \lvert\Gamma(i) \cap \Gamma(j)\rvert} \frac{1}{\log k(x)}$	The common neighbor nodes with a small degree will contribute more than those with a larger degree
Resource allocation (RA)	$S_{ij} = \sum_{x \in \lvert\Gamma(i) \cap \Gamma(j)\rvert} \frac{1}{k(x)}$	No directly connected nodes i and j in the network, some resources can be allocated from node i to j, during which their common neighbors will become the transmission medium
Preferential attachment (PA)	$S_{ij} = k(i)k(j)$	The probability of the new edge connecting nodes i and j is directly proportional to the product of the degree of the two nodes
Local path (LP)	$S = A^2 + \alpha A^3$	Considering the potential influence of the three-order path range based on the CN index, we can define the LP index
Average commute time (ACT)	$S_{ij}^{ACT} = \frac{1}{E(i,j)+E(j,i)}$	The smaller the ACT of the two nodes is, the closer the two nodes will be

connected edges are used as the training set. In addition, this paper uses AUC as the accuracy evaluation index of the link prediction algorithm, sets the number of random sampling to 10,000 times, conducts 10 experiments independently, and finally takes the arithmetic average of AUC of multiple experiments as the evaluation index. The prediction process is realized by Matlab R2021b software.

4.1 Link Prediction Results

Table 3 shows the prediction accuracy of the 6 algorithms in the ENMON models from 2000 to 2020, where the weight calculated in the second step of the LP index is set to 0.5, that is, the influence of the third-order path length is reduced to 50%. It is commonly believed that the closer the AUC value is to 1, the higher the accuracy of link prediction, but no specific threshold is set to determine whether the prediction is effective. As shown in Ref. [14], eight real networks in different fields were predicted, and the average AUC obtained in different networks was about 0.819 [15], which we can use as a standard to test the accuracy of the prediction model. In general, taking the average of link prediction accuracy in different years, the two lowest prediction accuracy are 0.716 and 0.788 respectively, and the rest prediction accuracy

Table 3 Link prediction accuracy of ENMON models

Years	AA	ACT	CN	LP	PA	RA
2000	0.845	0.201	0.843	0.994	0.711	0.846
2005	0.940	0.739	0.941	0.995	0.805	0.940
2010	0.954	0.848	0.951	0.991	0.823	0.956
2015	0.959	0.886	0.961	0.991	0.802	0.959
2020	0.964	0.906	0.963	0.992	0.799	0.964
Mean	0.932	0.716	0.932	0.992	0.788	0.933

is above 0.9, indicating the application of the link prediction method in the ENMON is effective. On the one hand, it has the validity of the prediction accuracy, and on the other hand, it can distinguish the prediction results of different indices. Results of link prediction are as shown in Table 3.

From the performance of different indices, the LP index has the highest prediction accuracy, and the difference in network scale has no significant impact on its results. Conversely, the ACT index has the lowest prediction accuracy, and the results with large errors appeared when the network scale was small. The results of the RA index, the AA index, and the CN index are very close. More specifically, the performances of the RA index and the AA index is better than the CN index slightly. Meanwhile, the prediction accuracy of the PA index and the ACT index is poor, lower than the threshold standard, and their AUC values fluctuate slightly under different network scales, indicating that they are less applicable in the ENMON.

4.2 Evolutionary Mechanism

From the performance of the link prediction algorithm, we can divide the selected six indices into four categories: the first category is the LP index and the CN index, the latter is a special case of the former, that is, the adjustment coefficient of the LP index is 0. The second category is the RA index and the AA index, the difference between them is that the RA index is more sensitive to network heterogeneity, so it is more accurate than the AA index in the case of higher average degree centrality. The third category is The ACT index, which measures the distance of the information transmission path between nodes from two directions. The fourth category is the PA index, which is related to a certain degree of importance of the nodes themselves, but has nothing to do with the paths between them. Next, we will discuss the internal dynamics of the formation of the competitive institution among high-tech enterprises, as reflected in the results of the six indices.

4.2.1 Industrial Convergence Intensifies the Competition Pattern

According to the results of the CN index and the LP index, what affects the relationship between enterprises is not only the direct competition, but also may come from a wider industrial chain level. This development path is reflected in the trend of industrial convergence. The impact of this trend on enterprise competition is mainly reflected in two aspects: First, industrial convergence intensifies the existing competition situation, because a mutual substitution relationship is generated in products that originally belong to different markets. Second, industrial convergence attracts more competitors, which urges enterprises to achieve industrial upgrading through mergers across industries and regions.

High-tech industries have the characteristics of high industrial growth and are related to other industries closely. Therefore, the process of industrial convergence not only affects the competition situation in a single field but also has a profound chain reaction, which will ultimately affect the innovative development of high-tech zones.

4.2.2 Enterprises Scramble to Seize the Information Highland

Structural hole theory holds that a third party who connects the two without a direct connection has a natural information advantage. The role of the RA index and the AA index is to find the node pairs with less common neighbors and give them a higher possibility of connection. In the ENMON, it means that if the relationship between the enterprise and its upstream or downstream enterprises is very weak, then other enterprises will establish a direct relationship across them. These "structural hole" enterprises can discover and seize new market opportunities promptly. Moreover, they can improve their brand influence in the process of information exchange continuously, to expand market share and gain a monopoly market advantage.

The RA index and the CN index have high prediction accuracy in ENMON, indicating that striving for information and control advantages are important factors to stimulate the competitiveness of enterprises. Under this circumstance, software and technical service enterprises with information and data processing capabilities often become the best in the competitive network and hold a leading position for a long time.

4.2.3 Powerful Alliances to Alleviate Excessive Competition

The PA index shown in the ENMON is that the newly added enterprises tend to establish connections with the most influential enterprises. Professional development brings higher influence to some enterprises, and it is more likely to establish a competitive relationship between them, that is, showing a pattern of “strong competition”. However, in the ENMON model, the performance of the PA index is poor, which just shows that there is an opposite force, the “strong alliance” is at play. The

cooperation-competition theory believes that blind competition will bring about a vicious cycle and lead to both losses, only both competition and cooperation can achieve mutual benefit and win–win results. In Z-Park, various industrial fields have formed industrial clusters with a certain scale. A new profit point will often lead to the pursuit of many enterprises, and finally intensifies the competition within the cluster. However, under the influence of the market mechanism, various competitors will cooperate in technology and form a harmonious competition environment.

4.3 Prediction of Potential Competition

Link prediction includes two aspects: the prediction of unknown links and the prediction of future links. This paper mainly focuses on the latter, which is to predict the possibility of establishing a future connection between two nodes in the network that have not yet been associated with the known network topology information. Through the tests of accuracy and stability, the link prediction method is proved to have good accuracy and feasibility in the ENMON model. Referring to previous studies using link prediction to discover potential trade relationships or corporate partners, this paper will apply the best indices in the latest year's corporate competition network to quantitatively evaluate the similarity between a group of firms that have not yet been linked, to provide a long-term reference for the future competition situation of Z-Park.

Using the LP index with the best prediction effect to predict the ENMON model in 2020, the similarity matrix based on the LP algorithm can be obtained. Each element in the matrix represents the relationship between any pair of unconnected nodes in the network. The degree of similarity can be used to measure the probability that they will be connected in the future.

$$Sim_{LP} = \begin{bmatrix} Sim_{11} & \dots & Sim_{1N} \\ \vdots & \dots & \vdots \\ Sim_{N1} & \dots & Sim_{NN} \end{bmatrix} \tag{1}$$

Among them, N is the number of nodes in the network, and Sim_{LP} represents the similarity matrix obtained based on the LP algorithm. After removing the diagonal elements of the matrix, arrange the similarity values between any pairs of nodes in descending order, and then take the 7 pairs of nodes with the highest similarity as the potential enterprise combinations that are most likely to have competitive associations in the future, as shown in Table 4.

Through the similarity ranking of node combinations, two obvious features can be observed: (1) The similarity of node combinations is highly heterogeneous. Among the 656,424 pairs of nodes with potential competitive associations, the largest similarity value is 788, from the "Kyland Technology—NJA Information", while the "Tuoremei Medical—KanCare Nutrition" with the smallest similarity is only 0.5.

Table 4 Potential competitors of ENMON

Serial number	Enterprise—enterprise combinations	Similarity value
1	Kyland Technology—NJA Information	788
2	Easted Information—Jrunion	702.5
3	Beijing United Information—NJA Information	698.5
4	NJA Information—LongRuan Technologies	685
5	Jrunion—Capital Online	673.5
6	UltraPower—Transtrue Technology	672
7	Kyland Technology—UltraPower	648.5

Therefore, the similarity calculated based on the LP index is highly discriminative for the possibility that different nodes will link in the future, and the top-ranked combinations are far more likely to establish a relationship in the future than others. (2) The corporate entities in the potential competitive combinations are highly overlapping. Among the top 20 enterprise combinations, there are 14 related to Jrunion, Kyland Technology, Easted Information, and UltraPower, 4 of them are the combinations of the above enterprises, which means that the industrial transformation and upgrading of Z-Park has a strong dependence on software and information technology services. These popular enterprises will face a more intense competitive environment in the following years, and they will also have more business overlap with each other.

5 Conclusion

At present, Z-Park's high-tech enterprises occupy 10% of the country's total, and Z-Park has become an important scientific and technological innovation center in China. Based on the exploration of its dynamic mechanism and the prediction of its development prospects, the following two policy suggestions are proposed:

(1) Strengthen the characteristics and advantages of the data intelligence industry. The electronics and information industries, especially software and information technology service enterprises are in a dominant position in Z-Park. The number of such enterprises is the largest and the competition is fierce. The digital transformation and intelligent upgrading of the industry play an indispensable intermediary role. To promote the development of data intelligence industry clusters, the managers should improve the service system for the transformation of scientific and technological achievements, cultivate high-level innovative talents, and increase financial support.
(2) Guide the transformation of enterprise competition and cooperative relationship. According to the research on the innovation ecosystem, in the industrial technology innovation ecosystem, whether the relationship between different

innovation groups is competitive or symbiotic is affected by many factors. Therefore, in the ENMON model, competition and cooperation can be transformed into each other under the influence of different influencing factors. On the one hand, park managers should guide the implementation of a moderate competition mechanism. The park should adopt a free market competition policy as a whole, and formulate regulatory policies for market entry in specific areas based on industry characteristics and existing competition intensity. On the other hand, an enterprise collaboration mechanism within the park and across the park should be created. The managers should establish professional associations in conjunction with industry leaders and other outstanding enterprises to fully understand the resources and development status of the industry, and formulate scientific guidance for the cooperation of enterprises in the industry.

Acknowledgements This research was funded by the National Natural Science Foundation of China (Grant No. 71971006) and 2021 High-level Technology Innovation Think Tank Youth Project (Project No. 2021ZZZLFZB1207016).

References

1. Liu, H.K., Lv, L.Y., Zhou, T.: Uncovering the network evolution mechanism by link prediction. Sci. Sin: Phys. Mech. Astron. **7**(41), 816–823 (2011). (in Chinese)
2. Liu, Z., Zhang, Q.M., Lv, L.Y., Zhou, T.: Link prediction in complex networks: a local naïve Bayes model. EPL **96**(4), 48007 (2011)
3. Guan, Q., An, H.Z., Gao, X.Y.: Estimating potential trade links in the international crude oil trade: a link prediction approach. Energy **102**, 406–415 (2016)
4. Feng, S., Li, H.J., Qi, Y.B., Guan, Q., Wen, S.B.: Who will build new trade relations? Finding potential relations in international liquefied natural gas trade. Energy **141**, 1226–1238 (2017)
5. Lu, Z.G., Chen, Q.: Link prediction of enterprise cooperation relationship in dynamic supply chain network. Comput. Eng. Appl. **58**(2), 9 (2022)
6. Xing, L.Z.: Study on industry transfer path in the process of collaborative development of Beijing, Tianjin and Hebei in the perspective of link prediction. Sci. Technol. Prog. Policy. **34**(004), 54–59 (2017)
7. Wang, B., Wang, W.P., Fei, W.Y.: Study of the prediction model of industrial network based on the dynamical links. J. Syst. Eng. **33**(06), 721–731 (2018)
8. Ma, J.Y.: Potential Trade Relationship of International Copper Resources Based on Link Prediction Method. China University of Geosciences, Beijing (2018)
9. Liu, S., Dong, Z.: Who will trade bauxite with whom? Finding potential links through link prediction. Resour. Policy **63**, 101417 (2019)
10. Li, B., Ding, K., Sun, X.L.: Predicting potential technology partners and competitors of enterprises: a case study on fuel cell technology. J. China. Soc. Sci. Tech. Inform. **40**(10), 1043–1051 (2021)
11. Wang, J.J., Liu, J.G., Li, Z.K.: Research on enterprise partnership in supply chain based on complex network. J. Syst. Sci. **29**(03), 110–115+130 (2021)
12. Zhang, X.L., Wang, J.J.: On the evolution cooperation mechanism of energy supply chain networks under link prediction. CAAI T. Intel. Syst. **12**(02), 221–228 (2017)
13. Xing, L.Z., Han, Y., Xu, J.Y.: Analyzing the co-competition mechanism of high-tech park from the perspective of complex socioeconomic network. Entropy-Switz. **23**, 978 (2021)

14. Xing, L.Z., Han, Y.: Finding the worldwide industrial transfer pattern under the perspective of econophysics. In: 11th International Conference on Complex Networks, CompleNet 2020 (2020)
15. Lv, L.Y., Zhou, T.: Link Prediction. Higher Education Press, Beijing (2013)

Analyzing Configuration Transitions Associated with Higher-Order Link Occurrences in Networks of Cooking Ingredients

Koudai Fujisawa, Masahito Kumano, and Masahiro Kimura

Abstract Time-varying higher-order interactions among more than two units are typically modeled as a temporal higher-order network in the framework of simplicial complex. For the temporal evolution of the higher-order structure of human proximity interactions in five different social settings, previous work found the characteristics of the configuration transitions in a temporal higher-order network before and after triplet interaction events occur. Recently, food science and computing have been attracting attention due to the increasing popularity of recipe sharing services in social media. In this paper, aiming to reveal the characteristics of the temporal evolution of homemade recipes in terms of combinations of ingredients, we propose a method of analyzing the configuration transitions in a temporal higher-order network of ingredients before and after new higher-order links are formed, in the framework of temporal simplicial complex by extending the previous work from a perspective of activity degree of configuration. Using real data of a Japanese recipe sharing service, we empirically demonstrate the effectiveness of the proposed method, and apply it to analyzing the dynamical properties of higher-order networks of ingredients for Japanese homemade recipes.

Keywords Higher-order interaction · Simplicial complex · Temporal network analysis

1 Introduction

By capturing pairwise interactions between units, complex network science has provided new insights into a variety of fields such as biology, ecology and sociology [2]. In the framework of ordinary dyadic-network (i.e., simple graph), a node (or vertex) stands for the unit of an underlying system and a link (or edge) represents the pairwise interaction between units. However, many real-world systems are very involved with

K. Fujisawa · M. Kumano · M. Kimura (✉)
Faculty of Advanced Science and Technology, Ryukoku University, Otsu, Japan
e-mail: kimura@rins.ryukoku.ac.jp

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_48

higher-order interactions among more than two units. Examples include coauthorship, email sending, human social interaction and neuronal stimulus. A *simplicial complex* [6, 13] is a typical model of higher-order network that can represent interactions among any number of units, and this approach has been successfully applied to several problem domains such as brain organization [14], protein interaction [7] and social influence spreading [8]. Moreover, Benson et al. [3] have explored the temporal evolution of simplicial complexes derived for 19 datasets from various domains in terms of *simplicial closure*, which is a distinctive phenomenon of higher-order link creation and cannot be captured by traditional network analysis methods.

Recently, Cencetti et al. [5] have investigated the temporal evolution of the higher-order structure of human proximity interactions in five different social settings in terms of temporal simplicial complex. Unlike link prediction tasks such as simplicial closure, they in particular addressed higher-order interactions involving exactly three individuals (i.e., triplet interaction events), and examined the transition rates for the configurations of three individuals before and after a triplet interaction event. Here, such configurations can be classified into the following four cases. For the three people, there are no pairwise interactions in the first case, there is a single pairwise interaction in the second case, and there are two pairwise interactions in the third case. Also, in the fourth case, there is a larger interaction including the three people. Then, they empirically showed that the temporal evolution towards and from such a social triplet interaction event is characterized by the above second and third cases [5]. However, very few attempts have been made at analyzing these characteristics for the case of more than three interactions and other domains. Moreover, the configuration transition analysis by Cencetti et al. [5] entirely ignored an idea of *activity degree of configuraion* before and after a higher-order interaction event, i.e., the activity degree of each configuration compared to its ordinary occurrence rate.

With the advent of social media for cooking recipes, a large amount of data on homemade recipes is becoming available, offering an opportunity to investigate food practices of ordinary people. Thus, attention has recently been devoted to food science and computing [12], and combinations of ingredients appearing in recipes have been analyzed from the point of view of complex network science [1, 9, 16]. However, those studies only focused on pairwise relationships between ingredients and analyzed their static properties.

Aiming to reveal the characteristics of the temporal evolution of homemade recipes in terms of combinations of ingredients, we investigate a higher-order network of ingredients derived from a recipe stream in social media in the framework of temporal simplicial complex. Here, for a positive integer n, a higher-order link of size $(n+1)$ (i.e., *n-simplex*) represents a set of $(n+1)$ ingredients occurring together in a recipe and can also be regarded as an interaction among $(n+1)$ ingredients. In this paper, by extending Cencetti et al.'s work [5] from a perspective of activity degree of configuraion, we propose a method of analyzing the configuration transition in a temporal network of ingredients associated with an occurrence of a new n-simplex (i.e., interaction event of $(n+1)$ ingredients in a newly posted recipe) in terms of the occurrence probabilities of the boundaries (i.e., $(n-1)$-*faces*) before and after the occurrence of the n-simplex. Using real data of a Japanese recipe sharing service, we

empirically evaluate the proposed method, and apply it to analyzing the dynamical properties of higher-order networks of ingredients for Japanese homemade recipes.

2 Related Work

Recently, a number of studies have been conducted in the field of food science and computing [12], and a variety of food-oriented applications are being explored. Examples include analysis of ingredient networks in terms of flavor compounds [1, 9], and analysis of population-wide dietary preferences based on recipe queries on the Web [18], and analysis of cuisines and culinary habits around the world from ingredients, flavors and nutritional values for large-scale data of online recipes [15]. As for recipe recommendation, Teng et al. [16] examined the use of complement and substitute networks for ingredients, and Trattner and Elswiler [17] explored how algorithmic solutions relate to the healthiness of recipes. As cross-region recipe analysis, Jiang et al. [10] jointly visualized recipe density and ingredient categories to compare food cultures in the world, and Min et al. gave a framework of culinary culture analysis by extracting cuisine-course topics of recipes from ingredient combinations (see [12]). In this paper, we explore the characteristics for the temporal evolution of higher-order interactions among ingredients in Japanese homemade recipes.

There have been many studies on link prediction for traditional dyadic-networks [11]. However, little is known about higher-order structure in large real-world data since its analysis can be computationally challenging. As for predicting higher-order links in higher-order networks, Xu et al. [19] proposed HPLSF which is a supervised method using latent features. When restricting candidate higher-order links, Zhang et al. [20] proposed Coordinated Matrix Minimization (CMM) which is a prediction method leveraging adjacency space, and empirically showed that CMM can outperform several baselines including HPLSF. On the other hand, Benson et al. [3] established a foundation for analyzing the basic structure of temporal higher-order networks, and examined higher-order link prediction in terms of simplicial closure. In this paper, as with Cencetti et al.'s work [5], we analyze the configuration transitions in temporal simplicial complex associated with new higher-order interaction events.

3 Preliminaries

Let V be a set of cooking ingredients to be considered. Based on discrete-time observations (e.g., daily observations), we explore a temporal higher-order network in V, which is derived from a cooking-recipe stream $\mathcal{R}$ consisting of recipes with time-stamps in recipe sharing services. We represent it as a temporal simplicial complex $\mathcal{K}$, where each element of V is referred to as a vertex (or a 03-*simplex*) in $\mathcal{K}$. A set of $n+1$ vertices $\sigma = \{v_0, v_1, \ldots, v_n\} \subset V$ is called a higher-order link of

size $n+1$ (or an *n-simplex*) in $\mathcal{K}$ at time t, i.e., $\sigma \in \mathcal{K}$ at time t, when there exists such a recipe $r \in \mathcal{R}$ with time-stamp t that includes ingredients $v_0, v_1, \ldots, v_n \in V$. An n'-simplex σ' is called an *n'-face* of n-simplex σ if $n' < n$ and $\sigma' \subset \sigma$. Every face of simplex $\sigma \in \mathcal{K}$ at time t is also a simplex in $\mathcal{K}$ at time t. For each recipe $r \in \mathcal{R}$, let $\sigma(r)$ denote the simplex consisting of all the ingredients in r, and let $t(r) \in \mathbb{Z}$ denote the time-stamp of r. Here, $\sigma(r)$ is referred to as the *simplex generated by recipe r*, or a *recipe simplex*, briefly. For each $r \in \mathcal{R}$, recipe simplex $\sigma(r)$ represents an essentially new simplex (i.e., an essentially new higher-order link) occurred in $\mathcal{K}$ (i.e., the temporal higher-order network of ingredients) at time $t(r)$.

For a recipe n-simplex $\sigma(r) = \{v_0(r), v_1(r), \ldots, v_n(r)\}$ at time $t(r)$, we investigate the configuration changes around recipe simplex $\sigma(r)$ in temporal simplicial complex $\mathcal{K}$ before and after the occurrence of $\sigma(r)$ at time $t(r)$ (see Fig. 1). As configurations around $\sigma(r)$ in $\mathcal{K}$, we first focus on its boundaries, i.e., its $(n-1)$-faces, $\sigma_0(r) = \{v_1(r), \ldots, v_n(r)\}$, $\sigma_1(r) = \{v_0(r), v_2(r), \ldots, v_n(r)\}$, $\ldots$, $\sigma_n(r) = \{v_0(r), v_1(r), \ldots, v_{n-1}(r)\}$ (see Fig. 2). We also explore the existence of such higher simplices in $\mathcal{K}$ that includes $\sigma(r)$. In order to incorporate an idea of *activity degree of configuration*, we consider three time-periods around the occurrence time $t(r)$, $I_A(r) = [t(r)+1, t(r)+\alpha]$, $I_B(r) = [t(r)-\alpha, t(r)-1]$ and $I'_B(r) = [t(r)-\alpha', t(r)-1]$, where α and α' are positive integers with $\alpha \ll \alpha'$ (see Fig. 1). Here, $I_A(r)$ indicates an observation period just after the occurrence of $\sigma(r)$, $I_B(r)$ indicates a short-term observation period just before the occurrence of $\sigma(r)$, and $I'_B(r)$ indicates a long-term observation period before the occurrence of $\sigma(r)$ (e.g., the last twelve months before it).

