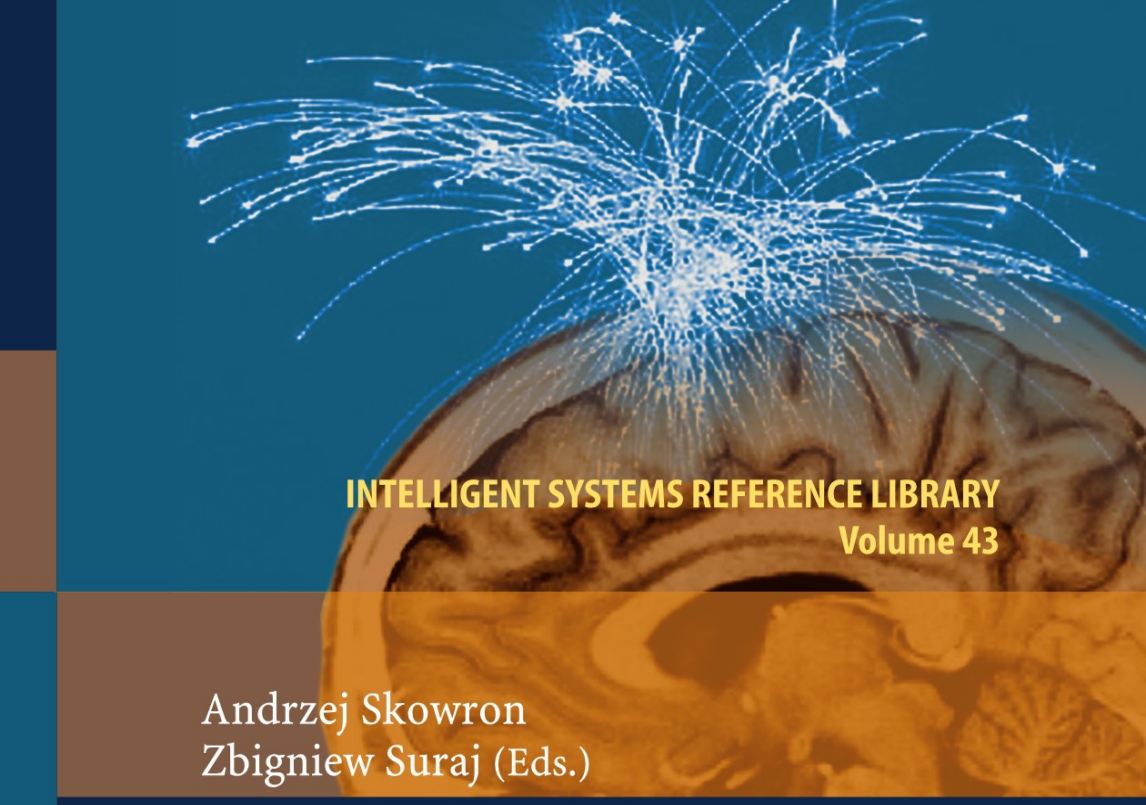


INTELLIGENT SYSTEMS REFERENCE LIBRARY
Volume 43

Andrzej Skowron
Zbigniew Suraj (Eds.)

Rough Sets and Intelligent Systems – Professor Zdzisław Pawlak in Memoriam

Volume 2

Springer

Intelligent Systems Reference Library 43

Editors-in-Chief

Prof. Janusz Kacprzyk
Systems Research Institute
Polish Academy of Sciences
ul. Newelska 6
01-447 Warsaw
Poland
E-mail: kacprzyk@ibspan.waw.pl

Prof. Lakhmi C. Jain
School of Electrical and Information Engineering
University of South Australia
Adelaide
South Australia SA 5095
Australia
E-mail: Lakhmi.jain@unisa.edu.au

For further volumes:
http://www.springer.com/series/8578

Andrzej Skowron and Zbigniew Suraj (Eds.)

Rough Sets and Intelligent Systems – Professor Zdzisław Pawlak in Memoriam

Volume 2

Springer

Editors
Prof. Andrzej Skowron
Institute of Mathematics
University of Warsaw
Warsaw
Poland

Prof. Zbigniew Suraj
Institute of Computer Science
University of Rzeszów
Rzeszów
Poland

ISSN 1868-4394
ISBN 978-3-642-30340-1
DOI 10.1007/978-3-642-30341-8
Springer Heidelberg New York Dordrecht London

e-ISSN 1868-4408
e-ISBN 978-3-642-30341-8

Library of Congress Control Number: 2012939519

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Affiliations of Authors (Vol. 1 & Vol. 2)

Abdulaziz Alkhalid
Mathematical and Computer Sciences and Engineering Division
King Abdullah University of Science and Technology
Thuwal, 23955-6900
Saudi Arabia
Email: abdulaziz.alkhalid@kaust.edu.sa

Talha Amin
Mathematical and Computer Sciences and Engineering Division
King Abdullah University of Science and Technology
Thuwal, 23955-6900
Saudi Arabia
Email: talha.amin@kaust.edu.sa

Piotr Artiemjew
Department of Mathematics and Computer Science
University of Warmia and Mazury
54, Słoneczna, Olsztyn, 10-710
Poland
Email: artem@matman.uwm.edu.pl

S. Asharaf
Indian Institute of Management Kozhikode
Kozhikode, Kerala
India 673570
Email: asharaf@iimk.ac.in

Nouman Azam
Department of Computer Science
University of Regina
Regina, Saskatchewan
Canada S4S 0A2
Email: azam200n@cs.uregina.ca

Sanghamitra Bandyopadhyay
Machine Intelligence Unit
Indian Statistical Institute
203, B.T. Road, Kolkata 700108
India
Email: sanghami@isical.ac.in

Mohua Banerjee
Indian Institute of Technology
Kanpur, 208016
India
Email: mohua@iitk.ac.in

Minakshi Banerjee
Center for Soft Computing Research
Indian Statistical Institute
203, B.T. Road, Kolkata 700108
India
Email: mbanerjee23@gmail.com

Jan G. Bazan
Institute of Computer Science
University of Rzeszów
2, Dekerta, Rzeszów, 35-030
Poland
Email: bazan@univ.rzeszow.pl
and
Institute of Mathematics
University of Warsaw
2, Banacha., Warsaw, 02-097
Poland

Stanisława Bazan-Socha
Second Department of Internal Medicine
Jagiellonian University Medical College
8, Skawińska, Cracow, 31-066
Poland
Email: mmsocha@cyf-kr.edu.pl

Theresa Beaubouef
Department of Computer Science and Industrial Technology
SLU 10847
Southeastern Louisiana University
Hammond, LA 70402
USA
Email: tbeaubouef@selu.edu

Parag Bhalchandra
School of Computational Sciences
Swami Ramanand Teerth Marathwada University
Nanded, Maharashtra
India 431606
Email: srtmun.parag@gmail.com

Jerzy Błaszczyński
Institute of Computing Science
Poznań University of Technology
2, Piotrowo, Poznań, 60-965
Poland
Email: jblaszczynski@cs.put.poznan.pl

Zbigniew Bonikowski
Institute of Mathematics and Informatics
Opole University
Poland
Email: zbonik@math.uni.opole.pl

Edward Bryniarski
Institute of Mathematics and Informatics
Opole University
Poland
Email: edlog@math.uni.opole.pl

Sylwia Buregwa-Czuma
Institute of Computer Science
University of Rzeszów
2, Dekerta, Rzeszów, 35-030
Poland
Email: sczuma@univ.rzeszow.pl

Cory Butz
Department of Computer Science
University of Regina
3737 Wascana Parkway
Regina, Saskatchewan
Canada S4S 0A2
Email: cory.butz@uregina.ca

Mihir Kr. Chakraborty
Center for Cognitive Science
Jadavpur University
Kolkata 700032
India
and
Indian Statistical Institute
203, B.T. Road, Kolkata 700108
India
Email: mihirc4@gmail.com

Nick Cercone
Department of Computer Science and Engineering
York University Toronto
ON M3J 1P3, Canada
Email: ncercone@yorku.ca

Chien-Chung Chan
Department of Computer Science, CAS 221
University of Akron
Akron, OH 44325-4003
USA
Email: chan@uakron.edu

Chiao-Chen Chang
Department of International Business
National Dong Hwa University
No.1, Sec.2, Da Hsueh Rd., Shoufeng, Hualien 97401
Taiwan
Email: aka@mail.ndhu.edu.tw

Hongmei Chen
School of Information Science and Technology
Southwest Jiaotong University & Key Lab of Cloud Computing and Intelligent Technology
Chengdu, 610031
China
Email: hmchen@swjtu.edu.cn

Igor Chikalov
Mathematical and Computer Sciences and Engineering Division
King Abdullah University of Science and Technology
Thuwal, 23955-6900
Saudi Arabia
Email: igor.chikalov@kaust.edu.sa

Yang-Chieh Chin
Department of Management Science
National Chiao Tung University
1001 Ta-Hsueh Road, Hsinchu 300
Taiwan
Email: jerry110888@gmail.com

Davydov Maksym Volodymyrovych
Lviv Polytechnic National University
12, Bandery, L'viv, 79013
Ukraine
Email: maks.davydov@gmail.com

Paweł Delimata
Institute of Computer Science
University of Rzeszów
2, Dekerta, Rzeszów, 35-030
Poland
Email: pdelimata@wp.pl

Ivo Düntsch
Brock University, St. Catharines, Ontario
Canada, L2S 3A1
Email: duentsch@brocku.ca

Aly A. Fahmy
Faculty of Computers and Information
Cairo University
Egypt
Email: aly.fahmy@gmail.com

Günther Gediga
Department of Psychology, Institut IV, Universität Münster
Fliednerstr. 21, Münster 48149
Germany
Email: gediga@uni-muenster.de

Salvatore Greco
Department of Economics and Business
University of Catania
55, Corso Italia, Catania, 95129
Italy
Email: salgreco@unict.it

Piotr Grochowalski
Institute of Computer Science
University of Rzeszów
2, Dekerta, Rzeszów, 35-030
Poland
Email: piotrg@univ.rzeszow.pl

Jerzy W. Grzymała-Busse
Department of Electrical Engineering and Computer Science
University of Kansas
3014 Eaton Hall
1520 W. 15th St., Lawrence, KS 66045-7621
USA
Email: jerzy@ku.edu
and
Institute of Computer Science, Polish Academy of Science
5, Jana Kazimierza, Warsaw, 01-237
Poland

Osman Gurdal
School of Medicine, IUPUI
975 W. Walnut Street, Indianapolis, IN 46202
USA
Email: ogurdal@iupui.edu

Aboul Ella Hassanien
Faculty of Computers and Information
Cairo University
Egypt
Email: aboitcairo@gmail.com

Hodych Oles Vasyliovych
Lviv Polytechnic National University
12, Bandery, L'viv, 79013
Ukraine
Email: oles.hodych@gmail.com

Jun Hu
Institute of Computer Science and Technology
Chongqing University of Posts and Telecommunications
Chongqing, 400065
P. R. China
Email: hujun@cqupt.edu.cn

Qinghua Hu
Harbin Institute of Technology
Box 458, Harbin 150001, Heilongjiang Province
P. R. China
Email: huqinghua@ hit.edu.cn

Shahid Hussain
Mathematical and Computer Sciences and Engineering Division
King Abdullah University of Science and Technology
Thuwal, 23955-6900
Saudi Arabia
Email: shahid.hussain@kaust.edu.sa

Masahiro Inuiguchi
Graduate School of Engineering Science
1-3 Machikaneyama-cho, Toyonaka, Osaka 560-8531
Japan
Email: inuiguti@sys.es.osaka-u.ac.jp

Ryszard Janicki
Department of Computing and Software
McMaster University
Hamilton, ON
Canada L8S 4K1
Email: janicki@mcmaster.ca

Janusz Kacprzyk
Systems Research Institute, Polish Academy of Sciences
6, Newelska, Warsaw, 01-447
Poland
Email:kacprzyk@ibspan.waw.pl
and
WIT - Warsaw School of Information Technology
6, Newelska, Warsaw, 01-447
Poland

Md. Aquil Khan
The Institute of Mathematical Sciences
Chennai, 600113
India
Email: mdaquilkhan@gmail.com
and
Institute of Mathematics
University of Warsaw
2, Banacha, Warsaw, 02-097
Poland

Ikno Kim
The Graduate School of Information, Production and Systems
Waseda University
2-7 Hibikino, Wakamatsu, Kitakyushu 808-0135
Japan
Email: octoberkim@hotmail.com

Beata Konikowska
Institute of Computer Science, Polish Academy of Sciences
5, Jana Kazimierza, Warszawa, 01-248
Poland
Email: beata.konikowska@ipipan.waw.pl

Maciej Kopczyński
Department of Computer Science
Białystok University of Technology
45A, Wiejska, Białystok, 15-351
Poland
Email: m.kopczynski@pb.edu.pl

Bożena Kostek
Multimedia Systems Department, Gdansk University of Technology
11/12, Narutowicza, Gdańsk, 80-233
Poland
Email: bozenka@sound.eti.pg.gda.pl

Marzena Kryszkiewicz
Institute of Computer Science
Warsaw University of Technology
15/19, Nowowiejska, Warsaw, 00-665
Poland
Email: mkr@ii.pw.edu.pl

Elżbieta Kubera
University of Life Sciences in Lublin
13, Akademicka, Lublin, 20-950
Poland
Email: elzbieta.kubera@up.lublin.pl

Leijun Li
Harbin Institute of Technology
Box 458, Harbin 150001, Heilongjiang Province
P. R. China
Email: lileijun1985@163.com

Tianrui Li
School of Information Science and Technology
Southwest Jiaotong University
Chengdu, 610031
China
Email: trli@swjtu.edu.cn

Chiun-Sin Lin
Department of Management Science
National Chiao Tung University
1001 Ta-Hsueh Road, Hsinchu 300
Taiwan
Email: netec@yahoo.com.tw

Tsau Young Lin
Department of Computer Science
San Jose State University
San Jose, CA 95192-0249
USA
Email: ty.lin@sjsu.edu

Pawan Lingras
Professor, Mathematics and Computing Science
Saint Mary's University, Halifax, Nova Scotia
Canada B3H3C3
Email: pawan.lingras@gmail.com

Kathy J. Liszka
Department of Computer Science, CAS 221
University of Akron
Akron, OH 44325-4003
USA
Email: liszka@uakron.edu

Mei-Chen Lo
Department of Business Management
National United University
No.1, Lienda Rd., Miaoli 36003
Taiwan
Email: meichen_lo@yahoo.com

Ilias Maglogiannis
University of Central Greece
Department of Computer Science and Biomedical Informatics
Papasiopoulou 2-4, PC 35100 Lamia
Greece
Email: imaglo@ucg.gr

Pradipta Maji
Machine Intelligence Unit
Indian Statistical Institute
203, B.T. Road, Kolkata 700108
India
Email: pmaji@isical.ac.in

A. Mani
Department of Pure Mathematics
University of Calcutta
35, Ballygunge Circular Road, Kolkata (Calcutta) 700019
India
Email: a.mani.cms@gmail.com

Victor W. Marek
Davis Marksbury Building
Department of Computer Science
University of Kentucky
Lexington, KY 40506-0633
USA
Email: marek@cs.uky.edu

Shantan R. Marepally
Department of Electrical Engineering and Computer Science
University of Kansas
Lawrence, KS 66045
USA
Email: shantan@ku.edu

Barbara Marszał-Paszek
Institute of Computer Science, University of Silesia
39, Będzińska 39, Sosnowiec, 41–200
Poland
Email: bpaszek@us.edu.pl

Benedetto Matarazzo
Department of Economics and Business
University of Catania
55, Corso Italia, Catania, 95129
Italy
Email: matarazz@unict.it

Mikhail Moshkov
Mathematical and Computer Sciences and Engineering Division
King Abdullah University of Science and Technology
Thuwal, 23955-6900
Saudi Arabia
Email: mikhail.moshkov@kaust.edu.sa

Kanlaya Naruedomkul
Department of Mahtematics, Faculty of Science
Mahidol University
Rama 6 Road, Rajadhevee, Bangkok, 10400
Thailand
Email: scknr@mahidol.ac.th

Hung Son Nguyen
Institute of Mathematics
University of Warsaw
2, Banacha, Warsaw, 02-097
Poland
Email: son@mimuw.ed.pl

Linh Anh Nguyen Institute of Informatics
University of Warsaw
2, Banacha Warsaw, 02-097
Poland
Email: nguyen@mimuw.edu.pl

Nikolski Iouri Volodymyrovych
Lviv Polytechnic National University
12, Bandery, L'viv, 79013
Ukraine
Email: y_nikol@yahoo.com

Hannu Nurmi
Public Choice Research Centre
Department of Political Science and Contemporary History
University of Turku
FIN-20014 Turku
Publicum (Assistentinkatu 7)
Finland
Email: hnurmi@utu.fi

Ewa Orłowska
National Institute of Telecommunications
1, Szachowa, Warsaw, 04-894
Poland
Email: E.Orlowska@itl.waw.pl

Sankar K. Pal
Center for Soft Computing Research
Indian Statistical Institute
203, B.T. Road, Kolkata 700108
India
Email: sankarpal@yahoo.com

Krzysztof Pancerz
Institute of Biomedical Informatics
University of Information Technology and Management
2, Sucharskiego, Rzeszów, 35-225
Poland
Email: kpancerz@wsiz.rzeszow.pl

Przemyslaw Wiktor Pardel
Institute of Computer Science
University of Rzeszów
2, Dekerta, Rzeszów, 35-030
Poland
Email: ppardel@univ.rzeszow.pl
and
Institute of Mathematics
University of Warsaw
2, Banacha, Warsaw, 02-097
Poland

Pasichnyk Volodymyr Volodymyrovych
Lviv Polytechnic National University
12, Bandery, L'viv, 79013
Ukraine
Email: vpasichnyk@gmail.com

Frederick E. Petry
Naval Research Lab
Geospatial Science and Technology Branch
Code 7444
Stennis Space Center, MS 39529
USA
Email: fpetry@nrlssc.navy.mil

Piotr Paszek
Institute of Computer Science, University of Silesia
39, Będzińska 39, Sosnowiec, 41–200
Poland
Email: paszek@us.edu.pl

Sushmita Paul
Machine Intelligence Unit
Indian Statistical Institute
203, B.T. Road, Kolkata 700108
India
Email: sushmita_t@isical.ac.in

Witold Pedrycz
Department of Electrical and Computer Engineering
University of Alberta
Edmonton
Canada T6R 2G7
Email: wpedrycz@ualberta.ca

James F. Peters
Computational Intelligence Laboratory
Department of Electrical & Computer Engineering
University of Manitoba
75A Chancellor's Circle, EITC Room E1-526
Winnipeg, Manitoba
Canada R3T 3E2
Email: jfpeters@ee.umanitoba.ca

Lech Polkowski
Polish–Japanese Institute of Information Technology
86, Koszykowa, Warszawa, 02-008
Poland
Email: polkow@pjwstk.edu.pl

Astrid A. Prinz
Department of Biology
Emory University
O. Wayne Rollins Research Center
Room 2105, 1510 Clifton Road, Atlanta, GA 30322
USA
Email: astrid.prinz@emory.edu

Anna Maria Radzikowska
Faculty of Mathematics and Information Science
Warsaw University of Technology
1, Plac Politechniki, Warsaw, 00-661
Poland
Email: A.Radzikowska@mini.pw.edu.pl

Elisabeth Rakus-Andersson
Blekinge Institute of Technology
School of Engineering
S-37179 Karlskrona
Sweden
Email: Elisabeth.Andersson@bth.se

Zbigniew W. Raś
University of North Carolina
Dept. of Computer Science, Charlotte, NC 28223
USA
Email: ras@uncc.edu
and
Institute of Computer Science
Warsaw University of Technology
15/19, Nowowiejska, Warsaw, 00-665
Poland
and
Institute of Computer Science
Polish Academy of Sciences
5, Jana Kazimierza, Warsaw, 01-248
Poland

Grzegorz Rozenberg
Leiden Center for Natural Computing
Leiden University
Niels Bohrweg 1, 2333 CA Leiden
The Netherlands
Email: rozenber@liacs.nl

Mostafa A. Salama
Department of Computer Science
British University in Egypt
Cairo
Egypt
Email: mostafa.salama@gmail.com

Maria Semeniuk-Polkowska
Chair of Formal Linguistics
University of Warsaw
8–12, Browarna, Warszawa, 00-950
Poland
Email: m.polkowska@uw.edu.pl

Shcherbyna Yuri Mykolaiovych
Ivan Franko National University of L'viv
1, Universytetska, L'viv, 79000
Ukraine
Email: yshcherbyna@yahoo.com

Andrzej Skowron
Institute of Mathematics
University of Warsaw
2, Banacha, Warsaw, 02-097
Poland
Email: skowron@mimuw.ed.pl

Dominik Ślęzak
Institute of Mathematics
University of Warsaw
2, Banacha, Warsaw, 02-097
Poland
and
Infobright Inc.
34, lok. 219, Krzywickiego, Warsaw, 02-078
Poland
Email: slezak@{mimuw.edu.pl,infobright.com}

Roman Słowiński
Institute of Computing Science
Poznań University of Technology
2, Piotrowo, Poznań, 60-965
Poland
Email: rslowinski@cs.put.poznan.pl
and
Systems Research Institute, Polish Academy of Sciences
6, Newelska, Warsaw, 01-447
Poland

Tomasz G. Smolinski
Department of Computer and Information Sciences
Delaware State University
Science Center North
Room 344, 1200 N. DuPont Highway, Dover, DE 19901
USA
Email: tsmolinski@desu.edu

Barbara Sokołowska
Second Department of Internal Medicine
Jagiellonian University Medical College
8, Skawińska, Cracow, 31-066
Poland
Email: basiasok1@gmail.com

Omar S. Soliman
Faculty of Computers and Information
Cairo University
Egypt
Email: dr.omar.soliman@gmail.com

Sumalee Sonamthiang
Institute for Innovative Learning
Mahidol University
999 Phuttamonthon 4 Road, Nakhon Pathom, 73170
Thailand
Email:tew197@gmail.com

Jarosław Stepaniuk
Department of Computer Science
Białystok University of Technology
45A, Wiejska, Białystok, 15-351
Poland
Email: j.stepaniuk@pb.edu.pl

Zbigniew Suraj
Institute of Computer Science
University of Rzeszów
2, Dekerta, Rzeszów, 35-030
Poland
Email: zsuraj@univ.rzeszow.pl

Yu Ru Syau
Department of Information Management
National Formosa University
Huwei 63201, Yunlin
Taiwan
Email: yrsyau@nfu.edu.tw

Andrzej Szałas
Institute of Informatics
University of Warsaw
2, Banacha Warsaw, 02-097
Poland
Email: andrzej.szalas@mimuw.edu.pl
and
Dept. of Computer and Information Science
Linköping University
Linköping, SE-581 83
Sweden

Marcin Szeląg
Institute of Computing Science
Poznań University of Technology
2, Piotrowo, Poznań, 60-965
Poland
Email: mszelag@cs.put.poznan.pl

Tomasz Szmuc
Department of Applied Computer Science
AGH University of Science and Technology
30, Al. Mickiewicza, Cracow, 30-059
Poland
Email: tsz@agh.edu.pl

Marcin Szpyrka
Department of Applied Computer Science
AGH University of Science and Technology
30, Al. Mickiewicza, Cracow, 30-059
Poland
Email: mszpyrka@agh.edu.pl

Hongmei Tian
School of Information Science and Technology
Southwest Jiaotong University
Chengdu, 610031
China
Email: 470777848@qq.com

Li-Shiang Tsay
School of Technology
213 Price Hall, NC A&T State University, Greensboro NC 27411
USA
Email: ltsay@ncat.edu

Shusaku Tsumoto
Department of Medical Informatics
Shimane University
School of Medicin
89-1 Enya-cho, Izumo-city, Shimane 693-8501
Japan
Email: tsumoto@med.shimane-u.ac.jp

Gwo-Hshiung Tzeng
Department of Business Administration
Kainan University
No. 1 Kainan Road, Luchu, Taoyuan 338
Taiwan
Email: ghtzeng@mail.knu.edu.tw
and
Institute of Management of Technology
National Chiao Tung University
1001 Ta-Hsueh Road, Hsinchu 300
Taiwan
Email: ghtzeng@cc.nctu.edu.tw

Alicja Wakulicz-Deja
Institute of Computer Science, University of Silesia
39, Będzińska 39, Sosnowiec, 41–200
Poland
Email: wakulicz@us.edu.pl

Guoyin Wang
Chongqing University of Posts and Telecommunications
Chongqing, 400065
P. R. China
Email: wanggy@ieee.org

Anita Wasilewska
Department of Computer Science
Stony Brook University
Stony Brook, NY
USA
Email: anita@cs.sunysb.edu

Piotr Wasilewski
Institute of Informatics
University of Warsaw
2, Banacha, Warsaw, 02-097
Poland
Email: piotr@mimuw.edu.pl

Junzo Watada
The Graduate School of Information, Production and Systems
Waseda University
2-7 Hibikino, Wakamatsuku, Kitakyushu 808-0135
Japan
Email: junzow@osb.att.ne.jp

Alicja A. Wieczorkowska
Polish-Japanese Institute of Information Technology
86, Koszykowa 86,Warsaw, 02-008
Poland
Email: alicja@poljap.edu.pl

Marcin Wolski
Department of Logic and Philosophy of Science
Maria Curie-Skłodowska University
4, Pl. Marii Curie-Skłodowskiej, Lublin, 20-031
Poland
Email: marcin.wolski@umcs.lublin.pl

Urszula Wybraniec-Skardowska
Group of Logic, Language and Information
Opole University
Poland
Email: uws@uni.opole.pl

JingTao Yao
Department of Computer Science
University of Regina
Regina, Saskatchewan
Canada S4S 0A2
Email: jtyao@cs.uregina.ca

Yiyu Yao
Department of Computer Science
University of Regina
Regina, Saskatchewan
Canada S4S 0A2
Email: yyao@cs.uregina.ca

Sławomir Zadrożny
Systems Research Institute, Polish Academy of Sciences
6, Newelska, Warsaw, 01-447
Poland
Email: zadrozny@ibspan.waw.pl
and
Technical University of Radom
29, Malczewskiego, Radom, 26-600
Poland

Pengfei Zhu
Harbin Institute of Technology
Box 458, Harbin 150001, Heilongjiang Province
P. R. China
Email: zhupengfeifly@gmail.com

Beata Zielosko
Mathematical and Computer Sciences and Engineering Division
King Abdullah University of Science and Technology
Thuwal, 23955-6900
Saudi Arabia
Email: beata.zielosko@kaust.edu.sa
and
Institute of Computer Science
University of Silesia
39, Będzińska St., Sosnowiec, 41-200
Poland

To the memory of

Professor Zdzisław I. Pawlak

Preface

This book is dedicated to the memory of Professor Zdzisław Pawlak, who passed away almost six years ago. He is the founder of the Polish school of Artificial Intelligence and one of the pioneers in Computer Engineering and Computer Science with worldwide influence. He was a truly great scientist, researcher, teacher and a human being.

Professor Pawlak's most important discovery was his invention of the rough set theory in 1982, which gained vast popularity throughout the World. More than 5000 English-language publications and also more than 5000 Chinese-language publications about Pawlak's theory and its applications have been published so far, including many books. Those publications include both specializations and extensions of rough set theory. Their goal is to solve new scientific problems, examining connections between the theory and other approaches and dealing with applications of the theory in practice. Moreover, a number of books devoted to rough sets theory were published worldwide. Numerous conferences, *e.g.*, in China, India, Japan, Canada, USA, in Europe and also recently in Australia and Africa, were organized. The rough sets theory has an immense following in China and research on rough sets also is significantly growing in India.

Rough set theory has attracted worldwide attention with many researchers and practitioners, who have contributed essentially to its development and applications. Rough set theory overlaps with many other theories. Despite this, rough set theory may be considered as an independent discipline in its own right. The rough set approach seems to be of fundamental importance in artificial intelligence and cognitive sciences, especially in research areas such as adaptive and autonomous systems, bioinformatics, data mining and knowledge discovery, decision analysis, expert systems, machine learning, intelligent systems, inductive reasoning, pattern recognition, mereology, digital image processing, digital image analysis and signal analysis. A wide range of applications of methods based on rough set theory alone or in combination with other approaches have been discovered in many areas including: acoustics, bioinformatics, business and finance, chemistry, computer engineering (*e.g.*, data compression, digital image processing, digital signal processing, parallel and distributed computer systems, sensor fusion, fractal engineering),

decision analysis and systems, economics, electrical engineering (*e.g.*, control, signal analysis, power systems), environmental studies, digital image processing, informatics, medicine, molecular biology, musicology, neurology, robotics, social science, software engineering, spatial visualization, Web engineering, and Web mining.

Professor Pawlak inspired many computer scientists and mathematicians both in both Poland and throughout the world. His students and collaborators created research teams in many countries, including, besides of his native Poland, the United States, Canada, Japan, Norway, Sweden and other places. It would be hardly possible t o find a computer science institution, in his native Poland without encountering a faculty influenced by Professor Pawlak. His scientific achievements continue to inspire his many students still working in these institutions and also the next generations of students.

This book prepared in two volumes contains more than 50 chapters. This demnonstrates that the scientific approaches discovered by of Professor Zdzisław Pawlak, especially the rough set approach as a tool for dealing with imperfect knowledge, are vivid and intensively explored by many researchers in many places throughout the world. The submitted papers prove that interest in rough set research is growing and is possible to see many new excellent results both on theoretical foundations and applications of rough sets alone or in combination with other approaches.

The book is divided into two volumes.

The first volume contains the following chapters.

Chapter "Professor Zdzisław Pawlak (1926-2006): Founder of the Polish School of Artificial Intelligence" by Andrzej Skowron, Mihir Kr. Chakraborty, Jerzy Grzymała-Busse, Victor Marek, Sankar K. Pal, James Peters, Grzegorz Rozenberg, Dominik §lęzak, Roman Słowiński, Shusaku Tsumoto, Alicja Wakulicz-Deja, Guoyin Wang, and Wojciech Ziarko is dedicated to the memory of Professor Zdzisław Pawlak, founder of the Polish school of Artificial Intelligence and one of the pioneers in Computer Engineering and Computer Science with worldwide influence. In particular, it contains a few selected speech fragments pointing to Professor's scientific achievements along with personal comments by Andrzej Skowron, one of the authors, on features of Professor Pawlak as a truly great scientist, teacher and a human being.

The list of works by Professor Zdzisław Pawlak, prepared by Andrzej Skowron, is included in the chapter "List of Works by Professor Zdzisław Pawlak (1926-2006)".

Chapter "Rough Sets: From Rudiments to Challenges" by Hung Son Nguyen and Andrzej Skowron contains a survey about rough sets together with comments on possible future research directions and challenges.

In Chapter "Zdzisław Pawlak, Databases and Rough Sets", Victor W. Marek presents work of Zdzisław Pawlak in the area of databases and the extension of that work to the theory of rough sets. In particular, the author concentrates on motivations of Professor Pawlak for introducing information storage and retrieval systems and describes how this, eventually, led to rough sets theory.

Chapter “jMAF - Dominance-based Rough Set Data Analysis Framework” by Jerzy Błaszczyński, Salvatore Greco, Benedetto Matarazzo, Roman Słowiński, and Marcin Szeląg, presents a rough set data analysis software jMAF. It employs java Rough Set (jRS) library in which are implemented data analysis methods provided by the (variable consistency) Dominance-based Rough Set Approach (DRSA). The chapter also provides some basics of the DRSA and of its variable consistency extension.

Chapter “Dynamic Programming Approach for Exact Decision Rule Optimization” by Talha Amin, Igor Chikalov, Mikhail Moshkov, and Beata Zielosko, discusses an extension of dynamic programming approach to sequential optimization of exact decision rules relative to the length and coverage. The chapter also contains results of experiments with decision tables from UCI Machine Learning Repository.

Chapter “Approaches for Updating Approximations in Set-Valued Information Systems While Objects and Attributes Vary with Time" by Hongmei Chen, Tianrui Li, and Hongmei Tian focuses on studying principles for incrementally updating approximations in a set-valued information system while attributes and objects are added. Methods for updating approximations of concepts in set-valued information systems with attributes and objects changing simultaneously are presented. Experimental evaluation of the proposed methods is included.

Ivo Düntsch and Günther Gediga describe in Chapter “On the gradual evolvement of things” the generic properties of a visual system without hard coding the environment. As a measure of approximation, Pawlak’s approximation quality is used. The considerations are related to some ideas of the Gibson ecological approach to perception.

The Chapter “On Empirical Comparison of Rule Sets Induced by LERS and Probabilistic Rough Classification" by Jerzy W. Grzymała-Busse, Shantan R. Marepally, and Yiyu Yao explores an extension of rough set theory, based on probability theory. In particular, parameterized approximations are used together with the corresponding positive, boundary, and possible rules. The results of parameter tuning on the quality of the induced classifiers based on such rules are reported.

In Chapter “Exploring Neighborhood Structures with Neighborhood Rough Sets in Classification Learning”, Qinghua Hu, Leijun Li, and Pengfei Zhu introduce neighborhoods of samples to granulate the universe and use the neighborhood granules to approximate classification, thus a model of neighborhood rough sets is derived. Some machine based on the model learning algorithms, including boundary sample selection, feature selection and rule extraction, are developed.

Chapter “Rough Representations of Ill-Known Sets and Their Manipulations in Low Dimensional Space” by Masahiro Inuiguchi, focuses on investigations of the rough representations of graded ill-known sets and the manipulations of possibility and necessity measures of graded ill-known sets using general conjunction and implication functions in the universe.

In Chapter “Property-Driven Rough Sets Approximations of Relations” by Ryszard Janicki, a problem of approximating an arbitrary relation by a relation with desired properties is formally defined and analysed. The concepts of α-lower and upper approximatios are introduced and their properties are discussed. Two special cases, approximation by partial orders and approximation by equivalence relations are discussed in detail.

Chapter “Towards a Comprehensive Similarity Analysis of Voting Procedures Using Rough Sets and Similarity Measures" by Janusz Kacprzyk, Hannu Nurmi, and Sławomir Zadrożny, presents and approach to the evaluation of similarity of voting procedures with respect to a set of criteria which are widely used in the social choice literature. First, a qualitative rough sets based analysis is proposed, and then an additional quantitative analysis is added by using two measures of similarity of binary patters widely employed in many areas , i.e. the one based on the Hamming distance and the one due to Jaccard-Needham. The approach proposed constitutes a step towards the solutution of a difficult problem of determining the (degree of) similarity of voting procdures by providing a comprehensive qualitative and quantitative view.

Chapter “Algebras for Information Systems” by Md. Aquil Khan and Mohua Banerjee, introduces algebraic structures for different kinds of information systems together with the representation theorems for classes of algebras corresponding to these structures. Finally, equational logics for deterministic, incomplete and non-deterministic information systems are presented.

Chapter “DNA Rough-Set Computing in the Development of Decision Rule Reducts” by Ikno Kim, Junzo Watada, and Witold Pedrycz, introduces a DNA rough-set computation technique for dealing with the NP-hard problem problem of optimal reduction of decision rules. The proposed technique is a composition of computational DNA molecular techniques and is effectively employed to alleviate the computational complexity of the considered optimization problem.

Chapter “Three-valued Logic for Reasoning about Covering-Based Rough Sets” by Beata Konikowska, introduces a tool for reasoning about covering-based rough sets in the form of three-valued logic with logical values corresponding to positive, negative region and the boundary regions of a set. The author presents a strongly sound sequent calculus for this logic, together with the proof of strong completeness for a subset of its language.

In Chapter “Music Information Retrieval in Music Repositories” by Bożena Kostek, the key concepts associated with automated music information retrieval and music recommendation are discussed. Experiments on a constructed music database with different kinds of classifiers are reported. A proposal for music retrieval and annotation aided by gaze tracking is also discussed.

In Chapter “Rough Support Vectors: Classification, Regression, Clustering” by Pawan Lingras, Parag Bhalchandra, Cory Butz, and S. Asharaf, it is shown that the concepts of margins in support vector techniques provides a natural relationship

with the rough set theory. The authors describe rough set theoretic extensions of support vector technologies for classification, prediction, and clustering. The theoretical formulations of rough support vector machines, rough support vector regression, and rough support vector clustering are supported with a summary of experimental results.

Chapter “Logic-based Roughification” by Linh Anh Nguyen and Andrzej Szałas includes novel roughification techniques for constructing equivalence/similarity relations adequate for Pawlak-like approximations. The authors also present applications of the proposed approach in granulating relational databases as well as concept learning and approximation in description logic-based information systems.

Chapter “How Near Are Zdzisław Pawlak’s Paintings? Merotopic Distance Between Regions-of-Interest” by James F. Peters, presents an approach to measuring the nearness of Pawlak’s paintings in terms of the merotopic distance between collections of neighborhoods in image regions-of-interest.

Chapter “An Implementation of the Zdzisław Pawlak Idea for Reasoning about Uncertainty. Approximate Reasoning by Parts” by Lech Polkowski and Maria Semeniuk-Polkowska, presents a mereological calculus of parts, in which concepts become elementary objects and relations among them are expressed as relations of being parts to degrees. This analysis allows, in particular, for approximations to various degrees. A characterization of continuous rough inclusions parallel to the Menu-Pavelka characterization of continuous t–norms is proposed.

Chapter “Granular Concept Mapping and Applications” by Sumalee Sonamthiang, Kanlaya Naruedomkul, and Nick Cercone, presents a granular concept hierarchy (GCH) construction and mapping of the hierarchy for granular knowledge. A granule description language and granule measurements are proposed to enable mapping for an appropriate granular concept that represents sufficient knowledge to solve the problem at hand. Applications of GCH are demonstrated through learning of higher order decision rules.

Chapter “Rough Sets and Medical Differential Diagnosis” by Shusaku Tsumoto, discusses a correspondence between the core ideas of rough sets and medical differential diagnosis.

In Chapter “Science and Semantics: A Note on Rough Sets and Vagueness” by Marcin Wolski, rough set theory is presented against the background of recent philosophical discussions about vagueness and empirical sciences.

The second volume contains the following chapters.

In chapter “From Logic to Computer Science – A Personal Experience”, Anita Wasilewska explains why she is so grateful to Professor Zdzisław Pawlak whom was possible to meet on her way from mathematics to computer science.

Chapter "Knowledge algebras and their discrete duality" by Ewa Orlowska and Anna Maria Radzikowska introduces a class of algebras referred to as knowledge

algebras and a class of knowledge frames. Representation theorems for these classes leading to a discrete duality are proved.

Chapter "Comparison of Greedy Algorithms for Decision Tree Optimization" by Abdulaziz Alkhalid, Igor Chikalov, and Mikhail Moshkov is devoted to the comparison of 16 types of greedy algorithms for decision tree construction with optimal decision trees generated by the dynamic programming approach. Optimization is performed relative to minimal values of different parameters of decision trees. The results of experiments are reported and discussed.

In Chapter "A Review of the Knowledge Granulation Methods: Discrete vs Continuous Algorithms" by Piotr Artiemjew, rough inclusions and some of their weaker variants are used to define similarity relations. Applications to classification problems are discussed.

In Chapter "Game-theoretic Rough Sets for Feature Selection" by Nouman Azam and JingTao Yao, a game-theoretic rough sets based method is formulated for selecting important features by combining multiple measures representing importance levels for a feature. The method incorporates the measures as players in a game where each player employs a three-way decision in selecting features. The included demonstrative example suggests that this method may be useful for feature selection in text categorization.

Chapter "A Clustering Approach to Image Retrieval Using Range Based Query and Mahalanobis Distance" by Minakshi Banerjee, Sanghamitra Bandyopadhyay, and Sankar K. Pal, puts forward a new approach to address a general purpose content-based image retrieval task. The effectiveness of the proposed algorithm is demonstrated with increased accuracy and reduced retrieval time.

Chapter "Classifiers Based on Data Sets and Domain Knowledge: A Rough Set Approach" by Jan G. Bazan, Stanisława Bazan-Socha, Sylwia Buregwa-Czuma, Przemysław Pardel, Andrzej Skowron, and Barbara Sokolowska, presents the ontology approximation method for inducing complex classifiers from experimental data and domain knowledge. The experimental results on different data sets are reported, in particular on (i) data set generated by a vehicular traffic simulator, (ii) real-life data set concerning coronary heart disease obtained from Second Department of Internal Medicine, Jagiellonian University Medical College, Cracow, Poland, and (iii) real-life data set concerning respiratory failure obtained from Neonatal Intensive Care Unit in the Department of Pediatrics, Jagiellonian University Medical College, Cracow, Poland.

Chapter "Incorporating Rough Data in Database Design for Imprecise Information Representation" by Theresa Beaubouef and Frederick E. Petry, provides discussions of how it is possible to design relational databases to allow the incorporation of uncertain data characterized using rough set theory. This included Entity-Relationship modeling, rough functional dependencies and rough normal forms. Security issues as dealt with in statistical databases are also discussed as well as an example of the representation of uncertain spatial data by rough sets.

The approach based on a pragmatic view of representation of knowledge is used in Chapter 'Rough Pragmatic Description Logic" by Zbigniew Bonikowski, Edward Bryniarski, and Urszula Wybraniec-Skardowska for introducing and for investigation of a rough description logic.

Chapter "Application of Rough Set Theory to Sentiment Analysis of Microblog Data" by Chien-Chung Chan and Kathy J. Liszka, presents the use of rough set theory to formulate sentimental approximation spaces based on key words for assessing sentiment of microblogging messages. The sentimental approximation space provides contextual sentiment from the entire collection of messages, and it enables the evaluation of sentiment of different subjects, not in isolation, but in context. The sentimental approximation space offers potentially more insightful information about a subject than simple polarity answers of positive or negative.

Chapter "Relationships for Cost and Uncertainty of Decision Trees" by Igor Chikalov, Shahid Hussain, and Mikhail Moshkov, presents the results of studies on the relationships between the cost and the uncertainty of decision trees as well as on the relationships between the number of nodes and the depth in the case of exact decision trees. The developed tools are based on dynamic programming approach and are applicable only to relatively small decision tables. The results of experiments are reported and discussed.

Chapter "The Impact Rules of Recommendation Sources for Adoption Intention of Micro-Blog Based on DRSA with Flow Network Graph" by Yang-Chieh Chin, Chiao-Chen Chang, Chiun-Sin Lin, and Gwo-Hshiung Tzeng, focuses on the micro-blog (*i.e.*, a new communication channel with which people share short text messages on public and private networks) on Facebook. The main purpose of the reported study is to explore and compare what recommendation sources influence the intention to use micro-blogs and to combine the personal characteristics/ attributes of gender, daily internet hour usage and past use experience to infer the usage of micro-blogs decision rules using a dominance-based rough-set approach (DRSA) with flow network graph.

Chapter "Providing Feedback in Ukrainian Sign Language Tutoring Software" by M.V. Davydov, I.V. Nikolski, V.V. Pasichnyk, O.V. Hodych, and Y.M. Scherbyna, focuses on video recognition methods implemented as part of the Ukrainian Sign Language Tutoring Software. The proposed software system for sign language recognition supports user interaction with the system during learning of signs and the verification process. The developed feedback mechanism significant improves the training experience for users. The results of experiments are reported and discussed.

Chapter "Hybrid Methods in Data Classification and Reduction" by Paweł Delimata and Zbigniew Suraj summarizes numerous results of the authors on issues of data reduction, feature subset selection and classifier construction. In particular, applications of reducts, deterministic and inhibitory decision rules for feature selection are presented.

Chapter "Uncertainty Problem Processing with Covering Generalized Rough sets" by Jun Hu and Guoyin Wang, focuses on applications of the covering generalized rough set approach. There are proposed two models: knowledge reduction model and covering generalized rough fuzzy model.

Chapter "Hardware Implementations of Rough Set Methods in Programmable Logic Devices" by Maciej Kopczynski and Jarosław Stepaniuk, discusses existing results on hardware realization of rough set algorithms in the Field Programmable Gate Array (FPGA) logic devices.

In Chapter "Determining Cosine Similarity Neighborhoods by Means of the Euclidean Distance", Marzena Kryszkiewicz presents a scalable method for computing cosine similarity neighborhoods of vectors by employing the Euclidean distance applied to $(\alpha-)$normalized forms of these vectors and the triangle inequality. There are considered three types of sets of cosine similar vectors: all vectors the similarity of which to a given vector is not less than an ε threshold value and two variants of k-nearest neighbors of a given vector.

Chapter "Time Variability-Based Hierarchic Recognition of Multiple Musical Instruments in Recordings" by Elżbieta Kubera, Alicja A. Wieczorkowska, and Zbigniew W. Raś, focuses on automatic identification of musical instruments in polyphonic audio recordings. The reported experiments demonstrate that the performance of classifiers enhanced, in particular by new temporal parameters introduced to supply additional temporal information for the timbre recognition, lead to improvement of the classifier performance.

In chapter "Unifying Variable Precision and Classical Rough Sets: Granular Approach" by Tsau Young Lin and Yu Ru Syau it is shown that neighborhood systems (NS) can integrate Ziarko's variable precision rough set model (VPRSM) and Pawlak 's classical rough sets into one concept.

Chapter "Fuzzy Hybrid MCDM for Building Strategy Forces" by Mei-Chen Lo and Gwo-Hshiung Tzeng, adopts Fuzzy multiple criteria decision making methods and discuss how the technology, marketing and research & development forces operate and suggests ways of adjusting to them, and, where possible, of taking advantage of them.

Chapter "Rough Set Based Feature Selection: Criteria of Max-Dependency, Max-Relevance, and Max-Significance" by Pradipta Maji and Sushmita Paul, reports a rough set based feature selection algorithm called maximum relevance - maximum significance (MRMS), and its applications on quantitative structure activity relationship (QSAR) and gene expression data. The importance of rough set theory for computing both relevance and significance of the features is also established. The results of experiments are reported and discussed.

In Chapter "Towards Logics of Some Rough Perspectives of Knowledge" by A. Mani, semantic frameworks for dealing with such issues as concepts of relative consistency of knowledge, conflict representation and resolution are introduced and

developed. The proposed semantics may be of interest for multi-agent systems, dynamic spaces and collections of general approximation spaces.

In Chapter "Classifiers Based on Nondeterministic Decision Rules" by Barbara Marszał-Paszek and Piotr Paszek, classifiers based on rough set theory and nondeterministic decision rules are discussed. The reported experiments are showing that enhancing rule-based classifiers with nondeterministic rules may lead to increasing of the classification quality.

In Chapter "Approximation and Rough Classification of Letter-like Polygon Shapes" by Elisabeth Rakus-Andersson, is presented a rough set based method to classification of discrete two-dimensional point sets resembling some letters together with a rough set technique for verifying decisions about the primary recognitions of the curves' appearance as letter shapes. The results are utilized in the classifications of internet packet streams or in the analysis of wave signals typical of, *e.g.*, medical examinations.

Chapter "Rough Set-based Identification of Heart Valve Diseases Using Heart Sounds" by Mostafa A. Salama, Omar S. Soliman, Ilias Maglogiannis, Aboul Ella Hassanien, and Aly A. Fahmy, presents an application of the rough set approach to classification of heart sound diseases using heart sounds. The reported experiments are showing that the rough set based approach outperforms several other well known machine learning techniques.

In Chapter "Rough Sets and Neuroscience" by Tomasz G. Smolinski and Astrid A. Prinz, examples of the existing and potential applications of rough set theory (and its hybridizations) in neuroscience and neurology are presented. Moreover, a discussion of further development of rough-neural computing, stimulated by relationships of rough sets with neuroscience, is provided.

In Chapter "On Knowledge Representation and Automated Methods of Searching Information in Bibliographical Data Bases: A Rough Set Approach" by Zbigniew Suraj, Piotr Grochowalski, and Krzysztof Pancerz, is presented an approach to searching for information in bibliographical data bases founded on rough set theory and the domain knowledge represented by ontologies. The reported experiments are performed on data gathered in the Rough Set Database System (RSDS).

In Chapter "Design and Verification of Rule-Based Systems for Alvis Models" by Marcin Szpyrka and Tomasz Szmuc, is presented a method of encoding and verification of rule-based systems with the Haskell functional language in order to include them into Alvis, a modeling language designed for embedded systems that provides a possibility of a formal model verification.

Chapter "On Objective Measures of Actionability in Knowledge Discovery" by Li-Shiang Tsay and Osman Gurdal, is included a rough set method for generating a set of rules by utilizing the domain experts' prior knowledge to formulate its inputs and to evaluate the observed regularities it discovers. The generated rule overcomes the traditional data-centered pattern mining resulting to bridge the gap and enhance real-world problem solving capabilities.

In Chapter "Pseudometric Spaces from Rough Sets Perspective" by Piotr Wasilewski, relationships between approximation spaces and pseudometric spaces are presented. Investigations are focused on the class of pseudometric spaces which are lower bounded in each point since open sets in these spaces coincide with definable sets of some prescribed approximation spaces.

Editors of this book are proud to present the readers this book.

Andrzej Skowron and Zbigniew Suraj
Warszawa, Rzeszów, March 2012

Acknowledgement

We are greatly indebted to all Authors of chapters submitted to this book, the Members of Editorial Committee, i.e., Janusz Kacprzyk, Sankar K. Pal, Roman Słowiński, Dominik Ślęzak, Shusaku Tsumoto, Guoyin Wang, Wojciech Ziarko who kindly joint us in the work on the book and to all reviewers of submitted papers including Mohua Banerjee, Jan Bazan, Anna Gomolińska, Salvatore Greco, Piotr Hońko, Andrzej Janusz, Beata Konikowska. Victor Marek, Hung Son Nguyen, Tuan Trung Nguyen, Witold Pedrycz, James Peters, Lech Polkowski, Sheela Ramanna, Piotr Synak, Dominik Ślęzak, Piotr Wasilewski, Alicja Wieczorkowska, Marcin Wolski for their work and help in making this important book available.

We extend an expression of gratitude to Professors Janusz Kacprzyk and Lakhmi C. Jain, to Dr. Thomas Ditzinger and to the Series *Intelligent Systems Reference Library* staff at Springer for their support in making this book possible.

Andrzej Skowron was partially supported by the grant NN516 077837, from the Ministry of Science and Higher Education of the Republic of Poland, the National Centre for Research and Development (NCBiR) under grant SP/I/1/77065/10 by the Strategic scientific research and experimental development program: “Interdisciplinary System for Interactive Scientific and Scientific-Technical Information".

Andrzej Skowron and Zbigniew Suraj

Warszawa, Rzeszów, March 2012

Contents

Chapter 1
From Logic to Computer Science – A Personal Experience

Anita Wasilewska

Abstract. The article explains why the author is so grateful to Professor Zdzisław Pawlak whom was possible to meet on her way from mathematics to computer science.

Keywords: Polish School of Mathematics, Warsaw and Lvov Schools of Mathematics, Warsaw-Lvov School of Logic, the Rasiowa-Pawlak seminar, rough sets.

The origins of Foundational Studies can be traced back to David Hilbert ([8]), a German mathematician, recognized as one of the most influential and universal mathematicians of the 19th and early 20th centuries. In 1920 he proposed a research project that became known as Hilbert's Program. He wanted mathematics to be formulated on a solid and complete logical foundation. He believed that in principle this could be done, by showing that all of mathematics follows from a correctly-chosen finite system of axioms and that some such axiom system is provably consistent. But in 1931 Kurt Gödel ([3]) showed that Hilbert's grand plan was impossible as stated. He proved in what is now called Gödel's Incompleteness Theorem that any non-contradictory formal system, which was comprehensive enough to include at least arithmetic, cannot demonstrate its completeness by way of its own axioms.

Nevertheless Hilbert's and Gödel's work led to the development of recursion theory and then mathematical logic and foundations of mathematics as autonomous disciplines. It inspired works of Alonzo Church and Alan Turing that became the basis for theoretical computer science and also led to the creation and development of a unique phenomenon which became to be known as the Polish School of Mathematics ([5]).

The term **Polish School of Mathematics** refers to groups of mathematicians of the 1920's and 1930's working on common subjects. The main two groups were

Anita Wasilewska
Department of Computer Science, Stony Brook University,
Stony Brook, NY, USA
e-mail: anita@cs.sunysb.edu

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 1–5.
springerlink.com

situated in Warsaw and Lvov (now Lviv, the biggest city in Western Ukraine). We talk hence more specifically about **Warsaw and Lvov Schools of Mathematics**, and additionally of **Warsaw-Lvov School of Logic** working in Warsaw.

When examining twentieth century mathematics one can't not notice the surprising depth, originality and quantity of Polish contributions to the discipline. Any list of important twentieth century mathematicians contains Polish names in a frequency out of proportion to the size of the country. Moreover, such creativity and mathematical influence developed in a country that had little tradition in research, that was partitioned by Russia, Germany, and Austria and was under foreign domination from 1795 until the end of World War I, and whose educational institutions were for over 200 years suppressed by respective foreign powers. What was to become known as the Polish School of Mathematics was possible because it was carefully planned, agreed upon, and executed.

University of Warsaw opened in 1918 with Janiszewski, Mazurkiewicz, and Sierpiński as professors of mathematics. The three initiated Janiszewki's proposals with Warsaw serving as the proposed mathematical research center, hence the name of *Warsaw School of Mathematics.* They chose logic, set theory, point-set topology and real functions as the area of concentration. Their journal *Fundamenta Mathematicae* founded in 1920 and is still in print was the *first specialized mathematical journal in the world*. The choice of title was deliberate to reflect that all areas were to be connected with foundational studies. It should be remembered that at the time these areas had not yet received full acceptance by the mathematical community. The choice reflected both insight and courage.

Some other members of Warsaw school were: Kazimierz Kuratowski, Edward Marczewski, Bronisław Knaster, Stanisław Saks, Karol Borsuk, and after the the second world war, Roman Sikorski, Nachman Aronszajn, and Samuel Eilenberg who emigrated in the 1930's to the USA.

The notable logicians of the **Lvov-Warsaw School of Logic**, working at Warsaw, included Stanisław Leśniewski, Adolf Lindenbaum, and Alfred Tarski (since 1942 in Berkeley, [2]), Jan Łukasiewicz, Andrzej Mostowski, and after the second world war Helena Rasiowa.

After about ten years, with the growth in numbers a second center concentrating on functional analysis was started in Lvov, where Banach and Steinhaus were professors, and the journal *Acta Arithmetica* devoted to functional analysis was founded in 1929. The Lvov center became consequently known as Lvov School of Mathematics. Some other members of this school were: Stanisław Mazur, Stanisław Ulam ([9]), Juliusz Schauder, and Mark Kac ([4]).

While both centers were very strong, cooperation between them was very good and their identities merged into the **Polish School of Mathematics**. For example, when the envisioned monograph series was initiated under the series title *Mathematical Monographs* (Monografie Matematyczne) in 1931, the editor was Wacław Sierpiński in Warsaw, and the first volume *Theorie Operations Lineares* by Stefan Banach appeared in Lvov. By 1936 the Polish Journals and Monographs had become widely read and highly respected, the achievements of Polish mathematicians

had gained international recognition, and the country's mathematical community was spirited and active.

World War II broke out in 1939 and Poland was to suffer great losses to its mathematical community. Lvov University ceased to belong to Poland, but Steinhaus and Knaster carried on their work and recreated some of its original community in western Poland's Wroclaw.

Shortly after the war Warsaw University reopened with Kuratowki, Sierpiński, Mostowski, and Borsuk as Mathematics Professors. They were soon joined there by Mostowski's students Rasiowa, Sikorski, and the others, too numerous to be listed.

In the early sixties Rasiowa, by then herself a prominent world logician and a driving force in the field of Algebraic Logic ([7]) started her collaboration with Zdzisław Pawlak ([6]), an engineer turned mathematician, turned computer scientist. They were one of the first to realize great importance of foundational studies in newly created field of computer science. Since then until her retirement they lead, at Mathematic Institute of Warsaw University a weekly **Foundations of Computer Science** research seminar. The seminar was designed mainly for faculty but advanced Mathematics and Computer Science students were allowed to attend. All students at that time were in what is called a 5 years Master Program and Ph.D. level Graduate Study didn't exist yet. After completing a Master degree one would be employed by the University as a lecturer, progress to senior lecturer, teach and use these kind seminars for educational and research development. One would solve problems which were often posed during the meetings, write and publish papers and in due time, after building evidence of scientific independence would work towards a Ph.D. Thesis. After obtaining Ph.D one would often be promoted to assistant professor and keep building more and more independent research and advise students. As the next step one would be promoted, after the defence of a next degree called Habilitation to the position of associate professor. The promotion to a Professor was (and still is) a State Affair, as it had to be accepted and signed by the President of the country. The process was long (definitely much longer then current processes there and everywhere) and not precisely defined. What was precise, at each level, was the level of scientific results. The seminar attendees formed hence quite a big crowd of various interests and stages of scientific development. Altogether it was a very stimulating and exciting place to belong to.

The weekly Rasiowa-Pawlak seminar was, of course not the only one at the Mathematics Department, or at the Mathematic Institute of Polish Academy of Science (IMPAN). There were *Logic* seminar of Rasiowa and Traczyk (from Warsaw Polytechnic), Mostowski' s *Foundation of Mathematics* seminars at both IMPAN and University, there also was Pawlak's student Blikle and Mazurkiewicz seminar at Informatics Institute of Polish Academy of Science to mention only these I regularly attended and collaborated with their participants.

At that time some founders and early members of the Polish School of Mathematics were still alive and so one could attend Mazur (Functional Analysis), Kuratowski-Borsuk (Topology), Sikorski (Real Functions) seminars and one could even meet one of its most prominent seniors, a gentle and courteous Sierpiński.

Rasiowa ([1]) and Pawlak became in 1977 the founders of *Fundamenta Informaticae* one of the the first world journal specialized in foundation of computer science. It was published by the Polish Mathematical Society as Series IV of the annals "Annales Societatis Mathematicae Polonae". It is currently published by IOS Press under the auspices of the European Association for Theoretical Computer Science with Pawlak's student Andrzej Skowron as its editor in chief since Rasiowa death in 1994[1]. Pawlak served as its co- Editor until his own death. in 2006. The choice of the title was again deliberate. It reflects not only the subject, but also stresses that the new research area being developed is a direct continuation of the tradition of the Polish School of Mathematics.

Such was the environment and tradition I scientifically and personally grew up with. Now almost 50 years later I can see and say that it was a very special one and that I and others of my generation were very lucky. At that time we didn't see it as anything special and thought that the whole world was like that—if not better. But maybe this is the thinking and privilege of the young.

I entered the Mathematic Department of Warsaw University as an undergraduate student in 1962. I started to work there as a faculty in Mathematic Department in 1967 as a member of Rasiowa's Logic and Foundations of Computer Science Division and continued until 1980 (on leave till 1983) when I departed for USA with my then 18 months old daughter Agatha to visit Mathematics Departments at Yale and Wesleyan Universities. The World and Polish politics intervened and what was supposed to be a one year visit became a new life in a new country. The new life also encompassed a new profession, as in 1986 I joined the faculty at Computer Science Department at Stony Brook University and became a full time computer scientist. Looking back at my scientific history this transition from mathematics to computer science seems now inevitable. Nevertheless I don't know if it would have happened if not for a re-appearance in my life and a new influence of Zdzisław Pawlak. It was year 1983 and I was teaching Mathematics at Lafayette College in Easton, PA., one hour drive from New York City. Due to my political involvement I couldn't go back to Poland and thought I would never see it again but my friends and former colleagues started to travel and Pawlak was one of my unexpected, respected and very welcomed quests. It was a joy to make him visit New York and watch him taking hundreds of (excellent!) pictures of the city and of my 5 years old daughter making cart-wheels on the streets. Her freedom of behavior and movements fascinated him. We, the three of us, would take walks in Pocono Mountains and swim in free floating Deleware river and he would talk about his newest idea and passion: rough sets. He left for Warsaw leaving me a stack of papers to read and kept sending more over the years to come. Such was the beginning of my now already lifelong involvement in Rough Sets based research. It also was the beginning of his frequent visits and of our friendship that lasted until his death. Two years after his first visit I was ready for a change and was interviewing for a position in Computer Science Departments giving talks with all my strength and enthusiasm about Rough Sets and my new results. The transition to become a full time computer scientists wouldn't be possible

[1] Since 2010 Damian Niwiński became Editor-in-Chief of Fundamenta Informaticae (eds. note).

of course without years of my previous foundations of computer science research and results, many of them influenced, or directly connected to Pawlak's work at that time.

References

1. Bartol, W., Orlowska, E., Skowron, A.: Helena Rasiowa, 1917-1994. Bulletin of the European Association for Theoretical Computer Science 62, 353–366 (1997)
2. Burdman Feferman, A., Feferman, S.: Alfred Tarski: Life and Logic. Cambridge University Press, Cambridge (2004)
3. Feferman, S., Dawson, J.W., Kleene, S.C., Moore, G.H., Solovay, R.M., van Heijenoort, J.: Collected Works by Kurt Gödel. Oxford University Press, New York (1986)
4. Kac, M.: Enigmas of Chance: An Autobiography. Sloan Foundation Series. Harper and Row, New York (1985)
5. Kuratowski, K.: A Half Century of Polish Mathematics. Pergamon, New York (1980)
6. Peters, J.F., Skowron, A.: Zdzislaw Pawlak life and work. Information Sciences 177(1), 1–2 (2007)
7. Rasiowa, H., Sikorski, R.: The Mathematics of Metamathematics. PWN, Warszawa (1963)
8. Reid, C.: Hilbert. Springer, Berlin (1996)
9. Ulam, S.M.: Adventures of a Mathematician. Charles Scribner's Sons, New York (1976)

Chapter 2
Knowledge Algebras and Their Discrete Duality

Ewa Orłowska and Anna Maria Radzikowska

Abstract. A class of knowledge algebras inspired by a logic with the knowledge operator presented in [17] is introduced . Knowledge algebras provide a formalization of the Hintikka knowledge operator [8] and reflect its rough set semantics. A discrete duality is proved for the class of knowledge algebras and a corresponding class of knowledge frames.

Keywords: Boolean algebra, knowledge operator, knowledge algebra, knowledge frame, rough set, discrete duality, representation theorem, canonical frame, complex algebra.

2.1 Introduction

In this chapter we present and discuss a class of algebras referred to as knowledge algebras. They are Boolean algebras with a unary operator intended to reflect in an abstract setting the rough-set-based semantics of a knowledge operator introduced in [17]. We define a class of knowledge frames and prove that the class of knowledge algebras and the class of these frames are dual to each other in the sense of a discrete duality.

Ewa Orłowska
National Institute of Telecommunications
ul. Szachowa 1, Warsaw, Poland
e-mail: E.Orlowska@itl.waw.pl

Anna Maria Radzikowska
Faculty of Mathematics and Information Science
Warsaw University of Technology
Plac Politechniki 1, 00-661 Warsaw, Poland
e-mail: A.Radzikowska@mini.pw.edu.pl

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 7–20.
springerlink.com

Discrete duality is a relationship between classes of algebras and classes of relational systems. Following terminology of non-classical logic, these relational systems are referred to as *frames*. Since a topology is not involved in the construction of frames, they may be viewed as having a discrete topology. Establishing a discrete duality between these two classes requires the following steps. Let $\mathcal{A}lg$ be a class of algebras and let $\mathcal{F}rm$ be a class of frames.

Step 1: For every algebra $\mathcal{A} \in \mathcal{A}lg$, define its *canonical frame* $\mathfrak{Cf}(\mathcal{A})$ and show that $\mathfrak{Cf}(\mathcal{A}) \in \mathcal{F}rm$.

Step 2: For every frame $\mathcal{X} \in \mathcal{F}rm$, define its *complex algebra* $\mathfrak{Cm}(\mathcal{X})$ and show that $\mathfrak{Cm}(\mathcal{X}) \in \mathcal{A}lg$.

Step 3: Show the following theorems:

(a) Every $\mathcal{A} \in \mathcal{A}lg$ is embeddable into $\mathfrak{Cm}(\mathfrak{Cf}(\mathcal{A}))$.
(b) Every $\mathcal{X} \in \mathcal{F}rm$ is embeddable into $\mathfrak{Cf}(\mathfrak{Cm}(\mathcal{X}))$.

Canonical frames correspond to dual spaces in the sense of Priestley-style duality ([3]). Complex algebras of canonical frames correspond to canonical extensions in the sense of Jónsson and Tarski ([11]). In the setting of discrete dualities, the canonical extension is built from two structures which explicitly refer to their algebraic and relational origin.

A discrete duality leads to so-called *duality via truth* ([21]). Duality via truth amounts to saying that the concept of truth associated with an algebraic semantics of a formal language determined by a class $\mathcal{A}lg$ of algebras is equivalent to the concept of truth associated with its relational (Kripke-style) semantics determined by a class $\mathcal{F}rm$ of frames. In other words, the same formulas are true in both classes of semantical structures. General principles and applications of discrete duality are briefly presented in [22].

The chapter is organized as follows. In Section 2.2 we recall the information operators ([31], [32], [15], [16]) obtained from a model of information systems as understood in [30]. In Section 2.3 we present a knowledge operator derived from an information system ([17]) and we list some of its properties. In Section 2.4 a Stone duality ([36]) for Boolean algebras is recalled in the context of discrete duality. In Section 2.5 we introduce knowledge algebras and knowledge frames and we apply a discrete duality framework to these classes of structures. Next, in Section 2.6, we prove representation theorems for knowledge algebras and knowledge frames. The chapter is completed by concluding remarks.

2.2 Rough-Set-Style Information Operators

In many applications the available data are represented as a collection of objects together with their properties. Formally, such an information is a triple $\Sigma = (Ob, \mathcal{A}t, \{Val_a : a \in \mathcal{A}t\})$ where Ob is a set of objects, $\mathcal{A}t$ is a set of attributes and Val_a, $a \in \mathcal{A}t$, is a set of values of an attribute a. More specifically, each attribute

$a \in \mathcal{A}t$ is a mapping $a : Ob \to Val_a$ and the pair $(a, a(x))$ is viewed as a property of an object x. In a more general setting, an attribute may assign a subset of values to an object. All the considerations in this chapter are relevant for both of these cases. In the theory of rough sets originated by Pawlak ([31], [32]) structures of the form Σ are referred to as *information systems*. In the following we will use the more concise notation $(Ob, \mathcal{A}t)$ instead of $(Ob, \mathcal{A}t, \{Val_a : a \in \mathcal{A}t\})$.

Given an information system $\Sigma = (Ob, \mathcal{A}t)$, various information relations can be derived from Σ, for a comprehensive survey see [6]. In particular, the well known *indiscernibility relation*, defined for every $A \subseteq \mathcal{A}t$ by: for all $x, y \in Ob$,

$$ind_A(x,y) \overset{\text{df}}{\Leftrightarrow} a(x) = a(y) \quad \text{for every } a \in A,$$

reflects a kind of "sameness" of objects. It is clear that ind_A, for every $A \subseteq \mathcal{A}t$, is an equivalence relation and hence it determines a partition of Ob. Equivalence classes of $ind(A)$ will be written as $[x]_A, x \in Ob$. Given a set $X \subseteq Ob$, usually it can be characterized only approximately in terms of the attributes from a set $A \subseteq \mathcal{A}t$. In the rough set theory, two approximation operators are defined as follows: for every $X \subseteq Ob$,

$$L_A(X) \overset{\text{df}}{=} \{x \in Ob : \forall y \in Ob,\ ind_A(x,y) \Rightarrow y \in X\}$$
$$U_A(X) \overset{\text{df}}{=} \{x \in Ob : \exists y \in Ob,\ ind_A(x,y)\ \&\ y \in X\}.$$

For any $X \subseteq Ob$, $L_A(X)$ is a A–lower approximation of X and $U_A(X)$ is an A–upper approximation of X. A well known interpretation of these operators is that the objects in $L_A(X)$ are those which certainly belong to X, whereas objects in $U_A(X)$ can only be viewed as possible members of X in view of properties of the elements of X determined by the attributes from $A \subseteq \mathcal{A}t$. Note that L_A and U_A are modal operations of necessity and possibility ([2]), respectively, determined by ind_A. We recall that given a set Z and a binary relation R on Z, the necessity operator $[R]$ and the possibility operator $\langle R \rangle$ determined by R are defined by: for every $Y \subseteq Z$,

$$[R]Y = \{z \in Z : \forall y \in Z, zRy \Rightarrow y \in Y\}$$
$$\langle R \rangle Y = \{z \in Z : \exists y \in Z, zRy\ \&\ y \in Y\}.$$

It is known that these operators are monotone, that is for all $Y_1, Y_2 \subseteq Z$ such that $Y_1 \subseteq Y_2$, $[R]Y_1 \subseteq [R]Y_2$ and $\langle R \rangle Y_1 \subseteq \langle R \rangle Y_2$. Also, from the correspondence theory ([1]) the following facts follow:

R is reflexive	iff	$\forall Y \subseteq Z,\ [R]Y \subseteq Y$	iff	$\forall Y \subseteq Z,\ Y \subseteq \langle R \rangle Y$
R is symmetric	iff	$\forall Y \subseteq Z,\ \langle R \rangle [R] Y \subseteq Y$	iff	$\forall Y \subseteq Z,\ Y \subseteq [R] \langle R \rangle Y$.

Given an information system $(Ob, \mathcal{A}t)$ and a set $A \subseteq \mathcal{A}t$ of attributes, for a set $X \subseteq Ob$ we define the following three regions of certainty determined by A, called *positive*, *negative*, and *borderline instances of* X ([15], [16]):

- $Pos_A(X) = L_A(X)$
- $Neg_A(X) = Ob - U_A(X)$
- $B_A(X) = U_A(X) - L_A(X)$.

Given a set A of attributes, the elements of $Pos_A(X)$ definitely belong to X, the elements of $Neg_A(X)$ definitely do not belong to X, and $B_A(X)$ is the region of uncertainty: we can only say that the elements of $B_A(X)$ possibly belong to X, but we cannot decide it for certain.

In terms of the approximations, we define several types of definability and indefinability of sets. Given an information system $(Ob, \mathcal{A}t)$ and a set $A \subseteq \mathcal{A}t$ of attributes, we say that a set $X \subseteq Ob$ of objects is:

- *A–definable* iff $Pos_A(X) = X$
- *roughly A–definable* iff $Pos_A(X) \neq \emptyset$, $Neg_A(X) \neq \emptyset$, and $B_A(X) \neq \emptyset$
- *bottom A–indefinable* iff $Pos_A(X) = \emptyset$
- *top A–indefinable* iff $Neg_A(X) = \emptyset$
- *totally A–indefinable* iff $Pos_A(X) = Neg_A(X) = \emptyset$.

If X is A–definable, then we can precisely determine the properties of all its elements, where by properties we mean the properties expressed by attributes in A. However, if it is roughly A–definable, we are only able to say which properties its elements certainly possess and which ones they certainly do not possess but we may not be able to determine all their properties. Accordingly, for the bottom (resp. top) A–indefinable set X we cannot indicate properties which it certainly possesses (resp. does not possess). In the last case we are unable to say anything about properties of X.

We quote lemmas from [17] which provide a characterization of degrees of definability of sets.

Lemma 2.1. *For every information system $(Ob, \mathcal{A}t)$, for every $A \subseteq \mathcal{A}t$, and for every $X \subseteq OB$, the following statements are equivalent:*

(a) X *is A–definable*
(b) $U_A(X) = X$
(c) $B_A(X) = \emptyset$
(d) $Pos_A(X) = Ob - Neg_A(X)$
(e) $X = \bigcup_{x \in X} [x]_A$

Lemma 2.2. *For every information system $(Ob, \mathcal{A}t)$, for every $A \subseteq \mathcal{A}t$, and for every $X \subseteq OB$,*

(a) X *is roughly A–definable iff* $B_A(X) \neq \emptyset$
(b) X *is top A–indefinable iff* $B_A(X) = Ob - Pos_A(X)$ *and* $X \neq Ob$
(c) X *is totally A–indefinable iff* $B_A(X) = Ob$.

2.3 Knowledge Operator

In the literature there is an extensive discussion of characteristics and formal representation of knowledge (see, e.g., [8], [13], [35], [38]). In Artificial Intelligence

and Computer Science, logics of knowledge (epistemic logics) are usually modal logics where knowledge operator is identified with modal operator of necessity. The modal system S5 is a familiar logic to model knowledge ([7], [14], [9], [37]). This system has nice mathematical properties which often motivate researchers to adopt it in their exploration of the field. However, it models an idealized notion of knowledge. In particular, it suffers from the problem of *veridicality* (what the agent knows is true) and the problem of *logical omniscience* ([28, 29]) – the agent knows all logical consequences of his knowledge. Formally, these properties are represented by logical axioms $K_a\varphi \to \varphi$ and $K_a\varphi \wedge K_a(\varphi \to \psi) \to K_a\psi$, where $K_a\varphi$ stands for "*an agent a knows* φ".

In [17] another definition of knowledge operator is provided in the context of modelling incomplete or uncertain knowledge. The underlying intuition is the following. It is a common agreement that knowledge of an agent is manifested in his abilities to find patterns and regularities in a set of data. In the context of information systems it means that a knowledgeable agent is able to classify the objects in terms of their properties determined by the attributes.

Let $(Ob, \mathcal{A}t)$ be an information system and let $A \subseteq \mathcal{A}t$ be a set of attributes. Following [17], see also [6], we define the *knowledge operator* determined by A as follows: for every $X \subseteq Ob$,

$$K_A(X) \stackrel{\text{df}}{=} Pos_A(X) \cup Neg_A(X). \tag{2.1}$$

This operator may be seen as a rough-set style formulation of the operator "an agent knows whether" discussed in [8]. Hintikka describes its intuitive meaning as "Clearly one knows whether φ is true if and only if one knows that φ is true or knows that φ is false". In the context of information systems, knowledge of an agent whose indiscernibility relation is ind_A, for some $A \subseteq \mathcal{A}t$, is reflected by his ability of classifying the objects in the system according to their properties determined by the attributes in A. The agent knows a set $X \subseteq Ob$ whenever for every $x \in Ob$ he can ascertain that $x \in X$ or $x \notin X$. The knowledge operator defined in (2.1) has the property that if an object, say x, is such that the equivalence class $[x]_A = \{x\}$, that is the agent has a crisp information about x, then for any set $X \subseteq Ob$ we have $x \in K_A(X)$. The S5 knowledge operator does not have this property. The disjoint representation of rough sets developed and studied in [26], see also [27], is based on this property.

We say that knowledge about a set X determined by attributes from a set A is:

- *complete*, if $K_A(X) = Ob$
- *rough*, if $Pos_A(X) \neq \emptyset$, $Neg_A(X) \neq \emptyset$, and $B_A(X) \neq \emptyset$
- *pos–empty*, if $Pos_A(X) = \emptyset$
- *neg–empty*, if $Neg_A(X) = \emptyset$
- *empty*, if it is pos–empty and neg–empty.

Lemma 2.3. [17] *For every information system $(Ob, \mathcal{A}t)$, for every $A \subseteq \mathcal{A}t$, and for every $X \subseteq OB$, the following statements hold:*

(a) $K_A(X)$ *is complete iff* X *is* A*–definable*
(b) $K_A(X)$ *is rough iff* X *is roughly* A*–definable*
(c) $K_A(X)$ *is pos–empty iff* X *is bottom* A*–indefinable*
(d) $K_A(X)$ *is neg–empty iff* X *is top* A*–indefinable*
(e) $K_A(X)$ *is empty iff* X *is totally* A*–indefinable.*

The following lemma gives some facts about interaction of information operators.

Lemma 2.4. [6] *For every information system* $(Ob, \mathcal{A}t)$*, for all* $A, B \subseteq \mathcal{A}t$*, and for every* $X \subseteq OB$*, the following hold:*

(a) $Pos_A(K_A(X)) = K_A(X)$ *and* $Neg_A(K_A(X)) = B_A(X)$
(b) $B_A(K_A(X)) = \emptyset$
(c) $K_A(Pos_A(X)) = K_A(Neg_A(X)) = Ob$
(d) $K_A(K_A(X)) = Ob$
(e) $K_A(X) = K_A(Ob - X)$
(f) $K_A(\emptyset) = Ob$
(g) *if* $A \subseteq B$*, then* $K_A(X) \subseteq K_B(X)$.

The knowledge operator K_A defined by (2.1) does not have the undesired property of veridicality typical for traditional epistemic logic based on S5 modal system. To see that $K_A(X)$ is not necessarily included in X consider the following example.

Example 2.1. Let $Ob = \{o_1, \ldots, o_5\}$ and let $X = \{o_1, o_2, o_3\}$. Assume that an indiscernibility relation determined by a set A of attributes generates the following partition: $\{\{o_1, o_2\}, \{o_3, o_4\}, \{o_5\}\}$. Then we have: $K_A(X) = Pos_A(X) \cup Neg_A(X) = \{o_1, o_2\} \cup \{o_5\}$ which obviously is not included in X.

Similarly, the knowledge operator K_A does not have the unwanted property of logical omniscience as the following example shows.

Example 2.2. Let $Ob = \{o_1, o_2, o_3, o_4\}$ and let an indiscernibility relation determined by a set A of attributes generates the partition $\{\{o_1, o_2\}, \{o_3, o_4\}\}$. Take $X = \{o_1\}$ and $Y = \{o_2, o_3\}$. Then

$$\begin{aligned} K_A(X) &= Pos_A(X) \cup Neg_A(X) = \emptyset \cup \{o_3, o_4\} = \{o_3, o_4\} \\ K_A(-X \cup Y) &= Pos_A(-X \cup Y) \cup Neg_A(-X \cup Y) = \{o_3, o_4\} \cup \emptyset = \{o_3, o_4\} \\ K_A(Y) &= Pos_A(Y) \cup Neg_A(Y) = \emptyset. \end{aligned}$$

Hence $K_A(X) \cap K_A(-X \cup Y) = \{o_3, o_4\}$. Clearly, it is not included in $K_A(Y)$.

However, the agents reasoning according to these knowledge operators are fully introspective: they know what they know (positive introspection) and they know what they do not know (negative introspection). Formally, positive introspection is reflected by $K_A(X) \subseteq K_A(K_A(X))$ and negative introspection by $-K_A(X) \subseteq K_A(-K_A(X))$ for every $X \subseteq Ob$.

2.4 Discrete Duality for Boolean Algebras

In this section we recall Stone duality for Boolean algebras ([36]).

Let $(B,\vee,\wedge,-,0,1)$ be a Boolean algebra where the operations of join ($\vee$), meet ($\wedge$) and complement ($-$) satisfy the following axioms:

$a\vee(b\vee c)=(a\vee b)\vee c$	$a\wedge(b\wedge c)=(a\wedge b)\wedge c$	associativity
$a\vee b=b\vee a$	$a\wedge b=b\wedge a$	commutativity
$a\vee(a\wedge b)=a$	$a\wedge(a\vee b)=a$	absorption
$a\vee(b\wedge c)=(a\vee b)\wedge(a\vee c)$	$a\wedge(b\vee c)=(a\wedge b)\vee(a\wedge c)$	distributivity
$a\vee -a=1$	$a\wedge -a=0$	complement.

For simplicity, we will write B to denote both a Boolean algebra and its underlying universe.

The natural ordering on a Boolean algebra B is defined by: for all $a,b\in B$, $a\leq b$ iff $a\vee b=b$ (or equivalently, $a\wedge b=a$).

A *filter* of a Boolean algebra B is a subset $F\subseteq B$ such that for all $a,b\in B$,

- if $a\in F$ and $a\leq b$, then $b\in F$
- if $a,b\in F$, then $a\wedge b\in F$.

These two conditions are equivalent to $a,b\in F$ if and only if $a\wedge b\in F$.

A filter F is said to be:

- a proper filter iff $F\neq B$ (equivalently, $0\notin F$)
- a prime filter iff $a\vee b\in F$ implies $a\in F$ or $b\in F$ (equivalently, for any $a\in B$, either $a\in F$ or $-a\in F$)
- an ultrafilter (maximal filter) iff it is a maximal element with respect to set inclusion in the family of all proper filters
- a filter generated by a set $A\subseteq B$ iff $F=\{a\in B:\exists n\geq 1,\ \exists a_1,\ldots,a_n\in A,$ $a_1\wedge\ldots\wedge a_n\leq a\}$
- a principal filter iff $F=\{b\in B:a\leq b\}$ for some $a\in B$.

The following facts are well known (see, for example, [12], [34]).

Theorem 2.1. *In every Boolean algebra the following are satisfied:*

(a) *A filter is an ultrafilter iff it is prime and proper.*
(b) *For every proper filter there is an ultrafilter including it.*
(c) *If $a\not\leq b$, then there is an ultrafilter F such that $a\in F$ and $b\notin F$.*

A *Boolean frame* is any non-empty set X. The *complex algebra of a Boolean frame* X is the powerset Boolean algebra $\mathfrak{Cm}(X)=(2^X,\cup,\cap,-,\emptyset,X)$. Clearly, the complex algebra of a Boolean frame is a Boolean algebra.

Given a Boolean algebra B, its *canonical frame* $\mathfrak{Cf}(B)$ is the set $Ult(B)$ of all ultrafilters of B.

The Stone mapping $h : B \to \mathfrak{Cm}(\mathfrak{Cf}(B))$ is defined by: for every $a \in B$,

$$h(a) \stackrel{\text{df}}{=} \{F \in Ult(B) : a \in F\}.$$

Since $\mathfrak{Cm}(\mathfrak{Cf}(B)) = 2^{Ult(B)}$, h is well defined.

Lemma 2.5. *The mapping h is an embedding.*

Proof. First, we prove that the mapping h preserves the operations $\vee$ and $\wedge$. We show that $h(a \vee b) = h(a) \cup h(b)$. Let $F \in h(a \vee b)$. Then $a \vee b \in F$. Since F is a prime filter, $a \in F$ or $b \in F$. Thus $F \in h(a) \cup h(b)$. Conversely, let $a \in F$ or $b \in F$. Since $a \leq a \vee b$ and $b \leq a \vee b$, in both cases $a \vee b \in F$. The preservation of meet can be proved in the similar way.

Since F is an ultrafilter, $-a \in F$ iff $a \notin F$, hence $h(-a) = -h(a)$.

Clearly, $h(0) = \emptyset$ and $h(1) = Ult(B)$.

Second, we show that h is injective. Let $\leq$ be the natural ordering on B. Take $a, b \in B$ such that $a \neq b$. Then, by antisymmetry of $\leq$, either $a \not\leq b$ or $b \not\leq a$. Without loss of generality, assume that $a \not\leq b$. Then by Theorem 2.1(c) there is an ultrafilter F such that $a \in F$ and $b \notin F$. Hence $h(a) \neq h(b)$.

Now, let X be a Boolean frame. Consider a mapping $k : X \to \mathfrak{Cf}(\mathfrak{Cm}(X))$ defined as: for every $x \in X$,

$$k(x) \stackrel{\text{df}}{=} \{A \in 2^X : x \in A\}.$$

Since $\mathfrak{Cf}(\mathfrak{Cm}(X)) = Ult(2^X)$ and $k(x)$ is a principal filter generated by $\{x\}$, k is well defined.

Lemma 2.6. *The mapping k is an embedding.*

Proof. To show that k is one-to-one, take $x, y \in X$ such that $k(x) = k(y)$. This means that for every $A \subseteq X$, $A \in k(x)$ iff $A \in k(y)$, that is $x \in A$ iff $y \in A$. In particular, since $x \in \{x\}$ we have $y \in \{x\}$ which yields $y = x$, as required.

By Lemma 2.5 and Lemma 2.6 we get the following discrete duality between Boolean algebras and Boolean frames.

Theorem 2.2.

(a) *Every Boolean algebra is embeddable into the complex algebra of its canonical frame.*
(b) *Every Boolean frame is embeddable into the canonical frame of its complex algebra.*

2.5 Knowledge Algebras and Knowledge Frames

In this section we introduce knowledge algebras as Boolean algebras with an operator K which captures in an abstract way the features of the knowledge operator

discussed in Section 2.3. Next, we present a discrete duality for the class of knowledge algebras.

A *knowledge algebra* (*K-algebra*) is a structure $\mathcal{K}=(B,\vee,\wedge,-,0,1,K)$ where $(B,\vee,\wedge,-,0,1)$ is a Boolean algebra and K is a unary operator on B satisfying the following axioms:

(K1) $a\wedge Ka\wedge K(-a\vee b)\leq Kb$
(K2) $K(a\vee -Ka)=1$
(K3) $Ka=K(-a)$.

A logic with the operator K is presented in [4], see also [6], and the above axioms are algebraic counterparts of the axioms of this logic. An extension of this logic to a logic with the family of knowledge operators each of which is explicitly indexed with a set of agents is presented in [5].

Operator K is not monotone. For example, for all $a,b\in B$ we have $a\wedge b\leq a$. Furthermore, since $K(1)=1$ by (K2), we have $K(0)=1$ by (K3). But it is not true that for every $a\in B$, $1=K(a\wedge 0)\leq K(a)$. However, for all $a,b\in B$, if $a\leq b$, then $a\wedge K(a)\leq K(b)$. Indeed, assume that $a\leq b$. Then $-a\vee b=1$ and since $K(1)=1$, we obtain the required condition by (K1). Similarly, K is not antitone. For example, for every $a\in B$ we have $a\leq 1$ but not necessarily $1=K(1)\leq K(a)$.

By a filter of an K-algebra we mean a filter of its Boolean reduct.

A *knowledge frame* (*K-frame*) is a structure $\mathcal{X}=(X,R)$ where X is a Boolean frame and R is an equivalence relation on X.