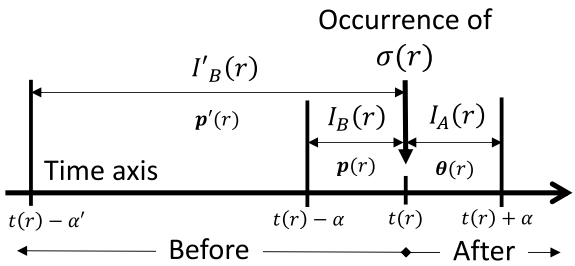


Fig. 1 Illustration of temporal transition for recipe simplex $\sigma(r)$

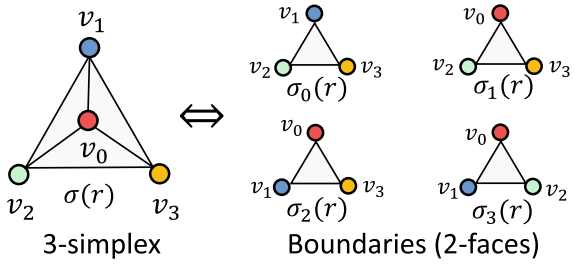


Fig. 2 Example of recipe simplex $\sigma(r)$ and its boundaries

First, we empirically estimate the probabilities $\theta_h(r)$, $p_h(r)$ and $p'_h(r)$ that a recipe simplex including $\sigma(r)$ occurs in $\mathcal{K}$ within time-periods $I_A(r)$, $I_B(r)$ and $I'_B(r)$, respectively. Next, by excluding such recipe simplices in $\mathcal{K}$ that include $\sigma(r)$ from our target, we identify the recipe simplices in which only one $\sigma_j(r)$ among the boundaries $\sigma_0(r), \sigma_1(r), \ldots, \sigma_n(r)$ occurs in $\mathcal{K}$. In this way, for $j = 0, 1, \ldots, n$, we empirically estimate the probabilities $\theta_j(r)$, $p_j(r)$ and $p'_j(r)$ that $\sigma_j(r)$ occurs in $\mathcal{K}$ within time-periods $I_A(r)$, $I_B(r)$ and $I'_B(r)$, respectively. For completeness, we also consider the recipe simplices in which none of the boundaries $\sigma_0(r), \sigma_1(r), \ldots, \sigma_n(r)$ occur in $\mathcal{K}$. We empirically estimate the probabilities $\theta_\ell(r)$, $p_\ell(r)$ and $p'_\ell(r)$ that none of $\sigma_0(r), \sigma_1(r), \ldots, \sigma_n(r)$ occur in $\mathcal{K}$ within $I_A(r)$, $I_B(r)$ and $I'_B(r)$, respectively. Note that $\sum_{j=0}^{n} \theta_j(r) + \theta_h(r) + \theta_\ell(r) = \sum_{j=0}^{n} p_j(r) + p_h(r) + p_\ell(r) = \sum_{j=0}^{n} p'_j(r) + p'_h(r) + p'_\ell(r) = 1$. To analyze the configuration transition around $\sigma(r)$ in $\mathcal{K}$, we focus on the following feature vectors (i.e., probability vectors) $\boldsymbol{\theta}(r)$, $\boldsymbol{p}(r)$ and $\boldsymbol{p}'(r)$ for $I_A(r)$, $I_B(r)$ and $I'_B(r)$, respectively (see Fig. 1);

$$\boldsymbol{\theta}(r) = (\theta_0(r), \ldots, \theta_n(r), \theta_h(r), \theta_\ell(r)),\ \boldsymbol{p}(r) = (p_0(r), \ldots, p_n(r), p_h(r), p_\ell(r)),$$
$$\boldsymbol{p}'(r) = (p'_0(r), \ldots, p'_n(r), p'_h(r), p'_\ell(r)).$$

In this paper, for each recipe simplex $\sigma(r)$, we investigate the relation between the probability vectors $\boldsymbol{p}(r)$ and $\boldsymbol{p}'(r)$ for before its occurrence and the probability vector $\boldsymbol{\theta}(r)$ for after its occurrence

4 Analysis Method

For a positive integer n, we consider a set of recipes $\mathcal{R}^n$ ($\subset \mathcal{R}$) such that each $\sigma(r) \in \mathcal{R}^n$ occurs during a specified time-period as a recipe n-simplex in temporal simplicial complex $\mathcal{K}$. We investigate the configuration transition in $\mathcal{K}$ associated with an occurrence of $\sigma(r)$ ($\forall r \in \mathcal{R}^n$).

4.1 Transition Analysis of Active Configuration

For any $r \in \mathcal{R}^n$, we explore the transition of active configuration around recipe n-simplex $\sigma(r)$ in $\mathcal{K}$. To this end, we decompose $\mathcal{R}^n$ as follows:

$$\mathcal{R}^n = \bigcup_{\lambda=1}^{n+1} \mathcal{R}^n_{A,\lambda} \cup \mathcal{R}^n_{A,h} \cup \mathcal{R}^n_{A,\ell} = \bigcup_{\lambda=1}^{n+1} \mathcal{R}^n_{B,\lambda} \cup \mathcal{R}^n_{B,h} \cup \mathcal{R}^n_{B,\ell} \quad \text{(disjoint union)}.$$

As explained below, by $\mathcal{R}^n_{A,h}$, $\mathcal{R}^n_{A,\ell}$ and $\mathcal{R}^n_{A,\lambda}$ ($\lambda = 1, \ldots, n+1$), we represent the active configurations around recipe n-simplices in $\mathcal{K}$ just after their occurrences.

Also, by $\mathcal{R}^n_{B,h}$, $\mathcal{R}^n_{B,\ell}$ and $\mathcal{R}^n_{B,\lambda}$ ($\lambda = 1, \ldots, n+1$), we represent the active configurations around recipe n-simplices in $\mathcal{K}$ just before their occurrences.

We determine to which subset each $r \in \mathcal{R}^n$ belongs in the following way: Suppose that $\sigma(r) = \{v_0(r), v_1(r), \ldots, v_n(r)\}$. First, we set $r \in \mathcal{R}^n_{A,h}$ if $\theta_h(r) > p'_h(r)$ and set $r \in \mathcal{R}^n_{B,h}$ if $p_h(r) > p'_h(r)$. Here, $r \in \mathcal{R}^n_{A,h}$ means that simplices including $\sigma(r)$ actively occur in $\mathcal{K}$ within time-period $I_A(r)$ just after the occurrence of $\sigma(r)$. Also, $r \in \mathcal{R}^n_{B,h}$ implies that simplices including $\sigma(r)$ actively occur in $\mathcal{K}$ within time-period $I_B(r)$ just before the occurrence of $\sigma(r)$. Next, in the case of $r \notin \mathcal{R}^n_{A,h}$, we examine whether $\theta_j(r) > p'_j(r)$ for each boundary $\sigma_j(r)$ of $\sigma(r)$ ($j = 0, 1, \ldots, n$). Let λ denote the number of such boundaries of $\sigma(r)$ that satisfy the above condition. We set $r \in \mathcal{R}^n_{A,\lambda}$ if $\lambda > 0$ and set $r \in \mathcal{R}^n_{A,\ell}$ if $\lambda = 0$. Here, $r \in \mathcal{R}^n_{A,\lambda}$ means that λ boundaries of $\sigma(r)$ actively occur in $\mathcal{K}$ within time-period $I_A(r)$, while $r \in \mathcal{R}^n_{A,\ell}$ implies that none of the boundaries of $\sigma(r)$ actively occur in $\mathcal{K}$ within time-period $I_A(r)$. Moreover, in the case of $r \notin \mathcal{R}^n_{B,h}$, we examine whether $p_j(r) > p'_j(r)$ for each boundary $\sigma_j(r)$ of $\sigma(r)$ ($j = 0, 1, \ldots, n$), and define the integer value $\lambda \in \{0, 1, \ldots, n+1\}$ in the same way as above. We set $r \in \mathcal{R}^n_{B,\lambda}$ if $\lambda > 0$ and set $r \in \mathcal{R}^n_{B,\ell}$ if $\lambda = 0$. Here, $r \in \mathcal{R}^n_{B,\lambda}$ means that λ boundaries of $\sigma(r)$ actively occur in $\mathcal{K}$ within time-period $I_B(r)$, while $r \in \mathcal{R}^n_{B,\ell}$ implies that none of the boundaries of $\sigma(r)$ actively occur in $\mathcal{K}$ within time-period $I_B(r)$.

We propose examining the transition rates of active configurations in $\mathcal{K}$ associated with occurrences of recipe n-simplices through *active-configuration transition matrix* T_n, which is defined by

$$T_n(x, y) = \frac{|\mathcal{R}^n_{B,x} \cap \mathcal{R}^n_{A,y}|}{|\mathcal{R}^n|} \quad (x, y = \ell, 1, \ldots, n+1, h). \tag{1}$$

Here, $|S|$ stands for the number of elements of a set S.

4.2 Influence Analysis

For $\forall r \in \mathcal{R}^n$, we consider the recipe n-simplex $\sigma(r) = \{v_0(r), v_1(r), \ldots, v_n(r)\}$, and investigate the configuration transition in temporal simplicial complex $\mathcal{K}$ under $\sigma(r)$. We analyze how the probability vectors $\boldsymbol{p}(r)$ and $\boldsymbol{p}'(r)$ for before its occurrence affect the probability vector $\boldsymbol{\theta}(r)$ for just after its occurrence. For example, since it is naturally speculated that $\boldsymbol{\theta}(r)$ can be closely related to the situation just before the occurrence of $\sigma(r)$, we might have relation $\boldsymbol{\theta}(r) \simeq \boldsymbol{p}(r)$, relation $\boldsymbol{\theta}(r) \simeq \boldsymbol{p}'(r)$, or relation $\theta_j(r) \propto p_j(r)/p'_j(r)$ ($j = 0, 1, \ldots, n$), $\theta_h(r) \propto p_h(r)/p'_h(r)$, $\theta_\ell(r) \propto p_\ell(r)/p'_\ell(r)$. In this paper, by assuming that

$$\theta_j(r) \propto \exp(w'')\{p_j(r)\}^{w}\{p'_j(r)\}^{w'} \quad (j = 0, 1, \ldots, n), \tag{2}$$
$$\theta_h(r) \propto \exp(w''_h)\{p_h(r)\}^{w_h}\{p'_h(r)\}^{w'_h}, \quad \theta_\ell(r) \propto \exp(w''_\ell)\{p_\ell(r)\}^{w_\ell}\{p'_\ell(r)\}^{w'_\ell},$$

and by employing a multinomial generative model with parameter $\boldsymbol{\theta}(r)$, we propose analyzing the influence of $\boldsymbol{p}(r)$ and $\boldsymbol{p}'(r)$ on $\boldsymbol{\theta}(r)$ in terms of weight vector $\boldsymbol{w} = (w, w', w'', w_h, w'_h, w''_h, w_\ell, w'_\ell, w''_\ell)$.

By conforming to the MAP estimation framework, we estimate the weight vector $\boldsymbol{w}$. Let $\mathcal{S}^n_{A,h}$ be a set of $r' \in \mathcal{R}^n$ such that the time-stamp $t(r')$ belongs to time-period $I_A(r)$ and the recipe simplex $\sigma(r')$ includes $\sigma(r)$. Note that for $\forall r' \in \mathcal{S}^n_{A,h}$, $\sigma(r')$ represents such a new interaction of ingredients that is higher than or equal to $\sigma(r)$. Next, for $\forall j \in \{0, 1, \ldots, n\}$, let $\mathcal{S}^n_{A,j}$ be a set of $r' \in \mathcal{R}^n \setminus \mathcal{S}^n_{A,h}$ such that $t(r') \in I_A(r)$ and $\sigma(r')$ includes the boundary $\sigma_j(r)$ of $\sigma(r)$. Note that for $\forall r' \in \mathcal{S}^n_{A,j}$, $\sigma(r')$ represents a new interaction including only $\sigma_j(r)$ among the boundaries of $\sigma(r)$. Next, let $\mathcal{S}^n_{A,\ell}$ be a set of $r' \in \mathcal{R}^n$ such that $t(r') \in I_A(r)$ and $\sigma(r')$ does not include $\sigma_j(r)$ for $\forall j \in \{0, 1, \ldots, n\}$. Note that for $\forall r' \in \mathcal{S}^n_{A,\ell}$, $\sigma(r')$ represents such a new interaction that is lower than any boundary $\sigma_j(r)$ within $\sigma(r)$, and $\mathcal{R}^n = \bigcup_{j=0}^n \mathcal{S}^n_{A,j} \cup \mathcal{S}^n_{A,h} \cup \mathcal{S}^n_{A,\ell}$ (disjoint union). We set

$$m_j(r) = |\mathcal{S}^n_{A,j}| \ \ (j = 0, 1, \ldots, n), \quad m_h(r) = |\mathcal{S}^n_{A,h}|, \quad m_\ell(r) = |\mathcal{S}^n_{A,\ell}|.$$

In the proposed model, the probability $P(\boldsymbol{m}(r) \mid \boldsymbol{w})$ of observing $\boldsymbol{m}(r) = (m_0(r), m_1(r), \ldots, m_n(r), m_h(r), m_\ell(r))$ in time-period $I_A(r)$ is given by

$$P(\boldsymbol{m}(r) \mid \boldsymbol{w}) \ \propto \ \{\theta_h(r)\}^{m_h(r)} \{\theta_\ell(r)\}^{m_\ell(r)} \prod_{j=0}^{n} \{\theta_j(r)\}^{m_j(r)} \tag{3}$$

To estimate the value of $\boldsymbol{w}$, we consider maximizing the objective function,

$$\mathcal{L}(\boldsymbol{w}) = \sum_{r \in \mathcal{R}^{(n)}} \left\{ m_h(r) \log \theta_h(r) + m_\ell(r) \log \theta_\ell(r) + \sum_{j=0}^{n} m_j(r) \log \theta_j(r) \right\} - \frac{1}{2\mu^2} \|\boldsymbol{w}\|^2, \tag{4}$$

where $\mu > 0$ is a hyper-parameter, $\|\boldsymbol{w}\|$ stands for the Euclidean norm of $\boldsymbol{w}$, and

$$\begin{aligned}
&\theta_h(r) = \frac{\exp\{a_h(r)\}}{Z(r)}, \ \theta_\ell(r) = \frac{\exp\{a_\ell(r)\}}{Z(r)}, \ \theta_j(r) = \frac{\exp\{a_j(r)\}}{Z(r)} \quad (j = 0, 1, \ldots, n), \\
&a_h(r) = w_h \log p_h(r) + w'_h \log p'_h(r) + w''_h, \\
&a_\ell(r) = w_\ell \log p_\ell(r) + w'_\ell \log p'_\ell(r) + w''_\ell, \\
&a_j(r) = w \log p_j(r) + w' \log p'_j(r) + w'' \quad (j = 0, 1, \ldots, n), \\
&Z(r) = \exp\{a_h(r)\} + \exp\{a_\ell(r)\} + \sum_{j=0}^{n} \exp\{a_j(r)\}
\end{aligned} \tag{5}$$

(see Eqs. 2 and 3). Here, a Gaussian prior for $\boldsymbol{w}$ is assumed. Note that maximizing $\mathcal{L}(\boldsymbol{w})$ reduces to a sort of softmax optimization problem in neural network (see Eqs. 4 and 5). Thus, we employed a gradient-based method [4].

5 Experiments

For real higher-order networks of ingredients derived from Japanese recipe-sharing service, Cookpad[1] we empirically evaluate the proposed analysis method, and present the analysis results for the dynamical properties of higher-order interactions among ingredients.

5.1 Datasets

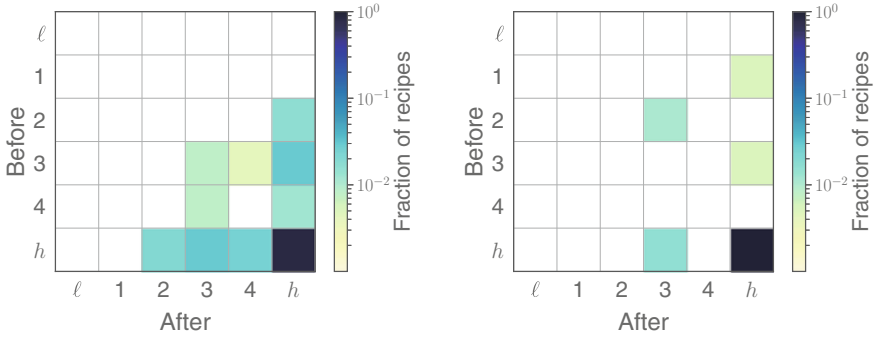
As recipe stream $\mathcal{R}$, we employed four datasets of recipe stream from Jan 1, 2010 to Jan 30, 2012, "Dessert", "Meat-dish", "Vegetable-dish" and "Fish-dish", each of which corresponds to a recipe category of Cookpad. Here, the numbers of recipes for these datasets were 5, 363, 3, 700, 6, 547 and 2, 147, respectively. For each of the recipe streams, we adopted the set of its major ingredients[2] as a set of vertices V. Then, the numbers of vertices for the recipe stream datasets were 824, 684, 860 and 610, respectively. Next, for each of the recipe streams, we constructed the dataset $\mathcal{R}^n$ by setting α to one month and α' to one year, and by extracting such recipes that consisted of exactly $(n+1)$ ingredients in V and occurred within 2011. Although the previous work [5] examined the case of $n = 2$ (i.e., three people) for social interactions, we are interested in the case of $n > 2$ since many recipes included more than three major ingredients. In this paper, we focused on the cases of $n = 3$ (i.e., four ingredients) and $n = 4$ (i.e., five ingredients) for simplicity. Then, the numbers of samples, $(|\mathcal{R}^3|, |\mathcal{R}^4|)$, for Desert, Meat-dish, Vegetable-dish and Fish-dish datasets were (350, 370), (350, 260), (740, 600), and (350, 200), respectively.

5.2 Results for Transition Analysis of Active Configuration

For each dataset, we investigated the transition of active configuration in temporal simplicial complex $\mathcal{K}$ associated with an occurrence of recipe n-simplex $\sigma(r)$ by employing the proposed active-configuration matrix T_n for the cases of $n = 3$ and

[1] https://cookpad.com/.

[2] First, we removed general-purpose ingredients for Japanese food such as soy sauce, salt, sugar, water, edible oil, and so on. Next, we extracted such ingredients that appeared in five or more recipes as a set of vertices V for each recipe stream.



(a) Dessert ($n = 3$) (b) Meat-dish ($n = 3$)

Fig. 3 Visualization results of active-configuration matrix T_3

$n = 4$ (see Eq. 1). Due to space limitation, we only display the results for Dessert and Meat-dish datasets in the case of $n = 3$ (i.e., four ingredients).

Figure 3 shows the visualization of T_3, where Fig. 3a, b indicate the results of Dessert and Meat-dish, respectively. As for the possibility of configuration transition for $n = 3$, there can be 36 transition patterns, which means that matrix T_3 consists of 36 entries. However, only a relatively small number of transition patterns were observed, that is, the number of non-zero entries of T_3 was relatively small. Also, $T_3(h, h)$ was significantly large in the entries of T_3, that is, the temporal transition of active configuration around $\sigma(r)$ was characterized as follows: Simplices including $\sigma(r)$ actively occurred in $\mathcal{K}$ just before $\sigma(r)$ occurs, and simplices including $\sigma(r)$ actively occur in $\mathcal{K}$ just after $\sigma(r)$ occurred. From these facts, we speculate that social media users for Japanese recipes tend to try to make easy recipes with fewer ingredients on the basis of recent nice recipes, and also try to create novel nice recipes by adding some ingredients for a recently posted recipe. Here, we note that the properties mentioned above were also true for the other datasets of $n = 3, 4$. Moreover, we note that these findings for interactions among ingredients in recipe sharing services were essentially different from the property empirically found by the previous work [5] for configuration transitions of human interactions in several social settings.

By examining T_n, we can also extract several interesting phenomena. For example, for Dessert ($n = 3$) dataset, we observe that $T_3(h, 4)$ was relatively large (see Fig. 3a), indicating that there were several recipes $r \in \mathcal{R}^3$ satisfying $r \in \mathcal{R}^3_{B,h} \cap \mathcal{R}^3_{A,4}$. Examples include Crispy Sesame Cookies recipe constructed from four ingredients "cake flour", "egg", "margarine" and "sesame" in V. For such a recipe r, the transition of active configuration around $\sigma(r)$ is described as follows: Just before the occurrence of $\sigma(r)$, simplices including $\sigma(r)$ actively occurred in $\mathcal{K}$. However, just after $\sigma(r)$ was produced, 3-simplex $\sigma(r)$ decays into its four boundaries $\sigma_j(r)$ ($j = 0, 1, 2, 3$) (see Fig. 2), meaning that all the four boundaries actively occur in $\mathcal{K}$ while there are no simplices that both include $\sigma(r)$ and actively occur in $\mathcal{K}$.

These results demonstrate that the proposed analysis method can reveal several interesting properties for transitions of active configurations associated with occurrences of recipe simplices in temporal higher-order networks of ingredients derived from social media data.

5.3 *Evaluation of Influence Analysis Model*

For analyzing the influence of $\boldsymbol{p}(r)$ and $\boldsymbol{p}'(r)$ on $\boldsymbol{\theta}(r)$ associated with an occurrence of recipe n-simplex $\sigma(r)$ ($\forall r \in \mathcal{R}^n$), we empirically evaluate the proposed generative model (see Eqs. 2 and 4) in terms of prediction performance. For each dataset, we split it into training and test sets at a ratio of 7 : 3 along the time-axis. We estimated the weight vector $\boldsymbol{w}$ from the training set, and evaluated the prediction capability of the learned model using the test set in terms of *prediction log-likelihood ratio* PLR. Here, PLR is defined by the difference between the learned model and the uniformly random model with respect to the log-likelihood for the test set (see Eqs. 3 and 4), and measures the relative performance versus the random guessing. Note that the uniformly random model is obtained by setting $\boldsymbol{w} = \boldsymbol{0}$, i.e., $w = w_h = w_\ell = w' = w'_h = w'_\ell = w'' = w''_h = w''_\ell = 0$ (see Eq. 2).

We compared the proposed model with the following four baseline models. The first and second baseline models simply assume that the probability vector $\boldsymbol{\theta}(r)$ for $I_A(r)$ is the same as the probability vectors $\boldsymbol{p}(r)$ for the time period $I_B(r)$ and $\boldsymbol{p}'(r)$ for the time period $I'_B(r)$, respectively. These models are referred to as *baselines 1* and *2*, respectively. Note that the baselines 1 and 2 are obtained by setting $w = w_h = w_\ell = 1$, $w' = w'_h = w'_\ell = w'' = w''_h = w''_\ell = 0$ and by setting $w' = w'_h = w'_\ell = 1$, $w = w_h = w_\ell = w'' = w''_h = w''_\ell = 0$, respectively (see Eq. 2). By taking into account activity and inactivity, the third and fourth baseline models assume that $\boldsymbol{\theta}(r)$ is proportional to the ratio of $\boldsymbol{p}(r)$ to $\boldsymbol{p}'(r)$ and the ratio of $\boldsymbol{p}'(r)$ to $\boldsymbol{p}(r)$, respectively. Here, the elementwise division of two vectors is used. These models are referred to as *baselines 3* and *4*, respectively. Note that the baselines 3 and 4 are obtained by setting $w = w_h = w_\ell = 1$, $w' = w'_h = w'_\ell = -1$, $w'' = w''_h = w''_\ell = 0$ and by setting $w = w_h = w_\ell = -1$, $w' = w'_h = w'_\ell = 1$, $w'' = w''_h = w''_\ell = 0$, respectively (see Eq. 2).

Figure 4 shows the evaluation results of the proposed model, baseline 1 and baseline 2 in terms of PLR metric for the datasets of $n = 3$ (i.e., 3-simplices) and $n = 4$ (i.e., 4-simplices) for the four recipe streams. We see that these three models provided much better performance than the random guessing although the value of PLR can depend on datasets. Here, we omit the results of baselines 3 and 4 since they were comparable in PLR to the uniformly random model for all datasets. We note that baseline 2 was comparable to and slightly better than baseline 1 except for the Vegetable-dish ($n = 4$) dataset. Thus, for almost all datasets, probability vectors $\boldsymbol{p}(r)$ for short-term period $I_B(r)$ and $\boldsymbol{p}'(r)$ for long-term period $I'_B(r)$ had comparable effects on prediction, and the idea of taking into account activity and inactivity

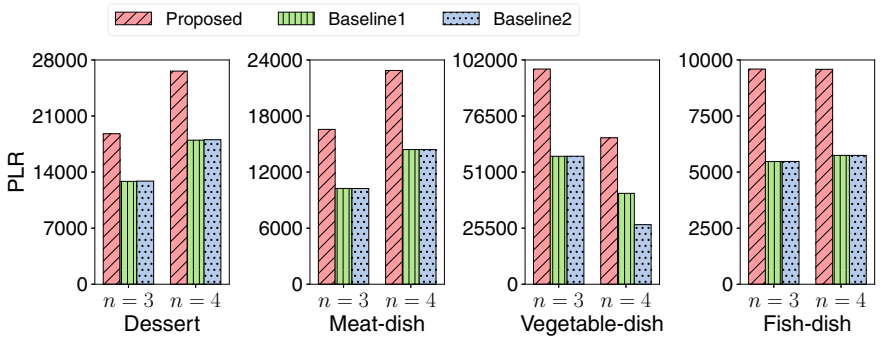


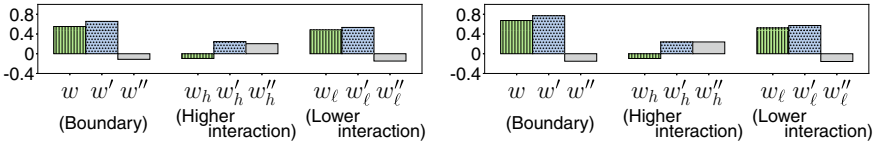
Fig. 4 Evaluation results of the proposed model

through $\boldsymbol{p}(r)$ and $\boldsymbol{p}'(r)$ was not useful. However, we see that the proposed model always performed the best. These results demonstrate the effectiveness of the proposed model. Hence, we employed the proposed model for our influence analysis.

5.4 *Results for Influence Analysis*

For the configuration transition in temporal simplicial complex $\mathcal{K}$ under an occurrence of recipe n-simplex $\sigma(r)$ ($\forall r \in \mathcal{R}^n$), we investigated the influence of $\boldsymbol{p}(r)$ and $\boldsymbol{p}'(r)$ on $\boldsymbol{\theta}(r)$ in terms of weight vector $\boldsymbol{w}$ for the eight datasets by employing the proposed analysis model. Here, we present the analysis results only for Dessert and Meat-dish datasets in the case of $n = 4$ (i.e., interactions of five ingredients).

Figure 5a, b indicate the results of $\boldsymbol{w}$ for Dessert and Meat-dish datasets, respectively. We first observe that these datasets had a quite similar tendency. For example, the bias w''_h toward such interactions that are higher than or equal to $\sigma(r)$ was larger, compared to the other biases w'' and w''_ℓ (see Eq. 2). As for each boundary $\sigma_j(r)$ of $\sigma(r)$ ($j = 0, 1, \ldots, 5$), the influence of probability $p'_j(r)$ within long-term period $I'_B(r)$ was slightly stronger than that of probability $p_j(r)$ within short-term period $I_B(r)$ (see the results of weights w and w'). Also, this property was true for such interactions that are lower than each boundary $\sigma_j(r)$ within $\sigma(r)$ (see the results of weights w_ℓ and w'_ℓ). On the other hand, from Eq. (5) and the results of weights w_h and w'_h, we see that feature log $p_h(r)$ within $I_B(r)$ negatively affected such future interactions that are higher than or equal to $\sigma(r)$, while feature log $p'_h(r)$ within $I'_B(r)$ positively affected them. Note that the observed properties largely depended on datasets. Thus, these facts show that the proposed influence analysis method can reveal interesting category-specific characteristics for the temporal evolution of Japanese homemade recipes in terms of combinations of ingredients.



(a) Dessert ($n = 4$) (b) Meat-dish ($n = 4$)

Fig. 5 Results for influence analysis

6 Conclusion

Based on the idea of activity degree of configuraion, we attempted to extend the previous work [5] that deals with temporal transition analysis of triplet structures of human proximity interactions. We have proposed a novel method of analyzing the configuration transitions in a temporal simplicial complex (i.e., a typical model of higher-order network to represent interactions among any number of units) associated with occurrences of new simplices, and applied it to a dynamical analysis of higher-order networks of ingredients derived from a recipe stream in social media. By taking into account the nature of recipe simplices (i.e., simplices generated by recipes), we in particular addressed interactions among more than three ingredients.

In the proposed analysis method, we focused on the boundaries of each new recipe simplex $\sigma(r)$ of a given dimension n. First, we have presented an analysis method for the temporal transition of active configuration associated with the occurrence of $\sigma(r)$ through visualizing the active-configuration transition matrix. Next, by proposing a probabilistic generative model for an influence analysis, we have provided a method of analyzing how the occurrence probabilities of the corresponding simplices (i.e., the boundaries of $\sigma(r)$ and the higher-order interactions including $\sigma(r)$) before the occurrence of $\sigma(r)$ affects those probabilities just after the occurrence of $\sigma(r)$. Using real data of a Japanese recipe sharing service, we empirically demonstrated the effectiveness of the proposed method for the cases of $n = 3$ (i.e., interactions of four ingredients) and $n = 4$ (i.e., interactions of five ingredients). Moreover, by employing the proposed method, we revealed several interesting category-specific characteristics for the temporal evolution of Japanese homemade recipes in terms of combinations of ingredients.

Acknowledgements This work was supported in part by JSPS KAKENHI Grant Number JP21K12152. The Cookpad dataset we used in this paper was provided by Cookpad Inc. via IDR Dataset Service of National Institute of Informatics.

References

1. Ahn, Y.Y., Ahnert, S.E., Bagrow, J.P., Barabási, A.L.: Flavor network and the principles of food pairing. Sci. Rep. **1**, 196:1–196:7 (2011)
2. Barabási, A.L.: Network Science. Cambridge University Press (2016)
3. Benson, A.R., Abebe, R., Schaub, M.T., Jadbabaie, A., Kleinberg, J.: Simplicial closure and higher-order link prediction. PNAS **115**(48), E11221–E11230 (2019)
4. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
5. Cencetti, G., Battiston, F., Lepri, B., Karsai, M.: Temporal properties of higher-order interactions in social networks. Sci. Rep. **11**, 7028:1–7028:10 (2021)
6. Croom, F.H.: Basic Concepts of Algebraic Topology. Springer (2007)
7. Estrada, E., Ross, G.J.: Centralities in simplicial complexes. Applications to protein interaction networks. J. Theor. Biol. **438**, 46–60 (2018)
8. Iacopini, I., Petri, G., Barrat, A., Latora, V.: Simplicial models of social contagion. Nat. Commun. **9**, 2485:1–1399:9 (2019)
9. Jain, A., N K, R., Bagler, G.: Analysis of food pairing in regional cuisines of India. PLoS One **10**(10), 1–17 (2015)
10. Jiang, Y., Skufca, J.D., Sun, J.: Bifold visualization of bipartite datasets. EPJ Data Sci. **6**, 2:1–2:19 (2017)
11. Lü, L., Zhou, T.: Link prediction in complex networks: a survey. Phys. A Stat. Mech. Appl. **390**(6), 1150–1170 (2011)
12. Min, W., Jiang, S., Liu, L.: A survey on food computing. ACM Comput. Surv. **52**(5), 92:1–92:36 (2019)
13. Preti, G., Moralest, G.D.F., Bonchi, F.: Strud: Truss decomposition of simplicial complexes. In: Proceedings of WWW'21, pp. 3408–3418 (2021)
14. Saggar, M., Sporns, O., Gonzalez-Castillo, J., Bandettini, P.A., Carlsson, G., Glover, G., Reiss, A.L.: Towards a new approach to reveal dynamical organization of the brain using topological data analysis. Nat. Commun. **9**, 1399:1–1399:14 (2018)
15. Sajadmanesh, S., Jafarzadeh, S., Ossia, S.A., Rabiee, H.R., Haddadiy, H., Mejovaz, Y., Musolesi, M., Cristofaro, E.D., Stringhini, G.: Kissing cuisines: exploring worldwide culinary habits on the web. In: Proceedings of WWW'17 Companion, pp. 1013–1021 (2017)
16. Teng, C.Y., Lin, Y.R., Adamic, L.A.: Recipe recommendation using ingredient networks. In: Proceedings of WebSci'12, pp. 298–307 (2012)
17. Trattner, C., Elsweiler, D.: Implications for meal planning and recommender systems. In: Proceedings of WWW'17, pp. 489–498 (2017)
18. West, R., White, R.W., Horvitz, E.: From cookies to cooks: insights on dietary patterns via analysis of web usage logs. In: Proceedings of WWW'13, pp. 1399–1410 (2013)
19. Xu, Y., Rockmore, D., Kleinbaum, A.M.: Hyperlink prediction in hypernetworks using latent social features. In: Proceedings of DS'13, pp. 324–339 (2013)
20. Zhang, M., Cui, Z., Jiang, S., Chen, Y.: Beyond link prediction: predicting hyperlinks in adjacency space. In: Proceedings of AAAI'18, pp. 4430–4437 (2018)

Role of Network Topology in Between-Community Beta Diversity on River Networks

Richa Tripathi, Amit Reza, and Justin M. Calabrese

Abstract The between-community beta diversity of fish species—characterized using similarity of species between river basins, shows a non-linear drop with topological distance on river networks. In this work, we investigate the pattern of this drop with network distances and the role of underlying topology. Using the framework of optimal channel networks, the species abundances are evolved under the neutral biodiversity model. We observe that the steady-state species-similarity shows a transition at a critical network distance. At this critical distance, the average degree over the nodes crosses the global average degree of the network. This study sheds light on the role of branching in dendritic networks in ecological community assembly rules.

Keywords Beta diversity · Optimal channel networks · The neutral model of biodiversity

1 Introduction

River systems across the globe, characterized by dendritic geometry, are natural habitats of many freshwater fish species. The structure of these dendritic networks has been shown to affect the biodiversity patterns and is pivotal in maintaining fish biodiversity [3, 4, 10, 14]. Network-based studies in conjugation with biodiversity models have successfully predicted emergent patterns in the distribution of biodiversity [15]. Network geometry not only assigns designated paths to fish movement but the dendritic structure is also known to increase the persistence times of species in the local community [17]. Studies using optimal channel networks (OCN) [6], which provide a theoretical basis for studying the effects of dendritic geometry in river networks, have helped in understanding how evolving network geometry and

R. Tripathi (✉) · J. M. Calabrese
Centre for Advanced Systems Understanding (CASUS), HZDR, 02826 Görlitz, Germany
e-mail: r.tripathi@hzdr.de

A. Reza
Nikhef, Science Park 105, Amsterdam 1098 XG, Netherlands

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_49

dispersal strategies shape the biodiversity patterns [16]. Network analysis [8, 11] and spectral study of OCNs [1] have contributed to our understanding of the relationship between topology and metapopulation dynamics in riverine ecosystems. These results suggest a systematic exploration of the impact of river network topology on species-similarity as a function of network distance.

Beta diversity, which here quantifies species-similarity between river basins, is observed to decline non-linearly with topological distances between the river basins [15] in real river systems. While species-similarity decays sharply over smaller network distances, there is long-tailed decay at intermediate to long distances. We are interested in identifying network topological factors responsible for this pattern of species-similarity decline. With this aim, we explore how the node properties with respect to their connectivity in the network, change statistically with topological distances.

Additionally, it has been found that the more branched the network is, the higher is the beta diversity [18, 19]. So, for a given network, it is peculiar to understand changes in branching patterns as a function of topological distance. As a measure of branchiness, we use average degree of nodes, and find that on average, OCNs have highly branched nodes at low network distance and low branched nodes at high network distance. In hyperbolic geometry also, the average degree of complex networks is found to be decaying exponentially with distance r from the origin [13]. These two pieces of evidence, the decay of species-similarity with distance and decay of average degree with distance, motivate a direct comparative study of these patterns.

In summary, we look at the statistical distribution of the average degree of network nodes as a function of network distance and correlate it with the biodiversity pattern of beta diversity. We use the neutral model of biodiversity [12] to evolve the species' abundances over time on the OCN and record the species-similarity between basins at system steady-state [15]. Neutral model has been remarkably successful in reproducing biodiversity patterns of fishes in river system and trees in forests [17]. We find that species-similarity shows phase transition-like behaviour as the average degree is increased beyond a critical value. Such a study can help us analyze the structure of the network and its impact on the function, which here is the dynamics of riverine biodiversity.

2 Methods

In this section we discuss the details of neutral model implementation that is used to evolve the fish species abundances on branches of optimal channel networks via three different dispersal kernels. We also discuss how species-similarity is calculated after the model reaches its steady-state.

2.1 Neutral Model Implementation

The neutral model of biodiversity assumes species to be "neutral" or functionally equivalent i.e., they do not have any competitive advantages over one another [12]. The abundances of species change stochastically over time under the processes of dispersal, and diversification [15]. At any time in the model run, two events occur: a fundamental entity of the system, here a fish, dies at randomly chosen site (local community), and the freed space is either occupied by an offspring of species within the network via dispersal or by a species from outside the system via a diversification process with a small probability (ν). The dispersal process is quantified by mathematical kernels that dictate the movement of fishes on the network. The fishes disperse locally more often due to shorter network distances, but there is a small but finite probability for a species to disperse to very large network distances. The model is run until the steady-state is reached and then the biodiversity patterns, e.g. alpha diversity, beta diversity and gamma diversity, are recorded.

We first assume all the sites (regions of fish habitat) on the network to be environmentally alike, i.e. they have the same habitat capacities (H), and hence hold same number of fishes at any given time. Subsequently, we scale the H with respect to the distance from the outlet, with sites closer to the outlet having higher H and vice-versa. To incorporate the effect of kind of dispersal in our study, we simulate the neutral model with three dispersal kernels D: *Exponential + Cauchy's* [15], *back-to-back Exponential* [16], and *Fat tailed radially symmetric Kernel* [9]. If r_{ij} is the distance between the sites i and j on the network, these dispersal kernels are expressed as follows:

$$D_{ij}^{EC} = C\left(a^{r_{ij}} + \frac{b^2}{b^2 + r_{ij}^2}\right) \tag{1}$$

$$D_{ij}^{BBE} = C\left(a^{r_{ij}}\right) \tag{2}$$

$$D_{ij}^{FRS} = -C\left(\frac{\eta + 2}{2\pi d^2}\left(1 + \frac{{r_{ij}}^2}{d}\right)^{\eta/2}\right) \tag{3}$$

Parameter C in above expressions is the normalization constant ensuring that $\sum_j D_{ij} = 1$, or that the dispersal occurs only within the network. Parameter a $(0 < a < 1)$ characterizes short range dispersal in exponential kernels, and b characterizes long distance decay pattern. d is mean dispersal distance and η is chosen such that the kernel has fat tails. The values of these parameters are presented in Table. 1.

The probability that a emptied site i will be colonized by a species from site j is obtained as follows:

$$P_{ij} = (1 - \nu)\frac{D_{ij} H_j}{\sum_m D_{im} H_m}, \tag{4}$$

Table 1 Parameter sets of three different Kernels

Kernel	Parameter value
Exponential + Cauchy's	$a = 0.26, b = 0.03$
Exponential	a = 0.26
Radially symmetric	$d = 1, \eta = -4.4$

where $(1 - \nu)$ is the probability of the diversification process not happening, and the rest of the terms determine the influence of distance-dependent dispersal and habitat capacity of the source site on the total probability.

2.2 Beta Diversity

To characterize beta diversity of fish species in two basins, we use Jaccard's Similarity Index (JSI) and study its decay with distance [7, 15]. For two basins i and j, JSI is defined as JSI $= \frac{S_{ij}}{S_i + S_j - S_{ij}}$, where S_i and S_j are the numbers of fish species in basins i and j, respectively, and S_{ij} is the number of common fish species between these basins. JSI generally shows sharper decay at shorter distances and slower decay at large distances.

2.3 Optimal Channel Networks

We use optimal channel networks (OCN) [2, 6] as the spatial framework for dispersal on river systems. These structures are optimal in the sense of minimizing energy dissipation of the erosional process that forms them. The energy is expressed as $E = \sum_i^{N-1} L_i q_i^{\gamma}$, where L_i and q_i represent length and discharge of link i, respectively and N is the number sites in the network. The variation of parameter γ in the energy function is used to generate OCNs with different bifurcation patterns.

3 Results

We present our results on three 20×20 OCNs as generated by setting $\gamma = 0.1,\ 0.5,$ and 0.9 using R-package OCNet [5]. These OCNs are shown in the top row of Fig. 1. The outlets are at the bottom left site for all three networks. The choice of the OCNs with 400 sites is arbitrary and γ values are chosen to cover its range $0 <= \gamma <= 1$ in our results. The variation of γ reflects in the drainage pattern, which changes from an intertwined river network to a more regular-looking pattern as γ is increased. The

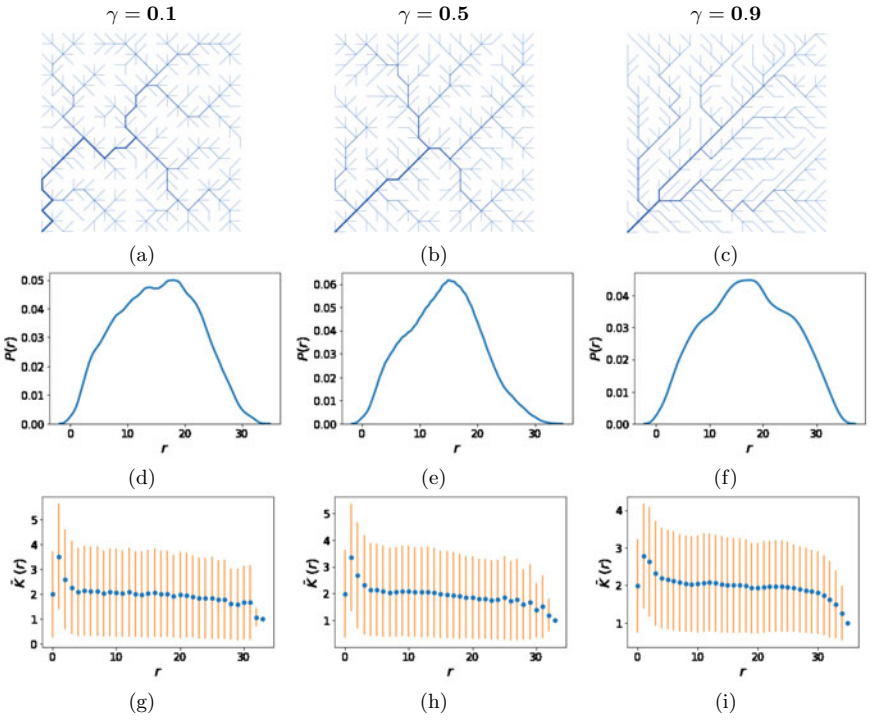


Fig. 1 Properties of OCNs. The first row in the figure shows three different OCN networks obtained by setting three different values of γ. The second row shows the corresponding probability distribution of shortest paths on the OCNs. The last row shows the average over degrees of nodes at a network distance r, as a function of r

distribution of pair-wise topological distances r, presented in the next row of Fig. 1 d–f, is almost similar for all three OCNs. The distributions for the first two OCNs are slightly more right-skewed than the third one, and the OCN with $\gamma = 0.9$ has slightly larger mean distance $\bar{r} \approx 17$ and broader range of r than the first two with means as $\bar{r} \approx 15$ and $\bar{r} \approx 14$, respectively. Next, we show the average over the degree of the nodes at topological distance r. Starting from a node i considered to be at the origin, we identify its neighbours at a given distance r, and calculate their degrees. This is repeated for all nodes $i = 1, 2, ...N$ and for all distances $r = 0, 1, \ldots r_{\max}$. Following this, the average degree at a distance r, $\bar{K}(r)$ is plotted as a function of r for all the three networks in parts (j–l) of Fig. 1. The profile for these functions is similar across all the three variations—it shows higher $\bar{K}$ at lower r, and lower $\bar{K}$ at higher r, and roughly flat $\bar{K}$ for intermediate the r. These three portions form a smoother curve for $\gamma = 0.9$ OCNs than for the other two. Note that at $r = 0$, $\bar{K}(r)$ is the average degree ($\langle K \rangle$) of the network.