The *complex algebra of a K-frame* is a structure

$$\mathfrak{Cm}(\mathcal{X})=(2^X,\cup,\cap,-,\emptyset,X,K_R)$$

where $(2^X,\cup,\cap,-,\emptyset,X)$ is the complex algebra of a Boolean frame X and for every $A\subseteq X$,

$$K_R(A)\stackrel{\text{df}}{=}[R]A\cup[R](-A).$$

Theorem 2.3. *The complex algebra of a K-frame is a K-algebra.*

Proof.
(K1) We have to show that for all $A,B\subseteq X$,

$$A\cap K_R(A)\cap K_R(-A\cup B)\subseteq K_R(B).$$

Suppose otherwise, that is there exist $A,B\subseteq X$ and there is some $x\in X$ such that $x\in A$, $x\in[R]A\cup[R](-A)$, $x\in[R](-A\cup B)\cup[R](A\cap -B)$, $x\notin[R]B$, and $x\notin[R](-B)$. If $x\in[R](-A)$, then by reflexivity of R, $x\notin A$, which contradicts the assumption that $x\in A$. Then $x\in[R]A$. Since $x\notin[R]B$ and $x\notin[R](-B)$, there exist $y,z\in X$ such that xRy, xRz, $y\notin B$, and $z\in B$. Now, xRy and $x\in[R]A$ imply $y\in A$. Assume that

$x \in [R](-A \cup B)$. Since xRy, we get $y \in -A \cup B$, and since $y \in A$, $y \in B$, a contradiction. Then $x \in [R](A \cap -B)$. Since xRz, we get $z \in A \cap -B$, thus $z \notin B$, which is again a contradiction.

(K2) We have to show that $K_R(-K_R(A) \cup A) = X$. Suppose that there is some $x \in X$ such that $x \notin K_R(-K_R(A) \cup A)$. This means that

$$x \notin [R](-K_R(A) \cup A) \cup [R](-(-K_R(A) \cup A)),$$

so (i) $x \notin [R](-K_R(A) \cup A)$ and (ii) $x \notin [R](-(-K_R(A) \cup A))$. By (i), there exists some $y \in X$ such that xRy and $y \notin -K_R(A) \cup A$, so $y \notin A$ and $y \in K_R(A)$, i.e., $y \in [R]A \cup [R](-A)$. By reflexivity of R, $[R]A \subseteq A$, so since $y \notin A$, we get $y \notin [R]A$. Hence, since $y \in [R]A \cup [R](-A)$, we get $y \in [R](-A)$. Also, (ii) means that there is some $z \in X$ such that xRz and $z \in -K_R(A) \cup A$. By symmetry and transitivity of R, from xRy and xRz it follows that yRz which together with $y \in [R](-A)$ yields $z \notin A$. Since $z \in -K_R(A)$, we get $z \notin K_R(A)$, that is $z \notin [R]A \cup [R](-A)$ which implies $z \notin [R](-A)$. Hence there is some $u \in X$ such that zRu and $u \in A$. Again by transitivity of R, from yRz and zRu we get yRu which together with $y \in [R](-A)$ gives $u \notin A$, a contradiction.

(K3) Obvious.

The *canonical frame of a K-algebra* is a structure $\mathfrak{Cf}(\mathcal{K}) = (Ult(B), R_K)$ where $Ult(B)$ is the set of all ultrafilters of B and R_K is a binary relation on $Ult(B)$ such that for all $F, G \in Ult(B)$,

$$F R_K G \overset{\text{df}}{\Leftrightarrow} K(F) \subseteq G$$

where $K(F) \overset{\text{df}}{=} \{a \in B : a, Ka \in F\}$.

Theorem 2.4. *The canonical frame of a K-algebra is a K-frame.*

Proof. We have to show that R_K is an equivalence relation.
Reflexivity of R_K is obvious. For symmetry, take $F, G \in Ult(B)$ and assume that $K(F) \subseteq G$. We show that for every $a \in B$, $a \notin F$ implies $a \notin G$ or $Ka \notin G$. Take $a \in B$ such that $a \notin F$. Then $-a \in F$. Since $-a \leq -a \vee -Ka$, we get $-a \vee -Ka \in F$. By axioms (K2) and (K3) we also have $1 = K(-a \vee -K(-a)) = K(-a \vee -Ka) \in F$. Hence $-a \vee -Ka \in K(F)$. By the assumption, $-a \vee -Ka \in G$. Since G is prime, $-a \in G$ or $-Ka \in G$, and hence $a \notin G$ or $Ka \notin G$.

Now we show that R_K is transitive. Let $F, G, H \in Ult(B)$ be such that $K(F) \subseteq G$ and $K(G) \subseteq H$. Take $a \in K(F)$, that is $a \in F$ and $Ka \in F$. Then $a \wedge Ka \in F$ and $a \in G$. By axioms (K2) and (K3) we get $1 = K(-a \vee -K(-a)) = K(-(a \wedge K(-a)))$ $= K(a \wedge Ka) \in F$. Since $a \wedge Ka \in F$, $a \wedge Ka \in K(F)$. By the assumption, $a \wedge Ka \in G$. Since $a \wedge Ka \leq Ka$, $Ka \in G$. We also have $a \in G$, so $a \in K(G)$. By the assumption $a \in H$, as required.

2.6 Representation Theorems for Knowledge Algebras and Knowledge Frames

Now we prove a representation theorem for knowledge algebras.

Theorem 2.5. *Every knowledge algebra is embeddable into the complex algebra of its canonical frame.*

Proof. The embedding is provided by the Stone mapping h defined in Section 2.4. By Lemma 2.5, h is an embedding of Boolean algebras. It suffices to show that h preserves the operator K. That is, we have to show that $h(Ka) = K_{R_K}(h(a))$. Observe that

$$\begin{aligned} F \in K_{R_K}(h(a)) &\Leftrightarrow F \in [R_K]h(a) \cup [R_K](-h(a)) \\ &\Leftrightarrow (\forall G \in Ult(B), FR_KG \Rightarrow a \in G) \\ &\quad \text{or } (\forall G \in Ult(B), FR_KG \Rightarrow a \notin G). \end{aligned}$$

($\Rightarrow$) Assume $Ka \in F$ and take G such that FR_KG. Then, for every $b \in B$, if $b, Kb \in F$, then $b \in G$. If $a \notin F$, then $-a \in F$. Since $Ka = K(-a)$ by (K3), $K(-a) \in F$. Hence $-a \in K(F)$. Thus by the assumption $-a \in G$, whence $a \notin G$. If $a \in F$, then since $Ka \in F$, $a \in K(F)$, whence $a \in G$.

($\Leftarrow$) Assume $Ka \notin F$. We have to show that there exists some $G_1 \in Ult(B)$ such that $K(F) \subseteq G_1$ and $a \notin G_1$ and there exists some $G_2 \in Ult(B)$ such that $K(F) \subseteq G_2$ and $a \in G_2$. Consider a filter G_1' generated by $K(F) \cup \{a\}$. It is a proper filter. Indeed, if we suppose otherwise, then there is some $b \in K(F)$ such that $b \wedge a = 0$. Hence $a \leq -b$, so $-b \vee a = -b$. Since $b \in K(F)$, we have $b \in F$ and $Kb \in F$. Hence, by (K3), $K(-b) \in F$, thus $K(-b \vee a) \in F$. Therefore, $b \wedge Kb \wedge K(-b \vee a) \in F$. Now, by (K1), $Ka \in F$ which contradicts the assumption.

Let G_1 be an ultrafilter containing G_1'. Thus $K(F) \subseteq G_1$ and $a \in G_1$.

Similarly, let G_2' be a filter generated by $K(F) \cup \{-a\}$. It is easy to see that it is a proper filter, so by Theorem 2.1(b) it can be extended to an ultrafilter including it, say G_2. Then we have $K(F) \subseteq G_2$ and $a \notin G_2$.

In the following we prove a representation theorem for knowledge frames.

Theorem 2.6. *Every knowledge frame is embeddable into the canonical frame of its complex algebra.*

Proof. The embedding is provided by the mapping k defined in Section 2.4. By Lemma 2.6, k is an embedding of Boolean frames. It suffices to show that k preserves the relation R. We show that $xRy \Leftrightarrow k(x)R_{K_R}k(y)$.
Observe that

$$\begin{aligned} k(x)R_{K_R}k(y) &\Leftrightarrow K_R(k(x)) \subseteq k(y) \\ &\Leftrightarrow \forall A \subseteq X, A \in k(x) \ \& \ K_R(A) \in k(x) \Rightarrow A \in k(y) \\ &\Leftrightarrow \forall A \subseteq X, \ x \in A \ \& \ x \in [R]A \cup [R](-A) \Rightarrow y \in A. \end{aligned}$$

($\Rightarrow$) Assume xRy, $x \in A$, and (i) $\forall z$, $xRz \Rightarrow z \in A$ or (ii) $\forall z$, $xRz \Rightarrow z \notin A$. Suppose $y \notin A$. Then taking y for z in (i) and x for z in (ii) we get a contradiction.

($\Leftarrow$) Assume that for every $A \subseteq X$, $A \in k(x)$ and $K_R(A) \in k(x)$ imply $A \in k(y)$. Consider $A = \langle R \rangle \{x\}$. Then we have $x \in \{x\} \subseteq \langle R \rangle \{x\}$. Since R is symmetric, $\{x\} \subseteq [R]\langle R \rangle \{x\}$, so $\langle R \rangle \{x\} \in k(x)$ and $K_R(\langle R \rangle \{x\}) \subseteq k(x)$. Then $\langle R \rangle \{x\} \in k(y)$ which gives yRx and by symmetry of R we finally obtain xRy.

2.7 Conclusions

In this chapter we have applied a discrete duality framework to a class of knowledge algebras and knowledge frames. This contributes to our project of developing discrete dualities for algebras and frames relevant to the three major theories of approximate reasoning, namely fuzzy logics, formal concept analysis, and rough-set-style information logics. Discrete duality for fuzzy logics has been developed, among others, for the logic MTL (see [24] and [20]) and for a monoidal fuzzy logic presented in [10] and some of its axiomatic extensions ([18], [19]). In the field of the formal concept analysis discrete duality has been developed for the class of context algebras ([23]). Discrete duality for rough-set-style information logics is discussed in [25] and [33].

References

1. van Benthem, J.: Correspondence theory. In: Gabbay, D., Guenthner, F. (eds.) Handbook of Pholosophical Logic, vol. 2, pp. 167–247. D. Reidel, Dordrecht (1984)
2. Chellas, B.F.: Modal Logic: An Introduction. Cambridge University Press, Cambridge (1980)
3. Davey, B.A., Priestley, H.: Introduction to Lattices and Order. Cambridge University Press, Cambridge (1990)
4. Demri, S.: A completeness proof for a logic with an alternative necessity operator. Studia Logica 58, 99–112 (1997)
5. Demri, S.: A logic with relative knowledge operators. Journal of Logic, Language and Information 8, 167–185 (1999)
6. Demri, S., Orłowska, E.: Incomplete Information: Structure, Inference, Complexity. EATCS Monographs in Teoretical Computer Science. Springer, Heidelberg (2002)
7. Fagin, R., Halpern, J.Y., Moses, Y., Vardi, M.Y.: Reasoning about Knowledge. MIT Press, Cambridge (1995)
8. Hintikka, J.: Knowledge and Belief. Cornell University Press, London (1962)
9. van der Hoek, W., Meyer, J.J.: A complete epistemic logic for multiple agents. In: Bacharach, M., Gérard-Varet, L.A., Mongin, P., Shin, H. (eds.) Epistemic Logic and the Theory of Games and Decisions, pp. 35–68. Kluwer Academic Publishers, Dordrecht (1999)
10. Höhle, U.: Commutative, residuated l–monoids. In: Höhle, U., Klement, U.P. (eds.) Non–Classical Logics and their Applications to Fuzzy Subsets, pp. 53–106. Kluwer Academic Publishers, Dordrecht (1996)

11. Jónsson, B., Tarski, A.: Boolean algebras with operators. Part I. American Journal of Mathematics 73, 891–939 (1951)
12. Koppelberg, S.: General Theory of Boolean algebras. North Holland, Amsteram (1989)
13. Lenzen, W.: Recent work in epistemic logic. Acta Philosophica Fennica 30, 1–129 (1978)
14. Meyer, J.J.C., van der Hoek, W.: Epistemic Logic for AI and Computer Science. Cambridge Tracks in Theoretical Computer Science, vol. 41. Cambridge University Press, Cambridge (1995)
15. Orłowska, E.: Representation of vague information. ICS PAS Report 503 (1983)
16. Orłowska, E.: Semantics of vague concepts. In: Dorn, G., Weingartner, P. (eds.) Foundations of Logic and Linguistics. Problems and Their Solutions. Selected Contributions to the 70th Intenational Congress of Logic Methodology and Philosophy of Science, Salzburg, 1983, pp. 465–482. Plenum Press, New York and London (1985)
17. Orłowska, E.: Logic for reasoning about knowledge. Zeitschrift für Mathematische Logik und Grundlagen der Mathematik 35, 556–568 (1989)
18. Orłowska, E., Radzikowska, A.M.: Relational Representability for Algebras of Substructural Logics. In: MacCaull, W., Winter, M., Düntsch, I. (eds.) RelMiCS 2005. LNCS, vol. 3929, pp. 212–226. Springer, Heidelberg (2006)
19. Orłowska, E., Radzikowska, A.M.: Representation theorems for some fuzzy logics based on residuated non–distributive lattices. Fuzzy Sets and Systems 159, 1247–1259 (2008)
20. Orłowska, E., Radzikowska, A.M.: Discrete duality for some axiomatic extensions of MTL algebras. In: Cintula, P., Hanikowá, Z., Svejdar, V. (eds.) Witnessed Years. Essays in Honour of Petr Hájek, pp. 329–344. College Publications, King's College London (2010)
21. Orłowska, E., Rewitzky, I.: Duality via truth: Semantic frameworks for lattice–based logics. Logic Journal of the IGPL 13, 467–490 (2005)
22. Orłowska, E., Rewitzky, I.: Discrete Duality and Its Applications to Reasoning with Incomplete Information. In: Kryszkiewicz, M., Peters, J.F., Rybiński, H., Skowron, A. (eds.) RSEISP 2007. LNCS (LNAI), vol. 4585, pp. 51–56. Springer, Heidelberg (2007)
23. Orłowska, E., Rewitzky, I.: Context Algebras, Context Frames, and Their Discrete Duality. In: Peters, J.F., Skowron, A., Rybiński, H. (eds.) Transactions on Rough Sets IX. LNCS, vol. 5390, pp. 212–229. Springer, Heidelberg (2008)
24. Orłowska, E., Rewitzky, I.: Algebras for Galois–style connections and their discrete duality. Fuzzy Sets and Systems 161(9), 1325–1342 (2010)
25. Orłowska, E., Rewitzky, I., Düntsch, I.: Relational Semantics Through Duality. In: MacCaull, W., Winter, M., Düntsch, I. (eds.) RelMiCS 2005. LNCS, vol. 3929, pp. 17–32. Springer, Heidelberg (2006)
26. Pagliani, P.: Rough sets and Nelson algebras. Fundamenta Informaticae 27(2–3), 205–219 (1996)
27. Pagliani, P., Chakraborty, M.: A Geometry of Approximation. Trends in Logic, vol. 27. Springer (2008)
28. Parikh, R.: Knowledge and the problem of logical omniscience. In: Raś, W.Z., Zemankova, M. (eds.) Proceedings of the Second International Symposium on Methodologies for Intelligent Systems (ISMIS 1987), Charlotte, North Carolina, USA, October 14-17, pp. 432–439. North-Holland/Elsevier, Amsterdam (1987)
29. Parikh, R.: Logical omniscience. In: Leivant, D. (ed.) LCC 1994. LNCS, vol. 960, pp. 22–29. Springer, Heidelberg (1995)
30. Pawlak, Z.: Information systems – theoretical foundations. Information Systems 6, 205–218 (1981)
31. Pawlak, Z.: Rough sets. International Journal of Computer and Information Sciences 11(5), 341–356 (1982)
32. Pawlak, Z.: Rough sets - Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Dordrecht (1991)

33. Radzikowska, A.M.: Discrete dualities for some information algebras based on De Morgan lattices (2011) (submitted)
34. Rasiowa, H., Sikorski, R.: Mathematics of Metamathematics. PWN, Warsaw (1963)
35. Read, S.: Thinking about Logic. Oxford University Press, Oxford (1994)
36. Stone, M.: The theory of representations of Boolean algebras. Transactions of the American Mathematical Society 40(1), 37–111 (1936)
37. Wansing, H.: A general possible worlds framework for reasoning about knowledge and belief. Studia Logica 49(4), 523–539 (1990)
38. von Wright, G.H.: An Essay in Modal Logic. North-Holland Publishing Company, Amsterdam (1952)

Chapter 3
Comparison of Greedy Algorithms for Decision Tree Optimization

Abdulaziz Alkhalid, Igor Chikalov, and Mikhail Moshkov

Abstract. This chapter is devoted to the study of 16 types of greedy algorithms for decision tree construction. The dynamic programming approach is used for construction of optimal decision trees. Optimization is performed relative to minimal values of average depth, depth, number of nodes, number of terminal nodes, and number of nonterminal nodes of decision trees. We compare average depth, depth, number of nodes, number of terminal nodes and number of nonterminal nodes of constructed trees with minimum values of the considered parameters obtained based on a dynamic programming approach. We report experiments performed on data sets from UCI ML Repository and randomly generated binary decision tables. As a result, for depth, average depth, and number of nodes we propose a number of good heuristics.

Keywords: Decision tree, greedy algorithms, dynamic programming

3.1 Introduction

Decision trees are widely used as a way for knowledge representation, as predictors and as algorithms for problem solving in rough set theory [25, 29], machine learning, and data mining [9], test theory [12], etc. To have more understandable decision trees we need to minimize the number of nodes in a tree. To have faster decision trees we need to minimize the depth or average depth of a tree. Unfortunately, the most problems of decision tree optimization are NP-hard [17, 23].

Several exact algorithms of decision tree optimization are known including brute force algorithms [27], algorithms based on dynamic programming [15, 22, 28], and algorithms using branch-and-bound technique [7].

Abdulaziz Alkhalid · Igor Chikalov · Mikhail Moshkov
Mathematical and Computer Sciences & Engineering Division
King Abdullah University of Science and Technology
Thuwal 23955-6990, Saudi Arabia
e-mail: {abdulaziz.alkhalid,igor.chikalov}@kaust.edu.sa,
mikhail.moshkov@kaust.edu.sa

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 21–40.
springerlink.com

There are different approaches to create approximate algorithm for decision tree optimization: genetic [10], simulated annealing [16], ant colony [8], and greedy.

The majority of approximate algorithms for decision tree optimization are based on greedy approach. These algorithms build tree in a top-down fashion and at each step, minimizing some impurity criteria designed using theoretical-information [26], statistical [9], and combinatorial [21, 23] reasoning. There are papers devoted to the comparison of different impurity criteria [3–5, 14, 18–20, 26].

In this chapter, we consider a comparative analysis of 16 greedy algorithms for decision tree optimization. These algorithms can construct both exact and approximate decision trees. We assume that the decision tables contain only categorical attributes and are free of inconsistency. Several cost functions are studied that characterize space and time complexity of decision trees: depth, average depth, number of nodes, number of terminal nodes, and number of nonterminal nodes.

We make experiments with data sets from UCI Machine Learning Repository [6] and randomly generated binary data sets. We compare the costs of trees constructed by the greedy algorithms with the cost of optimal trees constructed by a dynamic programming algorithm [1, 13]. As a result, for the three cost functions (average depth, depth and number of nodes) we found the 3-4 greedy heuristics that work efficiently.

We propose algorithms for decision tree optimization based on dynamic programming. The idea is close to algorithms described in [15, 22], but authors devised it independently and made several extensions: the algorithms are capable of finding a set of optimal trees, performing sequential optimization by different criteria, and finding relationships between two criteria [1, 2, 11, 13, 24].

The chapter is organized as follows. Section 3.2 introduces basic notions. Section 3.3 contains general schema of greedy algorithm. Section 3.4 describes an algorithm based on dynamic programming. Sections 3.5–3.7 present results of experiments. Section 3.8 is devoted to the analysis of these results. Section 3.9 contains conclusions.

3.2 Basic Notions

In this section, we consider notions of decision table and α-decision tree for a table, and describe some uncertainty measures, impurity functions, and cost functions.

3.2.1 Decision Tables and Trees

In this chapter, we consider only decision tables with categorical attributes.

These tables do not contain missing values and equal rows. Consider a *decision table* T depicted in Figure 3.1.

Here $f_1, \ldots, f_m$ are names of columns (conditional attributes); $c_1, \ldots, c_N$ are nonnegative integers which can be interpreted as decisions (values of the decision

f_1	...	f_m	d
δ_{11}	...	δ_{1m}	c_1
	...		...
δ_{N1}	...	δ_{Nm}	c_N

Fig. 3.1 Decision table

attribute d); δ_{ij} are nonnegative integers which are interpreted as values of conditional attributes (we assume that the rows $(\delta_{11},\dots,\delta_{1m}),\dots,(\delta_{N1},\dots,\delta_{Nm})$ are pairwise different). We denote by $N(T)$ the number of rows in the decision table T.

Let $f_{i_1},\dots,f_{i_t} \in \{f_1,\dots,f_m\}$ and $a_1,\dots,a_t$ be nonnegative integers. Denote by $T(f_{i_1},a_1)\dots(f_{i_t},a_t)$ the subtable of the table T, which consists of such and only such rows of T that at the intersection with columns $f_{i_1},\dots,f_{i_t}$ have numbers $a_1,\dots,a_t$, respectively. Such nonempty tables (including the table T) will be called as *separable subtables* of the table T. For a subtable Θ of the table T we will denote by $R(\Theta)$ the number of unordered pairs of rows that are labeled with different decisions. Later we will interpret the value $R(\Theta)$ as *uncertainty* of the table Θ.

Let Θ be a nonempty subtable of T. A minimal decision value which is attached to the maximal number of rows in Θ will be called as *the most common decision* for Θ.

A *decision tree* Γ *over* the table T is a finite directed tree with root in which each terminal node is labeled with a decision. Each nonterminal node is labeled with a conditional attribute and for each nonterminal node the outgoing edges are labeled with pairwise different nonnegative integers.

Let v be an arbitrary node of the considered decision tree Γ. Let us define a subtable $T(v)$ of the table T. If v is the root, then $T(v) = T$. Let v be different from the root. In the path from the root to v, nodes be labeled with attributes $f_{i_1},\dots,f_{i_t}$, and edges be labeled with numbers $a_1,\dots,a_t$, respectively. Then $T(v) = T(f_{i_1},a_1),\dots,(f_{i_t},a_t)$.

We denote by $E(T)$ the set of attributes (columns of the table T), each contains different numbers. For $f_i \in E(T)$, let $E(T,f_i)$ be the set of numbers from the column f_i.

Let Γ be a decision tree over T and α be a real number such that $0 \le \alpha < 1$. We will say that Γ is an α*-decision tree for* T if any node v of Γ satisfies the following conditions:

- If $R(T(v)) \le \alpha R(T)$ then v is a terminal node labeled with the most common decision for $T(v)$.
- Otherwise, v is labeled with an attribute $f_i \in E(T(v))$. If $E(T(v),f_i) = \{a_1,\dots,a_t\}$, then t edges leave the node v, and these edges are labeled with $a_1,\dots,a_t$, respectively.

If $\alpha = 0$, then we have the notion of exact decision tree (0-decision tree) for T.

3.2.2 Uncertainty Measures

Let rows of a decision table T be labeled with k different decisions $d_1,\dots,d_k$. For $i=1,\dots,k$, let N_i be the number of rows in T labeled with the decision d_i, and $p_i = N_i/N$.

We consider four uncertainty measures for decision tables: entropy $ent(T) = -\sum_{i=1}^{k} p_i \log_2 p_i$ (we assume $0\log_2 0 = 0$), Gini index $gini(T) = 1-\sum_{i=1}^{k} p_i^2$, minimum misclassification error $me(T) = N - \max_{1\le j\le k} N_j$, and the number $R(T)$ of unordered pairs of rows in T with different decisions (note that $R(T) = N^2 gini(T)/2$).

3.2.3 Impurity Functions

Let $f_i \in E(T)$ and $E(T,f_i) = \{a_1,\dots,a_t\}$. The attribute f_i divides the table T into subtables $T_1 = T(f_i,a_1),\dots,T_t = T(f_i,a_t)$. We now define an *impurity function* I which gives us the *impurity* $I(T,f_i)$ of this partition. Let us fix an uncertainty measure U from the set $\{ent, gini, me, R\}$ and type of impurity function: *sum*, *max*, *w_sum*, or *w_max*. Then:

- for the type *sum*, $I(T,f_i) = \sum_{j=1}^{t} U(T_j)$;
- for the type *max*, $I(T,f_i) = \max_{1\le j\le t} U(T_j)$;
- for the type *w_sum*, $I(T,f_i) = \sum_{j=1}^{t} U(T_j)N(T_j)/N(T)$;
- for the type *w_max*, $I(T,f_i) = \max_{1\le j\le t} U(T_j)N(T_j)/N(T)$.

As a result, we have 16 different impurity functions.

3.2.4 Cost Functions

We consider cost functions which are given in the following way: values of the considered cost function ψ, which are nonnegative numbers are defined by induction on pairs (T,Γ), where T is a decision table, and Γ is an α-decision tree for T for some α. Let Γ be an α-decision tree that contains only one node labeled with a decision. Then $\psi(T,\Gamma) = \psi^0$ where ψ^0 is a nonnegative number. Let Γ be an α-decision tree in which the root is labeled with an attribute f_i and t edges start in the root. These edges are labeled with numbers $a_1,\dots,a_t$ and enter roots of decision trees $\Gamma_1,\dots,\Gamma_t$. Then

$$\psi(T,\Gamma) = F(N(T),\psi(T(f_i,a_1),\Gamma_1),\dots,\psi(T(f_i,a_t),\Gamma_t)).$$

Here $F(n,\psi_1,\psi_2,\dots)$ is an operator which transforms the considered tuple of nonnegative numbers into a nonnegative number. Note that the number of variables $\psi_1,\psi_2,\dots$ is not bounded from above. Also note that for $j=1,\dots,t$, Γ_j is an

α_j- decision tree for $T(f_i,a_j)$ where $\alpha_j = \alpha R(T)/R(T(f_i,a_j))$ if $R(F(f_i,a_j)) > 0$ and $\alpha_j = \alpha$ otherwise.

The considered cost function will be called *monotone* if for any natural t, from inequalities $c_1 \le d_1, \ldots, c_t \le d_t$ the inequality $F(a,c_1,\ldots,c_t) \le F(a,d_1,\ldots,d_t)$ follows. Now we take a closer view of some monotone cost functions.

Number of nodes: $\psi(T,\Gamma)$ is the number of nodes in α-decision tree Γ. For this cost function, $\psi^0 = 1$ and $F(n,\psi_1,\psi_2,\ldots,\psi_t) = 1 + \sum_{i=1}^{t} \psi_i$.

Depth: $\psi(T,\Gamma)$ is the maximum length of a path from the root to a terminal node of Γ. For this cost function, $\psi^0 = 0$ and $F(n,\psi_1,\psi_2,\ldots,\psi_t) = 1 + \max\{\psi_1,\ldots,\psi_t\}$.

Total path length: for an arbitrary row $\bar{\delta}$ of the table T, we denote by $l(\bar{\delta})$ the length of the path from the root to a terminal node v of Γ such that $\bar{\delta}$ belongs to $T(v)$. Then $\psi(T,\Gamma) = \sum_{\bar{\delta}} l(\bar{\delta})$, where we take the sum over all rows $\bar{\delta}$ of the table T. For this cost function, $\psi^0 = 0$ and $F(n,\psi_1,\psi_2,\ldots,\psi_t) = n + \sum_{i=1}^{t} \psi_i$. Note that the *average depth* of Γ is equal to the total path length divided by $N(T)$.

Number of nonterminal nodes: $\psi(T,\Gamma)$ is the number of nonterminal nodes in α-decision tree Γ. For this cost function, $\psi^0 = 0$ and $F(n,\psi_1,\psi_2,\ldots,\psi_t) = 1 + \sum_{i=1}^{t} \psi_i$.

Number of terminal nodes: $\psi(T,\Gamma)$ is the number of terminal nodes in α-decision tree Γ. For this cost function, $\psi^0 = 1$ and $F(n,\psi_1,\psi_2,\ldots,\psi_t) = 1 + \sum_{i=1}^{t} \psi_i$.

3.3 Greedy Approach

Let I be an impurity function. We now describe a greedy algorithm V_I which for a given decision table T and real α, $0 \le \alpha < 1$, constructs an α-decision tree $V_I(T,\alpha)$ for the table T.

Step 1. Construct a tree consisting of a single node labeled with table T and proceed to the second step.

Suppose $t \ge 1$ steps have been made already. The tree obtained at the step t will be denoted by G.

Step $(t+1)$. If no node of the tree G is labeled with a table then we denote $V_I(T,\alpha)$ by the tree G. The work of the algorithm V_I is completed.

Otherwise, we choose a node v in the tree G which is labeled with a subtable Θ of the table T. If $R(\Theta) \le \alpha R(T)$, then instead of Θ we mark the node v by the most common decision for Θ and proceed to the step $(t+2)$. Let $R(\Theta) > \alpha R(T)$. Then for each $f_i \in E(\Theta)$, we compute the value $I(T,f_i)$. Instead of Θ we mark the node v with the attribute f_{i_0}, where i_0 is the minimum $i \in \{1,\ldots,m\}$ for which $I(T,f_i)$ has the minimum value. For each $\delta \in E(\Theta,f_{i_0})$, we add the tree G the node $v(\delta)$ and mark this node with the subtable $\Theta(f_{i_0},\delta)$, draw the edge from v to $v(\delta)$, and mark this edge with δ. Proceed to step $(t+2)$.

Example 1. Let us consider a decision table T_0 depicted on Figure 3.2 and the greedy algorithm which uses the impurity function I with the type *max* and Gini index as uncertainty measure.

$T_0 =$

f_1	f_2	d
0	0	0
0	2	1
0	1	0
1	1	1

Fig. 3.2 Decision table T_0

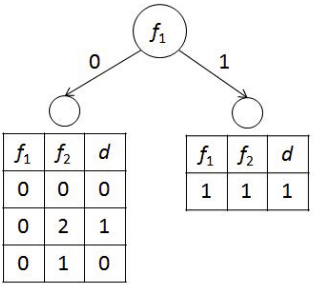


Fig. 3.3 The obtained tree after the second step of greedy algorithm

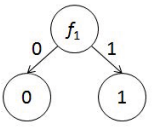


Fig. 3.4 $\frac{1}{2}$-decision tree obtained by the greedy algorithm

We compute the values $I(T_0, f_1)$ and $I(T_0, f_2)$. We have two possible values for f_1, so we will have two subtables $T_1 = T_0(f_1, 0)$ and $T_2 = T_0(f_1, 1)$. We compute Gini index for each of them: $gini(T_1) = 1 - \sum_{i=1}^{2} p_i^2 = 1 - ((2/3)^2 + (1/3)^2) = 4/9$, $gini(T_2) = 1 - \sum_{i=1}^{1} p_i^2 = 1 - (1/1)^2 = 0$, and $I(T_0, f_1) = 4/9$. For f_2, we have three possible values and hence we have three subtables $T_1' = T_0(f_2, 0)$, $T_2' = T_0(f_2, 1)$, and $T_3' = T_0(f_2, 2)$. We compute Gini index for each of them: $gini(T_1') = 1 - (1/1)^2 = 0$, $gini(T_2') = 1 - (1/1)^2 = 0$, and $gini(T_3') = 1 - ((1/2)^2 + (1/2)^2) = 1/2$, and $I(T_0, f_2) = 1/2$. We select f_1 since $I(T_0, f_1) < I(T_0, f_2)$. Figure 3.3 shows the tree obtained after the second step of greedy algorithm.

If we use the algorithm with $\alpha = 1/2$ then the partitioning stops at this level. Figure 3.4 shows the $\frac{1}{2}$-decision tree obtained by the greedy algorithm.

If we use the algorithm with $\alpha = 0$ then the algorithm divides subtable T_1 into subtables. Figure 3.5 shows the obtained tree after the third step of the algorithm

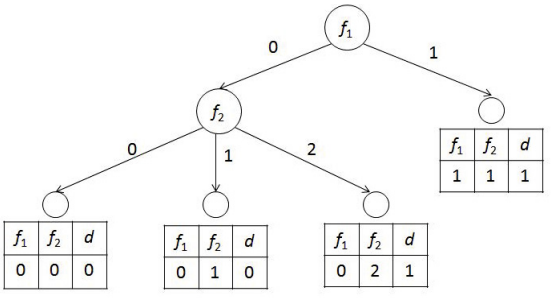


Fig. 3.5 The obtained tree after the third step of greedy algorithm

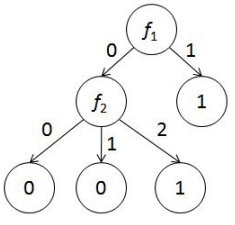


Fig. 3.6 0-decision tree obtained by the greedy algorithm

work. Figure 3.6 shows the 0-decision tree obtained by greedy algorithm. Note that the result of algorithm work for $\alpha = 1/4$ is the same.

3.4 Dynamic Programming Approach

In this section, we describe a dynamic programming algorithm for a monotone cost function ψ, a decision table T, and a real α, $0 \leq \alpha < 1$ which finds the minimum cost (relative to the cost function ψ) of an α-decision tree for T.

Consider an algorithm to construction of a graph $\Delta_\alpha(T)$. Nodes of this graph are some separable subtables of these table T. During each step we process one node and mark it with the symbol *. We start with the graph which consists of one node T and finish when all nodes of the graph are processed.

Let the algorithm have already performed p steps. Let us describe the step $(p+1)$. If all nodes are processed then the work of the algorithm is finished, and the resulted graph is $\Delta_\alpha(T)$. Otherwise, choose a node (table) Θ that has not been processed yet. If $R(\Theta) \leq \alpha R(T)$, mark the considered node with the symbol * and proceed to the step $(p+2)$. Let $R(\Theta) > \alpha R(T)$. For each $f_i \in E(\Theta)$, draw a bundle of edges from the node Θ. Let $E(\Theta, f_i) = \{a_1, \ldots, a_t\}$. Then draw t edges from Θ and label these edges with pairs $(f_i, a_1), \ldots, (f_i, a_t)$ respectively. These edges enter into nodes

$\Theta(f_i,a_1),\ldots,\Theta(f_i,a_t)$. If some nodes of $\Theta(f_i,a_1),\ldots,\Theta(f_i,a_t)$ are not present in the graph then add these nodes to the graph. Mark the node Θ with the symbol * and proceed to the step $(p+2)$.

Let ψ be a monotone cost function given by the pair ψ^0, F. Now we describe a procedure which attaches a number to each node of $\Delta_\alpha(T)$. We attach the number ψ^0 to each terminal node (node without outgoing edges) of $\Delta_\alpha(T)$.

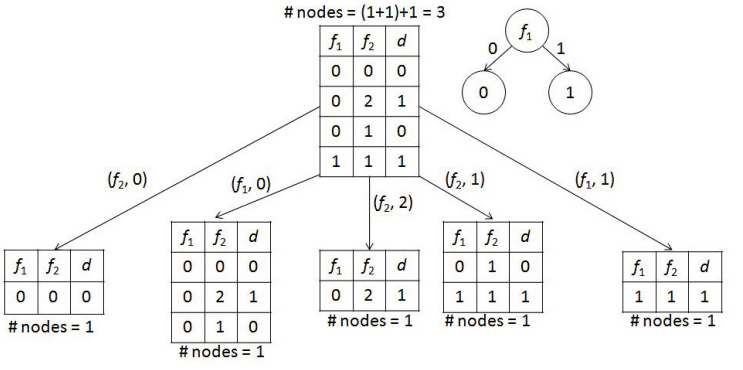


Fig. 3.7 The graph $\Delta_{\frac{1}{2}}(T_0)$ and optimal $\frac{1}{2}$-decision tree for T_0

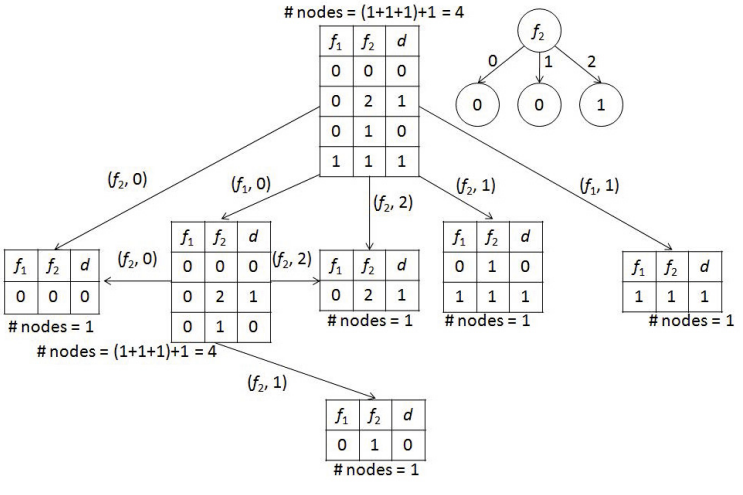


Fig. 3.8 The graph $\Delta_{\frac{1}{4}}(T_0)$ and optimal $\frac{1}{4}$-decision tree for T_0

Consider a node Θ which is not terminal, and a bundle of edges which starts in this node. Let edges be labeled with pairs $(f_i,a_1),\ldots,(f_i,a_t)$, and edges enter into nodes $\Theta(f_i,a_1),\ldots,\Theta(f_i,a_t)$, to which numbers $\psi_1,\ldots,\psi_t$ are attached already.

Then we attach to the considered bundle the number $F(N(\Theta),\psi_1,\ldots,\psi_t)$. Among numbers attached to bundles starting in Θ we choose the minimum number and attach it to the node Θ. We stop when a number will be attached to the node T_0 in the graph $\Delta_\alpha(T_0)$. One can show (see [1]) that this number is the minimum cost (relative to the cost function ψ) of an α-decision tree for T.

Example 2. Let T_0 be the decision table depicted in Figure 3.2. We apply to the table the dynamic programming algorithm along with parameters $\alpha = 1/2$, $\alpha = 1/4$, and $\alpha = 0$. As cost function we will consider the number of nodes. Results are depicted on Figures 3.7-3.9.

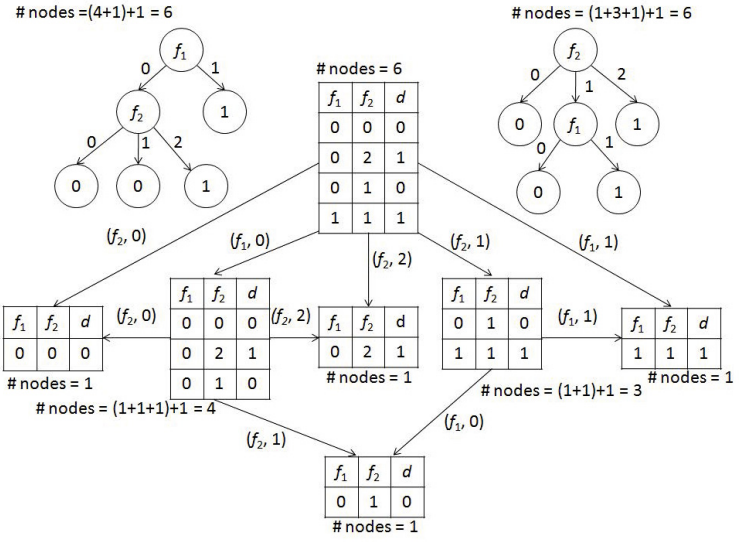


Fig. 3.9 The graph $\Delta_0(T_0)$ and optimal 0-decision trees for T_0

3.5 Experiments with Exact Decision Trees and Decision Tables from UCI ML Repository

We compare average depth, number of nodes, and depth of exact decision trees built by greedy algorithms with the minimum average depth, minimum number of nodes and minimum depth calculated by the dynamic programming algorithm. Tables 3.1-3.3 show results of experiments with 24 data sets and three cost functions: average depth, number of nodes, and depth respectively.

Table 3.1 Results of experiments with average depth

Name	Opt	*sum*				*w_sum*			
		ent	*gini*	*me*	*R*	*ent*	*gini*	*me*	*R*
adult-stretch	1.50	0.00	0.00	1.33	0.00	0.00	0.00	1.33	0.00
agaricus-lepiota	1.52	0.54	0.54	0.01	0.00	0.00	0.00	0.00	0.30
balance-scale	3.55	0.00	0.00	0.02	0.00	0.00	0.00	0.02	0.00
breast-cancer	3.24	0.96	0.96	0.25	0.02	0.08	0.14	0.02	0.03
cars	2.95	0.04	0.04	0.26	0.28	0.00	0.00	0.36	0.49
flags	2.72	2.43	2.58	0.18	0.04	0.16	0.16	0.04	0.03
hayes-roth-data	2.62	0.01	0.01	0.01	0.00	0.01	0.01	0.00	0.00
house-votes-84	3.54	0.66	0.97	0.49	0.06	0.04	0.07	0.06	0.02
lenses	1.80	0.00	0.00	0.67	0.67	0.67	0.00	0.67	0.67
lymphography	2.67	1.66	1.66	0.26	0.06	0.17	0.17	0.05	0.04
monks-1-test	2.50	0.80	0.80	0.00	0.00	0.00	0.00	0.00	0.00
monks-1-train	2.53	0.71	0.71	0.00	0.09	0.26	0.27	0.00	0.00
monks-2-test	5.30	0.01	0.01	0.01	0.05	0.02	0.02	0.05	0.05
monks-2-train	4.11	0.14	0.14	0.10	0.02	0.06	0.06	0.04	0.04
monks-3-test	1.83	1.24	0.52	0.52	0.00	0.00	0.14	0.00	0.00
monks-3-train	2.51	0.50	0.21	0.08	0.01	0.01	0.01	0.01	0.01
nursery	3.45	0.17	0.22	0.09	0.09	0.01	0.00	0.12	0.21
poker-hand-training-true	4.09	0.60	0.60	0.14	0.01	0.01	0.01	0.01	0.01
shuttle-landing-control	2.33	0.69	0.69	0.26	0.00	0.03	0.03	0.00	0.00
soybean-small	1.34	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.41
spect-test	2.95	1.01	0.88	0.67	0.18	0.03	0.13	0.17	0.16
teeth	2.78	0.58	0.62	0.00	0.02	0.02	0.00	0.02	0.02
tic-tac-toe	4.35	0.12	0.07	0.11	0.13	0.06	0.05	0.16	0.17
zoo-data	2.29	0.69	0.69	0.07	0.04	0.04	0.04	0.04	0.05
Average		0.565	0.539	0.230	0.073	0.069	0.055	0.131	0.113

The data sets were taken from UCI Machine Learning Repository [6]. Each data set is represented as a table contains several input columns and an output (decision) column. Some data sets contain index columns that take unique value for each row. Such columns were removed. In some tables there were rows that contain identical values in all columns, possibly, except the decision column. In this case, each group of identical rows was replaced with a single row with common values in all input columns and the most common value in the decision column. In some tables there were missed values. Each such value was replaced with the most common value in the corresponding column.

Each row in Tables 3.1–3.3 contains data set name, minimum cost of decision tree (*min_cost*), calculated with the dynamic programming algorithm (see column Opt), and information about cost of decision trees built by each of the considered greedy algorithms. Instead of the cost of decision tree constructed by greedy algorithm (*greedy_cost*), we consider relative difference of *greedy_cost* and *min_cost*:

$$\frac{greedy\_cost - min\_cost}{min\_cost} \tag{3.1}$$

Table 3.2 Results of experiments with number of nodes

Name	Opt	*sum*				*w_sum*			
		ent	*gini*	*me*	*R*	*ent*	*gini*	*me*	*rt*
adult-stretch	5	0.00	0.00	3.60	0.00	0.00	0.00	3.60	0.00
agaricus-lepiota	21	0.57	0.57	0.62	0.38	0.38	0.38	0.38	3.00
balance-scale	501	0.00	0.00	0.07	0.00	0.00	0.00	0.07	0.00
breast-cancer	161	0.45	0.45	0.37	0.32	0.25	0.25	0.29	0.41
cars	396	0.10	0.10	0.38	0.21	0.03	0.03	0.70	1.35
flags	97	0.80	1.01	0.34	0.62	0.25	0.25	0.59	0.61
hayes-roth-data	52	0.06	0.06	0.06	0.02	0.06	0.06	0.02	0.02
house-votes-84	45	1.02	1.38	1.02	0.36	0.18	0.27	0.31	0.27
lenses	8	0.00	0.00	0.88	0.88	0.88	0.00	0.88	0.88
lymphography	53	0.89	0.89	0.53	0.66	0.43	0.43	0.55	0.77
monks-1-test	37	3.51	3.51	0.11	0.11	0.11	0.11	0.11	0.11
monks-1-train	36	1.86	1.86	0.11	0.67	1.28	1.39	0.11	0.11
monks-2-test	403	0.00	0.00	0.09	0.58	0.19	0.19	0.58	0.58
monks-2-train	129	0.16	0.16	0.43	0.35	0.23	0.23	0.34	0.44
monks-3-test	17	3.65	1.71	2.71	0.00	0.00	0.12	0.00	0.00
monks-3-train	38	0.82	0.37	0.18	0.18	0.11	0.11	0.18	0.18
nursery	1066	0.58	1.11	0.96	0.95	0.09	0.02	1.35	1.37
poker-hand-training-true	18832	0.36	0.36	0.23	0.19	0.18	0.17	0.18	0.20
shuttle-landing-control	15	0.13	0.13	0.00	0.00	0.00	0.00	0.00	0.00
soybean-small	6	0.17	0.17	0.17	0.17	0.17	0.17	0.17	2.83
spect-test	29	1.72	1.45	1.66	0.83	0.14	0.34	0.69	0.76
teeth	35	0.09	0.09	0.00	0.03	0.03	0.00	0.03	0.03
tic-tac-toe	244	0.68	0.41	0.48	1.00	0.41	0.32	1.05	1.09
zoo-data	17	0.59	0.59	0.35	0.35	0.35	0.35	0.35	0.47
Average		0.758	0.682	0.639	0.368	0.239	0.216	0.522	0.645

The last line shows average relative difference of *greedy_cost* and *min_cost*. We evaluate greedy algorithms based on this parameter.

Let us remind that each impurity function is defined by its type (*sum*, *max*, *w_sum* or *w_max*) and uncertainty measure (*ent*, *gini*, *me*, or *R*).

Considering the average depth, we noticed that the type *sum* dominates *max*, i.e. it has less value of average relative difference between greedy_cost and min_cost for each uncertainty measure. Similarly, the type *w_sum* dominates *w_max*. Table 3.1 presents results for two best types: *sum* and *w_sum*. One can see that the three best impurity functions are given by combinations of *w_sum* with Gini index (the criterion used by CART [9]) and entropy (the criterion used by ID3 [26]), and by the combination of *sum* with *R*.

Analysis of experiments for the number of nodes lead us to the same results. The type *sum* dominates *max* and *w_sum* dominates *w_max*. Table 3.2 presents results for two best types: *sum* and *w_sum*. One can see that along with the average depth, the three best impurity functions for the number of nodes are given by combinations of *w_sum* with Gini index, entropy, and by the combination of *sum* with *R*.

Table 3.3 Results of experiments with depth

Name	Opt	*max*	*w_max*				*w_sum*			
		R	*ent*	*gini*	*me*	*R*	*ent*	*gini*	*me*	*R*
adult-stretch	2	1.00	1.00	1.00	1.00	1.00	0.00	0.00	1.00	0.00
agaricus-lepiota	3	0.00	0.00	0.00	0.00	0.33	0.33	0.33	0.33	0.33
balance-scale	4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
breast-cancer	6	0.00	0.00	0.00	0.00	0.00	0.17	0.33	0.00	0.00
cars	6	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
flags	4	0.25	0.25	0.25	0.25	0.25	0.75	0.75	0.25	0.25
hayes-roth-data	4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
house-votes-84	6	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33
lenses	3	0.33	0.33	0.33	0.33	0.33	0.33	0.00	0.33	0.33
lymphography	4	0.25	0.25	0.25	0.25	0.25	0.50	0.50	0.25	0.25
monks-1-test	3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
monks-1-train	3	0.00	0.00	0.00	0.00	0.00	1.00	1.00	0.00	0.00
monks-2-test	6	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
monks-2-train	5	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
monks-3-test	3	0.33	0.33	0.33	0.33	0.33	0.00	0.00	0.00	0.00
monks-3-train	4	0.00	0.00	0.00	0.00	0.00	0.25	0.25	0.00	0.00
nursery	8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
poker-hand-training-true	5	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00
shuttle-landing-control	4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
soybean-small	2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
spect-test	8	0.13	0.13	0.13	0.13	0.13	0.13	0.50	0.25	0.13
teeth	4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
tic-tac-toe	6	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17
zoo-data	4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Average		0.125	0.125	0.125	0.125	0.139	0.173	0.190	0.130	0.083

Experiments with depth lead to different ranking of impurity functions. The type *w_sum* dominates *sum*, and the *w_max* dominates *max* for each uncertainty measure with the exception of *R*. Table 3.4 shows results for the best functions: combinations of *w_max* and *w_sum* with all four uncertainty measures, and the combination of *max* and *R*.

One can see that the best impurity function is given by the combination of *w_sum* with *R*. The following combinations give us similar results: *w_max* and *ent*, *w_max* and *gini*, *w_max* and *me*, and *max* and *R*. The greedy algorithm based on the last combination is known to be close (from the point of view of accuracy) to the best approximate polynomial algorithms for minimization of decision tree depth under some assumptions about the class *NP* [23].

The considered results are represented in Table 3.4. Note that for *h* the combination (w_sum, R) is on the first place and the other four combinations are on the second place (not any one on the third position).

Table 3.4 The best thee greedy algorithms for three cost functions (for h we have four heuristics on the second place)

Cost function	The best three greedy algorithms
h_{av}	(*w_sum*, *gini*), (*w_sum*, *ent*), (*sum*, *R*)
h	(*w_sum*, *R*), (*w_max*, *ent*), (*w_max*, *gini*), (*w_max*, *me*), (*max*, *R*)
L	(*w_sum*, *gini*), (*w_sum*, *ent*), (*sum*, *R*)

3.6 Experiments with Approximate Decision Trees and Decision Tables from UCI ML Repository

We make experiments with 24 data sets from UCI Machine Learning Repository [6]: adult-stretch, agaricus-lepiota, balance-scale, breast-cancer, cars, flags, hayes-roth-data, house-votes-84, lenses, lymphography, monks-1-test, monks-1-train, monks-2-test, monks-2-train, monks-3-test, monks-3-train, nursery, poker-hand-training-true, shuttle-landing-control, soybean-small, spect-test, teeth, tic-tac-toe, zoo-data.We make the same preprocessing of the data sets as described in Section 3.5.

Table 3.5 Average value of a given cost function in 12024 experiments with a given greedy algorithm

	h_{av}	h	L	L_n	L_t
max					
ent	1.442	1.652	10.280	2.808	7.472
gini	1.442	1.651	10.374	2.847	7.527
me	1.206	1.294	10.820	2.317	8.503
R	1.183	**1.263**	10.695	2.202	8.493
sum					
ent	1.818	2.222	10.433	3.204	7.229
gini	1.777	2.185	10.345	3.129	**7.217**
me	1.280	1.420	10.034	2.322	7.712
R	1.170	1.273	10.049	**2.019**	8.030
w_max					
ent	1.190	1.270	10.575	2.218	8.357
gini	1.190	1.266	10.731	2.224	8.507
me	1.186	1.264	10.709	2.213	8.496
R	1.180	1.264	10.866	2.205	8.660
w_sum					
ent	1.227	1.392	9.634	2.062	7.572
gini	1.221	1.393	**9.558**	2.036	7.521
me	1.184	1.273	10.162	2.074	8.089
R	**1.167**	1.266	10.524	2.066	8.458
Average optimal values					
	1.137	1.244	6.931	1.750	4.888

Table 3.6 Number of experiments among 12024 ones in which a given greedy algorithm constructs an optimal decision tree relative to a given cost function

	h_{av}	h	L	L_n	L_t
max					
ent	8373	9126	4784	8434	4633
gini	8629	9383	**4964**	8690	4797
me	10188	11516	3114	10464	2963
R	10865	**11819**	3218	10739	3067
sum					
ent	6288	6487	4743	6424	**5080**
gini	6659	6925	4602	6824	4919
me	9389	10317	3926	9549	3729
R	10996	11671	3584	**10874**	3373
w_max					
ent	10688	11733	3237	10658	3013
gini	10701	11773	3147	10666	2996
me	10718	11802	3160	10679	3009
R	10968	11803	3197	10689	3046
w_sum					
ent	9784	10599	3794	10033	3491
gini	9684	10503	3899	10094	3592
me	10826	11691	3415	10701	3204
R	**11133**	11753	3339	10859	3201

Table 3.7 The best greedy algorithms: three with the minimum average cost of constructed trees and three with the maximum number of optimal trees among constructed

Cost function	The best three greedy algorithms	
	Average cost	Number of optimal solutions
h_{av}	(*w_sum*, *R*), (*sum*, *R*), (*w_max*, *R*)	(*w_sum*, *R*), (*sum*, *R*), (*w_max*, *R*)
h	(*max*, *R*), (*w_max*, *me*), (*w_max*, *R*)	(*max*, *R*), (*w_max*, *R*), (*w_max*, *me*)
L	(*w_sum*, *gini*), (*w_sum*, *ent*), (*sum*, *me*)	(*max*, *gini*), (*max*, *ent*), (*sum*, *ent*)
L_n	(*sum*, *R*), (*w_sum*, *gini*), (*w_sum*, *ent*)	(*sum*, *R*), (*w_sum*, *R*), (*max*, *R*)
L_t	(*sum*, *gini*), (*sum*, *ent*), (*max*, *ent*)	(*sum*, *ent*), (*sum*, *gini*), (*max*, *gini*)

We consider 501 values of α between 0 and 0.5 with step size equal to 0.001. We study 16 greedy algorithms corresponding to 16 impurity functions. Each impurity function is defined by a pair (*type*, *uncertainty measure*) where uncertainty measure belongs to the set {*ent*, *gini*, *me*, *R*} and type belongs to the set {*sum*, *max*, *w_sum*, *w_max*}.

We use five cost functions: average depth h_{av}, depth h, number of nodes L, number of nonterminal nodes L_n, and number of terminal nodes L_t.

For each of 16 greedy algorithms, we make $12024 = 24 \times 501$ experiments: for each of the 24 decision tables T considered and for each of the considered 501 values of α we construct an α-decision tree for T using given greedy algorithm. We find

values of the cost functions h_{av}, h, L, L_n, and L_t for this decision tree. Also, using the dynamic programming algorithm we find minimum values of the cost functions h_{av}, h, L, L_n, and L_t among all α-decision trees for T.

For each 16 greedy algorithms and five cost functions, we find the average value of the considered cost function for trees constructed by the considered greedy algorithm in 12024 experiments (see Table 3.5), and the number of experiments among 12024 ones in which the considered greedy algorithm constructs an optimal decision tree relative to the considered cost function (see Table 3.6).

Also, for each five cost functions, we find the average of minimum values of this function obtained by dynamic programming algorithm in 12024 experiments (see row “Average optimal values" in Table 3.5).

Table 3.7 shows the best three greedy algorithms for each of the cost functions h, h_{av}, L, L_n, L_t.

3.7 Experiments with Exact Decision Trees and Randomly Generated Decision Tables

In this section, we consider results of 40,000 experiments with randomly generated decision tables. Each table contains 50 rows and 10 conditional attributes. The values of conditional and decision attributes are from the set $\{0,1\}$. We choose values 0 and 1 with the same probability.

In some tables, there were rows that contains identical values in all columns, possibly, except the decision column. In this case, each group of identical rows was replaced with a single row with common values in all conditional columns and the most common value on the decision column.

We divide 40000 experiments into four groups with 10000 experiments in each. We study only exact decision trees, 16 greedy algorithms and five cost functions: h, h_{av}, L, L_t, and L_n.

Tables 3.8-3.11 show the values of average relative difference (see (3.1)) for the definition of relative difference) for each cost function, each heuristic and each group of experiments.

Table 3.12 shows the best three greedy algorithms for each cost function, (we have the same best three functions in each group of experiments).

3.8 Analysis of Experimental Results

In this section, we consider only three cost functions: depth h, average depth h_{av}, and number of nodes L. For these cost functions and 16 greedy heuristics we had four “competitions". In each “competitions" (see Tables 3.4, 3.7 and 3.12) we have the first, second and third place winner (with the exception of Table 3.4 when for h we have the first and the number of second place winners).

Table 3.8 Average value of relative difference for given cost function for the first 10000 experiments with a given greedy algorithm using randomly generated tables (10 attributes, 50 rows)

	h_{av}	h	L	L_n	L_t
max					
ent	0.304782	0.529542	0.853598	0.881208	0.827686
gini	0.304865	0.529457	0.853706	0.881318	0.827791
me	0.244603	0.377128	0.788759	0.814279	0.764809
R	0.167678	0.206453	0.639100	0.659751	0.619718
sum					
ent	0.270057	0.614633	0.662513	0.683836	0.642494
gini	0.269812	0.614177	0.661733	0.683030	0.641739
me	0.203394	0.434292	0.602357	0.621768	0.584135
R	0.114111	0.272645	0.460296	0.475098	0.446399
w_max					
ent	0.169921	0.210177	0.639265	0.659926	0.619873
gini	0.173310	0.216563	0.643132	0.663912	0.623629
me	0.175426	0.219522	0.649515	0.670506	0.629814
R	0.165935	0.204393	0.639306	0.659946	0.619933
w_sum					
ent	0.128242	0.440730	0.417583	0.431018	0.404970
gini	0.125142	0.427735	0.416971	0.430390	0.404373
me	0.140525	0.269668	0.512560	0.529063	0.497067
R	0.121462	0.221365	0.506136	0.522423	0.490844

Table 3.9 Average value of relative difference for given cost function for the second 10000 experiments with a given greedy algorithm using randomly generated tables (10 attributes, 50 rows)

	h_{av}	h	L	L_n	L_t
max					
ent	0.305162	0.531963	0.853261	0.880867	0.827353
gini	0.305048	0.531873	0.852970	0.880568	0.827069
me	0.245566	0.376347	0.792148	0.817765	0.768106
rt	0.167695	0.206953	0.638269	0.658882	0.618921
sum					
ent	0.271280	0.618595	0.664276	0.685647	0.644211
gini	0.271144	0.618103	0.664264	0.685638	0.644197
me	0.203429	0.435148	0.601539	0.620941	0.583326
rt	0.114165	0.272575	0.459519	0.474296	0.445645
w_max					
ent	0.170312	0.210923	0.639822	0.660496	0.620417
gini	0.174373	0.215938	0.646212	0.667095	0.626612
me	0.176658	0.219213	0.652403	0.673477	0.632623
rt	0.166464	0.205520	0.640040	0.660695	0.620651
w_sum					
ent	0.128920	0.442818	0.419319	0.432807	0.406657
gini	0.125799	0.430868	0.418550	0.432016	0.405907
me	0.140787	0.270515	0.513427	0.529965	0.497901
rt	0.121369	0.222027	0.505065	0.521319	0.489805

Table 3.10 Average value of relative difference for given cost function for the third 10000 experiments with a given greedy algorithm using randomly generated tables (10 attributes, 50 rows)

	h_{av}	h	L	L_n	L_t
max					
ent	0.305115	0.532015	0.853592	0.881213	0.827670
gini	0.305057	0.531647	0.853340	0.880953	0.827427
me	0.244307	0.375772	0.788406	0.813905	0.764475
rt	0.167766	0.207182	0.639119	0.659763	0.619742
sum					
ent	0.271189	0.617525	0.664806	0.686199	0.644722
gini	0.270936	0.616903	0.664500	0.685884	0.644424
me	0.202868	0.436132	0.600668	0.620046	0.582479
rt	0.113847	0.272843	0.458596	0.473350	0.444744
w_max					
ent	0.170172	0.210522	0.640091	0.660778	0.620675
gini	0.173779	0.215793	0.644979	0.665826	0.625412
me	0.176172	0.219157	0.651384	0.672429	0.631633
rt	0.166324	0.205603	0.640585	0.661266	0.621173
w_sum					
ent	0.129026	0.443065	0.419490	0.432990	0.406816
gini	0.125948	0.430685	0.418858	0.432341	0.406200
me	0.141226	0.269815	0.515113	0.531712	0.499531
rt	0.121313	0.222042	0.504982	0.521238	0.489720

Table 3.11 Average value of relative difference for given cost function for the fourth 10000 experiments with a given greedy algorithm using randomly generated tables (10 attributes, 50 rows)

	h_{av}	h	L	L_n	L_t
max					
ent	0.304364	0.530182	0.850598	0.878099	0.824786
gini	0.304260	0.529727	0.850383	0.877879	0.824577
me	0.245225	0.378018	0.789103	0.814610	0.765163
rt	0.167005	0.207123	0.635668	0.656183	0.616413
sum					
ent	0.270566	0.616868	0.661972	0.683266	0.641980
gini	0.270394	0.617112	0.661650	0.682936	0.641665
me	0.203550	0.434280	0.601649	0.621043	0.583443
rt	0.114046	0.271858	0.459410	0.474185	0.445539
w_max					
ent	0.169538	0.210553	0.637208	0.657776	0.617903
gini	0.173338	0.216723	0.642447	0.663194	0.622973
me	0.175919	0.218963	0.649086	0.670042	0.629417
rt	0.165750	0.205487	0.638142	0.658726	0.618819
w_sum					
ent	0.128567	0.442015	0.417987	0.431433	0.405363
gini	0.125746	0.429720	0.417998	0.431445	0.405374
me	0.140318	0.269610	0.511820	0.528300	0.496349
rt	0.121553	0.221110	0.505367	0.521624	0.490104

Table 3.12 The best thee greedy algorithms for each cost function

Cost function	The best three greedy algorithms
h_{av}	(*sum*, *R*), (*w_sum*, *R*), (*w_sum*, *gini*)
h	(*w_max*, *R*), (*max*, *R*), (*w_max*, *ent*)
L	(*w_sum*, *ent*), (*w_sum*, *gini*), (*sum*, *R*)
L_n	(*w_sum*, *ent*), (*w_sum*, *gini*), (*sum*, *R*)
L_t	(*w_sum*, *ent*), (*w_sum*, *gini*), (*sum*, *R*)

We choose for each cost function all heuristics which were the first place winners in at least one "competition", or were winners in at least three "competitions". The results can be found in Table 3.13. The tuple (2, 1, -) means that the considered heuristic was the first place winner in two "competitions" and the second place winner in one "competition" (and was not the third place winner in any "competition").

Table 3.13 Good heuristics for optimization of depth h, average depth h_{av}, and number of nodes L

h_{av}	h	L
(*w_sum*, *R*) (2, 1, -)	(*max*, *R*) (2, 2, -)	(*w_sum*, *gini*) (2, 1, -)
(*sum*, *R*)(1, 2, 1)	(*w_max*, *R*) (1, 1, 1)	(*w_sum*, *ent*) (1, 2, -)
(*w_sum*, *gini*)(1, -, 1)	(*w_sum*, *R*) (1, -, -)	(*max*, *gini*) (1, -, -)
	(*w_max*, *me*) (-, 2, 1)	

3.9 Conclusions

We compared 16 greedy algorithms for approximate optimization of decision trees with dynamic programming approach for decision tree optimization. For depth, average depth, and number of nodes, we chose 3–4 heuristics which looks better than the other one. The future study will be connected with performing of large number of experiments with randomly generated data sets from different classes.

Acknowledgements. The authors are greatly indebted to Professor Andrzej Skowron for his comments and suggestions.

References

1. Alkhalid, A., Chikalov, I., Moshkov, M.: On Algorithm for Building of Optimal α-Decision Trees. In: Szczuka, M., Kryszkiewicz, M., Ramanna, S., Jensen, R., Hu, Q. (eds.) RSCTC 2010. LNCS (LNAI), vol. 6086, pp. 438–445. Springer, Heidelberg (2010)
2. Alkhalid, A., Chikalov, I., Moshkov, M.: A Tool for Study of Optimal Decision Trees. In: Yu, J., Greco, S., Lingras, P., Wang, G., Skowron, A. (eds.) RSKT 2010. LNCS, vol. 6401, pp. 353–360. Springer, Heidelberg (2010)

3. Alkhalid, A., Chikalov, I., Moshkov, M.: Comparison of Greedy Algorithms for α-Decision Tree Construction. In: Yao, J., Ramanna, S., Wang, G., Suraj, Z. (eds.) RSKT 2011. LNCS (LNAI), vol. 6954, pp. 178–186. Springer, Heidelberg (2011)
4. Alkhalid, A., Chikalov, I., Moshkov, M.: Decision tree construction using greedy algorithms and dynamic programming – comparative study. In: Szczuka, M., et al. (eds.) Concurrency, Specification and Programming. Proceedings of International Workshop CS&P 2011, Pultusk, Poland, pp. 1–9. Bialystok University of Technology, Poland (2010)
5. Alkhalid, A., Chikalov, I., Moshkov, M.: Comparison of greedy algorithms for decision tree construction. In: Proceedings of International Conference on Knowledge Discovery and Information Retrieval KDIR 2011, Paris, France (2011) (to appear)
6. Frank, A., Asuncion, A.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine (2011), `http://www.ics.uci.edu/ml` (cited October 20, 2010)
7. Breitbart, Y., Reiter, A.: A branch-and-bound algorithm to obtain an optimal evaluation tree for monotonic boolean functions. Acta Inf. 4, 311–319 (1975)
8. Boryczka, U., Kozak, J.: Ant Colony Decision Trees – A New Method for Constructing Decision Trees Based on Ant Colony Optimization. In: Pan, J.-S., Chen, S.-M., Nguyen, N.T. (eds.) ICCCI 2010, Part I. LNCS, vol. 6421, pp. 373–382. Springer, Heidelberg (2010)
9. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth & Brooks (1984)
10. Chai, B.-B., Zhuang, X., Zhao, Y., Sklansky, J.: Binary linear decision tree with genetic algorithm. In: Proceedings of the International Conference on Pattern Recognition (ICPR 1996), vol. 7472, pp. 530–534. IEEE Computer Society Press, Los Alamitos (1996)
11. Chikalov, I., Hussain, S., Moshkov, M.: Relationships between Depth and Number of Misclassifications for Decision Trees. In: Kuznetsov, S.O., Ślęzak, D., Hepting, D.H., Mirkin, B.G. (eds.) RSFDGrC 2011. LNCS, vol. 6743, pp. 286–292. Springer, Heidelberg (2011)
12. Chegis, I.A., Yablonskii, S.V.: Logical methods of electric circuit control. Trudy MIAN SSSR 51, 270–360 (1958) (in Russian)
13. Chikalov, I., Moshkov, M., Zelentsova, M.: On Optimization of Decision Trees. In: Peters, J.F., Skowron, A. (eds.) Transactions on Rough Sets IV. LNCS, vol. 3700, pp. 18–36. Springer, Heidelberg (2005)
14. Fayyad, U.M., Irani, K.B.: The attribute specification problem in decision tree generation. In: Proceedings of the 10th National Conference on Artificial Intelligence (AAAI 1992), San Jose, CA, pp. 104–110 (1992)
15. Garey, M.R.: Optimal binary identification procedures. SIAM Journal on Applied Mathematics 23, 173–186 (1972)
16. Heath, D., Kasif, S., Salzberg, S.: Introduction to oblique decision trees. In: Ruzena, B. (ed.) The 13th International Joint Conference on Artificial Intelligence (IJCAI 1993), Chambéry, France, pp. 1002–1007 (1993)
17. Hyafil, L., Rivest, R.: Constructing optimal binary decision trees is NP-complete. Information Processing Letters 5, 15–17 (1976)
18. Kononenko, I.: On biases in estimating multi-valued attributes. In: Mellish, C. (ed.) Proceedings of the 14th International Joint Conference on Artificial intelligence (IJCAI 1995), vol. 2, pp. 1034–1040 (1995)
19. Martin, J.K.: An exact probability metric for decision tree splitting and stopping. Machine Learning 28, 257–291 (1997)
20. Mingers, J.: Expert systems — rule induction with statistical data. J. of the Operational Research Society 38, 39–47 (1987)
21. Moret, B.E., Thomason, R.C., Gonzalez, M.: The activity of a variable and its relation to decision trees. ACM Transactions on Programming Languages and Systems (TOPLAS) 2, 580–595 (1980)
22. Martelli, A., Montanari, U.: Optimizing decision trees through heuristically guided search. Commun. ACM 21, 1025–1039 (1978)

23. Moshkov, M.: Time Complexity of Decision Trees. In: Peters, J.F., Skowron, A. (eds.) Transactions on Rough Sets III. LNCS, vol. 3400, pp. 244–459. Springer, Heidelberg (2005)
24. Moshkov, M., Chikalov, I.: Consecutive optimization of decision trees concerning various complexity measures. Fundam. Inform. 61, 87–96 (2003)
25. Pawlak, Z.: Rough Sets – Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Dordrecht (1991)
26. Quinlan, J.R.: Induction of decision trees. Mach. Learn. 1, 81–106 (1986)
27. Riddle, P., Segal, R., Etzioni, O.: Representation design and brute-force induction in a Boeing manufacturing domain. Applied Artificial Intelligence 8, 125–147 (1994)
28. Schumacher, H., Sevcik, K.C.: The synthetic approach to decision table conversion. Commun. ACM 19, 343–351 (1976)
29. Skowron, A., Rauszer, C.: The discernibility matrices and functions in information systems. In: Słowiński, R. (ed.) Intelligent Decision Support. Handbook of Applications and Advances of the Rough Set Theory, pp. 331–362. Kluwer Academic Publishers, Dordrecht (1992)

Chapter 4
A Review of the Knowledge Granulation Methods: Discrete vs. Continuous Algorithms

Piotr Artiemjew

Abstract. The paradigm of granular rough computing has risen quite recently — was initiated by Professor Lotfi Zadeh in 1979. This paradigm is strictly connected with the Rough Set Theory, which was proposed by Professor Zdzisław Pawlak in 1982. Granular rough computing is a paradigm in which one deals with granules that are aggregates of objects connected together by some form of similarity. In the rough set theory granules are traditionally defined as indiscernibility classes, where as similarity relations we use rough inclusions. Granules have a really wide spectrum of application, starting from an approximation of decision systems and ending with an application to the classification process. In this article, approximation methods are shown in the framework of Rough Set Theory. In this chapter we introduce both discrete and continuous granular methods known in the literatureas well as our own modifications along with a practical description of the application of these methods. For described here granulation methods, we have chosen suitable methods of classification which can work properly with shown algorithms.

Keywords: Rough sets, granular computing, granulation of knowledge, decision systems.

4.1 Introduction

Granular computing is the paradigm in the field of approximative reasoning useful in research of data structures by introducing into data sets the similarity relation of objects.

Piotr Artiemjew
Department of Mathematics and Computer Science
University of Warmia and Mazury, Olsztyn, Poland
e-mail: artem@matman.uwm.edu.pl

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 41–59.
springerlink.com

In the rough set theory, similar objects are collected together and form similarity classes, which leads to new structures — granular structures.

In this chapter we describe chosen similarity relations — rough inclusions and some of their weaker variants — with applications to classification problems. These applications consist of methods for granulation of knowledge in data sets.

Granular structures are useful in two ways,

- as preprocessing of data set,
- as heuristics helpful in the process of classifier design.

This chapter therefore is devoted to the granulation of knowledge theory and some of its applications. That theory was initiated by L.A. Zadeh [39] in the framework of fuzzy set theory (1965). We can make an assumption that the uncertainty of the description of a concept X is expressed by the function of fuzzy inclusion $\mu_x : X \mapsto [0,1]$, the objects with the same value of function μ_x are treated as indiscernible and form the granules - see [31], [38].

4.1.1 Basic on Rough Sets

There is an analogous situation in the rough set theory initiated by Professor Zdzisław Pawlak [14] (1982), in which objects and concepts are described based on indiscernibility relations. Classes of indiscernibility relations form the basic set of objects, which is useful to computations in this theory - those classes are known as elementary granules of knowledge.

The formalization of knowledge used in this work, was proposed by Professor Zdzisław Pawlak as *information systems* i.e. pairs (U,A) where U is the universe of objects, A is the set of conditional attributes. By 'knowledge' we understand here indiscernibility relations, which classify objects by

$$IND_B = \{(u,v) \in U \times U : a(u) = a(v), a \in B\}, \tag{4.1}$$

where $B \subseteq A$, it is some form of logic in the sense of J.M. Bocheński [7].

Classification can be made by means of granules of knowledge in the sense of rough set theory based on transformation from objects to their granules — indiscernibility relations IND_B, for the attribute set B.

In the development of rough set theory, it turns out that it is worth trying to consider more general indiscernibility relations - that is similarity relations. Tolerance relations, which are reflexive and symmetric, were brought into rough set theory by J.Nieminen [13], L. Polkowski, A. Skowron, J. Żytkow [19], S. Marcus [12] and A. Skowron, J. Stepaniuk [36, 37], generalization to reflexive not necessarily symmetric ones comes from L. Polkowski, A. Skowron [20, 21] as well as R. Słowiński and D. Vanderpooten [32]. Introducing similarity relations by L.Polkowski and A.Skowron [20, 21] named *rough inclusions* was significant moment for this paradigm. In the framework of rough mereology it was developed by L. Polkowski.

For more information — see an interesting theoretical review of rough set methods in [17].

4.1.2 From Rough Inclusions to Granular Structures

A rough inclusion on the universe U of an information system (U,A) is defined as a three-argument relation

$$\mu(u,v,r_{gran}),$$

where

$$u,v \in U, r_{gran} \in [0,1],$$

"$object\ u\ is\ a\ part\ of\ object\ v\ in\ degree\ \geq r_{gran}$."

The term "part", leads us to theory of mereology by S.Leśniewski [10]. Mereology in the sense of S.Leśniewski describes individual objects in terms of part relations, with assumption, that it is anti-reflexive and transitive.

A relation of an ingredient *ing* is defined as 'to be a part', or 'identical to'.

Formal definition of μ refers to mereology in the sense that 'to be part in degree 1' is identical with 'to be ingredient' relation *ing*:

$$\mu(u,v,1) \Leftrightarrow u\ ing\ v \Leftrightarrow (u\ part\ v) \vee (u = v). \tag{4.2}$$

In this article we will apply the formal mechanism of knowledge granulation proposed by L.Polkowski [24, 30]. This mechanism will also be modified. This mechanism as the elementary notion assumes a *rough inclusion* $\mu(u,v,r_{gran})$ and defines granules of knowledge as mereological classes of μ. In more details, for fixed granulation radius $r_{gran} \in [0,1]$, we consider the property of objects $\mu_{r_{gran}}(v)$ which is fulfilled for objects v, when $\mu_{r_{gran}}(v,u,r_{gran})$. The granule of knowledge with a central object u and radius r_{gran} is defined as the class with property $\mu_{r_{gran}}(v)$: $g_\mu = Cls\ \mu_{r_{gran}}(v)$, cf. *e.g.*, Polkowski [31], where Cls is the mereological class operator in S.Leśniewski's sense [10].

The problem of determining specific *rough inclusions* in the system (U,A) was considered in L.Polkowski's works [22–24], see also [30,31], in which the following methods of μ introducing were described:

1. Inclusions induced from metrics on the universe U.
2. Inclusions induced from Archimedean t-norm.
3. Inclusions induced from continuous t-norms.
4. Some variants of inclusions, which are not rough inclusions, but are considered as having a measure of similarity between objects, take into consideration the range of attributes values in the space of attributes.

We present these inclusions with their variants in the short form below.

Ad1. Inclusions induced from metrics on the universe U, are inspired by Henri Poincaré idea [18], who gave the example of tolerance relation: for metric ρ on U with $\delta > 0$, we consider relation

$$u\ \tau_\rho(\delta)\ v \Leftrightarrow \rho(u,v) < \delta. \tag{4.3}$$

Simply, τ_ρ is reflexive and symmetrical — the tolerance relation is not generally transitive, if ρ is not a non-Archimedean metric, we cannot fulfil the condition,

$$\rho(u,v) \leq max\{\rho(u,w),\rho(w,v)\}. \tag{4.4}$$

For the metric ρ limited by 1 on U, we let:

$$\mu_\rho(u,v,r_{gran}) \Leftrightarrow \rho(u,v) \leq 1 - r_{gran}. \tag{4.5}$$

Hence μ is *rough inclusion*.

The particular and significant example of μ_ρ is obtained for ρ which is equal to the Hamming distance h modulo attributes set A on U, i.e.,

$$h(u,v) = \frac{|\{a \in A:\ a(u) \neq a(v)\}|}{|A|}. \tag{4.6}$$

Inclusion μ_h is the basic tool of our work.

Ad2. A t-norm $t : [0,1] \times [0,1] \mapsto [0,1]$ is the increasing in each coordinate, associative, symmetric and normalized by conditions $t(x,0) = 0$, $t(x,1) = x$.

The continuous Archimedean t-norm t fulfils condition $t(x,x) < x$ for $x \neq 0,1$.

The theorem of representation (Ling [11]): was obtained for the following t-norms

$$t(x,y) = g(f(x) + f(y)), \tag{4.7}$$

a particular case of Kołmogorow–Arnold theorem occurs [1], [9].

The rough inclusion μ_t is defined as,

$$\mu_t(u,v,r_{gran}) \Leftrightarrow g(\frac{|\{a \in A:\ a(u) \neq a(v)\}|}{|A|}) \geq r_{gran}. \tag{4.8}$$

In particular for the Łukasiewicz t-norm

$$L(x,y) = max\{0,\ x+y-1\}, \tag{4.9}$$

one obtains μ_L equal to μ_h.

Ad3. For continuous t-norm t, there does not generally exist such a simple representation (see V.Arnold's example [1] for minimum t-norm). In L.Polkowski's works [24, 30] were proposed other methods, like usage of residual implications, $x \Rightarrow_t y$. The implication $x \Rightarrow_t y$ is defined by duality

$$x \Rightarrow_t y \geq r_{gran} \Leftrightarrow t(x,r_{gran}) \leq y. \tag{4.10}$$

Thus,

$$x \Rightarrow_t y\ =\ max\{r_{gran}:\ t(x,r_{gran}) \leq y\}. \tag{4.11}$$

For continuous t-norms $max\{r_{gran} : t(x,y) \leq y\}$ is achieved.

If we assume

$$\mu_{\Rightarrow_t}(x,y,r_{gran}) \Leftrightarrow x \Rightarrow_t y \geq r_{gran}, \tag{4.12}$$

for $x,y \in [0,1]$ we define *rough inclusion* on the interval [0,1], cf. [31].

It is worth saying, that the inclusions mentioned in points 1–3 operate with the Hamming relation on attribute values and does not take into account the metrical relations in the attribute value space and between values of various attributes.

Ad4. We can take into account some basic variants of *rough inclusions* from [31],
a. we fix value of parameter $\varepsilon \in [0,1]$ and we let

$$\mu_{\rho}^{\varepsilon}(u,v,r_{gran}) \Leftrightarrow \frac{|\{a \in A : \rho(a(u),a(v)) < \varepsilon\}|}{|A|} \geq r_{gran}, \tag{4.13}$$

for established metric ρ on attributes' values (we let, that ρ is common for all attributes).

μ_{ρ}^{ε} generally is not the *rough inclusion*, but maintains a basic metrical relation of similarity between objects, and is measured as a fraction of the attributes number, whose values on pair u,v are ε close.

In the particular case

$$\mu_{\rho}^{\varepsilon}(u,v,1) \Leftrightarrow \rho(a(u),a(v)) < \varepsilon, \tag{4.14}$$

for all $a \in A$,
b. A more subtle metric relationship between objects — described before in [31] — is as follows,

$$ind_{\varepsilon}(u,v) = \frac{IND_{\varepsilon}(u,v)}{|A|} \text{ and } dis_{\varepsilon}(u,v) = \frac{DIS_{\varepsilon}(u,v)}{|A|}, \tag{4.15}$$

where

$$IND_{\varepsilon}(u,v) = |\{a \in A : \rho(a(u),a(v)) < \varepsilon\}|, \tag{4.16}$$

$$DIS_{\varepsilon}(u,v) = |\{a \in A : \rho(a(u),a(v)) \geq \varepsilon\}|. \tag{4.17}$$

The similarity relation $\mu_{\Rightarrow_t}^{\varepsilon}(u,v,r_{gran})$ is defined as follows,

$$\mu_{\Rightarrow_t}^{\varepsilon}(u,v,r_{gran}) \Leftrightarrow dis_{\varepsilon}(u,v) \Rightarrow_t ind_{\varepsilon}(u,v). \tag{4.18}$$

These similarity relations have been useful in the procedure of granulation of knowledge, see [2–6], [28,29].

4.2 General Strategies of Knowledge Granulation

The general strategy of knowledge granulation with regard to the considered similarity function $\mu(u,v,r)$, can be described as follows.

1. The granular radius $r_{gran} \in [0,1]$ has been chosen (in case of $r_{gran} = 0$, we have only one granule - universe U).
2. For considered granulation radius r_{gran}, we create a set of granules $g_{\mu}(u, r_{gran})$ with the central object $u \in U$.
3. From the set of all granules $g_{\mu}(u, r_{gran})$ we choose covering $C_{\mu}(r_{gran})$ of universe U, applying a strategy, $\wp$.

The idea shown in the works [24, 30], consists of representing the system (U,A) as a granular system $(C_{\mu}(r_{gran}),\bar{A})$, where $\bar{A} = \{\bar{a} : a \in A\}$ is the set of attributes filtrated by granules.

A definition of attribute $\bar{a}$, refers to the strategy $\mathcal{S}$ based on which we can choose $\bar{a}(g)$, of attribute $\bar{a}$ on the granule g, that is, $\bar{a}(g) = \mathcal{S}(\{a(u) : u \text{ w granuli } g\})$ System $(C_{\mu}(r_{gran}),\bar{A})$ is *the granular reflection* of system (U,A) relative to strategy $\wp,\mathcal{S}$.

In this article we review some strategies of granulation, which are based on the previously described rough inclusions and their weak variants, we show the following methods:

1. Standard strategy.
2. Relative to the decision classes strategy (*concept-dependent*).
3. Hierarchical strategy (*layered* granulation).
4. Hybrid strategy (*concept-dependent+ layered* granulation).
5. Standard ε - strategy.
6. Concept-dependent ε - strategy.

4.2.1 The Decision Systems in Professor Zdzisław Pawlak's Sense

The decision system in a formal sense is defined as the triple (U,A,d), where

$$U - \text{set of objects},$$

$$A - \text{set of attributes},$$

$$d - \text{decision attribute}.$$

The attributes from the set A are the conditional attributes, which are determined by a system designer, who has based them on a kind of intuition, experience, or premises of significance. The finite character of the set A is the reason for ambivalence: the objects are described in the rough sense and we cannot discern them certainly.

The decision attribute (which can assign, for example, decision classes) is implicated by the real world or a well-informed expert in some context of that world.

The objects form U are represented by *information vectors* defined as follows:

$$Inf_A(u) = \{(a, a(u)) : a \in A\}. \tag{4.19}$$

The relation between description in terms of attributes in A and in terms of the decision d, are given in fixed classifier, which is a set of the following decision rules,

$$r : \forall_{a \in B}(a, v_a) \Rightarrow (d, v), \text{ where } v_a \in V_a,\ d \in V_d, B \subseteq A. \tag{4.20}$$

4.3 Approximation of Decision Systems Methods

We start from a description of exemplary methods examined in the recent years by Polkowski and Artiemjew — see [2–6], [28, 29]. The standard method, concept-dependent method and the layered method are based on above mentioned algorithms and are discrete methods, whereas granular methods with parameter ε belonging in the group of continuous methods.

In the first step we start from a description of our basic standard granulation method.

4.3.1 Standard Granulation

Standard granulation — in the sense of L.Polkowski — is the method of decision systems approximation, which works best with discrete values of attributes. It works very well in the systems, in which conditional attributes and decision attributes are symbolic or integer. It is possible to granulate decision systems, with fractional values of attributes, but on condition that the domain of conditional attributes is limited to a small number of elements. For the decision systems with real value attributes, which have a huge domain, granules of knowledge can contain a small number of objects, and even at the small granular radius near zero, the granular systems approximate the original decision systems in slight degree. For this kind of granules we can use classifiers which works in discrete manner.

The downside of standard granulation is the possibility of losing decision classes at small granulation radii, since some classes of original systems may not appear in the granular system. For this reason standard granulation works best for an average size of granulation radii. The standard granulation procedure looks as follows,

The procedure

1. The original decision system (U,A,d) has been input.
2. The granular radius r_{gran} has been fixed.

3. For all objects $u, v \in U$, we compute $IND(u,v) = \{a \in A : a(u) \neq a(v)\}$ and we form granule g of the central object u, i.e.,

$$v \in g_{r_{gran}}(u) \Leftrightarrow \mu(v,u,r_{gran}) \Leftrightarrow \frac{|IND(u,v)|}{|A|} \geq r_{gran}, \tag{4.21}$$

hence

$$g_{r_{gran}}(u) = \{v : \frac{|IND(u,v)|}{|A|} \geq r_{gran}\}. \tag{4.22}$$

We use μ formally derived form the Łukasiewicz t-norm.

4. The granular covering U_{cover} of the original decision system can be chosen based on one of the following methods:

   - Hierarchical covering (granules are chosen sequentially),
   - Random covering,
   - Granules with minimal, average or maximal length,
   - Granules which add to the covering respectively the least, the most, or an average number of new objects,
   - Random granule choice dependent on decision concepts' size.

   In all of the above covering finding methods, the considered granule can be added to granular covering, only if it gives at least one new object to the covering.

5. The original decision system is considered as covered when the sum of objects from all chosen granules is equal to the original decision system, i.e.,

$$\bigcup \{g_{r_{gran}}(u) : g_{r_{gran}}(u) \in U_{cover}\} = U. \tag{4.23}$$

6. The *majority voting* strategy is applied inside covering granules, where ties are resolved randomly. Finally we obtain new objects, which are granular representatives of covering granules. The granular decision system is formed from obtained objects.

The simplifying pseudo-code of standard granulation look as follows:

Pseudo-code

Granule formation
Data:
decision system (U,A,d)
ganulation radius r_{gran}
for each u in U **do**
for each v in U **do**
if $\frac{|IND(u,v)|}{|A|} \geq r_{gran}$ **then** $g_{r_{gran}}(u) \leftarrow g_{r_{gran}}(u) \cup v$
return $\{g_{r_{gran}}(u) : u \in U\}$

Searching for granular covering of U
Data:
$G = \{g_{r_{gran}}(u) : u \in U\}$
Procedure *random*
Variables:
$U_{cover} \leftarrow \{\}$
$temp \leftarrow G$
loop:
$g \leftarrow random(temp)$
if $g \subseteq U_{cover}$ **then** $temp \leftarrow temp$ - g
else $U_{cover} \leftarrow U_{cover} \cup g$
if $U_{cover} = U$ **then break**

Granular objects creation
Data:
Variables:
GS $\leftarrow \{\}$
for each g in U_{cover} **do**
$gs \leftarrow majority\ voting\ (g)$
$GS \leftarrow GS \cup gs$
return GS

Finally, we obtain new objects, which are granular representations of covering granules. The granular decision system is formed from obtained objects.