For the neutral model simulations on these networks, at $t = 0$, we assume all the $N = 400$ sites on these OCNs to be occupied by one of the species chosen from

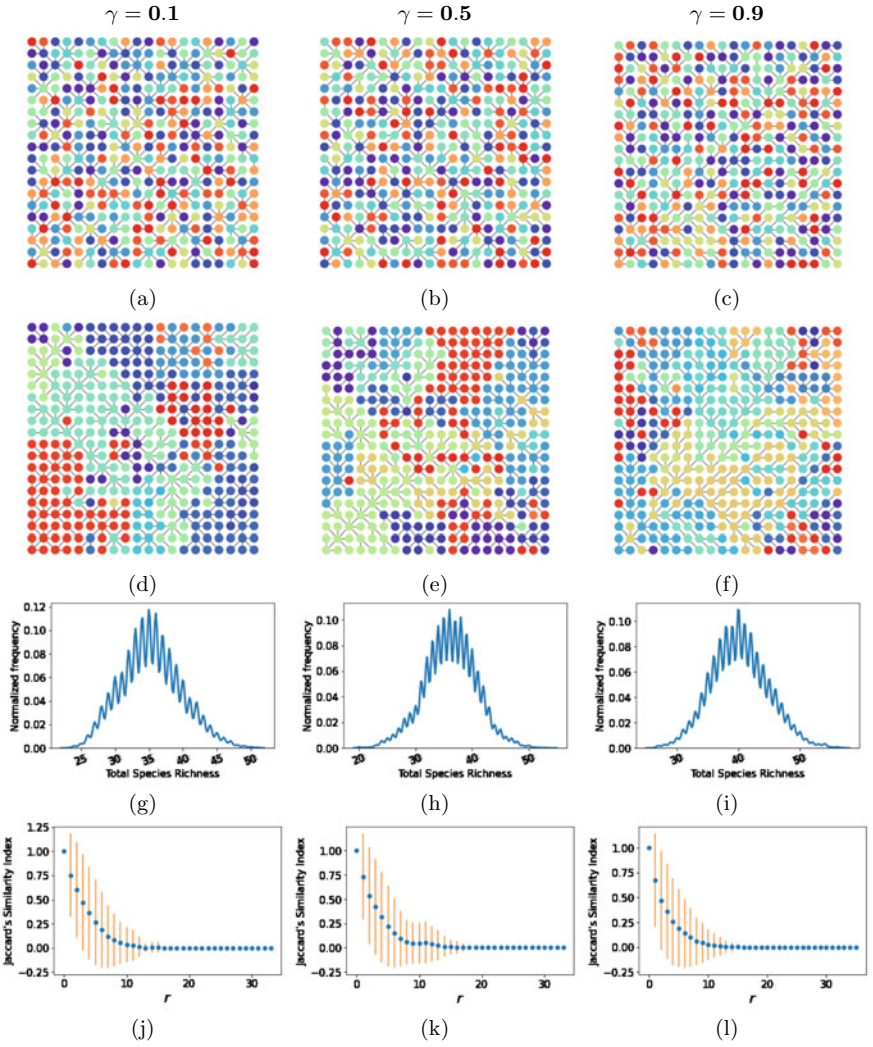


Fig. 2 Simulation of the neutral model of biodiversity on OCNs. The top row of the figure shows OCNs with a fish species occupying each node (indicated by colour) at the start of the simulation. The second row is a snapshot of the system at a steady-state, showing clear spatial assemblages of species over the networks. The third row is the distribution of TSR at the steady-state of the system. The third row shows Jaccard's Similarity Index as a function of network distance (r)

a list of 10 species. This also means that number of species at each site, or the local species richness (LSR) is one. Hence, the initial system state has each site occupied by a single species randomly chosen from a set of species. This is depicted by coloured dots (species) on the OCN sites in Fig. 2 a–c, where color identifies with species. The habitat capacities are assumed to be the same $H_i = 1, i = 1, 2, \ldots N$

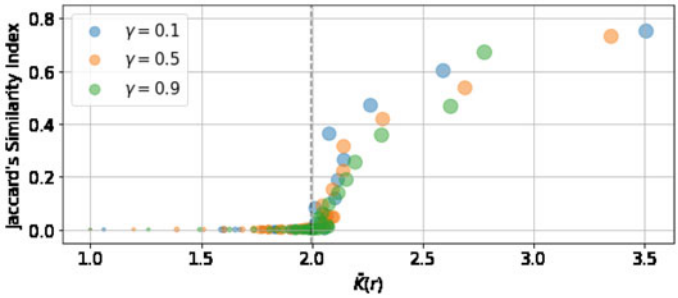
for all sites, and dispersal is agnostic to the location of the outlet, i.e., the fish units move both upstream and downstream with equal probabilities. At each time step, the system state is updated via a neutral process, with diversification probability $\nu = 1.5 * 10^{-2}$, and dispersal via one of the kernels. The results in Fig. 2 are for Exponential and Cauchy's Kernel (D^{EC}). These state updates continue until the system reaches a steady-state when the Total Species Richness (TSR) of the system shows no directional changes. The steady-state, a snapshot of which is shown in parts (d–f) of the figure, is characterized by a clear spatial organization of species into communities. The steady-state has around 35 species for the first two OCNs ($\gamma = 0.1, 0.5$) and 40 species for the last OCN ($\gamma = 0.9$) as shown by the distribution of TSR at the steady-state (a range of values of TSR at steady state). The alpha diversity or the local species richness is always one because each site is prescribed to hold only a single fish unit. The JSI measuring the species-similarity between two basins as a function of distance is shown in parts (j–l) of the figure. The JSI shows a decline in topological distance between species pairs, as expected. It is finite valued until topological distance ($r \approx 15$), beyond which it goes to 0 for all the three OCNs.

This transition of JSI motivates the search for corresponding changes in network topology at the network distance at which the transition occurs. Next, we focus on how beta diversity (shown in the last panel of Fig. 2) varies with average network degree at a distance r (shown in the last panel of Fig. 1). We plot the former against the latter for all the three OCNs to observe this relation. As shown in Fig. 3a, we observe that the JSI only becomes finite when the average degree at a distance r increases beyond the value 2. The sizes of the dots in the figure are inversely proportional to network distance r. So, starting from a finite non-zero value, both the JSI and the $\bar{K}(r)$ decrease with increasing r, and JSI becomes 0 as $\bar{K}(r) = 2$ is crossed. Notice that $\bar{K}(r) = 2$ is also the global average degree ($\langle K \rangle$) of the network shown by dashed vertical lines in the figures. Hence, $\langle K \rangle$ seems to be a critical degree at which JSI shows a characteristic change. This observation is robust across all the three OCNs and the three dispersal Kernel choices (see Methods). In all the figures the (expected) value of JSI = 1 at $r = 0$, and $\bar{K}(r) = 2$ is omitted for visualization purpose. For $\gamma = 0.1, 0.5$, maximum JSI (except JSI = 1) is observed for higher $\bar{K}(r)$ than for $\gamma = 0.9$. Overall, no pronounced differences are observed in the pattern of JSI transition either across OCNs or across dispersal kernel choices. Importantly, this suggests that our quantitative results are not sensitive to these modeling choices.

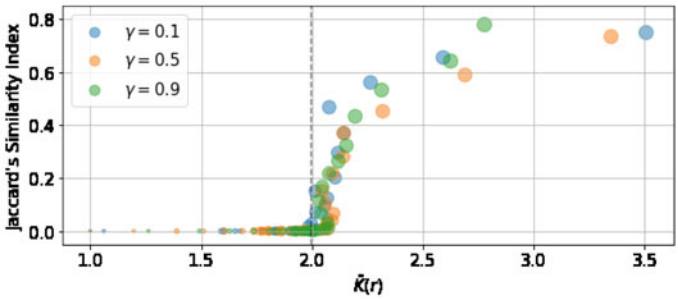
To further test the robustness of the transition of JSI on the dendritic network from zero to non-zero values as a function of $\bar{K}(r)$, we drop the two assumptions: H = 1 and LSR = 1, and simulate the model on higher H values in both two settings. In the first one, all sites are assumed to have H = 50, and the initial state had each site with at most 5 species chosen from a list of 10 species. In the second setting, the intial number of species are similarly assigned, but the H of a site is proportional to its distance from the outlet. The highest H = 50 at the outlet and the H = 2 headwaters (farthest sites from the outlet). The expression used to obtain Hs is:

$$H_i = \frac{C_H}{(1 + r_{io})}, \tag{5}$$

(a) Exponential+Cauchy's Kernel



(b) Exponential Kernel



(c) Fat-tailed radially Symmetrically Kernel

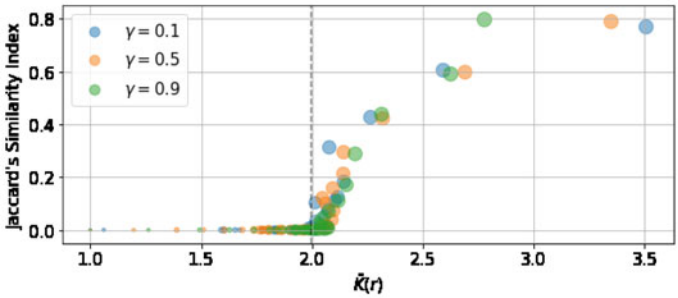


Fig. 3 Jaccard's similarity index as a function of average degree at a distance r. The figure shows a variation of Jaccard's Similarity Index, a measure of Beta Diversity of Species as a function of the average degree of nodes at a distance r. The dispersal of fishes over the network is governed by Kernel set in the neutral model; the top figure is for Exponential + Cauchy's Kernel, the middle is for back-to-back exponential Kernel, and the last one is for fat-tailed Gaussian Kernel. The colours of dots are indicative of the OCN example identified by the γ value. The sizes of the dots are inversely proportional to the magnitude of network distance. The dashed vertical line shows the global average degree of the network

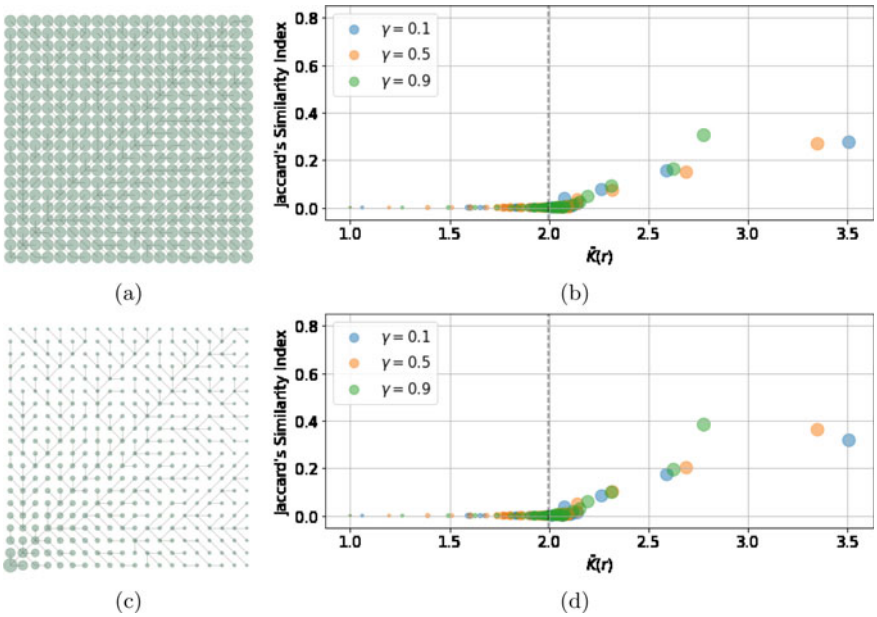


Fig. 4 Jaccard's similarity Index patterns on OCN with higher and non-uniform habitat capacities (H) of sites. Panels **a** and **c** show OCN with $\gamma = 0.9$ where node sizes are scaled according to H of the site. In (**a**), all sites have a habitat capacity of 50, and in (**c**), the habitat capacities are assigned based on distance from the outlet (bottom left site), with the outlet having H = 50 and the farthest point from outlet having H = 2. Panels **b** and **d** show variation of JSI for three different OCNs, all with H as depicted in the corresponding left panels

where $C_H = 50$ and r_{io} is the distance of ith site from the outlet (o). These settings are indicated in Fig. 4a, c. The size of the sites is indicative of habitat capacities. Since the kernel choice does not show much difference in the pattern, we show the JSI results only for the Cauchy+Exponential Kernel in parts (b) and (d) of the figure for all the three OCNs. Overall JSI range has decreased from the H = 1 case, and there is a corresponding rise in local species richness as the sites can now hold more fishes. In both of these settings the JSI again shows transition to non-zero values at critical $\bar{K}(r)$, i.e. at average degree of the network $\bar{K}(r) = 2$.

We also tested these results for a larger extent of dispersal by setting $a = 0.5$ (long-distance dispersal) and $a = 1$ (global dispersal) in the exponential kernel. While in the former case, the spatial communities still get formed, there are no communities formed in the latter case (result not shown). In the long distance dispersal scenario, the JSI as a function of topological distance shows gentler decline from high values to lower ones, and becomes 0 at a higher topological distance. In the second, JSI remains constant at very low non-zero values and shows almost no variation with distance. The first case shows similar results of JSI versus $\bar{K}(r)$, while for the global dispersal, the transition pattern is lost.

4 Discussion and Conclusion

In this work, we simulate the neutral model of biodiversity on OCNs and obtain patterns of species-similarity as a function of network distance. We observed that as one moves away from a given node, the degree of nodes encountered decreases statistically. At distance zero, the $\bar{K}(r)$ is obviously equal to $\langle K \rangle$, but in range $(0 > r <= r_{\max})$, $\bar{K}(r)$ goes from value greater than $\langle K \rangle$ to a value smaller than this. Similarly, a decline with topological distance is also observed for the species-similarity between the basins. We further correlate the obtained species-similarity pattern and degree of the nodes (as a function of network distance), by simultaneously studying them as function of each other.

When directly compared, JSI shows interesting variation with average node degree at a distance ($\bar{K}(r)$). For limited dispersal, we observe that JSI switched from zero to non-zero values as $\bar{K}(r)$ crosses global average network degree $\langle K \rangle$. Hence, $\langle K \rangle$ is the critical network degree at which this transition in JSI occurs. Also, the non -zero distance at which this crossover occurs also indicates the size of a typical ecological community. This transition is further shown to be robust under three choices of dispersal kernels and bifurcation geometries of the OCNs as set by energy exponent γ.

In the simulations of the neutral model, we had a specific choice of the parameters (a, b, d, η, ν, C_H). The first four parameters are chosen to guide the range of dispersal with more preference for local dispersal and have a small chance of long-range dispersal. The ν is chosen to balance extinction and speciation and to avoid the trivial steady states of neutral model. These are 'monodominance by single species' due to extinction of all species except one, and excess speciation such that TSR keeps on rising in the large time limit. C_H parameter sets the average habitat capacity of sites in the system. In most simulations, we chose $C_H = 1$ for computational ease and to visually show community structure on OCN maps. Note that the chosen parameters represent just one set, and in principle, there can be more parameters that work and give the same or similar results. However, in this work, we do not perform large parameter scans. We checked our results on a larger network of size 25×25, and the results still hold. The neutral model could be implemented on still larger networks that would require more computations to reach the system steady-state.

It is important to conduct similar studies in the real river system and look for the observations we report. River networks can be constructed from the geographical data of actual river systems, and species-similarity patterns extracted from the corresponding fish occurrence data of river basins. Further, the species-similarity patterns could be studied as a function of network degree for these river networks. These empirical analyses, however, are beyond the scope of the current paper. It would also be interesting to develop a theoretical framework that naturally reveals such observations. In summary, our work contributes to our present understanding of river network topology and its effect on biodiversity dynamics of fish species.

Acknowledgements This work was partially funded by the Center of Advanced Systems Understanding (CASUS), which is financed by Germany's Federal Ministry of Education and Research (BMBF) and by the Saxon Ministry for Science, Culture, and Tourism (SMWK) with tax funds on

the basis of the budget approved by the Saxon State Parliament. A.R is supported by the research program of the Netherlands Organisation for Scientific Research (NWO).

References

1. Abed-Elmdoust, A., Singh, A., Yang, Z.L.: Emergent spectral properties of river network topology: an optimal channel network approach. Sci. Rep. **7**(1), 1–9 (2017)
2. Balister, P., Balogh, J., Bertuzzo, E., Bollobás, B., Caldarelli, G., Maritan, A., Mastrandrea, R., Morris, R., Rinaldo, A.: River landscapes and optimal channel networks. Proc. Nat. Acad. Sci. **115**(26), 6548–6553 (2018)
3. Borthagaray, A.I., Teixeira-de Mello, F., Tesitore, G., Ortiz, E., Illarze, M., Pinelli, V., Urtado, L., Raftopulos, P., González-Bergonzoni, I., Abades, S., et al.: Community isolation drives lower fish biomass and species richness, but higher functional evenness, in a river metacommunity. Freshwater Biol. **65**(12), 2081–2095 (2020)
4. Brown, B., Swan, C.: Dendritic network structure constrains metacommunity properties in riverine ecosystems. J. Animal Ecol. **79**(3), 571–580 (2010)
5. Carraro, L., Altermatt, F., Fronhofer, E.A., Furrer, R., Gounand, I., Rinaldo, A., Bertuzzo, E., Carraro, M.L.: Package 'ocnet' (2021)
6. Colaiori, F., Flammini, A., Maritan, A., Banavar, J.R.: C. Phys. Rev. E **55**(2), 1298 (1997)
7. Condit, R., Pitman, N., Leigh, E.G., Jr., Chave, J., Terborgh, J., Foster, R.B., Núnez, P., Aguilar, S., Valencia, R., Villa, G., et al.: Beta-diversity in tropical forest trees. Science **295**(5555), 666–669 (2002)
8. Economo, E.P., Keitt, T.H.: Network isolation and local diversity in neutral metacommunities. Oikos **119**(8), 1355–1363 (2010)
9. Etienne, R.S., Rosindell, J.: The spatial limitations of current neutral models of biodiversity. PloS One **6**(3), e14717 (2011)
10. Fang, Y., Ceola, S., Paik, K., McGrath, G., Rao, P.S.C., Montanari, A., Jawitz, J.W.: Globally universal fractal pattern of human settlements in river networks. Earth's Future **6**(8), 1134–1145 (2018)
11. Gilarranz, L.J., Bascompte, J.: Spatial network structure and metapopulation persistence. J. Theor. Biol. **297**, 11–16 (2012)
12. Hubbell, S.P.: The unified neutral theory of biodiversity and biogeography (mpb-32). In: The Unified Neutral Theory of Biodiversity and Biogeography (MPB-32). Princeton University Press (2011)
13. Krioukov, D., Papadopoulos, F., Kitsak, M., Vahdat, A., Boguná, M.: Hyperbolic geometry of complex networks. Phys. Rev. E **82**(3), 036106 (2010)
14. Kuemmerlen, M., Reichert, P., Siber, R., Schuwirth, N.: Ecological assessment of river networks: from reach to catchment scale. Sci. Total Environ. **650**, 1613–1627 (2019)
15. Muneepeerakul, R., Bertuzzo, E., Lynch, H.J., Fagan, W.F., Rinaldo, A., Rodriguez-Iturbe, I.: Neutral metacommunity models predict fish diversity patterns in Mississippi-Missouri Basin. Nature **453**(7192), 220–222 (2008)
16. Muneepeerakul, R., Bertuzzo, E., Rinaldo, A., Rodriguez-Iturbe, I.: Evolving biodiversity patterns in changing river networks. J. Theor. Biol. **462**, 418–424 (2019)
17. Rinaldo, A., Gatto, M., Rodriguez-Iturbe, I.: River networks as ecological corridors: a coherent ecohydrological perspective. Adv. Water Resour. **112**, 27–58 (2018)
18. Terui, A., Ishiyama, N., Urabe, H., Ono, S., Finlay, J.C., Nakamura, F.: Metapopulation stability in branching river networks. Proc. Nat. Acad. Sci. **115**(26), E5963–E5969 (2018)
19. Terui, A., Kim, S., Dolph, C.L., Kadoya, T., Miyazaki, Y.: Emergent dual scaling of riverine biodiversity. Proc. Nat. Acad. Sci. **118**(47) (2021)

Can One Hear the Position of Nodes?

Rami Puzis

Abstract Wave propagation through nodes and links of a network forms the basis of spectral graph theory. Nevertheless, the sound emitted by nodes within the resonating chamber formed by a network are not well studied. The sound emitted by vibrations of individual nodes reflects the structure of the overall network topology but also the location of the node within the network. In this article a sound recognition neural network is trained to infer centrality measures from the nodes' wave-forms. In addition to advancing network representation learning, sounds emitted by nodes are plausible in most cases. Auralization of the network topology may open new directions in arts, competing with network visualization.

Keywords Deep learning · Centrality · Auralization · Diffusion

1 Introduction

Representation learning in graphs, a.k.a embedding, facilitates variety of downstream analysis tasks such as node classification [1], link prediction [2], community detection [3], network classification [4], and more. One challenging application of representation learning is inference of the importance (centrality) of nodes in a network according to their position in the network topology. There were many centrality measures defined over the years [5, 6]. The the most prominent ones are the connectivity degree (or just degree), closeness, betweenness, and eigenvector centrality. Researchers continue inventing new centrality measures and node properties from time to time to fit a particular task that was not properly covered yet [7, 8]. Formulating new structural node properties, including centrality measures, for a task at hand requires domain expertise, network analysis expertise, and of course time and effort. It is important to devise a generic method capable of learning centrality measures provided ground truth importance of nodes.

R. Puzis (✉)
Department of Software and Information Systems Engineering,
Ben-Gurion University of the Negev, Beer-Sheva, Israel
e-mail: puzis@bgu.ac.il

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_50

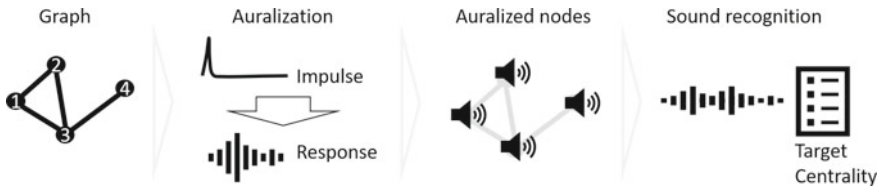


Fig. 1 The pipeline of centrality learning through network auralization and sound recognition

Standard machine learning techniques can utilize previously defined centrality measures to compute a new one [9–11]. However, such approaches inherit the pros and cons of the centrality measures used as features and may not be able to learn an entirely new concept of centrality. Attempts were made to use GNN for centrality learning. Maurya et al. [12] proposed architectures that do not rely on pre-computed centrality measures but, unfortunately, they use different GNN architectures for learning closeness and betweenness measures, falling back to relying on human expertise. A recently proposed method utilizes the generic nature of routing betweenness centrality to learn arbitrary centrality measures [13]. Unfortunately, this approach does not scale well to large networks.

Spectral graph analysis is a widely accepted tool for graph mining, comparison, and classification [14–17]. Eigenvalues represent the stationary frequencies (spectrum) of heat or wave propagation in a graph. The respective eigenvector components are used as a form of node representation for classification and clustering of nodes. In most cases this representation includes a part of the stationary state corresponding to the top k eigenvalues. In this article we consider the full wave-form-based representation of nodes rather then their spectrum and use sound recognition neural network to hear their centrality.

High level overview. The centrality learning approach proposed in this article relies on wave propagation along the links of the network. Instead of focusing on the stationary spectrum we monitor the impulse response of the network. Natural reflection and interference that within the network results in complex audible wave-forms that auralize the nodes and their positions within the network. We use the acoustic characteristics of the nodes to learn centrality measures using the M5 deep neural architecture for sound recognition [18]. Figure 1 summarizes the general architecture of the demonstrated centrality learning approach.

1.1 Summary of Contributions

Following is the list of preliminary results reported in this paper and their respective significance. This article shows that wave-form network analysis facilitates non-trivial downstream tasks such as centrality learning. The results show that the M5 sound recognition neural network [18] can learn centrality measures for variety of

network models and centrality measures. Yet, lattice networks, in particular Watts-Strogats [19], are challenging as well as learning the closeness centrality measure. Our preliminary results advance state-of-the-art centrality learning making a few more steps toward automated inference of this important concept. The presented technique exemplifies the importance of non-stationary pre-convergence state of network signal propagation. finally this paper demonstrates a surprising combination of graph analysis with sound recognition neural network, hoping to inspire additional architectural solutions for classification problems on graphs.

2 Related Work on Centrality Learning

2.1 Traditional Machine Learning

In recent years, with the advancement in the machine and deep learning fields, researchers utilized many of these algorithms to approximate the centrality of graph's nodes. Most works in this category infer some centrality measures from other centrality measures. In 2018, Grando et al. [9] rely on the degree and on the eigen vector centrality to learn other centrality measures. Mendonca et al. [10] improved the work presented by Grando et al. by proposing the NCA-GE model. The NCA-GE architecture utilizes Strucre2Vec and Graph Convolution Network (GCN) for generating a high dimensional feature vector for each node in a given graph. To these generated node embeddings, they added the degree centrality as an additional dimension. Finally, they utilized the embeddings obtained to approximate node centrality. Zhao et al. [11] detected influential nodes based on various features including nine centrality measures. The target measure was the influence of a node as simulated by epidemic propagation. Relying on pre-computed centrality measures helps inferring correlated measures, but might not generalize well to approximate un-correlated measures or tasks that require learning new measures.

2.2 Graph Neural Networks

Maurya et al. [12] proposed GNN-Bet and GNN-Close, two graph neural networks to approximate betweenness and closeness centralities respectively. Fan et al. [20] proposed a graph neural network encoder-decoder ranking model to identify nodes having the highest betweenness. Last year, Bachar et al. [13] proposed LRC, a centrality learning architecture that relies on routing betweenness centrality [21]. While being generic LRC is heavy computation-wise and can learn centrality in graphs with only dozens of nodes. Moreover, it requires high quality geometric embedding to produce accurate results.

Overall centrality learning must be *inductive*: trained and tested on different graphs. Centrality learning should be tested on real and synthetic networks with varying structures. It should be *generic* and suitable for arbitrary target centrality measures and preferably *scalable* facilitating applications on graphs larger than those it was trained on.

3 Methods

3.1 Network Auralization

In his subsection a simple wave-form generation process is described. Let $G = (V, E)$ be a simple undirected unweighted graph where V is a set of n nodes and E is a set of m edges.

Consider some quantity $s_{v,t} \in \mathbb{R}$ possessed by every node $v \in V$ at time t. Intuitively $s_{v,t}$ can be regarded as a potential of the node. $S_t = (s_v : v \in V)$ is the vector of potentials. Nodes strive to equalize their potentials by distributing energy to their neighbors. Please note that, although, some physical terms are used here to describe the network analysis they are not intended to discuss a real physical phenomenon and lack the rigorousity expected from a physics article.

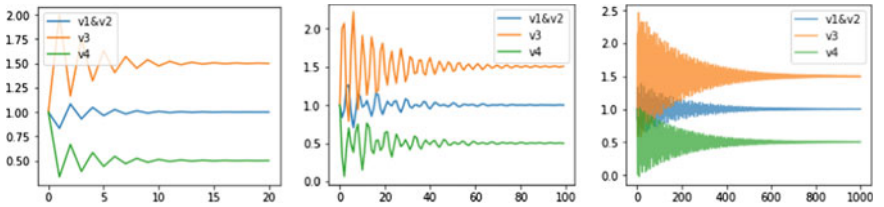
Let A denote the adjacency matrix of the graph G. Let $D = \sum_v A_v$ denote the vector of node degrees. We define $P_{u,v} = A_{u,v}/D_u$ as the power that u may apply on v. The total power a node may apply on its neighbors is equal for all nodes. The more neighbors a node has the less power it can apply on each one of them.

The amount of energy every node u passes to every neighbor u at time t is $\Delta S_{t,u,v} = S_{t-1,u} \cdot P_{u,v}$. In matrix form $\Delta S_t = Diag(S_{t-1}) \times P$, where $Diag(S_t)$ is a $n \times n$ matrix with values of S_t along the main diagonal. The incoming energy flow of a node v is $\sum_{u \in V} \Delta S_{t,u,v}$ and the outgoing energy flow is $\sum_{u \in V} \Delta S_{t,v,u}$. Note that $\Delta S_{t,u,v} = 0$ if u and v are not neighbors. Equation 1 describes a simple diffusion process where nodes exchange energy up to a certain fixed point, much like in the power iterations method.

$$S_t = S_{t-1} + \sum_{u \in V} \Delta S_{t,u,v} - \sum_{u \in V} \Delta S_{t,v,u} \tag{1}$$

Unlike power iterations Eq. 1 does not require normalizing the potential vector in every iteration due to energy conservation ($\sum S_t$ is constant).

The stable fixed point of the energy exchange iterations is not of interest for current paper. Let us take a close look at the dynamics of the energy exchange before the process stabilizes (see Fig. 2a). The plot shows the potential levels of nodes from the graph in Fig. 1. On the first iteration the potential of v_1 increases the most because it has low multiple low degree neighbors. The potential of other nodes decreases after the first iteration because they contribute more energy than they receive. It is



(a) No momentum (b) Momentum $m = 0.9$ (c) Momentum $m = 0.99$

Fig. 2 Energy exchange interactions with various levels of momentum

Algorithm 1: Network auralization

Input: m: momentum, A: adjacency matrix, l: number of output samples
Output: S: the $l \times n$ matrix representing node wave forms

$\forall_{v \in V} S_{0,v} = 1$; // Impulse
$\forall_{u,v \in V} \Delta S_{0,u,v} = 0$;
$P = (A/(A.sum(dim = 0) + \epsilon).T)$; // $\epsilon = 10^{-32}$ for numeric stability.
for $t \in [1, \ldots, l]$ **do**
  $\Delta S_t = Diag(S_{t-1}) \times P + m \cdot \Delta S_{t-1}$;
  $S_t = S_{t-1} + \Delta S_t.sum(dim = 0) - \Delta S_t.sum(dim = 1)$; // Response
end
$S = (S.T - S.T.mean(dim = 0)).T$; // Remove the DC component of the wave-forms.
return S

hard to see from this plot but, the energy from node v_4 reaches v_1 and v_2 after the second iteration and then bounces back to v_3 because it is the only neighbor of v_4. Nevertheless, it is clear that the location of the nodes within the network affects the magnitude and direction of the oscillations.

Next the oscillations are emphasized and the stabilization process is prolonged by retaining a portion m of the energy flow from previous iteration. We will refer to m as momentum. The energy flow with momentum is now represented by:

$$\Delta S_{t,u,v} = S_{t-1,u} \cdot P_{u,v} + m \cdot \Delta S_{t-1,u,v}. \tag{2}$$

Note that adding momentum to ΔS does not affect the energy preservation in Eq. 1.

Figure 2b, c show the oscillations with $m = 0.9$ and $m = 0.99$ respectively. We can see that the stabilization process is significantly prolonged. We can also see in Fig. 2b irregularities caused by interference and reflection as explained in the spectral analysis literature. Setting $m = 1$ will prevent the process from stabilizing.

Algorithm 1 presents the pseudo code of network auralization adapted for PyTorch implementation.[1] The operator T is matrix transpose. The operators sum and $mean$ aggregate elements of a matrix along the dimension specified by dim. The DC component of the output wave-forms (values on which S stabilizes after impulse response) is not of interest. It may also hinder the convergence of sound recognition

[1] The full source code is available on GitHub: https://github.com/puzis/centrality-learning.

models. Furthermore, it is a centrality measure on its own, roughly corresponding to eigenvector centrality. Thus we remove the DC component in Line 8 of Algorithm 1 in order show that the wave-form itself bears significant information about the location of a node within the graph.

3.2 Centrality Learning as a Sound Recognition Problem

Architecture Sound is the most common and most studied form of waves. Many deep convolutional neural networks were developed in the past years to recognize speech [22], emotions [23], background sounds [18], etc. Current study relies on the M5 very deep convolutional neural network proposed by Dai et al. for recognition of environmental sounds in urban areas [18]. There are two important details about their neural network architecture that should be mentioned here. First, the filter size of their first convolutional layer is set to 80, a sufficiently large value to cover the common wavelengths of natural sounds. Second, their last layers include global pooling and softmax activation function with 10 outputs to produce a classifier with 10 target classes.

In current study the M5 classifier is transformed into a regression architecture by replacing the softmax with a fully connected linear layer. Preliminary experiments with various activation functions showed that linear activation produces the best results in terms of Pearson correlation coefficient. Also not tested with other sound recognition architectures replacing softmax with a fully connected linear layer is a common tweak for turning a classifier into a regression model. We consider the M5 regressor as a function that maps node's wave-form to a real number: $M5\colon \mathbb{R}^l \rightarrow \mathbb{R}$. The learned centrality measure of the node $v \in V$ is therefore $M5(S_{\cdot,v})$ where $\cdot$ represents all possible values. $M5(S)$ is the centrality vector of all nodes in G.

Objective function Let C denote the target centrality measure and P denote the predicted centrality values. The target variable for training the M5 regressor was chosen to be the Pearson correlation coefficient:

$$\rho(C, P) = \frac{cov(C, P)}{std(C) \cdot std(P)}$$

. The loss for training the M5 neural network is:

$$loss = 1 - \rho(C, M5(S)) \tag{3}$$

Correlations are common performance indicators for centrality learning [9, 10, 12, 13]. Pearson correlation coefficient has the most accessible differentiable implementation in the PyTorch deep learning library. Other correlation coefficients can be used as long as they have differentiable implementations. A notable one, is differential implementation of Spearman correlation coefficient [24].

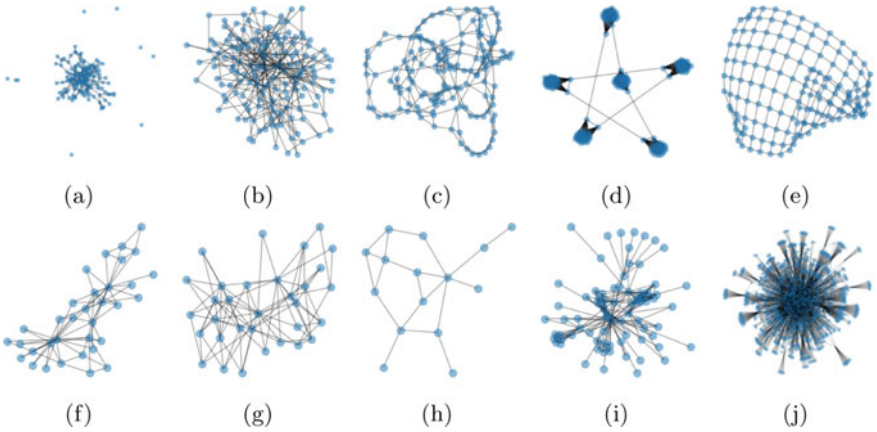


Fig. 3 Examples of some random and well known networks. On the top: Erdos-Renyi (ER) random graph [25], Barabasi-Albert (BA) scale-free graph [26], Watts-Strogatz (WS) small-world graph [19], connected caveman graph [27], and a regular grid. On the bottom: Karate club [28], Southern women graph [29], Florentine families graph [30], Les Miserables [31], Autonomous Systems level random Internet graph [32]

The training procedure In this subsection we consider the general task of centrality learning. Connectivity degree (Deg), closeness centrality (CC), eigenvector centrality (EC), and betweenness centrality (CC) are considered as the target centrality measures to demonstrate the learning process.

It is evident from past research and from the results in this article that certain kinds of networks are harder to learn than others. Therefore the models are trained on five different random network models shown in Fig. 3a–e. All training was performed on small graphs of 150 nodes and various densities.

The training phase was split to epochs (see Algorithm 2). In every epoch a set of small random graphs were generated. The number of graphs grows linearly with epochs (line 3). Graphs are generated from all the five models considered for training. Graphs are auralized in line 8. Next, ten iterations are made to optimize the parameters of the M5 regression model. The numbers in lines 3 and 9 were chosen empirically and are subject to experiments in future research. Multiple optimization steps on a set of random graphs (lines 9–12) are required, otherwise, the M5 model parameters fail to converge. As the model converges the batch size (number of random graphs) need to increase in order to reduce the noise in the optimization process.

The testing procedure. New random graphs are generated for the testing phase. Three types of test sets are considered:

1. Small random graphs ($n = 150$) similar to the training process.
2. Larger random graphs ($n = 1500$) generated using the same models.
3. Four famous graphs and a random Internet topology as depicted in Fig. 3f–j.

Algorithm 2: The centrality learning process

for *epoch* $\in [1, \ldots, 300]$ **do**
  $Gs \leftarrow \emptyset$; // graphs for current epoch
  for $10 + \lfloor \frac{epoch}{10} \rfloor$ *times* **do**
    $Gen \leftarrow$ pick a random graph generator;
    $G \leftarrow$ generate a random graph using Gen;
    $Gs \leftarrow Gs \cup \{G\}$;
  end
  for $G \in Gs$ **do** $S(G) \leftarrow$ network auralization (G) **for** *10 times* **do** // 10 batches on the same graphs
    $batchloss \leftarrow \frac{1}{10 \cdot |Gs|} \sum_{G \in Gs} (1 - \rho(M5(S(G))))$;
    $batchloss$ backprogation;
    optimizer step;
  end
end
return S

Pearson correlation was computed between the centrality leaned using the M5 sound recognition neural network and the ground truth centrality measures for each one of the test graphs.

Models trained on the small networks were applied on the large networks as well. This is possible due to the size-invariant representation of the nodes' wave-forms in time domain. The wave-forms of larger networks usually exhibit more frequencies (a richer sound) corresponding to a larger number of eigenvalues. We use fixed-size time series of 10K samples as the input for the M5 sound recognition network.

4 Results and Discussion

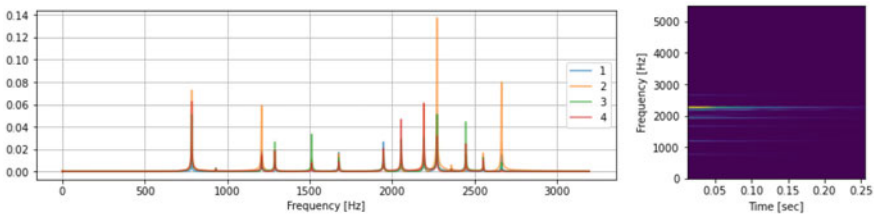
4.1 The Voice of Graphs and Nodes

The echo chambers formed by graphs produce sound patterns as can be seen and heard in the multimedia Fig. 4. On the left side of the spectrum of nodes depicted. While peak frequencies remains the same for different nodes in the graph their intensities vary. The proposed centrality learning approach relies on these differences.

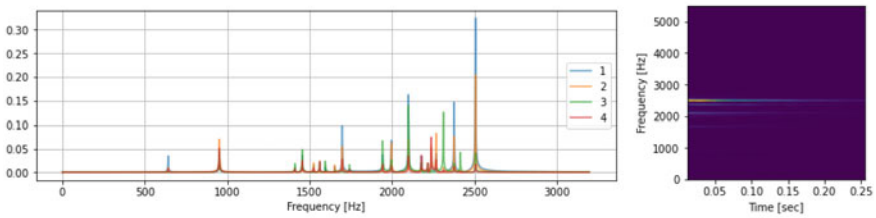
4.2 Centrality Learning Results

Figure 5 shows the training loss while learning nodes' connectivity degree from their auralization. It is clear that node auralization contains information about connectivity degree.

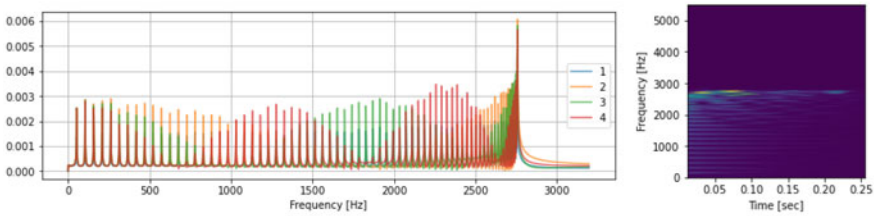
Table 1 presents the Pearson correlation coefficients while testing the trained M5 sound recognition model. Naturally, degree is the easiest to learn, but the correlation



(a) Florentine families Click to play (all nodes, 13 seconds)



(b) Karate club Click to play (all nodes, 30 seconds)



(c) Line graph with 150 nodes Click to play (20 nodes, 18 seconds)

Fig. 4 The voice of graphs and nodes. Left: the spectra of four nodes in a graph. Right: a spectrogram of one arbitrary node

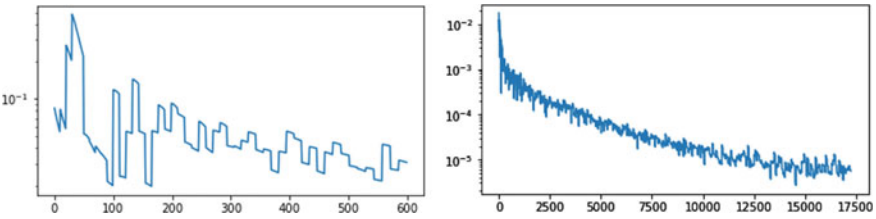


Fig. 5 The loss $(1 - \rho(M5(S)))$ as a function of the number of optimization steps when learning the connectivity degree. Left: 50 epochs. Right: 500 epochs

coefficients for other centrality measures are generally better than the respective correlations with other centrality measures (0.48–0.91 according to [13]). This fact suggests that the wave-forms produced by Algorithm 1 encode information about the position of nodes within their networks.

Table 1 Performance of the M5 sound recognition neural network when predicting centrality of auralized nodes on various network

	ER		BA		WS		Grid		Caveman	
	Small	Large	Small	Large	Small	Large	Small	Large	Small	Large
Degree	0.9907	0.987	0.9954	0.9947	0.9931	0.9868	0.9723	0.946	0.9843	0.9565
Closeness	0.9741	0.7717	0.8919	0.9223	0.8232	0.6526	0.999	0.6752	0.9997	0.9677
Betweenness	0.9451	0.9767	0.99	0.9724	0.8446	0.6976	0.998	0.7346	0.9999	0.9365
Eigenvector	0.9287	0.8291	0.9563	0.9634	0.7656	0.6867	0.9984	0.4441	0.999	0.9965
	Karate club		Southern women		Florentine families		Les Miserables		Random Internet	
Degree	0.988		0.9856		0.9897		0.8679		0.9931	
Closeness	0.8881		0.7071		0.6787		0.3855		0.7956	
Betweenness	0.9423		0.9843		0.9293		0.7505		0.974	
Eigenvector	0.9111		0.8806		0.8991		0.7462		0.8767	

Grid and Cavemen graphs show extreme results due to an over-fitting and sample leakage from train to test due to their low variability. The M5 network memorizes the node wave-forms and fails to extrapolate the learned centrality to larger networks. This result highlights the fact that the model successfully extrapolates to networks an order of magnitude larger than those it was trained on.

While performing reasonably well on most small real world networks and on the random Internet graph, the learned model fails on the Les Miserables social network. Even the predicted degree of nodes is loosely correlated with the actual degree. I could not find a reasonable explanation for such behavior.

5 Conclusion

In this paper, demonstrates a surprising application of sound recognition neural network for learning centrality measures from auralized nodes in graphs. The demonstrated model learns to infer centrality from the sound of nodes. The model extrapolates well to networks larger than network in its training set. This article also demonstrates by an example that the stabilization process of graph algorithms bears information about the graph structure. It may form a basis for new types of message passing graph neural networks where the energy exchange parameters are learned during the training process. Future work includes investigating a transfer learning problem where centrality learned from one kind of graphs (e.g. ER) is adjusted to fit graphs with significantly different structure (e.g. WS). Another theme with a lot of fun and unexplored research opportunities is natural graph auralization, not to be confused with artificial/engineered sonification.