The obtained object set GS is the $granular\ reflection\ of\ the\ system\ (U,A,d)$. The next model of granulation based on approximation in the decision concepts is the following.

4.3.2 Concept Dependent Granulation

The concept-dependent granulation differs from standard granulation with respect to granule creation, in this algorithm granules are formed in the range of decision concepts. It means that granules with the fixed central object, take into account only objects in the same decision class as the central object. Concept dependent granular decision systems contain at least one object from all classes of original decision systems. Even for small granulation radii, this method works effectively, as proven in [2, 6]. Concept dependent granulation is used for the same kind of data as the standard one. The procedure of this method is analogous to standard granulation with the difference that granules are defined as follows,

In

$$v \in g^{cd}_{r_{gran}}(u) \Leftrightarrow \mu^{cd}(v,u,r_{gran}) \wedge (d(u) = d(v)), \tag{4.24}$$

hence,

$$v \in g^{cd}_{r_{gran}} \Leftrightarrow (\frac{|IND(u,v)|}{|A|} \geq r_{gran}) \wedge (d(u) = d(v)), \tag{4.25}$$

thus,

$$g^{cd}_{r_{gran}}(u) = \{v : (\frac{|IND(u,v)|}{|A|} \geq r_{gran}) \wedge (d(u) = d(v))\}. \tag{4.26}$$

The need for maximal reduction of the size of the original system at fixed granulation degree leads to the idea of multiple-step granulation, which is described in the next subsection.

4.3.3 Layered Granulation

Layered granulation consists of granulation in the standard sense repeated until the reduction of the original decision system is maximal so it is impossible to reduce it further. The effectiveness of the described method can be studied in [2,6]. The pseudo-code of the described method looks as follows.

Pseudo-code

layer loop:
Granule firmation
Data:
decision system (U,A,d)
granulation radius r_{gran}
for each u in U **do**
for each v in U **do**
if $\frac{|IND(u,v)|}{|A|} \geq r_{gran}$ **then** $g_{r_{gran}}(u) \leftarrow g_{r_{gran}}(u) \cup v$
return $\{g_{r_{gran}}(u) : u \in U\}$

Searching for granular covering of U
Data:
$G = \{g_{r_{gran}}(u) : u \in U\}$
Procedure *random*
Variables:
$U_{cover} \leftarrow \{\}$
$temp \leftarrow G$
loop:
$g \leftarrow random(temp)$
if $g \subseteq U_{cover}$ **then** $temp \leftarrow temp$ - g
else $U_{cover} \leftarrow U_{cover} \cup g$

if $U_{cover} = U$ **then break**

Granular objects creation
Data:
Procedure of *majority voting*
Variable:
GS $\leftarrow \{\}$
for each g in U_{cover} **do**
$gs \leftarrow majority\ voting\ (g)$
$GS \leftarrow\ GS \cup\ gs$
return GS

if $|U| = |GS|$ **then break**
else $U \leftarrow GS$
//the end of layer loop

Clearly, this method can be along the layered granulation leading to the hybrid concept-dependent layered granulation.

4.3.4 Concept Dependent Layered Granulation

In this variant, as with earlier methods, the algorithm differs from the standard method in granule forming manner, granules are defined as,

$$g^{cd}_{r_{gran}}(u) = \{v : (\frac{|IND(u,v)|}{|A|} \geq r_{gran}) \wedge (d(u) = d(v))\}. \tag{4.27}$$

The substantial difference between the concept-dependent method and the standard one is the size of granules at fixed granulation radius. The concept-dependent strategy generates smaller granules, and the granular systems contains more objects in comparison with the standard method.

The last group of approximation algorithms consists of using indiscernibility ratio of descriptors, which leads to the effective granulation of decision systems with a huge domain of conditional attributes.

4.3.5 ε - Granulation

ε variants of granulation are designed to be used on numerical data, which can have a huge domain of conditional attributes. In these methods of approximation we choose objects which are indiscernible to the central object of granule to a degree at least r_{gran}. Discrete classifiers cannot be applied here because the domain of the granular system can differ from the domain of the original training system, hence,

the classification of test objects may not be possible. For these reasons, for that kind of granules, we have to use more general classifiers, some of which are described in the subsection 4.4.

The procedure

1. The original decision system has been input (U,A,d).
2. We create the table $max_{training\ set}a$ and $min_{training\ set}a$ of the original system, for $\forall a \in A$.
   We granulate at fixed parameter ε_{gran} as following.
3. The training decision system from point 1 is granulated by means of ε - granulation method with the radius r_{gran} and for the parameter ε_{gran}, a granular object is created by averaging the granule attributes. The decision value is assigned based on the MOM method, where after calculating the average value of decisions, we choose the closest existing decision value. The decision value is decided by the majority voting method.

   Pseudo-code

   Granule formation
   Input:
   decision system (U,A,d)
   granulation radius r_{gran}
   granulation epsilon ε_{gran}
   for each u in U **do**
   for each v in U **do**
   let $IND_{\varepsilon_{gran}}(u,v) = \{a \in A : \frac{|a(u)-a(v)|}{|max_{training\ set}a - min_{training\ set}a|} \leq \varepsilon_{gran}\}$
   if $\frac{|IND_{\varepsilon_{gran}}(u,v)|}{|A|} \geq r_{gran}$ **then** $g_{r_{gran},\varepsilon_{gran}}(u) \leftarrow g_{r_{gran},\varepsilon_{gran}}(u) \cup v$
   return $\{g_{r_{gran},\varepsilon_{gran}}(u) :\ u \in U\}$

   Search for granular covering U
   Input:
   $G = \{g_{r_{gran},\varepsilon_{gran}}(u) :\ u \in U\}$
   Procedure *random*
   Variable:
   $U_{cover} \leftarrow \{\}$
   $temp \leftarrow G$
   loop:
   $g \leftarrow random(temp)$
   if $g \subseteq U_{cover}$ **then** $temp \leftarrow temp$ - g
   else $U_{cover} \leftarrow U_{cover} \cup g$
   if $U_{cover} = U$ **then break**

Granular object creation
Input:
Procedure MOM
Variable:
GS $\leftarrow$ { }
for each g in U_{cover} **do**
$gs \leftarrow$ MOM (g)
$GS \leftarrow GS \cup gs$
return GS

The obtained set of objects GS is named as the granular reflection of the system (U,A,d).

4.3.6 Concept Dependent ε Granulation

Similarly as before, the key point of this method is granulation around decision values, where granules look as follows,

$$g^{cd}_{r_{gran}}(u) = \{v : (\frac{|IND(u,v)|}{|A|} \geq r_{gran}) \wedge (d(u) = d(v))\}. \tag{4.28}$$

For $\varepsilon-$ granular decision systems, we have to design appropriate classifiers, previously mentioned discrete classifiers do not work effectively on this kind of granules. A good solution is to use methods which can classify with some fixed indiscernibility ratio. We have to be aware that this kind of rough classifiers cannot be used for symbolic decision systems. In the next subsection, we will describe an exemplary method of classification which works well with the ε variants of granular decision systems.

4.4 Exemplary ε - Classification

The procedure

1. The training and test decision systems have been input.
2. We create the table $max_{training\ set}a$ and $min_{training\ set}a$ from the training decision system, for $\forall a \in A$.
3. Outliers of the original test table are mapped on the interval $min_{training\ set}a$, $max_{training\ set}a$

   We granulate the training decision systems at fixed ε_{gran} and r_{gran} parameters, and we obtain the granular decision system GS which is a granular reflection of the original system (U,A,d).

4. The descriptors indiscernibility ratio $\varepsilon_{class} \in \{0, 0.01, \ldots, 1.0\}$ has been fixed.
5. We classify chosen object u by means of granule g obtained from the decision system in point 4. We compare granular objects with a test object and check whether all attributes of the considered granule catch descriptors of test objects to a degree of ε_{class}. For all granules g we check whether

$$\frac{|a(u) - a(g)|}{|max_{training\ set}a - min_{training\ set}a|} \leq \varepsilon_{class}, \text{ for } a \in A. \tag{4.29}$$

   If after examination of all conditional descriptors, $|IND_{\varepsilon_{klas}}(u, g)| = |A|$, it means that the granule g catches test object and the decisive parameter param(decision concept of granule g) is incremented.
6. After classification of test object u by means of all granules, parameters of classification vote on decision according to

$$\frac{param(decision\ concept\ of\ training\ set)}{|\{v : d(v) = decision\ concept\ of\ training\ set\}|}. \tag{4.30}$$

   The test object u gets decision value from the training concept which has the highest value of parameter param. Such granular classification is supported by the training decision system from point 1 by $max_{training\ set}a$ and $min_{training\ set}a$. During the classification process, the size of training concepts is used for the normalization of parameters deciding on classification. We use those values because if we would compute $max_{training\ set}a$ and $min_{training\ set}a$ from granular decision system, for small granulation radii r_{gran} there could be a high disturbance of test systems during outlier mapping.

4.4.1 Exemplary Results for Combination of ε - Granulation and Classification

To prove that our ε classifier works well with ε granules, we have carried out exemplary experiments on real data from UCI Repository. We have applied Cross Validation 5 method (CV5), when the training decision systems from all folds of CV5 were granulated and used for classification of particular test sets. The results of classification are shown in Tables 4.1 – 4.3. It is obvious that it is difficult to find optimal parameters of classification for this kind of method. In this example we show some exemplary parameters, which have been chosen based on degree of classification accuracy. As we can see, ε granules are useful and seem to be effective with the considered classification method, in particular, they reduce the size of the original training systems.

Table 4.1 5-fold C-V; Australian Credit; Classification based on ε - granules for $r_{gran} = 0.785714$. $r_{gran} = granular\ radius$, $\varepsilon_{gran} = epsilon\ of\ granulation$, $\varepsilon_{class} = optimal\ epsilon\ of\ classification$, $acc = accuracy$, $cov = coverage$, $m\_trn = average\ size\ of\ training\ set$

r_{gran}	ε_{gran}	ε_{class}	acc	cov	m_trn
nil	*nil*	*nil*	0.845	1.0	552
0.785714	0.0	0.55	0.861673	0.995652	533.4
0.785714	0.01	0.57	0.863133	0.995652	427.6
0.785714	0.02	0.76	0.863345	0.997102	332.2
0.785714	0.03	0.83	0.851909	0.998551	280.0
0.785714	0.04	0.78	0.864826	0.997102	232.4
0.785714	0.05	0.71	0.858754	0.995652	193.0
0.785714	0.06	0.75	0.844462	0.997102	161.2
0.785714	0.07	0.78	0.844081	0.984058	145.8
0.785714	0.08	0.92	0.838877	0.998551	111.2
0.785714	0.09	0.78	0.835978	0.998551	99.4
0.785714	0.1	0.87	0.837406	0.998551	87.6
0.785714	0.11	0.89	0.847625	0.998551	80.4
0.785714	0.12	0.9	0.828721	0.998551	75.4
0.785714	0.13	0.79	0.843579	0.985507	61.8
0.785714	0.14	0.9	0.830213	0.998551	56.2
0.785714	0.15	0.68	0.833055	0.989855	53.8
0.785714	0.16	0.79	0.821411	0.998551	43.8
0.785714	0.17	0.69	0.830808	0.971015	44.2
0.785714	0.18	0.9	0.830181	0.998551	40.2
0.785714	0.19	0.74	0.818114	0.986957	38.4
0.785714	0.2	0.94	0.803988	0.998551	33.8

Table 4.2 5-fold C-V; Heart Disease; Classification based on ε - granules for $r_{gran} = 0.769231$. $r_{gran} = granular\ radius$, $\varepsilon_{gran} = epsilon\ of\ granulation$, $\varepsilon_{class} = optimal\ epsilon\ of\ classification$, $acc = accuracy$, $cov = coverage$, $m\_trn = average\ size\ of\ training\ set$

r_{gran}	ε_{gran}	ε_{class}	acc	cov	m_trn
nil	*nil*	*nil*	0.804	1.0	216
0.769231	0.0	0.88	0.790172	0.97037	209.8
0.769231	0.01	0.88	0.790172	0.97037	203.8
0.769231	0.02	0.75	0.797864	0.97037	195.8
0.769231	0.03	0.99	0.786398	0.97037	184.6
0.769231	0.04	0.99	0.790172	0.97037	172.2
0.769231	0.05	0.91	0.783315	0.974074	153.2
0.769231	0.06	0.69	0.785014	0.966666	144.6
0.769231	0.07	0.97	0.794969	0.992592	128.0
0.769231	0.08	0.97	0.781482	1.0	120.2
0.769231	0.09	0.95	0.784207	0.996296	110.4
0.769231	0.1	0.93	0.7826	0.988889	99.2
0.769231	0.11	0.92	0.810273	0.996296	90.0
0.769231	0.12	0.88	0.802236	0.992592	83.6
0.769231	0.13	0.92	0.796296	1.0	75.6
0.769231	0.14	0.91	0.788889	1.0	71.0
0.769231	0.15	0.9	0.803145	0.996296	65.0
0.769231	0.16	0.9	0.785185	1.0	61.8
0.769231	0.17	0.93	0.788889	1.0	56.8
0.769231	0.18	0.91	0.777778	1.0	52.8
0.769231	0.19	0.88	0.788889	1.0	50.2
0.769231	0.2	0.89	0.798532	0.992592	46.2

Table 4.3 5-fold C-V; Pima Indians Diabetes; Classification based on ε - granules for $r_{gran} = 0.625$. $r_{gran} = granular\ radius$, $\varepsilon_{gran} = epsilon\ of\ granulation$, $\varepsilon_{class} = optimal\ epsilon\ of\ classification$, $acc = accuracy$, $cov = coverage$, $m\_trn = average\ size\ of\ training\ set$

r_{gran}	ε_{gran}	ε_{class}	acc	cov	m_trn
nil	*nil*	*nil*	0.6528	0.9972	615
0.625	0.0	0.3	0.742296	0.989542	607.4
0.625	0.01	0.3	0.741041	0.989542	576.0
0.625	0.02	0.3	0.733645	0.986928	452.0
0.625	0.03	0.3	0.724776	0.983006	357.8
0.625	0.04	0.28	0.739153	0.956862	249.8
0.625	0.05	0.31	0.719509	0.946405	174.0
0.625	0.06	0.32	0.718681	0.933333	111.2
0.625	0.07	0.71	0.654902	1	83.8
0.625	0.08	0.68	0.656132	0.996078	66.0
0.625	0.09	0.52	0.660574	0.989542	49.2
0.625	0.1	0.8	0.653595	1	41.0
0.625	0.11	0.58	0.657892	0.993464	33.8
0.625	0.12	0.61	0.660965	0.994771	24.6
0.625	0.13	0.61	0.659661	0.994771	23.4
0.625	0.14	0.72	0.653604	0.996078	20.0
0.625	0.15	0.8	0.650981	1	16.2
0.625	0.16	0.69	0.654386	0.994771	13.8
0.625	0.17	0.87	0.649673	1	12.0
0.625	0.18	0.87	0.649673	1	11.2
0.625	0.19	0.86	0.649673	1	9.2
0.625	0.2	0.85	0.649673	1	8.4

4.5 Conclusions

Our considerations lead to the conclusion that an important element for the proper use of approximation methods is choosing the correct methods of classification, which will work effectively with them. In general, for discrete methods it is better to use classification algorithms which are focused on measuring similarity on symbolic or integer stable data (without any changes in the domain). We show that the described methods, which form rough granules, sometimes changing the values of domain attribute by averaging values, can be applied effectively with rough classifiers.

The need of analyzing a rising amount of information, sooner or later forces us to focus on some approximation methods which effectively maintain knowledge, significantly lowering size of data. For this reason, at the very least, granular rough computing is a promising field of research and worth trying to develop.

One of our future purposes, in addition to theoretical development of granular rough computing paradigms, is the application of granular methods in practice - especially in the decision modules of autonomous mobile robots.

Acknowledgements. This work is dedicated as homage to the memory of Professor Zdzisław Pawlak. The author is indebted to Professor Lech Polkowski for comments on this text.

References

1. Arnold, V.I.: On functions of three variables. Dokl. Akad. Nauk 114, 679–681 (1957); English transl., Amer. Math. Soc. Transl. 28, 51–54 (1963)
2. Artiemjew, P.: On strategies of knowledge granulation and applications to decision systems. PhD Dissertation, Polish Japanese Institute of Information Technology, L. Polkowski, Supervisor, Warsaw (2009)
3. Artiemjew, P.: Rough Mereological Classifiers Obtained from Weak Variants of Rough Inclusions. In: Wang, G., Li, T., Grzymala-Busse, J.W., Miao, D., Skowron, A., Yao, Y. (eds.) RSKT 2008. LNCS (LNAI), vol. 5009, pp. 229–236. Springer, Heidelberg (2008)
4. Artiemjew, P.: On Classification of Data by Means of Rough Mereological Granules of Objects and Rules. In: Wang, G., Li, T., Grzymala-Busse, J.W., Miao, D., Skowron, A., Yao, Y. (eds.) RSKT 2008. LNCS (LNAI), vol. 5009, pp. 221–228. Springer, Heidelberg (2008)
5. Artiemjew, P.: Natural versus Granular Computing: Classifiers from Granular Structures. In: Chan, C.-C., Grzymala-Busse, J.W., Ziarko, W.P. (eds.) RSCTC 2008. LNCS (LNAI), vol. 5306, pp. 150–159. Springer, Heidelberg (2008)
6. Artiemjew, P.: Classifiers from granulated data sets: Concept dependent and layered granulation. In: Proceedings RSKD 2007. Workshop at ECML/PKDD 2007, pp. 1–9. Warsaw University Press, Warsaw (2007)
7. Bocheński, J. M.: Die Zeitgenössischen Denkmethoden. A. Francke AG Verlag, Bern (Swiss Fed.) (1954)
8. Hájek, P.: Metamathematics of Fuzzy Logic. Kluwer, Dordrecht (1998)
9. Kolmogorov, A.N.: Representation of functions of many variables. Dokl. Akad. Nauk 114, 953–956 (1957); English transl., Amer. Math. Soc. Transl. 17, 369–373 (1961)
10. Leśniewski, S.: Podstawy ogólnej teoryi mnogosci (On the foundations of set theory, in Polish). The Polish Scientific Circle, Moscow (1916); see also a later digest: Topoi 2, 7–52 (1982), and Foundations of the General Theory of Sets. I. In: Surma, S.J., Srzednicki, J., Barnett, D.I., Rickey, F.V. (eds.) S. Lesniewski. Collected Works, vol. 1, pp. 129-173. Kluwer, Dordrecht (1992)
11. Ling, C.-H.: Representation of associative functions. Publ. Math. Debrecen 12, 189–212 (1965)
12. Marcus, S.: Tolerance rough sets, Cech topology, learning processes. Bulletin of the Polish Acad. Sci., Technical Sci. 42(3), 471–478 (1994)
13. Nieminen, J.: Rough tolerance equality. Fundamenta Informaticae 11, 289–294 (1988)
14. Pawlak, Z.: Rough sets. Int. J. Computer and Information Sci. 11, 341–356 (1982)
15. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning about Data. In: System Theory, Knowledge Engineering and Problem Solving, vol. 9. Kluwer Academic Publishers, Dordrecht (1991)
16. Pawlak, Z., Skowron, A.: Rough membership functions. In: Yager, R.R., Fedrizzi, M., Kasprzyk, J. (eds.) Advances in the Dempster–Shafer Theory of Evidence, pp. 251–271. John Wiley and Sons, New York (1994)
17. Pawlak, Z., Skowron, A.: Rough sets: Some extensions. Information Sciences 177(1), 28–40 (2007)
18. Poincare, H.: Science et Hypothese, Paris (1905)
19. Polkowski, L., Skowron, A., Żytkow, J.: Tolerance based rough sets. In: Lin, T.Y., Wildberger, M. (eds.) Soft Computing: Rough Sets, Fuzzy Logic, Neural Networks, Uncertainty Management, Knowledge Discovery, pp. 55–58. Simulation Councils Inc., San Diego (1994)
20. Polkowski, L., Skowron, A.: Rough Mereology. In: Raś, Z.W., Zemankova, M. (eds.) ISMIS 1994. LNCS (LNAI), vol. 869, pp. 85–94. Springer, Heidelberg (1994)
21. Polkowski, L., Skowron, A.: Rough mereology: A new paradigm for approximate reasoning. International Journal of Approximate Reasoning 15(4), 333–365 (1997)

22. Polkowski, L.: A Rough Set Paradigm for Unifying Rough Set Theory and Fuzzy Set Theory (a plenary lecture). In: Wang, G., Liu, Q., Yao, Y., Skowron, A. (eds.) RSFDGrC 2003. LNCS (LNAI), vol. 2639, pp. 70–78. Springer, Heidelberg (2003); cf. also Fundamenta Informaticae 54, 67–88 (2003)
23. Polkowski, L.: Toward Rough Set Foundations. Mereological Approach (a plenary lecture). In: Tsumoto, S., Słowiński, R., Komorowski, J., Grzymała-Busse, J.W. (eds.) RSCTC 2004. LNCS (LNAI), vol. 3066, pp. 8–25. Springer, Heidelberg (2004)
24. Polkowski, L.: Formal granular calculi based on rough inclusions (a feature talk). In: Proceedings of the 2006 IEEE Int. Conference on Granular Computing, GrC 2006, pp. 57–62. IEEE Computer Society Press (2006)
25. Polkowski, L.: Granulation of Knowledge in Decision Systems: The Approach Based on Rough Inclusions. The Method and Its Applications. In: Kryszkiewicz, M., Peters, J.F., Rybiński, H., Skowron, A. (eds.) RSEISP 2007. LNCS (LNAI), vol. 4585, pp. 69–79. Springer, Heidelberg (2007)
26. Polkowski, L.: The paradigm of granular rough computing. In: Proceedings of the 6th IEEE Intern. Conf. on Cognitive Informatics (ICCI 2007), pp. 145–163. IEEE Computer Society Press, Los Alamitos (2007)
27. Polkowski, L.: A unified approach to granulation of knowledge and granular computing based on rough mereology: A Survey. In: Pedrycz, W., Skowron, A., Kreinovich, V. (eds.) Handbook of Granular Computing, pp. 375–401. John Wiley & Sons, New York (2008)
28. Polkowski, L., Artiemjew, P.: On Classifying Mappings Induced by Granular Structures. In: Peters, J.F., Skowron, A., Rybiński, H. (eds.) Transactions on Rough Sets IX. LNCS, vol. 5390, pp. 264–286. Springer, Heidelberg (2008)
29. Polkowski, L., Artiemjew, P.: A Study in Granular Computing: On Classifiers Induced from Granular Reflections of Data. In: Peters, J.F., Skowron, A., Rybiński, H. (eds.) Transactions on Rough Sets IX. LNCS (LNAI), vol. 5390, pp. 230–263. Springer, Heidelberg (2008)
30. Polkowski, L.: On the Idea of Using Granular Rough Mereological Structures in Classification of Data. In: Wang, G., Li, T., Grzymala-Busse, J.W., Miao, D., Skowron, A., Yao, Y. (eds.) RSKT 2008. LNCS (LNAI), vol. 5009, pp. 213–220. Springer, Heidelberg (2008)
31. Polkowski, L.: Granulation of knowledge: Similarity based approach in information and decision systems. In: Meyers, R. (ed.) Encyclopedia of Complexity and System Sciences, article 00788. Springer, Heidelberg (2009)
32. Słowiński, R., Vanderpooten, D.: Similarity relation as a basis for rough approximations. In: Wang, P.P. (ed.) Advances in Machine Intelligence & Soft-Computing, vol. IV, pp. 17–33. Bookwrights, Raleigh (1997)
33. Skowron, A., Rauszer, C.: The discernibility matrices and functions in information systems. In: Słowiński, R. (ed.) Intelligent Decision Support. Handbook of Applications and Advances of Rough Set Theory, pp. 331–362. Kluwer Academic Publishers, Dordrecht (1992)
34. Skowron, A.: Boolean Reasoning for Decision Rules Generation. In: Komorowski, J., Raś, Z.W. (eds.) ISMIS 1993. LNCS, vol. 689, pp. 295–305. Springer, Heidelberg (1993)
35. Skowron, A.: Extracting laws from decision tables. Computational Intelligence. An International Journal 11(2), 371–388 (1995)
36. Skowron, A., Stepaniuk, J.: Generalized approximation spaces. In: Lin, T.Y., Wildberger, A.M. (eds.) The Third International Workshop on Rough Sets and Soft Computing Proceedings (RSSC 1994), San Jose, California, USA, November 10-12, pp. 56–163. San Jose State University (1994); see also: Skowron, A., Stepaniuk, J.: Generalized approximation spaces. ICS Research Report **41**/94, Warsaw University of Technology (1994); see also Skowron, A., Stepaniuk, J.: Generalized approximation spaces. In: Lin, T.Y., Wildberger, A.M. (eds.) Soft Computing, pp. 18–21. Simulation Councils, Inc., San Diego (1995)

37. Skowron, A., Stepaniuk, J.: Tolerance approximation spaces. Fundamenta Informaticae 27(2/3), 245–253 (1996)
38. Stepaniuk, J.: Rough - Granular Computing in Knowledge Discovery and Data Mining. Springer, Heidelberg (2008)
39. Zadeh, L.A.: Fuzzy sets and information granularity. In: Gupta, M., Ragade, R., Yager, R.R. (eds.) Advances in Fuzzy Set Theory and Applications, pp. 3–18. North Holland, Amsterdam (1979)

Chapter 5
Game-Theoretic Rough Sets for Feature Selection

Nouman Azam and JingTao Yao

Abstract. Feature selection plays an important role in text categorization. Term frequency and document frequency are commonly used measures in feature selection methods for text categorization. The term frequency provides document level information for a word while document frequency highlights dataset level information for a word. We introduced a Game-theoretic rough set based method for combining these measures in an effective and meaningful way. The method incorporates the measures as players in a game where each player employs a three-way decision in selecting features. The three-way decisions for features received inspiration from three-way decisions for classification of objects in rough sets. The selected decisions with respective measures are utilized in finding a corporative solution as in game-theoretic rough sets. A demonstrative example suggests that this method may be more efficient for feature selection in text categorization.

Keywords: Feature selection, game-theoretic rough sets, text categorization, three-way decisions

5.1 Introduction

Feature selection is a process of selecting a subset of features that is considered as important [10, 14, 15, 24, 32]. It remains as an effective technique in many applications, such as text categorization and bioinformatics [3, 4, 22, 23]. Feature selection techniques are helpful in various situations for these applications, such as high imbalance in data, limited availability of resources and data populated with noise [1, 5, 11, 35].

Feature selection methods in text categorization commonly employ term frequency and document frequency measures in calculating a feature's utility. Term frequency is the number of times a particular word appears in a given document

Nouman Azam and JingTao Yao
Department of Computer Science, University of Regina, Regina, Saskatchewan, Canada S4S 0A2
e-mail: {azam200n,jtyao}@cs.uregina.ca

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 61–78.
springerlink.com

while document frequency is the count of documents containing that word [2]. The two measures may be considered as complimentary as they provide information from different perspectives. Term frequencies provide information at a local level, comprising of an individual document. The document frequencies in contrast, provide information at a global level consisting of entire document set. The two measures may be considered as individually important as jointly they provide useful information which may be utilized in effective selection of features.

Existing feature selection methods are either based on term frequency or document frequency [5, 12, 16–18, 20, 25, 28]. This suggests that these methods are using limited information for calculating utility of a feature. The term frequency based methods ignore the document frequency information while the document frequency based methods ignore the term frequency. This may affect the ranking of features with respective methods. However, both measures may be used for effective ranking and selecting important features.

Table 5.1 Term frequency and document frequency of words

	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9
Document Frequency	75	70	65	59	55	50	45	10	10
Term Frequency Per Doc.	2	7	3	2	4	10	4	5	1

We demonstrate a problem in ranking the features with an example. Suppose that a dataset with nine words represented as $w_1, w_2, ..., w_9$ is considered. Table 5.1 shows the document frequencies and term frequencies of these words. We examine a couple of feature selection methods. Document frequency thresholding (DFT) [28] is chosen as document frequency based method while term frequency thresholding (TFT) [17] as term frequency based method. Feature ranks with DFT method can be calculated by sorting the words on their document frequencies. Obtaining ranks with TFT will require the total term frequency of each word across all documents which can be obtained by multiplying the term frequency per document by document frequency. Feature rankings with respective methods may be calculated as follows.

Ranking with DFT: $\{ w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8, w_9 \}$.
Ranking with TFT: $\{ w_6, w_2, w_5, w_3, w_7, w_1, w_4, w_8, w_9 \}$.

Words with the highest document frequency and term frequency receive the highest rank in respective methods. We note that the words are ranked differently with different methods. If we are interested in selecting four words, then DFT would select w_1, w_2, w_3 and w_4 while TFT would select w_6, w_2, w_5 and w_3. The word w_1 has the highest document frequency while w_6 has the highest term frequency. We expect relatively good ranks for both of these words regardless of the methods. However, w_1 is neither selected nor ranked appropriately by TFT. A similar situation is observed

for w_6 with DFT. This suggests that DFT ignores some term frequent information while TFT ignores some document frequent information. It is plausible to consider both of the measures for effective selection of features in this case. A method that incorporates the two measures is desired for such a purpose.

Suppose that we introduce a new method that combines the rankings of the two methods by calculating an average. The new rankings for words may be obtained as,

$$\{ w_2, [w_1, w_6, w_3], w_5, [w_7, w_4], w_8, w_9 \},$$

where words in square brackets denote equal ranks. We may select important features based on the new ranking. If we consider selection of four words again, w_2, w_1, w_6 and w_3 would be selected. The words w_1 and w_6, which were ignored earlier are now being selected. This means that the words which have higher term frequency value or document frequency value are now being selected.

A particular measure may consider a feature as important, relatively important or unimportant for selection. This intuitively suggests three-way decisions for features similar to three-way decisions for classification of objects in rough sets. Particularly, the important features may be considered as objects being positively identified for a set while unimportant features may be considered as objects negatively identified for a set. The features that are relatively important may be treated as objects associated with uncertain decisions. We try to find a systematic method which incorporates these decisions of measures in a unified framework for selecting a feature. A game-theoretic rough set (GTRS) based method is proposed for such a purpose.

The GTRS model is a recent extension to rough set theory for analyzing and making intelligent decisions [6–8, 21, 29, 30]. The model utilizes ideas from game theory [19] to form a systematic method in analyzing decision problems. We examine the model in combining measures for feature selection. In particular, the model was used for formulating a game between the measures of term frequency and document frequency. Each measure analyzes a feature for its importance using suitable payoff functions and may choose from three possible actions: accept (or select), reject and abstain. The actions of respective measures were used in finding a cooperative decision on selecting a feature.

This chapter is organized as follows: Section 5.2 reviews some background information regarding the GTRS model. Section 5.3 presents the proposed feature selection method, formulated with the model. Section 5.4 demonstrates the application of proposed method for text categorization with an example. Section 5.5 concludes the chapter with directions for future research.

5.2 Game-Theoretic Rough Sets

Game-theoretic rough sets model is a new extension to traditional rough sets. The model provides formulation for decision making problems in the aim of finding an effective and efficient solution, obtained with analysis of conflict or

cooperation among the measures. A game is formulated among the measures that seek to achieve dominance over others or tries teaming up with others in finding an optimal position. The basic components in GTRS formulation includes, information of measures considered as players in the game, the strategies or available actions for each player, and utility or payoff function for each action.

Yao and Herbert suggested the model for analyzing a major problem in probabilistic rough sets which can found in references [8, 31]. A method for decreasing the size of boundary region through further explorations of the data was proposed. A game was proposed between classification measures with actions corresponding to adjusting risk functions in order to increase the overall classification effectiveness of the system. New regions were obtained with modified risk functions. The model was also investigated in other studies [6, 7, 29, 30].

The GTRS model utilizes ideas from game theory [19] for analyzing decision problems with rough sets. A single game is defined as $G = \{P, S, F\}$, where $P = \{p_1, p_2, ..., p_n\}$ represents a player set, $S = \{a_1, a_2, ..., a_m\}$ an action or strategy set for a player, and $F = \{u_1, u_2, ..., u_m\}$ the respective payoff functions for each action. Each player chooses an action from S according to its expected payoff function in F.

A player set P contains measures in GTRS formulation. Each measure highlights an aspect of interest in a decision problem. The selection of measures depends on the desired aims and objectives that we may wish to achieve. For instance, in probabilistic rough sets, we may want to increase the overall classification ability of the system or obtaining optimal values for region parameters may be of interest. The former objective may be achieved with classification measures in a game while the later may be reached with region parameters.

The GTRS formulates strategies for each measure in order to make their effective participation in a game. We may observe multiple parameters or variables in decision problems that affect decision making in different ways. For instance, changing the values of loss functions in decision theoretic rough sets will result in different regions with different decisions on object classification [34]. A change to such variables may affect the measures differently. In particular, a measure may experience a gain, i.e. an increase in its value, in response to a particular change in a variable value. Different level of gains may be experienced by selecting different variables with their respective changes. This suggests that a particular measure may attain a certain level of gain by selecting a suitable variable with its associated change. A variable along with a change may be realized as an action or strategy for a particular

Table 5.2 Action scenario for a measure

Action	Description	Method	Desired conditions	Outcomes or influences
	What the action does?	How to carry out the action?	When the action is desired?	Possible effects on problem
a_j	....	....	....	....
....	....	....	....	....

measure in obtaining desired performance gain. A gain obtained by a measure in selecting an action may be considered as payoff or utility for that measure.

Each action is desired by a player in a specific situation of a game. The GTRS constructs action scenarios for analyzing individual actions with description of a corresponding desired situation. Table 5.2 presents the general structure for an action scenario. Each row in the table represents a particular action by providing its associated description, method for execution, desired conditions and possible outcomes or influences on the problem. This means that an action scenario represents the problem from a single measure perspective.

The objective of GTRS is to assist in analysis of competing or cooperating measures for intelligent decision making. This is facilitated with payoff tables which lists all possible strategies with their respective payoffs. A payoff or utility reflects a player's happiness in performing an action. The payoff tables are examined with equilibrium analysis for a possible solution. This means that we try to find a solution where each measure has maximized its payoff giving their opponents chosen actions. The solution obtained with equilibrium analysis is considered as an optimal solution for a problem.

5.3 Feature Selection with Game-Theoretic Rough Set

We will examine a GTRS based method for combining the measures in feature selection. In order to analyze problems with GTRS, we need to formulate them. We will start the formulation by identifying the main components of the GTRS model, i.e. a set of players, a set of strategies for each player and a set of payoff functions for respective actions.

5.3.1 Components

We will present the GTRS components in this section. The components are briefly explained and discussed in Section 5.2.

The Player Set: A player set in GTRS refers to the measures which may be analyzed for competition or cooperation. We consider two measures, i.e. TF and DF in this chapter. The player set is denoted as $P = \{TF, DF\}$, where TF represents the measure of term frequency and DF the document frequency. Each measure will analyze the importance of a feature with its respective payoff functions. The measures will effectively participate in a game for reaching a final decision on selection of a feature.

The Strategy Sets: Each player is expected to have a set of possible strategies. Individual strategies may be realized as actions when performed. We have two sets

of strategies with actions corresponding to the two measures. The strategy sets are represented as S^{TF} and S^{DF} for measures TF and DF, respectively. These sets are denoted as $S^{TF} = \{a_1, a_2, ..., a_n\}$ and $S^{DF} = \{a_1, a_2, ..., a_n\}$. This means that each measure can have n actions. Let's simplify this scenario with three actions, i.e. a_s, a_a and a_r.

a_s is the action of accepting or selecting a feature.
a_a is the action of abstaining in making a definite decision.
a_r is the action of rejecting a feature.

The Payoff Functions: A payoff function represents motivation level of a player towards an action. The set of all payoff functions in a game may be represented as F. We formulated $F = \{u_{TF}, u_{DF}\}$, where u_{TF} and u_{DF} represent the sets of payoff functions for measures TF and DF, respectively. Each measure may choose from three possible actions as discussed above. This suggests that for a given action of a measure, its opponent may choose from three possible actions. We need to define nine payoff functions for each measure in this case. Let us denote the payoff of measure i, performing an action a_j, given action a_k of his opponent as $u_{i(a_j|a_k)}$. The payoff set for a measure m_i has nine values of the form $u_{i(a_j|a_k)}$, where $i \in \{TF, DF\}$ and $\{a_j, a_k\} \in \{a_s, a_a, a_r\}$. The two payoff sets are given as follows.

$$u_{TF} = \{u_{TF(a_s|a_s)}, \quad u_{TF(a_s|a_a)}, \quad u_{TF(a_s|a_r)}, \quad u_{TF(a_a|a_s)}, \quad u_{TF(a_a|a_a)}, \quad u_{TF(a_a|a_r)}$$
$$u_{TF(a_r|a_s)}, \quad u_{TF(a_r|a_a)}, \quad u_{TF(a_r|a_r)}\}$$

$$u_{DF} = \{u_{DF(a_s|a_s)}, \quad u_{DF(a_s|a_a)}, \quad u_{DF(a_s|a_r)}, \quad u_{DF(a_a|a_s)}, \quad u_{DF(a_a|a_a)}, \quad u_{DF(a_a|a_r)}$$
$$u_{DF(a_r|a_s)}, \quad u_{DF(a_r|a_a)}, \quad u_{DF(a_r|a_r)}\}$$

We now define individual payoff functions. Let us consider probabilities of term frequency and document frequency. The probability of term frequency for a word, i.e. $P_{TF}(w)$, refers to its relative occurrence within a document while probability of document frequency, i.e. $P_{TF}(w)$, refers to its relative frequency with respect to documents set. The two probabilities for a particular word w may be defined as follows.

$P_{TF}(w)$ = Term frequency of w in a document / total words in that document.
$P_{DF}(w)$ = Document frequency of w / total documents in the data set.

Each measure will use its respective probability in analyzing available actions.

We utilize an approach inspired from decision-theoretic rough set model (DTRS) in making decisions with probabilities [9, 13, 27, 34, 36]. The model utilizes a threshold pair, i.e. α and β representing desired levels of precision in classifying objects [33]. The thresholds are defined with Bayesian decision-theoretic analysis using notions of risks and losses. A three-way decision for classifying objects are obtained with these thresholds. Particularly, an object is accepted as belonging to a particular set if its conditional probability with the set is greater than

or equal to threshold α. If the conditional probability is lesser than or equal to threshold β, the object is rejected for the set. When the conditional probability is between the two thresholds, a definite decision for object is deferred or abstained.

We adopt a similar idea in making decisions for features. A threshold pair was defined for each probability. The thresholds α_{TF} and β_{TF} are used for term frequency probability while α_{DF} and β_{DF} for document frequency probability. Three-way decisions were defined with each probability as in DTRS. This means that a particular measure m_i may decide to accept (select) a word w, if the probability P_{m_i} (i.e. the probability corresponding to the measure m_i) is greater than or equal to α_{m_i}. If the probability for a measure is lesser than or equal to β_{m_i}, the measure may decide to reject the word. In case when the probability is between the two thresholds, a measure may decide to abstain from making a definite decision.

Table 5.3 Action scenarios for measures

Player	Action	Description	Desired Conditions	Outcome
TF	a_s	Select	$P_{TF}(w) \geq \alpha_{TF}$	Select (accept) the feature
	a_a	Abstain	$\beta_{TF} < P_{TF}(w) < \alpha_{TF}$	Abstain from decision
	a_r	Reject	$P_{TF}(w) \leq \beta_{TF}$	Reject the feature
DF	a_s	Select	$P_{DF}(w) \geq \alpha_{DF}$	Select (accept) the feature
	a_a	Abstain	$\beta_{DF} < P_{DF}(w) < \alpha_{DF}$	Abstain from decision
	a_r	Reject	$P_{DF}(w) \leq \beta_{DF}$	Reject the feature

Table 5.3 presents the action scenarios for measures. There are two major rows in the table corresponding to measures TF and DF, respectively. Each major row contains three sub rows that represents the individual actions for a measure with corresponding conditions and an expected outcome. For instance, the first sub row corresponding to measure TF describes the action of selecting a feature as discussed above.

The probabilities and thresholds may be used to define three functions for each measure m_i ($m_i \in \{TF, DF\}$), corresponding to its three actions.

$$u_{m_i(a_s)} = \begin{cases} 1 & P_{m_i}(w) \geq \alpha_{m_i}, \\ 0.5 & \beta_{m_i} < P_{m_i}(w) < \alpha_{m_i}, \\ 0 & P_{m_i}(w) \leq \beta_{m_i} \end{cases} \tag{5.1}$$

$$u_{m_i(a_a)} = \begin{cases} 0 & P_{m_i}(w) \geq \alpha_{m_i}, \\ 1 & \beta_{m_i} < P_{m_i}(w) < \alpha_{m_i}, \\ 0 & P_{m_i}(w) \leq \beta_{m_i} \end{cases} \tag{5.2}$$

$$u_{m_i(a_r)} = \begin{cases} 0 & P_{m_i}(w) \geq \alpha_{m_i}, \\ 0.5 & \beta_{m_i} < P_{m_i}(w) < \alpha_{m_i}, \\ 1 & P_{m_i}(w) \leq \beta_{m_i} \end{cases} \tag{5.3}$$

The function $u_{m_i(a_s)}$ represents the utility for a measure in accepting a feature. When the value for a word is greater than or equal to threshold α_{m_i}, the function returns a maximum value of 1. This means that the action of accepting or selecting the word is highly desired by the measure. When the value for a word is lesser than the threshold β_{m_i}, the function returns a minimum value of 0. This suggests that the action of selecting the feature is least desired in this case. A value of 0.5 is returned when the value for a feature is between the two thresholds. This represents an uncertain situation for choosing the action. A similar interpretation may be used for the functions $u_{m_i(a_a)}$ and $u_{m_i(a_s)}$.

The above functions describe the utilities for measures when considered in isolation. An interaction among the measures is expected during a game. Individual beliefs and utilities of players are affected by selected actions of their opponents. To reflect this, we utilized an average of utilities corresponding to players in obtaining the payoff functions. This was a reasonable choice as we seek cooperation among the measures in finding a feature usefulness or importance level. Furthermore, the opinions of both measures were considered as useful without discrimination. This means that the payoff for player i in performing action a_j given action a_k of his opponent l is given by $u_{i(a_j|a_k)} = \{u_{i(a_j)} + u_{l(a_k)}\}/2$, where $\{i,l\} \in \{TF, DF\}$ and $\{a_j, a_k\} \in \{a_s, a_a, a_r\}$.

5.3.2 Implementing Competition

We finally express a game between the measures TF and DF in a payoff table. This is presented in Table 5.4. The rows represent the actions of measure

Table 5.4 Payoff table for TF and DF

		DF		
		a_s	a_a	a_r
TF	a_s	$u_{TF(a_s\|a_s)}, u_{DF(a_s\|a_s)}$	$u_{TF(a_s\|a_a)}, u_{DF(a_a\|a_s)}$	$u_{TF(a_s\|a_r)}, u_{DF(a_r\|a_s)}$
	a_a	$u_{TF(a_a\|a_s)}, u_{DF(a_s\|a_a)}$	$u_{TF(a_a\|a_a)}, u_{DF(a_a\|a_a)}$	$u_{TF(a_a\|a_r)}, u_{DF(a_r\|a_a)}$
	a_r	$u_{TF(a_r\|a_s)}, u_{DF(a_s\|a_r)}$	$u_{TF(a_r\|a_a)}, u_{DF(a_a\|a_r)}$	$u_{TF(a_r\|a_r)}$,$u_{DF(a_r\|a_r)}$

TF and columns of measure DF. Each cell in the table represents a payoff pair $< u_{TF(a_i|a_j)}, u_{DF(a_j|a_i)} >$. The pair corresponds to action a_i of TF and a_j of DF. A total of nine payoff pairs are required in this game. The payoffs are calculated with respective payoff functions and the table is populated with these values.

The GTRS utilizes Nash equilibrium [19] for analyzing payoff tables in order to obtain possible outcomes for games. This intuitively suggests that none of the players can be benefited by changing its strategy, given the opponent chosen action. In suggested game between TF and DF, a pair $(u^*_{TF(a_i|a_1)}, u^*_{DF(a_j|a_2)})$ is an equilibrium if for any action a_k, where $k \neq i, j$,

$$u^*_{TF(a_i|a_1)} \geq u_{TF(a_k|a_1)} \quad \text{and} \quad u^*_{DF(a_j|a_2)} \geq u_{DF(a_k|a_2)}. \tag{5.4}$$

The pair $(u^*_{TF(a_i|a_1)}, u^*_{DF(a_j|a_2)})$ is thus the optimal solution in determining actions for the measures.

The actions of measures may be used to define various sets of features. Let us denote the set of features corresponding to actions a_i and a_j for the two measures as $F_{(a_i,a_j)}$. We can obtain the following six sets of features corresponding to various actions of measures (the sets $F_{(a_i,a_j)}$ and $F_{(a_j,a_i)}$ are considered as equal).

$FS^{(a_s,\, a_s)}$, both measures choose a_s.
$FS^{(a_s,\, a_a)}$, one measure choose a_s while the other a_a.
$FS^{(a_s,\, a_r)}$, one measure choose a_s while the other a_r.
$FS^{(a_a,\, a_a)}$, both measures choose a_a.
$FS^{(a_r,\, a_a)}$, one measure choose a_r while the other a_a.
$FS^{(a_r,\, a_r)}$, both measures choose a_r.

Equilibrium analysis may be used to determine the inclusion of a feature into one of these sets. We may define various feature selection approaches with these feature sets. For instance, we may use an aggressive approach, where we select only those features for which both measures agree to select. This means that features with very high importance would be selected. Such selection of features may be used in an application like customer relationship management, where we are interested in identifying the most important features of customer behavior for making effective business decisions [26]. Alternatively, we may define a moderate approach, where we select features when atleast one measure agrees to select. This suggests that this approach selects features that have high importance along with those that are relatively importance. Such selection may be utilized in text categorization, where there are many features with relative importance. Denoting the selected feature sets for aggressive and moderate as F^A and F^M, respectively, we have $F^A = FS^{(a_s,\, a_s)}$ and $F^M = FS^{(a_s,\, a_s)} \bigcup FS^{(a_s,\, a_a)} \bigcup FS^{(a_s,\, a_r)}$, respectively. It may be noted that $|F^A| \leq |F^M|$, i.e. we select less number of features with an aggressive approach.

5.4 A Demonstrative Example

We will demonstrate the application of GTRS based method for feature selection using the example discussed in introduction. Suppose we have a dataset of hundred documents with each document containing one hundred words. The document frequency probability, i.e. $P_{DF}(w)$ and term frequency probability, i.e. $P_{TF}(w)$, may be calculated from Table 5.1 using their respective definitions as explained in Section 5.3. For instance, the $P_{DF}(w_1)$ can be calculated as the document frequency of w_1 divided by total documents, i.e. $75/100 = 0.75$. Similarly, $P_{TF}(w_1)$ may be calculated as $2/100 = 0.2$. Table 5.5 shows the probabilities $P_{DF}(w)$ and $P_{TF}(w)$ for the nine words.

Table 5.5 Probabilities of words

	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9
$P_{DF}(w)$	0.75	0.70	0.65	0.59	0.55	0.50	0.45	0.10	0.10
$P_{TF}(w)$	0.02	0.07	0.03	0.02	0.04	0.10	0.04	0.05	0.01

We define a threshold pair for each measure in order to determine the payoff functions. The thresholds may be calculated in various ways, such as, inspection of feature probabilities in a dataset, user perception of tolerance levels for accepting features, or number of features user wish to select. For considered example in this chapter, we choose the thresholds for TF as [0.06, 0.015] that is $\alpha_{TF} = 0.06$, and $\beta_{TF} = 0.015$ while thresholds for DF as [0.6, 0.15] that is $\alpha_{DF} = 0.6$ and $\beta_{DF} = 0.15$, respectively.

A payoff table for a particular word can be determined by computing all payoff pairs corresponding to individual cells of the table. Considering the payoff table represented in Table 5.4, the payoff pair $< u_{TF(a_s|a_s)}, u_{DF(a_s|a_s)} >$ corresponds to the first cell of the table. Let us calculate this payoff pair for the word w_1. From Table 5.5, we note that $P_{TF}(w_1) = 0.02$ while $P_{DF}(w_1) = 0.75$. We may calculate the utility for a measure m_i in selecting a word by using the function $u_{m_i}(a_s)$ presented in Equation 5.1. The utility for selecting w_1 by TF, represented as $u_{TF}(a_s)$ would be 0.5 as $\beta_{TF} < P_{TF}(w_1) = 0.02 < \alpha_{TF}$. This means that TF is uncertain for selecting the word. Similarly, the utility for selecting w_1 by DF would be 1.0 as $P_{DF}(w_1) = 0.75 \geq \alpha_{DF}$. This suggests that DF has a strong opinion in selecting the word. The payoff functions corresponding to pair $< u_{TF(a_s|a_s)}, u_{DF(a_s|a_s)} >$ can now be calculated as follows.

$$u_{TF(a_s|a_s)} = \frac{u_{TF}(a_s) + u_{DF}(a_s)}{2} = \frac{0.5 + 1.0}{2} = 0.75. \tag{5.5}$$

$$u_{DF(a_s|a_s)} = \frac{u_{DF}(a_s) + u_{TF}(a_s)}{2} = \frac{1.0 + 0.5}{2} = 0.75. \tag{5.6}$$

A payoff table may be obtained when all payoff pairs for the table are calculated as above.

We divide the words into three groups for facilitating analysis. The first group comprises of those words that have probabilities greater than both α_{DF} and α_{TF}. The second group contains those words that have atleast one probability greater than either α_{DF} or α_{TF}. The remaining words are considered in the third group. From Table 5.5, we note that for w_2, $P_{DF}(w_2) = 0.70 \geq \alpha_{DF}$ and $P_{TF}(w_2) = 0.07 \geq \alpha_{TF}$. This means that w_2 belongs to the first group. For w_1, we observe that $P_{DF}(w_1) = 0.75 \geq \alpha_{DF}$ while $P_{DF}(w_1) = 0.02 \ngeq \alpha_{TF}$. The word w_1 is therefore included in the second group. Similarly, the words w_3 and w_6 may also be verified as belonging to the second group. The remaining words, i.e. w_4, w_5, w_7, w_8 and w_9 are considered in the third group.

Table 5.6 Payoff table for w_2

		DF		
		a_s	a_a	a_r
TF	a_s	**1.0,1.0**	0.50, 0.50	0.50, 0.50
	a_a	0.50, 0.50	0.0, 0.0	0.0, 0.0
	a_r	0.50, 0.50	0.0, 0.0	0.0, 0.0

Let us consider the word w_2 in the first group. Table 5.6 presents its payoff table. The cell with bold numbers represents a Nash equilibrium. The actions of players in the state of equilibrium are a_s for both players. We may note that none of the players can achieve a higher payoff, given the other players chosen action. For instance, changing the action of DF from a_s to a_a or a_r will decrease the payoff from 1.0 to 0.5. The same is true for TF. The actions of players can be used to include w_2 in set $FS^{(a_s, a_s)}$.

We now consider the words in the second group. Table 5.7 shows the payoff table for w_1. The word has a moderate value for term frequency probability, i.e. between the two thresholds. However, the document frequency probability for the word has a higher value, i.e. $P_{DF}(w_1) \geq \alpha_{DF}$. The equilibrium analysis suggests action a_a for both players. The word w_1 is therefore included in the set $FS^{(a_s, a_a)}$.

The word w_6 is considered next. It has a higher term frequency probability while a moderate document frequency probability. Table 5.8 shows its payoff table. The

Table 5.7 Payoff table for w_1

		DF		
		a_s	a_a	a_r
TF	a_s	0.75, 0.75	0.25, 0.25	0.25, 0.25
	a_a	**1.0,1.0**	0.50, 0.50	0.50, 0.50
	a_r	0.75, 0.75	0.25, 0.25	0.25, 0.25

Table 5.8 Payoff table for w_6

		DF		
		a_s	a_a	a_r
TF	a_s	0.75, 0.75	**1.0,1.0**	0.75, 0.75
	a_a	0.25, 0.25	0.50, 0.50	0.25, 0.25
	a_r	0.25, 0.25	0.50, 0.50	0.25, 0.25

actions of the measures obtained with equilibrium analysis in this case are a_s for TF and a_a for DF, respectively. This suggests the inclusion of w_6 in set $FS^{(a_s,\, a_a)}$.

Table 5.9 Payoff table for w_3

		DF		
		a_s	a_a	a_r
TF	a_s	0.75, 0.75	0.25, 0.25	0.25, 0.25
	a_a	**1.0,1.0**	0.50, 0.50	0.50, 0.50
	a_r	0.75, 0.75	0.25, 0.25	0.25, 0.25

The word w_3 is finally considered in the second group. Its payoff table is presented as Table 5.9. We note that this word has similar probabilities as w_1. The equilibrium analysis results its inclusion in set $FS^{(a_s,\, a_a)}$.

We now consider words in the third group. We may divide this group into three subgroups. The first subgroup comprises of words having both probabilities between the two thresholds. The second subgroup have those words that have atleast one probability between the two thresholds. Finally, the third subgroup consists of those words having none of the their probabilities between the two thresholds. This means that the words w_4, w_5 and w_7 may be considered in the first group, w_8 in the second while w_9 in the third subgroup.

Table 5.10 Payoff table for w_4

		DF		
		a_s	a_a	a_r
TF	a_s	0.50, 0.50	0.75, 0.75	0.50, 0.50
	a_a	0.75, 0.75	**1.0,1.0**	0.75, 0.75
	a_r	0.50, 0.50	0.75, 0.75	0.50, 0.50

Let us analyze the words in the first subgroup. Table 5.10 shows the payoff table for w_4. The word has both probabilities in the two thresholds. The bold values in payoff table indicates its inclusion in set $FS^{(a_a,\, a_a)}$.

Table 5.11 Payoff table for w_5

		DF		
		a_s	a_a	a_r
TF	a_s	0.50, 0.50	0.75, 0.75	0.50, 0.50
	a_a	0.75, 0.75	**1.0,1.0**	0.75, 0.75
	a_r	0.50, 0.50	0.75, 0.75	0.50, 0.50

The remaining two words in the first subgroup are w_5 and w_7. Their payoff tables are presented as Table 5.11 and 5.12. These two words have similar characteristics as that of w_4, therefore they are also included in the set $FS^{(a_a,\, a_a)}$.

The word w_8 in the second subgroup is considered next. The actions in equilibrium suggest that DF can make a certain decision while TF may abstains from making a decision. The word is therefore included in the set $FS^{(a_r,\, a_a)}$.

Table 5.12 Payoff table for w_7

		DF		
		a_s	a_a	a_r
TF	a_s	0.50, 0.50	0.75, 0.75	0.50, 0.50
	a_a	0.75, 0.75	**1.0,1.0**	0.75, 0.75
	a_r	0.50, 0.50	0.75, 0.75	0.50, 0.50

Table 5.13 Payoff table for w_8

		DF		
		a_s	a_a	a_r
TF	a_s	0.25, 0.25	0.25, 0.25	0.75, 0.75
	a_a	0.50, 0.50	0.50, 0.50	**1.0,1.0**
	a_r	0.25, 0.25	0.25, 0.25	0.75, 0.75

Table 5.14 Payoff table for w_9

		DF		
		a_s	a_a	a_r
TF	a_s	0.0, 0.0	0.0, 0.0	0.50, 0.50
	a_a	0.0, 0.0	0.0, 0.0	0.50, 0.50
	a_r	0.50, 0.50	0.50, 0.50	**1.0,1.0**

Finally, w_9 is presented in Table 5.14. The actions of measures in equilibrium suggests its inclusion in set $FS^{(a_r,\, a_r)}$. This means that both measures considers the word as useless and therefore decides to reject it.

The six sets of features obtained with analysis of payoff tables as discussed above, may be summarized as,

$FS^{(a_s,\, a_s)} = \{w_2\}$,
$FS^{(a_s,\, a_a)} = \{w_1, w_3, w_6\}$,
$FS^{(a_s,\, a_r)} = \varnothing$,
$FS^{(a_a,\, a_a)} = \{w_4, w_5, w_7\}$,
$FS^{(a_r,\, a_a)} = \{w_8\}$, and
$FS^{(a_r,\, a_r)} = \{w_9\}$.

The sets of words corresponding to aggressive and moderate approaches are obtained as follows.

$F^A = FS^{(a_s,\, a_s)} = \{w_2\}$, and
$F^M = FS^{(a_s,\, a_s)} \bigcup FS^{(a_s,\, a_a)} \bigcup FS^{(a_s,\, a_r)} = \{w_2, w_1, w_3, w_6\}$.

This means that w_2 would be selected with an aggressive approach while w_2, w_1, w_3 and w_6 would be selected with a moderate approach. The words w_1 and w_6 which were previously considered as important by only one measure, are now being selected with the moderate approach. The above framework may provide a deeper insight if relative ranking is introduced among various feature sets. It is suggested that the GTRS based feature selection method may be useful in feature selection with multiple measures.

5.5 Conclusion

Existing feature selection methods in text categorization are either based on term frequency or document frequency measures. These methods may ignore certain information that is considered as important with other measures. Term frequency presents local document level information of words while document frequency presents entire dataset level information of words. For effective selection of features, we may consider both measures and treat them equally important. This chapter introduces a game-theoretic rough set (GTRS) based feature selection method in the aim of combining these measures in a unified framework. The significance of the method is that it combines the measures in a systematic way for evaluating and selecting features.

The GTRS model was previously suggested for probabilistic rough sets in combining measures for increasing the classification effectiveness within the rough set models. In this study the GTRS model was used for selecting features by formulating the measures as players in a game. Each measure uses its respective payoff functions in analyzing a feature and may choose from three possible actions, namely, accept (select), abstain and reject. The actions of measures are used in deriving a cooperative solution for selecting a feature. Particularly, six different types of feature sets were obtained that corresponds to various actions of measures. Two approaches,

aggressive feature selection and moderate feature selection, were defined with these feature sets. The moderate approach was found to be useful in a particular example considered in this study. Demonstrative examples suggest that the proposed method may be more efficient in feature selection for text categorization.

The presented formulation with the GTRS model may be utilized in investigating further problems requiring analysis of multiple measures and criteria for achieving a desired level of performance. Such problems are evident in many areas, such as searching, feature selection and rule mining.

References

1. Azam, N., Yao, J.T.: Incorporating Game Theory in Feature Selection for Text Categorization. In: Kuznetsov, S.O., Ślęzak, D., Hepting, D.H., Mirkin, B.G. (eds.) RSFDGrC 2011. LNCS, vol. 6743, pp. 215–222. Springer, Heidelberg (2011)
2. Azam, N., Yao, J.T.: Comparison of term frequency and document frequency based feature selection metrics in text categorization. Expert Systems with Applications 39(5), 4760–4768 (2012)
3. Dasgupta, A., Drineas, P., Harb, B., Josifovski, V., Mahoney, M.W.: Feature selection methods for text classification. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2007), pp. 230–239 (2007)
4. Ekenel, H.K., Sankur, B.: Feature selection in the independent component subspace for face recognition. Pattern Recognition Letters 25(12), 1377–1388 (2004)
5. Forman, G.: An extensive empirical study of feature selection metrics for text classification. Journal of Machine Learning Research 3, 1289–1305 (2003)
6. Herbert, J.P., Yao, J.T.: Game-Theoretic Risk Analysis in Decision-Theoretic Rough Sets. In: Wang, G., Li, T., Grzymala-Busse, J.W., Miao, D., Skowron, A., Yao, Y. (eds.) RSKT 2008. LNCS (LNAI), vol. 5009, pp. 132–139. Springer, Heidelberg (2008)
7. Herbert, J.P., Yao, J.T.: Learning Optimal Parameters in Decision-Theoretic Rough Sets. In: Wen, P., Li, Y., Polkowski, L., Yao, Y., Tsumoto, S., Wang, G. (eds.) RSKT 2009. LNCS, vol. 5589, pp. 610–617. Springer, Heidelberg (2009)
8. Herbert, J.P., Yao, J.T.: Game-theoretic rough sets. Fundamenta Informaticae 108(3-4), 267–286 (2011)
9. Jia, X., Li, W., Shang, L., Chen, J.: An Optimization Viewpoint of Decision-Theoretic Rough Set Model. In: Yao, J., Ramanna, S., Wang, G., Suraj, Z. (eds.) RSKT 2011. LNCS, vol. 6954, pp. 457–465. Springer, Heidelberg (2011)
10. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artificial Intelligence 97(1-2), 273–324 (1997)
11. Lakshmi, K., Mukherjee, S.: An improved feature selection using maximized signal to noise ratio technique for tc. In: Proceedings of 3rd International Conference on Information Technology: New Generations (ITNG 2006), pp. 541–546 (2006)
12. Lee, C., Lee, G.G.: Information gain and divergence-based feature selection for machine learning-based text categorization. Information Processing and Management 42(1), 155–165 (2006)
13. Li, H., Zhou, X., Zhao, J., Liu, D.: Attribute Reduction in Decision-Theoretic Rough Set Model: A Further Investigation. In: Yao, J., Ramanna, S., Wang, G., Suraj, Z. (eds.) RSKT 2011. LNCS, vol. 6954, pp. 466–475. Springer, Heidelberg (2011)
14. Liang, H., Wang, J., Yao, Y.Y.: User-oriented feature selection for machine learning. The Computer Journal 50(4), 421–434 (2007)

15. Liu, H., Yu, L.: Toward integrating feature selection algorithms for classification and clustering. IEEE Transactions on Knowledge and Data Engineering 17(4), 491–502 (2005)
16. López, F.R., Jiménez-Salazar, H., Pinto, D.: A Competitive Term Selection Method for Information Retrieval. In: Gelbukh, A. (ed.) CICLing 2007. LNCS, vol. 4394, pp. 468–475. Springer, Heidelberg (2007)
17. Mladenic, D., Grobelnik, M.: Feature selection for unbalanced class distribution and naive bayes. In: Proceedings of 16th International Conference on Machine Learning (ICML1999), pp. 258–267 (1999)
18. Moyotl-Hernández, E., Jiménez-Salazar, H.: Enhancement of DTP Feature Selection Method for Text Categorization. In: Gelbukh, A. (ed.) CICLing 2005. LNCS, vol. 3406, pp. 719–722. Springer, Heidelberg (2005)
19. von Neumann, J., Morgenstern, O.: Theory of Games and Economic Behavior. Princeton University Press (1944)
20. Ogura, H., Amano, H., Kondo, M.: Feature selection with a measure of deviations from poisson in text categorization. Expert Systems with Applications 36(3), 6826–6832 (2009)
21. Pawlak, Z.: Rough sets. International Journal of Computer and Information Sciences 11, 241–256 (1982)
22. Saeys, Y., Inza, I., Larrañaga, P.: A review of feature selection techniques in bioinformatics. Bioinformatics 23(19), 2507–2517 (2007)
23. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34(1), 1–47 (2002)
24. Swiniarski, R.W., Skowron, A.: Rough set methods in feature selection and recognition. Pattern Recognition Letters 24(6), 833–849 (2003)
25. Tang, B., Shepherd, M., Milios, E., Heywood, M.I.: Comparing and combining dimension reduction techniques for efficient text clustering. In: Proceeding of International Workshop on Feature Selection for Data Mining - Interfacing Machine Learning and Statistics in Conjunction with 2005 SIAM International Conference on Data Mining, Newport Beach, California, April 23, pp. 17–26 (2005)
26. Tseng, T.L.B., Huang, C.C.: Rough set-based approach to feature selection in customer relationship management. Omega 35(4), 365–383 (2007)
27. Yang, X.P., Song, H., Li, T.J.: Decision Making in Incomplete Information System Based on Decision-Theoretic Rough Sets. In: Yao, J., Ramanna, S., Wang, G., Suraj, Z. (eds.) RSKT 2011. LNCS, vol. 6954, pp. 495–503. Springer, Heidelberg (2011)
28. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Fisher, D.H. (ed.) Proceedings of the Fourteenth International Conference on Machine Learning (ICML 1997), Nashville, Tennessee, USA, July 8-12, pp. 412–420. Morgan Kaufmann (1997)
29. Yao, J.T., Herbert, J.P.: A game-theoretic perspective on rough set analysis. Journal of Chongqing University of Posts and Telecommunications (Natural Science Edition) 20(3), 291–298 (2008)
30. Yao, J.T., Herbert, J.P.: Analysis of Data-Driven Parameters in Game-Theoretic Rough Sets. In: Yao, J., Ramanna, S., Wang, G., Suraj, Z. (eds.) RSKT 2011. LNCS, vol. 6954, pp. 447–456. Springer, Heidelberg (2011)
31. Yao, J.T., Yao, Y.Y., Ziarko, W.: Probabilistic rough sets: Approximations, decision-makings, and applications. International Journal of Approximate Reasoning 49(2), 253–254 (2008)
32. Yao, J.T., Zhang, M.: Feature Selection with Adjustable Criteria. In: Ślęzak, D., Wang, G., Szczuka, M.S., Düntsch, I., Yao, Y. (eds.) RSFDGrC 2005. LNCS (LNAI), vol. 3641, pp. 204–213. Springer, Heidelberg (2005)

33. Yao, Y.Y.: Probabilistic rough set approximations. International Journal of Approximate Reasoning 49, 255–271 (2008)
34. Yao, Y.Y., Wong, S.K.M.: A decision theoretic framework for approximating concepts. International Journal of Man-Machine Studies 37, 793–809 (1992)
35. Zheng, Z., Wu, X., Srihari, R.: Feature selection for text categorization on imbalanced data. SIGKDD Exploration Newsletter 6(1), 80–89 (2004)
36. Zhou, B.: A New Formulation of Multi-category Decision-Theoretic Rough Sets. In: Yao, J., Ramanna, S., Wang, G., Suraj, Z. (eds.) RSKT 2011. LNCS, vol. 6954, pp. 514–522. Springer, Heidelberg (2011)

Chapter 6
A Clustering Approach to Image Retrieval Using Range Based Query and Mahalanobis Distance

Minakshi Banerjee, Sanghamitra Bandyopadhyay, and Sankar K. Pal

Abstract. This chapter puts forward a new approach to address a general purpose Content-Based Image Retrieval(CBIR) task. Six spatial color moments extracted from visually significant locations of an image are used as features to characterize an image. It utilizes information given by a set of queries, as opposed to a single image query. A set of similar images is posed as a query and Mahalanobis distance is used to evaluate the similarity between query images and target images of the database. Given a query set, the mean and covariance for computing Mahalanobis distance is obtained from the same. Unlike conventional CBIR methods in which images are retrieved based on considering similarities between the query image and the database images through a sequential search, a clustering technique using K-means algorithm is first used to create meaningful groups in the database. As clusters are created by considering similarities between images in the database, the image retrieval search space is reduced if clusters near to the query are searched. The effectiveness of the proposed algorithm is demonstrated with increased accuracy and reduced retrieval time.

Keywords: Image matching, Mahalanobis distance, clustering, precision, recall.

Minakshi Banerjee
Minakshi, Center for Soft Computing Research,Indian Statistical Institute
e-mail: mbanerjee23@gmail.com

Sanghamitra Bandyopadhyay
Sanghamitra, Machine Intelligence Unit, Indian Statistical Institute
e-mail: sanghami@isical.ac.in

Sankar K. Pal
Sankar, Center for Soft Computing Research, Indian Statistical Institute
e-mail: sankarpal@yahoo.com

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 79–91.
springerlink.com

6.1 Introduction

Content-Based Image Retrieval (CBIR) research has received intensive attention over a decade. CBIR aims at searching image databases for specific images that are similar to a given query image. It focuses on developing new techniques that support effective searching, and browsing of large digital image libraries based an automatically derived imagery features like (color, texture, shape etc.). It has extensive applications in accessing World- Wide web and in other application domains like biomedicine, crime prevention, digital libraries etc. Although, there has been an abundance of prior works [8] but CBIR research is still immature, owing to its inefficiency in retrieving semantically meaningful results for a query.

A typical CBIR system views the query image and images in the database (target images) as a collection of features, and ranks the relevance between the query image and any target image in terms of feature similarities based on some suitable similarity distance functions [4], [6], [10], [18], [25]. Unlike classical CBIR systems, which use a single image as a query, recently more emphasis is added towards category search in which a set of similar images are used a query [16], [27]. Information as obtained from the query set is utilized in the similarity evaluation process. Such information is required to retrieve visually similar images, which are quite close to human level perception. Despite several feature extraction algorithms and similarity functions, images with high feature similarities to the query image may be far apart from the query in terms of the interpretation made by a user (user semantics) arising due to limited descriptive power of low level imagery features. This is referred as the semantic gap [21].

To minimize semantic gap, an interactive method popularly known as relevance feedback is widely used to provide significant boost for meaningful retrieval. A relevance-feedback based approach allows a user to interact with the retrieval algorithm, by providing the information of which images the user thinks are relevant to the query [5], [13], [17], [20], [30], [33]. Based on the user's feedbacks, the model of similarity measure is dynamically updated to give a better approximation of the perception subjectivity. Although, effectiveness of relevance feedback is demonstrated for different applications, such a system may add burden to a user, especially when it is required to decide the degree of user's involvement in the retrieval process. On the other hand, a local approximation of the semantic structure of the whole image database may be obtained through some classification techniques. As an alternative to relevance feedback method, some image database preprocessing techniques using statistical classification methods, which group images into semantically meaningful categories may also be used to reduce semantic gap.

As each image of the database has an obvious semantic meaning, therefore clustering database images leading to a particular semantic concept is very important. Vailaya et al. [14] classify images into a hierarchical structure leading to different semantic concepts like indoor or outdoor scenes. The SIMPLIcity system [31]

classifies images into graph, textured photograph, or non-textured photograph, and thus narrows down the search space in a database. In CLUE, clusters are created depending on which images are retrieved in response to the query. The retrieved clusters are better associated with the semantic meaning of the images.

In a conventional CBIR, an image is usually represented by a set of features, where the feature vector is a point in the multidimensional feature space. Feature vectors extracted from images usually exist in a very high-dimensional space. However, this involves high computational cost. Better representation with lesser number of features is desirable for designing an effective CBIR system. There is an extensive literature, which deals with different low cost feature extraction mechanisms [7], [9], [12], [22], [23], [24], [32].

Another important aspect is that the same feature space may be endowed with a number of similarity distance leading to different retrieval results [19]. Several distance functions have been proposed and investigated in image retrieval applications. City block distance, Euclidean distance, Weighted Euclidean distance, Earth Mover's Distance [29], Mahalanobis distance etc., are some typical examples. Therefore, low dimensional feature and a suitable similarity metric which can produce satisfactory results is an important research issue. Mahalanobis distance is highly effective among these distance functions. Image retrieval using Mahalanobis distance has been reported in [15], where analysis of facial expression and facial pose retrieval is done using the concept of correlation and this distance. It has been used in an interactive fashion in [28] where this distance is used to update the weight matrix in every iteration based on users feedback, required to filter out irrelevant regions of the database for a particular query.

In view of the above facts, the proposed CBIR system is designed as follows. This approach uses six features only by considering the spatial color moments of visually significant points and the whole image. Before ranking the images according to feature similarities with respect to the query, we apply K-means clustering algorithm to analyze image similarities in the database, and generate semantically meaningful clusters. The pre-clustering technique considers similarities between images in the database, because it is based on pairwise distances so that within-cluster similarity is high and between-cluster similarity is low. Such organization into clusters additionally helps in identifying semantic relevance between images, even if the feature extraction and the similarity measure remain the same. This makes it possible to only search the clusters those are close to the query, instead of searching the whole database. In order to retrieve images which are similar a range of queries are posed instead of a single query. Here, information extracted from the query set is used in the similarity evaluation process. Such a range based query along with the clustering process generates quite promising results even if no relevance feedback mechanism is used. Feature based similarities between the query set and database images within the clusters are evaluated by comparing Mahalanobis distance between the same. The query set consist of a set of closely related images which are selected in a supervised way. All images of this set may not be a part of the database. The mean and covariance used to compute Mahalanobis distance is computed from the query

feature set. The database images are ranked and displayed in terms of the distance between an image of the query set.

The proposed methodology is described in Section 6.2. The important formulations are described in Section 6.3. The experimental procedure and results are demonstrated in Section 6.4 and the conclusion in Section 6.5.

6.2 System Overview

The overview of the proposed methodology is demonstrated in Fig. 6.1. Steps of which are also given below. The proposed methodology provides a guideline to search only those clusters, which have centroids near to the query feature set. The Mahalanobis distance which considers variability within the data obtained from the query samples (training set) tends to capture the underlying concept within an image also.

Algorithm 6.1.

Step1: Extract image features for each image in the database.
Step2: Apply K-means clustering to the feature database. (K is chosen equal to square root of the number of data points for a better approximation of the semantic structure of the whole image database.)
Step3: Consider a query set.
Step4: Select three clusters with centroids near to the query images.
Step5: Compute Mahalanobis distances between the query images and those in the clusters, with the mean and covariance generated from the feature distribution of the query set.
Step6: Rank the distances of all the images within the three clusters.

6.3 Theoretical Preliminaries

The definitions and the working principles of different parts used in the proposed work are explained in the following subsections.

6.3.1 Mahalanobis Distance

For a multivariate vector $x_i = [x_1, x_2, ..x_k]^T$ with a group of values with mean $\mu_i = [\mu_1, \mu_2, ...\mu_k]^T$ and covariance $\sum$, the Mahalanobis distance between x_i and μ_i is given by,

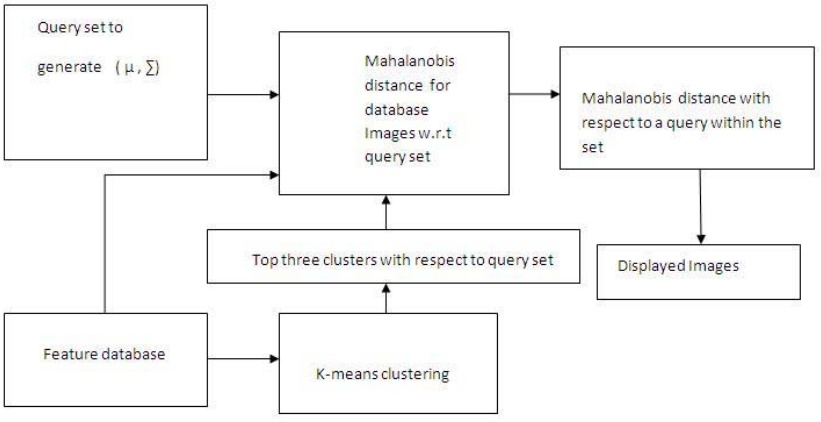


Fig. 6.1 Block diagram.

$$D(\mathbf{x_i}, \mu_\mathbf{i}) = \sqrt{(\mathbf{x_i} - \mu_\mathbf{i})^T \sum^{-1} (\mathbf{x_i} - \mu_\mathbf{i})} \tag{6.1}$$

x_i is the ith input vector with k attributes and $\sum^{-1}$ is the inverse of covariance matrix.

$$\sum = \begin{bmatrix} \sigma_{11} & \sigma_{12} & -- & \sigma_{1l} \\ \sigma_{21} & \sigma_{22} & - & \sigma_{2l} \\ - & - & & \\ \sigma_{l1} & \sigma_{l2} & -- & \sigma_{ll} \end{bmatrix}$$

$\sum_{jk}$=$cov(x_j, x_k)$ can be seen as the generalized scalar valued variance to higher dimensions. For a scalar valued random variable x, σ^2=var(x)=$E[(x-\mu)^2]$, where μ=E(X). The diagonal elements are variances of respective x and the off diagonal elements σ_{jk} are covariances of x_j and x_k The individual covariance values of $\sum^{-1}$ are computed from the outer product of the data.

Mahalanobis distance takes into account the variability of data unlike Euclidean distance. It is a weighted Euclidean distance, where the weighting is determined by the range of variability of the sample points expressed by its covariance matrix.

6.3.2 K-means Algorithm

Let X=$\{x_i\}$, $i = 1, \ldots, n$ be the set of n, d-dimensional points to be clustered into a set of k clusters, C=$\{c_k\}$, $k = 1, \ldots, k$. K-means algorithm finds a partition such that, the squared error between the empirical mean of a cluster and the points in the cluster is minimized. Let u_k be the mean of cluster c_k. The squared error between u_k and the points in cluster c_k is defined as $S(c_k) = \sum_{x_i \in c_k} ||x_i - u_k||^2$.

The goal of K-means is to minimize the within-clusters sum of squares, i.e.,to obtain $\arg\min \sum_{k=1}^{k} \sum_{x_i \in c_k} ||x_i - u_k||^2$. K-means starts with an initial partition with k clusters, and assign patterns to clusters so as to reduce the squared error. The squared error always decrease with an increase in the number of clusters k and reduces to 0 when k=n. It can be minimized only for a fixed number of clusters. The main steps of K-means algorithm are as follows: (1) An initial partition with K=k clusters is created. (2) A new partition by assigning each pattern to its closest cluster center is generated. (3) New cluster centers are computed. Steps (2) and (3) are repeated until cluster membership stabilizes.

6.3.3 Feature Extraction

Visually significant candidate points (high curvature points and corners) are extracted from each color image of the database. The extracted points are the reduced representatives of the original image. The detailed algorithm is explained in [1]. To compute invariant color moments from these locations, the color model (c_1,c_2,c_3) as proposed by Gevers et. al. [34] is chosen. In addition to traditional color spaces of HSI family, like Normalized RGB representation, illumination, and viewing geometry invariant representation, the new invariant color model (c_1,c_2,c_3) discounts the effects of shading and shadows also. The invariant feature model is defined in the following.

The RGB plane is converted to (c_1,c_2,c_3)

$$c_1 = arctan(R/max(G,B)) \tag{6.2}$$
$$c_2 = arctan(G/max(R,B)) \tag{6.3}$$
$$c_3 = arctan(B/max(R,G)) \tag{6.4}$$

In the next step, the color property of the selected candidates is extracted (using (6.5)) from each of the component planes i.e., (c_1,c_2,c_3) for computation of centralized second moments (μ_{20},μ_{02},μ_{11}). The normalized second central moments are computed from (6.5), which is used for computing invariant moments of order (p+q).

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}} \tag{6.5}$$

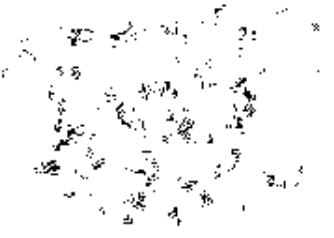
where $\gamma = \frac{p+q}{2} + 1$

Based on these, the moment invariant to translation, rotation, and scaling is derived as shown in (6.6). These set of features can also be considered as global descriptor of a shape with invariance properties, and with a built in ability to discern and filter noise [26], [11].

$$\phi = \eta_{20} + \eta_{02} \tag{6.6}$$

The image is characterized in the following manner. The moment (ϕ) of the extracted significant spatial locations will help to identify color similarity of the regions of interest (ROI). The features computed using the (c_1,c_2,c_3) values of the significant (ROI) may be used to produce efficient matching between images, due to its inherent structural information. However, for natural images (consisting of different objects) the representation obtained from shape signature (representation of the original image with groups of high curvature points), although important but may not be sufficient for discriminating them from other categories. Considering these facts, three additional set of moments taking all points from each of (c_1,c_2,c_3) planes are computed [2]. The components of the feature vector $F_k = [f_1, f_2, f_3 ..., f_6]$ are as follows, f_1, f_2, f_3 represent the (ϕ) values computed, considering all points of each component plane obtained from (6.4). These values did not vary with the thresholding levels. The components f_4, f_5, f_6 contain the (ϕ) values as obtained from the $\{c_1, c_2, c_3\}$ values of representative high curvature locations (ROI) and default white at other locations. A generated signature is shown in Fig. 6.2.

(a) Flower image.



(b) Fuzzy corner signature.

Fig. 6.2

6.3.4 Range Based Query and Mahalanobis Distance

A set of closely related query images is considered. For computing Mahalanobis distance between the query images and those in the clusters, μ_t and $\sum$ of (6.1) are estimated from the corresponding query set.

Let the query set S_t be composed of t distinct images I={I_{r1},I_{r2},...,I_{rt}}. Let an image database S_d be composed of d distinct images, i.e., I={I_1,I_2,...,I_d} where $I \in S_d$. An image is represented by a set of features F_k=$\{f_q\}_{q=1}^{N}$, where f_q is the qth feature component. This feature vector is a point in the N dimensional feature space.

The distance for each database image (point) with respect to the query set with mean μ_t is obtained as, d_{Id}=$D(\mathbf{I_d}, \mu_\mathbf{t})$. The images in the database are ranked according to that distance and displayed with respect to a particular image of the query set to evaluate the retrieval performance. If I_{rt} a point within query set, then the distance of this image (representing the point) with respect to the mean i.e., d_{It}=$D(\mathbf{I_{rt}}, \mu_\mathbf{t})$ is estimated. The images in the database are ranked with respect to the value of d_{idf}= d_{Id}-d_{It} and displayed.

6.4 Experimental Results

In order to show the retrieval performance using K-means clustering algorithm and Mahalanobis distance function, we perform a series of tests on a general-purpose image database (from COREL), named SIMPLIcity database, which includes 1000 images from 10 different semantic categories. Experimental results are shown in Figs. 6.2–Figs. 6.8. The feature vector is composed of six spatial color moments. The significant locations are extracted using a fuzzy corner detection algorithm proposed earlier in [1]. The significant locations are a representative set of the original image. This generates a perceptual representation of the original color image. The significant high curvature points of an example image are shown in Fig. 6.2. For each example, we examine the precision, and recall by considering the relevance of image semantics between the query and target images. These are defined by **Recall rate** and **Precision rate** [3]. Let n_1 be the number of images retrieved in top n positions that are close to a query image. Let n_2 represent the number of images in the database similar to the query. Evaluation standards, **Recall rate** (R) is given by $\frac{n_1}{n_2} \times 100\%$ and **Precision rate** (P) are given by $\frac{n_1}{n} \times 100\ \%$.

Top three clusters having centrionds closer to the query set are identified from minimum average Euclidean distance between the images of the query set and cluster centriods. The mean and covariance matrix are generated from the query set. The database images are ranked according to their distances with respect to an image within the set.

The query samples (training set) for a query type are shown in Fig. 6.3 and Fig. 6.4.

Fig. 6.3 Training query set. Images displayed with respect to the left most image.

Fig. 6.4 Continuation of the training query set. Images displayed with respect to the left most image.

Experimental results as shown in Fig. 6.5 and Fig. 6.6 considers a category (elephant) which is quite rich in meaningful semantics.

The displayed results are ranked respect to the leftmost image of the query set. The upper left corner image of Fig. 6.5 is the retrieved query image itself. In Fig. 6.6, are the other retrieved images. The results in Fig. 6.5 corresponds to those

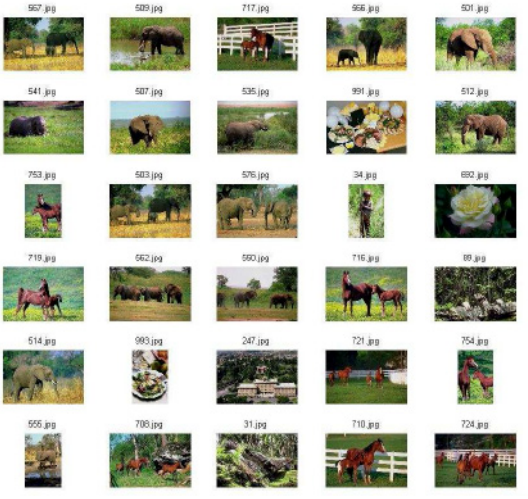


Fig. 6.5 Results without clustering (13/30) correctly retrieved.

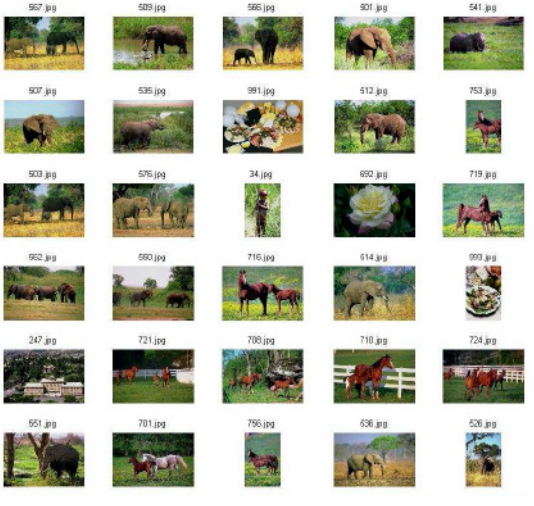


Fig. 6.6 Results with clustering (16/30) correctly retrieved with improved rank.

obtained without clustering, which is based on the Mahalanobis distance between the query set and all database images. Fig. 6.5, is having elephants of a special category. The displayed results are quite satisfactory because images retrieved are similar to each other and also to the query set. However, there is an outlier which is a horse with a similar background. It is to note that, this is also not very far apart from the category of the query. From the results obtained using K-means clustering

algorithm we observe in Fig. 6.6, this image is removed away and the recall rate is greater than 80%. It is to note that although this improvement is not very significant but the images retrieved have quite good semantic relevance.

Intuitively this may be explained as follows. Clustering explores similarities between images in the database so that within-cluster similarity is high. We observe that the application of K-means clustering algorithm in the image retrieval can throw away some images that are visually apart from the query image with a reduced retrieval space. The average precision curves with and without clustering, considering all database images as the query along with a generated training set, are shown in Fig. 6.7. As can be seen, the retrieval performance is improved with clustering. The experiment is performed in a Pentium 4 machine in windows XP using Matlab7. The average CPU time without and with clustering is 2 seconds and 600ms respectively. This can be accounted from the fact that, in case of the latter only the cluster of images close to the query are searched. As a result, the search space and the retrieval time are reduced.