References

1. Ma, X., Qin, G., Qiu, Z., Zheng, M., Wang, Z.: Riwalk: fast structural node embedding via role identification. In: 2019 IEEE International Conference on Data Mining (ICDM), pp. 478–487. IEEE (2019)
2. Mallick, K., Bandyopadhyay, S., Chakraborty, S., Choudhuri, R., Bose, S.: Topo2vec: a novel node embedding generation based on network topology for link prediction. IEEE Trans. Comput. Soc. Syst. **6**(6), 1306–1317 (2019)
3. Sun, F.-Y., Qu, M., Hoffmann, J., Huang, C.-W., Tang, J.: vgraph: a generative model for joint community detection and node representation learning. Adv. Neural Inf. Process. Syst. **32** (2019)
4. Riesen, K., Bunke, H.: Graph classification based on vector space embedding. Int. J. Pattern Recogn. Artif. Intell. **23**(06), 1053–1081 (2009)
5. Rodrigues, F.A.: Network Centrality: An Introduction, pp. 177–196. Springer International Publishing, Cham (2019)
6. Freeman, L.C.: Centrality in social networks conceptual clarification. Soc. Netw. **1**(3), 215–239 (1978)
7. Qi, X., Fuller, E., Qin, W., Yezhou, W., Zhang, C.-Q.: Laplacian centrality: a new centrality measure for weighted networks. Inf. Sci. **194**, 240–253 (2012)
8. Abbasi, A., Hossain, L.: Hybrid Centrality Measures for Binary and Weighted Networks, pp. 1–7. Springer, Berlin, Heidelberg (2013)
9. Grando, L.C.L.F., Granville, L.Z.: Machine learning in network centrality measures: tutorial and outlook. ACM Comput. Surv. **51**(5), 102 (2018)
10. Mendonça, M.R.F., Barreto, A., Ziviani, A.: Approximating network centrality measures using node embedding and machine learning. ACM **57** (2020)
11. Zhao, G., Jia, P., Huang, C., Zhou, A., Fang, Y.: A machine learning based framework for identifying influential nodes in complex networks. IEEE Access (2020). https://doi.org/10.1109/ACCESS.2020.2984286
12. Maurya, S.K., Liu, X., Murata, T.: Graph neural networks for fast node ranking approximation. TKDD **15**(5), 1–32 (2021)
13. Bachar, L., Elyashar, A., Puzis, R.: Learning centrality by learning to route. In: International Conference on Complex Networks and Their Applications, pp. 247–259. Springer (2021)
14. Von Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. **17**(4), 395–416 (2007)
15. Sahai, T., Speranzon, A., Banaszuk, A.: Hearing the clusters of a graph: a distributed algorithm. Automatica **48**(1), 15–24 (2012)
16. Avrachenkov, K., Jacquet, P., Sreedharan, J.K.: Distributed spectral decomposition in networks by complex diffusion and quantum random walk. In: IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, pp. 1–9. IEEE (2016)
17. Tsitsulin, A., Mottin, D., Karras, P., Bronstein, A., Müller, E.: Netlsd: hearing the shape of a graph. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2347–2356 (2018)
18. Dai, W., Dai, C., Qu, S., Li, J., Das, S.: Very deep convolutional neural networks for raw waveforms. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 421–425. IEEE (2017)
19. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature **393**(6684), 440–442 (1998)
20. Fan, C., Zeng, L., Ding, Y., Chen, M., Sun, Y., Liu, Z.: Learning to identify high betweenness centrality nodes from scratch: a novel graph neural network approach. In: CIKM, pp. 559–568 (2019)
21. Dolev, S., Elovici, Y., Puzis, R.: Routing betweenness centrality. J. ACM (JACM) **57**(4), 1–27 (2010)
22. Bahar, P., Bieschke, T., Ney, H.: A comparative study on end-to-end speech to text translation. In: 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pp. 792–799. IEEE (2019)

23. Abdullah, S.M.S.A., Ameen, S.Y.A., Sadeeq, M.A.M., Zeebaree, S.: Multimodal emotion recognition using deep learning. J. Appl. Sci. Technol. Trends **2**(02), 52–58 (2021)
24. Blondel, M., Teboul, O., Berthet, Q., Djolonga, J.: Fast differentiable sorting and ranking. In: International Conference on Machine Learning, pp. 950–959. PMLR (2020)
25. Erdős, P., Rényi, A., et al.: On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci **5**(1), 17–60 (1960)
26. Barabási, A., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)
27. Watts, D.J.: Networks, dynamics, and the small-world phenomenon. Am. J. Sociol. **105**(2), 493–527 (1999)
28. Zachary, W.W.: An information flow model for conflict and fission in small groups. J. Anthropol. Res. **33**(4), 452–473 (1977)
29. Davis, A., Gardner, B.B., Gardner, M.R.: Deep South: A Social Anthropological Study of Caste and Class. Univ. of South Carolina Press (2009)
30. Breiger, R.L., Pattison, P.E.: Cumulated social roles: the duality of persons and their algebras. Soc. Netw. **8**(3), 215–256 (1986)
31. Knuth, D.E.: The Stanford GraphBase: a platform for combinatorial computing. ACM (1993)
32. Elmokashfi, A., Kvalbein, A., Dovrolis, C.: On the scalability of BGP: the role of topology growth. IEEE J. Sel. Areas Commun. **28**(8), 1250–1261 (2010)

Memory Based Temporal Network Prediction

Li Zou, An Wang, and Huijuan Wang

Abstract Temporal networks are networks like physical contact networks whose topology changes over time. Predicting future temporal network is crucial e.g., to forecast and mitigate the spread of epidemics and misinformation on the network. Most existing methods for temporal network prediction are based on machine learning algorithms, at the expense of high computational costs and limited interpretation of the underlying mechanisms that form the networks. This motivates us to develop network-based models to predict the temporal network at the next time step based on the network observed in the past. Firstly, we investigate temporal network properties to motivate our network prediction models and to explain how the performance of these models depends on the temporal networks. We explore the similarity between the network topology (snapshot) at any two time steps with a given time lag/interval. We find that the similarity is relatively high when the time lag is small and decreases as the time lag increases. Inspired by such time-decaying memory of temporal networks and recent advances, we propose two models that predict a link's future activity (i.e., connected or not), based on the past activities of the link itself or also of neighboring links, respectively. Via seven real-world physical contact networks, we find that our models outperform in both prediction quality and computational complexity, and predict better in networks that have a stronger memory. Beyond, our model also reveals how different types of neighboring links contribute to the prediction of a given link's future activity, again depending on properties of temporal networks.

Keywords Temporal network prediction · Network-Based prediction · Temporal network property

L. Zou · H. Wang (✉)
Faculty of Electrical Engineering, Mathematics and Computer Science,
Delft University of Technology, Delft, The Netherlands
e-mail: h.wang@tudelft.nl

A. Wang
Department of Chemistry, University of Warwick, Coventry, UK

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_51

1 Introduction

Complex systems can be represented as networks, where nodes represent the components of a system and links denote the interaction or relation between the components. The interactions are, in many cases, not continuously active. For example, individuals connect via email, phone call or physical contact at specific times instead of constantly. Temporal networks [1, 2] could represent these systems more realistically with time-varying network topology. It has been shown that temporal network properties such as community structure, the degree distribution in the aggregated network, inter-event time and the non-Markovian evolution influence dynamic processes on the temporal network [3–6].

Temporal network prediction is a task of predicting temporal contacts at the next time step based on the temporal network topology observed in the previous L steps. Predicting the temporal network such as a physical contact network is essential to forecast and mitigate the spread of epidemics and misinformation on the network. The temporal network prediction problem is also equivalent to problems in recommender systems, e.g., predicting which user will purchase which product, which individuals will become acquaintance [7, 8].

Recently, machine learning algorithms have been developed to predict temporal networks. Examples include temporal network embedding [9–11], restricted Boltzmann machine (RBM) based methods [12] and Graph neural networks [13]. These methods, however, are at the expense of high computational costs and limited in providing insights regarding which network mechanisms are used for network prediction thus could possibly form temporal networks. Few network-based methods have been proposed to predict new links, i.e., the node pairs that will have contact in the future but have not had any contact in the past, instead of predicting all contacts at a future time step. These network-based methods consider a network property, also called similarity, of a node pair as the tendency that a new link will appear between the node pair [14–16]. Initial network-based methods for temporal network prediction have been explored recently, assuming that a link is more likely to have a contact (be active) in the future if it has contacts recently, depending possibly on the the previous contacts of other neighboring links [17].

However, we still lack deep understanding of how to design network-based methods to predict temporal networks and how the performance of prediction methods depend on properties of temporal networks. Hence, this work aims to explore basic temporal network properties, to motivate the design of network-based temporal network prediction methods and to explain these methods' network dependent performance. Firstly, we explore the similarity between the activity (i.e., connected or not) of a link at any two time steps with a given time lag/interval and the similarity between the network topology (snapshot) at any two time steps with a given time lag. Intuitively, if such similarity is relatively high, thus there exists memory in temporal networks, we may predict a temporal network in the future based on the network observed in the past. We find both similarities, or memories, decay as the time lag

increases and they are relatively high when the time lag is small. Correspondingly, we propose two temporal network prediction models utilizing the observed time decaying memory in temporal networks.

Our first model, called the self-driven (SD) model, assumes that a link's future state (connected or not) is only influenced by it's past states and the influence of its earlier state is smaller than that of more recent states. Specifically, it assumes that the tendency for link i being connected or active at time step $t+1$ is given by $w_i(t+1) = \sum_{k=t-L+1}^{k=t} e^{-\tau(t-k)} x_i(k)$, where $x_i(k)$ is the state of link i at time step k and τ is the decay factor controlling the contribution from each past state. This definition and concept is not new, and has been used in [17, 18]. The SD model is emphasized as one model here because we will explore in depth the decay factor and its implications and it is the basis to built our SCD model. We find the SD model performs well in network prediction when the decay factor is chosen arbitrarily in the broad range $\tau \in [0.5, 5]$ in each of the seven real-world physical contact temporal networks. This implies that our real-world physical contact networks measured in the context of school, hospital, workplace etc. may be formed by a universal class of time decaying memory. This common range of the decay factor $\tau \in [0.5, 5]$ suggests that the state of a link is mainly determined by the link's states in few recent steps.

Furthermore, we generalize the SD model to a self- and cross-driven (SCD) model that predicts a link's next step activity by using the SD connection tendency of the link itself and also of the other neighbor links. We find that SCD outperforms SD and both SCD and SD perform better than the baseline models such as linear regression. Both models perform better in networks with a stronger memory. The SCD model also reveals how different types of neighboring links contribute to the prediction of a given link's future activity, which we find also depend on temporal network properties.

We will introduce the presentation of temporal networks (Sect. 2), real-world temporal networks to be considered (Sect. 3), analyze key temporal networks (Sect. 4) to motivate our temporal network prediction models (Sect. 5). The proposed models will be evaluated and interpreted in Sects. 6 and 7 respectively.

2 Temporal Network Representation

A temporal network can be represented as a sequence of network snapshots $G = \{G_1, G_2, ..., G_T\}$, where T is duration of the observation window, $G_t = (V; E_t)$ is the snapshot at time step t with V and E_t being the set of nodes and contacts, respectively. If node j and k have a contact at time step t, $(j, k) \in E_t$. Here, we assume all snapshots share the same set of nodes, i.e., V. The links in the aggregated network G^w are defined as $E = \cup_{t=1}^{T} E_t$. That is, a pair of nodes is connected with a link in the aggregated network if at least one contact occurs between them in the temporal network. Hence, the link set E in the aggregated network contains all the node pairs that have contact(s) in the temporal network and the total number of links is $M = |E|$. We give each link in the aggregated network an index i, where

Table 1 The number of nodes ($N = |V|$), the number of node pairs that have contact(s) (M), the length of the observation time window (T), time resolution (δ s), the type of contacts and the location where the data is collected

Network	N	M	T	δ	Type	Location
Hospital	75	1139	9453	20	Physical	Hospital
Hypertext2009	113	2196	5246	20	Physical	Conference
Workplace	92	755	7104	20	Physical	Office
LH10	73	1381	12605	20	Physical	Hospital
HighSchool	327	5818	7375	20	Physical	School
PrimarySchool	242	8317	3100	20	Physical	School
SFHH	403	9565	3509	20	Physical	Conference

$i \in [1, M]$. The temporal connection or activity of link i over time could then be represented by a T-dimension vector $\mathbf{x_i}$ whose element is $x_i(t)$, where $t \in [1, T]$, $x_i(t) = 1$ when node pair i has a contact at time t and $x_i(t) = 0$ if no contact occurs at t.

3 Empirical Data Sets

To design and evaluate temporal network prediction methods, we consider seven empirical physical contact networks: Hospital, Workplace, PrimarySchool, HighSchool, LH10 [19], SFHH [20] and Hypertext2009 [21]. Basic properties of these data sets are given in Table 1. The time steps at which there is no contact in the whole network have been deleted.

4 Memory in Temporal Networks

In this section, we aim to understand whether a temporal network at different times shares certain similarity or has memory.

Auto-correlation Firstly, we explore the correlation of the activity of a link at two times with a given interval Δ, called time lag, via the auto-correlation of the activity series of each link. The auto-correlation of a time series is the Pearson correlation between the given time series and its lagged version. We compute, for each link i, the Pearson correlation coefficient $R_{x_i x_i}(\Delta)$ between $\{x_i(t)\}_{t=1,2,\ldots,T-\Delta}$ and $\{x_i(t)\}_{t=\Delta+1,\Delta+2,\ldots,T}$ as its auto-correlation coefficient. Figure 1a shows that the average auto-correlation coefficient over all links decays with the time lag Δ in each of the seven data sets. The average auto-correlation decays slower as the time lag increases.

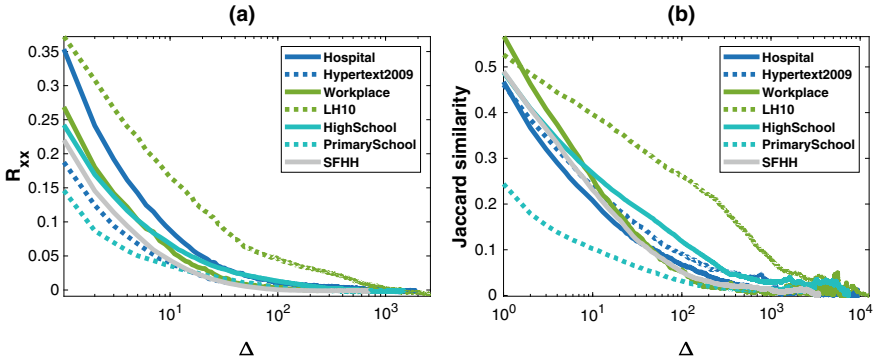


Fig. 1 **a** The average auto-correlation coefficient R_{xx} over all links as a function of the time lag Δ and **b** the average Jaccard similarity of two snapshots of a temporal network with a given time lag Δ in each of the seven data sets

Jaccard similarity Furthermore, the similarity of the network at two times with a given time lag Δ is examined via Jaccard similarity (JS). JS measures how similar two sets are by considering the percentage of shared elements between them. For two snapshots of a temporal network G_t and $G_{t+\Delta}$, their Jaccard similarity is defined as the size of their intersection in contacts divided by the size of the union of their contact sets, that is, $JS(G_t, G_{t+\Delta}) = \frac{E_t \cap E_{t+\Delta}}{E_t \cup E_{t+\Delta}}$. Large JS means large overlap/similarity between the two snapshots of the temporal network. Figure 1b shows the average Jaccard similarity over all possible pairs of temporal network snapshots that have a time lag Δ. Similar to auto correlation in link activity, the correlation between temporal snapshots decays with their time lag in all empirical data sets, manifesting the time decaying memory of real-world temporal networks.

5 Temporal Link Prediction Methods

Inspired by the time decaying memory of temporal networks, we propose two temporal link prediction models. Our previous work on Lasso Regression [22], a statistical learning model has found that a link's state at the next step is largely determined by the current state of the link itself and the neighboring links that share a common node with the link. Hence, our two network-based models in this work will predict a link's future activity based on the past activities of the link itself, and also of the neighboring links respectively by taking the memory effect into account.

5.1 Self-Driven (SD) Model

The self-driven (SD) model defines the tendency $w_i(t+1)$ for link i being active at time $t+1$ as:

$$w_i(t+1) = \sum_{k=t-L+1}^{k=t} e^{-\tau(t-k)} x_i(k). \tag{1}$$

where the decay factor τ controls the rate of the memory decay and $x_i(k)$ is the state of link i at time step k. A large τ corresponds a fast decay of memory, such that a small number of previous states affect the tendency of connection. When $\tau = 0$, all past states have equal influence on the future connection tendency and $w_i(t+1)$ reduces to the contact number of link i during the past L steps. Such exponential decay has also been considered in [17, 23]. In Sect. 6, we will show that the SD model performs well for a common wide range of the decay factor τ among all real-world networks considered and we do not need to learn τ from the temporal network observed in the past.

5.2 Self- and Cross-Driven (SCD) Model

Furthermore, we generalize the SD model to a self- and cross-driven (SCD) model that predicts a link's next step activity by using the SD connection tendency defined in Eq. (1) of the link itself and also of neighboring links that share a node with the link in the aggregated network. The union of the target link and its neighboring links is also called the ego-network centered at the target link, exemplified in Fig. 2. Furthermore, we differentiate three types of links in an ego-network, colored in differently in Fig. 2: the target link itself, links that form a triangle with the target link and the rest links. We believe the previous states of these three types of links may contribute differently to the estimation of the target link's next step activity, motivated by the finding of the Lasso Regression in temporal network prediction [22], the common neighbor similarity method in static network prediction and temporal motifs (e.g., three contacts that happen within a short duration and with a specific ordering in time, and form a triangle in topology) that have been widely observed in temporal networks [24, 25].

Hence, our SCD model assumes that the SCD tendency $h_i(t+1)$ for link i to be active at time step $t+1$ is a linear function

$$h_i(t+1) = \beta_0^* + \beta_1^* w_i(t+1) + \beta_2^* u_i(t+1) + \beta_3^* f_i(t+1). \tag{2}$$

of the contributions of the link itself $w_i(t+1)$ as defined in Eq. (1), the neighboring links that form a triangle with the target link $u_i(t+1)$ and the other neighboring links $f_i(t+1)$. The latter two factors $u_i(t+1)$ and $f_i(t+1)$ will be defined soon as a function of the SD tendency at $t+1$ of all the links in the ego-network.

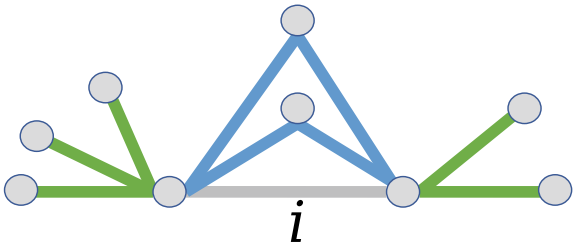


Fig. 2 An illustrative example of an ego-network centered at a target link i. The three types of links, the target link itself, links that form a triangle with the target link and the other links, are colored in grey, blue and green respectively

The contribution $u_i(t+1)$ of the neighboring links that form a triangle with the target link i is defined as follows. For each pair of neighboring links j and k that form a triangle with the target link i, the geometric mean $\sqrt{w_j(t+1) \cdot w_k(t+1)}$ suggests the strength that the two end nodes of link i interact with the corresponding common neighbor. We define $u_i(t+1)$ as the average geometric mean over all link pairs that form a triangle with the target link, a weighted version of common neighbor similarity. The contribution of the other links $f_i(t+1)$ in the ego-network is defined as the average SD tendency of connection. The coefficients β_0^*, β_1^*, β_2^*, and β_3^* in Eq.(2) will be learned through Lasso Regression from the temporal observed in the past L steps for each link. Using previous states of neighboring links to predict the future connection of a link has been explored in [17]. The design of SCD model in e.g., $u_i(t+1)$ aims to capture the weighted version of common neighbor similarity, which enables us later to discover the relation between model performance and the clustering coefficient of the aggregated network.

5.3 *Baseline Models*

Here, we introduce two baseline models.

Common neighbor similarity (CN). We generalize the common neighbor similarity method from static network prediction to the temporal network prediction problem. The number of common neighbors [14] of a node pair can be computed for each of the previous L snapshots. The sum of the number of common neighbors over the past L snapshots, are used to estimate this node pair's tendency of connection at the next time step.

Lasso Regression [26] assumes that the activity of link i at time $t+1$ is a linear function of the activities of all the links at time t, i.e.,

$$x_i(t+1) = \sum_{j=1}^{M} x_j(t)\beta_{ij} + c_i. \tag{3}$$

The objective is

$$\min_{\beta_i}\{\sum_{t=1}(x_i(t+1)-\sum_{j=1}^{M}x_j(t)\beta_{ij}-c_i)^2+\alpha\sum_{j=1}^{M}|\beta_{ij}|\}. \quad (4)$$

where M is the number of features as well as the number of links, c_i is the constant coefficient and $\beta_i = \{\beta_{i1}, \beta_{i2}, \ldots, \beta_{iM}\}$ are the regression coefficients of all the features for link i. The coefficients will be learned from the temporal network observed in the past L steps for each link. We use $L1$ regularization, which adds a penalty to the sum of the magnitude of coefficients $\sum_{j=1}^{M}|\beta_{ij}|$. The parameter α controls the penalty strength. The regularization forces some of the coefficients to be zero and thus lead to models with few non-zero coefficients (relevant features). The optimal α that achieves the best prediction is chosen from 50 logarithmically spaced points within $[10^{-4}, 10]$.

6 Model Evaluation

In this section, we firstly introduce the method to evaluate the models in link prediction quality. Secondly, we explore how to choose the decay factor in the SD model. Thirdly, we compare the link prediction quality of all the models.

6.1 Link Prediction Quality

Each model predicts the link activity at time step $t + 1$ based on the temporal network observed in the past L steps. The number of contacts at each time step shows periodic behaviour, i.e., large number of contact recurs at regular intervals. In order to capture such potential periodic patterns of a temporal network, we consider $L = T/2$, i.e., half of the length of a real-world temporal network's time window. The prediction step $t + 1$ is sampled 1000 times from $[T/2 + 1, T]$ with equal space. The average proportion of the M links that are active at a time step is lower than 1% in all the real-world networks we considered. The classification labels (the number of active links and inactive links per time step) are imbalanced. Hence, we evaluate the prediction quality via the area under the precision-recall curve (AUPR) [27]. AUPR provides an aggregate measure of performance across all possible classification thresholds. The average AUPR of a model over the 1000 prediction snapshots quantifies the prediction quality of the model.