The retrieval performance using Mahalanobis distance is compared against the popularly used Euclidean distance norms. The result is shown in Fig. 6.8. It is observed from the graph of Fig. 6.8, that the average precision is higher by (10-20)% than Euclidean distance, where the scope of display varies from 5-to 30 at a step of five. Euclidean distance is widely used in image retrieval applications owing to its simplicity and less complexity. However, Euclidean distance computes distances between feature points i.e., between query and database images and no query set is required. The enhancement of retrieval performance using Mahalanobis distance can be explained from the fact that, the distance is calculated considering error variances of each dimension within the query set. As a result, the variability within the query feature set is captured in the distance function itself and leading to better retrieved results.

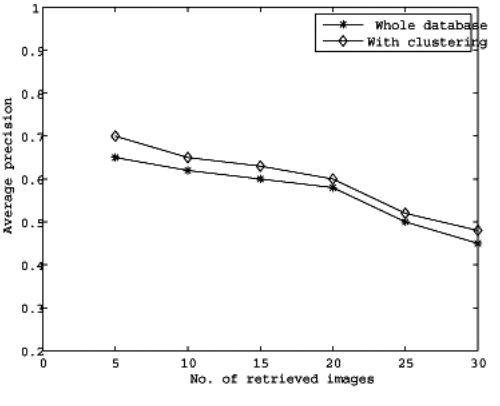


Fig. 6.7 Overall precision graph for sequential search and using clustering.

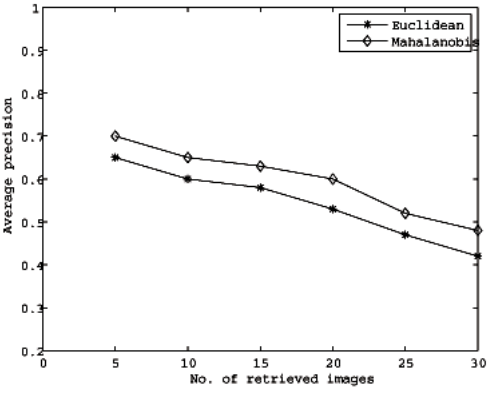


Fig. 6.8 Overall precision graph with Mahalanobis and Euclidean distance using top three clusters.

6.5 Conclusion

In the present work, similarity based retrieval is done using pre-categorization of the feature database with K-means clustering algorithm, and Mahalanobis distance as a similarity measure. Gain in computation time and performance is achieved due to clustering approach. Mahalanobis distance has been found to be significant in capturing the semantic relevance between images. In a CBIR system having a large number of features, the covariance matrix for Mahalanobis distance may be expensive. In an effective CBIR system, the feature dimension is generally very high as higher dimensional features are expected to capture visual properties more precisely. Each feature like color, texture, shape etc., tends to capture a particular visual aspect of an image. All these features may not be equally important for all types of queries. A feature selection scheme pertaining to a particular type of query may be used with this distance measure. A good clustering principle should result in a data partitioning that is stable with respect to perturbations in the data. The main difficulty using K-means clustering is the choice of initial K. Regarding computational complexity, the K-means clustering problem is NP-hard for a general number of clusters. Future research is necessary to address those issues.

Acknowledgement. The work was done when S. K. Pal was a J. C Bose National Fellow.

References

1. Banerjee, M., Kundu, M.K.: Handling of impreciseness in gray level corner detection using fuzzy set theoretic approach. Applied Soft Computing 8(4), 1680–1691 (2008)
2. Banerjee, M., Kundu, M.K., Maji, P.: Content-based image retrieval using visually significant point features. Fuzzy Sets and Systems 160(23), 3323–3341 (2009)

3. Bimbo, A.D.: Visual Information Retrieval. Morgan Kaufmann Publishers, Inc., San Francisco (2001)
4. Carson, C., Thomas, M., Belongie, S., Hellerstein, J.M., Malik, J.: Blobworld: A System for Region-Based Image Indexing and Retrieval. In: Huijsmans, D.P., Smeulders, A.W.M. (eds.) VISUAL 1999. LNCS, vol. 1614, pp. 509–517. Springer, Heidelberg (1999)
5. Chang, F.C., Hang, H.M.: A relevance feedback image retrieval scheme using multi-instance and pseudo image concepts. IEICE Transactions on Information and Systems D(5), 1720–1731 (2006)
6. Chen, Y., Wang, J.Z., Krovetz, R.: Clue: Cluster-based retrieval of images by unsupervised learning. IEEE Transactions on Image Processing 14(8), 1187–1201 (2005)
7. Daoudi, I., Idrissi, K.: A semi-supervised metric learning for content-based image retrieval. International Journal of Computer Vision and Image Processing (IJCVIP) 1(3) (2011)
8. Datta, R., Joshi, D., Li, J., Wang, J.Z.: Image retrieval: Ideas, influences and trends of the new age. ACM Computing Surveys 40(2) (2008)
9. Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Gorkani, M., Hafner, J., Lee, D., Petkovic, D., Yanker, P.: Query by image and video content: the qbic system. IEEE Computer 28(9), 23–32 (1995)
10. Gevers, T., Smeulders, A.W.M.: Combining color and shape invariant features for image retrieval. Image and Vision Computing 17(7), 475–488 (1999)
11. Gonzalez, R.C., Woods, R.E.: Digital Image Processing. Wiley, New York (1985)
12. Gouet, V., Boujemma, N.: About optimal use of color points of interest for content-based image retrieval. Tech report, RR-4439, INRIA, France (May 2006)
13. Han, J., Ngan, K.N., Li, M., Zhang, H.J.: A memory learning framework for effective image retrieval. IEEE Transactions on Image Processing 14(4), 521–524 (2005)
14. Jain, A.K., Vailaya, A.: Image retrieval using color and shape. Pattern Recognition 29, 1233–1244 (1996)
15. Kapoor, S., Khanna, S., Bhatia, R.: Facial gesture recognition using correlation and mahalanobis distance. International Journal on Computer Science and Information Security 7(2), 267–272 (2010)
16. Katherine, A., Ghahramani, Z.: A simple bayesian framework for content-based image retrieval. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2110–2117 (2006)
17. Kui, W., Hui, Y.K.: Content-based image retrieval using fuzzy perceptual feedback. Multimedia Tools and Applications 32(3), 235–251 (2007)
18. Kunttu, I., Lepisto, L., Rauhamaa, J., Visa, A.: Multiscale fourier descriptors for defect image retrieval. Pattern Recognition Letters 27(2), 123–132 (2006)
19. Lee, J., Nang, J.: Content-based image retrieval method using the relative location of multiple rois. Advances in Electrical and Computer Engineering 11(3) (2011)
20. Lim, J.H., Jin, J.S.: Combining intra-image and inter-class semantics for consumer image retrieval. Pattern Recognition 38(6), 847–864 (2005)
21. Liu, Y., Zhang, D., Lu, G., Ma, W.Y.: A survey of content-based image retrieval with high-level semantics. Pattern Recognition 40(1), 262–282 (2007)
22. Loupias, E., Sebe, N.: Wavelet-Based Salient Points: Applications to Image Retrieval Using Color and Texture Features. In: Laurini, R. (ed.) VISUAL 2000. LNCS, vol. 1929, pp. 223–232. Springer, Heidelberg (2000)
23. Matas, J., Marik, R., Kittler, J.: On representation and matching of multi-colored objects. In: 5th International Conference on Computer Vision, vol. 31, pp. 1369–1390 (1998)
24. Mikolajczyk, K., Schmid, C.: Scale and affine invariant interest point detectors. International Journal of Computer vision 1(60), 63–86 (2004)
25. Moghaddam, H.A., Taghizadeh, T.K., Rouhi, A.H., Saadatmand, M.T.: Wavelet correlogram: a new approach for image indexing and retrieval. Pattern Recognition 38(12), 2506–2518 (2005)

26. Nixon, M.S., Aguado, A.S.: Feature extraction and Image Processing. Reed Educational and Professional Publishing Ltd, Oxford (2002)
27. Rahmani, R., Goldman, S.A., Zhang, H., Choletti, S.R., Fritts, J.E.: Localized content-based image retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(11), 1902–1912 (2008)
28. Ramaswamy, S., Rose, K.: Fast adaptive mahalanobis distance - based search and retrieval in image databases. In: International Conference on Image Processing, pp. 181–184 (2008)
29. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover's distance as a metric for image retrieval. International Journal of Computer Vision 40(2) (2000)
30. Rui, Y., Huang, T.S., Ortega, M., Mehrotra, S.: Relevance feedback: a power tool for interactive content-based image retrieval. IEEE Transactions on Circuits and Systems for Video technology 8(5), 644–655 (1998)
31. Wang, J.Z., Li, J., Wiederhold, G.: Simplicity: Semantics-sensitive integrated matching for picture libraries. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(9), 947–963 (2001)
32. Weijer, J., Gevers, T., Bagdanov, A.: Boosting color saliency in image feature detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(1), 150–156 (2006)
33. Yin, P., Bhanu, B., Chang, K.C., Dong, A.: Integrating relevance feedback techniques for image retrieval using reinforcement learning. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(10), 1536–1551 (2005)
34. Zevers, T., Smeulders, A.W.M.: Color - based object recognition. Pattern Recognition 32, 453–464 (1999)

Chapter 7
Classifiers Based on Data Sets and Domain Knowledge: A Rough Set Approach

Jan G. Bazan, Stanisława Bazan-Socha, Sylwia Buregwa-Czuma,
Przemysław Wiktor Pardel, Andrzej Skowron, and Barbara Sokołowska

Abstract. The problem considered is how to construct classifiers for approximation of complex concepts on the basis of experimental data sets and domain knowledge that are mainly represented by concept ontology. The approach presented in this chapter to solving this problem is based on the rough set theory methods. Rough set theory introduced by Zdzisław Pawlak during the early 1980s provides the foundation for the construction of classifiers. This approach is applied to approximate spatial complex concepts and spatio-temporal complex concepts defined for complex objects, to identify the behavioral patterns of complex objects, and to the automated behavior planning for such objects when the states of objects are represented

Jan G. Bazan
Institute of Computer Science, University of Rzeszów, Dekerta 2, 35 - 030 Rzeszów, Poland
Institute of Mathematics, University of Warsaw, Banacha 2, 02-097 Warsaw, Poland
e-mail: bazan@univ.rzeszow.pl

Stanisława Bazan-Socha
Second Department of Internal Medicine, Jagiellonian University Medical College, Skawinska 8, 31-066 Cracow, Poland
e-mail: mmsocha@cyf-kr.edu.pl

Sylwia Buregwa-Czuma
Institute of Computer Science, University of Rzeszów, Dekerta 2, 35 - 030 Rzeszów, Poland
e-mail: sczuma@univ.rzeszow.pl

Przemysław Wiktor Pardel
Institute of Computer Science, University of Rzeszów, Dekerta 2, 35 - 030 Rzeszów, Poland
Institute of Mathematics, University of Warsaw, Banacha 2, 02-097 Warsaw, Poland
e-mail: ppardel@univ.rzeszow.pl

Andrzej Skowron
Institute of Mathematics, University of Warsaw, Banacha 2, 02-097 Warsaw, Poland
e-mail: skowron@mimuw.edu.pl

Barbara Sokołowska
Second Department of Internal Medicine, Jagiellonian University Medical College, Skawińska 8, 31-066 Cracow, Poland
e-mail: basiasok1@gmail.com

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 93–136.
springerlink.com

by spatio-temporal concepts requiring approximation. The chapter includes results of experiments that have been performed on data from a vehicular traffic simulator and the recent results of experiments that have been performed on medical data sets obtained from Second Department of Internal Medicine, Jagiellonian University Medical College, Cracow, Poland. Moreover, we also describe the results of experiments that have been performed on medical data obtained from Neonatal Intensive Care Unit in the Department of Pediatrics, Jagiellonian University Medical College, Cracow, Poland.

Keywords: Rough set, concept approximation, complex dynamical system, ontology of concepts, behavioral pattern identification, automated planning.

7.1 Introduction

Classifiers also known in literature as *decision algorithms*, *classifying algorithms* or *learning algorithms* may be treated as constructive, approximate descriptions of concepts (decision classes). These algorithms constitute the kernel of *decision systems* that are widely applied in solving many problems occurring in such domains as *pattern recognition*, *machine learning*, *expert systems*, *data mining* and *knowledge discovery* (see, *e.g.*, [16, 20, 23, 28–30, 41]).

In literature there can be found descriptions of numerous approaches to constructing classifiers, which are based on such paradigms of machine learning theory as *classical and modern statistical methods* (see, *e.g.*, [29, 41]), *neural networks* (see, *e.g.*, [29, 41]), *decision trees* (see, *e.g.*, [29]), *decision rules* (see, *e.g.*, [28, 29]), and *inductive logic programming* (see, *e.g.*, [29]). Rough set theory introduced by Zdzisław Pawlak during the early 1980s also provides the foundation for the construction of classifiers (see, *e.g.*, [32, 33, 35, 45]).

Recently, it has been noticed in the literature that with the development of modern civilization, not only the scale of the data gathered but also the complexity of concepts and phenomena which they concern are increasing rapidly. This crucial data change has brought new challenges to work out new data mining methods. Particularly, data more and more often concerns complex processes which do not give in to classical modeling methods. Of such a form may be medical and financial data, data coming from vehicles monitoring, or data about the users gathered on the Internet. Exploration methods of such data are in the center of attention in many powerful research centers in the world, and at the same time detection of models of complex processes and their properties (patterns) from data is becoming more and more attractive for applications (see, *e.g.*, [1, 11, 25, 31, 47, 49]).

When modeling complex real-world phenomena and processes mentioned above and solving problems under conditions that require an access to various distributed data and knowledge sources, the so-called *complex dynamical systems* (CDS) are often applied (see, *e.g.*, [3, 13, 27, 50]), or putting it in other way *autonomous multiagent systems* (see, *e.g.*, [19, 26, 27]) or *swarm systems* (see, *e.g.*, [34]). These

are collections of complex interacting objects characterized by constant change of parameters of their components and their interactions over time, numerous relationships between the objects, the possibility of cooperation/competition among the objects and the ability of objects to perform more or less compound actions. Examples of such systems are *traffic*, *a patient observed during treatment*, *a team of robots performing tasks*, etc. It is also worthwhile mentioning that the description of a CDS dynamics is often not possible with purely analytical methods as it includes many complex vague concepts (see, *e.g.*, [22, 40]). Such concepts concern properties of chosen fragments of the CDS and may be treated as more or less complex objects occurring in the CDS. Hence, are needed appropriate methods of extracting such fragments (granules [5]) that are sufficient to conclude about the global state of the CDS in the context of the analyzed types of changes and behaviors. The identification of complex spatio-temporal concepts and using them to monitor a CDS requires approximation of these concepts.

Making a progress in this field is extremely crucial, among other things, for the development of intelligent systems making decision under uncertainty on the basis of results of analysis of the available data sets. Therefore, working out methods of detection of process models and their properties from data and proving their effectiveness in different applications are of particular importance for the further development of decision supporting systems in many domains such as medicine, finance, industry, transport, telecommunication, and others.

However, essential limitations have been discovered concerning the existing data mining methods for very large data sets regarding complex concepts, phenomena, or processes (see, *e.g.*, [12, 38, 51–53]). A crucial limitation of the existing methods is, among other things, the fact that they do not support an effective approximation of complex concepts, that is, concepts whose approximation requires discovery of extremely complex patterns. Intuitively, such concepts are too far in the semantical sense from the available concepts, *e.g.*, sensory ones. As a consequence, the size of searching spaces for relevant patterns crucial for approximation are so large that an effective search of these spaces very often becomes unfeasible using the existing methods and technology. Thus, as it turned out, the ambition to approximate complex concepts with high quality from available concepts (most often defined by sensor data) in a fully automatic way, realized by the existing systems and by most systems under construction, is a serious obstacle since the classifiers obtained are often of unsatisfactory quality.

Moreover, it has been noticed in the literature (see, *e.g.*, [15, 24, 38, 46]) that one of the challenges for data mining is discovery of methods linking detection of patterns and concepts with domain knowledge. The latter term denotes knowledge about concepts occurring in a given domain and various relations among them. This knowledge greatly exceeds the knowledge gathered in data sets; it is often represented in a natural language and usually acquired during a dialogue with an expert in a given domain. One of the ways to represent domain knowledge is to record it in the form of the so-called *concept ontology* where ontology is usually understood as a finite hierarchy of concepts and relations among them, linking concepts from different levels (see, *e.g.*, [17, 21]).

The main aim of the chapter is to present the developed methods for approximation of complex vague concepts involved in specification of real-life problems and approximate reasoning used in solving these problems. However, methods presented in the chapter are assuming that additional domain knowledge in the form of the concept ontology is given. Concepts from ontology are often vague and expressed in natural language. Therefore, an approximation of ontology is used to create hints in searching for approximation of complex concepts from sensory (low level) data.

We propose to link automatic methods of complex concept learning, and models of detection of processes and their properties with domain knowledge obtained in a dialog with an expert. Interaction with a domain expert facilitates guiding the process of discovery of patterns and models of processes and makes the process computationally feasible.

As we mentioned before, our methods for approximating complex spatio-temporal concepts and relations among them assuming that the information about concepts and relations is given in the form of ontology. To meet these needs, by ontology we understand a finite set of concepts creating a hierarchy and relations among these concepts which link concepts from different levels of the hierarchy. At the same time, on top of this hierarchy there are always the most complex concepts whose approximations we are interested in aiming at practical applications. Moreover, we assume that the ontology specification contains incomplete information about concepts and relations occurring in ontology, particularly for each concept, sets of objects constituting examples and counterexamples for these concepts are given. Additionally, for concepts from the lowest hierarchical level (sensor level) it is assumed that there are also *sensor attributes* available which enable to approximate these concepts on the basis of positive and negative examples given (see example of ontology from Fig. 7.1 and [44]).

In this chapter, we present the following four types of methods for approximating spatial or spatio-temporal complex concepts.

1. Methods of approximation of spatial concepts — when a complex concept is a spatial concept not requiring an observation of changes over time (see Section 7.2).
2. Methods of approximation of spatio-temporal concepts — when a complex concept is a spatio-temporal concept; it requires observing changes of complex objects over time (see Section 7.3).
3. Methods of behavioral pattern identification — when a complex concept is represented as a certain directed graph which is called *a behavioral graph* (see Section 7.4).
4. Methods of automated behavior planning for complex object - when the states of objects are represented by spatio-temporal concepts requiring approximation (see Section 7.5).

The result of the works conducted is also a programming system the *Rough Set Interactive Classification Engine* (RoughICE), supporting the approximation of spatio-temporal complex concepts in the given ontology in the dialog with the user.

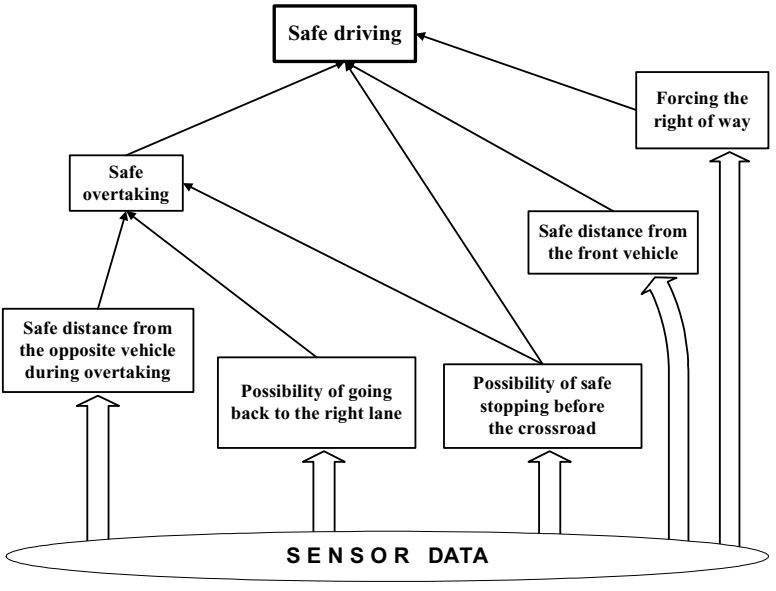


Fig. 7.1 An ontology for safe driving

The RoughICE includes an implementation of the algorithmic methods presented in this chapter and is available on the web side [42]. To simplify the use of RoughICE algorithms and make it more intuitive, the RoughICE graphical user interface was constructed that consists of three following parts (see Fig. 7.2):

1. *project designer* — directed toward visual representation of workflow,
2. *graph editor* — for representing domain knowledge in the form of ontology,
3. *script editor and compiler* — for representing domain knowledge in the form of scripts.

Sections 7.2, 7.4, and 7.5, apart from the method description, contain the results of computing experiments conducted on real-life data sets, supported by domain knowledge. It is worth mentioning that the requirements regarding data sets which can be used for computing experiments with modeling spatio-temporal phenomena are much greater than the requirements of the data which are used for testing process of classical classifiers. Not only have the data to be representative of the decision making problem under consideration but also they have to consist the relevant domain knowledge about approximated concepts (usually cooperation with experts in a particular domain is essential for acquisition of domain knowledge). It is important that such data should fully and appropriately represent complex spatio-temporal phenomena of the environment.

The authors of the chapter acquired such data sets from three sources.

The first source of data is the traffic simulator (see [44] and [4] for more details). The simulator is a computing tool for generating data sets connected to the traffic on

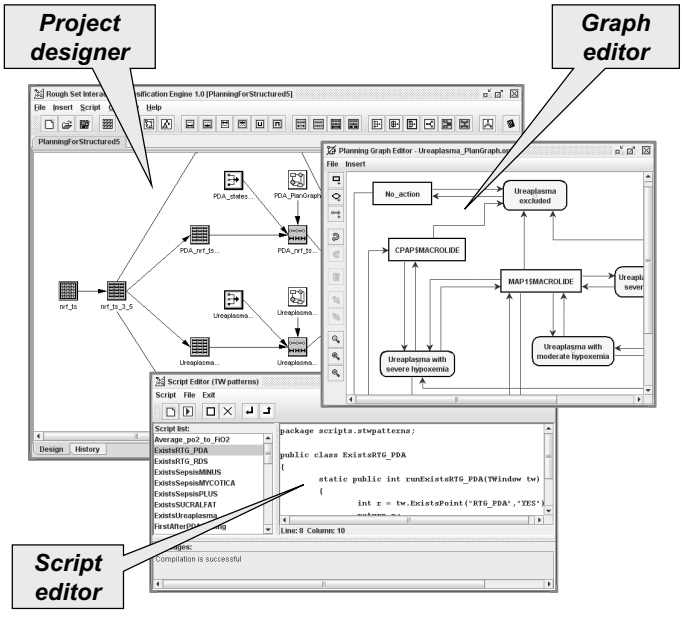


Fig. 7.2 The view of RoughICE graphical user interface

the street and at crossroads. During simulation each vehicle appearing on the simulation board behaves as an independently acting agent. On the basis of observation of the surroundings (other vehicles, its own location, weather conditions, etc.) this agent makes an independent decision what maneuvers it should make to achieve its goal which is to go safely across the simulation board and to leave the board using the outbound way given in advance. At any given moment of the simulation, all crucial vehicle parameters may be recorded, and thanks to this data sets for experiments can be obtained. The results of experiments with the data sets recorded in the road simulator were presented in Section 7.2.

The second collection of data sets used in computer experiments was provided by Second Department of Internal Medicine, Collegium Medicum, Jagiellonian University, Cracow, Poland. This data includes characteristics of patients with stable coronary heart disease: clinical status, past history, the laboratory tests results, electrocardiographic (ECG) recordings, applied therapeutic procedures, and coronary angiography outcomes. In the chapter we present recent results of experiments performed for this collection of data sets (see Section 7.4).

The third collection of data sets used in computer experiments was provided by Neonatal Intensive Care Unit, First Department of Pediatrics, Polish-American Institute of Pediatrics, Collegium Medicum, Jagiellonian University, Cracow, Poland. This data constitutes a detailed description of treatment of 300 infants, i.e., treatment results, diagnosis, operations, medication (see [4, 6–9]). The results for this data collection we present in Section 7.5.

7.2 Methods of Approximation of Spatial Concepts

The method of approximating concepts from ontology is proposed when a concept is a *spatial concept* (not requiring an observation of changes over time) and it is defined on a set of the same objects (examples) as the lower ontology level concepts; at the same time, the lower level concepts are also spatial concepts. An exemplary situation of this type is an approximation of the concept of *Safe overtaking* (concerning single vehicles on the road) using concepts such as *Safe distance from the opposite vehicle during overtaking*, *Possibility of going back to the right lane*, and *Possibility of safe stopping before the crossroads* (see Fig. 7.1).

In the chapter, the method of approximating concepts from ontology is proposed when a higher ontology level concept is a spatial concept (not requiring an observation of changes over time) and it is defined on a set of the same objects (examples) as the lower ontology level concepts; at the same time, the lower level concepts are also spatial concepts. An exemplary situation of this type is an approximation of the concept of *Safe overtaking* (concerning single vehicles on the road) using concepts such as *Safe distance from the opposite vehicle during overtaking*, *Possibility of going back to the right lane*, and *Possibility of safe stopping before the crossroads*.

The concept approximation method described in this subsection is an example of the general methodology of approximating concepts from ontology described in [4]. That is why its specificity is the domain knowledge usage expressed in the form of a concept ontology and application of rough set methods, mainly in terms of application of classifier construction methods.

The basic terms used in the presented method is *pattern* and *production rule*. Patterns are descriptions of examples of concepts from an ontology and they are constructed by *a stratifying classifier*, defined as a classifying algorithm stratifying concepts, that is, classifying objects to different concept layers (see [4] for more details). Two approaches have been proposed to the construction of these classifiers. One of them is *the expert approach* which is based on the defining, by an expert, an additional attribute in data which describes membership of the object to individual concept layers. Next, a classifier differentiating layers as decision classes is constructed. The second approach called *the automated approach* is based on the designing algorithms being the classifier extensions which enable to classify objects to concept layers on the basis of certain premises and experimental observations. In [4] a method of this type has been proposed which is based on shortening of decision rules relatively to various coefficients of consistency.

For example, we consider a concept C, where inclusion to this concept is described by six linearly ordered layers *"certainly NO"*, *"rather NO"*, *"possibly NO"*, *"possibly YES"*, *"rather YES"* and *"certainly YES"*. For the concept C we define a pattern ($C \geq$ "*rather YES*"), that has the following interpretation: *the inclusion to the concept C is at least "rather YES"*. It is easy to see that a stratifying classifier can be used to judge whether a tested object belongs to this pattern or not.

A production rule is a decision rule which is constructed on two adjacent levels of ontology. In the *predecessor* of this rule there are patterns for the concepts from the lower level of the ontology whereas in the *successor*, there is a pattern for one

concept from the higher level of the ontology (connected with concepts from the rule predecessor) where both patterns from the predecessor and the successor of the rule are chosen from patterns constructed earlier for concepts from both adjacent levels of the ontology. In Fig. 7.3 we present an example of production rule for concepts C_1, C_2 and C. This production rule has the following interpretation: if inclusion degree to a concept C_1 is at least *"possibly YES"* and to concept C_2 at least *"rather YES"* then the inclusion degree to a concept C is at least *"rather YES"*.

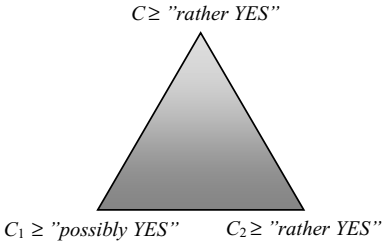


Fig. 7.3 The example of production rule

A rule constructed in such a way may serve as a simple classifier or an argument "for"/"against" the given concept, enabling classification of objects which match the patterns from the rule predecessor with the pattern from the rule successor. For example, the object u_1 from Fig. 7.4 is classified by production rule from Fig. 7.3 because it matches both patterns from the left hand side of the production rule whereas, the object u_2 from Fig. 7.4 is not classified by production rule because it does not match the second source pattern of production rule (the value of attribute C_2 is less than *"rather YES"*).

In [4], there was proposed an algorithmic method of induction of production rules, consisting in an appropriate search for data tables with attributes describing the membership of training objects to particular layers of concepts. These tables (called *a layer table*) are constructed using the so-called constraints between concepts thanks to which the information put in the tables only concerns those objects/examples which might be found there according to the production rule under construction. In Fig. 7.5, we illustrate the process of extracting production rule for concept C and for the approximation layer "*rather YES*" of concept C. Is is easy to see that if from the table from Fig. 7.5 we select all objects satisfying $a_C =$ "*rather YES*", then for selected objects minimal value of the attribute a_{C_1} is equal to "*possibly YES*" and minimal value of the attribute a_{C_2} is equal to "*rather YES*". Hence, we obtain the production rule:

$$(C_1 \geq \text{"}\textit{possibly YES}\text{"}) \wedge (C_2 \geq \text{"}\textit{rather YES}\text{"}) \Rightarrow (C \geq \text{"}\textit{rather YES}\text{"}).$$

Although a single production rule may be used as a classifier for the concept appearing in a rule successor, it is not a complete classifier yet, i.e., classifying all objects belonging to an approximated concept and not only those matching

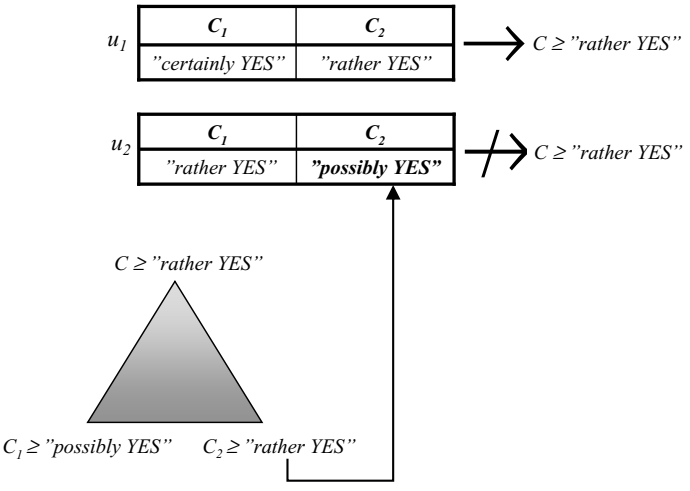
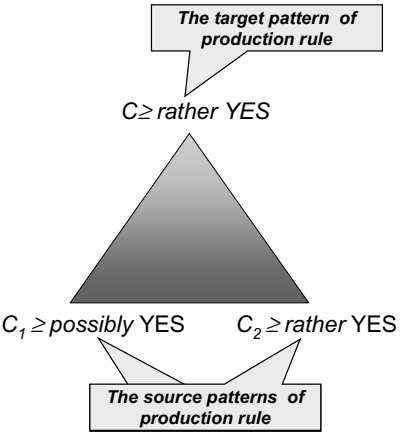


Fig. 7.4 Classifying tested objects by single production rule

a_{C1}	a_{C2}	a_C
certainly YES	certainly YES	**certainly YES**
certainly NO	certainly NO	certainly NO
rather YES	certainly YES	**rather YES**
possibly YES	possibly NO	possibly YES
possibly YES	possibly NO	rather NO
possibly YES	**rather YES**	**rather YES**
possibly YES	certainly NO	possibly NO
certainly YES	**rather YES**	**certainly YES**
certainly NO	possibly NO	certainly NO



certainly NO < rather NO < possibly NO < possibly YES < rather YES < certainly YES

Fig. 7.5 The illustration of production rule extracting

patterns of a rule predecessor. Therefore, in practice, production rules are grouped into the so-called *productions*, i.e., production rule collections, in a way that each production contains rules having patterns for the same concepts in a predecessor and the successor, but responding to their different layers. In Fig. 7.6 we present three production rules constructed for some concepts C_1, C_2, and C approximated by six linearly ordered layers *"certainly NO"*, *"rather NO"*, *"possibly NO"*, *"possibly YES"*, *"rather YES"* and *"certainly YES"*. This collection of production rules

is an exemplary production for concepts C_1, C_2, and C. Moreover, production rules from Fig. 7.6 have the following interpretation:

1. if inclusion degree to a concept C_1 is at least "*rather YES*" and to concept C_2 at least "*certainly YES*", then the inclusion degree to a concept C is at least "*certainly YES*";
2. if the inclusion degree to a concept C_1 is at least "*possibly YES*" and to a concept C_2 at least "*rather YES*", then the inclusion degree to a concept C is at least "*rather YES*";
3. if the inclusion degree to a concept C_1 is at least "*possibly YES*" and to a concept C_2 at least "*possibly YES*", then the inclusion degree to a concept C is at least "*possibly YES*".

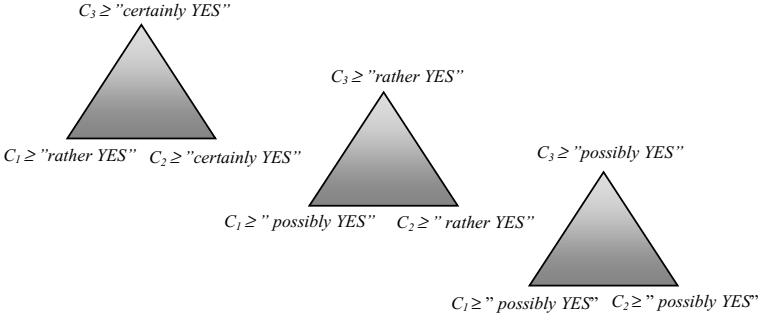


Fig. 7.6 The example of production as a collection of three production rules

In the case of production from Fig. 7.6, concept C is the target concept and C_1, C_2 are the source concepts.

Such production makes is possible to classify much more objects than a single production rule where these objects are classified into different layers of the concept occurring in a rule successor. Both productions and production rules themselves are only constructed for the two adjacent levels of ontology. Therefore, in order to use the whole ontology fully, there are constructed the so-called AR-schemes, i.e., *approximate reasoning schemes* which are hierarchical compositions of production rules (see, *e.g.*, [10, 14, 39]). The synthesis of an AR-scheme is carried out in a way that to a particular production from a lower hierarchical level of the AR-scheme under construction another production rule on a higher level may be attached, but only that one where one of the concepts for which the pattern occurring in the predecessor was constructed is the concept connected with the rule successor from the previous level. Additionally, it is required that the pattern occurring in a rule predecessor from the higher level is a subset of the pattern occurring in a rule successor from the lower level (in the sense of inclusion of object sets matching both patterns). To the two combined production rules other production rules can be attached (from above, from below, or from the side) and in this way a multilevel structure is made

which is a composition of many production rules. The AR-scheme constructed in such a way can be used as a hierarchical classifier whose entrance are predecessors of production rules from the lowest part of the AR-scheme hierarchy and the exit is the successor of a rule from the highest part of the AR-scheme hierarchy. That way, each AR-scheme is a classifier for a concept occurring in the rule successor from the highest part in the hierarchy of the scheme and, to be precise, for a concept for which a pattern occurring in the rule successor from the highest part in the hierarchy of the AR-scheme is determined.

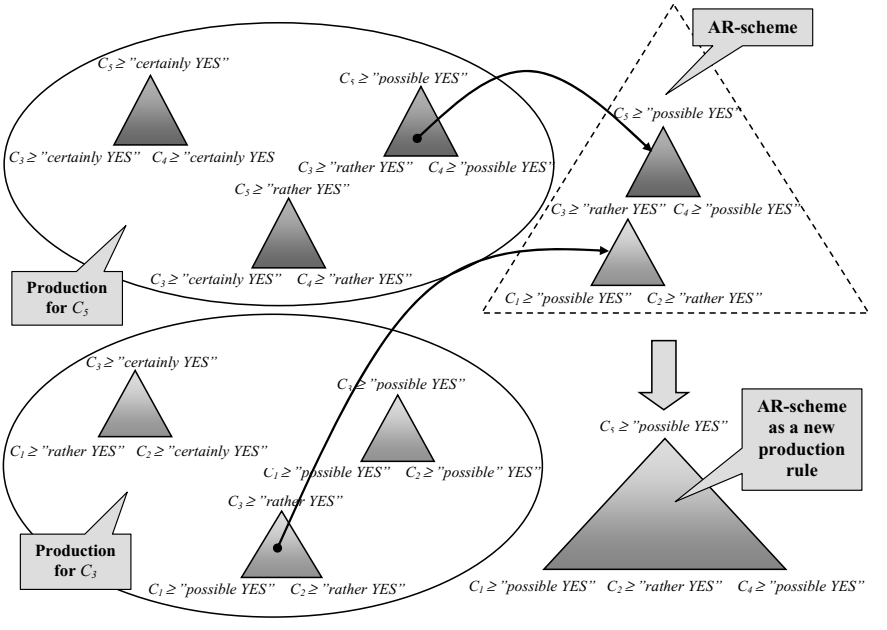


Fig. 7.7 Synthesis of approximate reasoning scheme

For example, in Fig. 7.7 we have two productions. The target concept of the first production is C_5 and the target concept of the second production is the concept C_3. We select one production rule from the first production and one production rule from the second production. These production rules are composed and then a simple AR-scheme is obtained that can be treated as a new two-levels production rule. Notice, that the target pattern of lower production rule in this AR-scheme is the same as one of the source patterns from the higher production rule. In this case, the common pattern is described as follows: inclusion degree (of some pattern) to a concept C_3 is at least *"possibly YES"*.

In this way, we can compose AR-schemes into hierarchical and multilevel structures using productions constructed for various concepts. AR-scheme constructed in such a way can be used as a hierarchical classifier whose input is given by

predecessors of production rules from the lowest part of AR-scheme hierarchy and the output is a successor of a rule from the highest part of the AR-scheme hierarchy.

However, similarly to the case of a single production rule, an AR-scheme is not a full classifier yet. That is why, in practice, for a particular concept there are many AR-schemes constructed which approximate different layers or concept regions.

In the paper [4], there are proposed two approaches for constructing AR-schemes. The first approach is based on memory with AR-schemes and consists in building many AR-schemes after determining production, which later on are stored and used for the classification of tested objects.

The second approach is based on a dynamic construction of AR-schemes. It is realized in a way that during classification of a given tested object, an appropriate AR-scheme for classifying this particular object is built on the basis of a given collection of productions ("lazy" classification).

7.2.1 Experiments with Data

To verify effectiveness of classifiers based on AR schemes, we have implemented our algorithms in the RoughICE (see Section 7.1 and [4]).

The experiments have been performed on the data set obtained from the road simulator (see [44]). Data set consists of 18101 objects generated by the road simulator. We have applied the train and test method. The data set was randomly divided into two parts: training and test ones (50% + 50%). In order to determine the standard deviation of the obtained results each experiment was repeated for 10 random divisions of the whole data set.

In our experiments, we compared the quality of two classifiers: RS and ARS. For inducing RS, we use RSES system (see [43]) generating the set of decision rules by algorithm LEM2 (see [4] for more details) that is next used for classifying situations from testing data. ARS is based on AR schemes (the implementation from [42]).

During ARS classifier construction, in order to approximate concepts occurring in ontology we also used the LEM2 algorithm.

For production rule construction, we used the expert method of stratifying classifier construction. However, to classify objects using the ARS classifier we used the method of dynamic construction of the AR-schemes for specific tested objects (see [4]).

We compared RS and ARS classifiers using the accuracy, the coverage, the accuracy for positive examples (also called as the sensitivity or the true positive rate), the accuracy for negative examples (also called as the specificity or the true negative rate), the coverage for positive examples, and the coverage for negative examples, the real accuracy (where Real accuracy = Accuracy * Coverage), the learning time, and the rule set size.

Table 7.1 Results of experiments for the concept: *Is the vehicle driving safely?*

Decision class	Method	Accuracy	Coverage	Real accuracy
YES	RS	0.977 ± 0.001	0.948 ± 0.003	0.926 ± 0.003
	ARS	0.967 ± 0.001	0.948 ± 0.003	0.918 ± 0.003
NO	RS	0.618 ± 0.031	0.707 ± 0.010	0.436 ± 0.021
	ARS	0.954 ± 0.016	0.733 ± 0.018	0.699 ± 0.020
All classes (YES + NO)	RS	0.963 ± 0.001	0.935 ± 0.003	0.901 ± 0.003
	ARS	0.967 ± 0.001	0.937 ± 0.004	0.906 ± 0.004

Table 7.2 Learning time and the rule set size for concept: *Is the vehicle driving safely?*

Method	Learning time	Rule set size
RS	488 ± 21 seconds	975 ± 28
ARS	33 ± 1 second	174 ± 3

Table 7.1 shows the results of the considered classification algorithms for the concept *Is the vehicle driving safely?* (see Fig. 7.1). Together with the results we present a standard deviation of the obtained results.

One can see that accuracy of algorithm ARS for the decision class *NO* is higher than the accuracy of the algorithm RS for analyzed data set. The decision class *NO* is smaller than the class *YES*. It represents atypical cases in whose recognition we are most interested in (dangerous driving a vehicle on a highway).

Table 7.2 shows the learning time and the number of decision rules induced for the considered classifiers. In the case of the algorithm ARS, we present the average number of decision rules over all concepts from the relationship diagram (see Fig. 7.1).

One can see that the learning time for ARS is much shorter than for RS and the average number of decision rules (over all concepts from the relationship diagram) for ARS algorithm is much lower than the number of decision rules induced for RS.

The experiments showed that classification quality obtained through classifiers based on AR-schemes is higher than classification quality obtained through traditional classifiers based on decision rules (especially in the case of the class *NO*). Apart from that the time spent on classifier construction based on AR-schemes is shorter than when constructing classical rule classifiers. Also, the structure of a single rule classifier (inside the ARS classifier) is less complicated than the structure of RS classifier (a considerably smaller average number of decision rules). It is worth noticing that the the performance of the ARS classifier is much more stable than the RS classifier because of the differences in data in samples supplied for learning (*e.g.*, to change the simulation scenario).

7.3 Methods of Approximation of Spatio-temporal Concepts

In this section, we propose a method of approximating concepts from hierarchical ontology when a higher ontology level concept is a spatio-temporal concept (it requires observing changes of complex objects over time) defined on a set of the same objects as the lower ontology level concepts; at the same time, the lower ontology level concepts are spatial concepts only. This case concerns a situation when during an observation of a single object in order to capture its behavior described by a higher ontology level concept, we have to observe it longer than it requires to capture behaviors described by lower ontology level concepts. For example, in case of data sets generated from the traffic simulator, lower ontology level concepts may concern simple vehicle behaviors such as *small increase in speed*, *small decrease in speed* or *small move towards the left lane*. However, the higher ontology level concept may be a more complex concept as, *e.g.*, *acceleration in the right lane*. Let us notice that determining whether a vehicle accelerates in the right lane requires its observation for some time called *a time window*. On the other hand, determining whether a vehicle speed increases in the right lane requires only a registration of the speed of a vehicle in two neighboring instants (time points) only. That is why spatio-temporal concepts are more difficult to approximate than spatial concepts whose approximation does not require observing changes of objects over time.

Similarly to spatial concept approximation (see Section 7.2), the method of concept approximation described in this subsection is an example of the general methodology of approximating concepts from ontology described in [4]. Its specificity is, therefore, the domain knowledge usage expressed in the form of a concept ontology and rough set method application, mainly in terms of application of classifier construction methods. However, in this case more complex ontologies are used, and they contain both spatial and spatio-temporal concepts.

The starting point for the method proposed is a remark that spatio-temporal concept identification requires an observation of a complex object over a longer period of time called *a time window* (see [4]). To describe complex object changes in the time window, the so-called *temporal patterns* (see [4]) are used, which are defined as functions determined on a given time window. These patterns, being in fact formulas from a certain language, also characterize certain spatial properties of the complex object examined, observed in a given time window. They are constructed using lower ontology level concepts and that is why identification whether the object belongs to these patterns requires the application of classifiers constructed for concepts of the lower ontology level. Moreover, temporal patterns are often defined using queries with binary answers such as *Yes* or *No*. For example, in the case of road traffic we have exemplary temporal patterns such as *Did vehicle speed increase in the time window?*, *Was the speed stable in the time window?*, *Did the speed increase before a move to the left lane occurred?*, or *Did the speed increase before a speed decrease occurred?*. We assume that any temporal pattern ought to be defined

by a human expert using domain knowledge accumulated for the given complex dynamical system.

On a slightly higher abstraction level, the spatio-temporal concepts (also called *temporal concepts*) are directly used to describe complex object behaviors (see [4]). Those concepts are defined by an expert in a natural language and they are usually formulated using questions about the current status of spatio-temporal objects, *e.g.*, *Does the vehicle examined accelerate in the right lane?*, *Does the vehicle maintain a constant speed during lane changing?* The method proposed here is based on approximating temporal concepts using temporal patterns with the help of classifiers. In order to do this, a special decision table is constructed called *a temporal concept table* (see [4]). In case of method presented in this chapter, the rows of this table represent the parameter vectors of lower level ontology concepts observed in a time window. Columns of this table (apart from the last one) are determined using temporal patterns. However, the last column represents membership of an object, described by parameters (features, attributes) from a given row, to the approximated temporal concept (see Fig. 7.8).

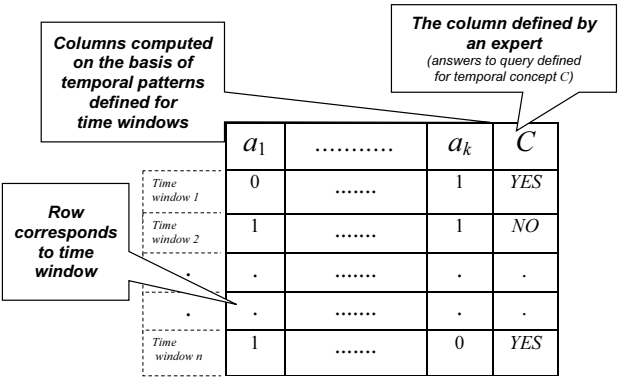


Fig. 7.8 The scheme of a temporal concept table

It is worth noticing that the presented above approach to temporal concept approximation can be extended to the case when higher ontology level concepts are defined on a set of objects which are structured objects in relation to objects (examples) of the lower ontology level concepts, that is, the lower ontology level objects are parts of objects from the higher ontology level. This case concerns a situation when during a structured object observation, which serves the purpose of capturing its behavior described by a higher ontology level concept, we must observe this object longer than it is required to capture the behavior of a single part of the structured object described by lower ontology level concepts (see [4] for more details).

7.4 Methods of Behavioral Pattern Identification

Temporal concepts may be treated as nodes of a certain directed graph which is called a *behavioral graph*. Links (directed edges) in this graph are the temporal relations between temporal concepts meaning a temporal sequence of satisfying two temporal concepts one after another. These graphs may be used to represent and identify the so-called behavioral patterns which are complex concepts concerning dynamic properties of complex objects expressed in a natural language depending on time and space. Examples of behavioral patterns may be: *overtaking on the road*, *driving in a traffic jam*, *behavior of a patient connected with a high life threat*. These types of concepts are much more difficult to approximate even than many temporal concepts. Fig. 7.9 presents an example of behavioral graph for a single object-vehicle exhibiting a behavioral pattern of vehicle while driving on a road. In this behavioral graph, for example, connections between the nodes *Acceleration on the right lane* and *Acceleration and changing lanes lanes from right to left* indicate that after an acceleration in the right lane, a vehicle can change to the left lane (maintaining its acceleration during both time windows).

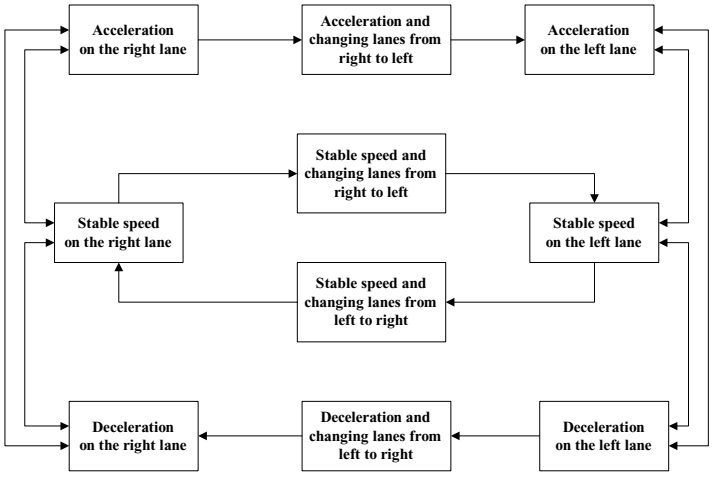


Fig. 7.9 A behavioral graph for a single object-vehicle

In [4] a new method of behavioral pattern identification is presented which is based on interpreting the behavioral graph of a complex object as a complex classifier enabling identification of a behavioral pattern described by this graph. This is possible based on the observation of the complex object behavior for a longer time and checking whether the behavior matches the chosen behavioral graph path. If this is so, then it is determined if the behavior matches the behavioral pattern represented by this graph, which enables a detection of specific behaviors of complex objects.

In order to test the quality and effectiveness of classifier construction methods based on behavioral patterns, there have been performed experiments on data generated from the road simulator and medical data connected to detection of higher-death risk in infants suffering from the respiratory failure (see [4,6,8,9]). The experiments showed that the algorithmic methods presented in this chapter provide very good results in detecting behavioral patterns and may be useful with complex dynamical systems monitoring.

Additionally, in this chapter we present recent results of experiments investigating membership of patients with stable coronary heart disease to behavioral pattern related to risk of *sudden cardiac death* (SCD). Using Holter ECG recordings and well known predictors of SCD, the concepts of SCD risk intensity were defined (see Section 7.4.1).

7.4.1 Risk Pattern Identification in Medical Data

An identification of some behavioral patterns can be very important for identification or prediction of behavior of complex dynamical system, especially when behavioral patterns describe some dangerous situations. In this case, we call such behavioral patterns as *risk patterns* and we need some tools for their identification (see, *e.g.*, [18]). If in the current situation some risk patterns are identified, then the control object (a driver of the vehicle, a medical doctor, a pilot of the aircraft, etc.) can use this information to adjust selected parameters to obtain the desirable behavior of the complex dynamical system. This can make it possible to overcome inconvenient or unsafe situations. For example, a very important element of the treatment of the patients with coronary heart disease is the appropriate assessment of the risk of SCD. The appropriate assessment of this risk leads to the decision of particular method and level of treatment. Therefore, if some complex behavior of a patient that causes a danger of SCD is identified, we can try to change her/his behavior by using some other methods of treatment (may be more radical) in order to avoid the patients's death. We describe how the presented approach can be applied to identify the patient's SCD risk caused by coronary heart disease (see next subsections). In this approach, a given patient is treated as an investigated complex dynamical system, whilst coronary heart disease is treated as a complex object changing and interacting over time with its environment.

7.4.2 Medical Temporal Patterns

Data sets used for complex object information storage occurring in a given complex dynamical system may be represented using information systems (see, *e.g.*, [4,32]). This representation is based on representing individual complex objects by object (rows) of information system and information system attributes represent the properties of these objects at the current time point.

The concepts concerning properties of complex objects at the current time point (spatial concepts) can be defined on the basis of domain knowledge by human experts and can be approximated by properties (attributes) of these objects at the current time point (for instance, using the standard rough set approach to classifier construction [4, 32]).

The concepts concerning properties of complex objects at the current time point in relation to the previous time point are a way of representing very simple behaviors of the objects. However, the perception of more complex types of behavior requires the examination of behavior of complex objects over a longer period of time. This period is usually called the time window, which is to be understood as a sequence of objects of a given temporal information system (a kind of information system with special attribute represents time) registered for the established complex object starting from the established time point over the established period of time or as long as the expected number of time points are obtained. Therefore, learning to recognize complex types of behavior of complex objects with use of gathered data as well as the further use of learned classifiers to identify the types of behavior of complex objects, requires working out of the mechanisms of extraction of time windows from the data and their properties. Hence, if we want to predict such more complex behaviors or discover a behavioral pattern, we have to investigate values of attributes registered in the current time window. Such investigation can be expressed using temporal patterns (see Section 7.3). For example, in the case of the medical example one can consider patterns expressed by following questions: "Did HRV increase in the time window?", "Was the heart rate stable in the time window?", "Did ST interval level increase?", or "Was the QT segment time higher then the right time at any point in time window?". Notice that all such patterns ought to be defined by a human, medical expert using domain knowledge accumulated for the coronary heart disease.

7.4.3 Medical Risk Pattern

The temporal patterns can be treated as new features that can be used to approximate temporal concepts. In the case of the treatment of patient with cardiovascular failure, one can define temporal concepts such as "Is the patient's SCD risk on low level?", "Is the patient's SCD risk on medium level?", or "Was high SCD risk detected?".

Temporal concepts defined for objects from a complex dynamical system and approximated by classifiers, can be treated as nodes of a graph called a behavioral graph, where connections between nodes represent temporal dependencies. Fig. 7.10 presents a behavioral graph for a single patient exhibiting a behavioral pattern of patient by analysis of the circulatory system failure caused by coronary heart disease. This graph has been created on the basis of observation of medical data sets and known factors for SCD risk stratification. In this behavioral graph, for

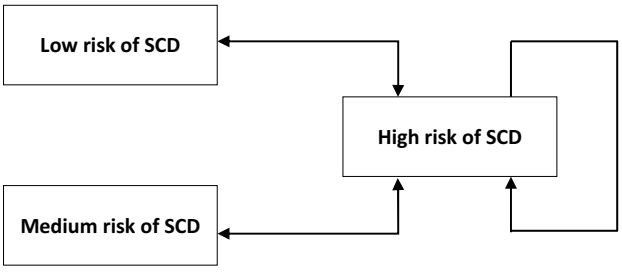


Fig. 7.10 A behavioral graph of SCD risk by analyzing cardiovascular failure

example, connection between node “Medium SCD risk" and node “High SCD risk" indicates that after some period of progress in cardiovascular failure on medium level, a patient can change his behavior to the period, when progress in cardiovascular failure is high.

This behavioral graph is an example of risk pattern. If the patient matches the “Low SCD risk" concept in the first time window, “Medium SCD risk" in the following window, after which his state returned to the previous one, then the patient’s behavior does not match this behavioral graph.

7.4.4 Experiments with Medical Data

The next experiments were performed on data obtained from Second Department of Internal Medicine, Collegium Medicum, Jagiellonian University, Cracow, Poland. The data collection contains informations about 95 patients with stable coronary heart disease, collected between 2006 and 2009. It includes a detail description of clinical status (age, sex, diagnosis), coexistent diseases, pharmacological management, the laboratory tests outcomes (level of cholesterol, troponin I, LDL — low density lipoproteins), Holter ECG recordings (long term, 24-hour signals) and various Holter-based indices such as: ST-segment deviations, HRV, arrythmias, or QT dispersion. Sinus (normal) rhythm was observed in 73 patients, while 22 patients had permanent FA (atrial fibrillation). Two 24-hour Holter ECG recordings were performed using Aspel’s HolCARD 24W system. There was coronary angiography after first Holter ECG.

All data was imported to *Infobright Community Edition* (ICE) environment (see [48]). ICE is an open source software solution designed to deliver a scalable data warehouse optimized for analytic queries (data volumes up to 50 TB, market-leading data compression (from 10:1 to over 40:1)). Database schema was designed to store all information about patients, including supplementing the target database in the future. For further processing data have been imported into the RoughICE environment.

For the experiment, one table with 744 objects was formed. Each object (row) contains information about the parameters of one patient with one hour of observation, being the average hourly values of observed parameters.

The experiments were performed in order to predict the behavioral pattern related to a high risk of SCD. This pattern was defined by medical experts on the base of well-known predictors of SCD. The evaluation of SCD risk includes: advanced age, male sex, coexisting diseases like DM (diabetes mellitus), HA (arterial hypertension), history of stroke, previous myocardial infarction, CRP (C-phase reaction protein) level, depressed LVEF (left ventricular ejection fraction), presence of arrhythmias and ischaemias, high heart rate, decreased HRV, and HRT (heart rate turbulence). Taking into account simplicity of example model and temporal aspect of patterns, in this approach only few factors were chosen, such as HRV index: SDNN (standard deviation of NN intervals — normal to normal beat segments), average heart rate, ST interval decrease, and QT segment changes. HRV parameter was calculated upon one hour period, though usually it is analyzed within 24 hour interval. Because of the lack of the appropriate data, such standard analyzes were not performed in this experiment.

We have applied the *train-and-test* method. However, because of the specificity of the analyzed data the method of data division differed slightly from the standard method. Namely, in each experiment the whole patient set was randomly divided into two groups (training group: 60% of patients and testing group: 40% of patients).

As a result of the above mentioned division of patients into training and testing ones, each of these parts made it possible to create time windows having duration of 2 time points (2 h of patients observation) and sequences of such time windows (training part: approximately 400 time windows, testing part: approximately 270 sequences of time windows). Time windows created on the basis of training patients created a training table for a given experiment, while time windows sequences created on the basis of tested patients created a test table for the experiment.

In order to determine the standard deviation of the obtained results each experiment was repeated for 10 random divisions of the whole data set.

A single experiment was as follows (see also Figure 7.11). First, for the training data the family of all time windows having duration of 2 time points were generated. Then, on the basis of temporal patterns proposed by experts, the behavioral graph from Figure 7.10 and the additional domain knowledge (represented by experts scripts in RoughICE — see [42] for more details) the temporal pattern tables were constructed for all concepts from the behavioral graph from the Figure 7.10. Then for all these tables a family of stratifying classifiers were generated that are able to classify objects (patients) to different concepts from the sequence of ordered layers. The first layer in this sequence represents objects which, without any doubt do not belong to the concept. The next layers in the sequence represent objects belonging to the concept more and more certainly. The last layer in this sequence represents objects certainly belonging to the concept (see [4] for more details). Next, *a complex classifier* was constructed on the basis of stratifying classifiers family that allow us to predict membership of a particular time window to various temporal concepts from the behavioral graph (see Figure 7.11). The main idea of working

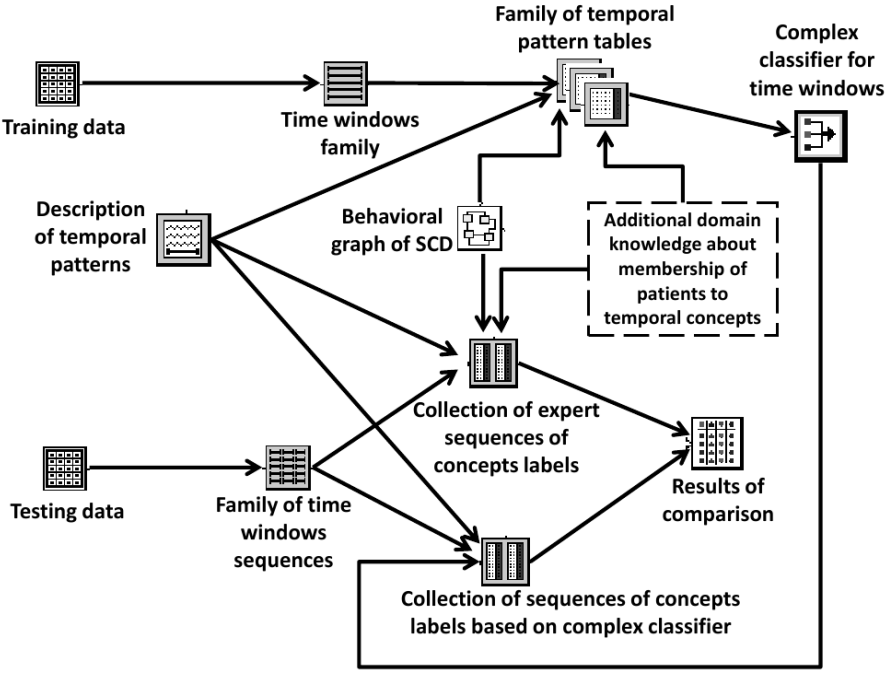


Fig. 7.11 A general scheme of experiments for the risk pattern of SCD

of such classifier is as follows. A given time window is classified to such temporal concept that a stratifying classifier corresponding to this concept classifies the time window to the top layer of all layers proposed by stratifying classifiers. Next, for the testing data, the family of all sequences of time windows having duration of 2 time windows were generated (the length of every time widow was 2 just as in the case of time windows for training data set). Then, for every sequence from this family, a sequence of labels of temporal concepts was generated in two different methods (see below). Such sequence of labels can be interpreted as a path of nodes from the behavioral graph. In this interpretation, the sequence of labels represents the answer if a given sequence matches the behavioral graph (behavioral pattern) from the figure 7.10.

The first method of the sequence of concepts labels generation is based on temporal patterns proposed by experts, the behavioral graph from Figure 7.10, and the additional domain (expert) knowledge about membership of patients to temporal concepts from behavioral graph. Therefore, this method we call as a *expert method* and the sequence of concepts labels generated with usage of this method as *an expert sequence of concepts labels*.

The second method of the sequence of concepts labels generation is based on the complex classifier generated for the training data. For a given sequence of time windows, the complex classifier has been used to generation of the concepts label

for every time window separately. In this way, we obtain the sequence of concepts labels, that can be also treated as a potential path of nodes from the behavioral graph. This method we call as an *classifier method* and the sequence of concepts labels generated with usage of this method we call as *a sequence of concepts labels based on classifier*.

Our method of presented approach evaluation is based on comparison of the expert and the classifier methods results. For a given sequence of time windows *stw*, the accuracy of identification of the sequence *stw* is computed in the following way:

- if the expert sequence of concepts labels computed for *stw* matches a path from the behavioral graph and a sequence of concepts labels based on the classifier also matches a path from the behavioral graph, the accuracy of identification of the sequence *stw* is equal 1,
- if the expert sequence of concepts labels computed for *stw* matches a path from the behavioral graph and a sequence of concepts labels based on classifier does not match a path from the behavioral graph, the accuracy of identification of the sequence *stw* is equal 0,
- if the expert sequence of concepts labels computed for *stw* does not match a path from the behavioral graph and a sequence of concepts labels based on classifier matches a path from the behavioral graph, the accuracy of identification of the sequence *stw* is equal 0,
- if the expert sequence of concepts labels computed for *stw* does not match a path from the behavioral graph and a sequence of concepts labels based on classifier does not match a path from the behavioral graph, the accuracy of identification of the sequence *stw* is equal 1.

The accuracy of identification of the whole family of time windows sequences is computed as an average value of accuracies computed for every sequence separately.

Table 7.3 shows the results of applying this algorithm for the concept related to the risk pattern of SCD. We present the accuracy, the coverage, the accuracy for positive examples (the expert sequence of concepts labels matches a path from the behavioral graph) and negative examples (the expert sequence of concepts labels computed does not match a path from the behavioral graph), the coverage for positive and negative examples and the real accuracy (Real accuracy = Accuracy * Coverage). Together with the results we present a standard deviation of the obtained results.

Table 7.3 Results of experiments for the risk pattern of SCD

Decision class	Accuracy	Coverage	Real accuracy
Yes (the high risk of SCD)	0.953 ± 0.048	1.0 ± 0.000	0.953 ± 0.048
No (the low risk of SCD)	0.971 ± 0.010	1.0 ± 0.000	0.971 ± 0.010
All classes (Yes + No)	0.967 ± 0.013	1.0 ± 0.000	0.967 ± 0.013

Notice, that the accuracy of decision class Yes in medical statistics [2] is called a *sensitivity* (the proportion of those cases having a true positive test result of all

positive cases tested), whereas the accuracy of decision class No is called a *specificity* (the proportion of true negatives of all the negative samples tested). We see both main parameters of our classifier (i.e., sensitivity and specificity) are sufficiently high.

Experimental results showed that the suggested method of behavioral patterns identification gives good results, also in the opinion of medical experts (compatible enough with the medical experience) and may be applied in medical practice as a supporting tool for medical diagnosis and treatment evaluation.

Finally, let us notice that the specific feature of the methods considered here is not only high accuracy (with low standard deviation) but also very high coverage (equal 1.0).

7.5 Methods of Automated Planning

In this section, we present a method of automated planning behavior planning for complex object. This method also works on the basis of data sets and a domain knowledge represented by a concept ontology. A crucial novelty in the method proposed here, in comparison with the already existing ones, is the fact that performing actions according to plan depends on satisfying complex vague spatio-temporal conditions expressed in a natural language, which leads to the necessity of approximation of these conditions as complex concepts. Moreover, these conditions describe complex concept changes which should be reflected in the concept ontology.

Behavior of unstructured complex objects (meaning those which may be treated as indivisible wholes) is modeled using the so-called *planning rules* being formulas of the type: *the state before performing an action* $\rightarrow$ *action* $\rightarrow$ *state* 1 *after performing an action* | ... | *state k after performing an action*, which are defined on the basis of data sets and a domain knowledge (see [4]). Each rule includes the description of the complex object state before applying the rule (that is, before performing an action), expressed in a language of features proposed by an expert, the name of the action (one of the actions specified by the expert which may be performed at a particular state), and the description of sequences of states which a complex object may turn into after applying the action mentioned above. It means that the application of such a rule gives indeterministic effects, i.e., after performing the same action the system may turn into different states.

Let us consider the planning rule from Fig. 7.12. This is the planning rule for treating RDS (respiratory distress syndrome) obtained from domain knowledge (see [4, 7]). The rule may be applied when RDS with very severe hypoxemia is present. The application of the rule consists in performing a medical action utilizing the respirator in the MAP3 mode (see [4, 7] for more medical details). As an effect of the application of this action at the following time point of observation (*e.g.*, the following morning), the patient's condition may remain unchanged or improve so as to reach the condition of RDS with severe hypoxemia.

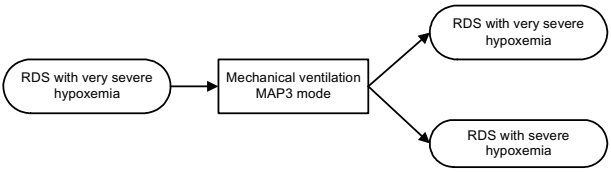


Fig. 7.12 The medical planning rule

All planning rules may be represented in a form of the so-called *planning graphs* whose nodes are state descriptions (occurring in predecessors and successors of planning rules) and action names occurring in planning rules. Let us consider planning graph from the Fig. 7.13, where the states are represented using ovals, and actions are represented using rectangles. Each link between the nodes of this graph represents a time dependencies. For example, the link between state s_1 and action a_1 tells us that in state s_1 of the complex object action a_1 may be performed, whereas the link between action a_1 and state s_3 means that after performing action a_1 the state of the complex object may change to s_1. An example of a path in this graph is sequence (a_2, s_2, a_3, s_4) whereas path $(s_1, a_2, s_2, a_3, s_3)$ is an exemplary plan in this graph.

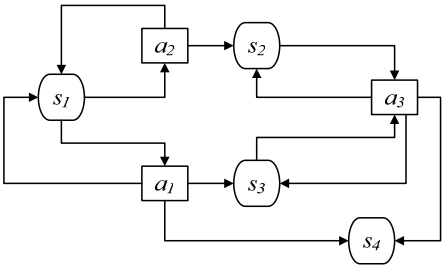


Fig. 7.13 An exemplary planning graph

In the graphical interpretation, solving the problem of automated planning is based on finding a path in the planning graph from the initial state to an expected final state. It is worth noticing that the conditions for performing an action (object states) are described by vague spatio-temporal complex concepts which are expressed in the natural language and require an approximation. For example, Fig. 7.14 presents a solution to the problem of finding a plan bringing state s_1 to state s_4 in the planning graph from Fig. 7.13.

For specific applications connected with the situation when it is expected that the proposed plan of a complex object behavior is to be strictly compatible with the determined experts' instructions (*e.g.*, the way of treatment in a specialist clinic is to

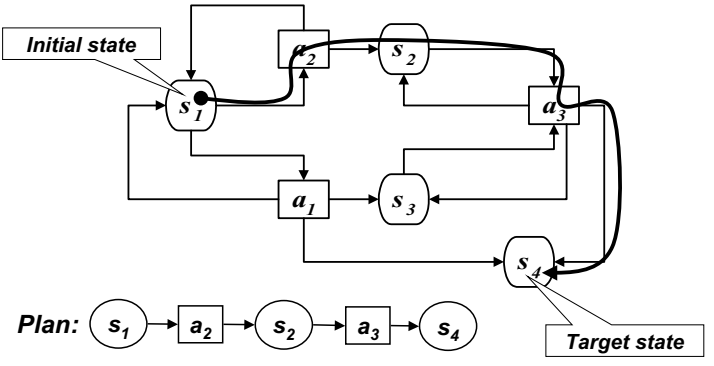


Fig. 7.14 The output for the planning problem

be compatible with the treatment schemes used there), there has also been proposed an additional mechanism enabling to resolve the nondeterminism occurring in the application of planning rules. This mechanism is an additional classifier based on data sets and domain knowledge. Such classifier (called *a resolving classifier*) suggests the action to be performed in a given state and show the state which is the result of the indicated action. A resolving classifier is a kind of stratifying classifier and is constructed on the basis of *resolving table* (see Fig. 7.15 and [4] for more details).

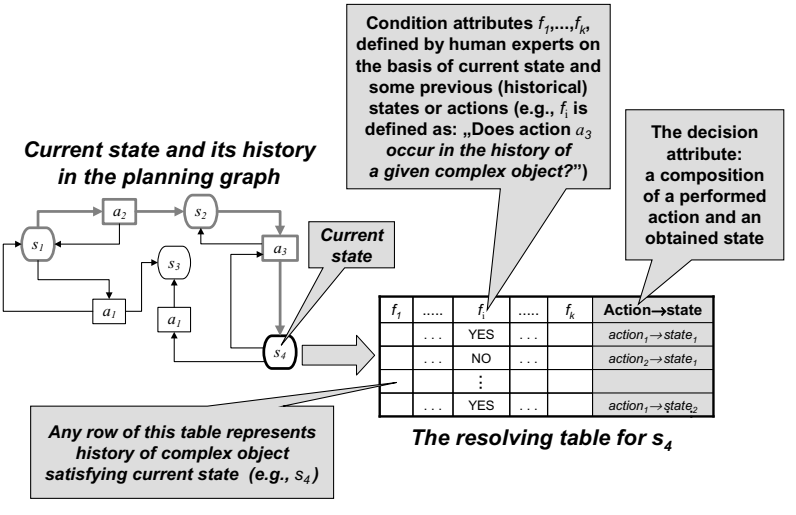


Fig. 7.15 The scheme of construction of the resolving table for a given state

7.5.1 Automated Planning for Structured Complex Objects

In planning the behavior of structured objects, an effective planning of the behaviors of all objects which are parts of these objects at the same time is not possible. Therefore, in such cases the behavior of all objects which are parts of a structured object is planned separately. However, this approach to planning of the behavior for a structured object requires a certain synchronization of the plans constructed for individual parts in such a way that these plans would not contradict each other and even complement each other in order to plan the best behavior for a structured object. For example, the treatment of illness A which is the resultant of two illnesses B and C requires such illnesses B and C treatment that the treatments of both illnesses would not be contradictory to each other, but even support and complement each other. For example, it may happen that in treating illness B a certain medicine M_1 may be used which is usually an appropriate medicine but it may be applied only when illness C does not occur. Hence, the synchronization of both illnesses' treatment should exclude the application of medicine M_1. In a different situation it may happen that as a result of application of medicine M_2 for illness C the treatment of illness B is safer, for instead of giving a certain strong medicine M_3, which has negative side effects, it is enough to give a safer medicine M_4 which leads to the same improvement in the patient's condition as in the case of giving medicine M_3.

The automated planning method for unstructured objects has been generalized also in the case of planning of the behavior of structured objects (consisting of parts connected with one another by dependencies) (see [4]). The generalization is based on the fact that on the level of a structured object there is an additional planning graph defined where there are double-type nodes and directed edges between the nodes. The nodes of the first type describe vague features of states (meta-states) of the whole structured object, whereas the nodes of the second type concern complex actions (meta-actions) performed by the whole structured object (all its constituent parts) over a longer period of time (a time window). The edges between the nodes represent temporal dependencies between meta states and meta-actions as well as meta-actions and meta states.

In Fig. 7.16, we present an exemplary planning graph for a structured object, that is a group of four diseases: sepsis, Ureaplasma, RDS, and PDA, related to the planning of the treatment of the infant during the respiratory failure. This graph was created on the basis of observation of medical data sets and with support of human experts (see [4, 7] for more medical details).

As we see, there are two kinds of nodes in the planning graph for structured object, namely, *meta states nodes* (denoted by ovals) that represent the current state of a structured object specified as complex concepts by a human expert in natural language, and *meta action nodes* (denoted by rectangles) that represent actions defined for structured objects.

The major difference between the planning graph for the unstructured complex object and the planning graph for the structured object is that in the last one instead of actions performed at a single time point meta-actions occur which are performed over a longer period of time, that is, a time window.

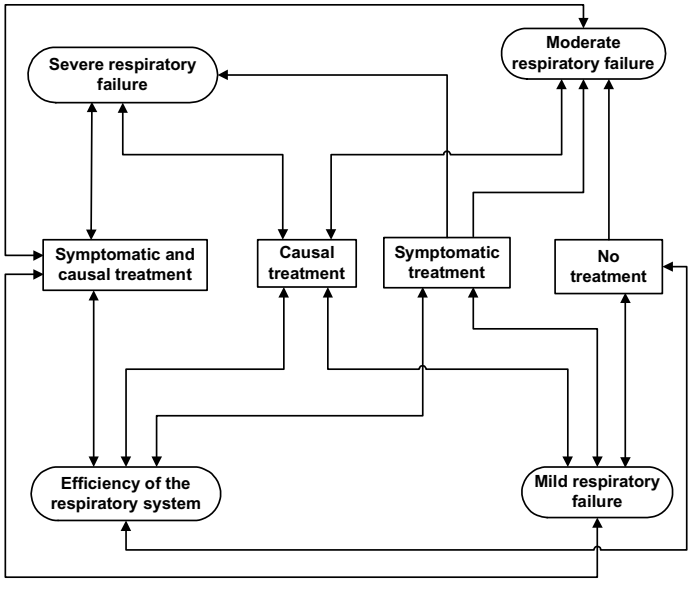


Fig. 7.16 A planning graph for the treatment of infants during the respiratory failure

At the beginning of planning for a structured object, we have to identify the current meta state of this object. Any meta state node from a planning graph for structured objects can be treated as a complex spatio-temporal concept that is specified by a human expert in natural language. Such concepts can be approximated by classifiers using data sets and domain knowledge accumulated for a given complex dynamical system. Similarly to states from the planning graph for unstructured complex objects, any state from the planning graph for structured objects can be approximated as a temporal concept for structured object using method from Section 7.3. As a result, it is possible to recognize the initial state at the beginning of planning for a particular structured object.

Similarly to the previous case of unstructured objects, planning of a structured object behavior is based on finding a path in a planning graph from the initial meta-state to the expected final meta state; and, at the same time, each meta-action occurring in such a path must be planned separately on the level of each constituent part of the structured object. In other words, it should be planned what actions each part of a structured object must perform in order for the whole structured object to be able to perform the meta-action which has been planned. For example, in the case of the treatment of infants with respiratory failure, if the infant is suffering from severe respiratory failure, we try to change the patient status using some methods of treatment to change its status to moderate or mild respiratory failure. However, any meta action from such constructed path should be checked on the lower level, i.e., on the level of any part of the structured object separately, if such action can be realized in practice in case of particular part of this structured object. In other words, it means

that for any part of the structured object, the sequence of action should be planned in order to obtain meta-action on the level of the structured object.

The plan of execution of a single meta-action, which consists of short plans which execute this meta-action on the levels of individual parts of the structured object, is called *a g-plan* (see [4]). The g-plan is, thus, a family of plans assigned to be executed for all parts of the established structured object.

Let us notice that determining the plan for a structured object requires not only determining sets of plans for all parts of the structured object but also synchronizing them in time. In practise, all constructed plans for objects (parts) belonging to a given structured object should be compatible. Therefore, during planning a meta-action for a structured object, we use a special tool for verifying the compatibility of plans generated for all members of a structured object. This verification can be performed by using some special decision rules that we call *elimination rules*. Such rules make it possible to eliminate combination of plans that are not compatible relative to domain knowledge. This is possible because elimination rules describe all important dependencies between plans that are joined together. If any combination of plans is not consistent with any elimination rule, then it is eliminated. A set of elimination rules can be specified by human experts or can be computed from data sets. In both of these cases, we need a set of attributes (features) defined for a single plan that are used for explaining elimination rules. Such attributes are specified by human experts on the basis of domain knowledge and they describe some important features of the plan (generated for some part of structured object) with respect to proper joining of a plan with plans generated for other parts of structured object. These features are used as a set of attributes in the special table that we call *an elimination table*. Any row of an elimination table represents information about features of plans assigned for structured objects from the training data. For example, the respiratory failure may be treated as a result of four following diseases: RDS, PDA, sepsis, and Ureaplasma. Therefore, treating respiratory failure requires simultaneous treatment of all of these diseases. This means that the treatment plan of respiratory failure comes into existence by joining the treatment plans for diseases RDS, PDA, sepsis, and Ureaplasma, and at the same time the synchronization of the plans is very important. In this chapter, one of the synchronizing tools for this type of plans is the elimination table. In constructing the elimination table for treatment of respiratory failure, patterns describing the properties of the joint plans are needed. Moreover, planning graphs for all four diseases are necessary. In Fig. 7.17 the planning graph for RDS treatment is shown. In a very similar way the features of treatment plans for PDA, sepsis, and Ureaplasma diseases may be defined.

On the basis of the elimination table, a set of elimination rules can be computed that can be used to eliminate inappropriate plan arrangements for individual parts of the structured object. So, the set of elimination rules can be used as a filter of inconsistent combinations of plans generated for members of groups. Any combination of plans is eliminated when there exists an elimination rule that is not supported by features of a combination, while the combination matches a predecessor of this rule. In other words, a combination of plans is eliminated when the combination matches

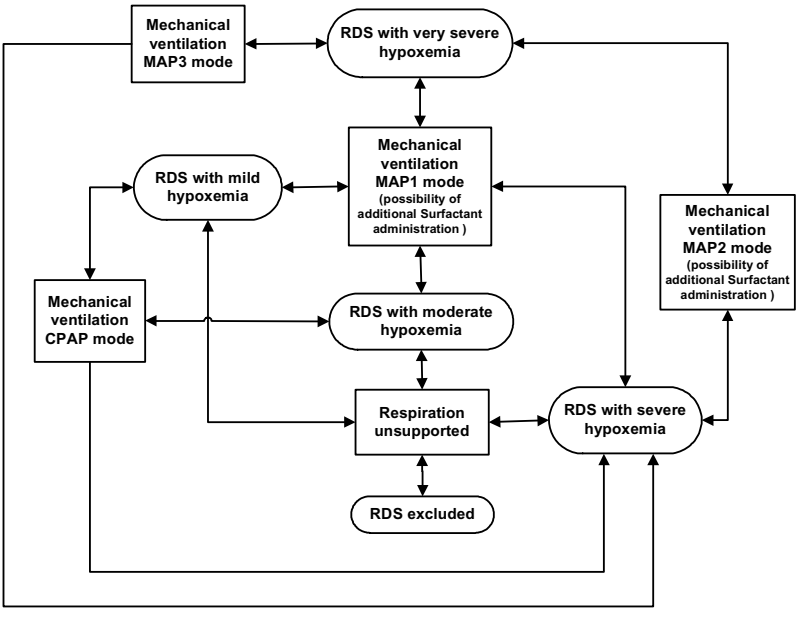


Fig. 7.17 A planning graph for the treatment of infants during the RDS

to the predecessor of some elimination rule and does not match the successor of a rule.

Fig. 7.18 shows the scheme of elimination rules of not acceptable g-plans constructed in the case of the treatment of respiratory failure, which is a result of the four following diseases: sepsis, Ureaplasma, RDS, and PDA.

As we see, for any attribute from the elimination table, we compute the set of rules with minimal number of descriptors treating this attribute as a decision attribute. In this way, we obtain a set of dependencies in the elimination table explained by decision rules. In practice, it is necessary to filter elimination rules to remove the rules with low support, because such rules can be too strongly matched to the training data.

On the basis of the set of elimination rules, *an elimination classifier* may be constructed that enable elimination of inappropriate plan arrangements for individual parts of the structured object.

If the combination of plans for parts of the structured object is consistent (it was not eliminated by elimination rules), we should check if the execution of this combination allows us to realize the expected meta action from the level of structured objects. This can be done by a special classifier constructed for a table called a *meta action table*. The structure of a meta action table is similar to the structure of an elimination table, i.e., attributes are defined by human experts, where rows represent information about features of plans assigned for parts of exemplary structured objects from the training data. In addition, we add to this table a decision attribute. Values of such decision attributes represent names of meta actions which are

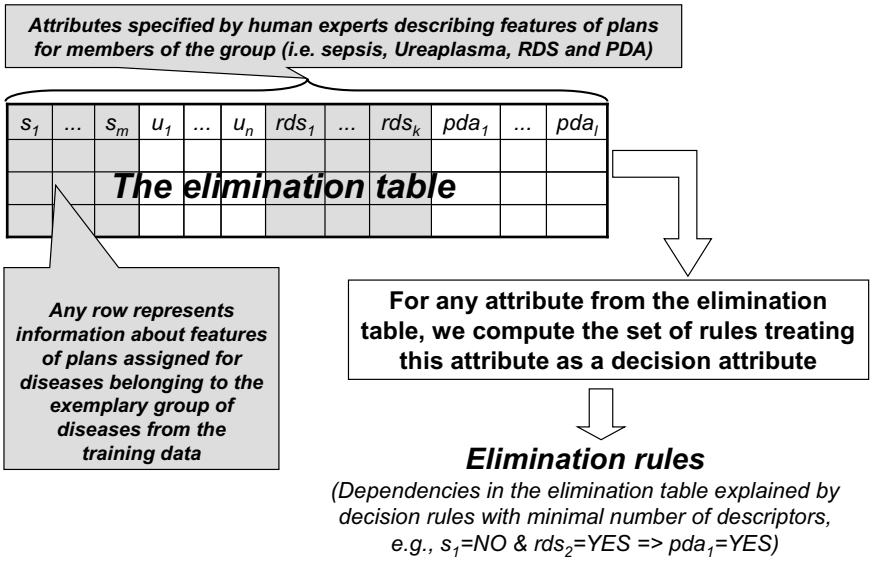


Fig. 7.18 The scheme of construction of elimination rules for group of four diseases: sepsis, Ureaplasma, RDS and PDA

realized as an effect of the execution of plans described in the current row of a training table.

The classifier computed for an action table makes it possible to predict the name of a meta action for a given combination of plans from the level of parts of a structured object. The last step is the selection of combinations of plans that makes it possible to obtain a target meta action with respect to a structured object (see Fig. 7.19).

After planning the selected meta action from the path of actions from the planning graph (for a structured object), the system begins the planning of the next meta action from this path. The planning is stopped, when the planning of the last meta action from this path is finished.

7.5.2 Estimation of the Similarity between Plans

In construction and application of classifiers approximating complex spatio-temporal concepts, there may appear a need to construct, with a great support of the domain knowledge, a similarity relation of two elements of similar type, such as complex objects, complex object states, or plans generated for complex objects. Hence, in the paper [4], a new method of similarity relation approximation has been proposed which is based on the use of data sets and a domain knowledge expressed mainly in the form of a concept ontology. We apply this method, among other things, to verify

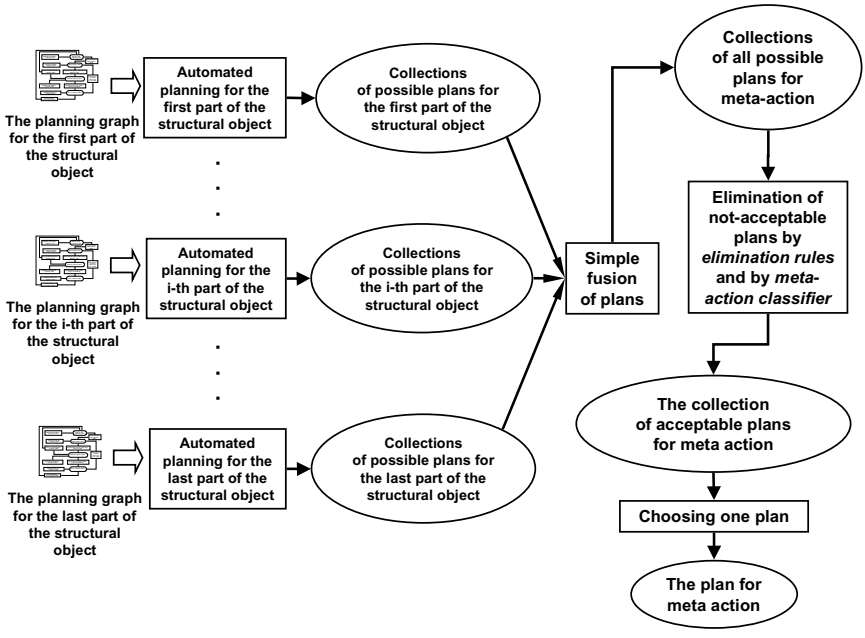


Fig. 7.19 The scheme of meta action planning

automated planning methods, that is, to compare the plan generated automatically with the plan suggested by experts from a given domain.

The problem of inducing classifiers for similarity relations is one of the challenging problems in data mining and knowledge discovery (see bibliography from [4]). The existing methods are based on building models for similarity functions using simple strategies for fusion of local similarities. The optimization of the assumed parameterized similarity formula is performed by tuning parameters relative to local similarities and their fusion. For instance, if we want to compare two medical plans of treatments, *e.g.*, one plan generated automatically by our computer system and another one proposed by medical expert, we need a tool to estimate the similarity. This problem can be solved by introducing a function measuring the similarity between medical plans. For example, in the case of our medical data, a formula is used to compute a similarity between two plans as the arithmetic mean of similarity between all corresponding pairs of actions (nodes) from both plans, where the similarity for the single corresponding pair of actions is defined by a consistence measure of medicines and medical procedures comprised in these actions. For example, let $M = \{m_1, ..., m_k\}$ be a set consisting of k medicines. Let us assume that actions in medical plans are specified by subsets of M. Hence, any medical plan P determines a sequence of actions $\mathcal{A}(P) = (A_1, ..., A_n)$, where $A_i \subseteq M$ for $i = 1, \ldots, n$ and n is the number of actions in P. In our example, the similarity between plans is defined by a similarity function *Sim* established on pairs of medical plans (P_1, P_2) (of the same

length) with the sequences of actions $\mathcal{A}(P_1) = (A_1, ..., A_n)$ and $\mathcal{A}(P_2) = (B_1, ..., B_n)$, respectively as follows

$$Sim(P_1, P_2) = \frac{1}{n} \sum_{i=1}^{n} \frac{|A_i \cap B_i| + |M \setminus (A_i \cup B_i)|}{|M|}.$$

However, such an approach seems to be very abstract and ad hoc, because it does not take into account any deeper knowledge about the similarity of plans, *e.g.*, domain knowledge. Whereas, the similarity relations for real-life problems are usually more complex objects, i.e., their construction from local similarities cannot be obtained by simple fusion functions. Hence, such similarity relations cannot be approximated with the satisfactory quality by employing the existing simple strategies. For this reason we treat this similarity measure, *Sim*, only as an example and do not take into account in our further research (and in our proposed method). Whereas, to support the process of similarity relation approximation, we propose to use domain knowledge represented by concept ontology expressed in natural language. The ontology consists of concepts used by expert in his explanation of similarity and dissimilarity cases. Approximation of the ontology makes it possible to obtain some relevant concepts for approximation of the similarity relation.

7.5.3 Ontology of the Similarity between Plans

According to the domain knowledge, it is quite common, that there are many aspects of similarity between plans. For example, in case of comparison of medical plans used for the treatment of infants with respiratory failure, we should take into consideration, *e.g.*, the similarity of the antibiotics use, the ventilation mode and the similarity of PDA closing (see [4] for mor medical details). Moreover, every aspect of the similarity should be understood in a different way. For example, in estimation of the similarity in the antibiotic treatment, it should be evaluated the kind of antibiotic, as well as the time of administration. Therefore, it is necessary to investigate and take into account all incompatibilities of the antibiotic use between corresponding pairs of nodes from both plans. Excessive doses are rather acceptable (based on expert knowledge), whilst the lack of medicine (if it is necessary) should be taken as a very serious mistake. In such situation, the difference in our assessment is estimated as very significant. A bit different interpretation of similarity should be used in case of the ventilation. As in antibiotic use, we investigate all incompatibilities of the ventilation mode between corresponding pairs of nodes from both plans. However, sometimes, according to expert knowledge, we simplified our assessments, *e.g.*, respiration unsupported and CPAP are estimated as similar for more medical details). More complicated situation is present if we want to judge the similarity in treatment of PDA. We have to assign the ventilation mode, as well as the similarity of PDA closing procedure. In summary, any aspect of the similarity between plans should be taken into account in the specific way and the domain knowledge

is necessary for joining all these similarities (obtained for all aspects). Therefore, the similarity between plans should be assigned on the basis of a special ontology specified in a dialog with human experts. Such ontology we call *similarity ontology*. Using such similarity ontology, we developed methods for inducing classifiers predicting the similarity between two plans (generated automatically and proposed by human experts).

In the chapter, we assume that each similarity ontology between plans has a tree structure. The root of this tree is always one concept representing general similarity between plans. In each similarity ontology there may exist concepts of two-way type. In this chapter, the concepts of the first type will be called *internal concepts* of ontology. They are characterized by the fact that they depend on other ontology concepts. The concept of the second type will be called *input concepts* of ontology (in other words the concepts of the lowest ontology level). The input concepts are characterized by the fact that they do not depend on other ontology concepts. Fig. 7.20 shows an exemplary ontology of similarity between plans of the treatment of newborn infants with the respiratory failure. This ontology has been provided by human experts. However, it is also possible to present some other versions of such ontology, instead of that presented above, according to opinions of some other group of human experts.

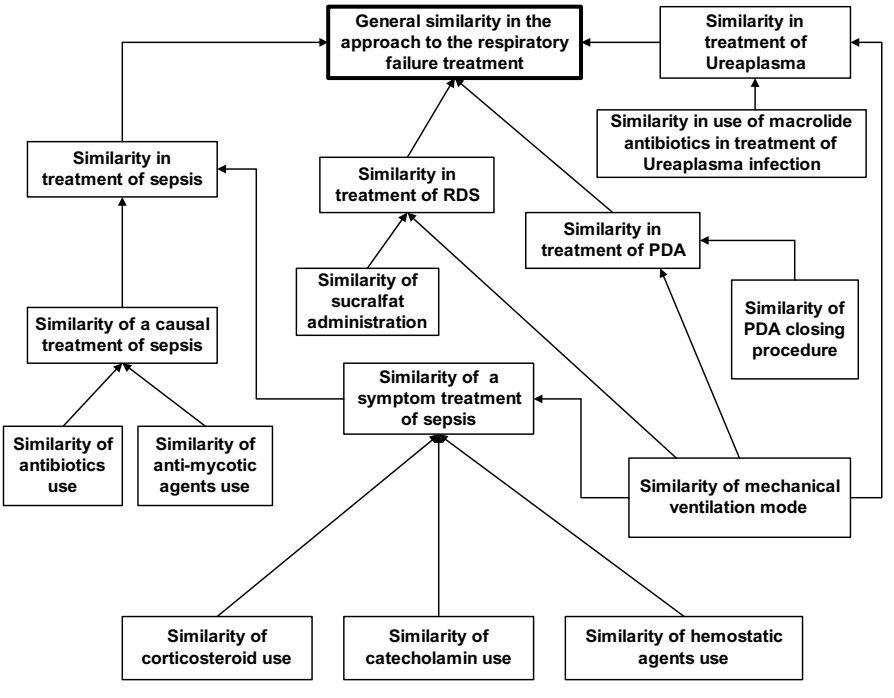


Fig. 7.20 An exemplary ontology of similarity between plans of the treatment of newborn infants with respiratory failure

7.5.4 Similarity Classifier

Using the similarity ontology (*e.g.*, the ontology presented in Fig. 7.20), we developed methods for inducing classifiers predicting the similarity between two plans (generated automatically and proposed by human experts).

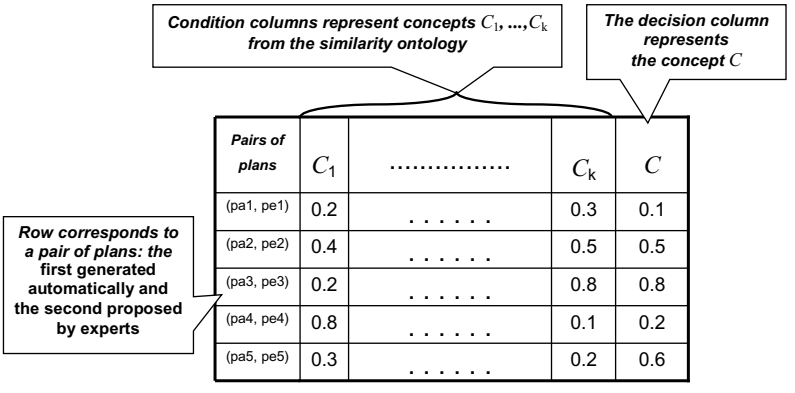


Fig. 7.21 The scheme of the similarity table of plans

The method for construction of such classifier can be based on *a similarity table of plans*. The similarity table of plans is the decision table which may be constructed for any concept from the similarity ontology. The similarity table is created in order to approximate a concept for which the table has been constructed. The approximation of the concept takes place with the help of classifiers generated for the similarity table. However, because of the fact that in the similarity ontology there occur two types of concepts (internal and input), there are also two types of similarity tables. Similarity tables of the first type are constructed for internal concepts, whereas the tables of the second type are constructed for input concepts.

Similarity tables for internal concepts of similarity ontology are constructed for a certain fragment of similarity ontology which consists of a concept of this ontology and concepts on which this concept depends. In the case of ontology from Fig. 7.20 it may be for instance the concept *Similarity of a symptom treatment of sepsis* and concepts *Similarity of corticosteroid use*, *Similarity of catecholamin use*, and *Similarity of hemostatic agents use*. To simplify further discussion, let us assume that it is the concept C that depends in the similarity ontology on the concepts C_1, ..., C_k. The aim of constructing a similarity table is approximation of concept C using concepts C_1, ..., C_k (see Fig. 7.21). Condition columns of such similarity table represent concepts C_1, ..., C_k. Any row corresponds to a pair of plans: generated automatically and proposed by experts. Values of all attributes have been provided by experts from the set $\{0.0, 0.1, ..., 0.9, 1.0\}$. Finally, the decision column represents the concept C.

The stratifying classifier computed for a similarity table (called *a similarity classifier*) can be used to determine the similarity between plans (generated by our

methods of automated planning and plans proposed be human experts) relatively to a given internal concept C.

Such stratifying classifiers may be constructed for all concepts from the similarity ontology which depend, in this ontology, on other concepts. However, we also need stratifying classifiers for input concepts of ontology, that is, those lying on the lowest level of the ontology. Hence, they are the concepts which do not depend on other concepts in this ontology. To approximate them, we do not use other ontology concepts but we apply the features of comparable plans which are expressed in the form of patterns defined in the special language. Obviously, such types of patterns are also concepts determined in the set of pairs of plans. However, they are usually not placed in the similarity ontology between plans. Therefore, approximation tables of input concepts of the similarity ontology should be treated as a specific type of similarity table.

Let us notice that the similarity table defined above is constructed in the way that concept C is approximated on the basis of the features of both plans corresponding to a given object from the set U.

It is worth noticing that for approximation of complex concepts from the similarity ontology one can use also features (attributes) describing relations between plans. Such features are formulated in a natural language using special questions about both plans. Examples of such questions are: *Were antibiotics used simultaneously in both plans?*, *Was the average difference between mechanical ventilation mode in both plans significant?*. However, it requires a simple extension of the language.

Classifiers constructed for similarity tables corresponding to all concepts from the similarity ontology may be used to construct a complex classifier which gives the general similarity between plans (represented by the concept lying in the root of the similarity ontology). We provide an example of how such a classifier works. Let us assume that there is a certain similarity ontology between pairs of plans in which there occur six following concepts: C_1, C_2, C_3, C_4, C_5, and C_6. The concept C_1 depends on concepts C_2 and C_3, the concept C_2 depends on concepts C_4 and C_5, and the concept C_3 depends on concepts C_5 and C_6. In this ontology, concept C_1 is the concept of general similarity between plans, whereas concepts C_4, C_5, and C_6 are input concepts of the similarity ontology (see Fig. 7.22).

First, we construct similarity tables for concepts C_4, C_5, C_6 and stratifying classifiers μ_{C_4}, μ_{C_5}, μ_{C_6} corresponding to them. Let us also assume that there are given stratifying classifiers μ_{C_1}, μ_{C_2}, μ_{C_3} which were constructed for similarity tables which correspond to concepts C_1, C_2 and C_3. Tested object $u = (p_1, p_2)$ which is a pair of compared plans is classified to the layer of concept C corresponding to it in the following way. At the beginning, the object u is classified by classifiers μ_{C_4}, μ_{C_5}, and μ_{C_6}. This way we obtain values $\mu_{C_4}(u)$, $\mu_{C_5}(u)$, and $\mu_{C_6}(u)$. Next, values $\mu_{C_4}(u)$ and $\mu_{C_5}(u)$ are used as the values of conditional attributes in the similarity table constructed for concept C_2. Thus, the object u may be classified by classifier μ_{C_2}, which gives us value $\mu_{C_2}(u)$. At the same time, values $\mu_{C_5}(u)$ and $\mu_{C_6}(u)$ are used as the values of conditional attributes in the similarity table constructed for concept C_3. It gives the possibility to classify object u by classifier μ_{C_3} and obtain

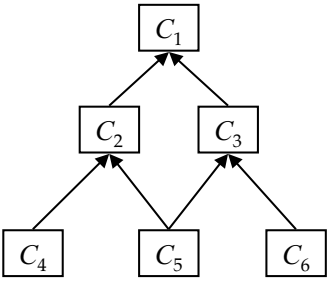


Fig. 7.22 The scheme of a simple similarity ontology

value $\mu_{C_3}(u)$. Finally, values $\mu_{C_2}(u)$ and $\mu_{C_3}(u)$ are used as the values of conditional attributes of the similarity table constructed for concept C_1. Thus, the object u may be classified by classifier μ_{C_1} to layer $\mu_{C_1}(u)$.

The complex classifier described above can be used to determine the general similarity between plans generated by our methods of automated planning and plans proposed by human experts, *e.g.*, during the real-life clinical treatment (see Section 7.5.5).

7.5.5 Experiments with Medical Data

To verify the effectiveness of the presented in this chapter methods of automated planning, we have implemented the algorithms in the RoughICE system (see [42]).

It should be emphasized that, in general, automated planning of treatment is a very difficult and complicated task because it requires extensive medical knowledge combined with sensor information about the state of a patient. Even so, the proposed approach makes it possible to obtain quite satisfactory results in the short-term planning of treatment of infants with respiratory failure. The reason is that medical data sets have been accurately prepared for purposes of our experiments using the medical knowledge. For example, the collection of medical actions, that are usually used during the treatment of infants with respiratory failure, has been divided into a few groups of similar actions (for example: antibiotics, anti-mycotic agents, mechanical ventilation, catecholamines, corticosteroids, hemostatic agents). It is very helpful in the prediction of actions because the number of actions is significantly decreased.

The experiments have been performed on the medical data sets obtained from Neonatal Intensive Care Unit, First Department of Pediatrics, Polish-American Institute of Pediatrics, Collegium Medicum, Jagiellonian University, Cracow, Poland (see [4,6,8,9]). We used one data table, that consists of 11,099 objects. Each object of this table describes parameters of one patient in single time point. There were prepared 7,022 situations on the basis of this data table, where the plan of treatment has been proposed by human experts during the real-life clinical treatment.

We have applied the *train-and-test* method. In each experiment the whole set of patients was randomly divided into two groups (training and tested one). Each of these groups allowed creating approximately 4000 time windows which have duration of 7 time points. Time windows created on the basis of patients from the training part created a training table for a given experiment (when plans of treatment have been assigned), whereas time windows created on the basis of patients from the tested part created a test table for the experiment (when plans have been generated by automated method and expert plans are known in order to compare both plans)

In the discussed experiments, the distance between time points recorded for a specific patient was constant (one day). In a single experiment concerning a patient's treatment, a 7-point sequence of time points was used. In terms of planning the treatment each such sequence may be written as s_1, a_1, s_2, a_2, s_3, a_3, s_4, a_4, s_5, a_5, s_5, a_6, s_7, where s_i (for $i = 1,...,7$) is a patient state and a_i (for $i = 1,...,6$) is a complex medical action performed in the state s_i. The first part of the above sequence of states and actions, that is, from state s_1 to state s_3, was used by the method of automated planning as the input information (corresponding to the values of conditional attributes in the classic approach to constructing classifiers). The remaining actions and states were automatically generated to create plan (s_3, a'_3, s'_4, a'_4, s'_5, a'_5, s'_6, a'_6, s'_7). This plan may be treated as a certain type of a complex decision value. Verification of the quality of the generated plan consisted in comparing plan (s_3, a'_3, s'_4, a'_4, s'_5, a'_5, s'_6, a'_6, s'_7) with plan (s_3, a_3, s_4, a_4, s_5, a_5, s_5, a_6, s_7). It is worth adding that a single complex action concerned one time point, meta action concerned two time points and a single experiment consisted in planning two meta actions. Hence, in a single experiment four actions were planned (patient's treatment for 4 days). In other words, at the beginning of the automated planning procedure the information about the patient's state in the last 3 days of his hospitalization was used (s_1, s_2, s_3) together with the information about complex medical actions undertaken one or 2 days before (a_1, a_2). The generated plan included information about a suggested complex medical action on a given day of hospitalization (a'_3), information about actions which should be undertaken in the 3 following days of hospitalization (a'_4, a'_5, a'_6) and information about the patient's state anticipated as a result of the planned treatment in the four following days of hospitalization (s'_4, s'_5, s'_6, s'_7).

As a measure of planning success (or failure) in our experiments, we use the special classifier that can predict the similarity between two plans as a number between 0.0 (very low similarity between two plans) and 1.0 (very high similarity between two plans) (see Section 7.5.4). We use this classifier to determine the similarity between plans generated by our methods of automated planning and plans proposed be human experts during the real-life clinical treatment. In order to determine the standard deviation of the obtained results, each experiment was repeated for 10 random divisions of the whole data set.

The average similarity between plans for all tested situations was **0.802**. The corresponding standard deviations was **0.041**. The coverage of tested situation by generated plans was **0.846** with standard deviation **0.018**.

Due to the fact that the average similarity is not too high (less than 0.9) and the standard deviation is relatively high for our algorithm, we present also the

Table 7.4 The average percent of plans belonging to the specified interval and the average similarity of plans in this interval

Intervals	Average percent of plans	Average similarity of plans
[0.0, 0.2]	**12.1%** $\pm$ 4.5%	**0.139** $\pm$ 0.002
(0.2, 0.4]	**6.2%** $\pm$ 1.5%	**0.349** $\pm$ 0.003
(0.4, 0.6]	**7.1%** $\pm$ 1.7%	**0.563** $\pm$ 0.002
(0.6, 0.8]	5.8% $\pm$ 0.9%	0.773 $\pm$ 0.004
(0.8, 1.0]	68.9% $\pm$ 5.6%	0.987 $\pm$ 0.002

distribution of the results. We describe results in such a way that we present how many generated plans belong to the specified interval of similarity. For this reason we divided interval [0.0, 1.0] into 5 equal intervals, i.e., [0.0, 0.2], [0.2, 0.4], [0.4, 0.6], [0.6, 0.8], and [0.8, 1.0]. Table 7.4 shows the average percent of the plans belonging to the specified interval and the average similarity of plans in this interval.

It is easy to see that some group of plans generated automatically is not enough similar to the plans proposed by the experts. If we assume that inadequate similarity is lower than 0.6, in this group we found about 25% of all plans (see Table 7.4). To explain this issue, we should observe more carefully plans, which are incompatible with the proposals prepared by experts. In practice, the main medical actions influencing the similarity of plans in accordance with ontology of the similarity from Fig. 7.20 are mechanical ventilation, antibiotics, anti mycotic agents, and macrolide antibiotics. Therefore, it may be interesting how the treatment similarity changed in the range of applying these actions in the individual intervals of similarity between the plans.

On Fig. 7.23 we can see that a significant incompatibility of treatment plans most often concerns mechanical ventilation and perhaps antibiotic therapy — the situation when a patient develops a sudden and severe infection (*e.g.*, sepsis). Such circumstances cause rapid exacerbation of respiratory failure are required higher level of mechanical ventilation and immediate antibiotic treatment. For example, although microbiological confirmation of current infection is achieved after 2-3 days, physician starts treatment after first symptoms of suspected disease and often intensify mechanical ventilation mode. It would seem that the algorithms of automated planning presented in this chapter may imitate the strategy of treatment described above. Unfortunately, in practice, these algorithms are not able to learn this strategy for a lot of information because they were not introduced to the base records or were introduced with delay. For instance, hemoglobin saturation which is measured for the whole time, as the dynamic marker of patients respiratory status, was not found in the data, whilst results of arterial blood gases were introduced irregularly, with many missing values. So, the technical limitation of the current data collection lead to the intensive work modifying and extending both, the equipment and software, served for gathering clinical data. It may be expected that in several years the

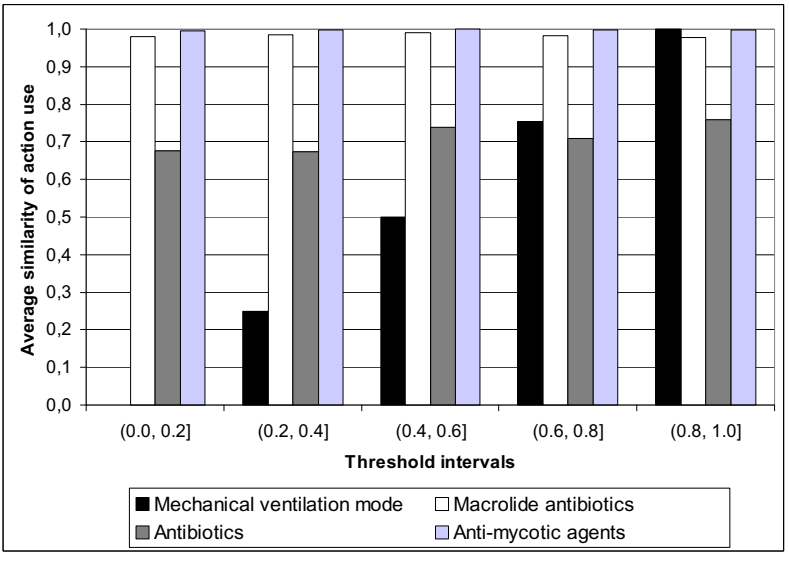


Fig. 7.23 The average similarity of plans in the specified interval for medical actions

automated planning algorithms, described in this chapter, will achieve much better and useful results.

A separate problem is a relatively low coverage of the algorithms described in this chapter which equals averagely 0.846. Such a low coverage results from the specificity of the automated planning method used which synchronizes the treatment of four diseases (RDS, PDA, sepsis, and Ureaplasma). We may identify two reasons of a low coverage. First, because of data shortage the algorithm in many situations may not synchronize the treatment of the above mentioned diseases. It happens this way because each proposed comparison of plans may be debatable in terms of the knowledge gathered in the system. Therefore, in these cases the system does not suggest any treatment plan and says *I do not know*. The second reason for low coverage is the fact that the automated planning method used requires application of a complex classifier which consists of many classifiers of lesser complexity. Combining these classifiers together often causes the effect of decreasing the complex classifier coverage. For instance, let us assume that making decision for tested object u requires application of complex classifier μ, which consists of two classifiers μ_1 and μ_2. We apply classifier μ_1 directly to u, whereas classifier μ_2 is applied to the results of classification of classifier μ_1. In other words, to make classifier μ_2 work for a given tested object u we need value $\mu_1(u)$. Let us assume that the coverage for classifiers μ_1 and μ_2 equals respectively 0.94 and 0.95. Hence, the coverage of classifier μ is equal $0.94 \cdot 0.95 = 0.893$, that is the coverage of classifier μ is smaller than the coverage of classifier μ_1 as well as the coverage of classifier μ_2.

In summation, we conclude that experimental results showed that the proposed automated planning method gives good results, also in the opinion of medical

experts (compatible enough with the plans suggested by the experts), and may be applied in medical practice as a supporting tool for planning the treatment of infants suffering from respiratory failure.

7.6 Conclusion

The aim of this chapter was to present new methods of approximating complex concepts on the basis of experimental data and domain knowledge which is mainly represented using concept ontology.

At the beginning of the chapter, a method of spatial complex concepts approximation was presented (see Section 7.2). Next, in Sections 7.3 we presented the method of approximate spatio-temporal complex concepts. In the further part of the chapter, the method of behavioral pattern identification was overviewed (see Section 7.4). Finaly, in Section 7.5, we described the method of automated planning of behavior of complex objects when the states of objects are represented by spatio-temporal concepts which require an approximation.

We have also described the results of computer experiments conducted on real-life data sets which were obtained from the road traffic simulator (see [44]) and on medical data sets which were made available by Neonatal Intensive Care Unit, First Department of Pediatrics, Polish-American Institute of Pediatrics, Collegium Medicum, Jagiellonian University, Cracow, Poland and by Second Department of Internal Medicine, Collegium Medicum, Jagiellonian University, Cracow, Poland.

In light of theoretical discourse and the results of computer experiments presented in the chapter the following conclusions may be drawn:

1. The method of approximation of complex spatial concepts, described in the chapter (see Section 7.2), with the help of approximate reasoning schemes (AR-schemes) leads to better results than the classical methods based on decision rules induced directly from sensor data, because the quality of classifier classification based on AR-schemes is higher than the quality of classification obtained by classifiers based on decision rules, particularly for small decision classes representing atypical cases in the recognition of which we are most interested in, *e.g.*, a dangerous driving vehicle on a highway. Moreover, for larger data sets, the time of constructing classifiers based on AR-schemes is much shorter than the time of inducing classifiers based on decision rules, and the structure of classifiers based on AR-schemes is less complex than the structure of classifiers based on decision rules. It is also worth mentioning that the classifiers based on AR-schemes are more robust (stable or tolerant) when it comes to changes in training data sets serving the construction of classifiers, that is, a classifier based on AR-schemes, constructed for one data set, often proves itself good for another data set. For example, a classifier constructed for data generated from the traffic simulator with one simulation scenario proves itself useful in classification of objects generated by the simulator with the use of another simulation scenario.

2. The methodology of modeling complex object behavior with the use of behavioral graphs of these objects, proposed in the chapter (see Section 7.4), is a convenient and effective tool for identifying behavioral or risk patterns of complex objects. On the one hand this methodology, enables to represent concepts on a high abstraction level, and on the other hand, owing to the use of a domain knowledge, it enables to approximate these concepts on the basis of sensor data and using a domain knowledge.
3. The methods of automated planning of complex object behavior proposed in the chapter facilitate an effective planning of behavior of objects whose states are defined in a natural language using vague spatio-temporal conditions (see Section 7.5). The authenticity of conditions of this type is usually not possible to be verified on the basis of a simple analysis of available information about the object and that is why these conditions must be treated as spatio-temporal complex concepts and their approximation requires methods described in this chapter which are based on data sets and domain knowledge.

In summation, it may be concluded that in executing real-life projects related to the construction of the intelligent systems supporting decision-making, apart from data sets it is necessary to apply domain knowledge. Without its application successful execution of many such projects becomes extremely difficult or impossible. On the other hand, appropriate space must be found for the automated methods of classifier construction wherever it is feasible. It means, thus, finding a certain type of “the golden mean" to apply appropriate proportions in domain knowledge usage and automated methods of data analysis. Certainly, it will determine the success or failure of many projects.

Acknowledgement. This work was supported by the grant N N516 077837 from the Ministry of Science and Higher Education of the Republic of Poland, the Polish National Science Centre (NCN) grant 2011/01/B/ST6/03867 and by the Polish National Centre for Research and Development (NCBiR) grant No. SP/I/1/77065/10 in frame of the the strategic scientific research and experimental development program: “Interdisciplinary System for Interactive Scientific and Scientific-Technical Information”.

References

1. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, Berlin (2011)
2. Altman, D.G.: Practical Statistics for Medical Research. Chapman and Hall/CRC, London (1997)
3. Bar-Yam, Y.: Dynamics of Complex Systems. Addison-Wesley, New York (1997)
4. Bazan, J.G.: Hierarchical Classifiers for Complex Spatio-temporal Concepts. In: Peters, J.F., Skowron, A., Rybiński, H. (eds.) Transactions on Rough Sets IX. LNCS, vol. 5390, pp. 474–750. Springer, Heidelberg (2008)
5. Bazan, J.G.: Rough sets and granular computing in behavioral pattern identification and planning. In: Pedrycz, W., Skowron, A., Kreinovich, V. (eds.) Handbook of Granular Computing, pp. 777–799. John Wiley & Sons, The Atrium (2008)

6. Bazan, J.G., Kruczek, P., Bazan-Socha, S., Skowron, A., Pietrzyk, J.: Automatic planning based on rough set tools: Towards supporting treatment of infants with respiratory failure. In: Proceedings of the Workshop on Concurrency, Specification, and Programming (CS&P 2006), Wandlitz, Germany, September 27-29. Informatik-Bericht, vol. 170, pp. 388–399. Humboldt University, Berlin (2006)
7. Bazan, J.G., Kruczek, P., Bazan-Socha, S., Skowron, A., Pietrzyk, J.: Automatic Planning of Treatment of Infants with Respiratory Failure Through Rough Set Modeling. In: Greco, S., Hata, Y., Hirano, S., Inuiguchi, M., Miyamoto, S., Nguyen, H.S., Słowiński, R. (eds.) RSCTC 2006. LNCS (LNAI), vol. 4259, pp. 418–427. Springer, Heidelberg (2006)
8. Bazan, J.G., Kruczek, P., Bazan-Socha, S., Skowron, A., Pietrzyk, J.: Risk pattern identification in the treatment of infants with respiratory failure through rough set modeling. In: Proceedings of the Eleventh Conference of Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU 2006), Paris, France, July 2-7, pp. 2650–2657 (2006)
9. Bazan, J.G., Kruczek, P., Bazan-Socha, S., Skowron, A., Pietrzyk, J.: Rough set approach to behavioral pattern identification. Fundamenta Informaticae 75(1-4), 27–47 (2007)
10. Bazan, J.G., Skowron, A.: Classifiers based on approximate reasoning schemes. In: Dunin-Kęplicz, B., Jankowski, A., Skowron, A., Szczuka, M. (eds.) Monitoring, Security, and Rescue Techniques in Multiagent Systems. Advances in Soft Computing, pp. 191–202. Springer, Heidelberg (2005)
11. Borrett, S.R., Bridewell, W., Langley, P., Arrigo, K.R.: A method for representing and developing process models. Ecological Complexity 4, 1–12 (2007)
12. Breiman, L.: Statistical modeling: the two cultures. Statistical Science 16(3), 199–231 (2001)
13. Desai, A.: Adaptive complex enterprises. Communications ACM 5(48), 32–35 (2005)
14. Doherty, P., Łukaszewicz, W., Skowron, A., Szałas, A.: Knowledge Engineering: A Rough Set Approach. Springer, Heidelberg (2006)
15. Domingos, P.: Toward knowledge-rich data mining. Data Mining and Knowledge Discovery 1(15), 21–28 (2007)
16. Friedman, J., Hastie, T., Tibshirani, R.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction, vol. I-V. Springer, Heidelberg (2001)
17. Guarino, N.: Formal ontology and information systems. In: Proceedings of the First International Conference on Formal Ontology in Information Systems (FOIS 1998), Trento, Italy, June 6-8, pp. 3–15. IOS Press (1998)
18. Hillerbrand, R., Sandin, P., Peterson, M., Roeser, S. (eds.): Handbook of Risk Theory: Epistemology, Decision Theory, Ethics, and Social Implications of Risk. Springer, Berlin (2012)
19. Hoen, P.J., Tuyls, K., Panait, L., Luke, S., Poutré, J.A.L.: An Overview of Cooperative and Competitive Multiagent Learning. In: Tuyls, K., 't Hoen, P.J., Verbeeck, K., Sen, S. (eds.) LAMAS 2005. LNCS (LNAI), vol. 3898, pp. 1–46. Springer, Heidelberg (2006)
20. Ignizio, J.P.: An Introduction to Expert Systems. McGraw-Hill, New York (1991)
21. Jarrar, M.: Towards Methodological Principles for Ontology Engineering. Ph.D. thesis, Vrije Universiteit Brussel (2005)
22. Keefe, R.: Theories of Vagueness. Cambridge University Press, New York (2000)
23. Kloesgen, E., Zytkow, J. (eds.): Handbook of Knowledge Discovery and Data Mining. Oxford University Press, Oxford (2002)
24. Kriegel, H.P., Borgwardt, K.M., Kröger, P., Pryakhin, A., Schubert, M., Zimek, A.: Future trends in data mining. Data Mining and Knowledge Discovery 1(15), 87–97 (2007)
25. Langley, P.: Cognitive architectures and general intelligent systems. AI Magazine 27, 33–44 (2006)
26. Liu, J., Jin, X., Tsui, K.: Autonomy Oriented Computing: From Problem Solving to Complex Systems Modeling. Kluwer/Springer, Heidelberg (2005)

27. Luck, M., McBurney, P., Shehory, O., Willmott, S.: Agent technology: Computing as interaction. a roadmap for agent-based computing. Agentlink iii, the european coordination action for agent-based computing. University of Southampton, UK (2005)
28. Michalski, R., et al. (eds.): Machine Learning, vol. I-IV. Morgan Kaufmann, Los Altos (1983, 1986, 1990, 1994)
29. Michie, D., Spiegelhalter, D.J., Taylor, C.C.: Machine learning, neural and statistical classification. Ellis Horwood Limited, England (1994)
30. Mitchel, T.M.: Machine Learning. McGraw-Hill, Boston (1997)
31. Pancerz, K., Suraj, Z.: Rough sets for discovering concurrent system models from data tables. In: Hassanien, A.E., Suraj, Z., Ślęzak, D., Lingras, P. (eds.) Rough Computing: Theories, Technologies and Applications. Idea Group, Inc. (2007)
32. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning about Data. In: D: System Theory, Knowledge Engineering and Problem Solving, vol. 9, Kluwer Academic Publishers, Dordrecht (1991)
33. Pawlak, Z., Skowron, A.: Rudiments of rough sets. Information Sciences 177, 3–27 (2007)
34. Peters, J.F.: Rough Ethology: Towards a Biologically-Inspired Study of Collective Behavior in Intelligent Systems with Approximation Spaces. In: Peters, J.F., Skowron, A. (eds.) Transactions on Rough Sets III. LNCS, vol. 3400, pp. 153–174. Springer, Heidelberg (2005)
35. Peters, J.F., Skowron, A.: Zdzisław Pawlak life and work (1926–2006). Information Sciences 177, 1–2 (2007)
36. Bazan, J.G., Szczuka, M.S.: The Rough Set Exploration System. In: Peters, J.F., Skowron, A. (eds.) Transactions on Rough Sets III. LNCS, vol. 3400, pp. 37–56. Springer, Heidelberg (2005)
37. Peters, J.F., Skowron, A., Rybiński, H. (eds.): Transactions on Rough Sets IX. LNCS, vol. 5390. Springer, Heidelberg (2008)
38. Poggio, T., Smale, S.: The mathematics of learning: Dealing with data. Notices of the American Mathematical Society (AMS) 5(50), 537–544 (2003)
39. Polkowski, L., Skowron, A.: Rough Mereology. In: Raś, Z.W., Zemankova, M. (eds.) ISMIS 1994. LNCS, vol. 869, pp. 85–94. Springer, Heidelberg (1994)
40. Read, S.: Thinking about Logic: An Introduction to the Philosophy of Logic. Oxford University Press, New York (1994)
41. Ripley, B.D.: Pattern Recognition and Neural Networks. Cambridge University Press, Cambridge (1996)
42. Rough ICE: Project web site, `http://logic.mimuw.edu.pl/~bazan/roughice/`
43. RSES: Project web site, `http://logic.mimuw.edu.pl/~rses`
44. Simulator: Project web site, `http://logic.mimuw.edu.pl/~bazan/simulator`
45. Skowron, A., Stepaniuk, J., Swiniarski, R.W.: Modeling rough granular computing based on approximation spaces. Information Sciences 184(1), 20–43 (2012)
46. Stone, P., Sridharan, M., Stronger, D., Kuhlmann, G., Kohl, N., Fidelman, P., Jong, N.K.: From pixels to multi-robot decision-making: A study in uncertainty. Robotics and Autonomous Systems 54(11), 933–943 (2006)
47. Suraj, Z.: Discovering concurrent data models and decision algorithms from data: A rough set approach. International Journal on Artificial Intelligence and Machine Learning, IRSI, 51–56 (2004)
48. The Infobright Community Edition (ICE) Homepage at, `http://www.infobright.org/`
49. Unnikrishnan, K.P., Ramakrishnan, N., Sastry, P.S., Uthurusamy, R.: Service-oriented science: Scaling escience impact. In: Proceedings of the Fourth KDD Workshop on Temporal Data Mining: Network Reconstruction from Dynamic Data, The Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data (KDD 2006), Philadelphia, USA, August 20-23 (2006)

50. Urmson, C., et al.: High speed navigation of unrehearsed terrain: Red team technology for grand challenge. Report CMU-RI-TR-04-37, The Robotics Institute, Carnegie Mellon University (2004)
51. Vapnik, V. (ed.): Statistical Learning Theory. Wiley, New York (1998)
52. Zadeh, L.A.: From computing with numbers to computing with words – from manipulation of measurements to manipulation of perceptions. IEEE Transactions on Circuits and Systems – I: Fundamental Theory and Applications 1(45), 105–119 (1999)
53. Zadeh, L.A.: Toward a generalized theory of uncertainty (GTU) - an outline. Information Sciences 171, 1–40 (2005)

Chapter 8
Incorporating Rough Data in Database Design for Imprecise Information Representation

Theresa Beaubouef and Frederick E. Petry

Abstract. This chapter describes issues of database design for databases that provide representations that allow rough set data. We first present the definition of a rough relational database. A database can be designed by Entity-Relationship modeling or by defining the functional dependencies for the system. We describe rough set E-R models and rough functional dependencies. Based on the rough functional dependencies we provide the various related rough normal forms used in a database schema design. Finally, we discuss issues of database security and the use of rough spatial data in a rough relational database.

Keywords: Rough relational database, rough functional dependency, rough spatial data, imprecise information, database security, rough normal form, rough entropy.

8.1 Introduction

As databases continue to grow in size and complexity they are used in many diverse applications. For many real world applications, it is necessary to incorporate some type of uncertainty management into the underlying data model. One characteristic of many imprecise databases is that they allow sets of values in their tuples. This is referred to as a non-first form or nested database [30,31]. If the value of an

Theresa Beaubouef
Department of Computer Science and Industrial Technology
Southeastern Louisiana University
Hammond, LA 70402 USA

Frederick E. Petry
2Naval Research Lab
Geospatial Science and Technology Branch
Stennis Space Center, MS 39529 USA

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 137–155.
springerlink.com

attribute is non-atomic, i.e. set-valued, then there is uncertainty as to which one of the values in the set corresponds to the attribute. There are specific aspects in different uncertain database models such as those using fuzzy sets [7,14,22], but all share use of set values. An early consideration of incomplete information related to databases was the approach of Lipski [16]. Of particular interest here is database modeling utilizing rough set approaches [6] to represent uncertainty.

Rough set theory [19,20] is a mathematical formalism for representing uncertainty. An approximation region in rough sets partitions some universe into equivalence classes. This partitioning can be adjusted to increase or decrease its granularity, to group items together that are considered indiscernible for a given purpose, or to "bin" ordered domains into range groups.

- U is the *universe*, which cannot be empty,
- R : *indiscernibility* relation, or equivalence relation,
- $A = (U,R)$, an ordered pair, called an *approximation space*,
- $[x]_R$ denotes the equivalence class of R containing x, for any element x of U,
- *elementary sets* in A — an equivalence classes of R,
- *definable set* in A — any finite union of elementary sets in A.

A *rough set* $X \subseteq U$, is defined in terms of the definable sets by specifying its lower ($\underline{R}X$) and upper ($\overline{R}X$) approximation regions:

$$\underline{R}X = \{x \in U | [x]_R \subseteq X\} \text{ and } \overline{R}X = \{x \in U | [x]_R \cap X \neq \emptyset\}.$$

$\underline{R}X$ is the positive region, $U - \overline{R}X$ is the negative region, and $\overline{R}X - \underline{R}X$ is the boundary or borderline region of the rough set X, allowing for the distinction between certain and possible inclusion in a rough set.

For example: Let $U = \{medium, small, little, tiny, big, large, huge, enormous\}$, and let the equivalence relation R be defined as follows:

$$R^* = \{[medium],\ [small,\ little,\ tiny],\ [big,\ large],\ [huge, enormous]\}.$$

A given set $X = \{medium, small, little, tiny, big, huge\}$, can be defined in terms of its lower and upper approximations:

$$\underline{R}X = \{medium, small, little, tiny\},$$

and

$$\overline{R}X = \{medium, small, little, tiny, big, large, huge, enormous\}.$$

The major rough set concepts of interest are the use of an indiscernibility relation to partition domains into equivalence classes and the concept of lower and upper approximation regions to allow the distinction between certain and possible, or partial, inclusion in a rough set. The indiscernibility relation allows the grouping of items based on some definition of 'equivalence' as it relates to the application domain. Those equivalence classes included in their entirety in X belong to the lower

approximation region. The upper approximation region includes those equivalence classes that are included either entirely or partially in X. The results in the lower approximation region are certain, corresponding to exact matches. The boundary region of the upper approximation contains results that are possible, but not certain.

In Pawlak [21] there was consideration of multi-valued and rough information systems. This shows that the concept of operating with data similar to relational data framework has a long-standing background in information systems as an equivalent form of representing tabular data.

8.2 Rough Relational Databases

The rough relational database model [6] is an extension of the standard relational database model of Codd [10]. It captures all the essential features of rough sets theory including indiscernibility of elements denoted by equivalence classes and lower and upper approximation regions for defining sets which are indefinable in terms of the indiscernibility.

Every attribute domain is partitioned by some equivalence relation designated by the database designer or user. Within each domain, those values that are considered indiscernible belong to an equivalence class. This is compatible with the traditional relational model where every value belongs to its own class. The indiscernibility information is used by the query mechanism to retrieve information based on equivalence with the class to which the value belongs rather than equality, resulting in less critical wording of queries as shown in [6].

Recall is also improved in the rough relational database because rough relations provide *possible* matches to the query in addition to the *certain* matches that are obtained in the standard relational database. This is accomplished by using set containment in addition to equality of attributes in the calculation of lower and upper approximation regions of the query result.

The rough relational database has several features in common with the ordinary relational database. Both models represent data as a collection of relations containing tuples. These relations are sets. The *tuples* of a relation are its elements, and like elements of sets in general, are unordered and non-duplicated. A tuple $\mathbf{t}_i$ takes the form $(d_{i1}, d_{i2}, \ldots, d_{im})$, where d_{ij} is a *domain value* of a particular domain set D_j. In the ordinary relational database, $d_{ij} \in D_j$. In the rough database, however, as in other non-first normal form (NF2) extensions to the relational model [7, 23, 25], $d_{ij} \subseteq D_j$, and although it is not required that d_{ij} be a singleton, $d_{ij} \neq \emptyset$. Let $P(D_i)$ denote the powerset of D_i without the empty set $\emptyset$.

Definition 8.1. A *rough relation* R is a subset of the set cross product

$$P(D_1) \times P(D_2) \times \ldots \times P(D_m).$$

A rough tuple $\mathbf{t}$ is any member of R, which implies that it is also a member of $P(D_1) \times P(D_2) \times \ldots \times P(D_m)$. If $\mathbf{t}_i$ is some arbitrary tuple, then $\mathbf{t}_i = (d_{i1}, d_{i2}, \ldots, d_{im})$

where $d_{ij} \subseteq D_j$. A tuple in this model differs from ordinary databases in that the tuple components may be sets of domain values rather than single values. The set braces are omitted here from singletons for notational simplicity.

Definition 8.2. An *interpretation* $\alpha = (a_1, a_2, \ldots, a_m)$ of a rough tuple

$$\mathbf{t}_i = (d_{i1}, d_{i2}, \ldots, d_{im})$$

is any value assignment such that $a_j \in d_{ij}$ for all j.

The interpretation space is the cross product $D_1 \times D_2 \times \ldots \times D_m$, but is limited for a given relation R to the set of those tuples which are valid according to the underlying semantics of R. In an ordinary relational database, because domain values are atomic, there is only one possible interpretation for each tuple $\mathbf{t}_i$, the tuple itself. In the rough relational database, this is not always the case when there is a set of values.

Let $[d_{xy}]$ denotes the equivalence class to which d_{xy} belongs. When d_{xy} is a set of values, the equivalence class is formed by taking the union of equivalence classes of members of the set; if $d_{xy} = \{c_1, c_2, \ldots, c_n\}$, then $[d_{xy}] = [c_1] \cup [c_2] \cup \ldots \cup [c_n]$.

Definition 8.3. Tuples $\mathbf{t}_i = (d_{i1}, d_{i2}, \ldots, d_{im})$ and $\mathbf{t}_k = (d_{k1}, d_{k2}, \ldots, d_{km})$ are redundant if $[d_{ij}] = [d_{kj}]$ for all $j = 1, \ldots, m$.

In the rough relational database, redundant tuples are removed in the merging process since duplicates are not allowed in sets, the structure upon which the relational model is based.

Since the rough relational database is in non-first normal form there are some attribute values that are sets. Another definition, which will be used for upper approximation tuples, is needed to capture redundancy between elements of attribute values that are sets.

Definition 8.4. Two sub-tuples $X = (d_{x1}, d_{x2}, \ldots, d_{xm})$ and $Y = (d_{y1}, d_{y2}, \ldots, d_{ym})$ are *roughly-redundant*, $\approx_R$, if for some $[p] \subseteq [d_{xj}]$ and $[q] \subseteq [d_{yj}], [p] = [q]$ for all $j = 1, \ldots, m$.

There are two basic types of relational operators. The first type arises from the fact that relations are considered sets of tuples. Therefore, operations which can be applied to sets also apply to relations. The most useful of these for database purposes are set *difference, union*, and *intersection*. Operators which do not come from set theory, but which are useful for retrieval of relational data are select, project, and join. In the rough relational database, relations are rough sets as opposed to ordinary sets. Therefore, new rough operators $(-, \cup, \cap, \sigma, \pi, \bowtie)$, comparable to the standard relational operators, were developed for the rough relational database. Properties of the rough relational operators can be found in [6].

An example of a question-answering system designed over non-deterministic data can be found in [25], and a related paper [24] deals with numeric columns

by introducing conditions based on resolution over their domains which is similar to this chapter's approach of partitioning columns' domains in order to define query conditions. An extension for relational data is found in [26] which adds an additional rough data layer (sometimes called database knowledge grid) that contains non-crisp values (corresponding to statistics of groups of original values stored underneath) and introduces a number of data operations (projection, join, aggregation) working in particular at rough level.

8.3 E-R Modeling for Rough Databases

We must first design our database using some type of semantic model. We use a variation of the entity-relationship [8] diagram that we call a fuzzy-rough E-R diagram. This diagram is similar to the standard E-R diagram in that entity types are depicted in rectangles, relationships with diamonds, and attributes with ovals. However, in the fuzzy-rough model, it is understood that membership values exist for all instances of entity types and relationships. Attributes which allow values where we want to be able to define equivalences are denoted with an asterisk $(^*)$ above the oval. These values are defined in the indiscernibility relation, which is not actually part of the database design, but inherent in the fuzzy-rough model.

Our fuzzy-rough E-R model [2] is similar to the second and third levels of fuzziness defined by Zvieli and Chen [32]. However, in our model, all entity and relationship occurrences (second level) are of the fuzzy type so we do not mark an 'f' beside each one. Zvieli and Chen's third level considers attributes that may be fuzzy. They use triangles instead of ovals to represent these attributes. We do not introduce fuzziness at the attribute level of our model in this chapter, only roughness, or indiscernibility, and denote those attributes with the '*'. In this present work, we only consider aspects of the design related to rough databases. However, this model is general enough to also apply to fuzzy rough databases as introduced in [6].

Consider, for example, a database to record public concerns about environmental quality in industrial areas where various chemical or nuclear plants are abundant. Data is collected through surveys or entered as a "best guess" by interviewers or recorders at public meetings. Demographic data is not always exact or may include a mixture of possible domain values. The situation is even less certain when probable values must be ascertained by observers having little knowledge of the individuals. A fuzzy-rough E-R diagram for our example appears in Fig. 8.1.

Once a rough entity-relationship semantic design has been established, the logical model for the rough relational database can be developed. In order to formally determine the "goodness" of this design, it can be evaluated in terms of increasingly restrictive normal forms. Normal forms are based on functional dependencies, both of which are discussed in the following section.

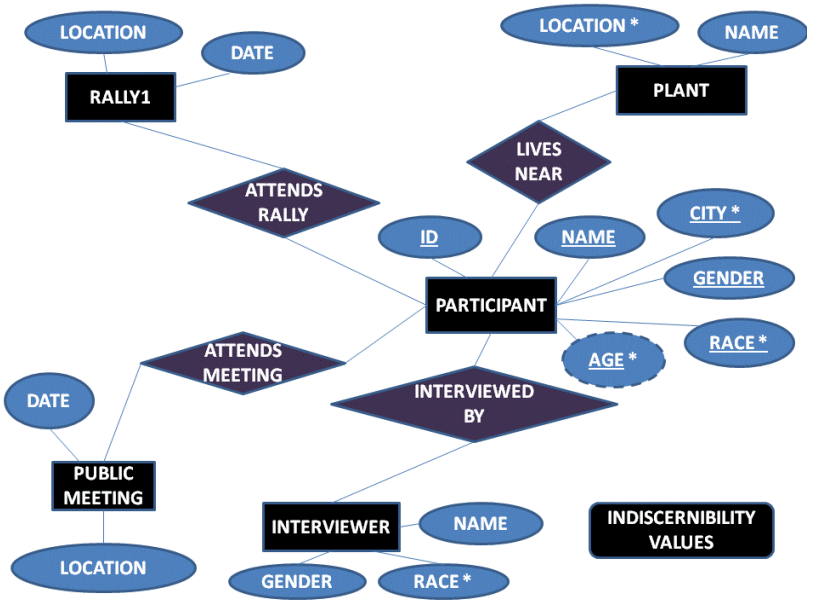


Fig. 8.1 E-R Diagram

8.4 Rough Functional Dependencies and Normalization

A functional dependency can be defined as in [12] through the use of a universal database relation concept. Let $R = \{A_1, A_2, \ldots, A_n\}$ be a universal relation schema describing a database having n attributes. Let X and Y be subsets of R. A functional dependency between the attributes of X and Y is denoted by $X \rightarrow Y$. This dependency specifies the constraint that for any two tuples of an instance r of R, if they agree on the X attribute(s) they must agree on their Y attributes(s): if $t_1[X] = t_2[X]$, then it must be true that $t_1[Y] = t_2[Y]$. Tuples that violate the constraint cannot be inserted into the database.

Functional dependencies are data dependencies that are functional in the same sense as functions in mathematics. Therefore, if the functional dependency $X \rightarrow Y$ holds, then the values of X functionally determine the values of Y; equivalently, Y is functionally dependent on X. The functional dependencies are used to specify constraints on tuple values based on the semantics of the relation attributes. A functional dependency must hold for all instances of the database on which it is defined. With these constraints incorporated into the design of the database schema, it is possible to restrict the tuples that comprise relations. These constraints aid in the maintenance of data integrity and the prevention of update anomalies.

The database designer may specify functional dependencies on relation schemas. However, there are usually many additional functional dependencies that will also hold. These dependencies can be inferred from those specified through the use of inference axioms, some of which are shown below; the first three of these are known as Armstrong's axioms:

(1) Reflexive:	$X \supseteq Y$	$\Rightarrow$	$X \rightarrow Y$
(2) Augmentation:	$X \rightarrow Y$	$\Rightarrow$	$XZ \rightarrow YZ$
(3) Transitive:	$X \rightarrow Y, Y \rightarrow Z$	$\Rightarrow$	$X \rightarrow Z$
(4) Projective:	$X \rightarrow YZ$	$\Rightarrow$	$X \rightarrow Y$
(5) Union:	$X \rightarrow Y, X \rightarrow Z$	$\Rightarrow$	$X \rightarrow YZ$
(6) Pseudotransitive:	$X \rightarrow Y, WY \rightarrow Z$	$\Rightarrow$	$WX \rightarrow Z$

Because Armstrong's axioms are sound and complete, all the functional dependencies of a database can be inferred by application of these rules to the set of given functional dependencies.

The rough functional dependency is based on the rough relational database model. The classical notion of functional dependency for relational databases does not naturally apply to the rough relational database, since all the "roughness" would be lost. In the rough querying of crisp data [1], however, the data is stored in the standard relational model having ordinary functional dependencies imposed upon it and rough relations result only from querying; they are not a part of the database design in which the designer imposes constraints upon relation schemas. Rough functional dependencies for the rough relational database model are defined as follows [3]:

Definition 8.5. A *rough functional dependency*, $X \rightarrow Y$, for a relation schema R exists if for all instances $T(R)$,

(1) for any two tuples $t, t' \in \underline{R}T$,
$$redundant(t(X), t'(X)) \Rightarrow redundant(t(Y), t'(Y))$$
(2) for any two tuples $s, s' \in T$,
$$roughly-redundant(s(X), s'(X)) \Rightarrow roughly-redundant(s(Y), s'(Y)).$$

Y is roughly functionally dependent on X, or X roughly functionally determines Y, whenever the above definition holds. This implies that constraints can be imposed on a rough relational database schema in a rough manner that will aid in integrity maintenance and the reduction of update anomalies without limiting the expressiveness of the inherent rough set concepts.

It is obvious that the classical functional dependency for the standard relational database is a special case of the rough functional dependency. Indiscernibility reduces to simple equality and part (2) of the definition is unused since all tuples in relations of a standard relational model belong to the lower approximation region of a similar rough model.

The first part of the definition of rough functional dependency compares with that of fuzzy functional dependencies discussed in [28], where adherence to Armstrong's axioms was proven. The results apply directly in the case of rough functional dependencies when only the lower approximation regions are considered. It is additionally necessary to show that axioms hold for upper approximations as in [3].

Theorem 8.1. *Rough functional dependencies satisfy Armstrong's axioms.*

Proof.
(1) **Reflexive**
If $Y \subseteq X \subseteq U$, then
$redundant(t(X),t'(X)) \Rightarrow redundant(t(Y),t'(Y))$, and
$roughly-redundant(t(X),t'(X)) \Rightarrow roughly-redundant(t(Y),t'(Y))$.
Hence, $X \rightarrow Y$.

(2) **Augmentation**
If $Z \subseteq U$ and the rough functional dependency $X \rightarrow Y$ holds, then
$redundant(t(XZ),t'(XZ)) \Rightarrow redundant(t(YZ),t'(YZ))$, and
$roughly-redundant(t(XZ),t'(XZ)) \Rightarrow roughly-redundant(t(YZ),t'(YZ))$.
Hence, $XZ \rightarrow YZ$.

(3) **Transitive**
If the rough functional dependencies $X \rightarrow Y$ and $Y \rightarrow Z$ hold, then
$redundant(t(X),t'(X)) \Rightarrow redundant(t(Z),t'(Z))$, and
$roughly-redundant(t(X),t'(X)) \Rightarrow roughly-redundant(t(Z),t'(Z))$.
Hence, $X \Rightarrow Z$. □

Hence, rough functional dependencies satisfy Armstrong's axioms. Given a set of rough functional dependencies, the complete set of rough functional dependencies can be derived using Armstrong's axioms. The rough functional dependency, therefore, is an important formalism for design in the rough relational database [4], and is inherent in the process of normalization.

8.5 Rough Normal Forms

Normalization [10, 12] of relational databases is a process of evaluating the functional dependencies in a relation, and determining whether the dependencies meet certain conditions, which will minimize redundancy in the database and reduce the insertion, deletion, and update anomalies that could occur. These normal forms are based on the traditional definitions of key, superkey, candidate key, and prime attribute, as can be found in [12]. In general, a key is an attribute on which all other attributes are functionally dependent, and a prime attribute is one, that is part of a key.

During the normalization process, if a relation schema does not meet the conditions for a particular normal form, then steps are taken to decompose relations in order to meet the specified criteria. Although normal forms range from first normal form (1NF), a basic structure of the standard relational model, through fifth normal form (5NF), typically 3NF or Boyce-Codd normal form, a stricter version of 3NF, is used. Each normal form is more restrictive than the previous one. For example, a relation in 3NF is also in 2NF, but the opposite is not necessarily true. In non-first

normal form extensions [17, 23] to the relational model, such as the rough relational model discussed here, we need not concern ourselves with the 1NF restriction.

8.5.1 Rough Second Normal Form

A rough relation schema is in rough 2NF if every non prime attribute is fully roughly functionally dependent on the key. In such a rough relation, there will be no partial dependencies [4].

Definition 8.6. Let F be the set of rough functional dependencies for schema R, and let K be a key of R. Then R is in *rough 2NF* only if none of the nonprime attributes is partially roughly dependent on K.

Consider, for example, a rough relation schema $R(A,B,C,D,E)$ having rough functional dependencies $B \to A, BC \to D$ and $BC \to E$. Here BC is the key, D and E are fully roughly functionally dependent on BC, and A is partially roughly-functionally dependent on BC.

In order to normalize R so that our database schema is in rough 2NF, we must do the following:

1. For each partial key form, a new rough relation containing the partial key and all of the attributes that are fully roughly-functionally dependent on it.
2. Remove those nonprime attributes from the original rough relation that are in this new rough relation.

Performing this procedure on the relation schema R above yields the following database schema: $R(B,C,D,E)$, $S(B,A)$. This is now in rough 2NF since every attribute of R is fully roughly-functionally dependent on the key AB and every attribute of S is fully roughly-functionally dependent on the key B.

8.5.2 Rough Third Normal Form

A rough relation schema is in rough 3NF if every nonprime attribute is fully roughly functionally dependent on the key and there exist no transitive dependencies. In such a rough relation schema, there will be no dependencies on attributes other than the key.

Definition 8.7. Let F be the set of rough functional dependencies for schema R, and let K be a key of R. Then R is in *rough 3NF* if whenever some nontrivial dependency $G \to H$ holds in R, then either (a) G is a superkey or (b) H is a prime attribute.

Consider, for example, a rough relation schema $R(B,C,G,H)$ having rough functional dependencies $B \to C, B \to G, B \to H$, and also $G \to H$. Here B is the key, but

notice that H is dependent on G, and G is dependent on B. This is a transitive dependency that prevents rough schema R from being in rough 3NF. G is not a superkey, and H is not a prime attribute.

In order to normalize our schema so that it will meet the requirements for 3NF, perform the following:

1. For each transitive dependency form a new rough relation containing the nonprime attribute that functionally determines the others in the dependency (this becomes the key) and all of the attributes that are roughly- functionally dependent on it.
2. Remove those attributes from the original rough relation that are nonprime attributes in this new rough relation.

In order to normalize $R(B,C,G,H)$ in the example above so that it is in rough 3NF, a new rough relation schema is created: $R(B,C,G), S(G,H)$. Notice that no transitive dependencies exist.

It is important that decomposition into rough third normal form also results in additional desirable properties, rough lossless join, and rough dependency preservation. The rough lossless join property insures that the original relations can be recovered from their decompositions and that spurious tuples are not generated when the decomposed relations are joined. Such spurious tuples represent erroneous information that is not part of the database.

A rough dependency preserving decomposition insures that all rough functional dependencies that exist before the decomposition remain after the decomposition. If this property does not hold, such that rough dependencies cannot be represented by individual rough relations, inefficient and unnecessary join operations would be required in order for constraints based on the dependency to be checked.

We can insure lossless join decomposition that preserves dependencies if the following steps are taken:

1. For the set of rough functional dependencies F for rough relation schema R, find a minimal cover G, with no partial functional dependencies.
2. Eliminate from R any attributes that are not included in G and place them in a separate rough relation schema.
3. For each X, left hand side, of dependencies in G, create new rough relation schema in the decomposition with attributes $\{X \cup \{A_1\} \cup \ldots \cup \{A_k\}\}$, where $X \rightarrow A_1, \ldots X \rightarrow A_k$ are the rough functional dependencies in G with X as the left hand side.

If no relation schema in the decomposition contains a key of R, then create an additional rough relation schema that contains attributes that form a key of R. Each relation schema in the decomposition will be in rough 3NF, and it can be shown that the lossless join and dependency preservation properties hold.

8.5.3 Rough Boyce Codd Normal Form (BCNF)

In terms of reducing redundancy, rough BCNF is the most desirable to achieve. It is stricter than rough 3NF since it eliminates condition (b) from the definition. Some decompositions result in "losing" the functional dependency and we must be careful not to decompose the schema in such a way as to not generate spurious tuples from a join operation.

Definition 8.8. Let F be the set of rough functional dependencies for schema R, and let K be a key of R. Then R is in *rough BCNF* if R is in rough 3NF and for any nontrivial dependency $G \to H$ in F, G is a superkey.

Although more restrictive normal forms have been defined for relational databases, a database design in BCNF is often considered "good" with respect to functional dependencies. For the rough relational database, rough 3NF or rough BCNF is usually sufficient. More uncommonly used normal forms such as 4NF are used in special circumstances such as for multivalued dependencies [12]. These do not commonly occur in ordinary database design and so are not considered here.

8.6 Security Design Issues

Security, or the protection of the database against unauthorized use, has several aspects [11, 13, 15, 27], but our concern here is with protection against unauthorized viewing of data. In particular for imprecise databases, this means access to some information if the exact correlations of data items remain unknown which is similar to the idea of security in statistical databases [9, 18]. For example, statistical information such as the average salary of a large group of individuals may be available, but not the exact salary of any one individual.

8.6.1 Security Approaches

The primary approaches to compromising these databases are to isolate a specific data value by intersecting a set of queries often based on the isolation of a single data value at the intersection of several query sets. Some ways to combat this include:

1. Minimum query size controls: These restrict very large or small query sets by a formula giving a lower bound on allowable query set size.
2. Minimum overlap control: This prevents replies to queries based on the number of records overlapping prior queries.

A basic consideration for uncertainty data representations in a rough database is the form of of non first normal form NF2 relations in which an attribute may have a set of values. Since, we are concerned with security violations we must discuss the possible meaning or interpretation of such relations. For example, consider a simple tuple with two attributes A_1,A_2 having the values: $A_1 = \{a\}$; $A_2 = \{b_1, b_2, \ldots b_m\}$. Then there are m possible meanings or interpretations for this tuple:

$$\{[a, b_1], [a, b_2], \ldots [a, b_m]\}$$

Assume we are concerned with not allowing some exact values of attribute A_2 corresponding to A_1 to be known. In this case there are m possible correspondences and a query that returned this one tuple, $(a, \{b_1, b_2, \ldots b_m\})$, would not directly violate our security concern. If the value of attribute A_1 were also a set of values, the possible relationship between the attribute values of A_1 and A_2 would be even more uncertain.

However, it may be possible that multiple querying of such set valued tuples could still lead to security violations. We will now carefully consider under what conditions in NF2 databases, security could still be violated. To do this we must consider the idea of tuple interpretations more formally.

For a given tuple $t(A_1, A_2, \ldots A_n)$ let the value of attribute A_i be the set $d_i \subseteq D_i$. Then each interpretation of the tuple t has a specific value v_i for each attribute A_i:

$$I = [v_1, v_2, \ldots v_n], v_i \in d_i.$$

In general for every interpretation, I_j,

$$I_j \in d_1 \times d_2 \times \ldots d_n.$$

To count the number of interpretations P_h of the tuple t_h, let the cardinality of the value of the ith attribute be $|d_{hi}| = p_i$. So then

$$P_h = \prod_{i=1}^{n} p_i.$$

Our discussion of interpretations allows us to address the question of whether a sequence of queries can isolate a single interpretation, thus violating security. Consider the general case in which we want to prevent an exact association between values of two attributes, A_j and A_k, when a set of r tuples $\{t_1, t_2, \ldots t_r\}$ is retrieved by querying. Then we have for each tuple the set of interpretations for the two attributes:

$$\begin{aligned}
&I_1(j,k) = \{I_{11}(j,k), I_{12}(j,k), \ldots I_{1n_1}(j,k)\} \\
&I_2(j,k) = \{I_{21}(j,k), I_{22}(j,k), \ldots I_{2n_2}(j,k)\} \\
&\ldots \\
&I_r(j,k) = \{I_{r1}(j,k), I_{r2}(j,k), \ldots I_{rn_r}(j,k)\},
\end{aligned}$$

where $I_h(j,k)$ is a specific interpretation of tuple t_h for these attributes.

Let us consider an example of how interpretations can be related to violations of security in retrieved set-valued tuples with the two attributes A_j and A_k. Assume the following three tuples are retrieved:

$$t_1 = (\{a,b,c\},\{r,s,t\})$$
$$t_2 = (\{a,d,e\},\{r,v,x\})$$
$$t_3 = (\{e,f\},\{s,t,z\}).$$

So these tuples have the interpretations:

$$I_1(j,k) = \{[a,r],[a,s],[a,t],[b,r],\ldots[c,s],[c,t]\}$$
$$I_2(j,k) = \{[a,r],[a,v],[a,x],[d,r],\ldots[e,v],[e,x]\}$$
$$I_3(j,k) = \{[e,s],[e,t],[e,z],[f,s],\ldots[f,z]\}.$$

Now these interpretations can be pairwise intersected:

$$I_1(j,k) \cap I_2(j,k) = \{[a,r]\},$$

and the other intersections are null:

$$I_1(j,k) \cap I_3(j,k) = I_2(j,k) \cap I_3(j,k) = \emptyset.$$

Since the intersection of tuples t_1 and t_2 produces a set of cardinality 1, we can definitely say the value a for attribute A_j is uniquely associated with the value r of attribute A_k which represents a security violation with respect to these attributes. So in general there will be a security violation if for any p,q

$$|I_p(j,k) \cap I_q(j,k)| = 1, \quad p \neq q.$$

These sort of security violations are similar to the minimum overlap control (2) mentioned previously. Also if a query can return a large query set size (1), then it is more likely some tuple interpretations might overlap resulting in a security violation as described.

As discussed previously in the rough relational database, the nature of rough set data causes a different view of tuple redundancy. So in the rough relational database, redundant tuples are removed in the merging process based on this definition. As a result of the underlying indiscernibility relation, the intersection of tuples in a single relation cannot produce a security violation. This follows because redundant tuples are not allowed in a rough relation, and so there cannot be two tuples having the same interpretation.

8.6.2 Rough Databases and Security Measures

This section will describe for the rough relational database information-theoretic measures for uncertainty defined for rough schemas and rough relations [5, 29]: In order to introduce these a rough entropy form for rough sets must first be given using the roughness measure for a rough set as given by Pawlak [20].

Definition 8.9. The rough entropy of a *rough* set X is H_R :

$$H_R(X) = -(\rho_R(X))[\Sigma\ Q_i log(P_i)]$$

for $i = 1,\ldots,n$ equivalence classes.

The term $\rho_R(X)$ denotes the roughness of the set X. The second term is the summation of the probabilities for each equivalence class belonging either wholly or in part to the rough set X. There is no ordering associated with individual class members. Therefore the probability of any one value of the class being named is the reciprocal of the number of elements in the class. If c_i is the cardinality of, or number of elements in, equivalence class i and all members of a given equivalence class are equal, $P_i = 1/c_i$ represents the probability of one of the values in class i. Q_i denotes the probability of equivalence class i within the universe. Q_i is computed by taking the number of elements in class i and dividing by the total number of elements in all equivalence classes combined.

In the rough relational database all domains are partitioned into equivalence classes and relations are not restricted to first normal form. We therefore have a type of rough set for each attribute of a relation. This results in a rough relation, since any tuple having a value for an attribute which belongs to the boundary region of its domain is a tuple belonging to the boundary region of the rough relation. There are two things to consider when measuring uncertainty in databases: uncertainty or entropy of a rough relation that exists in a database at some given time and the entropy of a relation schema for an existing relation or query result. We must consider both since the approximation regions only come about by set values for attributes in given tuples. Without the extension of a database containing actual values, we only know about indiscernibility of attributes. We cannot consider the approximation regions.

So we can define the entropy for a rough relation schema as follows:

Definition 8.10. The *rough schema entropy* for a *rough* relation *schema S* is H_{RS} is

$$H_{RS}(S) = -\Sigma_j\ [\Sigma\ Q_i \log(P_i)],\ \text{ for } i = 1,...n;\ j = 1,...,m$$

where there are n equivalence classes of domain j, and m attributes in the schema $R(A_1,A_2,\ldots,A_m)$.

This is similar to the definition of entropy for rough sets without factoring in roughness since there are no elements in the boundary region (lower approximation = upper approximation). However, because a relation is a cross product among the

domains, we must take the sum of all these entropies to obtain the entropy of the schema. The schema entropy provides a measure of the uncertainty inherent in the definition of the rough relation schema taking into account the partitioning of the domains on which the attributes of the schema are defined.

We extend the schema entropy H_{RS} to define the entropy of an actual rough relation instance of some database D by multiplying each term in the product by the roughness of the rough set of values for the domain of that given attribute.

Definition 8.11. The *rough relation entropy* of a particular extension of a schema is H_{RR}:

$$H_{RR}(R) = -\Sigma_j \, D\rho_j(R)[\Sigma \, DQ_i \log(DP_i)], \text{ for } i = 1,...n; \; j = 1,...,m$$

where $D\rho_j(R)$ represents a type of database roughness for the rough set of values of the domain for attribute j of the relation, m is the number of attributes in the database relation, and n is the number of equivalence classes for a given domain for the database. DQ_i is the probability of a tuple in the database relation having a value from class i, and DP_i is the probability of a value for class i occurring in the database relation out of all the values which are given We obtain the $D\rho_j(R)$ values by letting the non-singleton domain values represent elements of the boundary region, computing the original rough set accuracy and subtracting it from one to obtain the roughness. The entropy of an actual rough relation instance $H_{RR}(R)$ is an extension of the schema entropy obtained by multiplying each term in the product by the roughness of the rough set of values for the domain of that given attribute.

Consider the example relations in Figure 8.2 where domains for soil color and size have been defined as

$$\text{COLOR} = \{[black, ebony], [brown, tan, sienna], [white], [gray], [orange]\},$$

and

$$\text{PARTICLE-SIZE} = \{[big, large], [huge, enormous], [medium], [small, little, tiny]\}.$$

The rough relation entropy of the relations SAMPLE-114 and SAMPLE-115 shown in the tables are calculated as follows:

$$\begin{aligned} H_{RS}(SAMPLE-114) = &-(4/7)[(2/5)\log(2/7) + (3/5)\log(3/7) + 0 + \\ &+ (2/5)\log(2/7) + 0] - (2/6)[(2/5\log(2/6) + 0 + \\ &+ (2/5)\log(2/6) + (2/5)\log(2/6)] = .56, \end{aligned}$$

$$\begin{aligned} H_{RS}(SAMPLE-115) = &-(7/7)[(1/2)\log(1/7) + (2/2)\log(2/7) + \\ &+ (2/2)\log(2/7) + (1/2)\log(1/7) + (1/2)\log(1/7)] - \\ &- (5/5)[(1/2)\log(1/5) + (1/2)\log(1/5) + \\ &+ (2/2)\log(2/5) + (1/2)\log(1.5)] = \\ &= 3.7821 \end{aligned}$$

SAMPLE-114

BIN-NO	COLOR	PARTICLE-SIZE
P21	brown	medium
P22	{black, tan}	large
P23	gray	{medium, small}
T01	black	tiny
T04	{gray, brown}	large

SAMPLE-115

BIN-NO	COLOR	PARTICLE-SIZE
M43	{black, tan, white}	{big, huge, medium}
M46	{brown, orange, white, gray}	{medium, small}

Fig. 8.2 Rough relations for entropy analysis.

In the second sample the specific relationships between the values of the color and particle size are less exactly specified than in first sample and so more secure. This correlates to the entropy being higher in the second sample. From this example it is clear that our concept of security in the rough relational database corresponds to uncertainty in this sense, so we can use these measures of entropy as a quantitative measure for security in a rough relational database.

8.7 Example of Use of Rough Spatial Data

Many of the problems associated with data are prevalent in all types of database systems. Spatial databases and geographic information systems (GIS) contain descriptive as well as positional data. The various forms of uncertainty occur in both types of data, so many of the issues apply to ordinary databases as well, such as integration of data from multiple sources, time-variant data, uncertain data, imprecision in measurement, inconsistent wording of descriptive data, and "binning" or grouping of data into fixed categories, also are employed in spatial contexts.

First consider an example of the use of rough sets in representing spatially related data. Let

$U = \{\text{tower, stream, creek, river, forest, woodland, pasture, meadow}\}$

and let the equivalence relation R be defined as follows:

$R^* = \{[\text{tower}], [\text{stream, creek, river}], [\text{forest, woodland}], [\text{pasture, meadow}]\}.$

Given some set

$X = \{\text{tower, stream, creek, river, forest, pasture}\},$

we would like to define it in terms of its lower and upper approximations:

$$\underline{R}X = \{tower, stream, creek, river\},$$

and

$$\overline{R}X = \{tower, stream, creek, river, forest, woodland, pasture, meadow\}.$$

A *rough set* in A is the group of subsets of U with the same upper and lower approximations. In the example given, the rough set is

{ {tower, stream, creek, river, forest, pasture }
{tower, stream, creek, river, forest, meadow }
{tower, stream, creek, river, woodland, pasture }
{tower, stream, creek, river, woodland, meadow} }.

Often spatial data is associated with a particular grid. The positions are set up in a regular matrix-like structure and data is affiliated with point locations on the grid. This is the case for raster data and for other types of non-vector type data such as topography or sea surface temperature data. There is a tradeoff between the resolution or the scale of the grid and the amount of system resources necessary to store and process the data. Higher resolutions provide more information, but at a cost of memory space and execution time.

If we approach these data issues from a rough set point of view, it can be seen that there is indiscernibility inherent in the process of gridding or rasterizing data. A data item at a particular grid point in essence may represent data near the point as well. This is due to the fact that often point data must be mapped to the grid using techniques such as nearest-neighbor, averaging, or statistics. The rough set indiscernibility relation may be set up so that the entire spatial area is partitioned into equivalence classes, where each point on the grid belongs to an equivalence class. If the resolution of the grid changes, then, in fact, this is changing the granularity of the partitioning, resulting in fewer, but larger classes.

The approximation regions of rough sets are beneficial whenever information concerning spatial data regions is accessed. Consider a region such as a forest. One can reasonably conclude that any grid point identified as **forest** that is surrounded on all sides by grid points also identified as **forest** is, in fact, a point represented by the feature **forest**. However, consider points identified as **forest** that are adjacent to points identified as **meadow**. Is it not possible that these points represent meadow area as well as **forest** area but were identified as forest in the classification process? Likewise, points identified as **meadow** but adjacent to **forest** points may represent areas that contain part of the forest. This uncertainty maps naturally to the use of the approximation regions of the rough set theory, where the lower approximation region represents certain data and the boundary region of the upper approximation represents uncertain data. It applies to spatial database querying and spatial database mining operations.

If we force a finer granulation of the partitioning, a smaller boundary region results. This occurs when the resolution is increased. As the partitioning becomes finer and finer, finally a point is reached where the boundary region is non-existent. Then the upper and lower approximation regions are the same and there is no uncertainty in the spatial data as can be determined by the representation of the model.

8.8 Conclusion

We have provided discussions of how it is possible to design relational databases to allow the incorporation of uncertain data characterized using rough set theory. This included Entity-Relationship modeling, rough functional dependencies and rough normal forms. Security issues as dealt with in statistical databases were also discussed as well as an example of the representation of uncertain spatial data by rough sets.

Some interesting possible new directions that were pointed out by a reviewer include how to learn the most efficient (resulting in crispest answers) with at the same time, the most meaningful (for the database users) columns' domain partitions and also how to extend the columns' domain partitions into coverings.

Acknowledgments. The authors would like to thank the Naval Research Laboratory's Base Program, Program Element No. 0602435N for sponsoring this research.

References

1. Beaubouef, T., Petry, F.: Rough Querying of Crisp Data in Relational Databases. In: Proc. Third International Workshop on Rough Sets and Soft Computing (RSSC 1994), San Jose, California, pp. 368–375 (1994)
2. Beaubouef, T., Petry, F.: Fuzzy Rough Set Techniques for Uncertainty. Processing in a Relational Database. International Journal of Intelligent Systems 15, 389–424 (2000)
3. Beaubouef, T., Petry, F.: Rough Functional Dependencies. In: Multiconferences: International Conference on Information and Knowledge Engineering (IKE 2004), Las Vegas, pp. 175–179 (2004)
4. Beaubouef, T., Petry, F., Ladner, R.: Normalization in a Rough Relational Database. In: Ślęzak, D., Wang, G., Szczuka, M.S., Düntsch, I., Yao, Y. (eds.) RSFDGrC 2005. LNCS (LNAI), vol. 3641, pp. 275–282. Springer, Heidelberg (2005)
5. Beaubouef, T., Petry, F., Arora, G.: Information-theoretic measures of uncertainty for rough sets and rough relational databases. Information Sciences 109, 185–195 (1998)
6. Beaubouef, T., Petry, F., Buckles, B.: Extension of the relational database and its algebra with rough set techniques. Computational Intelligence 11, 233–245 (1995)
7. Buckles, B., Petry, F.: A fuzzy model for relational databases. Int. Jour. Fuzzy Sets and Systems 7, 213–226 (1982)
8. Chen, P.: The entity-relationship model–toward a unified view of data. ACM Transactions on Database Systems 1, 9–36 (1976)
9. Chin, F., Ozsoyoglu, G.: Statistical database design. Transactions on Database Systems 6, 113–139 (1981)

10. Codd, E.: A relational model of data for large shared data banks. Communications of the ACM 13, 377–387 (1970)
11. Denning, D., Denning, P.: Data security. ACM Computing Surveys 11, 2–54 (1979)
12. Elmasri, R., Navathe, S.: Fundamentals of Database Systems, 6th edn. Addison-Wesley (2010)
13. Fernandez, E., Summers, R., Wood, C.: Database Security and Integrity. Addison-Wesley, Reading (1981)
14. Galindo, J., Urrutia, A., Piattini, M.: Fuzzy Databases: Modeling, Design and Implementation. Idea Group Publishing, Hershey (2006)
15. Leiss, E.: Principles of Data Security. Plenum Press, New York (1982)
16. Lipski, W.: On databases with incomplete information. Journal of the ACM 28, 41–70 (1981)
17. Makinouchi, A.: A consideration on normal form of not-necessarily normalized relation in the relational data model. In: Proc. of the Third Int. Conf. on Very Large Databases, pp. 447–453 (1977)
18. McLeish, M.: Further results on the security of partitioned dynamic statistical database. Transactions on Database Systems 14, 98–113 (1989)
19. Pawlak, Z.: Rough sets. Int. Journal of Man-Machine Studies 21, 127–130 (1984)
20. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning About Data. Kluwer Academic Publishers, Norwell (1991)
21. Pawlak, Z.: Systemy Informacyjne. Podstawy Teoretyczne (Information Systems. Theoretical Foundations). WNT, Warszawa (1983) (in Polish)
22. Petry, F.: Fuzzy Databases: Principles and Applications. Kluwer Academic Publishers, Boston (1996)
23. Roth, M., Korth, H., Batory, D.: SQL/NF: A query language for non-1NF databases. Information Systems 12, 99–114 (1987)
24. Sakai, H., Koba, K., Nakata, M.: Rough sets based rule generation from data with categorical and numerical values. Journal of Advanced Computational Intelligence and Intelligent Informatics 12, 426–434 (2008)
25. Sakai, H., Okuma, H., Nakata, M.: Ślęzak D. Stable rule extraction and decision making in rough non-deterministic information analysis. Int. J. Hybrid Intell. Syst. 8(1), 41–57 (2011)
26. Ślęzak, D., Synak, P., Wroblewski, J., Toppin, G.: Infobright Analytic Database Engine using rough sets and granular computing. In: IEEE International Conference on Granular Computing, pp. 432–437. IEEE Computer Society Press (2010)
27. Stallings, W., Brown, L.: Computer Security: Principles and Practice. Prentice Hall (2007)
28. Shenoi, S., Melton, A., Fan, L.: Functional dependencies and normal forms in the fuzzy relational database model. Information Sciences 60, 1–28 (1992)
29. Sui, Y., Xia, Y., Wang, J.: The Information Entropy of Rough Relational Databases. In: Wang, G., Liu, Q., Yao, Y., Skowron, A. (eds.) RSFDGrC 2003. LNCS (LNAI), vol. 2639, pp. 320–324. Springer, Heidelberg (2003)
30. Thomas, S., Fischer, P.: Nested relational structures. Advances in Computing Research 3, 269–307 (1989)
31. Yazici, A., Soysal, A., Buckles, B., Petry, F.: Uncertainty in a nested relational database model. Data and Knowledge Engineering 30, 275–301 (1999)
32. Zvieli, A., Chen, P.: Entity-relationship modeling and fuzzy databases. In: Proc. of International Conference on Data Engineering, pp. 320–327 (1986)

Chapter 9
Rough Pragmatic Description Logic

Zbigniew Bonikowski, Edward Bryniarski, and Urszula Wybraniec-Skardowska

Abstract. In this chapter, a rough description logic is built on the basis of a pragmatic standpoint of representation of knowledge. The pragmatic standpoint has influenced the acceptance of a broader definition of the semantic network than that appearing in the literature. The definition of the semantic network is a motivation of the introduced semantics of the language of the descriptive logic. First, the theoretical framework of representation of knowledge that was proposed in the papers [24, 25] is adjusted to the description of data processing. The pragmatic system of knowledge representation is determined, as well as situations of semantic adequacy and semantic inadequacy for represented knowledge are defined. Then, it is shown that general information systems (generalized information systems in Pawlak's sense) presented in the paper [5] can be interpreted in pragmatic systems of knowledge representation. Rough sets in the set-theoretical framework proposed in papers [7, 8] are defined for the general information systems. The pragmatic standpoint about objects is also a motivation to determine a model of semantic network. This model is considered as a general information system. It determines a formal language of the descriptive logic. The set-theoretical framework of rough sets, which was introduced for general information systems, makes it possible to describe the interpretation of this language in the theory of rough sets. Therefore this interpretation includes situations of semantic inadequacy. At the same time, for the class of all interpretations of this type, there exists a certain descriptive logic, which — in this chapter — is called *rough pragmatic description logic*.

Zbigniew Bonikowski
Institute of Mathematics and Informatics, Opole University, Poland
e-mail: zbonik@math.uni.opole.pl

Edward Bryniarski
Institute of Mathematics and Informatics, Opole University, Poland
e-mail: edlog@math.uni.opole.pl

Urszula Wybraniec-Skardowska
Group of Logic, Language and Information, Opole University, Poland
e-mail: uws@uni.opole.pl

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 157–184.
springerlink.com

Keywords: Pragmatic knowledge representation, semantically adequate knowledge, vague knowledge, pragmatic information system, generalized pragmatic information system, rough set, semantic network, description logic, rough pragmatic descriptive logic.

9.1 Introduction

The passing decade has seen a rapid increase in the number of scientific publications dealing with description logic, which were inspired by the research program of semantic network WEB (see cf. [1]). A certain departure from the cognitive standpoint called here pragmatic standpoint of representation of knowledge about objects can be observed in the popular trends of this research, referred to, in brief, as SHIF, SHOIN, SROIQ. This standpoint, making reference to traditional philosophical considerations, links the represented notions (constituents of knowledge about objects) to the expressions of language which are used by agents and which refer to the represented objects. It is aimed to precisely describe such situations in processing the data that lead to semantic adequacy, i.e. an accordance between the represented notions and the objective world being described. Identification of the semantic inadequacy that manifests itself in vagueness, uncertainty or fuzziness of the represented knowledge, becomes possible. A precise description of the pragmatic making use of: notions, meanings of terms and objective references of terms in processes of communication between people, allows arriving at an answer to the question: can AI systems know anything or comprehend anything, thus can they participate in a discourse with humans, in a pragmatic sense, in the manner that is similar to human's (like in Turing's test [23])? In other words, this is a question about whether there exist systems of a pragmatic artificial intelligence. For this reason the pragmatic standpoint of representation of knowledge about objects can be important for the AI research.

In this chapter, having been inspired by papers [10, 11], in contrast to their authors' ideas, though, we will build a rough description logic. The pragmatic standpoint of representation of knowledge will also influence the acceptance of a broader definition of the semantic network than that appearing in the literature. The definition of the semantic network will influence the introduced semantics of the language of descriptive logic. First, we will adjust the theoretical framework of representation of knowledge that was proposed by U. Wybraniec-Skardowska [24, 25] to the description of data processing. The *pragmatic system of knowledge representation* will be determined, as well as situations of semantic adequacy and semantic inadequacy for represented knowledge will be defined. Then, it will be shown that general information systems (generalized information systems in Pawlak's sense) presented in the paper (Bonikowski, Wybraniec-Skardowska [5]) can be interpreted in pragmatic systems of knowledge representation. Rough sets in the set-theoretical framework proposed by E. Bryniarski [7, 8] are defined for the general information systems. The pragmatic standpoint of representation of knowledge about objects is also a

motivation to determine a model of semantic network. This model is considered as a general information system. It determines a formal language of descriptive logic DL. The set-theoretical framework of rough sets, which was introduced for general information systems, makes it possible to describe the interpretation of this language in the theory of rough sets. Therefore, this interpretation includes situations of semantic inadequacy. At the same time, for the class of all interpretations of this type, there exists a certain descriptive logic, which we call the *rough pragmatic description logic* (RPDL).

9.2 The Pragmatic System of Knowledge Representation

In the research into the existence of systems of an artificial intelligence it is vital to obtain an answer to the question whether AI systems cannot represent only knowledge, but also if they can "know something", are able to "conceive something". According to the traditional pragmatic standpoint relating to usage of language, knowledge, its representation and understanding this representation, are determined through making use of specimens, *tokens* of expressions of language in the communication process. If we replace the user of language in this process with a certain system of artificial intelligence, then the subject that understands the represented knowledge and learns about this knowledge is the vary system itself. In order to design this behavior of the artificial intelligence system, it is necessary to precisely define the notion of this system. Such a system will be called a *pragmatic system of knowledge representation*.