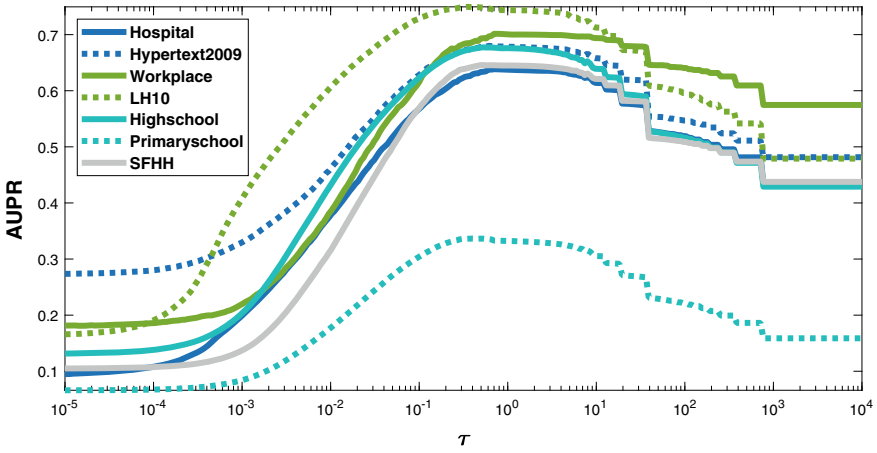


Fig. 3 Link prediction quality AUPR of the SD model as a function of the decay factor τ in seven data sets

6.2 Choice of Decay Factor

How to chose the decay factor τ will be motivated by comparing two possibilities. We first consider a simple case where τ is a control parameter and does not vary over time, i.e., remaining the same for the 1000 samples of the prediction steps. For a given τ, the tendency $w_i(t+1)$ ($i \in [1, 2, ..., M]$) is obtained at each prediction step $t+1$ based on Eq.(1). Figure 3 shows that the decay factor τ indeed affects the prediction quality AUPR of the SD model. A universal pattern is that the optimal performance is obtained by a common and relatively broad range of $\tau \in [0.5, 5]$ in all networks. This implies that our real-world physical contact networks measured at school, hospital, workplace etc. may be formed by a universal class of time decaying memory. Hence, τ can be chosen arbitrarily within [0.5, 5].

In the second method of choosing τ, a $\tau(t+1)$ for each prediction step $t+1$ is learned from the network observed in the past L steps. The $\tau(t+1)$ is choose as the one that allows the SD model to best predict the temporal network at t based on the network observed in the past $L-1$ steps just before t. The prediction quality from the first (second) method of choosing τ are 0.63 (0.61), 0.68 (0.67), 0.69 (0.63), 0.75 (0.74), 0.68 (0.67), 0.34 (0.33) and 0.65 (0.63), for the seven data sets, respectively.

Hence, τ could be chosen arbitrarily from [0.5, 5], which has lower computational complexity and better prediction quality than learning τ dynamically over time. We consider $\tau = 0.5$ to derive the SD tendency and SCD tendency in the rest analysis.

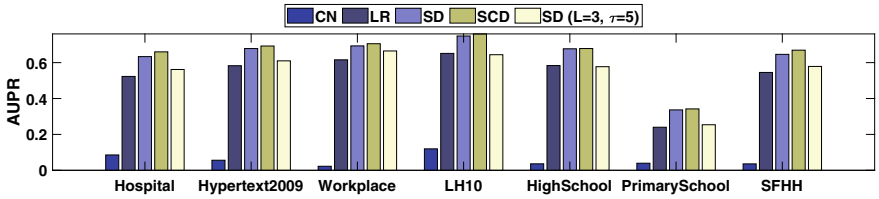


Fig. 4 Temporal network prediction quality AUPR of baseline CN method, baseline Lasso Regression (LR), SD model and SCD model. All methods consider $L = T/2$ and $\tau = 0.5$ except for SD ($\tau = 5$, $L = 3$), which is needed only for Sect. 7.2

6.3 Comparison of Models

We further compare the prediction quality of our SD model, SCD model and baseline models in Fig. 4. We find that both SD and SCD models perform better than the baselines. The SCD model, which predicts a link's connection utilizing SD tendency of the neighboring links and of the link itself, indeed performs better than the SD model that uses only the SD tendency of the link itself. Moreover, the SD and SCD models perform the best (worst) in LH10 (PrimarySchool), in line with the strongest(weakest) memory/similarity of LH10(PrimarySchool) observed in Fig. 1.

7 Model Interpretation

In this section, we interpret firstly the SCD model, to understand how the past states of different types of links in the ego-network (neighborhood) contribute to the prediction of the center link of the ego-network and interpret afterwards the common range $\tau \in [0.5, 5]$.

7.1 Interpretation of SCD Model

As defined in Eq. (2), SCD model predicts a link's future connection, based on the SD tendency of the link itself, links that form a triangle with the link and the rest links that share a common node with the link. The contribution of these three types of links are reflected in the learned coefficients in Eq. (2). The average coefficients over all prediction steps are given in Table 2. We can see a link's next step activity is mainly influenced by the past activities of the link itself, and slightly influenced by the neighboring links that form a triangle with the target link. The other neighboring links have very limited impact on target link's activity. The predictive power of neighboring links that form a triangle with the target link may come from the nature

Table 2 The learned coefficient β_1^*, β_2^* and β_3^* in SCD model averaged over 1000 prediction steps are also provided

Network	β_1^*	β_2^*	β_3^*	cc
Hospital	0.31	0.07	0.00	0.37
Hypertext2009	0.32	−0.02	0.00	0.32
Workplace	0.32	0.00	0.00	0.28
LH10	0.32	0.21	0.00	0.41
HighSchool	0.33	0.04	0.00	0.38
PrimarySchool	0.24	0.48	−0.02	0.54
SFHH	0.32	0.03	0.01	0.21

And the clustering coefficient (cc) of the aggregated network in each empirical network

of physical contact networks: contacts are often determined by physical proximity and two people that are close to a third but not yet close to each other are likely to already be in relatively close proximity.

One exception is the PrimarySchool, where $\beta_2^* > \beta_1^*$. The aggregated network of PrimarySchool has the largest clustering coefficient[1] in the aggregated network as shown in Table 2. In general, we find the contribution of links that form a triangle with the target link tends to be more significant in temporal networks whose aggregated network has more triangles.

7.2 Decay Factor

Finally, we interpret the common range of the decay factor $\tau \in [0.5, 5]$ where the SD performs optimally. According to the definition of SD tendency of connection in Eq.(1), only the coefficients/contributions $y = e^{-\tau(t-k)}$ of the previous 24 steps (3 steps) are larger than 10^{-5} when $\tau = 0.5$ ($\tau = 5$), out of $L = T/2 > 1000$ previous steps observed. We wonder whether considering only few previous steps instead of $L = T/2$ steps would be sufficient for a good prediction. The prediction quality of our SD model when $L = 3$ and $\tau = 5$ is given in Fig. 4. It is worse than the performance of SD model when $L = 50\%T$ and $\tau = 0.5$. This suggests that although the contribution of each early state of a link is small, the accumulated contribution of many early states improves the prediction quality. The prediction quality of SD model when $L = 3$ and $\tau = 5$, whose computational complexity is extremely low, is still better or similar to that of Linear Regression, reflecting the prediction power of recent states of a link.

[1] The clustering coefficient of network is the probability that two neighbors of node are connected.

8 Conclusion

In this work, we propose two network-based temporal network prediction models motivated by the observed time decaying similarities/memory in temporal networks. The proposed self-driven (SD) model and self- and cross-driven (SCD) model predict a link's future activity based on the past activities of the link itself, and also of the neighboring links, respectively. Both models perform better than the baseline models and the SCD outperforms SD model. Interestingly, we find that both models perform better in temporal networks with a stronger memory (similarity over time). The SCD model reveals that a link's future activity is mainly determined by (the past activities of) the link itself, moderately by neighboring links that form a triangle with the target link, and hardly by other neighboring links. However, if the temporal network has a high clustering coefficient in its aggregated, the contribution of the neighboring links that form a triangle with the target link tends to be significant and possibly dominant.

Our work is a starting point to explore network-based temporal network prediction methods, especially how methods could be designed based on network proprieties and how their performance could be explained again by network properties. It is interesting to evaluate network-based prediction methods more systematically in comparison with learning-based methods and explore the integration of both types of methods.

Acknowledgements The authors would like to thank the support of the Netherlands Organisation for Scientific Research NWO (TOP Grant no. 612.001.802) and also thank the support of the China Scholarship Council.

References

1. Holme, P., Saramäki, J.: Temporal networks. Phys. Rep. **519**, 97–125 (2012)
2. Masuda, N., Lambiotte, R.: A guide to temporal networks. In: Series on complexity science, vol. 4, pp. 252. World scientific, Europe (2016)
3. Masuda, N., Klemm, K., Eguíluz, V.M.: Temporal networks: slowing down diffusion by long lasting interactions. Phys. Rev. Lett. **111**, 188701 (2013)
4. Delvenne, J.-C., Lambiotte, R., Rocha, L.E.C.: Diffusion on networked systems is a question of time or structure. Nat. Commun. **6**, 7366 (2015)
5. Peixoto, T., Rosvall, M.: Modelling sequences and temporal networks with dynamic community structures. Nat. Commun. **8**, 582 (2017)
6. Zhan, X.-X., Hanjalic, A., Wang, H.: Information diffusion backbones in temporal networks. Sci. Rep. **9**, 6798 (2019)
7. Lü, L., Medo, M., Yeung, C.H., Zhang, Y.-C., Zhang, Z.-K., Zhou, T.: Recommender systems. Phys. Rep. **519**, 1–49 (2012)
8. Aleta, A., Tuninetti, M., Paolotti, D., Moreno, Y., Starnini, M.: Link prediction in multiplex networks via triadic closure. Phys. Rev. Res. **2**, 042029 (2020)
9. Zhou, L., Yang, Y., Ren, X., Wu, F., Zhuang, Y.: Dynamic network embedding by modeling triadic closure process. In: 32nd AAAI Conference on Artificial Intelligence, pp. 571-578. AAAI Press, California USA (2018)
10. Rahman, M. Saha, T.K., Hasan, M.A., Xu, K.S., Reddy, C.K.: DyLink2Vec: Effective Feature Representation for Link Prediction in Dynamic Networks (2018)

11. Zhan, X.-X., Li, Z., Masuda, N., Holme, P., Wang, H.: Susceptible-infected-spreading-based network embedding in static and temporal networks. EPJ Data Sci. **9**, 30 (2020)
12. Li, X., Du, N., Li, H., Li, K., Gao, J., Zhang, A.: A deep learning approach to link prediction in dynamic networks. In: 2014 SIAM International Conference on Data Mining, pp. 289-297 (2014)
13. Pareja, A., Domeniconi, G., Chen, J., Ma, T., Suzumura, T., Kanezashi, H., Kaler, T., Schardl, T.B., Leiserson, C.E.: EvolveGCN: evolving graph convolutional networks for dynamic graphs. In.: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 5363–5370. AAAI Press, California, USA (2020)
14. Liben-Nowell, D., Kleinberg, J.: The link prediction problem for social networks. In: The Twelfth International Conference on Information and Knowledge Management, vol. 4, pp. 556–559. Association for Computing Machinery, New York (2003)
15. Ahmed, N.M., Chen, L., Wang, Y., Li, B., Li, Y., Liu, W.: Sampling-based algorithm for link prediction in temporal networks. Inf. Sci. **374**, 1–14 (2016)
16. Xu, H.H., Zhang, L.J.: Application of link prediction in temporal networks. Adv. Mater. Res. **2231**, 756–759 (2013)
17. Li, X., Liang, W., Zhang, X., Liu, X., Wu, W.: A universal method based on structure subgraph feature for link prediction over dynamic networks. In: 39th International Conference on Distributed Computing Systems, pp. 1210–1220. IEEE, Dallas, USA (2019)
18. Jo, H.-H., Perotti, J.I., Kaski, K., Kerteéz, J.: Correlated bursts and the role of memory range. Phys. Rev. E. **92**, 022814 (2015)
19. Génois, M., Barrat, A.: Can co-location be used as a proxy for face-to-face contacts? EPJ Data Sci. **7**, 11 (2018)
20. Rossi, R.A., Ahmed, K.: The network data repository with interactive graph analytics and visualization. In: The Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 4292–4293. AAAI Press, Palo Alto, California (2015)
21. Isella, L., Stehlé, J., Barrat, A., Cattuto, C., Pinton, J.-F., den Broeck, W.V.: What's in a crowd? Analysis of face-to-face behavioral networks. J. Theor. Biol. **271**, 166–180 (2011)
22. Zou, L., Zhan, X.-X., Sun, J., Hanjalic, A., Wang, H.: Temporal network prediction and interpretation. IEEE Trans. Netw. Sci. Eng. **9**, 1215–1224 (2022)
23. Yu, W., Cheng, W., Aggarwal, C.C., Chen, H., Wang, W.: Link prediction with spatial and temporal consistency in dynamic networks. In: The Twenty-Sixth International Joint Conference on Artificial Intelligence, pp. 3343–3349. International Joint Conferences on Artificial Intelligence, Melbourne, Australia (2017)
24. Paranjape, A., Benson, A.R., Leskovec, J.: Motifs in temporal networks. In: Tenth ACM International Conference on Web Search and Data Mining, pp. 601–610. Association for Computing Machinery, Cambridge, United Kingdom (2017)
25. Saramäki, J., Moro, E.: From seconds to months: an overview of multi-scale dynamics of mobile telephone calls. Eur. Phys. J. B. **88**, 164 (2015)
26. Tibshirni, R.: Regression shrinkage and selection via the lasso. J. R. Stat. Soc. Series B. **58**, 267–88 (1996)
27. Davis, J., Goadrich, M.: The relationship between precision-recall and ROC curves. In: The 23rd International Conference on Machine Learning, vol. 8, pp. 233–240. Association for Computing Machinery, New York, USA (2006)

Drug Trafficking in Relation to Global Shipping Network

Louise Leibbrandt, Shilun Zhang, Marijn Roelvink, Stan Bergkamp, Xinqi Li, Lieselot Bisschop, Karin van Wingerde, and Huijuan Wang

Abstract This paper aims to understand to what extent the amount of drug (e.g., cocaine) trafficking per country can be explained and predicted using the global shipping network. We propose three distinct network approaches, based on topological centrality metrics, Susceptible-Infected-Susceptible spreading process and a flow optimization model of drug trafficking on the shipping network, respectively. These approaches derive centrality metrics, infection probability, and inflow of drug traffic per country respectively, to estimate the amount of drug trafficking. We use the amount of drug seizure as an approximation of the amount of drug trafficking per country to evaluate our methods. Specifically, we investigate to what extent different methods could predict the ranking of countries in drug seizure (amount). Furthermore, these three approaches are integrated by a linear regression method in which we combine the nodal properties derived by each method to build a comprehensive model for the cocaine seizure data. Our analysis finds that the unweighted eigenvector centrality metric combined with the inflow derived by the flow optimization method best identifies the countries with a large amount of drug seizure (e.g., rank correlation 0.45 with the drug seizure). Extending this regression model with two extra features, the distance of a country from the source of cocaine production and a country's income group, increases further the prediction quality (e.g., rank correlation 0.79). This final model provides insights into network derived properties and complementary country features that are explanatory for the amount of cocaine seized. The model can also be used to identify countries that have no drug seizure data but are possibly susceptible to cocaine trafficking.

Keywords Drug trafficking · Drug seizure · Shipping network · Network method

L. Leibbrandt · S. Zhang · M. Roelvink · X. Li · H. Wang (✉)
Faculty of Electrical Engineering, Mathematics and Computer Science,
Delft University of Technology, Delft, The Netherlands
e-mail: H.Wang@tudelft.nl

S. Bergkamp
Faculty of Applied Sciences, Delft University of Technology, Delft, The Netherlands

L. Bisschop · K. van Wingerde
Erasmus School of Law, Erasmus University Rotterdam, Rotterdam, The Netherlands

H. Cherifi et al. (eds.), *Complex Networks and Their Applications XI*,
Studies in Computational Intelligence 1078,
https://doi.org/10.1007/978-3-031-21131-7_52

1 Introduction

Complex networks have been widely used to represent real-world complex systems, where nodes denote the components and links represent relations or interaction between components. Significant contributions have been made to characterize complex networks and to understand the effect of a network on a dynamic process unfolding on the network. Diverse nodal centrality metrics [1, 2] have been proposed to measure various topological properties of a node. Nodal centrality metrics have been applied in general to estimate the importance of nodes in their function, e.g., to identify nodes with high spreading capacity and to select nodes to be immunized when a virus is prevalent [3–5]. Spreading models such as Susceptible-Infected (SI) model and Susceptible-Infected-Susceptible (SIS) model have been intensively studied [6, 7] to model the spread of epidemic and information on networks. Deep understanding has been achieved regarding how the underlying network topology affects a spreading process, how to predict and control a spreading process on a network [8].

Shipping networks play a crucial role in world trade as around 80% of global trade by volume is carried by sea.[1] From the aspect of network topology, prior literature explored the overall structure of the global shipping network, revealing its scale-free property [9] and modular structure [10]. Li et al. investigated the relationship between the centrality of nodes in the global shipping network and the economy of corresponding areas [11]. Global shipping network has been found to have "economic small-world" characteristic [12], i.e., with high transportation efficiency and low wiring cost. Some other efforts have been devoted to model dynamic processes on shipping networks, e.g., marine species invasion process [13, 14]. Nonetheless, how illicit trafficking, like drug trafficking, is linked to the shipping network from the angle of network science remains unexplored.

In this paper, we investigate how to explain and predict drug (e.g., cocaine) trafficking using the global shipping network. The amount of drug seizure in each country is used as an approximation of the amount of drug trafficking to evaluate our methods. We propose three types of network-based methods. These methods are evaluated via their capability to predict the ranking countries in drug seizure (amount) thus to identify countries with a large drug seizure. The first method uses traditional nodal centrality metrics of a country in the shipping network to estimate the volume of drugs seized in the country [15]. Secondly, we employ the SIS spreading model on the shipping network. The infection probability of a node (country) in the metastable state is derived to indicate a country's drug seizure. In the third method, we formulate drug trafficking as the optimal flow on the shipping network, where the number of links to route the traffic from countries that produce drugs and countries that consume drugs is minimized. The inflow to a country is used to estimate the drug seizure of that country. We finally combine the above three methods using linear regression, showing a better prediction of the ranking of countries in drug seizure and identifying key factors that explain the amount of drug seizure per country. This

[1] https://unctad.org/webflyer/review-maritime-transport-2018.

linear regression model has also been extended by including two extra country-level properties, namely the distance of a country from the source of cocaine production and a country's income group/level. We find that these extra features could further improve the prediction quality and the essential role of network-based properties.

The paper is structured as follows. Section 2 introduces the construction of the shipping network and drug seizure data. Section 3 describes and evaluates our methods. Section 4 summarizes our key findings.

2 Datasets

Shipping Network Construction. The Global Liner Shipping Network has been derived from service routes data of the world's top 100 liner shipping companies in 2015, by mapping each service route as a complete graph where any two ports in the service route were connected via a link [16, 17]. It is composed of 977 unique ports and 16,680 inter-port connections. We construct the country-level shipping network as follows. Based on the country code of each port extracted from the Marine Traffic ports database [18], each port can be mapped to the country it belongs to. In the unweighted shipping network, nodes are the countries, and two nodes i and j are connected by a link, i.e., $a_{ij} = 1$ in the adjacency matrix A if at least two ports from the two countries respectively have an inter-port connection, otherwise $a_{ij} = 0$. A weighted network can be further constructed by having the same topology as the unweighted network and associating each link with a weight w_{ij} that equals the total number of inter-port connections between countries i and j. The weight $w_{ij} = 0$ if there is no inter-port connection between i and j. Both networks have $N = 174$ countries and $L = 2743$ links in 2015 and are relatively stable over time. The unweighted network is visualized in Fig. 1.

Drug Seizure Data. The drug seizure data is from the United Nations Office on Drugs and Crime annual drug seizures report [15]. The data contains reports for 144 unique countries between the years 2012–2016. We extract all entries that pertain to the drug group of *Cocaine-type* and can reliably be converted to kilogram equivalents. The average amount of drug seizures per year per country over 2012–2016 is considered. There is an overlap of $N = 110$ countries between the shipping network and the drug seizure data.

Hence, we consider the sub-shipping weighted and unweighted networks, that contain these 110 common countries as nodes and their connections in this paper. Both shipping networks contain $N = 110$ countries and $L = 1794$ links and the average (standard deviation) of link weights is 6.1 (13.4) in the weighted network.

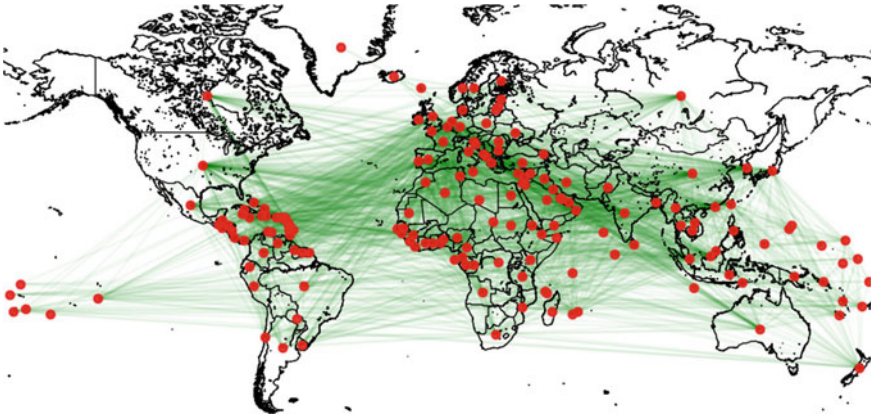


Fig. 1 Unweighted global liner shipping network. Each node represents a country (or district)

3 Methods

Firstly, we propose three distinct network based approaches to model the drug seizure. Each approach uses the shipping network as main ingredient and creates an output nodal property that is meant to be representative of the drug seizure of a country.

We evaluate the quality of all approaches in predicting the ranking of countries in drug trafficking/seizure amount via two measures: Spearman's rank correlation and the recognition rate between an output nodal property and drug seizure of a country. The top f recognition rate $r_{\phi v}(f)$ is defined as

$$r_{\phi v}(f) = \frac{|R_f^{\phi} \cap R_f^{v}|}{|R_f^{\phi}|},$$

where R_f^{ϕ} is the set of f percentage of countries that have the highest drug seizure amount and R_f^{v} is the set of f percentage of countries that have the highest (lowest[2]) value in property v. The recognition rate $r_{\phi v}(f)$ equals the size of the overlap between R_f^{ϕ} and R_f^{v}, or the number of common nodes, normalized the size of each set $|R_f^{\phi}| = |R_f^{v}| = Nf$. We consider $f = 10$, 20 and 40% as examples. Using these two measures, we aim to understand the capability of our approaches in identifying the countries with a high drug seizure.

[2] When drug seizure and the nodal property are supposed to be negatively correlated.

3.1 Topological Metrics

Our first approach uses each nodal centrality metric derived from the shipping network to estimate the drug seizure per country. We are interested in which metric has the highest correlation with drug seizure and is therefore most representative of drug seizure. We will briefly introduce our four chosen centrality metrics, our reasoning and hypotheses for their correlation with drug seizure.

- *Degree* d_i of a node i is the number of links incident to the node i. The degree of a country represents the number of countries directly connected to it in the shipping network.
- *Betweenness Centrality* b_i of a node i in the unweighted shipping network is the number of shortest paths that traverse the node i between all possibly node pairs [19, 20].
- *Clustering Coefficient* c_i of a node i equals the number of links among the neighbors of the node normalized by $\binom{d_i}{2}$. It tells the link density among the d_i neighbors of node i.
- *Eigenvector Centrality*. The unweighted (weighted) eigenvector centrality of a node is the principal eigenvector component and the principal eigenvector is the eigenvector corresponds to the largest eigenvalue of the unweighted (weighted) adjacency matrix A (W). A country with a high eigenvector centrality tends to connect to many countries who themselves have a high connection (eigenvector centrality).

We suspect that countries with a high degree, betweenness or Eigenvector centrality are susceptible to large amount of drug trafficking due to their good network connection, infrastructure and the corresponding flexibility to change modus operandi (e.g., shifting drugs to another port). In contrary, countries with a large clustering coefficient could be less susceptible since they do not have a large degree and drug trafficking can be through their mutually connected neighbors without the need of traversing the country itself.

Results. The Spearman's rank correlation and recognition rate between the drug seizure and each chosen centrality metric is given in Table 1. We find that all metrics have a statistically significant ($p < 0.01$) rank correlation with drug seizure, and the sign of each correlation is in line with our hypothesis for the corresponding metric. All centrality metrics can contribute to the identification of countries with large seizure since their recognition rate $r_{\phi v}(f) > f$, thus is better than that of a random guess. We find that the unweighted eigenvector centrality metric provides the highest correlation in strength, however the weighted eigenvector centrality and betweenness leads to the highest recognition rate when $f = 20\%$ and $f = 40\%$ respectively.

Table 1 Evaluation of prediction performance of topological metrics, SIS infection probability, and inflow (derived from the flow optimization model), via rank correlation and recognition rate

Metric	Corr.	p-value	$r_{\phi v}(10\%)$	$r_{\phi v}(20\%)$	$r_{\phi v}(40\%)$
Degree	0.39,	< 0.01	0.27	0.41	0.57
Betweenness	0.34,	< 0.01	0.27	0.36	**0.61**
Clustering	−0.35,	< 0.01	0.27	0.36	0.60
Eigenvector (unweighted)	**0.40**	< 0.01	0.27	0.41	0.59
Eigenvector (weighted)	0.39	< 0.01	0.27	**0.50**	0.52
SIS infection prob. (unweighted net) $\tau = 0.045$	**0.41**	< 0.01	0.27	0.41	0.59
SIS infection prob. (weighted net) $\tau = 0.021$	0.40	< 0.01	0.27	0.41	0.59
Inflow	0.33	< 0.01	0.36	0.36	0.57

The best performance of each category is in bold

3.2 SIS Spreading Process

We have shown recently that the SIS epidemic spreading model can be generalized to model the contagion of traffic congestion at an airport on an airline network and the infection probability of an airport can be used to estimate the probability of congestion at the airport [21]. Inspired by this, we propose our second approach: model the contagion of drug trafficking as an SIS spreading process on the shipping network and uses the infection probability of a country in the meta-stable state to estimate the ranking of countries in drug seizure. We will briefly introduce the SIS model, method to derive infection probabilities and evaluate this approach.

SIS Model. The homogeneous SIS model is defined as follows. At any time t, a node is either susceptible or infected. A susceptible node can be infected by each of its infected neighbors with an infection rate β and each infected node recovers to be susceptible again with a recovery rate δ. Both the infection and recovery processes are independent Poisson processes. For a given network upon which the SIS process is deployed, a critical epidemic threshold τ_c exists. When the effective infection rate $\tau = (\beta/\delta) > \tau_c$, a non-zero fraction of infected nodes persists in the meta-stable state. When $\tau < \tau_c$, the epidemic dies out.

Mean-Field Approximation of SIS Model. We derive nodal infection probabilities in the meta-stable state via N-Intertwined Mean-Field Approximation (NIMFA) [22]. NIMFA is chosen as it preserves the network topology in its governing equations, coupling the infection probability of neighboring nodes. It assumes that the states of neighboring nodes are uncorrelated. Under NIMFA, the governing equation for a node i in our heterogeneous SIS spreading model is

$$\frac{\mathrm{d}v_i(t)}{\mathrm{d}t} = -\delta v_i(t) + (1 - v_i(t)) \sum_{j=1}^{N} \beta w_{ij} v_j(t), \tag{1}$$

where $v_i(t)$ is the infection probability of node i at time t, and βw_{ij} is the infection rate associated to the link (i, j) with weight w_{ij}. The time derivative of the infection probability $v_i(t)$, is determined by two competing processes (a) while node i is infected, node i is cured at rate δ and (b) while node i is susceptible, each infected neighbour j infects node i with a rate βw_{ij}. In the meta-stable state, $\frac{\mathrm{d}v_i(t)}{\mathrm{d}t} = 0$, for any $i \in \mathcal{N}$. Hence, the infection probability of each node in this case, $v_{i\infty}$, can be derived by solving the equations $\delta v_{i\infty} + (1 - v_{i\infty}) \sum_{j=1}^{N} \beta w_{ij} v_{j\infty} = 0$. The trivial all-zero solution corresponds to the absorbing state where all nodes are susceptible.

Both the unweighted and weighted shipping networks will be considered. When the underlying network is the unweighted, the governing Equation (1) should be updated by replacing the weighted adjacency matrix element w_{ij} by the unweighted adjacency matrix element a_{ij}.

Results. In Fig. 2, the Spearman's rank correlation and recognition rate between the infection probability and drug seizure amount are plotted as a function of the effective infection rate τ, when the underlying network is the unweighted shipping network. The Spearman's rank correlation varies only slightly when the effective infection rate is small around the critical epidemic threshold $\tau_c = 0.020$. This is in line with the theoretical and empirical finding in [23] that the ranking of nodal infection probability tends to change more with τ, when τ is small. The top f recognition rate is insensitive of the effective spreading rate τ. The same trends have been observed when the underlying network is the weighed shipping network. The maximal rank correlation and recognition rate that can be reached by choosing τ (just above the threshold) are summarized in Table 1. We also find the infection probability of a country derived from the unweighted (weighted) shipping network at the optimal effective infection rate, is strongly correlated with the unweighted (weighted) principal eigenvector component derived in Sect. 3.1, with a correlation coefficient around 0.99. This is in line with the theoretical proof that when the effective infection rate is just above the epidemic threshold, the meta-stable state infection probability of a node, obtained by NIMFA is proportional to the principal eigenvector component of the adjacency matrix A (W) [24]. By tuning the effective infection rate β to optimize the evaluation metrics, we improve the performance marginally compared to the principal eigenvector component.

3.3 Flow Optimization Model

For this method we view transnational drug trafficking as an economic process where drugs go from production countries to consumption countries through a chain of intermediaries [25]. This is motivated by two main factors that contribute to Cocaine trafficking. The first is that the price of Cocaine is largely attributable to

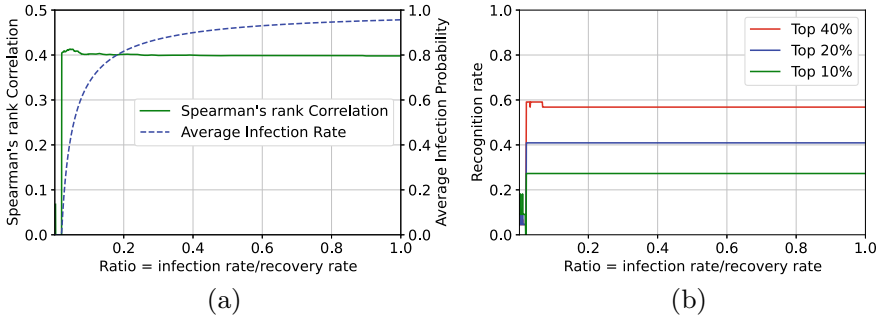


Fig. 2 **a** Spearman's rank correlation (and the average infection probability over all nodes) and **b** recognition rates between nodal infection probability and drug seizure as a function of the effective infection rate. The underlying network is the unweighted shipping network

risk compensation[25]; in an economic system, distributors may try to minimize the number of links used to transport drugs from supply countries to consumption countries. The second factor is that a large portion of Cocaine trafficking uses the shipment network as way of transportation [26]. We therefore create an optimization problem in which we minimize the number of links used to transport drugs in our shipping network whilst adhering to constraints that model the supply and demand for each country. The total inflow at each country will be used to predict the ranking of countries in drug seizure.

The following steps will be taken to derive the (normalized) supply and demand per country. Firstly, we extract the amount of cocaine production s_i of each country i in 2012 from [27] and the drug usage u_i(as % of population) of each country i from [28]. We define the normalized supply of a country i as $s_i^* = \frac{s_i}{\sum_{i=1}^{N^s} s_i}$, where N^s is the number of supply countries. Similarly, the normalized demand of a country i is defined as $u_i^* = \frac{u_i * m_i}{\sum_{i=1}^{N^u} u_i * m_i}$, where m_i is the population of country i and N^u is the number of consumption countries. The total normalized supply (or demand) of all countries is one.

Countries of supply or consumption may not have any port, thus not exist in our shipping networks. Hence, we extend our unweighted shipping network with 174 nodes by including all the supply and consumption countries listed in the data [27] and [28] and adding directed link(s) from (to) a supply (consumption) country that has no port to (from) countries that have a port and share a border with it, using country border dataset [29]. In total, 49 nodes and 130 directed links have been added.

We assume that the flow from the multiple supply countries to the consumption countries follows the paths on the shipping network with the minimal number of links, i.e. $\sum_{i,j} \mathbf{1}_{\{A_{ij} \cdot F_{ij} > 0\}}$, where F_{ij} is the flow from country i to j and the indicator function $\mathbf{1}_{\boldsymbol{x}}$ is one (zero) when the condition $\boldsymbol{x}$ is true (false). For each country, the following constraint must hold:

$$s_i^* + \sum_{j=1}^{N^*} A_{ji} F_{ji} = \sum_{j=1}^{N^*} A_{ij} F_{ij} + u_i^* \tag{2}$$

The constraint ensures the total incoming and outgoing flow combined with the supply and demand in each node is balanced. After the optimization, 68 nodes have a positive inflow or outflow.

Results. We use the final inflow, $F_i = \sum_{j=1}^{N^*} A_{ji} F_{ji}$, per country to estimate ranking of the same set of 110 countries in drug seizure as in previous analysis. The rank correlation and recognition rate between the amount of in-flow and drug seizure of a country can be found in Table 1. We find that whilst this method produces a lower rank-correlation than the methods described in Sects. 3.1 and 3.2 likely due to the zero inflow of many countries, it achieves a significantly higher recognition rate at $f = 10\%$. This implies the complementary nature of the three methods and the potential synergy when combining them.

3.4 Regression Model

We have shown that all three methods (their output nodal properties) contribute to the estimation of countries with the highest amount of drug seizure. In order to explore their synergy and identify the key nodal properties in explaining drug seizure, we build a regression model that uses aforementioned network derived features in combination with other country features. We fit multiple Gaussian linear regression models on the logarithm of the drug seizure data, taking different features as independent variables. We split our analysis into three parts; in Analysis 1 we investigate our network derived features, in Analysis 2 we investigate extra country features, and in Analysis 3 we combine the results from analyses 1 and 2 to propose a final model.

Table 2 provides an overview of the results for each analysis. The adjusted R^2 value indicates the predictive power of each regression model. We also provide Spearman's rank correlation and recognition rate between the drug seizure amounts predicted by each regression model and the actual drug seizure amounts. The correlation and recognition rate in Table 2 for the single feature regression in $A.1$ are in line with the results produced by each method in Sects. 3.1, 3.2 and 3.3 and serve as a baseline for the subsequent models.

Analysis 1. We start with regression models that use each output property derived in Sects. 3.1, 3.2 and 3.3 as the single feature. We find that the unweighted Eigenvector Centrality (EC), derived in Sect. 3.1, results in the highest scoring R^2 value and therefore provides the best fit over all network based methods. Betweenness and clustering coefficient perform worse than EC and are not considered in this analysis. The Infection Probability (IP), Eigenvector Centrality (EC) on weighted and unweighted networks are strongly correlated and the unweighted EC performs the best among these four properties. Hence, we consider the combination of the unweighted EC and

Table 2 Results for regression analyses. Country level features considered include: principal eigenvector component of the unweighted shipping network (EC unweighted), of the weighted network (EC weighted), infection probability (IP) in unweighted and unweighted network respectively, inflow (IF), distance to source countries (D) and the Income Group (IG)

	Formula for regression	Adj. R^2	Corr.	$r_{\phi v}(10\%)$	$r_{\phi v}(20\%)$	$r_{\phi v}(40\%)$
A.1	(3.1) EC unweighted	0.13	0.40	0.27	0.41	0.59
	(3.1) EC weighted	0.10	0.39	0.27	0.50	0.52
	(3.2) IP unweighted	0.11	0.41	0.27	0.41	0.59
	(3.2) IP weighted	0.11	0.40	0.27	0.55	0.55
	(3.3) IF	0.07	0.33	0.36	0.36	0.57
	EC_u + IF	**0.16**	**0.45**	**0.36**	**0.55**	**0.64**
A.2	D	0.27	0.55	0.64	0.59	0.64
	IG	0.20	0.41	0.27	0.41	0.43
	D + IG	**0.41**	**0.68**	**0.64**	**0.50**	**0.77**
A.3	**EC unweighted + IF + D + IG**	**0.55**	**0.79**	**0.73**	**0.82**	**0.82**

The model that performs the best in each of the three categories are in bold

the inflow (IF). Whilst the total country Inflow (IF), derived in Sect. 3.3, does not provide relatively high rank correlation, the combined regression of unweighted EC and IF performs better than that of a single network based property, revealing that these two features contain cumulative predictive power.

Analysis 2. Besides network based features, we investigate another two key country level features: the Distance (D) of a country from the countries where cocaine is produced and the Income Group (IG) of a country, because they are known to influence cocaine seizure rates [26]. For the Distance (D), we take the geodesic distance in kilometers between the coordinate of each country and the central coordinate of the source countries. The betweenness and D are weakly correlated (rank correlation 0.14). We also extract the Income Group (IG) of each country [30] in 2016; this assigns each country as either a *low*, *lower middle*, *upper middle* or *high* income country.

We find that both distance and Income Group have a high predictive power for drug seizure, and distance is an especially effective method to obtain high recognition rates, i.e. $r_{\phi v}(10\%) = 0.64$. The combined predictive power of the two country features is strong and this model is able to outperform the best model from Analysis 1 in all but one, $r_{\phi v}(20\%)$, of the evaluation metrics.

Analysis 3. For our final investigation, we combine the best performing models from Analysis 1 and 2. A correlation analysis reveals low rank correlation amongst EC, IF, D and IG (< 0.150). Therefore, we conclude that each feature provides unique information and should be used in the final model. This model outperforms all the other models. It is able to correctly predict 73% of the countries that are in the top 10% of the highest scoring countries based on drug seizure amounts, and 82% of the countries in both the top 20% and 40% lists. This observation highlights the essential role of both network based properties and the two country level properties in estimating drug seizure.

4 Conclusion

In this work, we propose different methods to explain and predict the amount of cocaine trafficking/seized per country using the global shipping network. We firstly propose three distinct network-based approaches: a centrality metrics based, an SIS spreading process based and a flow optimization model based approach. Correspondingly, the derived centrality metrics, infection probabilities, and inflow of drug traffic of nodes are used to estimate the ranking of countries in drug seizure. Furthermore, a linear regression analysis is designed to investigate the cumulative power of each approach with other country features.

We find that each approach with its output nodal property could contribute to the estimation of the ranking of countries in drug seizure. The eigenvector centrality of the unweighted shipping network seems to perform the best, also in view of the amount of data needed. Furthermore, the regression analysis reveals that inflow derived from the optimization model contains unique information when compared with the eigenvector centrality; combining in-flow and unweighted eigenvector centrality results in an evidently better estimation of drug seizure. We end our investigation by showing the benefit of combining our network-based results with other features, e.g., a country's distance to the source of cocaine and its income group.

The identification of countries that are susceptible to a large amount of cocaine trafficking is crucial in stopping the illicit substance from reaching its destination. Our final proposed model provides a starting point to tackling this problem. For countries that have no drug seizure data, the model can be used to predict their amount of drug trafficking. Furthermore, our methods can be applied or extended for drug trafficking of other drug groups such as amphetamine-type stimulants and opioids on a multi-layer network that includes diverse transportation modes. Our method could be further developed to distinguish between drug producing and non-producing countries, between domestic and trade/port related seizures and to understand whether the difference in the predicted and actual ranking of countries in drug seizure could be explained by the variance in law enforcement efforts of countries and other social and political factors.

References

1. Li, C., Li, Q., Van Mieghem, P., Eugene Stanley, H., Wang, H.: Correlation between centrality metrics and their application to the opinion model. Euro. Phys. J. B **88**(3), 1–13 (2015)
2. Lü, L., Chen, D., Ren, X.-L., Zhang, Q.-M., Zhang, Y.-C., Zhou, T.: Vital nodes identification in complex networks. Phys. Rep. **650**, 1–63 (2016)
3. Pastor-Satorras, R., Vespignani, A.: Immunization of complex networks. Phys. Rev. E **65**(3), 036104 (2002)
4. Kitsak, M., Gallos, L.K., Havlin, S., Liljeros, F., Muchnik, L., Eugene Stanley, H., Makse, H.A.: Identification of influential spreaders in complex networks. Nat. Phys. **6**(11), 888–893 (2010)
5. Borge-Holthoefer, J., Moreno, Y.: Absence of influential spreaders in rumor dynamics. Phys. Rev. E **85**(2), 026116 (2012)

6. Pastor-Satorras, R., Castellano, C., Van Mieghem, P., Vespignani, A.: Epidemic processes in complex networks. Rev. Mod. Phys. **87**(3), 925 (2015)
7. Kiss, I.Z., Miller, J.C., Simon, P.L., et al.: Mathematics of Epidemics on Networks, p. 598. Springer, Cham (2017)
8. Zhang, S., Zhao, X., Wang, H.: Mitigate sir epidemic spreading via contact blocking in temporal networks. Appl. Netw. Sci. **7**(1), 1–22 (2022)
9. Kaluza, P., Kölzsch, A., Gastner, M.T., Blasius, B.: The complex network of global cargo ship movements. J. R. Soc. Interface **7**(48), 1093–1103 (2010)
10. Pan, J.-J., Bell, M.G.H., Cheung, K.-F., Perera, S., Yu, H.: Connectivity analysis of the global shipping network by eigenvalue decomposition. Maritime Policy Manage. **46**(8), 957–966 (2019)
11. Li, Z., Mengqiao, X., Shi, Y.: Centrality in global shipping network basing on worldwide shipping areas. GeoJournal **80**(1), 47–60 (2015)
12. Xu, M., Pan, Q., Muscoloni, A., Xia, H., Cannistraci, C.V.: Modular gateway-ness connectivity and structural core organization in maritime network science. Nat. Commun. **11**(1), 1–15 (2020)
13. Xu, J., Wickramarathne, T.L., Chawla, N.V., Grey, E.K., Steinhaeuser, K., Keller, R.P., Drake, J.M., Lodge, D.M.: Improving management of aquatic invasions by integrating shipping network, ecological, and environmental data: data mining for social good. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1699–1708 (2014)
14. Saebi, M., Xu, J., Grey, E.K., Lodge, D.M., Corbett, J.J., Chawla, N.: Higher-order patterns of aquatic species spread through the global shipping network. Plos One **15**(7), e0220353 (2020)
15. UNODC: Annual Drug Seizure Report. https://dataunodc.un.org/drugs/seizures, 2012–2016. Online. Accessed 20 Mar 2022
16. Kojaku, S., Mengqiao, X., Xia, H., Masuda, N.: Multiscale core-periphery structure in a global liner shipping network. Sci. Rep. **9**(1), 1–15 (2019)
17. Xu, M.: Resource. http://www.mengqiaoxu.com/resource.html, 2019. Online. Accessed 28 Mar 2022
18. Marine Traffic. Ports Database. https://www.marinetraffic.com/en/data/, 2022. Online. Accessed 28 Mar 2022
19. Anthonisse, J.M.: The rush in a directed graph. In: Stichting Mathematisch Centrum. Mathematische Besliskunde, (BN 9/71) (1971)
20. Wang, H., Hernandez, J.M., Van Mieghem, P.: Betweenness centrality in a weighted network. Phys. Rev. E **77**(4), 046105 (2008)
21. Ceria, A., Köstler, K., Gobardhan, R., Wang, H.: Modeling airport congestion contagion by heterogeneous sis epidemic spreading on airline networks. Plos One **16**(1), e0245043 (2021)
22. Van Mieghem, P., Omic, J., Kooij, R.: Virus spread in networks. IEEE/ACM Trans. Netw. **17**(1), 1–14 (2008)
23. Bo, Q., Li, C., Van Mieghem, P., Wang, H.: Ranking of nodal infection probability in susceptible-infected-susceptible epidemic. Sci. Rep. **7**, 9233 (2017)
24. Van Mieghem, P.: Performance Analysis of Complex Networks and Systems. Cambridge University Press (2014)
25. Boivin, R.: Drug trafficking networks in the world-economy. Crime Netw. (2013)
26. UNODC and EUROPOL. The illicit trade of cocaine from latin america to europe—from oligopolies to freefor-all. Cocaine Insights 1, Sept 2021
27. UNODC. Statistical Annex: 6.1.3 Cocaine manufacture. https://www.unodc.org/unodc/en/data-and-analysis/wdr2021_annex.html, 2019. Online. Accessed 20 Mar 2022
28. UNODC. Statistical Annex: 1.2 Prevalence of drug use in the general population, including NPS—national data. https://www.unodc.org/unodc/en/data-and-analysis/wdr2021_annex.html, 2019. Online. Accessed 20 Mar 2022
29. Geo Data Source. Country Borders. https://www.geodatasource.com/addon/country-borders, 2022. Online. Accessed 20 Mar 2022
30. The World Bank. World Development Indicators. https://data.worldbank.org/indicator, 2020. Online. Accessed 03 May 2022