The most significant properties of natural or artificial languages, such as colloquial speech or programming languages, for a description of the objective world, are specimens, *tokens* of data, specimens of descriptions of objects in an established situation, which are available in these languages. Moreover, possible relations between these specimens, tokens (including operations that can be performed on them) are also available. *Specimens, tokens of data* are concrete occurrences, representatives of an abstractive value called a *datum*. Simultaneously, for the same datum there can exist many of its specimens and all these specimens are used in the same way to indicate an object or objects of a given type. Thus, a datum is a class of specimens used in the same way (analogously). A set of values possible to use in a natural language or a programming language, such as numerical data, inscriptions, records, is usually infinite, although given communicating (a given program) uses, obviously, a finite number of them. For this reason we will consider only language structures determined on finite sets. Any subject of communicating that uses expressions of a language in a way comprehensible to a human being is called an *agent*. Agents are people, means or systems of information technology (*e.g.* relevant computer programs, suitably programmed computers, computer networks, programs servicing WEB network), as well as all kinds of organizations, human or technological, which process data. Agents group specimens of data and thanks to their analogous usage they obtain data. Next, they connect the data which represent the same elementary

features and properties of objects, obtaining *types of data*. The most frequently encountered types of data are types of numerical data and textual types: relevant sets of numerical data or sets of texts. Classification of specimens of data can occur in such a way, yet it also can take place according to many other criteria. The classification can be performed inside and outside the agent's activity. Due to agents' communicating with one another only external classification of data will be considered, that is one established by relations of common *usage of the data by the agents*. These relations are a kind of *protocols* of communicating: algorithms used to obtain access to data or obtaining the right of access to these data through using specimens of these data, or obtaining the right of access to a specimen of these data. A result of agents' communicating with one another is grouping types of data in certain classes in which the people communicating with one another normally acknowledge to *knowledge*. In this sense, thus in the pragmatic sense, types of data *mean* something, they *represent certain knowledge*. Furthermore, specimens of data refer to certain objects established by data indicating only one of these objects. These data are proper names, singular terms, pronouns, including complex names: addresses of placing the object or addresses of the sources of knowledge which points to this object (*e.g.* addresses of the Internet resources). We will identify the set of all such data with a certain field of *ontology*: a structure encompassing all of the described beings. We will denote this set with Ont. Similarly, we will identify the set of all data such that each of the data is a datum about only one agent, each of the data points to only one agent, with a set of all agents. We will denote this set by $User$. Since agents communicating with one another, identify any agent with the datum $u \in User$, unambiguously point to agent u, and the object is identified with the datum $o \in Ont$, pointing — in an unambiguous manner — to this object o, thus the data u, describing objects that are agents, belong to Ont, which means that $User \subseteq Ont$. Precisely speaking, the system of agents' communicating, in which agents can know something, can be defined in the following way:

Definition 9.1. (see [24]) A pragmatic system of knowledge representation is the multi-sort structure $KRPS = \langle S, \mathbf{S}, User, Ont, use \rangle$, where

A1. $S \neq \emptyset$, $User \neq \emptyset$ and $Ont \neq \emptyset$, S is a set of ***specimens of data*** (***specimens, tokens of language expressions-types*** [24]), $User$ is a set of agents, and Ont is a set of objects indicated by the specimens of data.

A2. use is a relation: $\emptyset \neq use \subseteq User \times S \times Ont$ such that $D_1(use) = User$, $D_2(use) \subseteq S$, $D_3(use) \subseteq Ont$; we call relation use — a ***relation of using of specimens of data*** of set S, expression "$use(u,e,o)$" is read: the ***agent*** u ***uses*** e ***as a specimen of a datum about object*** o.

Relation $\sim \subseteq S \times S$ satisfying the condition: for any $e, e' \in S$,
$e \sim e'$ iff there exist $u \in User, o \in Ont$, such that $use(u,e,o)$ and $use(u,e',o)$,
is called a relation of ***identical usage of specimens of data***. The equivalence class $[e]_\sim = \{e' \in S : e \sim e'\}$ is called a ***datum represented by*** e.

A3. $\mathbf{S} \subseteq 2^S$; for all $e \in S$, $[e]_\sim \in \mathbf{S}$ and for any $\mathbf{e} \in \mathbf{S}$ there exists such a set D of data that $\bigcup D = \mathbf{e}$; $User \subseteq Ont$, $User \subseteq \mathbf{S}$, $Ont \subseteq \mathbf{S}$; the elements of the set $\mathbf{S}$ are called ***data types***, and the elements of data types — ***representatives*** of these types.

In compliance with Def. 9.1, in the pragmatic system of knowledge representation, for any type **e** of data, agents possess procedures of checking whether a specimen $e \in S$ of a datum about the object o, which is used by them, represents type **e**, i.e. whether $[e]_{\sim} \subseteq \mathbf{e}$, and thus $e \in \mathbf{e}$. In this sense, an agent u uses a type **e** of a datum, when for some $e \in \mathbf{e}$ and some $o \in Ont$, the agent u uses a specimen e of a datum about the object o, i.e. $use(u,e,o)$.

Definition 9.2. Let us consider, for the system *KRPS*, the relation $Use = \{\langle u, \mathbf{e}\rangle \in User \times \mathbf{S} :$ for some $e \in \mathbf{e}$ and $o \in Ont, use(u,e,o)\}$. We call the relation Use — a ***relation of usage of data types***. Instead of "$\langle u, \mathbf{e}\rangle \in Use$" we write: $u\ Use\ \mathbf{e}$ (we read: the agent u uses the type **e** of data).

From Def. 9.1 and Def. 9.2 there follows immediately

Corollary 9.1. *For each $u \in User$ and some $o \in Ont$ there is such a specimen $e \in S$ of a datum about the object o that $u\ Use\ [e]_{\sim}$.*

In other words, for any agent, as well as for an object, there exists a datum about this object, which is used by this agent. Thus, each relation of the agent and the object in the pragmatic system of knowledge representation can be interpreted as a state of representation of knowledge, in which this representation is a datum used by this agent. This means that an agent u, by using a specimen e of a datum about an object o, *knows something about the object*. Hence, it is right to call the pair $\langle u, o\rangle$ — an ***epistemic state*** in the pragmatic system of knowledge representation. We will denote the set of all epistemic states by U, and the set of epistemic states, in which agent u knows something about objects indicated by the examples of data about type **e** will be denoted by $U(u, \mathbf{e})$:

$$U(u,\mathbf{e}) = \{\langle u,o\rangle \in U : \text{ there exists } e \in \mathbf{e}, \text{ such that } use(u,e,o)\}.$$

In different epistemic states, any agent can — in the same way — use (apply the same procedures of data processing) different data types, e.g. the ones listed below:

1. $\mathbf{e} = [\textit{John-son of Joseph}]_{\sim}$, $\mathbf{e}' = [\textit{John, whose father is Joseph}]_{\sim}$,
2. $\mathbf{e} = [\textit{triangle, one of the angles of which is right}]_{\sim}$,
   $\mathbf{e}' = [\textit{a right-angled triangle}]_{\sim}$,
3. $\mathbf{e} = \bigcup\{[1]_{\sim}, [2]_{\sim}, [3]_{\sim}\}$, $\mathbf{e}' = \bigcup\{[\textit{one}]_{\sim}, [\textit{two}]_{\sim}, [\textit{three}]_{\sim}\}$.

This manner of using two data types for each agent is made precise by the following definition:

Definition 9.3. (*of the same manner of using data types*)
For any data types **e**, **e**′

$\mathbf{e} \equiv \mathbf{e}'$ iff for any agent $u \in User$ there hold the conditions
(i) $u\ Use\ \mathbf{e}$ iff $u\ Use\ \mathbf{e}'$,
(ii) $\{o \in Ont :$ there exists $e \in \mathbf{e}$ such that $use(u,e,o)\}$
$= \{o \in Ont :$ there exists $e' \in \mathbf{e}'$ such that $use(u,e',o)\}$.

We read expression "$\mathbf{e} \equiv \mathbf{e}'$": types $\mathbf{e}$, $\mathbf{e}'$ of data are used in the same manner.

One can ask the following question: When are the data types which are used in the same manner, identical?

Fact 9.1 *If, in the system KRPS, for all* $\mathbf{e} \in \mathbf{S}$*, for each* $e \in \mathbf{e}$ *there holds* $[e]_{\sim} \subseteq \mathbf{e}$*, then for any* $\mathbf{e_1}, \mathbf{e_2} \in \mathbf{S}$*, if* $\mathbf{e_1} \equiv \mathbf{e_2}$*, then* $\mathbf{e_1} = \mathbf{e_2}$.

Observation 9.1 *If two data types* $\mathbf{e}$*,* $\mathbf{e}'$ *are used in the same manner by agent u, then the set of epistemic states in which agent u knows something about the objects indicated by specimens of* $\mathbf{e}$ *is equal to the set of epistemic states, in which agent u knows something about the objects indicated by specimens of* $\mathbf{e}'$*:*

$$U(u, \mathbf{e}) = U(u, \mathbf{e}').$$

Intuitively, if each agent uses different data types about some objects in the same manner, then he has the same *knowledge* about the objects described by the first data type as that described by the second data type, or these data types have the same *meaning*.

Definition 9.4. (*the meaning and knowledge [24, 25]*)

We call the equivalence class $[\mathbf{e}]_{\equiv}$ of the relation $\equiv$ — a ***unit of knowledge*** represented by the data type $\mathbf{e}$ or ***meaning*** of the data type $\mathbf{e}$, while any subset K of the set $\mathbf{K} = \{[\mathbf{e}]_{\equiv} : \mathbf{e} \in \mathbf{S}\}$ of all units of knowledge, will be called ***knowledge*** in the pragmatic system of knowledge representation. The operation that establishes the meanings is the operation $\mu : \mathbf{S} \to \mathbf{K}$, defined by the formula $\mu(\mathbf{e}) = [\mathbf{e}]_{\equiv}$, for any $\mathbf{e} \in \mathbf{S}$. We say about the data type $\mathbf{e} \in \mathbf{S}$, such that a unit of knowledge $[\mathbf{e}]_{\equiv}$ belongs to knowledge K that it ***represents knowledge*** K.

Does the knowledge represented by specimens of data refer to the same objects which are indicated by the specimens? Or are the epistemic states, in which agents possess this knowledge, the same epistemic states, in which the agents — on the basis of specimens of data types representing this knowledge — find out about the same? Pragmatics includes these questions in the sphere of determining the denotation and adequacy of knowledge representation. The following definitions and theorems define this problem area more accurately.

Definition 9.5. (*denotation and operations of denotation of data types as well as units of knowledge [24, 25]*)

The operation $\delta : \mathbf{S} \to 2^{Ont}$ defined by the formula

$$\delta(\mathbf{e}) = \{o \in Ont : \text{ there exist } u \in User, e \in \mathbf{e} \text{ such that } use(u, e, o)\},$$

is called an ***operation of denotation of data types***, and the values of this operation are called ***denotations of data types***.

The operation $\delta_{\mathbf{K}} : \mathbf{K} \to 2^{Ont}$ defined by the formula

$$\delta_{\mathbf{K}}(k) = \{o \in Ont : \text{ there exists } \mathbf{e} \in \mathbf{S} \text{ such that } k = \mu(\mathbf{e}) \text{ and there exist } u \in User, e \in \mathbf{e} \text{ such that } use(u, e, o)\},$$

is called an ***operation of denotation of units of knowledge***, and the values of this operation are called ***denotations of units of knowledge***.

From the above definition there follows immediately

Corollary 9.2. *(of semantic adequacy [24])*

$$\delta(\mathbf{e}) = \delta_{\mathbf{K}}(\mu(\mathbf{e})), \textit{ for any } \mathbf{e} \in \mathbf{S}.$$

The equation of semantic adequacy permits, for any data type $\mathbf{e}$ and knowledge K, to check whether there exists a solution x of equation $\delta(\mathbf{e}) = \delta_{\mathbf{K}}(x)$ which belongs to K and which is the meaning of $\mathbf{e}$, or — in a broader sense — whether there is a unit of knowledge $k \in K$ satisfying the expression $\delta_{\mathbf{K}}(k) \subseteq \delta(\mathbf{e})$, i.e. whether knowledge K is a knowledge about the objects indicated by specimens of data type $\mathbf{e}$. Intuitively, we are asking about the adequacy of knowledge K represented by $\mathbf{e}$, as well as whether this knowledge determines exactly the semantic denotation of data type $\mathbf{e}$, in this sense we are asking whether K is exact for $\mathbf{e}$.

Definition 9.6. We call knowledge K ***semantically adequate*** if there is a data type $\mathbf{e}$ such that for any unit of knowledge $k \in K$ the following conditions hold:

(a) $\delta_{\mathbf{K}}(k) \subseteq \delta(\mathbf{e})$,
(b) there is a data type $\mathbf{e}' \subseteq \mathbf{e}$, for which $k = \mu(\mathbf{e}')$,
(c) there is such a unit of knowledge $k' \in K$ that $k' = \mu(\mathbf{e})$, i.e. there holds the ***condition of semantic adequacy***: $\delta(\mathbf{e}) = \delta_{\mathbf{K}}(\mu(\mathbf{e}))$.

In the opposite case, we call this knowledge ***semantically inadequate***.
We call knowledge K ***semantically exact*** if there is a data type $\mathbf{e}$ that
(d) $\mathbf{e}$ satisfies the equation $\delta(\mathbf{e}) = \bigcup \delta_{\mathbf{K}}(K)$,
(e) for any unit of knowledge $k \in K$ there is a data type $\mathbf{e}' \subseteq \mathbf{e}$, for which $k = \mu(\mathbf{e}')$ and $\mathbf{e} = \bigcup\{\mathbf{e}' : \text{ there is } k \in K \text{ such that } k = \mu(\mathbf{e}')\}$.

Knowledge that is not exact is called ***semantically vague (fuzzy)*** knowledge.

Corollary 9.3. *Knowledge consisting of one unit of knowledge is semantically adequate.*

Corollary 9.4. *Knowledge represented by each of the data contained in a data type is a semantically exact knowledge.*

Corollary 9.5. *Each semantically adequate knowledge is a semantically exact knowledge.*

Corollary 9.6. *Each semantically vague knowledge is semantically inadequate.*

Agents, by using data type $\mathbf{e}$, establish knowledge about objects indicated by specimens of the used data of the type $\mathbf{e}$. In this way, it is determined that agent u possesses knowledge about object o, that is there holds the epistemic state $\langle u, o \rangle$, in which agent u knows something about object o. Thus, data types and their meanings refer to some epistemic states that take place. These references are precisely defined by the following:

Definition 9.7. (*epistemic extensions of data types and units of knowledge*)
The operation $ex\colon \mathbf{S} \to 2^U$ defined by the formula

$$ex(\mathbf{e}) = \{\langle u,o\rangle : o \in Ont, u \in User, \text{ there exists } e \in \mathbf{e} \text{ such that } use(u,e,o)\},$$

is called an ***operation of epistemic denotation of data types***, and the values of this operation are called ***epistemic denotations of data types***.

The operation $ex_{\mathbf{K}}\colon \mathbf{K} \to 2^U$ defined by the formula

$$ex_{\mathbf{K}}(k) = \{\langle u,o\rangle : o \in Ont, u \in User, \text{ there exists } \mathbf{e} \in \mathbf{S} \text{ such that } k = \mu(\mathbf{e}) \text{ and there exist } e \in \mathbf{e} \text{ such that } use(u,e,o)\},$$

is called an ***operation of epistemic denotation of units of knowledge***, and the values of this operation are called ***epistemic denotations of units of knowledge***.

An agent u ***possesses knowledge*** about an object o or the epistemic state $\langle u,o\rangle$ holds iff there is a constituent of knowledge $k \in \mathbf{K}$ such that $\langle u,o\rangle \in ex_{\mathbf{K}}(k)$.

Hence, there follows

Corollary 9.7. *(of epistemic adequacy)*

$$ex(\mathbf{e}) = ex_{\mathbf{K}}(\mu(\mathbf{e})), \textit{ for any } \mathbf{e} \in \mathbf{S}.$$

Similarly as for the theorem of semantic adequacy, the one of epistemic adequacy allows checking, for any data type $\mathbf{e}$, as well as for knowledge K, whether there is a solution x of equation $ex(\mathbf{e}) = ex_{\mathbf{K}}(x)$ which belongs to K and which is a meaning of $\mathbf{e}$, or — in a boarder way — whether there is a unit of knowledge $k \in K$ satisfying the expression $ex_{\mathbf{K}}(k) \subseteq ex(\mathbf{e})$, i.e. whether knowledge K is a knowledge obtained in the same epistemic states as its representation through specimens of data type $\mathbf{e}$. Intuitively, we are asking about the adequacy of knowledge K represented by specimens of $\mathbf{e}$ in accordance to what agents know about objects, and we are also asking whether this knowledge determines the epistemic denotation of data type $\mathbf{e}$ in an exact manner. In this sense, we are asking whether K is exact for $\mathbf{e}$ in compliance with what agents know about the objects being described.

Definition 9.8. We call knowledge K ***epistemically adequate*** if there is a data type $\mathbf{e}$ such that for any unit of knowledge $k \in K$ the following conditions hold:

(a) $ex_{\mathbf{K}}(k) \subseteq ex(\mathbf{e})$,
(b) there is a data type $\mathbf{e}' \subseteq \mathbf{e}$, for which $k = \mu(\mathbf{e}')$,
(c) there is such a unit of knowledge $k' \in K$ that $k' = \mu(\mathbf{e})$, i.e. there holds the ***condition of epistemic adequacy***: $ex(\mathbf{e}) = ex_{\mathbf{K}}(\mu(\mathbf{e}))$.

In the opposite case, we call this knowledge ***epistemically inadequate***.
We call knowledge K ***epistemically exact*** if there is a data type $\mathbf{e}$ that

(d) $\mathbf{e}$ satisfies the equation $ex(\mathbf{e}) = \bigcup ex_{\mathbf{K}}(K)$,
(e) for any unit of knowledge $k \in K$ there is a data type $\mathbf{e}' \subseteq \mathbf{e}$, for which $k = \mu(\mathbf{e}')$ and $\mathbf{e} = \bigcup\{\mathbf{e}' : \text{ there is } k \in K \text{ such that } k = \mu(\mathbf{e}')\}$.

Knowledge that is not exact is called ***epistemically vague (fuzzy)*** knowledge.

From the above definitions of adequacy and the exactness of knowledge we get the following theorems:

Corollary 9.8. *Knowledge consisting of one unit of knowledge is epistemically adequate.*

Corollary 9.9. *Knowledge represented by each of the data contained in a data type is an epistemically exact knowledge.*

Corollary 9.10. *Each epistemically adequate knowledge is an exact knowledge.*

Corollary 9.11. *Each epistemically vague knowledge is epistemically inadequate.*

Summing up, we are going to a broader justification of equivalence of semantic adequacy/exactness and epistemic adequacy/exactness of knowledge.

Theorem 9.1. *Knowledge is semantically adequate iff it is epistemically adequate.*

Proof. ($\Rightarrow$) Let knowledge K be semantically adequate. Then, by conditions (a) and (c) of Def. 9.6, for some data type $\mathbf{e}$ and some unit of knowledge $k_0 = \mu(\mathbf{e}) \in K$, the inclusion $\delta_{\mathbf{K}}(k) \subseteq \delta_{\mathbf{K}}(k_0) = \delta(\mathbf{e})$ holds for all $k \in K$. Let $\langle u,o\rangle \in ex_{\mathbf{K}}(k)$. From the condition (b) of Def. 9.6 it follows that there is a data type $\mathbf{e}'$ such that $\mathbf{e}' \subseteq \mathbf{e}$ and $k = \mu(\mathbf{e}')$. Thus $use(u,e,o)$ holds for some $e \in \mathbf{e}$, and, in consequence, $\langle u,o\rangle \in ex(\mathbf{e})$, according to Def. 9.7. Hence, we get: $ex_{\mathbf{K}}(k) \subseteq ex(\mathbf{e})$. Since $k_0 = \mu(\mathbf{e}) \in K$, all conditions of Def. 9.8 hold for the knowledge K. Thus, knowledge K is epistemically adequate.

($\Leftarrow$) Now, let us assume, conversely that knowledge K is epistemically adequate. Then, by conditions (a) and (c) of Def. 9.8, for some data type $\mathbf{e}$ and some unit of knowledge $k_0 = \mu(\mathbf{e}) \in K$, the inclusion $ex_{\mathbf{K}}(k) \subseteq ex_{\mathbf{K}}(k_0) = ex(\mathbf{e})$ holds for all $k \in K$. Let $o \in \delta_{\mathbf{K}}(k)$. From the condition (b) of Def. 9.8 it follows that there is a data type $\mathbf{e}'$ such that $\mathbf{e}' \subseteq \mathbf{e}$ and $k = \mu(\mathbf{e}')$. Thus, for some $e \in \mathbf{e}$ and some $u \in User$ we have $use(u,e,o)$, which – according to Def. 9.7 – yields $o \in \delta(\mathbf{e})$. Hence, we obtain: $\delta_{\mathbf{K}}(k) \subseteq \delta(\mathbf{e})$. Since $k_0 = \mu(\mathbf{e}) \in K$, all the conditions of Def. 9.6 hold for knowledge K. Thus, knowledge K is semantically adequate. □

Theorem 9.2. *Knowledge is semantically exact iff it is epistemically exact.*

Proof. ($\Rightarrow$) Let knowledge K be semantically exact. Then, by the condition (e) of Def. 9.6, for some data type $\mathbf{e}$ and some unit of knowledge $k \in K$, there is a data type $\mathbf{e}' \subseteq \mathbf{e}$ such that $k = \mu(\mathbf{e}')$ and $\mathbf{e} = \bigcup\{\mathbf{e}' :$ there is $k \in K$ such that $k = \mu(\mathbf{e}')\}$. The following equivalences hold:

$\langle u,o\rangle \in ex(\mathbf{e}) \Leftrightarrow$ there is $e \in \mathbf{e}, use(u,e,o) \Leftrightarrow$ there are $k \in K$ and $\mathbf{e}' \subseteq \mathbf{e}$, such that $k = \mu(\mathbf{e}'), e \in \mathbf{e}'$ and $use(u,e,o) \Leftrightarrow$ there is $k \in K$, such that $\langle u,o\rangle \in ex_{\mathbf{K}}(k) \Leftrightarrow \langle u,o\rangle \in \bigcup ex_{\mathbf{K}}(K)$.

From this, it follows that $ex(\mathbf{e}) = \bigcup ex_{\mathbf{K}}(K)$. Hence, conditions (d) and (e) of Def. 9.8 are satisfied. Thus, knowledge K is epistemically exact.

($\Leftarrow$) Now, let us assume that knowledge K is epistemically exact. Then, by the condition (e) of Def. 9.8, for some data type $\mathbf{e}$ and some unit of knowledge $k \in K$ there is a data type $\mathbf{e}' \subseteq \mathbf{e}$ such that $k = \mu(\mathbf{e}')$ and $\mathbf{e} = \bigcup\{\mathbf{e}' :$ there is $k \in K$ such that $k = \mu(\mathbf{e}')\}$. The following equivalences hold:

$o \in \delta(\mathbf{e}) \Leftrightarrow$ there are $e \in \mathbf{e}$ and $u \in User$, such that $use(u,e,o) \Leftrightarrow$ there are $k \in K$ and $\mathbf{e}' \subseteq \mathbf{e}$, such that $k = \mu(\mathbf{e}')$ and $use(u,e,o)$ holds for some $u \in User$ and some $e \in \mathbf{e}' \Leftrightarrow$ there is $k \in K$, such that $o \in \delta_{\mathbf{K}}(k) \Leftrightarrow o \in \bigcup \delta_{\mathbf{K}}(K)$.

Hence, it follows that $\delta(\mathbf{e}) = \bigcup \delta_{\mathbf{K}}(K)$. Thus, knowledge K is semantically exact because conditions (d) and (e) of Def. 9.6 are satisfied. □

The last two theorems allow speaking, in a short way, about adequate, inadequate, exact, vague knowledge, in the semantic or epistemic sense, as *adequate, inadequate, exact, vague knowledge*.

9.3 Information Systems

Each classification of a specimen e of a datum is carried out in a pragmatic system of knowledge representation $KRPS = \langle S, \mathbf{S}, User, Ont, use\rangle$ through assigning a datum $[e]_\sim$ to the epistemic state $\langle u,o\rangle$, such that $use(u,e,o)$. Let $U = \{\langle u,o\rangle \in User \times Ont :$ there exists $e \in S, use(u,e,o)\}$ be a set of all the epistemic states and V be a set of all the data. Any function $a\colon U \to V$ is called an *attribute* if every epistemic state $\langle u,o\rangle$ corresponds to exactly one datum $[e]_\sim$ about the object o. Intuitively, this is what agent u wants to find out or does learn about object o. Following Z. Pawlak [14], we will call the function $\iota\colon A \times U \to V$, where A is a set of attributes, mapping every pair $\langle a, \langle u,o\rangle\rangle$ to a datum $a(\langle u,o\rangle)$ about the object o — ***information***. Such a use of the name *information* justifies calling the systems of data processing, which are considered further — *information systems* [14,19].

Let $\mathbf{S}'$ be a subfamily of the family $\mathbf{S}$ of sets, that is a covering of $\mathbf{S}$. At the same time, let $\mathbf{S}'$ be a set of data types available to all agents. Let set A be a set of attributes $a\colon U \to V_a \subseteq V$ such that there is exactly one data type $\mathbf{e} \in \mathbf{S}'$, for which $V_a = \{\mathbf{v} \in V : \mathbf{v} \subseteq \mathbf{e}\}$. Moreover, for any pair $\langle u,o\rangle \in U$, there exists such a datum $\mathbf{v} \in V_a$ and a specimen e of this datum that $use(u,e,o)$. Additionally, we assume that for each data type $\mathbf{e} \in \mathbf{S}'$ there exists such an attribute $a \in A$ that $a\colon U \to V_a = \{\mathbf{v} \in V : \mathbf{v} \subseteq \mathbf{e}\}$.

Definition 9.9. We call the system

$$PIS = \langle S, \mathbf{S}, User, Ont, use, U, A, \{V_a\}_{a\in A}\rangle,$$

a ***pragmatic information system***.

If determination of the function $a \in A$, as given above, is impossible, then we will define these functions as functions $a\colon U \to 2^{V_a}$ such that there exists exactly one data type $\mathbf{e} \in \mathbf{S}'$, for which $V_a = \{\mathbf{v} \in V : \mathbf{v} \subseteq \mathbf{e}\}$. Then we will call the system

$$PIS = \langle S, \mathbf{S}, User, Ont, use, U, A, \{V_a\}_{a\in A}\rangle,$$

a ***pragmatic non-deterministic information system***.

We call functions $a \in A$ — ***attributes***.

The definition of pragmatic information system allows applying Pawlak's theory of information systems to an analysis of processing data in pragmatic systems of knowledge representation.

Definition 9.10. (*Pawlak's information systems [9, 13, 14, 16]*)
Let Σ be an ordered system

$$\Sigma = \langle U, A, \{V_a\}_{a \in A} \rangle,$$

where U is a finite set and A is the set of all attributes, V_a is the set of all values of the attributes $a \in A$. We will call any object $x \in U$ an ***information state***, and any value of an attribute — a ***datum***.

If for any $a \in A, a\colon U \to V_a$, then Σ is called an ***information system***.

If for any $a \in A, a\colon U \to 2^{V_a}$, then Σ is called a ***non-deterministic information system***.

Any non-deterministic information system, whose all of the values of the attributes are singletons, will be called a ***deterministic information system***.

Corollary 9.12. *Any pragmatic information system is an information system.*

Agents' communicating with one another concerns most often realization of such pragmatic systems of knowledge representation $KRPS = \langle S, \mathbf{S}, User, Ont, use \rangle$, in which knowledge refers also to relations between distinguished simple objects belonging to set O, $Ont = O \cup O^2 \cup \ldots \cup O^n$. Let $\mathbf{S}' \subseteq \mathbf{S}$ be a certain covering of S, $U = \{\langle u, o \rangle \in User \times Ont :$ there exists $e \in S, use(u, e, o)\}$ and for each $x \in U \cup U^2 \cup \ldots \cup U^n$, if $x = \langle \langle u_1, o_1 \rangle, \langle u_2, o_2 \rangle, \ldots, \langle u_k, o_k \rangle \rangle$, then for $o = \langle o_1, o_2, \ldots, o_k \rangle \in Ont$ there exists a datum $\mathbf{v}$ contained in a data type $\mathbf{e} \in \mathbf{S}'$. Moreover, specimens v of datum $\mathbf{v}$, used by an agent u, refer to an object o, according to the relation $use(u, v, o)$. It can be noticed that in such a system *KRPS* the following definition possesses its interpretation.

Definition 9.11. (*general information systems [5]*)
Let Σ be an ordered system

$$\Sigma = \langle U, A, A_1, A_2, \ldots, A_n, \{V_a\}_{a \in A} \rangle,$$

where U is the finite set and A is the set of all attributes, $A_k (k = 1, 2, \ldots, n)$ is the set of k-ary attributes understood as k-ary functions, V_a is the set of all cods of the attribute $a \in A$.

If for any $a \in A$ and $a \in A_k$, $a\colon U^k \to V_a$, then Σ is called a ***general information system***.

If for any $a \in A$ and $a \in A_k$, $a\colon U^k \to 2^{V_a}$, then Σ is called a ***general non-deterministic information system***.

Any general non-deterministic information system, whose all of the values of the attributes are singletons, will be called a ***general deterministic information system***.

Accepting for the general non-deterministic information system $\Sigma = \langle U, A, A_1, A_2, \ldots, A_n, \{V_a\}_{a\in A}\rangle$ that $V'_a = 2^{V_a}$, we obtain a general information system $\Sigma' = \langle U, A, A_1, A_2, \ldots, A_n, \{V'_a\}_{a\in A}\rangle$.

Fact 9.2 *Each general non-deterministic information system is isomorphic to a general information system.*

Let $\Sigma = \langle U, A, A_1, A_2, \ldots, A_n, \{V_a\}_{a\in A}\rangle$ be a general deterministic information system. Let us replace each attribute $a\colon U^k \to 2^{V_a}$ with an attribute $a'\colon U^k \to V_a$, such that if $a(x_1, x_2, \ldots, x_k) = \{y\}$, then $a'(x_1, x_2, \ldots, x_k) = y$. In this way, we obtain, isomorphic to Σ, an information system $\Sigma' = \langle U, A', A'_1, A'_2, \ldots, A'_n, \{V'_a\}_{a'\in A'}\rangle$.

Fact 9.3 *Each general deterministic information system Σ is isomorphic to a general information system Σ', formed out of system Σ in the way described above.*

Let $\Sigma = \langle U, A, A_1, A_2, \ldots, A_n, \{V_a\}_{a\in A}\rangle$ be a general information system. Let us replace each attribute $a\colon U^k \to V_a$ with attribute $a'\colon U^k \to 2^{V_a}$, such that if $a(x_1, x_2, \ldots, x_k) = y$, then $a'(x_1, x_2, \ldots, x_k) = \{y\}$. In this way, we obtain a general deterministic information system $\Sigma' = \langle U, A', A'_1, A'_2, \ldots, A'_n, \{V'_a\}_{a'\in A'}\rangle$, which is isomorphic to Σ.

Fact 9.4 *Each general information system Σ is isomorphic to the general deterministic information system Σ', formed out of system Σ in the above-described manner.*

Fact 9.5 *Each general information system can be considered as a general non-deterministic information system.*

For any general information system $\Sigma = \langle U, A, A_1, A_2, \ldots, A_n, \{V_a\}_{a\in A}\rangle$ we can establish a new set of attributes $A' = \bigcup\{A'_1, A'_2, \ldots, A'_n\}$, such that

- $A'_1 = A_1$,
- for $k > 1$,
  $a' \in A'_k \Leftrightarrow$ for some $\langle x_1, x_2, \ldots, x_{k-1}\rangle \in U^{k-1}$ and $a \in A_k$, $a' = a_{x_1,x_2,\ldots,x_{k-1}}$, where $a_{x_1,x_2,\ldots,x_{k-1}}\colon U \to V_a$ and $a_{x_1,x_2,\ldots,x_{k-1}}(x) = a(x, x_1, x_2, \ldots, x_{k-1})$ for $x \in U$.

Moreover, we can consider the family of sets $\{V'_{a'}\}_{a'\in A'}$, such that

- $V'_{a'} = V_a$, for $a \in A_1$,
- $V'_{a'} = \{a(x, x_1, x_2, \ldots, x_{k-1}) : a' = a_{x_1,x_2,\ldots,x_{k-1}}, x \in U\}, k > 1$.

Then the structure $\Sigma' = \langle U, A', \{V'_{a'}\}_{a'\in A'}\rangle$ is an information system. From the manner of determining of this system it can be seen that sometimes there can exist other general information systems which determine the same information system Σ' in the same manner.

Fact 9.6 *Each general information system can be considered as an information system.*

Similarly,

Fact 9.7 *Each general non-deterministic information system can be considered as a non-deterministic information system.*

Fact 9.8 *Sometimes there can exist two different general information systems determining the same information system.*

The main aim of information systems is to juxtapose the data obtained in individual states of these systems (to gather information obtained in these states). We can ask the following question: Which states are described by the distinguished types of expressions in the same manner, or — in a different way — are they ***indiscernible*** in such a description?

Definition 9.12. (*indiscernibility of states in an information system [15]*)
For any information system $\Sigma = \langle U, A, \{V_a\}_{a\in A}\rangle$, the relation $\approx \subseteq U \times U$ such that for any $x, y \in U, a \in A, a(x) = a(y)$, is called the ***indiscernibility relation*** in the system Σ.

The family $C = \{[s]_\approx : s \in U\}$ of the equivalence classes of the relation $\approx$ is a partition of the set U of information states. Such a family of equivalence classes of the relation $\approx$ is determined uniquely in every information system. For this family of sets of information states, one can determine — in a unique manner — all sets $X = \bigcup B$, for $B \subseteq C$. We will say about such sets that they are ***deterministic*** in an information system. Is it possible to characterize, by means of exact sets, those that are not exact?

In an analogous way, we can define indiscernibility of information states in the general information system $\Sigma = \langle U, A, A_1, A_2, \ldots, A_n, \{V_a\}_{a\in A}\rangle$. Since the arguments of attributes are elements of the set $U_{gen} = U \cup U^2 \cup \ldots \cup U^n$, therefore $\approx \subseteq U_{gen} \times U_{gen}$, and for any $a \in A, x, y \in U_{gen}$, $x \approx y$ iff $a(x) = a(y)$.

9.4 Approximation in Information Systems

According to the rough set theory of Pawlak [16], in any general information system $\Sigma = \langle U, A, A_1, A_2, \ldots, A_n, \{V_a\}_{a\in A}\rangle$ (including also a non-deterministic information system) the operation of ***lower approximation*** $C^- : 2^{U_{gen}} \to 2^{U_{gen}}$, as well as that of ***upper approximation*** $C^+ : 2^{U_{gen}} \to 2^{U_{gen}}$ for sets $\subseteq U_{gen} = U \cup U^2 \cup \ldots \cup U^n$ can be determined. Let us accept the notation $C = \{[s]_\approx : s \in U_{gen}\}$. For this reason we use the letter 'C' in operations C^-, C^+. A system $\langle U_{gen}, C\rangle$ is called an ***approximation space*** of subsets of U_{gen} [7, 26].

Definition 9.13. (Pawlak [16])
For any $X \subseteq U_{gen}$,

$$C^-(X) = \{x \in U_{gen} : [x]_\approx \subseteq X\},$$
$$C^+(X) = \{x \in U_{gen} : [x]_\approx \cap X \neq \emptyset\},$$
$$Bn(X) = C^+(X) \setminus C^-(X),$$

where the relation $\approx$ is a discernibility relation in the general information system Σ. The operation Bn is called the ***boundary operation***.

Fact 9.9 *(cf. [4, 7]) For any $X, Y \subseteq U_{gen}$,*

1. $C^-(X) = \bigcup\{K \in C : K \subseteq X\}$,
2. $C^+(X) = \bigcup\{K \in C : K \cap X \neq \emptyset\}$,
3. $C^-(X) \subseteq C^+(X)$,
4. *sets $C^-(X)$ and $C^+(X)$ are exact,*
5. *if X is exact, then $C^-(X) = C^+(X) = X$.*

Definition 9.14. (*indiscernibility of sets of information states*)
Any sets of information states $X, Y \subseteq U_{gen}$ are ***indiscernible***, which we write down as follows: $X \sim Y$ iff

$$C^-(X) = C^-(Y),$$
$$C^+(X) = C^+(Y).$$

The relation $\sim$ is an equivalence relation. We denote equivalence classes $[X]_\sim$ of this relation with the representative X by X_C. We denote $\emptyset_C$ by 0_C and $(U_{gen})_C$ by 1_C.

Definition 9.15. We call equivalence classes of the relation $\sim$ — ***rough sets*** in the information system Σ. We call the rough sets of exact representatives — ***exact rough sets***.

Definition 9.16. (*an element of a rough set [7, 8]*)
For any $X, Y \subseteq U_{gen}$,

$$X \in_C Y_C \text{ iff } X \neq \emptyset \text{ and there is such } x \in U_{gen}, \text{ that } X \subseteq [x]_\sim,$$
$$C^-(X) \subseteq C^-(Y) \text{ and } [x]_\sim \subseteq C^+(Y).$$

We call the relation $\in_C$ — a ***rough membership relation***. We read the expression "$X \in_C Y_C$": X is a rough element of the rough set Y_C.

Using the relation $\in_C$, one can define inclusion of rough sets.

Definition 9.17. For any $X, Y \subseteq U_{gen}$,

$$X_C \subseteq_C Y_C \text{ iff for every } Z \subseteq U_{gen}, \text{ if } Z \in_C X_C, \text{ then } Z \in_C Y_C.$$

Fact 9.10

1. *$X_C \subseteq_C Y_C$ iff $C^-(X) \subseteq C^-(Y)$ and $C^+(X) \subseteq C^+(Y)$,*
2. *$X_C = Y_C$ iff $C^-(X) = C^-(Y)$ and $C^+(X) = C^+(Y)$,*
3. *$X_C = Y_C$ iff for every $Z \subseteq U_{gen}$, $Z \in_C X_C$ iff $Z \in_C Y_C$,*
4. *It is not the case, that there is $Z \subseteq U_{gen}$ such that $Z \in_C 0_C$,*
5. *For every $X \subseteq U_{gen}, X_C \subseteq_C 1_C$.*

Bryniarski, in his works ([7, 8]), defines the operations of addition $\cup_C$, multiplication $\cap_C$, substraction $\setminus_C$ and complement $'^C$ of rough sets in the family of rough sets:

Definition 9.18. For any rough sets X_C, Y_C, for any $Z \subseteq U_{gen}$,

$$\begin{aligned} Z \in_C X_C \cup_C Y_C &\text{ iff } Z \in_C X_C \text{ or } Z \in_C Y_C, \\ Z \in_C X_C \cap_C Y_C &\text{ iff } Z \in_C X_C \text{ and } Z \in_C Y_C, \\ Z \in_C X_C \setminus_C Y_C &\text{ iff } Z \in_C X_C \text{ and not } Z \in_C Y_C, \\ Z \in_C (X_C)^{'C} &\text{ iff } Z \in_C U_C \setminus_C X_C. \end{aligned}$$

Let us denote the family of all rough sets in the information system Σ by $R(\Sigma)$.

Fact 9.11 *([6–8]) For any rough sets X_C , Y_C, there is the set $Z \subseteq U_{gen}$ such that $X_C \cup_C Y_C = Z_C$ or $X_C \cap_C Y_C = Z_C$ or $X_C \setminus_C Y_C = Z_C$ or $(X_C)^{'C} = Z_C$.*

It was shown in [6, 7] that

Theorem 9.3. *The structure $\langle R(\Sigma), \cup_C, \cap_C, 0_C, 1_C\rangle$ is a distributive lattice with the zero 0_C and the unit 1_C.*

On the other hand, in [3] it is proved that

Theorem 9.4. *The structure $\langle R(\Sigma), \cup_C, \cap_C, '^C, 0_C, 1_C\rangle$ is a Stone algebra.*

Moreover,

Theorem 9.5. *The structure $\mathbf{R}(\Sigma) = \langle R(\Sigma), \cup_C, \cap_C, \setminus_C, '^C, 0_C, 1_C\rangle$ restricted to exact rough sets is homomorphic to a set-theoretical field of sets.*

Let us extend the structure $\mathbf{R}(\Sigma)$ by relations of rough membership and inclusion of rough sets. Let us introduce, for any family $\mathbf{A} \subseteq \mathbf{R}(\Sigma)$, the generalized sum $\bigcup_C \mathbf{A}$:

$$X \in_C \bigcup_C \mathbf{A} \text{ iff } \text{ there is } Y_C \in \mathbf{A} \text{ such that } X \in_C Y_C.$$

Now, in a way analogous with the set-theoretical construction of approximation of sets (homomorphic with these constructions), one can provide the construction of approximation of rough sets in the ***approximation space*** $\langle (U_{gen})_C, \{K_C : K \in C\}\rangle$ ***of rough sets***:

Definition 9.19. (*approximation of rough sets*)
For any $X_C \in R(\Sigma)$,

$$\begin{aligned} F^-(X_C) &= \bigcup_C \{K_C : K \in C, K_C \subseteq_C X_C\}, \\ F^+(X_C) &= \bigcup_C \{K_C : K \in C, K_C \cap_C X_C \neq 0_C\}, \\ Fbn(X_C) &= F^+(X_C) \setminus_C F^-(X_C). \end{aligned}$$

We call these operations, respectively: ***lower approximation, upper approximation*** and ***boundary approximation*** of the rough set X_C.

Fact 9.12 *For any* $X_C \in R(\Sigma)$,

1. $F^-(X_C) = (C^-(X))_C$,
2. $F^+(X_C) = (C^+(X))_C$,
3. $Fbn(X_C) = (Bn(X))_C$,
4. $F^-(X_C) = \{C^-(X)\}$,
5. $F^+(X_C) = \{C^+(X)\}$,
6. $Fbn(X_C) = \{Bn(X)\}$.

Theorem 9.6. *Let* $\langle U_{gen}, C\rangle$ *be an approximation space of subsets of* U_{gen} *and* $\langle (U_{gen})_C, \{K_C : K \in C\}\rangle$ *be a corresponding approximation space of rough sets. The structure* $\langle 2^{U_{gen}}, C^-, C^+, Bn\rangle$ *is homomorphic to the structure* $\langle R(\Sigma), F^-, F^+, Fbn\rangle$.

Proof. (sketch)

Let $h\colon 2^{U_{gen}} \to R(\Sigma)$ be a function such that $h(X) = X_C$ for $X \subseteq U_{gen}$. The function h is a homomorphism from $\langle 2^{U_{gen}}, C^-, C^+, Bn\rangle$ to $\langle R(\Sigma), F^-, F^+, Fbn\rangle$, i.e.:

- if $X \subseteq Y$, then $h(X) \subseteq_C h(Y)$,
- $h(C^-(X)) = F^-(h(X))$,
- $h(C^+(X)) = F^+(h(X))$,
- $h(BnX)) = Fbn(h(X))$,

for any $X, Y \subseteq U_{gen}$. □

The above theorem allows providing the properties of the upper and lower approximation operations, as well as boundary operation of rough sets as homomorphic to the standard properties:

Corollary 9.13. *(cf. [17]) For any* $X, Y \subseteq U_{gen}$,

1. $F^-(X_C) \subseteq_C F^+(X_C)$,
2. $F^-(X_C) \subseteq_C X_C \subseteq_C F^+(X_C)$,
3. $F^-(F^-(X_C)) = F^-(X_C)$,
4. $F^+(F^-(X_C)) = F^-(X_C)$,
5. $Fbn(F^-(X_C)) = 0_C$,
6. $F^-(F^+(X_C)) = F^+(X_C)$,
7. $F^+(F^+(X_C)) = F^+(X_C)$,
8. $Fbn(F^+(X_C)) = 0_C$,
9. $F^-(Fbn(X_C)) = Fbn(X_C)$,
10. $F^+(Fbn(X_C)) = Fbn(X_C)$,
11. $Fbn(Fbn(X_C)) = 0_C$,
12. $F^-(X_C) \cup_C F^-(Y_C) \subseteq_C F^-(X_C \cup_C Y_C)$,
13. $F^-(X_C \cap_C Y_C) = F^-(X_C) \cap_C F^-(Y_C)$,
14. $F^+(X_C \cup_C Y_C) = F^+(X_C) \cup_C F^+(Y_C)$,
15. $F^+(X_C \cap_C Y_C) \subseteq_C F^+(X_C) \cap_C F^+(Y_C)$,
16. *If* X_C *is an exact rough set, then* $F^-(X_C) = F^+(X_C) = X_C$ *and* $Fbn(X_C) = 0_C$,
17. $X_C \subseteq_C Y_C$ *iff* $F^-(X_C) \subseteq_C F^-(Y_C)$ *and* $F^+(X_C) \subseteq_C F^+(Y_C)$,
18. $X \in_C Y_C$ *iff* $X_C \subseteq_C Y_C$, $F^-(X_C) = 0_C$ *and there is an exact rough set* K_C *such that* $F^+(X_C) = K_C$.

We will call the structure $\mathbf{F}(\Sigma) = \langle R(\Sigma), F^-, F^+, Fbn, \cup_C, \cap_C, \setminus_C, {}^{'C}, 0_C, 1_C, \in_C, \subseteq_C \rangle$ — a ***set-theoretical structure of rough sets*** determined by the information system Σ.

Let $PIS = \langle S, \mathbf{S}, User, Ont, use, U, A, \{V_a\}_{a \in A} \rangle$ be any non-deterministic pragmatic information system. Since *PIS* is a non-deterministic information system, we can determine — within this system — the operation of ***lower approximation*** $C^-\colon 2^U \to 2^U$ and that of ***upper approximation*** $C^+\colon 2^U \to 2^U$ for sets of epistemic states $X \subseteq U$, in compliance with the rough set theory of Pawlak. Let us determine — in the *PIS* system — the family $Ex = \{ex(\mathbf{e}) : \mathbf{e} \in \mathbf{S}\}$ of all epistemic extensions.

Then, the approximated epistemic extensions in the *PIS* system are rough sets determined by representatives of the set Ex. And thus, the pragmatic information system can be extended not only by approximated epistemic extensions, but also by rough sets represented by data types and by constituents of knowledge:

Definition 9.20. Two types $\mathbf{e}_1, \mathbf{e}_2 \in \mathbf{S}$ are ***semantically indiscernible*** (which we write down as: $\mathbf{e}_1 \sim_{\mathbf{S}} \mathbf{e}_2$) iff

$$C^-(ex(\mathbf{e}_1)) = C^-(ex(\mathbf{e}_2)),$$
$$C^+(ex(\mathbf{e}_1)) = C^+(ex(\mathbf{e}_2)).$$

We call the equivalence class $[\mathbf{e}]_{\sim_{\mathbf{S}}}$ of the relation $\sim_{\mathbf{S}}$ – a ***rough data type of representative*** $\mathbf{e}$.

Two constituents of knowledge $k_1, k_2 \in \mathbf{K}$ are ***semantically indiscernible*** (which we write down as: $k_1 \sim_{\mathbf{K}} k_2$) iff

$$C^-(ex_{\mathbf{K}}(k_1)) = C^-(ex_{\mathbf{K}}(k_2)),$$
$$C^+(ex_{\mathbf{K}}(k_1)) = C^+(ex_{\mathbf{K}}(k_2)).$$

We call the equivalence class $[k]_{\sim_{\mathbf{K}}}$ of the relation $\sim_{\mathbf{K}}$ – a ***rough unit knowledge*** of the representative k or a ***rough meaning*** of the rough data type $[\mathbf{e}]_{\sim_{\mathbf{S}}}$, when $\mu(\mathbf{e}) = k$.

Like in a pragmatic system of knowledge representation, we will define the operation μ^C of the rough meaning by means of the formula: for any $\mathbf{e} \in \mathbf{S}$

$$\mu^C([\mathbf{e}]_{\sim_{\mathbf{S}}} = [k]_{\sim_{\mathbf{K}}}, \text{ where } \mu(\mathbf{e}) = k,$$

and the rough epistemic extensions by means of the formulas:

$$ex^C([\mathbf{e}]_{\sim_{\mathbf{S}}}) = (ex(\mathbf{e}))_C,$$
$$ex^C_{\mathbf{K}}([k]_{\sim_{\mathbf{K}}}) = (ex(\mathbf{e}))_C, \text{ when } \mu^C([\mathbf{e}]_{\sim_{\mathbf{S}}}) = [k]_{\sim_{\mathbf{K}}}.$$

Hence, there follows

Fact 9.13 *(of adequate deffuzzyfication and sharpening of vague knowledge)* *For any* $\mathbf{e} \in \mathbf{S}$, $ex^C([\mathbf{e}]_{\sim_{\mathbf{S}}}) = ex^C_{\mathbf{K}}(\mu^C([\mathbf{e}]_{\sim_{\mathbf{S}}}))$.

Let us notice that if the knowledge $\{\mu(\mathbf{e})\}$ is vague, then there does not hold the condition of epistemic adequacy $ex(\mathbf{e}) = ex_{\mathbf{K}}(\mu(\mathbf{e}))$, for $\mathbf{e} \in \mathbf{S}$. Despite this, for

the rough data type $[\mathbf{e}]_{\sim_\mathbf{S}}$, there holds the equation $ex^C([\mathbf{e}]_{\sim_\mathbf{S}}) = ex^C_\mathbf{K}(\mu^C([\mathbf{e}]_{\sim_\mathbf{S}}))$, corresponding to this condition. Intuitively, one can regard the rough knowledge $\{\mu^C([\mathbf{e}]_{\sim_\mathbf{S}})\}$ as adequate. For this reason, we will call the above fact — a *fact of adequate sharpening of vague knowledge*.

Observation 9.2 *Determining a set of specimens of data for knowledge that consists of units of rough knowledge and determining appropriate data, as well as data types, we will obtain some other pragmatic system of representation of data, in which knowledge that is determined by these data types is adequate and exact.*

9.5 The Proposed Description Logic – The Rough Pragmatic Description Logic

In this section we will use the earlier defined concepts to propose an outline of some description logic, which will be called the ***rough pragmatic description logic*** (RPDL).

Accepting that we have an information system determined by a pragmatic system of knowledge representation, we can name — in this system — every epistemic state belonging to the distinguished set NO, and also consider the set RS of all the pairs: the value ds_n of n-argument attribute and n-tuple $\langle s_1, s_2, \ldots, s_n\rangle \in NO^n$ of epistemic states, for which this attribute has the given value ds_n. We obtain, then, another system of knowledge representation, other than the information system, determined by epistemic states and relations between these states. This system is called a ***semantic network*** [12].

Definition 9.21. (*semantic network*)

We call a ***semantic network*** the following ordered system:

$$SN = \langle NO, IN, RS, \{DS_i\}_{i\in N, i<n+1}\rangle,$$

where the sets:

NO (network objects) — a set of *nodes* of the semantic network representing the objects being described,

IN (individual names) — a set of unit names of the described objects or pronouns which point individually to the objects being described,

RS (relational structure) — a set of *edges* of the network that determines a relational structure,

$DS_n, n \in N$ - sets of the descriptions, called sets of *descriptions of n-argument relations*,

satisfy the following conditions:

- every name of the set IN corresponds to, mutually unequivocally, only one node;
- $RS \subseteq (DS_1 \times NO) \cup (DS_2 \times NO^2) \cup \ldots \cup (DS_n \times NO^n)$;
- $ds_i \in DS_i$ is a description of the relation R such that

$$\{ds_i\} \times R = (\{ds_i \times NO^i) \cap RS.$$

When, within the definiton of the set RS, we substitute the set NO by the set IN, the set RS will transform into the following set

$$AS \subseteq (DS_1 \times IN) \cup (DS_2 \times IN^2) \cup \ldots \cup (DS_n \times IN^n),$$

where $ds_i \in DS_i$ is a desription of the relation R such that

$$\{ds_i\} \times R = (\{ds_i\} \times IN^i) \cap AS.$$

We call elements of the set AS — ***assertions***. We call any relation R, when it is a one-argument one, satisfying the above equation — a ***concept*** (a notion), and when R is, at the least, a two-argument one — a ***role***. For example, the role of "filiality" that links the person bearing the name "John" with the one named "Charles", the latter being the father to the former, leads to the assertion: ⟨filiality, John, Charles⟩, which we also write down: *filiality(John, Charles)* or *(John,Charles):filiality*. We write the assertion expressed in the sentence "Eve is sitting between John and Charles" in the following way:

sitting_between(Eve, John, Charles) or *(Eve, John, Charles): sitting_between*.

Let us notice that when in the following three *(Eve, John, Charles)*, cyclically, we reverse the names, we will obtain the following three *(John, Charles, Eve)*, which is also an occurrence of a role, *e.g.* expressed in the sentence "John and Charles are sitting beside Eve". We can write down this assertion as follows: *(John, Charles, Eve):are sitting_beside*. We will say about the role *sitting_beside* that it is ***cyclically reverse*** towards that of *sitting_between*. When the three *(Eve, John, Charles)*, which is an occurrence of the assertion *sitting_between*, is reduced by the first name, then the pair *(John, Charles)*, is also an occurrence of the assertion, *e.g.* one expressed in the sentence "someone is sitting between John and Charles": *(John, Charles): someone_sitting_between*. We will say about this role that it is a reduction of that *sitting_between*.

For any description of the relation R (a concept or a role) there is the characteristic function $a\colon IN^i \to V_a \subseteq \{0,1\}$ such that when $\langle ds_i, x_1, x_2, \ldots, x_i\rangle \in AS$, $a(x_1, x_2, \ldots, x_i) = 1$, and for $\langle ds_i, x_1, x_2, \ldots, x_i\rangle \notin AS$, $a(x_1, x_2, \ldots, x_i) = 0$. Let us denote the set of all such characteristic functions by A. Let $V_a = a(IN^i)$. Then the structure $\Sigma(SN) = \langle U, A, \{V_a\}_{a\in A}\rangle$, where $U = IN$, will be an information system.

Definition 9.22. We call the information system $\Sigma(SN)$ — an ***information system determined by the semantic network*** SN.

In the context of research of *Semantic Web* (for example in the works [1]), knowledge representation in the semantic network is determined by two systems of representation: the terminology called **TBox**, as well as a set of representation of assertion called **ABox**.

A semantic network can be extended with nodes which render available the knowledge about concepts or roles, and also extended with boundaries determining dependences between concepts or roles. Descriptions of these dependences are called ***axioms***, and the system of representing this knowledge is called **RBox**. In the presented research trend, the base of the knowledge represented in the

semantic network, in the descriptive language *AL* (*attributive language*) of the logic *DL* (*Description Logic*), is defined as the following triple $\langle Ab, Tb, Rb\rangle$, where the sets Ab, Tb, Rb are finite sets of expressions (descriptions of nodes of a semantic network) that can be computer-processed, respectively: assertions, concepts, axioms. Describing rough knowledge in information systems determined by the semantic network *SN*, we will apply the suitably reformulated and extended language *AL*. We will call this language — ***rough pragmatic language*** (***RPL***; a set of its expressions will be denoted by *RPL*). It will be the language of the proposed logic RPDL.

9.5.1 Syntax of the Language RPL

Let the data be some non-empty sets: individual variables, individual names, names of concepts, names of roles and symbols of modifying agents of concepts.

Syntax of occurrences of concepts and roles

Occurrences of a concept are the symbols $x, y, z, v, \ldots, x_1, y_1, \ldots$ variables and the symbols $a, b, c, \ldots, a_1, b_1, \ldots$, determining individual names. The *variables run over individual names*.

Occurrences of a role are tuples $(t_1, t_2, \ldots, t_k)$ of occurrences of concepts.

Syntax of TBox

The following names belong to the set of names of concepts and roles:

$\top$ (*Top*) – **universal concept** and **universal role**,

$\bot$ (*Bottom*) – **empty concept** and **empty role**.

Top includes all occurrences of concepts and roles, and *Bottom* includes knowledge about a lack of any occurrences of concepts and roles.

$\{t\}$ – **singleton of the occurrences of** t, a concept determined univocally by an occurrence of concept t,

$\{(t_1, t_2, \ldots, t_k)\}$ – a role being a **singleton of** n**-tuple of occurrences**.

Let A, B be the names of concepts, R be the name of a role, a m — the symbol of a **modifier**, then the following are concepts:

$\neg A$ — **negation of a concept** — the expression denoting all the occurrences of concepts that are not occurrences of the concept A;

$A \wedge B$ — **intersection (conjunction) of the concepts** A and B — the expression denoting all the occurrences of the concepts A and B;

$A \vee B$ — **union (alternative) of the concepts** A and B — the expression denoting all the occurrences of the concept A or the concept B;

$A \setminus B$ — **difference of the concepts** A and B — the expression denoting all the occurrences of the concept A, which are not an occurrence of the concept B;

$\exists A.R$ — **existential quantification** — the concept, whose occurrences are those of the concept A remaining in the role R in the first place, at least once with appearances of some concepts related to the role R in the successive places of the role;

$\forall A.R$ — **general quantification** — the concept, whose occurrences are those of the concept A, remaining in the role R in the first place, together with all the occurrences of some concepts related to the role R in the successive places of this role;

$m(A)$ — **modification** m **of the concept** A — denoting a concept that is the concept C altered by the word m, *e.g.* m can have such occurrences as: very much, more, the most, or high, higher, the highest; in the approximated calculus, modifications are lower approximation or upper approximation, or the boundary – $m \in \{Upper, Lower, Boundary\}$;

$m(R)$ –* **modification** m **of the role** $R - m \in \{Upper, Lower, Boundary\}$;

R^{-1} — **the role cyclically reverse** to the role R;

$A.R$ — **restriction of the occurrences of the role** R **by the occurrences of the concept** A – such a role that the occurrences of A remain in the role R in the first place, together with those of some concepts in the other places of the role R;

R^{-} — **reduction of the role** R by the first argument — is the role for at least a three-argument role R, and the concept for a two-argument role;

$A_1 \times A_2 \times \ldots \times A_n$ — **Cartesian product** of the concepts $A_1, A_2, \ldots, A_n$.

Syntax of ABox

For any variables x, y, individual names a, b, the names of the concept C and those of the role R, which is a two-argument one, assertions are denoted by means of expressions in the form "$x : C$", "$a : C$", "$(x,y) : R$", "$(a,y) : R$", "$(x,b) : R$", "$(a,b) : R$". Generally, for the n-argument role R, expressions of assertion take on the form "$(t_1, t_2, \ldots, t_n) : R$", where t_i are any occurrences of the concepts. Inscriptions of the form "$t_1, : A$", "$(t_1, t_2, \ldots, t_k) : R$" are read as follows: t_1 *is an occurrence of the concept A, n-th tuple* $(t_1, t_2, \ldots, t_k)$ *is an occurrence of the role R.*

Syntax of RBox

For any names of the concepts A, B, the names of the roles R_1, R_2, as well as expressions of the assertion α, β, ***axioms*** are expressions rendered in the following form:

$A \subseteq B$ – *inclusion of the concepts A, B*,

$A = B$ – *identity of the concepts A, B*,

$R_1 \subseteq R_2$ – *inclusion of the roles* R_1, R_2,

$R_1 = R_2$ – *identity of the roles* R_1, R_2,

$\alpha :- \beta$ – *Horn's clause for the assertion* α, β; we read this in the following way: *if there holds an occurrence of the assertion* β, *then there holds an occurrence of the assertion* α .

9.5.2 The Distinguished Axioms for RPDL

Let us distinguish some selected axioms for RPDL, divided into three groups.

For any names of concepts or roles A, B, C and R and any occurences $t, t_1, t_2, \ldots, t_k$

Ax.1 $\top = \neg\bot, A \subseteq \top, \bot \subseteq A, A \subseteq A$,
Ax.2 $A = A, R = R$,
Ax.3 $A \vee \bot = A, A \wedge \top = A$,
Ax.4 $A \wedge \bot = \bot, A \vee \top = \top$,
Ax.5 $A \vee B = B \vee A, A \wedge B = B \wedge A$,
Ax.6 $(A \vee B) \wedge C = (A \wedge C) \vee (B \wedge C), (A \wedge B) \vee C = (A \vee C) \wedge (B \vee C)$,
Ax.7 $A \vee \neg A \subseteq \top, A \wedge \neg A = \bot$,
Ax.8 $\neg A \subseteq \top \setminus A$,
Ax.9 $\forall C.R \subseteq \exists C.R$,
Ax.10 $(t : \top) :- (t : A)$,
Ax.11 $(t : \{t\} :- (t : A)$,
Ax.12 $(t_1, t_2, \ldots, t_k) : \{(t_1, t_2, \ldots, t_k)\} :- (t_1, t_2, \ldots, t_k) : R$,
Ax.13 $\exists\{t_1\}.R :- (t_1, t_2, \ldots, t_k) : R$,
Ax.14 $(t_1, t_2, \ldots, t_k) : R :- \forall\{t1\}.R$,
Ax.15 $(t_2, t_3, \ldots, t_k, t_1) : R^{-1} :- (t_1, t_2, \ldots, t_k) : R$,
Ax.16 $(t_2, t_3, \ldots, t_k) : R^{-} :- (t_1, t_2, \ldots, t_k) : R$, for $k > 2$,
Ax.17 $t_2 : R^{-} :- (t_1, t_2) : R$,
Ax.18 $A^{-} = \bot$.

Ax.19 $Lower(A) \subseteq Upper(A)$,
Ax.20 $Boundary(A) \subseteq Upper(A)$,
Ax.21 $Lower(A) \subseteq A \subseteq Upper(A)$,
Ax.22 $Upper(Upper(A)) = Upper(A)$,
Ax.23 $Lower(Upper(A)) = Upper(A)$,
Ax.24 $Boundary(Upper(A)) = \bot$,
Ax.25 $Upper(Lower(A)) = Lower(A)$,
Ax.26 $Lower(Lower(A)) = Lower(A)$,
Ax.27 $Boundary(Lower(A)) = \bot$,
Ax.28 $Upper(Boundary(A)) = Boundary(A)$,
Ax.29 $Lower(Boundary(A)) = Boundary(A)$,
Ax.30 $Boundary(Boundary(A)) = \bot$.

Ax.31 $Lower(A \vee B) \supseteq Lower(A) \vee Lower(B)$,
Ax.32 $Lower(A \wedge B) = Lower(A) \wedge Lower(B)$,
Ax.33 $Upper(A \vee B) = Upper(A) \vee Upper(B)$,
Ax.34 $Upper(A \wedge B) \subseteq Upper(A) \wedge Upper(B)$.

The above axioms will be satisfied in a selected set-theoretical structure of rough sets:

$$\mathbf{F}(\Sigma(SN)) = \langle F, F^{-}, F^{+}, Fbn, \cup_C, \cap_C, \setminus_C, {}'^C, 0_C, 1_C, \in_C, \subseteq_C \rangle,$$

determined by the information system $\Sigma(SN) = \langle U, A, \{V_a\}_{a \in A} \rangle$, where U is a set of names of all the occurrences of the epistemic states in the semantic network SN under consideration, and $F = R(\Sigma(SN))$ is a set of all the rough sets determined in the information system $\Sigma(SN)$.

9.5.3 Semantics of the Language RPL

Let us determine interpretation $\mathbf{I} = (RPL, {}^I)$ of ***RPL*** language for which the interpretation function ^{I}: $RPL \to F$ (we write the values ${}^I(E)$ as E^I) satisfies the following conditions:

I1. any occurrences of concepts and roles are assigned elements of rough sets:

- $t^I \in_C U_C$,
- $(t_1,t_2,\ldots,t_k)^I \in_C \{\langle x_1,x_2,\ldots,x_k\rangle\}_C$, for $\langle x_1,x_2,\ldots,x_k\rangle \in U^k, t_1^I \in_C \{x_1\}_C$, $t_2^I \in_C \{x_2\}_C,\ldots,t_k^I \in_C \{x_k\}_C$,

I2. names of the concepts A, including the singletons $\{t\}$, are assigned the following rough sets:

- $\{t\}^I = \{x\}_C$, for some $x \in U$,
- $\{(t_1,t_2,\ldots,t_k)\}^I = \{\langle x_1,x_2,\ldots,x_k\rangle\}_C$, for $\langle x_1,x_2,\ldots,x_k\rangle \in U^k, t_1^I \in_C \{x_1\}_C$, $t_2^I \in_C \{x_2\}_C,\ldots,t_k^I \in_C \{x_k\}_C$,
- $A^I \in F$,

I3. names of the role R are assigned the rough sets $R^I \in F$,

I4. the modifiers $m \in \{Upper, Lower, Boundary\}$ assign some functions m^I: $F \to F$, from the set $\{F^-, F^+, Fbn\}$:

- $Lower^I = F^-$,
- $Upper^I = F^+$,
- $Boundary^I = Fbn$.

Semantics of the concepts of TBox language

For any names of the concepts A, B, the name of the role R and the modifier m

I5. $\top^I = 1_C$,
I6. $\bot^I = 0_C$,
I7. $(\neg A)^I = (A^I)'^C$,
I8. $(A \wedge B)^I = (A^I \cap_C B^I)$,
I9. $(A \vee B)^I = (A^I \cup_C B^I)$,
I10. $(A \setminus B)^I = (A^I \setminus_C B^I)$.

For $R^I \subseteq_C (U^k)_C$, where R is a k-argument role, there hold the following conditions of interpretation of concepts and roles:

I11. $(\exists A.R)^I = \bigcup_C \{\{t_1\}^I : ((t_1,t_2,\ldots,t_k)^I \in_C R^I \wedge t_1^I \in_C A^I)\}$,
I12. $(\forall A.R)^I = \bigcap_C \{\{t_1\}^I : ((t_1,t_2,\ldots,t_k)^I \in_C R^I \wedge t_1^I \in_C A^I)\}$,
I13. $(A.R)^I = \bigcup_C \{\{(t_1,t_2,\ldots,t_k)\}^I : ((t_1,t_2,\ldots,t_k)^I \in_C R^I \wedge t_1^I \in_C A^I)\}$,
I14. $(R^{-1})^I = \bigcup_C \{\{(t_1,t_2,\ldots,t_k)\}^I : (t_2,\ldots,t_k,t_1)^I \in_C R^I\}$,
I15. $(R^-)^I = \bigcup_C \{\{(t_1,t_2,\ldots,t_k)\}^I :$ if for some $t_{k+1}, (t_{k+1},t_1,t_2,\ldots,t_k)^I \in_C R^I\}$,

where the operations $\bigcup_C$, $\bigcap_C$ are generalized operations of addition and multiplication that are defined on subsets of the family F of rough sets. For an empty set the value of these generalized operations is an empty rough set 0_C.

I16. $(m(A))^I = m^I(A^I)$, for $m \in \{Upper, Lower, Boundary\}$,
I17. $(m(R))^I = m^I(R^I)$, for $m \in \{Upper, Lower, Boundary\}$.

Semantics of the assertion of ABox language

I18. For any occurences $t, t_1, t_2, \ldots, t_k$ concepts or roles, the name of the concept C, as well as the name of the role R

- $(t : C)^I$ iff $(t^I) \in_C C^I$,
- $((t_1, t_2, \ldots, t_k) : R)^I$ iff $(t_1, t_2, \ldots, t_k)^I \in_C R^I$.

Semantics of the RBox axioms

I19. For any A, B the names of concepts or roles and any assertions α, β

- $(A \subseteq B)^I$ iff $A^I \subseteq_C B^I$,
- $(A = B)^I$ iff $A^I = B^I$,
- $(\alpha :- \beta)^I$ iff if β^I then α^I.

Remark. The set of axioms is denumerable infinite. Some axioms, for given interpretations, are satisfied in some structure $\mathbf{F}(\Sigma(SN))$, but some axioms are not. The axioms distinguished in this paper are satisfied for any interpretation. This follows from the fact that interpretations of these axioms are formulas satisfied in any $\mathbf{F}(\Sigma(SN))$. For example, Axioms 19 – 24 are satisfied by Corollary 9.13, Axioms 1 - 6 – by Theorems 9.3-9.6, Satisfying other axioms follows easily from these theorems, Facts 9.11–9.12, and the definitions given in Section 9.4.

We call the ***rule of conceiving of axioms*** — the expression in the form of $\alpha_1, \alpha_2, \ldots, \alpha_k / \beta$, for any axioms $\alpha_1, \alpha_2, \ldots, \alpha_k, \beta$. The rule is ***adequate*** for the given interpretation function ^{I}, if — from the fact that there hold the interpretations $\alpha_1^I, \alpha_2^I, \ldots, \alpha_k^I$ — there follows the fact that the interpretation β^I holds.

Distinguished adequate rules of conceiving

Rule 1 $A \subseteq B, B \subseteq C \,/\, A \subseteq C$,
Rule 2 $A \subseteq B, B \subseteq A \,/\, A = B$,
Rule 3 $A \subseteq B \,/\, (t : B) :- (t : A)$,
Rule 4 $R_1 \subseteq R_2 \,/\, ((t_1, t_2, \ldots, t_k) : R_2) :- ((t_1, t_2, \ldots, t_k) : R_1)$,
Rule 5 $t_1 : A_1, t_2 : A_2, \ldots, t_k : A_k \,/\, (t_1, t_2, \ldots, t_k) : A_1 \times A_2 \times \ldots \times A_n$,
Rule 6 $(\{t_k\}.(\ldots(\{t_2\}.(\{t_1\}.R)^-)^-\ldots)^-)^- = \bot \,/\, (t_1, t_2, \ldots, t_k) : R$.

The adequate rules of conceiving do not serve the purpose of proving theorems — they merely determine the logical relations between axioms. If agents of the pragmatic system of representation of knowledge apply these rules, it means that they *conceive*, in an appropriate manner, interpretations of axioms in the structure of rough sets.

9.6 Prospects of Being Applied in Research into Artificial Intelligence

Rough pragmatic descriptive logic defines accurately how to interpret formulas describing vague (fuzzy) knowledge within the structure of rough sets, which is determined by a semantic network. This network is determined in some pragmatic information system, in which agents make use of only some distinguished data types. And this system is determined by a pragmatic system of knowledge representation. It is in this system that, for the distinguished data types, the vague knowledge is determined. Therefore, it is right to say that formulas of the *RPDL* refer to vague knowledge. Since, in practice, man most often communicates with other people, he passes to them data representing vague knowledge, thus, he conceives his utterances in approximation, approximating their sense. This means that people communicating with one another apply the *RPDL*.

Can one, in this communication process, replace man by an agent of an artificial intelligence? We cannot exclude that it is possible. *How to design such an agent?*

In compliance with the presented work, the procedure of designing the agent of an artificial intelligence can take on the following form:

Step 1. Establishing procedures to determine the pragmatic system of representation of knowledge.

Step 2. Determining the set of data types that will be available to the agent being designed.

Step 3. Checking whether an information system in Pawlak's sense can be determined for the available set of types of knowledge. If so, such a system has to be determined and we pass on to **Step 7**. If not, we go on to the next step.

Step 4. Checking whether a non-deterministic information system in Pawlak's sense can be determined for the available set of types of knowledge. If so, we pass on to **Step 7**; if not — to the next step.

Step 5. Checking whether a general information system can be determined for the available set of types of knowledge. If so, we determine this system and pass on to **Step 7**; if not — we go to the next step.

Step 6. We determine a non-deterministic information system for the available set of data types and pass on to the next step.

Step 7. We determine a semantic network corresponding to the determined information system and go on to the next step.

Step 8. For the semantic network defined in **Step 7** we formulate a language of descriptive logic and provide the syntax of **Tbox**, **ABox** and **RBox** blocks.

Step 9. For the semantic network defined in **Step 7** we determine the general information system.

Step 10. For the determined system we define the set-theoretical structure of rough sets.

Step 11. We establish procedures of interpretation of the language of descriptive logic in the structure of rough sets.

Step 12. We distinguish primitive axioms.

Step 13. We distinguish a set of rules conceiving the axioms.

Step 14. We determine the base of the knowledge $\langle Ab, Tb, Rb\rangle$, where the sets Ab, Tb, Rb are finite sets of expressions (descriptions of nodes of a semantic network), which are possible to be computer-processed, respectively: 1) assertions, about which the agent **knows**, 2) concepts, which the agent **has knowledge of** and also 3) axioms, which the agent **conceives by means of the rules of conceiving**. Since we say about a human being who is an agent availing himself of the base of knowledge that he knows something, has knowledge about something and conceives something, hence by replacing this human being by an AI agent, we can say the same about this very agent. We will call such an AI agent — a **pragmatic agent of an AI** and we will say that he is one who **knows something about** holding of assertions, **knowing** some concepts and **conceiving** axioms.

Step 15. The agent of an AI is **conscious**, if he has access to a semantic network which contains all the concepts relating to himself, known to people-agents, within the same system of representation of knowledge, and — moreover — this very agent makes use of all the roles between occurrences of the concepts.

Step 16. Implementation of the rough pragmatic descriptive logic can be performed in PROLOG language. In this implementation, one can realize translators of programming languages of WEB network, and then write a programme of the conscious agent of an AI, defined in **Step 14**, **Step 15**.

Due to the dynamic nature of language communication, it can turn out already at **Step 1** and **Step 2** that the relation of using data specimens is fuzzy, and the classification of data leading to determination of data types can not be carried out accurately. Then, realization of the above-mentioned procedure is not possible and should be preceded by application of methods of fuzzy sets theories [2,18,21,22]: fuzzyfication and deffuzzyfication, which lead to sharpening of knowledge necessary to determine the pragmatic system of representation of knowledge. Evolution methods and other methods of theories of learning machines can be applied as well. Perception-based methods [20] may be useful too.

Acknowledgements. We would like to thank an anonymous referee for his useful comments and helpful remarks.

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic. HandbookTheory, Implementation and Application. Cambridge University Press, Cambridge (2003)
2. Bobillo, F., Straccia, U.: fuzzyDL: An expressive fuzzy description logic reasoner. In: Proc. IEEE Int. Conference on Fuzzy Systems FUZZ-IEEE 2008 (IEEE World Congress on Computational Intelligence), pp. 923–930 (2008)
3. Bonikowski, Z.: Algebraic structures of rough sets. In: Ziarko, W. (ed.) Rough Sets, Fuzzy Sets and Knowledge Discovery, pp. 242–247. Springer (1994)
4. Bonikowski, Z.: Algebraic structures of rough sets in representative approximation spaces. Electr. Notes Theor. Comput. Sci. 82(4) (2003), doi:10.1016/S1571-0661(04)80705-9
5. Bonikowski, Z., Wybraniec-Skardowska, U.: Vagueness and Roughness. In: Peters, J.F., Skowron, A., Rybiński, H. (eds.) Transactions on Rough Sets IX. LNCS, vol. 5390, pp. 1–13. Springer, Heidelberg (2008)
6. Bonikowski, Z., Bryniarski, E., Wybraniec-Skardowska, U.: Extensions and intensions in the rough set theory. J. Inform. Sciences 107, 149–167 (1998)
7. Bryniarski, E.: A calculus of rough sets of the first order. Bull. Pol. Ac.: Math. 37, 109–136 (1989)
8. Bryniarski, E.: Formal conception of rough sets. Fund. Infor. 27(2-3), 103–108 (1996)
9. Demri, S.P., Orłowska, E.S.: Incomplete Information: Structure, Inference, Complexity. Springer, Heidelberg (2002)
10. Fanizzi, N., d'Amato, C., Esposito, F., Lukasiewicz, T.: Representing uncertain concepts in rough description logics via contextual indiscernibility relations. In: Bobillo, F., da Costa, P.C.G., d'Amato, C., et al. (eds.) Proc. 4th Int. Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2008). CEUR-WS, 423. CEUR-WS.org (2008), `http://ceur-ws.org/Vol-423/paper7.pdf` (cited May 13, 2011)
11. Keet, C.M.: Ontology Engineering with Rough Concepts and Instances. In: Cimiano, P., Pinto, H.S. (eds.) EKAW 2010. LNCS (LNAI), vol. 6317, pp. 503–513. Springer, Heidelberg (2010)
12. Kowalski, R.A.: Logic for Problem Solving. North Holland, New York (1979)
13. Orłowska, E., Pawlak, Z.: Representation of nondeterministic information. Theor. Computer Science 29, 27–39 (1984)
14. Pawlak, Z.: Information systems – Theoretical foundations. Inform. Systems 6, 205–218 (1981)
15. Pawlak, Z.: Rough sets. Intern. J. Comp. Inform. Sci. 11, 341–356 (1982)
16. Pawlak, Z.: Rough Sets. Theoretical Aspects of Reasoning about. Data. Kluwer Academic Publishers, Dordrecht (1991)
17. Pawlak, Z.: Rough set elements. In: Polkowski, L., Skowron, A. (eds.) Rough Sets in Knowledge Discovery 1. Methodology and Applications, pp. 10–30. Springer, Heidelberg (1998)
18. Polkowski, L.: Reasoning by Parts. An Outline of Rough Mereology. Warszawa (2011)
19. Skowron, A., Polkowski, L.: Rough mereology and analytical morphology. In: Orłowska, E. (ed.) Incomplete Information: Rough Set Analysis, pp. 399–437. Springer, Heidelberg (1996)
20. Skowron, A., Wasilewski, P.: An Introduction to Perception Based Computing. In: Kim, T.-h., Lee, Y.-h., Kang, B.-H., Ślęzak, D. (eds.) FGIT 2010. LNCS, vol. 6485, pp. 12–25. Springer, Heidelberg (2010)
21. Simou, N., Stoilos, G., Tzouvaras, V., Stamou, G., Kollias, S.: Storing and querying fuzzy knowledge in the semantic web. In: Bobillo, F., da Costa, P.C.G., d'Amato, C., et al. (eds.) Proc. 4th Int. Workshop on Uncertainty Reasoning for the Semantic Web URSW 2008, May 13. CEUR-WS, vol. 423, CEUR-WS.org (2008), `http://ceur-ws.org/Vol-423/paper8.pdf` (cited May 13, 2011)

22. Simou, N., Mailis, T., Stoilos, G., Stamou, S.: Optimization techniques for fuzzy description logics. In: Description Logics. Proc. 23rd Int. Workshop on Description Logics (DL 2010). CEUR-WS, vol. 573, CEUR-WS.org (2010), http://ceur-ws.org/Vol-573/paper_25.pdf (cited May 13, 2011)
23. Turing, A.M.: Computing machinery and intelligence. Mind 59(236), 433–460 (1950)
24. Wybraniec-Skardowska, U.: Meaning and interpretation. Studia Logica 85, 107–134 (2007)
25. Wybraniec-Skardowska, U.: On meta-knowledge and truth. In: Makinson, D., Malinowski, J., Wansing, H. (eds.) Towards Mathematical Philosophy. Trends in Logic, vol. 28, pp. 319–343. Springer, Heidelberg (2009)
26. Żakowski, W.: Approximations in the space (U, Π). Demonstratio Math. 16, 761–769 (1983)

Chapter 10
Application of Rough Set Theory to Sentiment Analysis of Microblog Data

Chien-Chung Chan and Kathy J. Liszka

Abstract. Microblogging has become a popular medium for broadcasting short text messages of 140 characters or less through social networks. There are millions of such posts shared each day, publically expressing sentiment over a variety of topics using popular Internet sites such as Twitter.com, Plurk.com, and identi.ca. Most of the existing works have focused on polarity sentiment analysis of these data by applying machine learning algorithms. In this chapter, we show that the rough set theory introduced by Pawlak provides an effective tool for deriving new perspectives of sentiment analysis from microblogging messages. More specifically, we introduce the use of rough set theory to formulate sentimental approximation spaces based on key words for assessing sentiment of microblogging messages. The sentimental approximation space provides contextual sentiment from the entire collection of messages, and it enables the evaluation of sentiment of different subjects, not in isolation, but in context. Sentiment, itself, is subjective. The degree of emotion that a word invokes in one person will be different than in another. It is for this reason that sentimental approximation space offers potentially more insightful information about a subject than simple polarity answers of positive or negative.

Keywords: Rough sets, approximation space, sentiment analysis, microblogging.

10.1 Introduction

There is a small, but growing body of research in opinion mining from microblog data. Twitter [1] is one of the free social microblogging services, and is currently the most popular one. Usage of Twitter is evolving, growing from "I'm feeling

Chien-Chung Chan · Kathy J. Liszka
Department of Computer Science, University of Akron
Akron, OH, 44325-4003, USA
e-mail: {chan, liszka}@uakron.edu

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 185–202.
springerlink.com

bored" type messages as the main content to more sophisticated usage in social networking, politics, and business marketing. This might imply that more recent data contains significantly different content. Among Twitter users, microblog posts are called “tweets". The underlying Short Message Service (SMS) protocol allowing up to 140 characters is a common element among the popular microblogging web sites [2]. In this chapter, we refer to all of these short posts as tweets, noting that the research in sentiment analysis discussed herein encompasses more than Twitter data.

The seminal work by Pang et al. shows that machine learning is a viable tool for sentiment analysis using a corpus of movie reviews [5]. They apply three standard machine learning algorithms: Naïve Bayes, maximum entropy (MaxEnt), and support vector machines (SVMs). Their positive and negative word lists were relatively small, from five to 11 in different experiments, but nonetheless, the results are good. More notable, they bring to light the difficulty of the task compared to topic based classification. The corpus, however, is not based on microblog data, which has a unique signature. In tweets, users speak succinctly, often in slang and using symbols. The movie reviews had no 140 character limit, and as such, contained much more content both in length and in traditional language usage.

Kim et al. give a compelling case for using Twitter lists for a corpus in sentiment analysis [3]. In this context, lists are groups of people who share a common interest such as music. They show that even though tweets are brief, they contain enough information to express identifiable characteristics, interests, and sentiments.

Recent work by Cheong et al. shows an approach to sentiment analysis of tweets using unsupervised self-organizing feature maps [4]. Once generated, they are used in visualizations to identify characteristics within select communities that share a common interest or trending topic. They start by identifying tweets using the API to search for a “term of interest". User demographics and metadata are collected with those tweets. They look for clusters of users based on attributes such as gender, age, number of tweets per day, geographical location, and so forth. Although they are classifying their case studies as sentiment analysis, they are looking at it from a different angle than we do.

The work in Go et al. is very similar to Pang in using the same three classifiers, but microblog data from Twitter is used as opposed to the lengthier text movie reviews [6]. The results are remarkably similar, showing promise that applying these tools for sentiment analysis cross the boundaries from longer text blocks to 140 character restricted tweets. Their research is polarity based, but excludes neutral sentiments from the corpora. Only positive and negative tweets are collected, mined through queries in the Twitter search utility using common emoticons. Once collected, the emoticons are removed from the tweets before training with the classifiers. Manually collected test data retains emoticons, if present.

Pak and Paroubek [7] collect data from Twitter, filter it and then classify tweets as positive or negative by the use of popular emoticons (smiley faces, sad faces, and variations). Neutral tweets are collected from newspaper accounts to round out the corpora. An analysis indicates the distribution of word frequencies in the collection is normal. They apply a Naïve Bayes classifier to test the posts. Their best results

are those experiments using bigrams. This is contrary to the findings of Pang, but may easily be explained by the very nature of the differing corpora. Movie reviews may contain more words and users may take more time to think about their post where tweeters tend to give lightening quick, brief snapshots of a thought sent from a cell phone or other small device. In fact, one very interesting observation that this chapter makes is the amount of slang used and frequent misspellings in tweets.

In the work by Read, sentiment analysis is performed on Usenet group data and movie reviews using the Naïve Bayes and SVM classifiers [8]. His corpus is created using emoticons to identify positive and negative texts. No neutral or objective texts are included in either the training or testing data sets. Read also looks at topic, domain, and temporal dependency classifications.

Several online sentiment tools are freely available. TweetFeel[1] claims they use a limited set of indicators. The site does not provide documentation on their methodology or algorithms. At the time of testing, it presented no results for sentiment on either the iPhone or Microsoft. It appears that they are geared specifically toward trending topics. Its usage is intended for purely entertainment value. Another site, Twitter Sentiment[2] gives 58% positive and 42% negative for the iPhone for the period July 2010 through November 2010. Sentiment for Microsoft over the same period was 51% positive, 49% negative. Sample positive, and negative tweets are presented with user input radio buttons that allows reclassification. This website was created by Alec Go, Richa Bhayani, and Lei Huang at Standford University and is based on their machine learning classifier work [6].

Other research exists on the social aspects of tweeters themselves. There is interest on what time of day people tend to tweet the most, what we tweet about, and what our social network links look like [9, 10].

The rough set theory was introduced by Pawlak in 1982 [14, 17]. It is a theory for reasoning about data represented as information systems [15, 16], which are generalizations of relational data bases. No implicit assumptions are required in applying rough sets, the starting point is an information system used to represent a finite collection of data characterized by a finite set of attributes or features. Our knowledge derived from information systems may be uncertain, since the data may be incomplete or imprecise. The rough set theory provides fundamental concepts and properties of approximations derived from information systems for studying and analyzing uncertain information, and for developing systems, and tools to deal with uncertainty. Rough set theory has led to many interesting applications and extensions for solving problems in research areas such as machine learning, intelligent systems, inductive reasoning, pattern recognition, mereology, knowledge discovery, decision analysis, and expert systems [18, 19].

More specifically, we believe it can be used as an effective tool to the analysis of microblog data represented as information systems. Microblog data are usually incomplete and imprecise, and collectively, they may contain a wide range of explicit and implicit topics or subjects. They are one of the major *crowd sources*

[1] http://www.tweetfeel.com

[2] http://twittersentiment.appspot.com

generated by a large number of people on a daily basis. In our previous review of existing research works, no tools have been developed to address the challenging issues such as how to measure the quality of information derived from these data, how to represent contextual information, how to evaluate the contextual impact of topics, and how to represent evolutionary dynamic changes of data. In rough set theory, contexts can be represented by approximation spaces derived from microblog information systems. Topics can be defined as subsets of objects in an information system, and their contextual quality can be measured by the concept of quality of approximations introduced by Pawlak [14, 16]. The impact of dynamic changes of data can be computed from incremental update of approximations [16, 26, 27, 28, 29], and incomplete information can be represented by incomplete information systems [30, 31].

The contribution of this chapter is a new methodology for determining sentiment of a topic in short microblog data. We present the application of rough set theory to the analysis of tweets using sentimental approximation space to provide a common basis for providing contextual sentiment with respect to different subjects. Dimensionality reduction is a problem in all information retrieval, and the large number of terms present in a 3-year span tweet corpus is no exception. This is addressed by demonstrating two different methods of grouping based on term frequency and by manually ranking sentiment words. Experiments are performed to derive general sentiment from tweets related to specific topics such as the iPhone, iPod, and Microsoft. We use the upper and lower approximation to measure accuracy and quality of results. These can be used for evaluating effectiveness of different grouping methods and further attribute reduction or refinement.

In Section 10.2, we formulate the problem and the use of sentimental approximation space to represent sentiment of tweets based on key words. We review the basic concept of the Bayesian rough set model [23] and describe how to determine lower and upper approximations in our application. Section 10.3 presents the processes involved in key word driven sentiment analysis. Here we describe the application of sentimental approximation space. Section 10.4 presents experiments with a specific subject word as an example. Finally, we present our concluding remarks in Section 10.5.

10.2 Problem Formulation

Tweet analysis can be viewed as an application of text categorization, which dates back to the work on probabilistic text classification by Maron [11]. The main task of text classification is to label texts with a predefined set of categories. Text categorization has been applied in other areas such as document indexing, document filtering, word sense disambiguation, etc. as surveyed in Sebastiani [12]. One of the central issues in text classification is how to represent the content of a text in order to facilitate an effective classification. From research in information retrieval systems, one of the most popular and successful methods is to represent a text by the

collection of terms that appear in it. The similarity between documents is defined by using the term frequency inverse document frequency (TFIDF) measure [13]. In this approach, the terms or features used to represent a text is determined by taking the union of all terms that appear in the collection of texts used to derive the classifier. This usually results in a large number of features. Therefore, dimensionality reduction is a related issue that needs to be addressed. Likewise, this is an issue common to tweet analysis when a feature set consists of all unique words or symbols, called *tokens*, of the set of collected tweets. In addition, each tweet is a short text message up to 140 characters collected from Twitter. Thus, the context information of a tweet is much more limited in comparison to a text document. The challenge here is how to find an effective way to represent a tweet's context to facilitate the analysis of tweets.

In this chapter, our main focus is to show the representation of contextual information using rough sets for sentiment analysis of tweets. The dimensionality reduction problem may be dealt with by applying existing well-known rough sets methods or other methods based on Latent Semantic Analysis (LSA) [24, 25]. We take a direct approach to dimensionality reduction by grouping tokens based on a set of key words. The contextual information of tweets is collectively represented as an approximation space in the rough set theory. In the following, we provide a brief review of related concepts in rough sets that will be used to formulate the problem considered in the chapter.

Let U be a set of tweets described by a finite set of tokens, which may be words or symbols; we will treat them equivalently unless mentioned otherwise. Let X be a specific subject of interest described by a finite set of words. The extension of X is defined as the set of tweets in U containing some of the subject words. Thus, we have $X \subseteq U$.

Let A be a finite set of attributes. An approximation space S is a pair $S = (U,A)$ where A defines an equivalence relation on U. A collection of tweets can be represented by an approximation space where A is the set of all unique words in the tweets. We define a *sentimental approximation space* as a pair $S_K = (U_K,K)$ where K is a finite set of sentimental key words and U_K is the set of tweets in U represented by key words in K. For simplicity, we will use $S = (U,K)$. Note that words used to define a subject X are not key words.

In the rough set approach, knowledge or information of a subject X that can be induced from an approximation space $S = (U,K)$ is represented by the lower and upper approximations of X, denoted by $\underline{K}X$ and $\overline{K}X$, in the approximation space S. Accuracy of approximations is defined as the ratio of the extensions of the lower approximation to the upper approximation. More precisely, the lower approximation of X by K in the approximation space S is defined as

$$\underline{K}X = \{e \in U : [e] \subseteq X\}, \tag{10.1}$$

and the upper approximation of X by K in S is defined as

$$\overline{K}X = \{e \in U : [e] \cap X \neq \emptyset\}. \tag{10.2}$$

where $[e]$, called *elementary sets*, denotes the equivalence class containing e. The accuracy $\alpha_K(X)$ of approximation of X in S is defined as $|\underline{K}X|/|\overline{K}X|$, where $|Y|$ denotes the cardinality of set Y. The quality of lower approximation is defined as $|\underline{K}X|/|U|$.

One probabilistic extension of rough set theory is called Variable Precision Rough Sets (VPRS) [20]. The basic idea is to use two thresholds, l and u, to control the precision of lower and upper approximations with respect to prior probability $P(X)$. The thresholds are constrained by $0 \leq l < P(X) < u \leq 1$. The u-lower approximation of a set X is called u-positive region of X and is defined as

$$POS_u(X) = \cup\{E : P(X|E) \geq u\}, \tag{10.3}$$

the l-negative region of X is defined as

$$NEG_l(X) = \cup\{E : P(X|E) \leq l\}, \text{ and} \tag{10.4}$$

the (l,u)-boundary region is defined as

$$BNR_{l,u}(X) = \cup\{E : l < P(X|E) < u\} \tag{10.5}$$

where E denotes an elementary set, which is the same as $[e]$ in previous definitions.

The classical rough sets introduced by Pawlak can be defined as VPRS with $l = 0$ and $u = 1$. In practical applications, it might be a challenge to decide the values for l and u. One way to alleviate this issue is to use the Bayesian Rough Set Model [23] where the positive, negative, and boundary regions are defined with respect to prior probability as follows. The positive region of X is defined as

$$POS^*(X) = \cup\{E : P(X|E) > P(X)\}, \tag{10.6}$$

the negative region of X is defined as

$$NEG^*(X) = \cup\{E : P(X|E) < P(X)\}, \tag{10.7}$$

and the boundary region of X is defined as

$$BND^*(X) = \cup\{E : P(X|E) = P(X)\} \tag{10.8}$$

where $P(X|E)$ is the conditional probability of X given an elementary set E.

The Bayesian rough set model will include an elementary set E into a lower approximation, if it provides an improvement of information related to X as measured by the conditional probability. In our experiments of sentiment analysis, the Bayesian rough set model is used to compute the lower and upper approximations of a subject in a sentimental approximation space.

In summary, the problem we consider in this chapter is how to evaluate polarity sentiment of different subjects from a collection of tweets on a common ground.

10.3 Key Word Driven Sentiment Analysis

The use of Internet slang must be addressed in any work involving microblog data. The original motivation for users to create these abbreviations was to reduce keystrokes. Texting on cell phones made this form of writing even more pervasive. In some cases, this has grown into social cultures with different dialects (ex., leet, netspeak, chatspeak) rather than a timesaving utility. In our case, we observe that the words or phrases used in tweets may include many of these abbreviated words such as abt (about), afaik (as far as I know), alol (actual laugh out loud), and so forth. This may cause missed matches with words or phrases that appear on the positive and negative word list. To evaluate the impact of irregular expressions in tweets to our strategy of tweet labeling, we have compiled our own list of 500 abbreviated words by personal observation combined with data from various web sites. We observed that the overlap is small between this list and the positive and negative word lists used in our experiments. Therefore, the impact is minimal, which is confirmed by our experiments on the iPhone-related tweets where the hit rate of positive words versus negative words remains quite similar with and without substitutions of abbreviated words or phrases. Thus, it does not affect the result of labeling tweets based on a sentiment word list. However, the excessive amount of abbreviated words in tweets may need to be dealt with in different types of tweet analysis.

We also note that some emoticons may be neutral, for example "(_/)" indicating bunny ears or "0w0" meaning representing either a cat's face or a face with wide eyes. We do not include these or use them as indicators of a neutral tweet. This is a possible addition to future work on tweet sentiment analysis since microblogging use and strategies are constantly evolving.

The following steps were applied for text mining Twitter data for our sentiment analysis.

10.3.1 Data Collection and Preprocessing

The corpus for testing our work with sentiment analysis came from a publicly available dataset, provided for research purposes under Creative Commons license from Choudhury [21]. This data set contains more than 10.1 million tweets collected from over 200,000 users in the time period from 2006 through 2009. Table 10.1 shows statistics for three sample subjects of interest used in our experiments. For subjects of interest, we use "iPhone", "iPod" and "Microsoft" as query terms to retrieve tweets from the raw data.

We collected a set of sentiment terms provided by Twitrratr [22]. The lists contain emoticons which we have removed, leaving 106 positive and 138 negative words. Examples from this list include *excellent*, *hilarious*, *thx* (positive), and *worried*, *creepy*, *repulsive* (negative).

The first step in preprocessing the corpus data was to remove emoticons. This is necessary so that they do not get confused with normal punctuation and symbols

Table 10.1 Subjects of interest

Query term	Number of related tweets	Example
iPhone	59,916	I can have a meaningful conversation with these lil faces! iPhone or die!
iPod	10,566	my iPod is taking forever to back up! :(
Microsoft	14,320	Dear Microsoft - when you do system updates, you download first, and *then* apply. Sigh.

included in hyperlinks. Once this is accomplished, we separate the tweets into individual tokens. Each token retains the original tweet identifier. It is common to find URLs in tweets, as people often share interesting links with friends. The next preprocessing task was to identify hyperlinks in the text and replace them with the tag URL. Symbols were also removed except for those that make up the set of emoticons listed in Table 10.1. These groupings will be discussed further in Section 10.3.2. Overall, we detected 426,540 sentimental emoticons present in the 10.1 million tweet corpus.

Punctuation tends to be puzzling in tweets. Perhaps this is due to the platform being tweeted on, a keyboard as opposed to a tiny device keypad. Tweeting while driving, although unadvisable and potentially illegal, may contribute to strange use of commas and periods, as well as typos. An unprintable control character infrequently appears, but human inspection of those affected tweets revealed that there was absolutely no case where there was an impact on the sentiment of a tweet. Therefore, we removed all control characters and punctuation from tweets that were not clearly being used for an emoticon. The punctuation symbols include (, ; . ? !).

At this point in the process, we have 138.5 million “clean" terms from the original corpus. *Stop words* are words commonly filtered out when doing any type of text processing. In our data, we removed prepositions and pronouns along with common, nondescript words such as *been*, *have*, *is*, *being*, and so forth. They can easily be removed without affecting the sentiment of the message as they do not convey any positive or negative connotation. After removing stop words, we are left with 86.8 million tokens. Finally, we consider the set of unique terms and tabulate their frequency. In this corpus, we have 3,373,178 unique terms.

Table 10.2 Set of emoticons

Positive sentiment	Negative sentiment
:) :-) :] :D :p ;) ;-) ;] ;D ;p =) =-) =D	:(:-(;(;-(=(=-(

10.3.2 Dimensionality Reduction

In the approximation space, $S = (U,A)$, we have $|A| = 3.3$ million unique tokens. We use simple key word grouping methods for dimensionality reduction to identify a subset of A. Our set of K sentimental key words consists of the set of positive and negative words and emoticons mentioned in Section 10.3.1. Thus, $|K| = 263$. We further reduce the number of attributes for our analysis by two methods; word frequency and manually. In Section 10.4, we compare the two methods of discretization.

10.3.2.1 Grouping by Equal Word Frequency

The first way of grouping is by taking the set of positive sentiment words and ranking them by frequency. Then we divide the words into groups of the same approximate frequency. The set of positive sentiment words appear in the tweet corpus 2.4 million times. These are divided into four groups of approximately 640K words. We label the groups P_i. Group 1 consists of two words, good, with a frequency of 325,448, and great, which has a frequency of 314,837. The sizes of the other groups are 4, 11, and 89 words, respectively. The set of negative sentiment words totals 0.5 million words. Sets are labeled N_i, each approximately 135,000 words in size. Positive emoticons total 0.4 million occurrences, and are divided into two groups, P_{e1} and P_{e2}, with a cutoff point of approximately 224,000 words. Similarly, negative emoticons total 53K words with a cutoff point of 36K words. These groups are labeled P_{n1} and P_{n2}. Thus, we have

$$\begin{aligned} K &= P_1P_2P_3P_4N_1N_2N_3N_4P_{e1}P_{e2}P_{n1}P_{n2}, \\ K^+ &= P_1P_2P_3P_4P_{e1}P_{e2}, \text{ and} \\ K^- &= N_1N_2N_3N_4P_{n1}P_{n2}. \end{aligned}$$

The five most frequently occurring positive sentiment words in this corpus are *good*, *great*, *thanks*, *love*, and *LOL*. The :) emoticon ranks highest. Among the negative sentiment words, the top five are *bad*, *crazy*, *hate*, *tired*, and *wrong*. The :(emoticon ranks highest.

10.3.2.2 Manual Grouping

An alternative to equal weight frequency based classification is to manually rank the strength of sentiment for the words. For example, *thoughtless* does not invoke as negative a sentiment as *despise*. The word *nice* does not hold as strong of a connotation as *awesome*.

We divided the positive emoticons into two groups based on perceived intensity of emotion. For example, a wide grin, :-D, is perceived to convey a stronger positive emotion than :-]. In the small set of negative emoticons, we did not see a clear distinction between the degrees of emotion and so we kept them as one group.

The lists of positive and negative sentiment terms were given to a group of university students. They were asked to rank the words in terms of intensity of emotion, they felt the words had a scale of 1 – 5. Their results were averaged and a cutoff point was manually selected, dividing the words into five groups. In this case, we have

$$\begin{aligned} K &= P_1P_2P_3P_4P_5N_1N_2N_3N_4N_5P_{e1}P_{e2}P_{n1}, \\ K^+ &= P_1P_2P_3P_4P_5P_{e1}P_{e2}, \text{ and} \\ K^- &= N_1N_2N_3N_4N_5P_{n1}. \end{aligned}$$

Note that the cutoffs for this method are independent of word frequency, nor are there an equal number of terms in each group. They were separated by average rank value, much like assigning a grade in a class. The purpose is not to advocate this method over the other, but rather to demonstrate an alternative way to express sentiment. Many reasonable methods could be derived to measure sentiment of a query term based on what one believes is positive versus negative.

10.3.3 Generation of Sentimental Approximation Space

The following is a description of how sentiments of a set U of tweets can be represented as a sentimental approximation space $S = (U,K)$ defined by a set K of key word groups, which are categorized into two types of groups, such as positive and negative, in polarity based sentiment analysis. The basic idea is to map each tweet of an arbitrary number of words into a tuple of $|K|$ integers. For each word w in a tweet, the frequency count of K_i is increased by one, if w is in the i-th group of K. So, if all the words of a tweet do not belong to any of the K groups, then it is represented as a tuple of all zeros.

Once we have the mapping of tweets as a collection of tuples, we can use the SQL group-by operation to generate a sentimental approximation space $S = (U,K)$ where each elementary set or block denoting tweets contain the same frequency count for each sentimental word group.

10.3.4 Subject Sentiment Analysis

Subjects are identified by subject words such as “Android tablet", “iPod", etc. For our purposes, a subject X is a set of tweets containing subject words of X. Typical

polarity sentiment analysis is to find out if the sentiment towards a subject is positive or negative from a corpus. The proposed approach can be applied to the analysis of multi-level or multi-value sentiment. However, for simplicity, we consider only polarity sentiment in this work.

The first step in determining the polarity sentiment of a subject X from a set U of tweets is to identify tweets belonging to X, and then divide the tweets of X into three subsets: $X_+, X_-, and X_=$ denoting positive, negative, and neutral subsets, respectively. In this work, they are defined by comparing the sum of frequency counts from the positive and negative key word groups. More precisely, given a tweet t, let P_+ be the sum of frequency counts of positive words in the tweet, and N_+ be the frequency sum of negative words in t. Then, tweet t is in the positive sentiment set X_+ if $P_+ > N_+$, else if $P_+ < N_+$, then it is in the negative sentiment set X_-; otherwise, it is in the neutral set $X_=$.

In the next step, we determine the sentiments of these subsets in an approximation space $S = (U, K)$. The results of our sentiment analysis of a subject X are represented by approximations of X in S such as the positive regions: $POS^*(X)$, $POS^*(X_+)$, $POS^*(X_-)$, and $POS^*(X_=)$. They denote the number of tweets that can be regarded certainly with the specific type of sentiment. In addition, the quality of lower approximation of the subject set X is computed as $|\underline{K}X|/|U|$, and the degree of approximation is measured by the accuracy of approximation defined as $|\underline{K}X|/|\overline{K}X|$ where $|U|$ denoting the number of tweets in the collection, and $|\underline{K}X|$, and $|\overline{K}X|$ denoting the number of tweets in the lower and upper approximations of X, respectively. In this study, the lower approximation is computed as the positive region $POS^*(X)$ of X based on definition (6) given in Section 10.2. Upper approximations are computed by using definition (2) of Section 10.2, i.e., an elementary set E is included in the upper approximation of X if the conditional probability of X given E is greater than zero, $P(X|E) > 0$.

10.4 Experimental Results

We present the experimental results of applying the proposed approach to the collection of tweets described in Section 10.3.1. The preprocessing of tweets into tokens was performed using programs written in Java. Subsequently, the tweet ids and their associated tokens were uploaded to Microsoft SQL Server 2005. The generation of sentimental approximation spaces and the computing of lower and upper approximations of subjects were implemented using SQL scripts. Similarly, we used SQL scripts to compute the sentiments of three subjects, “iPhone", “Microsoft", and “iPod", together with the qualities and accuracies of approximations. Table 10.3 shows the sentiment analysis of the subject X = “iPhone". The approximation space is generated by using the 4-equal-frequency grouping strategy described in Section 10.3.2, i.e., the set of keywords are divided based on equal frequency counts into four groups of positive and negative sentiment words and two groups of positive and negative emoticons. Thus, we have 12 attributes in this sentimental

approximation space. Among the set U of 10,157,205 tweets, there are 59,916 tweets containing the token “iPhone" as indicated in the first row of the column X. The first row *BaseLine* denotes the number of tweets divided into subsets by comparing the frequency counts of the positive and negative words as defined in Section 10.3.2.2. The *Non-Senti* column denotes the number of non-sentimental tweets, i.e., those whose values of their attributes are all zero. The *Senti* column denotes the number of sentimental tweets, which are further divided into positive sentiment tweets X_+, negative sentiment tweets X_-, and neutral tweets $X_=$. The $P(X)$ row shows the prior probability of each sentiment set. It is used as a threshold to determine if an elementary set should be included in the positive region set $POS^*(X)$ or not. The Upper Approximation row shows the number of tweets in the upper approximations. The accuracy and quality are shown in the corresponding rows.

Table 10.3 Sentiment analysis of iPhone using four-equal-frequency grouping strategy

	X_+	X_-	$X_=$	*Senti*	*Non* $-$ *Senti*	X
Baseline	9842	2045	420	12307	47609	59916
$POS^*(X)$	469938	188864	22964	681766	7706016	8387782
Upper Approx.	1976991	377902	82765	2437658	7706016	10143674
$P(X)$ Threshold	0.0059	0.0059	0.0059	0.0059	0.0059	0.0059
Accuracy	0.2377	0.4998	0.2775	0.2797	1.0000	0.8269
Quality	0.6893	0.2770	0.0337	0.0813	0.9187	0.8258

Table 10.4 shows the experimental results of the “iPhone" subject based on manual grouping, and Table 10.5 shows the results based on the 5-equal frequency grouping method.

Experimental results on the “Microsoft" subject based on the three different methods are shown in Tables 10.6–10.8.

Tables 10.9–10.11 show the experimental results on the “iPod" subject based on the three different methods.

Table 10.12 shows the comparative performance of the three different grouping methods applied to the three different subjects. From the table, the 4-equal-frequency method performs best in the “iPhone" subject, the manual grouping method is slightly better than the other two methods in the Microsoft data set, and the 5-equal-frequency method performs best in the iPod data set.

Discussion

The first distinguishing information derived from the experiments is that in all cases, most tweets in the subject areas are categorized as non-sentimental. This is not a

Table 10.4 Results of iPhone based on manual grouping

	X_+	X_-	$X_=$	$Senti$	$Non-Senti$	X
Baseline	9842	2045	420	12307	47609	59916
$POS^*(X)$	215685	68393	22373	306451	7706016	8012467
Upper Approx.	1974224	377153	82304	2433681	7706016	10139697
$P(X)$ Threshold	0.0059	0.0059	0.0059	0.0059	0.0059	0.0059
Accuracy	0.1093	0.1813	0.2718	0.1259	1.0000	0.7902
Quality	0.7038	0.2232	0.0730	0.0382	0.9618	0.7888

Table 10.5 Results of iPhone based on 5-equal-frequency grouping

	X_+	X_-	$X_=$	$Senti$	$Non-Senti$	X
Baseline	9842	2045	420	12307	47609	59916
$POS^*(X)$	388330	156643	24995	569968	7706016	8275984
Upper Approx.	197207	376947	82755	2431778	7706016	10137794
$P(X)$ Threshold	0.0059	0.0059	0.0059	0.0059	0.0059	0.0059
Accuracy	0.1969	0.4156	0.3020	0.2344	1.0000	0.8163
Quality	0.6813	0.2748	0.0439	0.0689	0.9311	0.8148

Table 10.6 Microsoft results based on 4-equal-frequency grouping

	X_+	X_-	$X_=$	$Senti$	$Non-Senti$	X
Baseline	1413	485	83	1981	12339	14320
$POS^*(X)$	3041	94302	26827	124170	7706016	7830186
Upper Approx.	1951435	374688	81915	2408038	7706016	10114054
$P(X)$ Threshold	0.0014	0.0014	0.0014	0.0014	0.0014	0.0014
Accuracy	0.0016	0.2517	0.3275	0.0516	1.0000	0.7742
Quality	0.0245	0.7595	0.2161	0.0159	0.9841	0.7709

Table 10.7 Microsoft results based on manual grouping

	X_+	X_-	$X_=$	$Senti$	$Non-Senti$	X
BaseLine	1413	485	83	1981	12339	14320
$POS^*(X)$	21531	87468	18735	127734	7706016	7833750
Upper Approx.	1945158	373058	71781	2389997	7706016	10096013
$P(X)$ Threshold	0.0014	0.0014	0.0014	0.0014	0.0014	0.0014
Accuracy	0.0111	0.2345	0.2610	0.0534	1.0000	0.7759
Quality	0.1686	0.6848	0.1467	0.0163	0.9837	0.7713

Table 10.8 Microsoft results based on 5-equal-frequency grouping

	X_+	X_-	$X_=$	$Senti$	$Non-Senti$	X
BaseLine	1413	485	83	1981	12339	14320
$POS^*(X)$	22324	77667	19349	119340	7706016	7825356
Upper Approx.	1938332	371201	77526	2387059	7706016	10093075
$P(X)$ Threshold	0.0014	0.0014	0.0014	0.0014	0.0014	0.0014
Accuracy	0.0115	0.2092	0.2496	0.0500	1.0000	0.7753
Quality	0.1871	0.6508	0.1621	0.0153	0.9847	0.7704

Table 10.9 iPod results based on 4-equal-frequency grouping

	X_+	X_-	$X_=$	$Senti$	$Non-Senti$	X
BaseLine	1703	366	81	2150	8416	10566
$POS^*(X)$	120827	104639	46722	272188	7706016	7978204
Upper Approx.	1959659	371772	81524	2412955	7706016	10118971
$P(X)$ Threshold	0.00104	0.00104	0.00104	0.00104	0.00104	0.00104
Accuracy	0.0617	0.2815	0.5731	0.1128	1.0000	0.7884
Quality	0.4439	0.3844	0.1717	0.0341	0.9659	0.7855

Table 10.10 iPod results based on manual grouping

	X_+	X_-	$X_=$	$Senti$	$Non-Senti$	X
BaseLine	1703	366	81	2150	8416	10566
$POS^*(X)$	41851	3833	10971	56655	7706016	8032788
Upper Approx.	1956858	370855	78168	2405881	7706016	10111897
$P(X)$ Threshold	0.00104	0.00104	0.00104	0.00104	0.00104	0.00104
Accuracy	0.0214	0.0103	0.1404	0.0235	1.0000	0.7944
Quality	0.7387	0.0677	0.1936	0.0071	0.9593	0.7908

Table 10.11 iPod results based on 5-equal-frequency grouping

	X_+	X_-	$X_=$	$Senti$	$Non-Senti$	X
BaseLine	1703	366	81	2150	8416	10566
$POS^*(X)$	452889	81770	38658	573317	7706016	8279333
Upper Approx.	1951935	370154	74828	2396917	7706016	10102933
$P(X)$ Threshold	0.00104	0.00104	0.00104	0.00104	0.00104	0.00104
Accuracy	0.2320	0.2209	0.5166	0.2392	1.0000	0.8195
Quality	0.7899	0.1426	0.0674	0.0692	0.9308	0.8151

Table 10.12 Performance of the three different grouping methods

	Accuracy (%)			Quality of Lower Approx. (%)		
	iPhone	**Microsoft**	**iPod**	**iPhone**	**Microsoft**	**iPod**
4-Equal-Freq.	82.58	77.42	78.84	82.58	77.09	78.55
Manual	79.02	77.59	79.44	78.88	77.13	79.08
5-Equal-Freq.	81.63	77.04	81.95	81.48	77.02	81.51

surprise; it merely shows those tweets do not contain any emotional expression as defined by our word list.

Accuracy is really a measure relative to the size of the base; therefore, it is related to the distribution. It is an indicator of how well a sentiment is approximated. It can be used as a weighted summary for the polarity of the sentiment. It is also a measure of the degree of approximation to the subject. The higher the accuracy, the better the approximation. How to apply it is a subject of further study and future work.

All three experiments reveal that the different grouping methods yield a similar performance. We find this to be counter-intuitive. Further study should be done with the manual grouping of key words. We conjecture that the subject and number of groups may have an impact. We also applied the Twitrratr key word set, but feel that a more scientific and systematic study should be done to identify a set of words that accurately identifies tweet sentiment language. This is a complex issue and a research topic in itself.

The Microsoft results shown in Tables 10.6-10.8 reveal an interesting result. Looking at the baseline distribution, one might conclude that tweeters view Microsoft positively, overall. However, from a contextual sentiment viewpoint, the lower approximation clearly shows that overall sentiment for Microsoft is negative. This is also reflected by observing the quality of X_+ in Microsoft is low whereas the X_- quality value is much higher. This indicates the value of using sentimental approximation space for opinion mining tweets.

10.5 Conclusions

We have introduced a new method based on rough sets for assessing sentiment of microblog messages using key words. The sentimental approximation space provides contextual sentiment from the entire collection of tweets. It enables the evaluation of sentiment of different subjects, not in isolation, but in context. The accuracy and quality measures can be used for evaluating effectiveness of different grouping methods and further attribute reduction or refinement. Furthermore, we demonstrate different methods for dimensionality reduction. Frequency based grouping appears to outperform manually ranking sentiment words, but the rankings may be improved on with more study. Sentiment is subjective. The degree of emotion that a word invokes in one person will be different than in another. It is for this reason that sentimental approximation space offers potentially more insightful information about a subject than simple polarity answers of positive or negative. Much work remains to be done in the exploding field of microblogging text mining and analysis.

References

1. Twitter hits 50 million tweets per day, `http://mashable.com/2010/02/22/twitter-50-million-tweets/`

2. Sagolla, D.: 140 Characters: A Style Guide for the Short Form. Wiley (2009) ISBN 0470556137, 978-0470556139
3. Kim, D., Jo, Y., Moon, I.-C., Oh, A.: Analysis of Twitter Lists as a Potential Source for Discovering Latent Characteristics of Users. In: Workshop on Microblogging at the ACM Conference on Human Factors in Computer Systems (CHI 2010) (2010)
4. Cheong, M., Lee, V.: A study on detecting patterns in twitter intra-topic user and message clustering. In: 2010 International Conference on Pattern Recognition, pp. 3125–3128 (2010)
5. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment classification using machine learning techniques. In: Proc. of the Conf. on Empirical Methods in Natural Language Processing (EMNLP), pp. 79–86 (July 2002)
6. Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision. In: Proc. of the 4th International Conf. on Computer and Information Technology (CIT 2004), pp. 1147–1152 (2004)
7. Pak, A., Paroubek, P.: Twitter as a corpus for sentiment analysis and opinion mining. In: Proc. of the Seventh Conf. on International Language Resources and Evaluation (LREC 2010) (May 2010)
8. Read, J.: Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In: Proc. of ACL 2005, 43rd Meeting of the Association for Computational Linguistics, pp. 43–48 (2005)
9. Java, A., Song, X., Fin, T., Tseng, B.: Why We Twitter: Understanding microblogging usage and communities. In: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis, pp. 56–65 (2007)
10. Krishnamurthy, B., Gill, P., Arlitt, M.: A few chirps about Twitter. In: Proceedings of the 1st Workshop on Online Social Networks (WOSN 2008), pp. 19–24 (2008)
11. Maron, M.: Automatic indexing: An experimental inquiry. J. Assoc. Comput. Mach. 8(3), 404–417 (1961)
12. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34(1), 1–47 (2002)
13. Salton, G., Wong, A., Yang, C.: A vector space model for automatic indexing. Communication of ACM 18(11), 613–620 (1975)
14. Pawlak, Z.: Rough sets: Basic notions. International Journal of Computer and Information Science 11(15), 344–356 (1982)
15. Pawlak, Z.: Rough Sets and Decision Tables. In: Skowron, A. (ed.) SCT 1984. LNCS, vol. 208, pp. 186–196. Springer, Heidelberg (1985)
16. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishing (1991)
17. Pawlak, Z., Grzymała-Busse, J., Słowiński, R., Ziarko, W.: Rough sets. Communication of ACM 38(11), 89–95 (1995)
18. Pawlak, Z., Skowron, A.: Rudiments of rough sets. Information Sciences 177(1), 3–27 (2007)
19. Pawlak, Z., Skowron, A.: Rough sets: Some extensions. Information Sciences 177(1), 28–40 (2007)
20. Ziarko, W.: Variable precision rough sets model. Journal of Computer and System Sciences 46(1), 39–59 (1993)
21. Choudhury, M.D., Lin, Y.-R., Sundaram, H., Candan, K.S., Xie, L., Kelliher, A.: How does the sampling strategy impact the discovery of information diffusion in social media? In: Proc. of the 4th Int'l AAAI Conference on Weblogs and Social Media, George Washington University, Washington, DC, May 23-26 (2010)
22. Twitter compatible positive and negative word list, `http://www.twitrratr.com`
23. Ślęzak, D., Ziarko, W.: The investigation of the Bayesian rough set model. International Journal of Approximate Reasoning 40, 81–91 (2005)
24. Deerwester, S., et al.: Improving information retrieval with latent semantic indexing. In: Proceedings of the 51st Annual Meeting of the American Society for Information Science, vol. 25, pp. 36–40 (1988)

25. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society for Information Science 41, 391–407 (1990)
26. Chan, C.-C.: Incremental learning of production rules from examples under uncertainty: A rough set approach. Int. J. of Software Engineering and Knowledge Engineering 1(4), 439–461 (1991)
27. Chan, C.-C.: A rough set approach to attribute generalization in data mining. Information Sciences 107(1-4), 169–176 (1998)
28. Shan, N., Ziarko, W.: Data-based acquisition and incremental modification of classification rules. Computational Intelligence 11(2), 357–370 (1995)
29. Li, T., Ruan, D., Geert, W., Song, J., Xu, Y.: A rough sets based characteristic relation approach for dynamic attribute generalization in data mining. Knowledge-Based Systems 20(5), 485–494 (2007)
30. Kryszkiewicz, M.: Rough set approach to incomplete information systems. Information Sciences 112, 39–49 (1998)
31. Grzymała-Busse, J.W.: Characteristic Relations for Incomplete Data: A Generalization of the Indiscernibility Relation. In: Peters, J.F., Skowron, A. (eds.) Transactions on Rough Sets IV. LNCS, vol. 3700, pp. 58–68. Springer, Heidelberg (2005)

Chapter 11
Relationships for Cost and Uncertainty of Decision Trees

Igor Chikalov, Shahid Hussain, and Mikhail Moshkov

Abstract. This chapter is devoted to the design of new tools for the study of decision trees. These tools are based on dynamic programming approach and need the consideration of subtables of the initial decision table. So this approach is applicable only to relatively small decision tables. The considered tools allow us to compute:

1. The minimum cost of an approximate decision tree for a given uncertainty value and a cost function.
2. The minimum number of nodes in an exact decision tree whose depth is at most a given value.

For the first tool we considered various cost functions such as: depth and average depth of a decision tree and number of nodes (and number of terminal and non-terminal nodes) of a decision tree. The uncertainty of a decision table is equal to the number of unordered pairs of rows with different decisions. The uncertainty of approximate decision tree is equal to the maximum uncertainty of a subtable corresponding to a terminal node of the tree. In addition to the algorithms for such tools we also present experimental results applied to various datasets acquired from UCI ML Repository [4].

Keywords: Decision tree, cost functions, uncertainty measures, dynamic programming.

11.1 Introduction

Decision trees are widely used as *predictors*, as a way of *knowledge representation* and as *algorithms* for problem solving. Each such use has a different optimization

Igor Chikalov · Shahid Hussain · Mikhail Moshkov
Mathematical and Computer Sciences & Engineering Division
King Abdullah University of Science and Technology
Thuwal 23955-6990, Saudi Arabia.
e-mail: {igor.chikalov, shahid.hussain}@kaust.edu.sa
mikhail.moshkov@kaust.edu.sa

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 203–222.
springerlink.com

objective. That is, we need to minimize the number of misclassifications in order to achieve more accurate decision trees (from the perspective of *prediction*). To have more understandable decision trees we need to minimize the number of nodes in a decision tree (*knowledge representation*). And decision trees, when used as *algorithms*, need to be shallow, i.e., we need to minimize either the depth or average depth (or in some cases both) of a decision tree in order to reduce algorithm complexity.

Exact decision trees can be overlearned (overfitting), so we need to consider not only exact but also approximate decision trees. In this chapter, we concentrate on consideration of five different cost functions for decision tree optimization: depth, average depth, number of nodes, number of nonterminal nodes, and number of terminal nodes. We consider an uncertainty measure $R(T)$ that is equal to the number of unordered pairs of rows in the decision tree T labeled with different decisions. The uncertainty measure $R(T)$ for the decision tree T allows us to define the notion of α-decision trees. For a fixed non-negative integer α, the uncertainty of subtable corresponding to each terminal node of α-decision tree is at most α.

The first aim of this chapter is to study the relationships between the cost and the uncertainty of decision trees. That is, for a given cost function ψ, decision table T, and a nonnegative integer α, we should find the minimum cost of α-decision tree for T relative to ψ. The second aim is to study the relationships between number of nodes and depth of an exact decision tree (and vice versa). That is, for a given decision table T, we should find the minimum number of nodes in an exact decision tree with depth at most d (and vice versa).

To this end we have designed and implemented the two algorithms (one for each task) based on dynamic programming approach and applied it to different datasets from UCI ML Repository [4] and integrated them into our software system [1, 3]. These tools can be useful for the specialists in the rough set theory [5]. A similar tool is considered in [2] which allows us to study the relationships between decision tree depth and number of misclassifications (and vice versa).

This chapter is divided into five sections. Section 11.2 explains basic notions. Section 11.3 presents the algorithm to compute the relationships between cost and uncertainty of decision trees, this section ends with the experimental results for the algorithm. Section 11.4 presents the algorithm to compute the relationships between number of nodes and depth (and vice versa) of decision trees. This section also ends with the respective experimental results. References and an appendix for transformation of functions follow the conclusion in Sect. 11.5.

11.2 Basic Notions

In the following section we define the main notions related with the study of decision trees and tables and different cost functions for the decision tree construction.

11.2.1 Decision Tables and Decision Trees

In this chapter, we consider only decision tables with discrete attributes. These tables do not contain missing values and equal rows. Consider a *decision table* T depicted in Fig. 11.1.

f_1	$\cdots$	f_m	d
b_{11}	$\cdots$	b_{1m}	c_1
	$\vdots$		$\vdots$
b_{N1}	$\cdots$	b_{Nm}	c_N

Fig. 11.1 Decision table

Here $f_1,\ldots,f_m$ are the conditional attributes; $c_1,\ldots,c_N$ are nonnegative integers which can be interpreted as the decisions (values of the decision attribute d); b_{ij} are nonnegative integers which are interpreted as values of conditional attributes (we assume that the rows $(b_{11},\ldots,b_{1m}),\ldots,(b_{N1},\ldots,b_{Nm})$ are pairwise different). We denote by $E(T)$ the set of attributes (columns of the table T), each of which contains different values. For $f_i \in E(T)$, let $E(T,f_i)$ be the set of values from the column f_i.

Let $f_{i_1},\ldots,f_{i_t} \in \{f_1,\ldots,f_m\}$ and $a_1,\ldots,a_t$ be nonnegative integers. We denote by $T(f_{i_1},a_1)\ldots(f_{i_t},a_t)$ the subtable of the table T, which consists of such and only such rows of T that at the intersection with columns $f_{i_1},\ldots,f_{i_t}$ have numbers $a_1,\ldots,a_t$, respectively. Such nonempty tables (including the table T) will be called *separable subtables* of the table T.

For a subtable Θ of the table T we will denote by $R(\Theta)$ the number of unordered pairs of rows that are labeled with different decisions. Later we will interpret the value $R(\Theta)$ as the *uncertainty* of the table Θ. A minimum decision value which is attached to the maximum number of rows in a nonempty subtable Θ will be called the *most common decision* for Θ. The subtable Θ will be called *degenerate* if $R(\Theta)=0$.

A *decision tree* Γ *over* the table T is a finite directed tree with a root in which each terminal node is labeled with a decision. Each nonterminal node is labeled with a conditional attribute, and for each nonterminal node, the outgoing edges are labeled with pairwise different nonnegative integers. Let v be an arbitrary node of Γ. We now define a subtable $T(v)$ of the table T. If v is the root then $T(v)=T$. Let v be a node of Γ that is not the root, nodes in the path from the root to v be labeled with attributes $f_{i_1},\ldots,f_{i_t}$, and edges in this path be labeled with values $a_1,\ldots,a_t$, respectively. Then $T(v)=T(f_{i_1},a_1)\ldots(f_{i_t},a_t)$.

Let α be a nonnegative integer. We will say that Γ is an α*-decision tree for* T if any node v of Γ satisfies the following conditions:

- If $R(T(v)) \le \alpha$ then v is a terminal node labeled with the most common decision for $T(v)$.

- Otherwise, v is labeled with an attribute $f_i \in E(T(v))$ and, if $E(T(v), f_i) = \{a_1, \ldots, a_t\}$, then t edges leave node v, and these edges are labeled with $a_1, \ldots, a_t$ respectively.

The notion of a (exact) decision tree for T coincides with the notion of a 0-decision tree for T.

11.2.2 Cost Functions

We will consider cost functions which are given in the following way: values of considered cost function ψ, which are nonnegative numbers, are defined by induction on pairs (T, Γ), where T is a decision table and Γ is an α-decision tree for T. Let Γ be an α-decision tree represented in Fig. 11.2. Then $\psi(T, \Gamma) = \psi^0$, where ψ^0 is a nonnegative number. Let Γ be an α-decision tree depicted in Fig. 11.3. Then

$$\psi(T, \Gamma) = F(N(T), \psi(T(f_i, a_1), \Gamma_1), \ldots, \psi(T(f_i, a_t), \Gamma_t)).$$

Here $N(T)$ is the number of rows in the table T, and $F(n, \psi_1, \psi_2, \ldots)$ is an operator which transforms the considered tuple of nonnegative numbers into a nonnegative number. Note that the number of variables $\psi_1, \psi_2, \ldots$ is not bounded from above.

The considered cost function will be called *monotone* if for any natural t, any nonnegative numbers $a, c_1, \ldots, c_t, d_1, \ldots, d_t$ and the inequalities $c_1 \leq d_1, \ldots, c_t \leq d_t$ the inequality $F(a, c_1, \ldots, c_t) \leq F(a, d_1, \ldots, d_t)$ follows. We will say that ψ is *bounded from below* if $\psi(T, \Gamma) \geq \psi^0$ for any decision table T and any α-decision tree Γ for T.

Now, we take a closer view of some monotone cost functions, which are bounded from below.

Number of nodes: $\psi(T, \Gamma)$ is the number of nodes in α-decision tree Γ. For this cost function $\psi^0 = 1$ and

$$F(n, \psi_1, \psi_2, \ldots, \psi_t) = 1 + \sum_{i=1}^{t} \psi_i.$$

Number of nonterminal nodes: $\psi(T, \Gamma)$ is the number of nonterminal nodes in α-decision tree Γ. For this cost function $\psi^0 = 0$ and

$$F(n, \psi_1, \psi_2, \ldots, \psi_t) = 1 + \sum_{i=1}^{t} \psi_i.$$

Number of terminal nodes: $\psi(T, \Gamma)$ is the number of terminal nodes in α-decision tree Γ. For this cost function $\psi^0 = 1$ and

$$F(n, \psi_1, \psi_2, \ldots, \psi_t) = \sum_{i=1}^{t} \psi_i.$$

Depth: $\psi(T,\Gamma)$ is the maximum length of a path from the root to a terminal node of Γ. For this cost function $\psi^0 = 0$ and

$$F(n,\psi_1,\psi_2,\ldots,\psi_t) = 1 + \max\{\psi_1,\ldots,\psi_t\}.$$

Total path length: for an arbitrary row $\bar{\delta}$ of the table T we denote by $l(\bar{\delta})$ the length of the path from the root to a terminal node v of Γ such that $\bar{\delta}$ is in $T(v)$. Then $\psi(T,\Gamma) = \sum_{\bar{\delta}} l(\bar{\delta})$, where we take the sum on all rows $\bar{\delta}$ of the table T. For this cost function $\psi^0 = 0$ and

$$F(n,\psi_1,\psi_2,\ldots,\psi_t) = n + \sum_{i=1}^{t} \psi_i.$$

Note that the *average depth* of Γ is equal to the total path length divided by $N(T)$.

11.2.3 Constructing the Graph $\Delta(T)$

We consider an algorithm for construction of a directed acyclic graph (DAG) $\Delta(T)$ for a decision table T. Nodes of this graph are some separable subtables of the table T. During each step we process one node and mark it with the symbol *. We start with the graph that consists of one node T and finish when all nodes of the graph are processed.

Let the algorithm have already performed p steps. We now describe the step number $(p+1)$. If all nodes are processed then the work of the algorithm is finished, and the resulted graph is $\Delta(T)$. Otherwise, choose a node (table) Θ that has not been processed yet. Let b be the most common decision for Θ. If $R(\Theta) = 0$, label the considered node with b, mark it with symbol * and proceed to the step number $(p+2)$. Let $R(\Theta) > 0$. For each $f_i \in E(\Theta)$ draw a bundle of edges from the node Θ (this bundle of edges will be called *f_i-bundle*). Let $E(\Theta,f_i) = \{a_1,\ldots,a_t\}$. Then draw t edges from Θ and label these edges with pairs $(f_i,a_1),\ldots,(f_i,a_t)$ respectively. These edges enter into the nodes $\Theta(f_i,a_1),\ldots,\Theta(f_i,a_t)$. If some of the nodes from $\Theta(f_i,a_1),\ldots,\Theta(f_i,a_t)$ are not present in the graph then add these nodes to the graph. Mark the node Θ with the symbol * and proceed to the step number $(p+2)$.

Now for each node Θ of the graph $\Delta(T)$ we describe the set of (exact) decision trees (0-decision trees) corresponding to it. It is clear that $\Delta(T)$ is a directed acyclic graph. A node of such graph will be called *terminal* if there are no edges leaving this node. We will move from terminal nodes, which are labeled with numbers, to the node T. Let Θ be a node, which is labeled with a number b. Then the only trivial decision tree depicted in Fig. 11.2 corresponds to the considered node.

Let Θ be a node (table), for which $R(\Theta) > 0$. There is a number of bundles of edges starting in Θ. We consider an arbitrary bundle and describe the set of decision trees corresponding to this bundle. Let the considered bundle be an f_i-bundle where

Fig. 11.2 Trivial α-decision tree

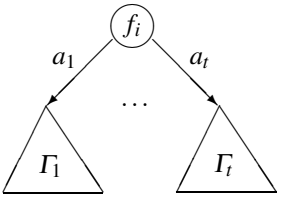


Fig. 11.3 Aggregated α-decision tree

$f_i \in E(\Theta)$ and $E(\Theta, f_i) = \{a_1, \ldots, a_t\}$. Let $\Gamma_1, \ldots, \Gamma_t$ be decision trees from sets corresponding to the nodes $\Theta(f_i, a_1), \ldots, \Theta(f_i, a_t)$. Then the decision tree depicted in Fig. 11.3 belongs to the set of decision trees, which correspond to this bundle. All such decision trees belong to the considered set, and this set does not contain any other decision trees. Then the set of decision trees corresponding to the node Θ coincides with the union of sets of decision trees corresponding to bundles starting in Θ. We denote by $D(\Theta)$ the set of decision trees corresponding to the node Θ in the graph $\Delta(T)$.

The following proposition shows that the graph $\Delta(T)$ can represent all decision trees for the table T.

Proposition 11.1. *Let T be a decision table and Θ a node in the graph $\Delta(T)$. Then the set $D(\Theta)$ coincides with the set of all decision trees for the table Θ.*

Proof. We prove this proposition by induction on nodes in the graph $\Delta(T)$. For each terminal node Θ, only one decision tree exists as depicted in Fig. 11.2, and the set $D(T)$ contains only this tree. Let Θ be a non-terminal node and the statement of proposition hold for all its descendants.

Consider an arbitrary decision tree $\Gamma \in D(\Theta)$. Obviously, Γ contains more than one node. Let the root of Γ be labeled with an attribute f_i and the edges leaving root be labeled with the numbers $a_1, \ldots, a_t$. For $j = 1, \ldots, t$, denote by Γ_j the decision tree connected to the root with the edge labeled with the number a_j. From the definition of the set $D(\Theta)$ it follows that f_i is contained in the set $E(\Theta)$, $E(\Theta, f_i) = \{a_1, \ldots, a_t\}$ and for $j = 1, \ldots, t$, the decision tree Γ_j belongs to the set $D(\Theta(f_i, a_j))$. According to the inductive hypothesis, the tree Γ_j is a decision tree for the table $\Theta(f_i, a_j)$. Then the tree Γ is a decision tree for the table Θ.

Now we consider an arbitrary decision tree Γ for the table Θ. According to the definition, the root of Γ is labeled with an attribute f_i from the set $E(\Theta)$, edges leaving the root are labeled with numbers from the set $E(\Theta, f_i)$ and the subtrees whose roots are nodes, to which these edges enter, are decision trees for corresponding descendants of the node Θ. Then, according to the definition of the set $D(\Theta)$ and to inductive hypothesis, the tree Γ belongs to the set $D(\Theta)$. □

We applied the algorithm to Table 11.1 and show the resulting DAG in Fig. 11.4.

Table 11.1 Example decision table

f_1	f_2	f_3	d
0	0	1	0
1	1	1	1
2	1	0	0
3	1	1	1
4	1	1	1

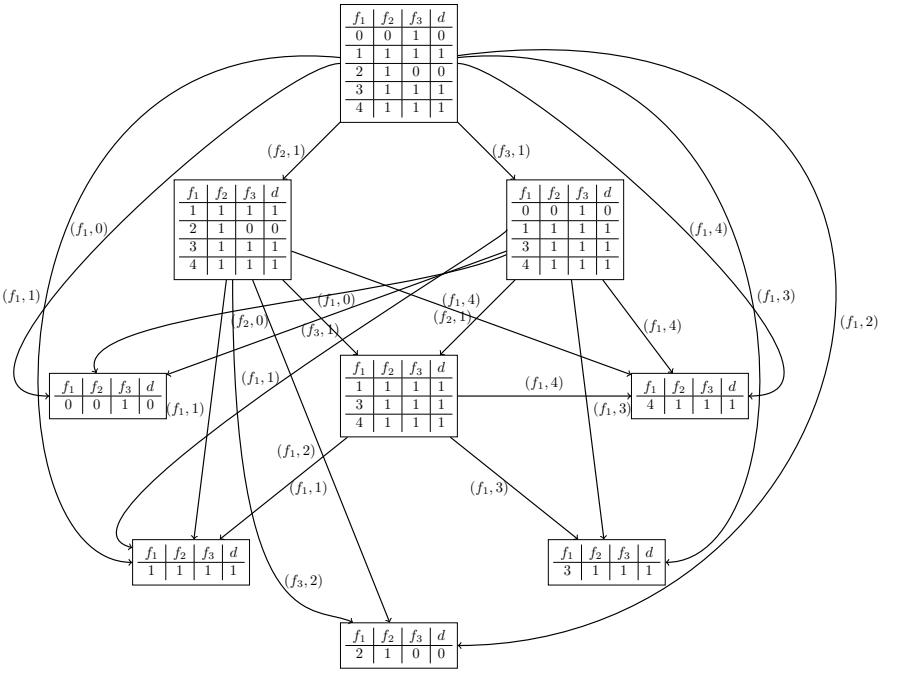


Fig. 11.4 Directed acyclic graph for Table 11.1

11.3 Relationships: Cost vs. Uncertainty

In the following we present an algorithm for computing the relationships between the cost functions and the uncertainty measure R. Let T be a decision table with n columns labeled with $f_1, \ldots, f_n$ and ψ be a monotone and bounded from below cost function given by the pair (ψ^0, F). The algorithm consists of two parts: a) construction of a directed acyclic graph (DAG) $\Delta(T)$ such that the nodes of this DAG are some separable subtables Θ of the table T (this is described in detail in Sect. 11.2.3), and b) computation of $\mathcal{F}_{\psi,\Theta}$ in the bottom-up fashion for each

node Θ of $\Delta(T)$ (the function $\mathcal{F}_{\psi,\Theta}$ is defined in the following). We also show experimental results of application of this algorithm to some datasets from UCI ML Repository [4].

11.3.1 The Function $\mathcal{F}_{\psi,T}$

Let T be a decision table and ψ be a monotone and bounded from below cost function. The function $\mathcal{F}_{\psi,T}(\alpha)$ is defined on the set $\{0,\dots,R(T)\}$. For any $\alpha \in \{0,\dots,R(T)\}$, the value of $\mathcal{F}_{\psi,T}(\alpha)$ is equal to the minimum cost of an α-decision tree for T, relative to the cost function ψ. This function can be represented by the tuple.

$$\left(\mathcal{F}_{\psi,T}(0),\dots,\mathcal{F}_{\psi,T}(R(T))\right).$$

This function allows us to describe relationship of decision tree cost and uncertainty.

11.3.2 Computing the Relationship

For each node Θ of the graph $\Delta(T)$ we compute the function $\mathcal{F}_\Theta = \mathcal{F}_{\psi,\Theta}$ (we compute the $R(\Theta)$-tuple describing this function).

A node of $\Delta(T)$ is called *terminal* if there are no edges leaving this node. We will move from the terminal nodes, which are labeled with numbers, to the node T.

Let Θ be a terminal node of $\Delta(T)$ which is labeled with a number b that is the most common decision for Θ. We know that $R(\Theta) = 0$. Therefore, b is a common decision for Θ, and the decision tree depicted in Fig. 11.2 is the only 0-decision tree for Θ. Since $R(\Theta) = 0$, we should consider only one value of α – the value 0. It's clear that the minimum cost of 0-decision tree for Θ is equal to ψ^0. Thus, the function $\mathcal{F}_\Theta$ can be described by the tuple (ψ^0).

Let Θ be a nonterminal node of $\Delta(T)$ then it means that $R(\Theta) > 0$. Let $\alpha \in \{0,\dots,R(\Theta)\}$. We need to find the value $\mathcal{F}_\Theta(\alpha)$, which is the minimum cost relative to ψ of an α-decision tree for Θ. Since $R(\Theta) > 0$, the root of any α-decision tree for Θ is labeled with an attribute from $E(\Theta)$. For any $f_i \in E(\Theta)$, we denote by $\mathcal{F}_\Theta(\alpha, f_i)$ the minimum cost relative to ψ of an α-decision tree for Θ such that the root of this tree is labeled with f_i. It is clear that

$$\mathcal{F}_\Theta(\alpha) = \min\{\mathcal{F}_\Theta(\alpha, f_i) : f_i \in E(\Theta)\}. \tag{11.1}$$

Let $f_i \in E(\Theta)$ and $E(\Theta, f_i) = \{a_1,\dots,a_t\}$. Then any α-decision tree Γ for Θ with the attribute f_i attached to the root can be represented in the form depicted in Fig. 11.3, where $\Gamma_1,\dots,\Gamma_t$ are α-decision trees for $\Theta(f_i,a_1),\dots,\Theta(f_i,a_t)$. Since ψ

is a monotone cost function, the tree Γ will have the minimum cost if the costs of trees $\Gamma_1, \ldots, \Gamma_t$ are minimum. Therefore,

$$\mathcal{F}_{\Theta}(\alpha, f_i) = F(N(\Theta), \mathcal{F}_{\Theta(f_i,a_1)}(\alpha), \ldots, \mathcal{F}_{\Theta(f_i,a_t)}(\alpha)). \quad (11.2)$$

If for some j, $1 \leq j \leq t$, we have $\alpha > R(\Theta(f_i, a_j))$ then $\mathcal{F}_{\Theta(f_i,a_j)}(\alpha) = \psi^0$, since the decision tree depicted in Fig. 11.2, where b is the most common decision for $\Theta(f_i, a_j)$, is an α-decision tree for $\Theta(f_i, a_j)$. The cost of this tree is ψ^0. Since ψ is a cost function, which is bounded from below, the cost of any α-decision tree for $\Theta(f_i, a_j)$ is at least ψ^0.

The formulas (11.1) and (11.2) allow us to find the value of $\mathcal{F}_{\Theta}(\alpha)$ if we know the values of $\mathcal{F}_{\Theta(f_i,a_j)}(\alpha)$, where $f_i \in E(\Theta)$ and $a_j \in E(\Theta, f_i)$. When we reach to the node T we will obtain the function $\mathcal{F}_T = \mathcal{F}_{\psi,T}$.

As an illustration we applied the algorithm to Table 11.1 and used number of nodes as a cost function. We obtained the following results (as depicted in Fig. 11.5).

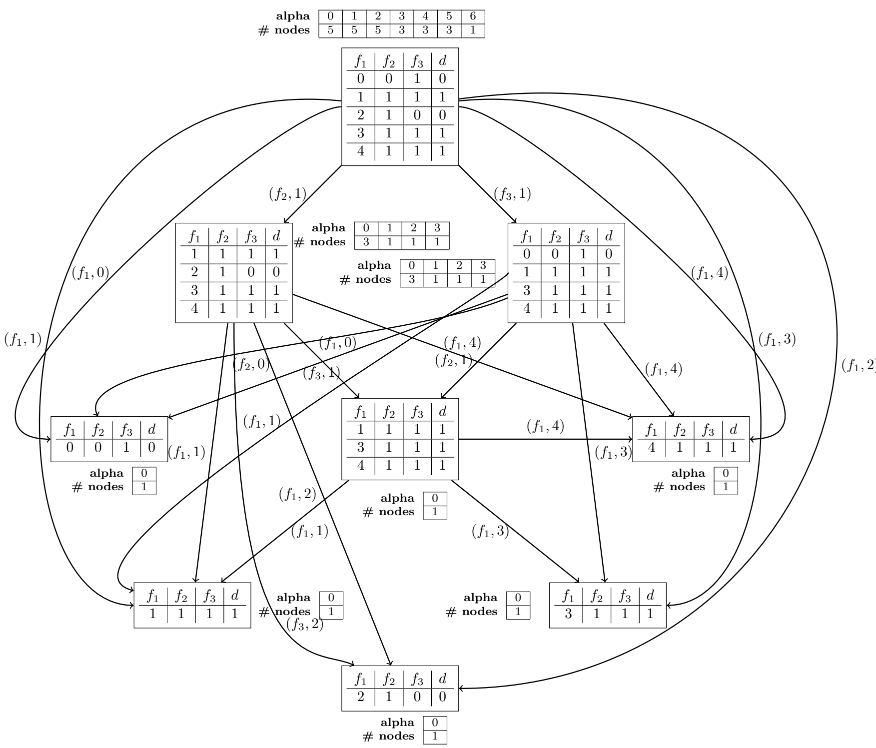


Fig. 11.5 DAG for Table 11.1 with relationships between alpha and number of nodes

11.3.3 Experimental Results

We implemented the algorithm presented in previous subsection in our software system codenamed as DAGGER (this is a software system for the study of decision trees and decision rules, developed at KAUST) and performed several experiments on datasets (decision tables) acquired from UCI ML Repository [4]. In the following, we present the experimental results and show the plots depicting relationships between the uncertainty measure $R(T)$ and various cost functions such as the depth, average depth, number of terminal nodes, etc. We depict here some of the plots generated by the system.

11.3.4 Tic-Tac-Toe Dataset

We show four plots in Fig. 11.6 and Fig. 11.7 for the decision table (dataset) TIC-TAC-TOE (9 attributes and 958 rows). Figure 11.6 contains two plots; the first plot shows the relationship between the depth and alpha and the second plot shows the relationship between the number of nodes and alpha. Figure 11.7 contains two plots; the first plot shows the relationship between the average depth and alpha and the second plot shows relationship between the number of terminal nodes and alpha.

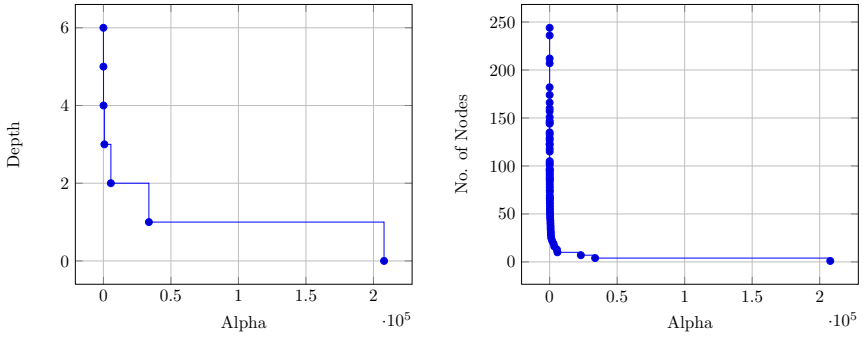


Fig. 11.6 Relationships between depth and alpha and between no. of nodes and alpha for the dataset TIC-TAC-TOE

11.3.5 Lymphography Dataset

For the decision table (dataset) LYMPHOGRAPHY (18 attributes and 148 rows) we show four plots in Fig. 11.8 and Fig. 11.9. Figure 11.8 contains two plots; the first plot shows the relationship between the depth and alpha and the second plot shows

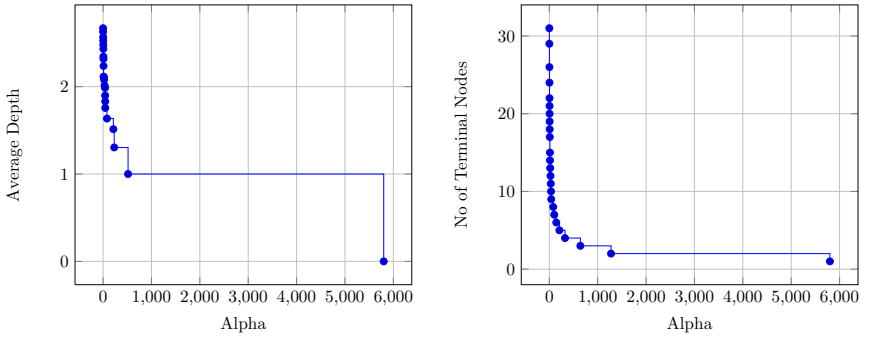


Fig. 11.7 Relationships between average depth and alpha and between no. of terminal nodes and alpha for the dataset TIC-TAC-TOE

the relationship between the number of nonterminal nodes and alpha. Figure 11.9 contains two plots; the first plot shows the relationship between the average depth and alpha and the second plot shows relationship between the number of terminal nodes and alpha.

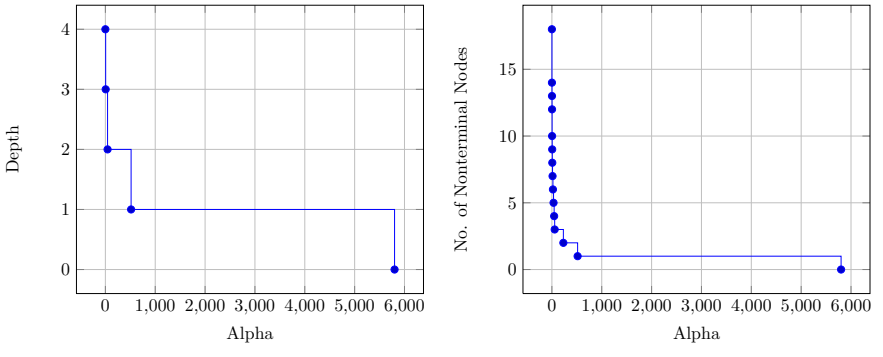


Fig. 11.8 Relationships between depth and alpha and between no. of nonterminal nodes and alpha for the dataset LYMPHOGRAPHY

11.3.6 Breast-Cancer Dataset

For the decision table (dataset) BREAST-CANCER (9 attributes and 266 rows) we show two plots in Fig. 11.10. The first plot shows the relationship between the depth and alpha and the second plot shows the relationship between the number of terminal nodes and alpha.

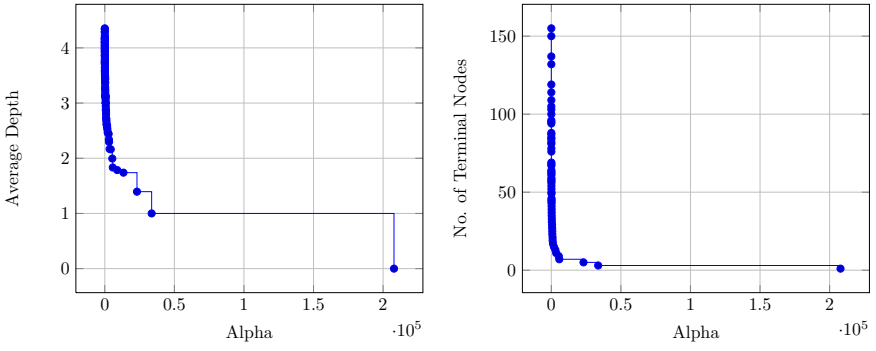


Fig. 11.9 Relationships between average depth and alpha and between no. of terminal nodes and alpha for the dataset LYMPHOGRAPHY

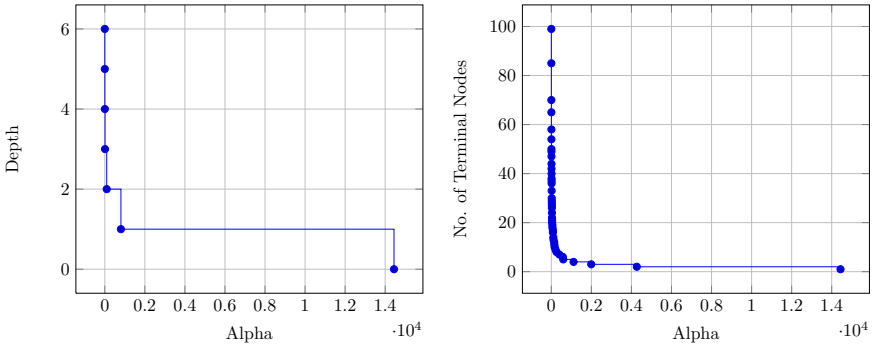


Fig. 11.10 Relationships between depth and alpha and between no. of terminal nodes and alpha for the dataset BREAST-CANCER

11.3.7 Agaricus-Lepiota Dataset

For the decision trable AGARICUS-LEPIOTA (23 attributes and 8125 rows) we show two plots in Fig. 11.11. The first plot shows the relationship between the average depth and alpha and the second plot shows the relationship between the number of nonterminal nodes and alpha.

11.4 Relationships: Number of Nodes vs. Depth

In the following we present an algorithm for computing the relationship between minimum number of nodes and depth of a decision tree (and vice versa). We also

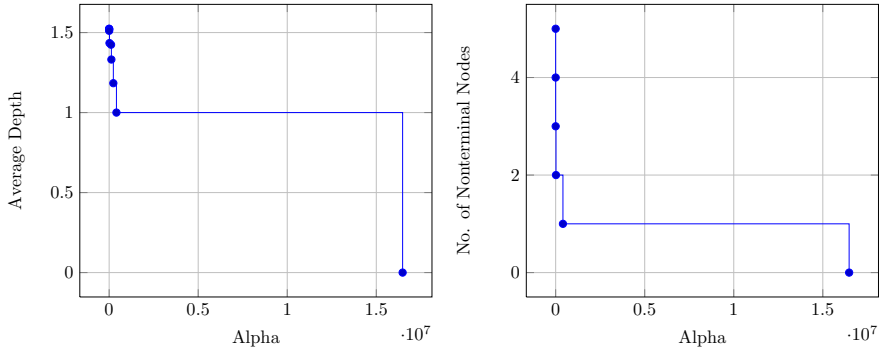


Fig. 11.11 Relationships between average depth and alpha and between no. of nonterminal nodes and alpha for the dataset AGARICUS-LEPIOTA

show experimental results of application of this algorithm to some datasets from UCI ML Repository [4]. This algorithm has also two parts. The first part constructs the graph $\Delta(T)$ (as discussed in Sect. 11.2.3) and the second part computes the relationship.

11.4.1 Computing the Relationships

Let T be a decision table with N rows and m columns labeled with $f_1,\ldots,f_m$ and $D(T)$ be the set of all decision trees for T. Let $\Gamma \in D(T)$, then the depth of Γ, depicted as $h(\Gamma)$, is the maximum length of a path from the root to a terminal node of Γ and the number of nodes for decision tree Γ, denoted as $L(\Gamma)$, is the total number of nodes in Γ.

It is clear that m and $2N-1$ are upper bounds for h and L, respectively, on the set $D(T)$ for any table T.

We denote $B_{h,T} = \{\beta, \beta+1,\ldots,m\}$ and $B_{L,T} = \{\alpha, \alpha+1,\ldots,2N-1\}$, here β and α are minimum depth and minimum number of nodes, respectively, of some decision tree in $D(T)$ (not necessarily the same tree.) We now define two functions $\mathcal{G}_T : B_{h,T} \to B_{L,T}$ and $\mathcal{F}_T : B_{L,T} \to B_{h,T}$ as follows:

$$\mathcal{F}_T(n) = \min\{h(\Gamma) : \Gamma \in D(T), L(\Gamma) \le n\},\ \forall n \in B_{L,T} \text{ and}$$
$$\mathcal{G}_T(n) = \min\{L(\Gamma) : \Gamma \in D(T), h(\Gamma) \le n\},\ \forall n \in B_{h,T}.$$

The function $\mathcal{G}_T$ can be represented by the tuple $(\mathcal{G}_T(\beta), \mathcal{G}_T(\beta+1),\ldots,\mathcal{G}_T(m))$ of its values. The function $\mathcal{F}_T$ can also be represented similarly.

We now describe an algorithm which allows us to construct the function $\mathcal{G}_\Theta$ for any node (subtable) Θ from the graph $\Delta(T)$. We begin from terminal nodes and move to the node T.

Let Θ be a terminal node. It means that all rows of Θ are labeled with the same decision b and the decision tree Γ_b as depicted in Fig. 11.2 belongs to $D(\Theta)$. It is clear that $h(\Gamma_b) = 0$ and $L(\Gamma_b) = 1$ for the table Θ. Therefore $\mathcal{G}_\Theta(n) = 1$ for any $n \in B_{h,\Theta}$.

Consider a node Θ, which is not a terminal node and a bundle of edges, which start from this node. Let these edges be labeled with the pairs $(f_i, a_1), \ldots, (f_i, a_t)$ and enter into the nodes $\Theta(f_i, a_1), \ldots, \Theta(f_i, a_t)$, respectively, to which the functions $\mathcal{G}_{\Theta(f_i,a_1)}, \ldots, \mathcal{G}_{\Theta(f_i,a_t)}$ are already attached.

Let $b_1, \ldots, b_t$ be the minimum values from $B_{h,\Theta(f_i,a_1)}, \ldots, B_{h,\Theta(f_i,a_t)}$, respectively. Let $B_{h,\Theta,f_i} = \{\beta_i, \beta_i + 1, \ldots, m\}$, where $\beta_i = 1 + \max\{b_1, \ldots, b_t\}$. One can show that β_i is the minimum depth of a decision tree from $D(\Theta)$ for which f_i is attached to the root and $\beta_\Theta = \min\{\beta_i : f_i \in E(\Theta)\}$, where β_Θ is the minimum value from $B_{h,\Theta}$.

We correspond to this bundle (f_i-bundle) the function $\mathcal{G}_\Theta^{f_i}$, which for any $n \in B_{h,\Theta,f_i}$ is defined as follows:

$$\mathcal{G}_\Theta^{f_i}(n) = \min\{L(\Gamma) : \Gamma \in D(\Theta, f_i), h(\Gamma) \le n\},$$

where $D(\Theta, f_i)$ is the set of decision trees for Θ corresponding to the considered bundle. In this set we have all trees from $D(\Theta)$ in which the root is labeled with f_i and only such trees. It is not difficult to show that for any $n \in B_{h,\Theta,f_i}$

$$\mathcal{G}_\Theta^{f_i}(n) = \sum_{1 \le j \le t} \mathcal{G}_{\Theta(f_i,a_j)}(n-1) + 1.$$

We can now prove that for any $n \in B_{h,\Theta}$

$$\mathcal{G}_\Theta(n) = \min\left\{\mathcal{G}_\Theta^{f_i}(n) : f_i \in E(\Theta), n \in B_{h,\Theta,f_i}\right\}.$$

We can use the following proposition to construct the function $\mathcal{F}_T$ (using transformation of functions as defined in the Appendix.)

Proposition 11.2. *For any $n \in B_{L,T}$,*

$$\mathcal{F}_T(n) = \min\{p \in B_{h,T} : \mathcal{G}(p) \le n\}.$$

Note that to find the value $\mathcal{F}_T$ for $n \in B_{L,T}$ it is enough to make $O(\log |B_{h,T}|)$ operations of comparison.

As an illustration we applied the algorithm to Table 11.1 to obtain relationships between the depth and number of nodes (see Fig. 11.12).

11.4.2 Experimental Results

We implemented the algorithm presented in the previous subsection in our software system (codenamed as DAGGER) and performed several experiments on datasets

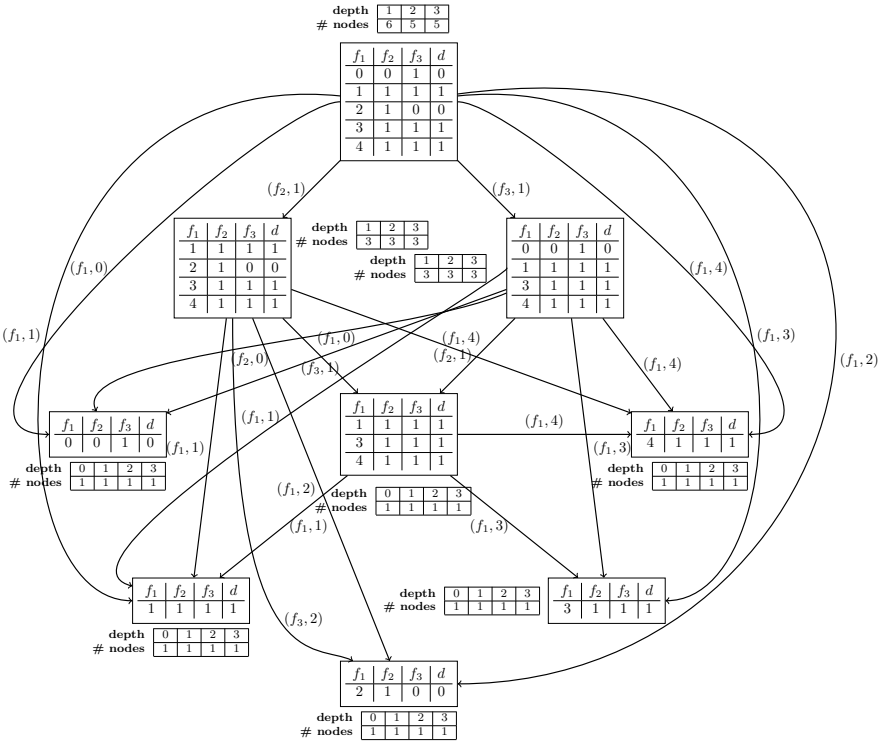


Fig. 11.12 DAG for Table 11.1 with relationships between depth and number of nodes

(decision tables) acquired from UCI ML Repository [4]. In the following, we present the experimental results and show the plots depicting relationships between the number of nodes and depth of decision trees.

11.4.3 Tic-Tac-Toe Dataset

Figure 11.13 contains two plots for the decision table (dataset) TIC-TAC-TOE (9 attributes and 958 rows). The first plot shows the relationship between the number of nodes and the depth and the second one shows the relationship between the depth and the number of nodes of decision trees. In fact we need to extend the second plot up to $2 \times 958 - 1$ but the values of the function will be same, i.e., 6. Note that we have similar situation in each of the considered examples.

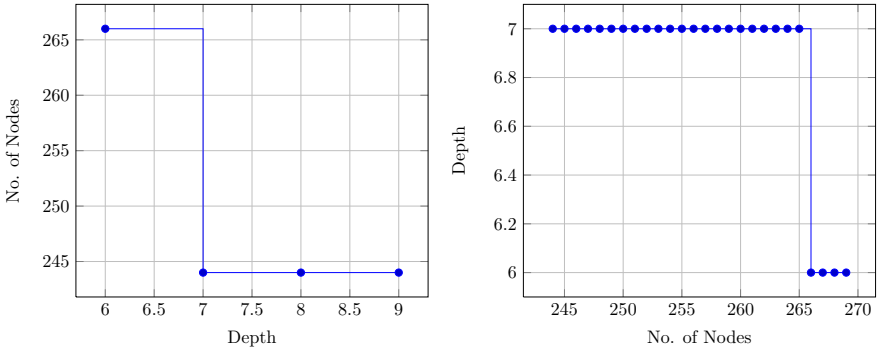


Fig. 11.13 Relationships between the number of nodes and the depth for the dataset TIC-TAC-TOE

11.4.4 Lymphography Dataset

Figure 11.14 contains two plots for the decision table (dataset) LYMPHOGRAPHY (18 attributes and 148 rows). The first plot shows the relationship between the number of nodes and the depth and the second one shows the relationship between the depth and the number of nodes of decision trees.

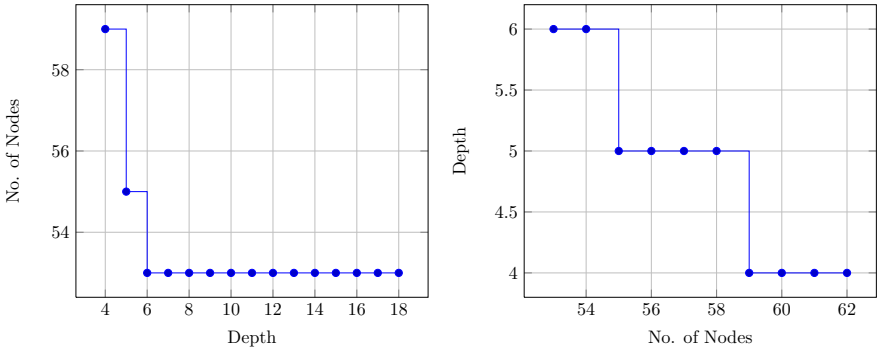


Fig. 11.14 Relationships between the number of nodes and the depth for the dataset LYMPHOGRAPHY

11.4.5 Breast-Cancer Dataset

Figure 11.15 contains two plots for the decision table (dataset) BREAST-CANCER (9 attributes and 266 rows). The first plot shows the relationship between the number of nodes and the depth and the second one shows the relationship between the depth and the number of nodes.

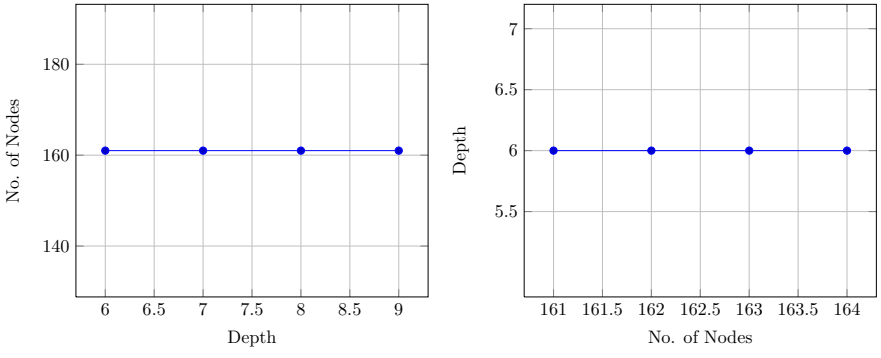


Fig. 11.15 Relationships between the number of nodes and the depth for the dataset BREAST-CANCER

11.4.6 House-Votes-84 Dataset

Figure 11.16 contains two plots for the decision table (dataset) HOUSE-VOTES-84 (16 attributes and 280 rows). The first plot shows the relationship between the number of nodes and the depth and the second one shows the relationship between the depth and the number of nodes.

11.4.7 Agaricus-Lepiota Dataset

Figure 11.17 contains two plots for the decision table (dataset) AGARICUS-LEPIOTA (22 attributes and 8125 rows). The first plot shows the relationship between the number of nodes and the depth and the second one shows the relationship between the depth and the number of nodes.

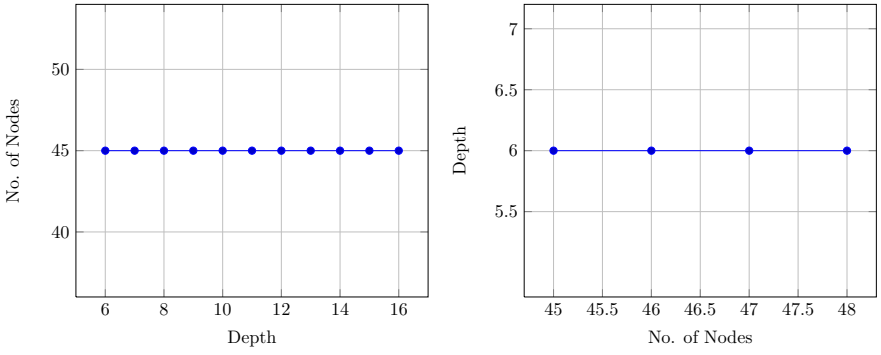


Fig. 11.16 Relationships between the number of nodes and the depth for the dataset HOUSE-VOTES-84

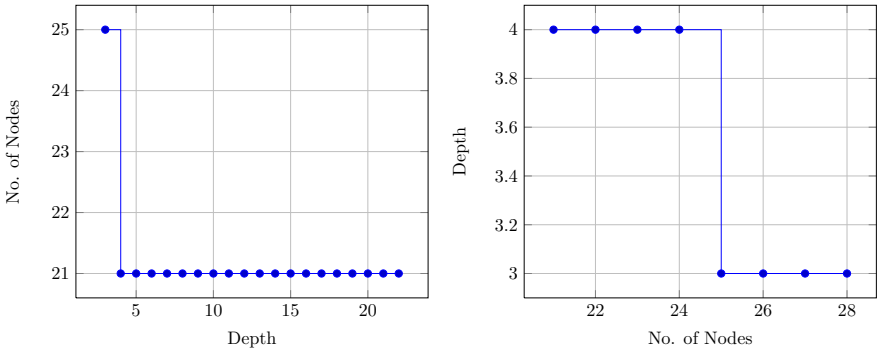


Fig. 11.17 Relationships between the number of nodes and the depth for the dataset AGARICUS-LEPIOTA

11.5 Conclusion

This Chapter is devoted to the developing of two new tools for studying of decision trees. The first tool we consider allows us to compute relationship between cost and uncertainty of decision trees. We have considered a variety of cost functions and a measure of uncertainty. The second tool presented in this chapter deals with computation of relationship between number of nodes and depth of decision trees. We also give a simple mechanism utilizing the "transformation of functions" (as described in the appendix) to reverse the order of computation of relationship for the second tool i.e., the relationship between the depth and number of nodes. We have integrated these tools in our software system (codenamed as DAGGER) and performed several experiments on datasets from UCI ML Repository [4]. We plan to extend the research in this area by examining the relationships between various other cost functions such

as relationship between average depth and number of nodes of decision trees, as well as other uncertainty measures and inconsistent decision tables [5].

Acknowledgements. The authors are greatly indebted to Prof. Andrzej Skowron for his comments and suggestions.

References

1. Alkhalid, A., Chikalov, I., Moshkov, M.: On Algorithm for Building of Optimal α-Decision Trees. In: Szczuka, M., Kryszkiewicz, M., Ramanna, S., Jensen, R., Hu, Q. (eds.) RSCTC 2010. LNCS (LNAI), vol. 6086, pp. 438–445. Springer, Heidelberg (2010)
2. Chikalov, I., Hussain, S., Moshkov, M.: Relationships between Depth and Number of Misclassifications for Decision Trees. In: Kuznetsov, S.O., Ślęzak, D., Hepting, D.H., Mirkin, B.G. (eds.) RSFDGrC 2011. LNCS, vol. 6743, pp. 286–292. Springer, Heidelberg (2011)
3. Chikalov, I., Moshkov, M., Zelentsova, M.: On Optimization of Decision Trees. In: Peters, J.F., Skowron, A. (eds.) Transactions on Rough Sets IV. LNCS, vol. 3700, pp. 18–36. Springer, Heidelberg (2005)
4. Frank, A., Asuncion, A.: UCI Machine Learning Repository (2010), `http://archive.ics.uci.edu/ml` (cited September 19, 2011)
5. Pawlak, Z.: Rough Sets – Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Dordrecht (1991)

Appendix: Transformation of Functions

Let f and g be two functions from a set A onto C_f and C_g respectively, where C_f and C_g are finite sets of nonnegative integers. Let $B_f = \{m_f, m_f+1, \ldots, M_f\}$ and $B_g = \{n_g, n_g+1, \ldots, N_g\}$ where $m_f = \min\{m : m \in C_f\}$ and $n_g = \min\{n : n \in C_g\}$. Furthermore, M_f and N_g are natural numbers such that $m \leq M_f$ and $n \leq N_g$ for any $m \in C_f$ and $n \in C_g$, respectively.

We define two functions $\mathcal{F} : B_g \to B_f$ and $\mathcal{G} : B_f \to B_g$ as following:

$$\mathcal{F}(n) = \min\{f(a) : a \in A, g(a) \leq n\}, \ \forall n \in B_g, \text{ and}$$

and

$$\mathcal{G}(m) = \min\{g(a) : a \in A, f(a) \leq m\}, \ \forall m \in B_f.$$

It is clear that both $\mathcal{F}$ and $\mathcal{G}$ are nonincreasing functions.

The following proposition states that the functions $\mathcal{F}$ and $\mathcal{G}$ can be used interchangeably and we can evaluate $\mathcal{F}$ using $\mathcal{G}$ and vice versa, i.e., it is enough to know only one function to evaluate the other.

Proposition 11.3. *For any $n \in B_g$,*

$$\mathcal{F}(n) = \min\{m \in B_f : \mathcal{G}(m) \leq n\},$$

and for any $m \in B_f$,

$$\mathcal{G}(m) = \min\{n \in B_g : \mathcal{F}(n) \leq m\}.$$

Proof. Let for some $n \in B_g$

$$\mathcal{F}(n) = m_0. \tag{11.3}$$

Furthermore, we assume that

$$\min\{m \in B_f : \mathcal{G}(m) \leq n\} = t.$$

From (11.3) it follows that

(i) there exists $b \in A$ such that $g(b) \leq n$ and $f(b) = m_0$;
(ii) for any $a \in A$ if $g(a) \leq n$ then $f(a) \geq m_0$.

From (i) it follows that $\mathcal{G}(m_0) \leq n$. This implies $t \leq m_0$. Let us assume that $t < m_0$. In this case, there exits $m_1 < m_0$ for which $\mathcal{G}(m_1) \leq n$. Therefore, there exists $a \in A$ such that $f(a) \leq m_1$ and $g(a) \leq n$, but from (ii) it follows that $f(a) \geq m_0$, which is impossible. So $t = m_0$.

Similarly we can prove the second part of the statement. □

Proposition 11.3 allows us to transform the function $\mathcal{G}$ given by a tuple $\big(\mathcal{G}(m_f), \mathcal{G}(m_f+1), \ldots, \mathcal{G}(M_f)\big)$ into the function $\mathcal{F}$ and vice versa. We know that $\mathcal{G}(m_f) \geq \mathcal{G}(m_f+1) \geq \cdots \geq \mathcal{G}(M_f)$, to find the minimum $m \in B_f$ such that $\mathcal{G}(m) \leq m$ we can use binary search which requires $O(\log|B_f|)$ comparisons of numbers. So to find the value $\mathcal{F}(n)$ for $n \in B_g$ it is enough to make $O(\log|B_f|)$ operations of comparison.

Chapter 12
The Impact Rules of Recommendation Sources for Adoption Intention of Micro-blog Based on DRSA with Flow Network Graph

Yang-Chieh Chin, Chiao-Chen Chang, Chiun-Sin Lin, and Gwo-Hshiung Tzeng

Abstract. A micro-blog is a social media tool that allows users to write short text messages for public and private networks. This research focuses specifically on the micro-blog on Facebook. The main purposes of this study are to explore and compare what recommendation sources influence the intention to use micro-blogs and to combine the personal characteristics/attributes of gender, daily internet hour usage and past use experience to infer the usage of micro-blogs decision rules using a dominance-based rough-set approach (DRSA) with flow network graph. Data for this study were collected from 382 users and potential users. The analysis is grounded in the taxonomy of induction-related activities using a DRSA with flow network graph to infer the usage of micro-blogs decision rules. Finally, the study of the nature of micro-blog reflects essential practical and academic value in real world.

Keywords: Micro-blog, Facebook, dominance-based rough set approach (DRSA), flow network graph, recommendation source, adoption intention.

Yang-Chieh Chin
Department of Management Science, National Chiao Tung University
1001 Ta-Hsueh Road, Hsinchu 300, Taiwan
e-mail: jerry110888@gmail.com

Chiao-Chen Chang
Department of International Business, National Dong Hwa University
No.1, Sec.2, Da Hsueh Rd., Shoufeng, Hualien 97401, Taiwan
e-mail: aka@mail.ndhu.edu.tw

Chiun-Sin Lin
Department of Management Science, National Chiao Tung University
1001 Ta-Hsueh Road, Hsinchu 300, Taiwan
e-mail: netec@yahoo.com.tw

Gwo-Hshiung Tzeng
Department of Business Administration, Kainan University
No. 1 Kainan Road, Luchu, Taoyuan 338, Taiwan
e-mail: ghtzeng@mail.knu.edu.tw

and

Institute of Management of Technology, National Chiao Tung University
1001 Ta-Hsueh Road, Hsinchu 300, Taiwan
e-mail: ghtzeng@cc.nctu.edu.tw

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 223–239.
springerlink.com

12.1 Introduction

A micro-blog is a new communication channel with which people can share information for public and private networks. Micro-blog platforms, primarily on social network sites such as Facebook, Twitter, and Orkut have become popular. The concept of a social network is that two of your friends would have a greater probability of knowing each other than would two people chosen at random from the population [5]. Extensions of micro-blog communications include status updates from social networks such as Facebook, and message-exchange services such as Twitter. User growth on Facebook, one of the biggest social networking sites in the world, is still expanding. Statistics from `www.checkfacebook.com` showed that Facebook's international audience totaled 350 million people at the beginning of 2010, including more than 5 million Taiwan users engaged in platform applications. A site that allows users to share daily updates through micro-blog helps people to keep in touch [23], and businesses can increase sales as well by improving communications to and collaborations with customers [5]. With the growth of users on the micro-blog services, the biggest benefit of micro-blog is its ability to generate platform revenues by means of advertisements [45] and other applications. Thus, how to stimulate the micro-blog adoption intention becomes a critical issue to platform marketers.

Even though micro-blog offers conveniences and benefits, some people are concerned about the use of micro-blog as another form of background check and that their privacy may be lost in cyberspace [43]. However, such concerns can be addressed by better and more accurate recommendations, because people are influenced by others' recommendations when making decisions [24]. These recommendations can be classified as interpersonal sources, impersonal sources [4] and neutral sources [12]. Researchers have shed some light on the importance of recommendation sources in the context of product purchases [33], but little has been done on the relevance of these recommendation sources in the context of micro-blog usage. Thus, our primary goal in this study is to fill that gap by increasing our understanding of how the three primary categories of recommendation sources-interpersonal recommendations (*e.g.*, word-of-mouth recommendations), impersonal recommendations (*e.g.*, advertising recommendations), and neutral recommendations (*e.g.*, expert recommendations)-influence users intention to adopt micro-blogs.

The classical rough set theory (RST) was proposed by Pawlak [36] as an effective mathematical approach for discovering hidden deterministic rules. However, the main restriction for the use RST in multicriteria decision making (MCDM) was that the domain of attributes is preference ordered. To help fill the gap, Greco et al. [20] proposed an extension of the rough set theory based on the dominance principle, which incorporate the ordinal nature of the preference data into the classification problem. This innovation is called the dominance-based rough set approach (DRSA). It derived a set of decision rules based on preference-ordered data [21] and substituted the indiscernibility relation in classical rough set theory with a dominance relation that is reflexive and transitive [9]. Furthermore, the DRSA represents preference models for multiple criteria decision analysis (MCDA) problems, where preference orderings on domains of attributes are typical in exemplary-based

decision-making (Liou et al., 2010 [30]; Liou and Tzeng, 2010 [31]; Shyng et al., 2011 [44]). In the DRSA, these models take the following form: "if object *a* is preferred to object *b* in at least (at most) given degrees with respect to some attributes, then *a* is preferred to *b* in at least (at most) given degree" (Słowiński, 2010 [46]). Decision rules derived from the options of users, expressing the relationships between attributes, values and the intention to adopt a micro-blog, are also considered. Although the rules developed by RST can be directly translated into a path-dependent flow network to infer decision paths and parameters (Ford and Fulkerson, 1962 [17]; Pawlak, 2002 [37], 2004 [38], 2005 [39]; Wang et al., 2010 [51]; Ou Yang et al., 2010 [34]; Lin et al., 2011 [29]; Fang et al., 2012 [16]), most studies have not yet adequately induced and extracted the decision rules using the DRSA and flow network graph. The flow network graph is heavily exploited by using if-then rules in easily understanding the impact decision rules of recommendation sources. Therefore, another purpose of this study is to combine control variables (gender, daily internet hour usage, and past use experience), grounded in the taxonomy of induction-related activities using the DRSA, to infer the micro-blog-related decision rules.

As a result, our primary goal in this study is to fill that gap by increasing our understanding of how recommendation sources (word-of-mouth recommendations, advertising recommendations and expert recommendations) and personal characteristics (gender, daily Internet hour usage, and past use experience) influence user intention by combining the DRSA and flow network graph to infer the micro-blog adoption decision rules.

The remainder of this paper is organized as follows. Section 12.2 reviews prior studies on recommendation sources for adoption intention. Section 12.3 introduces the DRSA and Flow Network Graph Algorithm. Section 12.4 illustrates the empirical example of Facebook to demonstrate the proposed methods. Finally, Section 12.5 presents conclusions and remarks.

12.2 Review on Recommendation Sources for Adoption Intention

12.2.1 Micro-blog

Micro-blog systems provide a lightweight, easy form of communication that enables users to broadcast and share information about their current activities, thoughts, opinions and status. Compared to regular blogging, micro-blogging lowers the investment of the time and thought required to generate content and fulfills a need for a faster and more immediate mode of communication [27]. Micro-blogging, communication via short, real-time message broadcasts, is relatively a new communication channel for people to share information about their daily activities that they would not otherwise publish using other media (*e.g.*, e-mail, phone, instant messaging (IM) or weblogs). In a micro-blogging community, users can publish brief messages and

tag them with keywords. Others may subscribe to these messages based on who publishes them or what they are about [23]. Popular micro-blogging platforms such as Facebook have risen to prominence in recent years.

12.2.2 Adoption Intention

Adoption is a widely researched process that is often used to investigate the spread of information technology [15, 41, 42]. According to the literature on information technology adoption, adoption intention is an individual's intention to use, acquire, or accept a technology innovation [42]. Adoption intention is the subjective likelihood of an individual performing a specified behavior and serves as the major determinant of actual usage behavior (Ajzen, 1988 [1]; Ajzen and Fishbein, 1980 [2]; Yi et al., 2006 [52]). In a virtual environment, adoption intention has been shown to predict the likelihood of an individual performing a conscious act, such as determining to accept (or use) a technology (Chau and Hu, 2002 [10]). Service providers should encourage usage when users are willing to use social network. Thus, it becomes necessary to probe the adoption intentions of users of micro-blogs.

12.2.3 Recommendation Source

Prior studies have suggested that peer communications (such as families, friends, and colleges) may be considered the most trustworthy type of recommendation source in making decisions (Richins and Root-Shaffer, 1988 [40]). In addition, advertising recommendations, such as recommendations from site-sponsored advertisements, may be also regarded as a credibility cue (Smith et al., 2005 [49]). Previous research has also demonstrated that the perceived level of expertise positively impacts acceptance of source recommendations (Crisci and Kassinove, 1973 [13]). These recommendations may be also considered a credibility cue when making decisions [49].

12.3 Basic Concepts of the DRSA and Flow Network Graph Algorithm

Pawlak (1982) [36] proposed the classical rough set theory (RST) as an effective mathematical approach for discovering hidden deterministic rules and associative patterns in all types of data and for managing unknown data distributions and information uncertainty. Therefore, many studies have adopted the RST approach to extract rules and patterns from original data and unclassified information. For a long time, however, the main restriction on the use of RST has been the preference-ordering of domain of attributes because RST cannot handle inconsis-

tencies due to violations of the dominance principle (Greco et al., 2001 [21]). For this reason, Greco et al. (1998) [19] proposed the dominance-based rough set approach (DRSA) to substitute a reflexive and transitive dominance relation for the indiscernible relation used in the classical RST approach (Greco et al., 2007 [22]). The DRSA derives a set of decision rules from preference-ordered data (Słowiński et al., 2009 [48]) that are then used in a classifier (Błaszczyński et al., 2007 [7]). To understand the construction of decision rules of recommendation sources that influence the adoption intentions of customers toward micro-blogs, this research starts by combining the DRSA with flow network graph to extract and discover recommendation sources.

12.3.1 Data Table

DRSA uses an ordered information table where in each row represents an object, which is defined a respondent to our survey, and each column represents an attribute, including preference-ordered domain and regular (no preference-ordered domain). Thus, the entries of the table are attribute values. Formally, an information system (IS) can be represented by the quintuple $IS = (U, Q, V, f)$, where U is a finite and non-empty set of objects (universe), $Q = \{a_1, \ldots, a_m\}$ is a non-empty finite set of ordered or non-ordered attributes, V_a is the domain of attribute a, $V = \bigcup_{a \in Q} V_a$, and $f : U \times Q \longrightarrow V$ is a total information function such that $f(x,a) \in V_a$ for every $a \in Q$ and $x \in U$ [7,30,36]. The set Q is usually divided into a set C of ordered or non-ordered attributes and a set D of decision attributes.

12.3.2 Approximation of the Dominance Relation

In DRSA, it is used as a dominance relation instead of an indisceribility relation [7]. According to Greco et al. [21], first, let $\succeq_a$ be an outranking relation on U with respect to criterion $a \in Q$, such that $x \succeq_a y$ means "x is at least good as y with respect to criterion a." Suppose that $\succeq_a$ is a complete preorder. Furthermore, let $\mathbf{Cl} = \{C_t : t \in T\}$, where $T = \{1, \ldots, n\}$, be a set of decision classes on U that each $x \in U$ belongs to one and only one class $C_l \in \mathbf{Cl}$. Assume that, for all $r, s \in T$ such that $r > s$, the elements of C_r are preferred to the elements of C_s . Given the set of decision classes $\mathbf{Cl}$, it is possible to define upward and downward unions of classes, respectively, by

$$CL_t^{\geq} = \bigcup_{s \geq t} Cl_s, \quad Cl_t^{\leq} = \bigcup_{s \leq t} Cl_s, \quad t = 1, \ldots, n. \tag{12.1}$$

In dominance-based approaches, we say that x dominates y with respect to $P \subseteq C$, in symbols $x \succeq_P y$, if $x \succeq_a y$ for all $a \in P$. Given $P \subseteq C$ and $x \subseteq U$, let

$$D_p^+(x) = \{y \in U : y \succeq_P x\},$$

represent a set of objects dominating x , called a P-dominating set, and let

$$D_p^-(x) = \{y \in U : x \succeq_P y\},$$

represent a set of objects dominated by x, called a P-dominated set. One can adopt $D_p^+(x)$ and $D_p^-(x)$ to approximate collections of upward and downward unions of decision classes.

The P-lower approximation $\underline{P}(Cl_t^{\geq})$ of $Cl_t^{\geq}$ (where $t \in \{2,3,\ldots,n\}$) relative to $P \subseteq C$ contains all objects x from the universe U, such that objects y having at least the same evaluations as x for all of the considered ordered attributes from P also belong to class Cl_t or better (i.e., to Cl_s, where $s > t$). More formally $\underline{P}(Cl_t^{\geq})$ is defined by

$$\underline{P}(Cl_t^{\geq}) = \{x \in U : D_P^+(x) \subseteq Cl_t^{\geq}\}. \tag{12.2}$$

The P-upper approximation $\overline{P}(Cl_t^{\geq})$ of $Cl_t^{\geq}$, $t \in \{2,3,\ldots,n\}$ relative to $P \subseteq C$ is composed of all objects x from the universe U whose evaluations on the criteria from P are not worse than the evaluations of at least one object y belonging to class Cl_t, or better. More formally $\overline{P}(Cl_t^{\geq})$ is defined by

$$\overline{P}(Cl_t^{\geq}) = \{x \in U : D_P^-(x) \cap Cl_t^{\geq} \neq \varnothing\}. \tag{12.3}$$

Analogously, the P-lower and P-upper approximations $\underline{P}(Cl_t^{\leq})$ and $\overline{P}(Cl_t^{\leq})$ of the class unions $Cl_t^{\leq}$ (where $t \in \{2,\ldots,n\}$) relative to $P \subseteq C$ are defined by

$$\underline{P}(Cl_t^{\leq}) = \{x \in U : D_P^-(x) \subseteq Cl_t^{\leq}\}, \tag{12.4}$$

and

$$\overline{P}(Cl_t^{\leq}) = \{x \in U : D_P^+(x) \cap Cl_t^{\leq} \neq \varnothing\}, \tag{12.5}$$

respectively.

The P-boundaries (P-doubtable regions) of $Cl_t^{\geq}$ and $Cl_t^{\leq}$ are defined as

$$Bn_P(Cl_t^{\geq}) = \overline{P}(Cl_t^{\geq}) - \underline{P}(Cl_t^{\geq}), \tag{12.6}$$

$$Bn_P(Cl_t^{\leq}) = \overline{P}(Cl_t^{\leq}) - \underline{P}(Cl_t^{\leq}). \tag{12.7}$$

For any set $P \subseteq U$ we can estimate the accuracy of approximation of $Cl_t^{\geq}$ and $Cl_t^{\leq}$ by

$$\alpha_p(Cl_t^{\geq}) = \left|\frac{\underline{P}(Cl_t^{\geq})}{\overline{P}(Cl_t^{\geq})}\right|, \quad \alpha_p(Cl_t^{\leq}) = \left|\frac{\underline{P}(Cl_t^{\leq})}{\overline{P}(Cl_t^{\leq})}\right| \tag{12.8}$$

and the ratio

$$\gamma_p(\mathbf{Cl}) = \left| \frac{U - \left(\bigcup_{t \in \{2,\ldots,n\}} Bn_p(Cl_t^{\geq}) \right)}{U} \right| = \left| \frac{U - \left(\bigcup_{t \in \{1\ldots,n-1\}} Bn_p(Cl_t^{\leq}) \right)}{U} \right|. \tag{12.9}$$

12.3.3 Extraction of Decision Rules

A decision rule can be expressed as a logical manner of the if (antecedent) then (consequence) type of decision. The procedure of capturing decision rules from a set of initial data is known as induction (Pawlak, 1982 [36]). For a given upward union of classes, $Cl_t^{\geq}$, the decision rules included under the hypothesis that all objects belonging to $\underline{P}(Cl_t^{\geq})$ are positive and the others are negative. There are two types of decision rules as follows:

(1) $D_{\geq}$ decision rules ("at least" decision rules):
If $f(x,a_1) \geq r_{a_1}$ and $f(x,a_2) \geq r_{a_2}$ and $\ldots f(x,a_p) \geq r_{a_p}$, then $x \in Cl_t^{\geq}$.

(2) $D_{\leq}$ decision rules ("at most" decision rules):
If $f(x,a_1) \leq r_{a_1}$ and $f(x,a_2) \leq r_{a_2}$ and $\ldots f(x,a_p) \leq r_{a_p}$, then $x \in Cl_t^{\leq}$.

12.3.4 Decision Rules Based on Flow Network Graph

Under the assumption of decision rules of customer adoption intention characteristics, this research finds a path-dependent figure that depends on the rule and initial characteristics of adoption intention potential. The basis of the flow network graph can be traced back to Ford and Fulkerson (1962) [17]. According to the flow network graph and Bayes's theorem (Pawlak, 2002 [37]), the model was used to capture and describe the nature of decision processes within flow network graphs rather than as a description of flow optimization. The relationship between flow network graphs and decision algorithms is presented as follows (Pawlak, 2004 [38], 2005 [39]; Ou Yang et al., 2011 [34]; Wang et al., 2010 [51]; Lin et al., 2011 [29]).

More precisely, a flow graph is a directed acyclic finite graph $G = (V, \beta, h)$, where V is a set of nodes, $\beta \subseteq V \times V$ is a set of directed branches, $h : \beta \to R^+$ is a flow function and R^+ is the set of non-negative real numbers. The through-flow of a branch $(x,y) \in \beta$ and can be defined as $r(x,y)$. The input of a node $x \in V$ is the set $I(x) = \{y \in V | (y,x) \in \beta\}$, and the output of a node $x \in V$ is defined as $O(x) = \{y \in V | (x,y) \in \beta\}$. Based on these concepts, the input and output of a graph G are defined as $I(G) = \{x \in V | I(x) = \varnothing\}$ and $O(G) = \{x \in V | O(x) = \varnothing\}$. For every node x in flow graph, inflow is defined as $h_+(x) = \sum_{y \in I(x)} h(y,x)$, and outflow is defined as $h_-(x) = \sum_{y \in O(x)} h(x,y)$. Analogously, the inflow and outflow of the whole flow graph can be defined as $h_+(G) = \sum_{x \in I(G)} h_-(x)$ and $h_-(G) = \sum_{x \in O(G)} h_+(x)$,

respectively. This research assumes that "flow graph conservation" holds, i.e., $h_+(x) = h_-(x) = h(x)$ for any node in a flow graph G.

To measure the strength of every branch (x,y) in a flow graph $G = (V, \beta, h)$, this research defines the strength $\rho(x,y) = h(x,y)/r(G)$. Obviously, $0 \leq \rho(x,y) \leq 1$. The strength of the branch simply expresses the amount of total flow through the branch. Every branch (x,y) of a flow graph G associates with certainty and coverage coefficients. The certainty and coverage of every branch are defined by $\mathbf{cer}(x,y) = \rho(x,y)/\rho(x)$ and the $\mathbf{cov}(x,y) = \rho(x,y)/\rho(y)$, respectively, where $\rho(x,y) = h(x,y)/h(G)$, $\rho(x) = h(x)/h(G)$ and $\rho(y) = h(y)/h(G)$ are normalized through-flow and $\rho(x) \neq 0$, $\rho(y) \neq 0$, and $0 \leq \rho(x,y) \leq 1$. The meaning of certainty coefficient expresses outflow distribution between outputs of a node, whereas the coverage coefficient exhibits the distribution of inflow between inputs of the node. The above coefficients simply explain properties of flow distribution among branches in the whole flow network graph. Hence, in this research, combining the DRSA with a flow network graph also applied influence diagrams to help decision-makers and managers by presenting a set of adoption intention decision rules to easily describe appropriate decisions. The influence diagram connects as many rules as possible from the contextual aspects of the data; hence, the relationship between the DRSA and influence diagram is complementary.

12.4 An Empirical Example of Micro-blog

Micro-blogging appeals to a wide range of individuals for various purposes, such as posting personal anecdotes, sharing new information, finding new friends and connecting acquaintances. A recommendation source that matches consumer-specified criteria to the product assortment can help consumers reduce perceived risks and save time when considering a wide variety of alternative products. In this section, we design and develop a flow network dependent by combining DRSA and flow network graph decision rules for adopting micro-blog, such as Facebook, etc. In this section, we use the JAMM software [47] to generate decision rules. The results are used to understand the influence of recommendation sources on the intention to adopt Facebook. The proposed approach was successfully employed in the academic empirical study.

12.4.1 Selection Variables and Data

In this study, a total of 1,108 undergraduate students from a university in Northern Taiwan participated in the survey by completing a questionnaire containing study measures of their intentions to adopt micro-blogs. Student subjects were selected for this study because the focus of the survey, social networks (and in this case, Facebook), is relevant for university students. Within the sample population, 585

(52.8%) were female and 523 (47.2%) were male. More than 83.1% of participants spent more than an average of three hours on the Internet per day. Only a few participants (3.7%) had no micro-blog experience, while 62% of participants had used micro-blogs, such as Facebook, more than 10 times in the last month. Finally, most of the participants (90.3%) stated that they were familiar with the term "Facebook" prior to taking the survey.

Previous studies pertaining to the feeling and the efficacy of communicating on the Internet have confirmed the effect of gender differences on Internet usage (Akhter, 2003 [3]; Janda, 2008 [26]). In addition, daily Internet usage (i.e., the average number of hours a person is on the Internet in a 24-hour period) (Hoadley et al., 2010 [25]) and past use experience (i.e., past micro-blog use) (Sung et al., 2005 [50]) can also reflect the composition of the users. Therefore, this study also included three personal attributes of participants (gender, daily Internet usage, and past use experience) and the attributes of the recommendation sources (WOM, advertising, and experts). In addition, one decision attribute, the adoption intention, was included to pre-process the data to construct the information table, which represents knowledge in a DRSA model.

The attributes of recommendation sources were measured in three dimensions: WOM (friend or classmate reviews, *e.g.*, "Your friend/classmate talked to you about the advantage of a micro-blog, such as Facebook"), advertising (*e.g.*, "The platform providers presented advertisements on the web page to attract users"), and expert recommendations (professional comments, *e.g.*, "A relevant professional introduced the benefits of micro-blogs in a magazine"). The respondents were asked to choose the recommendation source they would normally consult and to indicate the extent to which they perceived the influence of the recommendation on a 5-point Likert-type scale, with anchors ranging from not very important (1 score) to very important (5 score). Furthermore, the participants were asked to evaluate their micro-blog usage intentions using multi-item scales adapted from Ajzen (1988) [1], measured on a 5-point Likert-type scale from strongly disagree (1 score) to strongly agree (5 score). The domain values of these personal attributes and recommendation sources are shown in Table 12.1.

12.4.2 Rules for the Intention to Adopt Micro-blog

In this study, we used the jMAF software (Słowiński, 2006 [47]) to generate decision rules[1]. The decision rules extraction procedures of the DRSA enable the generation of a large number of rules related to intentions to adopt micro-blogs. The first results obtained from the DRSA analysis of the coded information table approximated the decision classes and the quality of their classifications. These results revealed that the data were very well categorized and appropriate for understanding how personal and recommendation source attributes would influence micro-blog adoption

[1] Editor comment: See also Chapter 5 on jMAF software in the first volume of this book.

Table 12.1 Attribute specification for adoption intention of micro-blog analysis

Attribute Name	Attribute Values	Preference
Condition attributes		
Gender (a_1)	1: Female; 2: Male	Non-ordered
Daily internet hour usage (a_2)	1: <2 ; 2: 2-4; 3: >5	Non-ordered
Past Facebook experience (a_3)	1: Yes; 2: No	Non-ordered
Word-of-mouth recommendations (a_4)	1: Not very important; 2: Not important; 3: Neutral; 4: Important; 5: Very important	Ordered
Advertising recommendations (a_5)	1: Not very important; 2: Not important; 3: Neutral; 4: Important; 5: Very important	Ordered
Expert recommendations (a_6)	1: Not very important; 2: Not important; 3: Neutral; 4: Important; 5: Very important	Ordered
Decision attributes		
Adoption intention (d_1)	1: Strong disagree; 2: Disagree; 3: Neutral; 4: Agree; 5: Strong agree	Ordered

intentions. The results of the accuracy and quality of approximation for the four decision classes are shown in Table 12.2. We classified our samples into four classes: "at most 3" (corresponds to having no intention to adopt a micro-blog), "at most 4" (corresponds to having no or weak intention to adopt a micro-blog), "at least 4" (corresponds to having weak or more intention to adopt a micro-blog) and "at least 5" (corresponds to having strong intention to adopt a micro-blog). The accuracy of the classification of the four decision classes is 0.94, 0.98, 0.99, and 0.96 respectively, so most samples of the data were correctly classified. The overall quality of approximation is 0.97. In this manner, the results represent that the six condition attributes play an important role in determining intentions to adopt a micro-blog.

In this study, the "minimum cover rules" approach, involving a set that does not contain redundant rules, generated rules. Through the DRSA analysis, a total of 37 rules were obtained from the coded informational table. To interpret the rules, this study set up a threshold value of 100 for each decision class; thus, the reduced rule set only considered 8 rules, as illustrated in Table 12.3. These rules have been selected with attention to categorization in terms of correctly classified objects and in terms of recommendation sources and their intention understanding.

The antecedents of the "at least 5" and "at least 4" classes of rules explain which attributes micro-blog-related organizations need to attract, and the "at most 3" and "at most 4" classes of rules tell micro-blog-related organizations what attributes they should avoid. Therefore, as Table 12.3 shows, some variables have a higher degree of dependence and may impact user intentions to adopt a

Table 12.2 Accuracy of classification and quality of classification

Behavioral intentions	Numbers of objects	Lower approximation	Upper approximation	Accuracy of classification	Quality of classification
d_1	-	-	-	-	0.97
at most 3 ($d_1 \leq 3$)	282	273	291	0.94	-
at most 4 ($d_1 \leq 4$)	566	559	573	0.98	-
at least 4 ($d_1 \geq 4$)	826	820	826	0.99	-
at least 5 ($d_1 \geq 5$)	542	531	553	0.96	-

Table 12.3 Rules on the intention to adopt a micro-blog

Rules	Decision	Support	Certainty	Strength	Coverage
The user intends to adopt a micro-blog ($d_1 \geq 5$) or ($d_1 \geq 4$)					
1 ($a_4 \geq 5$) & ($a_6 \geq 5$)	$d_1 \geq 5$	430	1	0.39	0.79
2 ($a_3 \geq 3$) & ($a_4 \geq 5$)	$d_1 \geq 5$	332	1	0.30	0.61
3 ($a_3 = 1$) & ($a_4 \geq 4$)	$d_1 \geq 4$	174	1	0.16	0.61
4 ($a_1 = 1$) & ($a_4 \geq 4$)& ($a_6 \geq 4$)	$d_1 \geq 4$	147	1	0.13	0.52
5 ($a_1 = 2$) & ($a_5 \geq 4$)	$d_1 \geq 4$	129	1	0.12	0.45
The user has no or weak intention to adopt a micro-blog (($d_1 \leq 3$) or $d_1 \leq 3$)					
6 ($a_4 \leq 3$)	$d_1 \leq 4$	416	1	0.38	0.73
7 ($a_2 = 2$) & ($a_5 \leq 3$)	$d_1 \leq 3$	174	1	0.16	0.62
8 ($a_2 \leq 2$) & ($a_5 \leq 3$) & ($a_6 \leq 3$)	$d_1 \leq 3$	163	1	0.15	0.58

$a_1 = 1$ means female; $a_1 = 2$ means male; $a_2 \leq 2$ means that daily Internet per hour usage is less than two hours; $a_2 \geq 3$ means that daily Internet per hour usage is more than five hours; $a_3 = 1$ means that respondent has past Facebook experience; $a_4 \geq 5$ means that WOM recommendation sources are very important; $a_4 \geq 4$ means that WOM recommendation sources are important or very important; $a_4 \leq 3$ means that WOM recommendation sources are neutral or less; $a_5 \geq 4$ means that advertising recommendation sources are important or very important; $a_5 \leq 3$ means that advertising recommendation sources are neutral or less; $a_6 \geq 4$ means that expert recommendation sources are very important; $a_6 \leq 3$ means that expert recommendation sources are neutral or less; $d_1 \geq 4$ means having weak or strong intentions to adopt a micro-blog; $d_1 \leq 3$ means having no intentions to adopt a micro-blog.

micro-blog more than others. Given these classes of rules, micro-blog-related organizations could formulate marketing strategies based on "at least 5" and "at least 4" classes (Rules 1 to 5). If micro-blog-related organizations want to achieve an overall rating of 4 or better, they must achieve a rating of 4 or better from recommendation sources. In addition, the support of a rule corresponds to the number of surveyed students supporting that rule. Because Rule 1 has higher support than Rule 2,

micro-blog-related organizations should work to satisfy the conditions in Rule 1 first before working on Rule 2.

12.4.3 The Flow Network Graph

The rules in Table 12.3 can be translated into one decision algorithm represented by the decision flow network graph shown in Figures 12.1 and 12.2. To simplify the flow network graph, not only normalized supports are shown in the figure, and certainty, strength and coverage are omitted. The flow network graphs represent decision algorithms, and each branch describes a decision rule. Flow network graphs also can represent the causal-and-effect relationships among the recommendation sources and personal variables of the intention to adopt a micro-blog. Thus, this study used the DRSA decision rule sets in the algorithms for diagnosing and extracting recommendation source decisions to increase the diagnostic performance and provide useful information for such algorithms. The analyses also provide the possible path and useful information regarding the effect of recommendation sources on the degree of dependencies. Therefore, these flow networks and decision algorithms are valuable for identifying whether there are possible paths and practices that ensure an appropriate set of decision rules for recommendation sources.

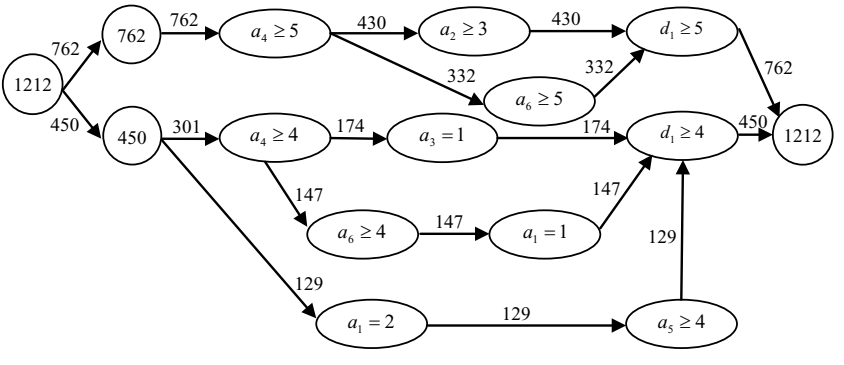


Fig. 12.1 Decision flow graph and rule-set of $d_1 \geq 5$ and $d_1 \geq 4$

$a_1 = 1$ means female; $a_1 = 2$ means male; $a_2 \geq 3$ means that daily Internet per hour usage is more than five hours; $a_3 = 1$ means that respondent has past Facebook experience; $a_4 \geq 5$ means that WOM recommendation sources are very important; $a_4 \geq 4$ means that WOM recommendation sources are important or very important; $a_5 \geq 4$ means that advertising recommendation sources are important or very important; $a_6 \geq 4$ means that expert recommendation sources are very important; $d_1 \geq 4$ means having weak or strong intentions to adopt a micro-blog.

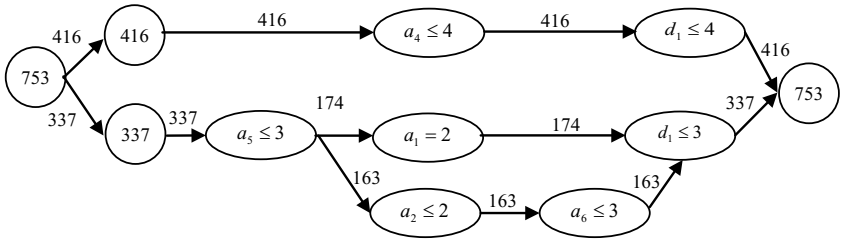


Fig. 12.2 Decision flow graph and rule-set of $d_1 \geq 4$ and $d_1 \geq 3$

$a_1 = 2$ means male; $a_2 \leq 2$ means that daily Internet per hour usage is more than two hours; $a_3 = 1$ means that respondent has past use Facebook experience; $a_4 \leq 3$ means that WOM recommendation sources are neutral or less; $a_5 \leq 3$ means that advertising recommendation sources are neutral or less; $a_6 \leq 3$ means that expert recommendation sources re neutral or less; $d_1 \leq 4$ means having no or weak intentions to adopt a micro-blog; $d_1 \leq 3$ means having no intentions to adopt a micro-blog.

12.4.4 Discussions and Managerial Implications of Research Findings

This investigation examined how personal variables and recommendation sources influence user intentions to adopt a micro-blog. The "at least 5" and "at least 4" classes correspond to user behavioral intentions for e-books in academic libraries, and the antecedents to these classes of rules tell academic libraries what they should consider. The analytical results show that users who trust recommendation sources that vouch for micro-blogs are more likely to adopt micro-blogs and that WOM recommendation sources and expert recommendation sources play a more important role than advertising recommendation sources in determining the perception and intentions regarding the adoption of micro-blogs from Rule 1. This result agrees with Gilly et al. (1998), who indicated that the reliability of the level of expert opinions was lower than that of WOM recommendation sources in the decision-making process. In addition, previous studies pertaining to the feeling and the efficacy of communication on the Internet have confirmed the effect of gender differences on Internet usage. Rule 2 indicates that users who conduct more daily Internet usage rely on recommendation sources to adopt micro-blogs, especially from WOM recommendation sources and advertising recommendation sources. In addition, users who conduct more daily Internet usage rely on recommendation sources to adopt micro-blogs, especially from WOM recommendation sources from Rule 3. Finally, this study, from Rule 4 and Rule 5, also finds that females who trust WOM recommendation sources and expert recommendation sources and males who trust expert recommendation sources are more likely to adopt micro-blogs if those recommendation sources give positive feedback.

The "at most 3" and "at most 4" classes correspond to users who have no or little intention to adopt a micro-blog. The analytical results (Rule 6 to Rule 8) showed that the adoption intentions of users who have no confidence in recommendation sources would decrease, especially with respect to WOM recommendation sources, as would those users who have fewer hours of daily Internet use. Furthermore, males who don't trust advertising recommendation sources are more likely to decrease the intention to adopt a micro-blog.

The results of this study have implications for industry decision-makers. As mentioned previously, users can obtain effective recommendation sources from different sources. One implication is that the market may use strategies that combine WOM recommendation sources and expert recommendation sources to promote micro-blog usage. Platform providers can design recommendation activities to reward users who recommend e-books to others. In addition, platform providers should create an exclusive discussion forum to allow greater exposure to information. Such an approach will increase the desire to adopt micro-blogs and navigate topics users wish to investigate more easily. In addition, platform providers can use other famous platforms to allow other micro-blog information exposure, especially considering the ideas and suggestions of experts, which are useful in influencing usage intentions and stimulating potential users.

Another implication is that different types of recommendations attract different types of users. There are differences in how recommendation sources impact the two genders, so platform providers can apply different recommendation strategies, such as targeting mass media (*e.g.*, a news report) to male users and alternative media (*e.g.*, a discussion forum) to female users. Combining the DRSA with flow network graph-applied influence diagrams will help decision makers and managers by establishing a set of adoption intention decision rules that describe appropriate decision paths and directions, which is a significant contribution.

12.5 Conclusions and Remarks

Past research has not widely combined DRSA and a flow network graph to predict the intention to use micro-blogs. Thus, this research presents a new approach to identifying micro-blog decision rules that infer the antecedents of the intention to adopt micro-blogs under the influence of different recommendation sources. The advantages of combining the DRSA with flow network graphs for recommendation sources are summarized by two points. The first point is that the platform provider can discover hidden information regarding recommendation sources and predict and act on the new information arising from the scale information. The second point is that such a model will be welcomed for its ability to capture the effect of recommendation sources on behavioral intentions and the ability to turn that information into useful marketing strategies, eventually improving user adoption intentions for micro-blogs.

To sum up, this study draws several implications. Firstly, a better understanding of the relationship between recommendation sources and intentions to adopting micro-blogs may further develop marketing strategies. For instance, paltform providers should strengthen micro-blog advantages to receive positive recommendations if users follow all of the recommendations of a source. Understanding the characteristics of users is also important. Collecting and analyzing the background information of users, such as gender or daily Internet usage, can provide abundant information that decision-makers can use to characterize customers for strategic planning and decision-making purposes, thus increasing usage of certain products.

References

1. Ajzen, I.: Attitudes, Personality, and Behavior. Dorsey Press, Chicago (1988)
2. Ajzen, I., Fishbein, M.: Understanding Attitudes and Predicting Social Behavior. Prentice Hall, Englewood Cliffs (1980)
3. Akhter, S.H.: Digital divide and purchase intention: Why demographic psychology matters. Journal of Economic Psychology 24(3), 321–327 (2003)
4. Andreasen, A.R.: Attitudes and customer behavior: A decision model. In: Kassarjian, H.H., Robertson, T.S. (eds.) Perspectives in Consumer Behavior, pp. 498–510. Scott, Foresman and Company, Glenview (1968)
5. Awareness: Enterprise Social Media: Trends and Best Practices in Adopting Web 2.0 (2008), http://www.awarenessnetworks.com/resources/resources-whitepapers.asp
6. Bagozzi, R.P., Baumgartner, H., Yi, Y.: State versus action orientation and the theory of reasoned action: An application to coupon usage. Journal of Consumer Research 18(4), 505–518 (1992)
7. Błaszczyński, J., Greco, S., Słowiński, R.: Multi-criteria classification - A new scheme for application of dominance-based decision rules. European Journal of Operational Research 181(3), 1030–1044 (2007)
8. Carchiolo, V., Malgeri, M., Mangioni, G., Nicosia, V.: Emerging structures of P2P networks induced by social relationships. Computer Communications 31(3), 620–628 (2008)
9. Chan, C.-C., Tzeng, G.-H.: Dominance-Based Rough Sets Using Indexed Blocks as Granules. In: Wang, G., Li, T., Grzymala-Busse, J.W., Miao, D., Skowron, A., Yao, Y. (eds.) RSKT 2008. LNCS (LNAI), vol. 5009, pp. 244–251. Springer, Heidelberg (2008)
10. Chau, P.Y.K., Hu, P.J.H.: Investigating healthcare professionals' decisions to accept telemedicine technology: An empirical test of competing theories. Information & Management 39(4), 297–311 (2002)
11. Cheong, M., Lee, V.: Integrating Web-based intelligence retrieval and decision-making from the Twitter trends knowledge base. In: Proceeding of the 2nd ACM Workshop on Social Web Search and Mining, pp. 1–8 (2009)
12. Cox, D.F.: Risk taking and information handling in consumer behavior. In: Cox, D.F. (ed.) Risk Taking and Information Handling in Consumer Behavior, pp. 604–639. Boston University Pres, Boston (1967)
13. Crisci, R., Kassinove, H.: Effects of perceived expertise, strength of advice, and environmental setting on parental compliance. The Journal of Social Psychology 89(2), 245–250 (1973)
14. Crow, S.M., Fok, L.Y., Hartman, S.J., Payne, D.M.: Gender and values: What is the impact on decision making? Sex Roles 25(3-4), 255–268 (1991)

15. Davis, F.D.: Perceived usefulness, Perceived ease of use, and user acceptance of information technology. MIS Quarterly 13(3), 319–340 (1989)
16. Fang, S.K., Shyng, J.Y., Lee, W.S., Tzeng, G.H.: Combined data mining techniques for exploring the preference of customers between financial companies and agents based on TCA. Knowledge-Based Systems 27(1), 137–151 (2012)
17. Ford, L.R., Fulkerson, D.R.: Flows in Networks. Princeton University Press, Princeton (1962)
18. Gefen, D., Straub, D.W.: Gender differences in the perception and use of e-mail: An extension to the technology acceptance model. MIS Quarterly 21(4), 389–400 (1997)
19. Gilly, C.M., Graham, J.L., Wolfinbarger, M.R., Yale, L.J.: A dyadic study of interpersonal information search. Journal of the Academy of Marketing Science 2(2), 83–100 (1998)
20. Greco, S., Matarazzo, B., Słowiński, R.: A new rough set approach to evaluation of bankruptcy risk. In: Zopounidis, C. (ed.) Operational Tools in the Management of Financial Risk, pp. 121–136. Kluwer Academic Publishers, Boston (1998)
21. Greco, S., Matarazzo, B., Słowiński, R.: Rough set theory for multicriteria decision analysis. European Journal of Operation Research 129(1), 1–47 (2001)
22. Greco, S., Matarazzo, B., Słowiński, R.: Dominance-Based Rough Set Approach as a Proper Way of Handling Graduality in Rough Set Theory. In: Peters, J.F., Skowron, A., Marek, V.W., Orłowska, E., Słowiński, R., Ziarko, W.P. (eds.) Transactions on Rough Sets VII. LNCS, vol. 4400, pp. 36–52. Springer, Heidelberg (2007)
23. Günther, O., Krasnova, H., Riehle, D., Schöndienst, V.: Modeling microblogging adoption in the enterprise. In: Proceedings of the 15th Americas Conference on Information Systems, San Francisco, CA, USA (2009)
24. Häubl, G., Trifts, V.: Consumer decision making in online shopping environments: the effects of interactive decision aids. Marketing Science 19(1), 4–21 (2000)
25. Hoadley, C.M., Xu, H., Lee, J.J., Rosson, M.B.: Privacy as information access and illusory control: The case of the Facebook news feed privacy outcry. Electronic Commerce Research and Applications 9(1), 50–60 (2010)
26. Janda, S.: Does gender moderate the effect of online concerns on purchase likelihood? Journal of Internet Commerce 7(3), 339–358 (2008)
27. Java, A., Song, X., Finin, T., Tseng, B.: Why We Twitter: An Analysis of a Microblogging Community. In: Zhang, H., Spiliopoulou, M., Mobasher, B., Giles, C.L., McCallum, A., Nasraoui, O., Srivastava, J., Yen, J. (eds.) WebKDD 2007. LNCS, vol. 5439, pp. 118–138. Springer, Heidelberg (2009)
28. Kim, S.S., Malhotra, N.K., Narasimhan, S.: Two competing perspectives on automatics use: A theoretical and empirical comparison. Information Systems Research 16(4), 418–432 (2005)
29. Lin, C.S., Tzeng, G.H., Yang-Chieh Chin, Y.H.: Combined rough set theory and flow graph to predict customer churn in credit card accounts. Expert Systems with Applications 38(1), 8–15 (2011)
30. Liou, J.H., Tzeng, G.H.: A dominance-based rough set approach to customer behavior in the airline market. Information Sciences 180(11), 2230–2238 (2010)
31. Liou, J.H., Yen, L., Tzeng, G.H.: Using decision-rules to achieve mass customization of airline service. European Journal of Operation Research 205(3), 680–686 (2010)
32. Mark, B., Munakata, T.: Rule extraction from expert heuristics: A comparative study of rough sets with neural networks and ID3. European Journal Operational Research 136(1), 212–229 (2002)
33. Murray, K.B.: A test of services marketing theory: Consumer information acquisition activities. Journal of Marketing 55(1), 10–25 (1991)
34. Ou Yang, Y.P., Shieh, H.M., Tzeng, G.H., Yen, L., Chan, C.C.: Combined rough sets with flow graph and formal concept analysis for business aviation decision-making. Journal of Intelligent Information Systems 36(3), 347–366 (2011)

35. Park, D.H., Lee, J., Han, I.: The effect of on-line consumer reviews on consumer purchasing intention: The moderating role of involvement. International Journal of Electronic Commerce 11(4), 125–148 (2007)
36. Pawlak, Z.: Rough sets. International Journal of Computer and Information Sciences 11(1), 341–356 (1982)
37. Pawlak, Z.: Rough sets, decision algorithms and Bayes' theory. European Journal of Operational Research 136(1), 181–189 (2002)
38. Pawlak, Z.: Decision rules and flow networks. European Journal of Operation research 154(1), 184–190 (2004)
39. Pawlak, Z.: Flow graphs and intelligent data analysis. Fundamenta Informaticae 64(1), 369–377 (2005)
40. Richins, M.L., Root-Shaffer, T.: The Role of Involvement and Opinion Leadership in Consumer Word-of-Mouth: An Implicit Model Made Explicit. Advances in Consumer Research 15(1), 32–36 (1988)
41. Rogers, E.M.: The 'Critical Mass' in the diffusion of interactive technologies in organizations. In: Kraemer, K.L. (ed.) Information Systems Research Challenge.: Survey Research Methods. Harvard Business School Research Colloquium, vol. 3, pp. 245–264. Harvard Business School, Boston (1991)
42. Rogers, E.M.: Diffusion of Innovations, 5th edn. Free Press, New York (1995)
43. Sankey, D.: Networking sites used for background checks, `http://www2.canada.com/components/print.aspx?id=ba0dcc0d-f47d-49c9-add4-95d90fd969ab`
44. Shyng, J.Y., Shieh, H.M., Tzeng, G.H.: Compactness rate as a rule selection index based on rough set theory to improve data analysis for personal investment portfolios. Applied Soft Computing 11(4), 3671–3679 (2011)
45. Sledgianowski, D., Kulviwat, S.: Social network sites: antecedents of user adoption and usage. In: Proceedings of the Fourteenth Americas Conference on Information Systems (AMCIS), Toronto, ON, Canada, pp. 1–10 (2008)
46. Słowiński, R.: New Applications and Theoretical Foundations of the Dominance-based Rough Set Approach. In: Szczuka, M., Kryszkiewicz, M., Ramanna, S., Jensen, R., Hu, Q. (eds.) RSCTC 2010. LNCS (LNAI), vol. 6086, pp. 2–3. Springer, Heidelberg (2010)
47. Słowiński, R.: The international summer school om MCDM 2006. Class note. Kainan University, Taiwan (2006), Software available on line, `http://idss.cs.put.poznan.pl/site/software.html`
48. Słowiński, R., Greco, S., Matarazzo, B.: Rough sets in decision making. In: Meyers, R.A. (ed.) Encyclopedia of Complexity and Systems Science, pp. 7753–7786. Springer, New York (2009)
49. Smith, D., Menon, S., Sivakumar, K.: Online peer and editorial recommendation, trust, and choice in virtual markets. Journal of Interactive Marketing 19(3), 15–37 (2005)
50. Sung, S.K., Malhotra, N.K., Narasimhan, S.: Two competing perspectives on automatics use: a theorectical and empirical comparison. Information Systems Research 17(2), 418–432 (2005)
51. Wang, C.H., Chin, Y.C., Tzeng, G.H.: Mining the R&D innovation performance processes for high-tech firms based on rough set theory. Technovation 30(7-8), 447–458 (2010)
52. Yi, Y., Wu, Z., Tung, L.L.: How individual differences influence technology usage behavior? Toward an integrated framework. Journal of Computer Information Systems 46(2), 52–63 (2006)

Chapter 13
Providing Feedback in Ukrainian Sign Language Tutoring Software

M.V. Davydov, I.V. Nikolski, V.V. Pasichnyk, O.V. Hodych, and Y.M. Shcherbyna

Abstract. This chapter focuses on video recognition methods implemented as part of the Ukrainian Sign Language Tutoring Software. At the present time the sign language training software can easily verify how users understand signs and sentences. However, currently there is no good solution to the problem of verifying how the person reproduces signs due to a large variety of training conditions and human specifics. The new approach to user interaction with the Sign Tutoring Software is proposed as well as new algorithms implementing it. The use of body posture recognition methods allows interaction with users during learning of signs and the verification process. The software provides a feedback to the user by capturing person's gestures via a web camera improving the success of training. A single web camera is used without utilising depth sensors. The process of human posture reconstruction from a web camera in real-time involves background modelling, image segmentation and machine learning methods. The success rate of 91.7% has been achieved for sign recognition on the test set of 85 signs.

Keywords: Sign language, image segmentation, interactive tutoring, tutoring software, neural-networks.

M.V. Davydov, I.V. Nikolski, V.V. Pasichnyk
L'viv Polytechnic National University
12 Bandery St., L'viv, Ukraine
e-mail: maks.davydov@gmail.com

O.V. Hodych, Y.M. Shcherbyna
Ivan Franko National University of L'viv
1 Universytetska St., L'viv, Ukraine
e-mail: oles.hodych@gmail.com

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 241–261.
springerlink.com

13.1 Introduction

Sign languages are based on hand signs, lip patterns and body language instead of sounds to convey meaning. The development of sign languages is generally associated with deaf communities, which may include hearing or speech impaired individuals, their families and interpreters. The only currently viable ways to enable interactive communication between hearing impaired and not impaired people is to use services provided by interpreters (specially trained individuals or software applications), or to learn a sign language.

A manual communication has developed in situations where speech is not practical. For instance, scuba diving or loud work places such as stock exchange. In such cases learning a sign language is the most effective solution.

Teaching a sign language is a challenging problem, which often requires a professional tutor, presence of students in a class room and many hours of practice. There is a number of commercial software packages and research projects directed at developing software applications for sign language interpretation. The majority of such software is directed at interpreting a spoken to sign language, which covers all major cases where interpretation is required for deaf people (*e.g.* conventions, television). For example, researchers at IBM have developed a prototype system, called SiSi (say it, sign it), which offers some novel approaches by utilising avatars to communicate speech into sign language [41]. Some other applications, such as iCommunicator [42], are based on a database of sign language videos and provide means to adaptively adjust to user's speech. The are several online sign language tutoring software packages such as Handspeek ASL Dictionary [1] that can demonstrate signs, but cannot provide any feedback to the user. Some systems that provide feedback [2] require wearing coloured gloves for better hand shape processing. Other approaches to capture data for sign language feedback include the use of Kinect hardware or 3DV Systems ZCam [5].

The problem of tracking head and hands in sign language recognition was studied by Jörg Zieren and Karl-Friedrich Kraiss [39]. The tracking makes the use of several methods of ambiguity prevention such as Bayesian trust network, expectation maximisation algorithm, CAMSHIFT algorithm [4]. This approach allows tracking of hands in motion even if they overlap the face. For the dictionary of 152 signs for one user 2.4% of Word Error Rate (WER) was obtained.

The accuracy of face localisation in case of face/hand overlapping was improved in the research by Suat Akyol and Jörg Zieren [1]. In this study the Active Shape Model was applied in order to locate the face. The research conducted by Jörg Zieren and others [40] resulted in 0.7% of WER for the test based on 232 gestures of the same person segmented manually in ideal conditions. WER of 55.9% was achieved for the user who did not participate in the system training based on gestures by six speakers under controlled light conditions.

[1] www.handspeak.com

In the study by Morteza Zahedi [38] for the video camera-based sign recognition utilised the hidden Markov model and two image comparison methods (the comparison method with deformations and the tangent distance method). Based on BOSTON50 signs the 17.2% of WER was reached. This method was improved by Philippe Dreuw and others [14] by introducing the trigram-based linguistic model, motion path determination of the main hand movement and the reduction in the number of characteristics using the principle component analysis method. The 17.9% of WER was reached for the RWTH-Boston-104 signs. The application of the special algorithm of hand shape tracking [15], the model for image creation and the sentence model for the sign recognition in the study [16] resulted in 11.24% of WER for the RWTH-Boston-104 signs. This has been the best result so far.

In the research conducted by authors [10], [11], [12], [9], [13] achieved 91.7% sign recognition rate on the test set of 85 signs.

This chapter provides a review of the developed by authors methodology for sign language recognition with an emphasis on video recognition methods used for developing the Ukrainian Sign Language Tutoring Software. In relation to our previous publications, this is the first time where the problem of providing feedback as part of the developed software is presented.

13.2 Problem Formulation

We are working on a generic solution, which could be available to every Ukrainian School for hear-impaired students. The requirement for such solution is to be inexpensive. Thus, it has been decided to utilise a commodity hardware — PC and a web-camera as a video input device. One of the main goals formulated for the software it to enable a tutor-like experience, where students would be provided with a feedback about the correctness of the sign they're trying to reproduce. In order to reflect this the term *tutoring software* (or simply *tutor*) is used instead of *training software*.

The developed sign language tutoring software incorporates solutions to the following problems:

- system setup in a new environment;
- tracking position for left and right hands;
- hand shape matching;
- providing feedback during tutoring.

The Ukrainian Sign Language Video Dictionary, which was originally developed by Ivanusheva and Zueva, has been used to collect the necessary data. This dictionary consists of three continuous DVD video files, which have indexed in order to extract separate signs and phrases. For the purpose of providing feedback, the tutoring software needs to know the shape and the location of hands and the face in the video

tutorials. Although, it is possible to pre-process the video data, the developed software processes video streams from both the video tutorial being played back and the web-camera capturing the student simultaneously in real-time during the tutoring process. This approach provides an easy way to extend the database of videos featuring additional signs without the need for special preparation.

The following sections provide more details on how each of the specified above problems are addressed in the developed tutoring software.

13.3 System Setup in the New Environment

The initial stage of using the developed tutoring software requires tuning of the system parameters to the conditions of the new environment. The feedforward neural network classifier and Inverse Phong Lighting Model [36] are utilised to identify the face and hands from the video. Both classifiers require a training set for adjustment to the new environment.

The problem of hand classifier setup can be solved in different ways. One of the commonly used approaches requires user to put the hand in a particular place of the web-camera frame. The approached utilised in this research requires a user to move the hand in front of and in close proximity to the web-camera in order to make it the largest object captured by the camera. During this process a method based on the Self-Organising Map (SOM) is applied to separate the hand from the background [24, 25].

13.3.1 SOM-Based Image Segmentation

It is well known that SOM operates based on the principles of the brain. One of the key SOM features is the preservation of the topological order of the input space during the learning process. Some relatively recent research of the human brain has revealed that the response signals are obtained in the same topological order on the cortex in which they were received at the sensory organs [32]. One of such sensory organs are eyes, thus making the choice of SOM for analysing the visual information one of the most natural.

Colour is the brain's reaction to specific visual stimulus. Therefore, in order to train SOM for it to reflect the topological order of the image perceived by a human eye, it is necessary to choose the colour space, which closely models the way sensors obtain the visual information. The eye's retina samples colours using only three broad bands, which roughly correspond to red, green and blue light [17]. These signals are combined by the brain providing several different colour sensations, which are defined by the CIE (Commission Internationale de l'Eclairage (French), International Commission on Illumination) [28]: Brightness, Hue and Colourfulness. The CIE commission defined a system, which classifies colour according to the human

visual system, forming the tri-chromatic theory describing the way red, green and blue lights can match any visible colour based on the eye's use of three colour sensors.

The colour space is the method, which defines how colour can be specified, created and visualised. As can be deduced from the above, most colour spaces are three-dimensional. There are more than one colour space, some of which are more suitable for certain applications than others. Some colour spaces are perceptually linear, which means that an n-unit change in stimulus results in the same change in perception no matter where in the space this change is applied [17]. The feature of linear perception allows the colour space to closely model the human visual system. Unfortunately, the most popular colour spaces currently used in image formats are perceptually non-linear. For example, BMP and PNG utilise RGB[2] colour space, JPEG utilises YCbCr, which is a transformation from RGB, HSL[3] is another popular space, which is also based on RGB.

The CIE based colour spaces, such as CIELuv and CIELab, are nearly perceptually linear [17], and thus are more suitable for the use with SOM. The CIEXYZ space devises a device-independent colour space, where each visible colour has non-negative coordinates X, Y and Z [27]. The CIELab is a nonlinear transformation of XYZ onto coordinates $L*, a*, b*$ [27].

The image format used in our research is uncompressed 24-bit BMP (8 bit per channel), which utilises the RGB colour space. In order to convert vectors $(r,g,b) \in RGB$ into $(L*, a*, b*) \in CIELab$ it is necessary to follow an intermediate transformation via the CIE XYZ colour space. These transformations are described in details in [26] and [27]. Application of the two-step transformation to each pixel of the original image in RGB space produces a transformed image in CIELab space used for further processing.

It is important to note that when using SOM it is common to utilise Euclidean metric for calculation of distances during the learning process [32][4]. Conveniently, in CIELab space the colour difference is defined as Euclidean distance [27].

Instead of using every image pixel for the SOM training process, the following approach was employed to reduce the number of data samples in the training dataset.

The basic idea is to split an image into equal segments $n \times n$ pixels. Then for each such segment find two the most diverged pixels and add them to the training dataset. Finding the two most diverged pixels is done in terms of the distance applicable to the colour space used for image representation. Due to the fact that each pixel is a three dimensional vector, each segment is a matrix of vector values. For example, below is an image A of 4×4 pixels in size represented in the CIELab space, and split into four segments 2×2 pixels each.

[2] Uncompressed BMP files, and many other bitmap file formats, utilise a colour depth of 1, 4, 8, 16, 24, or 32 bits for storing image pixels.

[3] Alternative names include HSI, HSV, HCI, HVC, TSD etc. [17]

[4] The selection of the distance formula depends on the properties of the input space, and the use of Euclidean metric is not mandatory.

$$A = \left(\begin{array}{cc|cc} (L_1^1,a_1^1,b_1^1)^T & (L_2^1,a_2^1,b_2^1)^T & (L_3^1,a_3^1,b_3^1)^T & (L_4^1,a_4^1,b_4^1)^T \\ (L_1^2,a_1^2,b_1^2)^T & (L_2^2,a_2^2,b_2^2)^T & (L_3^2,a_3^2,b_3^2)^T & (L_4^2,a_4^2,b_4^2)^T \\ \hline (L_1^3,a_1^3,b_1^3)^T & (L_2^3,a_2^3,b_2^3)^T & (L_3^3,a_3^3,b_3^3)^T & (L_4^3,a_4^3,b_4^3)^T \\ (L_1^4,a_1^4,b_1^4)^T & (L_2^4,a_2^4,b_2^4)^T & (L_3^4,a_3^4,b_3^4)^T & (L_4^4,a_4^4,b_4^4)^T \end{array}\right)$$

Thus, the first segment is:

$$S_1 = \left(\begin{array}{cc} (L_1^1,a_1^1,b_1^1)^T & (L_2^1,a_2^1,b_2^1)^T \\ (L_1^2,a_1^2,b_1^2)^T & (L_2^2,a_2^2,b_2^2)^T \end{array}\right)$$

The above approach can be summarised as the following algorithm. Let n denote the size of segments used for image splitting, the value of which is assigned based on the image size. T — the training set, which is populated with data by the algorithm. Let's also denote jth pixel in segment S_i as $S_i(j)$. Further in the text both terms *pixel* and *vector* are used interchangeably.

Algorithm 13.1. Training dataset composition

Initialisation. Split image into segments of $n \times n$ pixels; $N > 0$ – number of segments; $T \leftarrow \varnothing$; $i \leftarrow 1$.

1. Find two the most diverged pixels $p' \in S_i$ and $p'' \in S_i$ using Euclidean distance.
   - 1.1 $max \leftarrow -\infty$, $j \leftarrow 1$
   - 1.2 $k \leftarrow j+1$
   - 1.3 Calculate distance between pixels $S_i(j)$ and $S_i(k)$: $dist \leftarrow \|S_i(j) - S_i(k)\|$
   - 1.4 If $dist > max$ then $p' \leftarrow S_i(j)$, $p'' \leftarrow S_i(k)$ and $max \leftarrow dist$
   - 1.5 If $k < n \times n$ then $k \leftarrow k+1$ and return to step 1.3
   - 1.6 If $j < n \times n - 1$ then $j \leftarrow j+1$ and return to step 1.2
2. Add $p' \in S_i$ and $p'' \in S_i$ to the training set: $T \leftarrow T \cup \{p', p''\}$
3. Move to the next segment $i \leftarrow i+1$. If $i \leq N$ then return to step 1, otherwise stop.

The above algorithm provides a way to reduce the training dataset. It is important to note that an excessive reduction could cause omission of significant pixels resulting in poor training. At this stage it is difficult to state what rule can be used to deduce the optimal segment size. The segmentation used for the presented results was obtained though experimentation. However, even applying segmentation 2×2 pixels to an image of 800×600 pixels in size reduces the training dataset from 460000 down to 240000 elements, which in turn enables the use of a smaller lattice and reduces the processing time required for SOM training.

There are several aspects to a successful application of SOM, among which are:

- Self-organisation process, which encompasses a problem of selecting a learning rate and a neighbourhood function.
- The size and structure of the SOM lattice.

In this research the guidelines from [32] and [22] were followed to conduct the self-organisation process. The structure of the SOM lattice may differ in its dimensionality and neighbourhood relation between elements. The use of 2-dimensional lattice with hexagonal neighbourhood relation proved to be the most efficient in our research producing more adequate clustering results comparing to other evaluated configurations.

Once the SOM structure and parameters for self-organisation process are selected, the SOM is trained on the training set T, which is composed for the image to be clustered. The trained SOM is then used for the actual image clustering.

The topology preservation SOM feature is fundamental to the developed image segmentation approach. Its basic underlying principles are:

- Image pixels represented by topologically close SOM elements should belong to the same cluster and therefore segment.
- The colour or marker used for segment representation is irrelevant as long as each segment is associated with a different one.

These two principles suggest that the position of SOM elements in the lattice (i.e. coordinates on the 2D plane) can be used for assigning a marker to a segment represented by any particular element instead of the elements' weight vectors. This way weight vectors are used purely as references from 2D lattice space into 3D colour space, and locations of SOM elements represent the image colour distribution. As the result of a series of conducted experiments the following formulae for calculating an RGB colour marker for each element have produced good results.

$$R_j \leftarrow x_j + y_j \times \lambda; G_j \leftarrow x_j + y_j \times \lambda; B_j \leftarrow x_j + y_j \times \lambda; \tag{13.1}$$

Values x_j and y_j in formula (13.1) are the coordinates of SOM elements in the lattice $j = \overline{1,M}$, where M is the total number of elements. Constant λ should be greater or equal to the diagonal of the SOM lattice. For example, if SOM lattice has a rectangular shape of 16×16 elements then λ could be set to 16. Applying the same formula for R, G, and B components produces a set of grayscale colours. However, each element has its own colour, and one of the important tasks is grouping of elements based on the assigned colours into larger segments. There are several approaches, which are being currently developed to provide automatic grouping of SOM elements into clusters and have shown good results [23, 25]. One of the possible approaches is to apply a threshold to the segmented with SOM image, which requires human interaction in specifying the threshold value. This image segmentation approach can be summarised as the following algorithm.

The following figures depict the original and segmented images corresponding to several frames of the same video, which have been processed using the same trained SOM. The recorded video captured an open palm closing and opening again

Algorithm 13.2. Image segmentation

Initialisation. $p_j = (R_j, G_j, B_j)$ – pixel j; $j = \overline{1,K}$; $K > 0$ – total number of pixels; $j \leftarrow 1$; $i^*(p_j) = (R_{i^*}, G_{i^*}, B_{i^*})$ – a weight vector of the best matching unit (BMU – winning element) for input vector p_j; (x_{i^*}, y_{i^*}) – coordinates of element i^*; choose appropriate values for λ.

1. Find $BMU(p_j)$ for vector p_j in the trained SOM utilising the distance used for training (Euclidean for CIELab).
2. Calculate marker for pixel p_j: $R_j \leftarrow x_{i^*} + y_{i^*} \times \lambda$, $G_j \leftarrow R_j$, $B_j \leftarrow R_j$.
3. Move to the next image pixel: $j \leftarrow j + 1$;
4. If $j \leq K$ return to step 1, otherwise stop.

during a period of several seconds. The recording was done using an ordinary PC web camera capable of 30FPS throughput with a frame size of 800×600 pixels. The background of the captured scene is non-uniform, which increases the complexity of image segmentation.

Figure 13.1 depicts a fully open palm (frame 25), contracted fingers (frame 35) and a fully closed palm (frame 40).

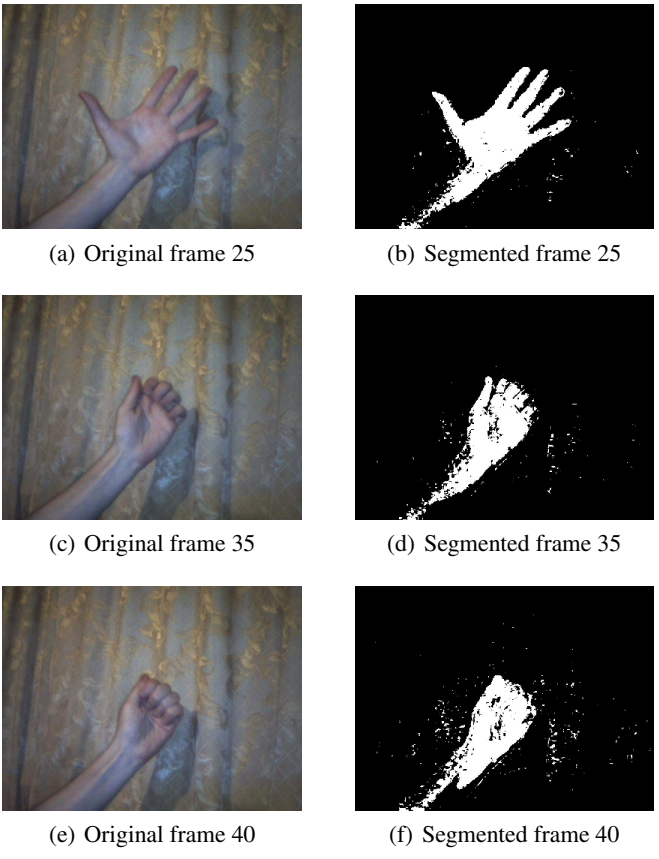
The important aspect of the presented results is the use of SOM trained only on a single frame. This initial frame as well as all subsequent ones have been successfully segmented with clear separation of the human palm from the nonuniform background. The use of only one frame for SOM training allows much faster dynamic image segmentation needed for video, avoiding SOM retraining for every frame.

The trends of the past decade in architecture of the central processing unit show a clear direction towards multi-core processors with the number of cores increasing every eighteen months according to the Moore's law. The shift from fast single-core to slower multi-core CPUs poses a question of scalability for a vast class of computational algorithms including algorithm for image processing.

The developed sing tutoring sofrware provides effective utilisation of multi-core processors, which includes parallelisation of SOM training based on methods for SOM decomposition published in [18, 19, 33, 34]. The flowchart diagram depicted on Fig. 13.2 outlines the SOM training algorithm for multi-core processors, which is incorporated into the tutoring software.

More details on the computational aspects of the developed software is provided in our monograph [25], which also includes the caching algorithm used to speedup segmentation of frames in the video stream eliminating the need to search for BMU for each input pixel.

Once the regions containing hands are identified, the feedforward neural network classifier or Inverse Phong Lighting model classifier is used to segment skin coloured areas in real-time.



(a) Original frame 25 (b) Segmented frame 25

(c) Original frame 35 (d) Segmented frame 35

(e) Original frame 40 (f) Segmented frame 40

Fig. 13.1 Frames 25, 35 and 40

13.3.2 Feedforward Neural Network Classifier

In order to segment skin sections the neural network classifier has been applied to the surrounding area of each pixel in the image matrix $M = \{m_{ij}\}$, $i = 1,2,\ldots,w$, $j = 1,2,\ldots,h$, where w and h are image width and height respectively.

The significant features of the area are selected from the cross-shaped or square surrounding areas (Fig. 13.3). The best result was achieved when pixel are represented in the YCbCr colour space.

The feature extraction can be achieved by utilising rough sets for calculating the upper and lower approximations of the surrounding areas [43]. The developed software utilises neural network-based classifiers with one hidden layer and one or two outputs. For the neural network with a single output the area is considered to be of skin colour if the output value is greater than 0.5.

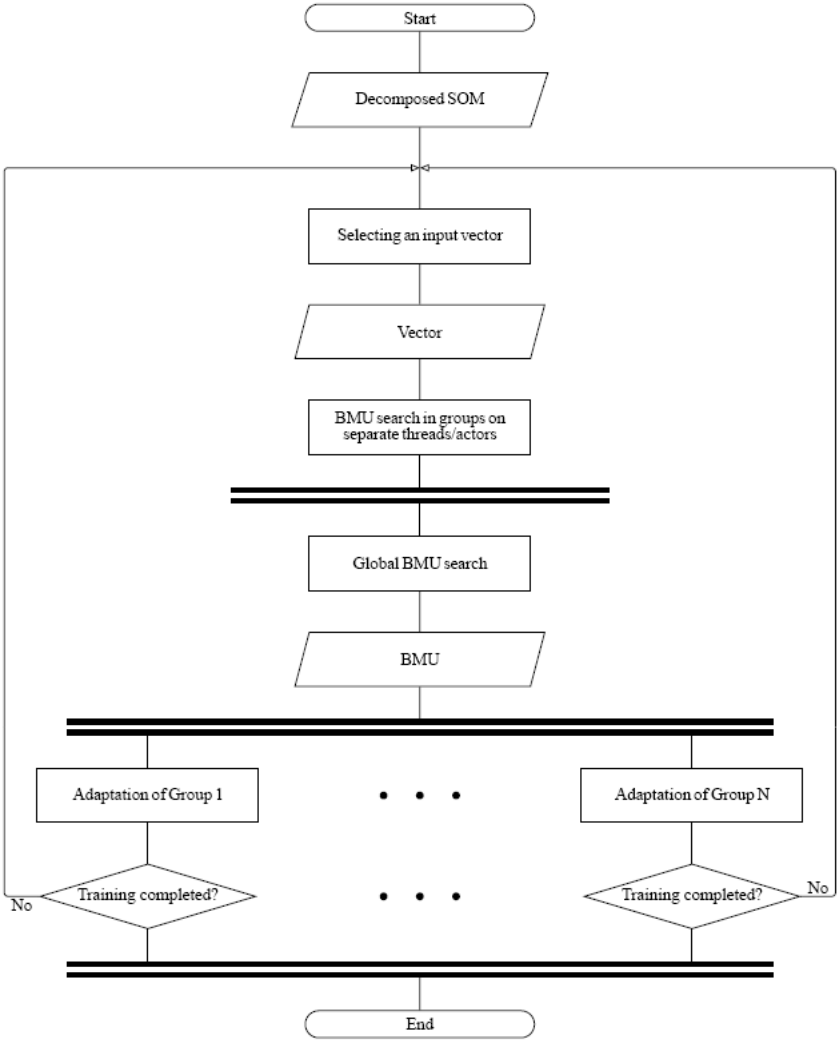


Fig. 13.2 Decomposed SOM training

When the neural network with two outputs is applied the area is considered to be of skin colour if the output value of the first neuron is greater than the output value of the second one. The neural network classifier with two outputs proved to be susceptible to noise. Thus, additional smoothing is required, which can be achieved by averaging each pixel by four neighbouring pixels. The smoothing prevents faulty reactions. The set of positive training samples is formed from the skin area and the

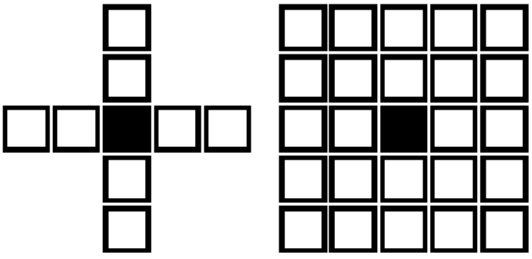


Fig. 13.3 The cross-shaped (a) and square (b) surrounding area for image feature extraction. The image pixel, the surrounding area of which is considered, is marked black. The surrounding area has radius $r = 2$.

negative training samples are formed from the image area not marked as a skin. The complete training set consists of positive and negative training samples. The modification of the back-propagation of error algorithm was developed for training of the feedforward neural network-based classifier [9].

This modification includes the following:

1) training samples are divided into groups according to the value of the required output providing the uniform selection of training samples from each group;
2) the groups for the areas with the worst results are represented by larger sets of training samples;
3) additional random value from the interval $[-\varepsilon, +\varepsilon]$ is added to the neural network weights; the value of this parameter is decreased during the training from the initial value ε_0 to zero;
4) in case where the error cannot be reduced during three iterations of the training algorithm the weight values of the neural network are reset to original;
5) an additional parameter has been introduced to accelerate the scale shift of the neural network weights of the hidden network layer.

The application of the developed method reduces the training time 10x in comparison to the classical back-propagation of error algorithm [9]. This result is significant for the development of interactive software and software trained during the usage. The result of the trained neural network classifier application is depicted on Fig. 13.4.

13.3.3 Inverse Phong Lighting Model Classifier

The skin-colour segmentation can be implemented by using the simplified Inverse Phong lighting model. In this research only the diffuse and ambient components of the Phong lighting model is utilised to estimate the lighting parameters.

(a) Original image (b) Processed image

Fig. 13.4 Skin segmentation by neural network classifier

$$\begin{pmatrix} I_R \\ I_G \\ I_B \end{pmatrix} = \begin{pmatrix} k_{aR} & i_{aR} \\ k_{aG} & i_{aG} \\ k_{aB} & i_{aB} \end{pmatrix} + (\overline{L} \cdot \overline{N}) \begin{pmatrix} k_{dR} & i_{dR} \\ k_{dG} & i_{dG} \\ k_{dB} & i_{dB} \end{pmatrix} + \begin{pmatrix} \varepsilon_R \\ \varepsilon_G \\ \varepsilon_B \end{pmatrix}, \tag{13.2}$$

where $I = (I_R, I_G, I_B)^T$ — the intensity of red, green and blue pixel components, $k_a = (k_{aR}, k_{aG}, k_{aB})^T$ — ambient dispersion coefficients, $i_a = (i_{aR}, i_{aG}, i_{aB})^T$ — the intensity of ambient lighting, $k_d = (k_{dR}, k_{dG}, k_{dB})^T$ — diffuse dispersion coefficients, $i_d = (i_{dR}, i_{dG}, i_{dB})^T$ — the intensity of diffusive lighting, $\overline{L}$ — the unit vector denoting lighting direction, $\overline{N}$ — the unit vector normal to the surface, $\varepsilon = (\varepsilon_R, \varepsilon_G, \varepsilon_B)^T$ — the normally diffused error.

In order to setup the illumination model parameters the hand area is used. It is assumed that the illumination properties are constant for the whole area of the hand.

Equation (13.2) can be rewritten as

$$I = c_a + \alpha c_d + \varepsilon \tag{13.3}$$

where $c_a = (k_{aR} \cdot i_{aR}, k_{aG} \cdot i_{aG}, k_{aB} \cdot i_{aB})^T$ — the permanent intensity of ambient lighting in the frame, $\alpha = (L \cdot N)$, $\alpha \in [0,1]$ — the coefficient determining how surface is oriented in respect to the source of light, $c_d = (k_{dR} \cdot i_{dR}, k_{dG} \cdot i_{dG}, k_{dB} \cdot i_{dB})^T$ — the permanent intensity of the diffuse illumination in the frame, $\varepsilon = (\varepsilon_R, \varepsilon_G, \varepsilon_B)^T$ — the normally distributed error. Parameters of the illumination model c_a and c_d are determined by the method of linear regression based on the training samples.

Thus, having a training set of pixel colours I_i corresponding to the skin colour of hand it is possible to calculate c_a and c_d:

$$I = \frac{1}{n} \sum_{i=1}^{n} I_i \tag{13.4}$$

$$k = norm\left(\sum_{i=1}^{n}(I_{iR} - I_R)\cdot(I_i - I)\right) \tag{13.5}$$

$$v_{min} = \min_i(I_i - I, k) \tag{13.6}$$

$$v_{max} = \max_i(I_i - I, k) \tag{13.7}$$

$$c_a = I + v_{min}\cdot k \tag{13.8}$$

$$c_d = (v_{max} - v_{min})\cdot k \tag{13.9}$$

The pixel of image with intensity of colour I is classified as similar to the colour of skin if there is a value $\alpha \in [0,1]$ such that

$$|I - c_a - \alpha\cdot c_d| \leq \theta \tag{13.10}$$

where θ is the threshold value.

Once lighting model is determined it is used to calculate the truth level t_p of pixel coloured $P = (p_r, p_g, p_b)^T$.

$$t_p = max\left(0, \frac{1}{\theta}\cdot(\theta - |P - c_d\cdot\max(0, \min(1, (P - c_a, c_d)))|)\right) \tag{13.11}$$

For each pixel the degree of truth is calculated utilising formula (13.11) in order to classify the pixel as skin coloured. Formula (13.11) calculates how far is the pixel from the skin diffuse illumination model. The result of this calculation is a grayscale image where lighter pixels represent hand or face segments.

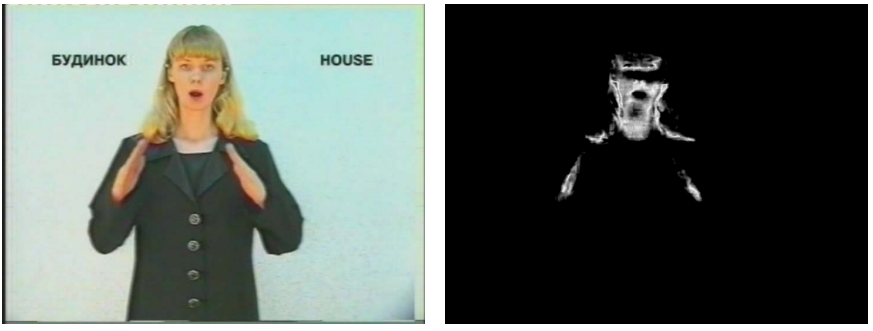
Fig. 13.5 depicts the result of skin segmentation with the simplified Phong illumination model. As can be observed the result achieved for depicted example from the sign dictionary with the illumination model is slightly better for elimination of the hair colour.

13.4 Hands and Face Extraction

The next step is to find centroids of left and right hand. The K-means clustering and connected points labelling algorithm are used for this purpose.

There are several cases that should be considered for the application of K-means: only the head is in the frame, the head and one hand, the head and two well distinguished hands, the head and two hands located close to each other. Three clusters are specified for the initial K-means clustering and then the result is checked as to how well are the clusters connected with each other.

A better result is achieved when using metric space similarity joins of the skin regions [29] by providing minimal distance d between the clusters and minimal

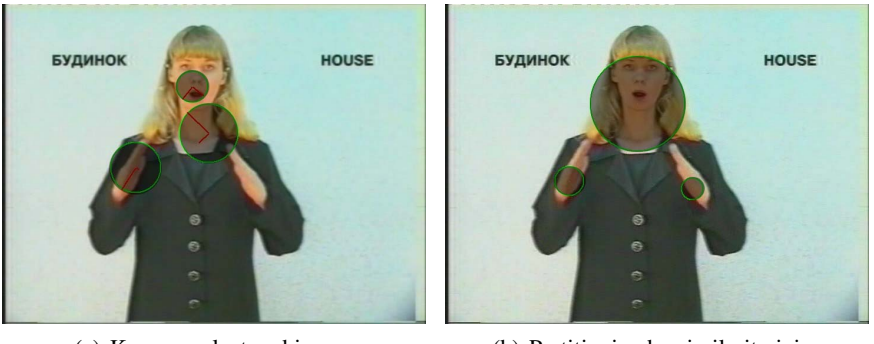


(a) Original image (b) Processed image

Fig. 13.5 Skin segmentation with the simplified Phong illumination model

weight of each cluster w. The task is to divide objects into clusters, so that the minimal distance between each cluster is more that d, and the cluster cannot be divided into smaller clusters in order to preserve this property.

The comparison of the results produced by k-means and similarity joins algorithms is depicted on Fig. 13.6.



(a) K-means clustered image (b) Partitioning by similarity joins

Fig. 13.6 Using k-means (a) and similarity joins (b) for locating hands and face

The K-means clustering produces more errors in comparison to the result produced by the similarity joins partitioning for the head and neck when hands are located closer to the face.

The Haar Cascade Classifier face detector from OpenCV library was used for the face location. The face detection classifier from OpenCV library, which is based on the integral image classifiers, was used in the dictionary pre-processing stage to improve the face recognition. However, two major drawbacks have been identified for the face classifier:

1) high computational complexity — it takes 100 ms per frame to calculate;
2) inadequate results for cases of hands intersecting the head.

It was proposed to utilise the fact that the head is a large skin-coloured cluster at the top of an image, which is well separated from segments representing hands. This was used to apply a fast similarity joins partitioning algorithm for identifying the exact location of hands and face.

The example of the extracted hand area is depicted on Fig. 13.7.

Fig. 13.7 Example of automatically extracted hand shapes

13.5 Feedback During Tutoring

The results of the undertaken studies [25] lay the foundation for the developed Ukrainian Sign Language Tutoring software. The software consists of the sign dictionary and verification modules. The student can observe simultaneously the sign video and his/her own feedback-image in the main window of the program (Fig. 13.8).

The extracted hand shapes and the positions of the tutor on the video and the student hands are used to provide feedback to the student.

As depicted on Fig. 13.8, the tutoring software consists of three windows — the window on the left listing the words from the dictionary, the window in the middle plays back the video with a sign corresponding to the selected in the list word, the window on the right displays the video stream of the student as captured by a web-camera.

Fig. 13.8 Main window of the tutor application

The provided feedback allows the student to repeat the sign after the reference video (middle window) at any speed and correct the way it is being reproduced by adjusting hand positions. The flowchart of the sign tutoring algorithms is depicted on Fig 13.9.

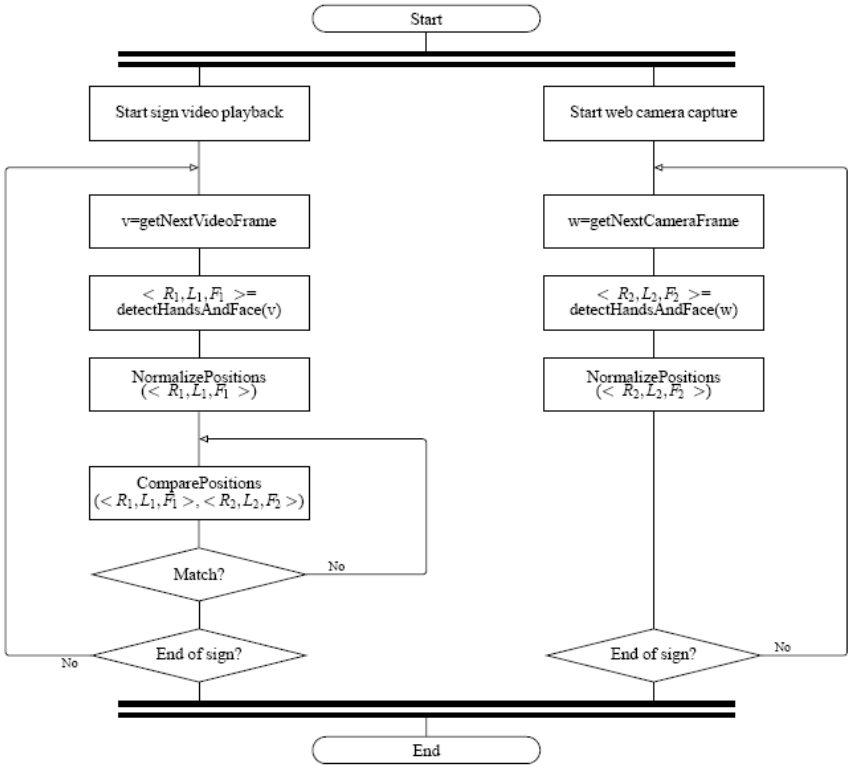


Fig. 13.9 Sign tutor feedback algorithm

The sign dictionary video and the student video from the web-camera are processed simultaneously in two different hardware threads. The hand and head positions are extracted and normalised according the frame size. The best correspondence of hands and face positions is estimated and if the distance from the student and tutor hands locations is too large then the sign video is paused. If the positions are close then the hand shapes are compared using pseudo 2-dimensional image deformation model (P2DIDM) [10]. In case where the student makes a mistake in reproducing the sign, the video with the tutor is paused, and the system awaits the student to correct the positions of the hands.

The user can playback the sign at different speeds, frame-by-frame or in a loop mode. The provided feedback allows synchronisation of the sign performed by the student and tutor. The interactive part of the tutoring software is intended to improve the practice experience of speaking the sign language.

13.6 Conclusion

The developed tutoring software has been successfully trialled in several Ukrainian schools for hearing impaired. The main advantage of the incorporated into the software methods is their ability to adjust to different light conditions and skin colour.

The interactive tutoring process with the help of gestures is much more interesting than traditional, and provides the opportunity for students to practice gestures on their own. This is especially important for distance learning.

The proposed algorithm is not robust enough when processing skin coloured objects. Another complex problem, which is not fully resolved with the developed software is the recognition of hand shapes in cases where hands overlap the head. The robustness of the software can be improved by utilising a depth camera. Currently the developed algorithms are being adjusted to make the use of additional depth information provided by the Microsoft Kinect device.

The sign language tutoring software allows teacher to select the necessary gestures for teaching quickly as well as to control gesture reproduction, which increases the efficiency of the sign language classes in comparison to the use of video materials on tapes or DVDs.

The developed software makes an effective use of the multi-core processors. The Amdahl's law $\frac{1}{(1-P)+\frac{P}{N}}$, where P is the portion of the program that can be paralleled and N — the number of hardware threads, provides a way to calculate the maximum expected improvement to an overall system. Formula (13.12) can be used to estimate the value of P.

$$P_{est} = \frac{\frac{1}{SU} - 1}{\frac{1}{NP} - 1}, \tag{13.12}$$

where SU — empirically calculated speedup coefficient for N hardware threads. Specifically, the estimation for the developed software running on the four-core processor provided by formulae (13.13) and (13.14).

$$P_{est} = \frac{\frac{1}{2.8} - 1}{\frac{1}{4} - 1} \approx 0.86 \tag{13.13}$$

$$\lim_{N \to \infty} \frac{1}{(1 - 0.86) + \frac{0.86}{N}} \approx 7.14 \tag{13.14}$$

Fig. 13.10 depicts the charts for different values of P, where the black solid chart corresponds to the discussed here software.

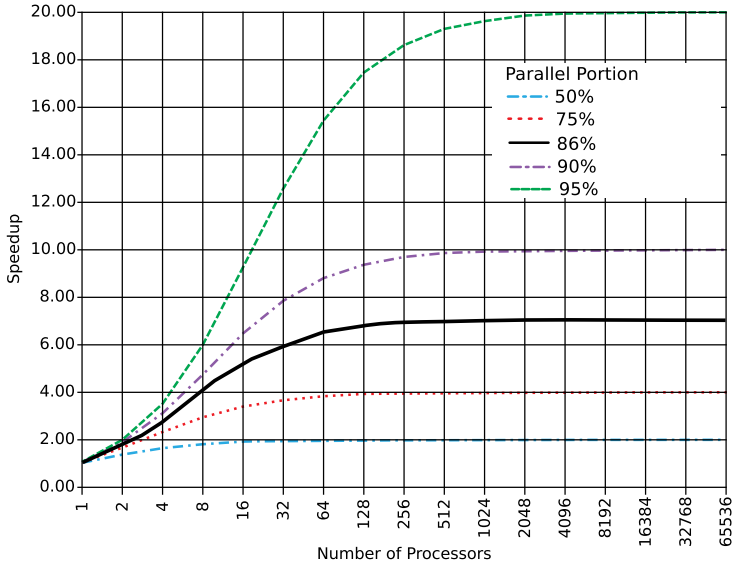


Fig. 13.10 Amdahl's Law

The developed tutoring software is not limited to Ukrainian Sign Language and can easily be adapted to any sign language as it only utilises recorded sign videos as an input.

References

1. Akyol, S., Zieren, J.: Evaluation of ASM head tracker for robustness against occlusion. In: Arabnia, H.R., Mun, Y. (eds.) Proceedings of the International Conference on Imaging Science, Systems, and Technology (CISST 2002), Las Vegas, Nevada, June 24-27, vol. I, pp. 28–34. CSREA Press (2002)
2. Aran, O., Ari, I., Benoit, A., Carrillo, A.H., Fanard, F.X., Campr, P., Akarun, L., Caplier, A., Rombaut, M., Sankur, B.: Sign language tutoring tool. In: HAL - CCSD: eNTERFACE 2006, The Summer Workshop on Multimodal Interfaces, Istanbul, Turkey, pp. 23–33 (2006)
3. Akgül, C.B.: Cascaded self-organizing networks for color image segmentation (2004), http://www.tsi.enst.fr/~akgul/oldprojects/CascadedSOM_cba.pdf (cited April 15, 2011)
4. Bradski, G.: Computer vision face tracking for use in a perceptual user interface. Intel Technology Journal 2(2), 12–21 (1998)
5. Brashear, H., Zafrulla, Z., Starner, T., Hamilton, H., Presti, P., Lee, S.: CopyCat: A corpus for verifying american sign language during gameplay by deaf children. In: Proceedings of the 4th Workshop on the Representation and Processing of Sign Languages: Corpora and Sign Language Technologies (2010)
6. Brown, D., Craw, I., Lewthwaite, J.: A SOM based approach to skin detection with application in real time systems, University of Aberdeen (2001), http://www.bmva.ac.uk/bmvc/2001/papers/33/accepted_33.pdf (cited April 15, 2011)
7. Campbell, N.W., Thomas, B.T., Troscianko, T.: Neural networks for the segmentation of outdoor images. Journal of Systems Engineering 6(3), 343–346 (1996)
8. Campbell, N.W., Thomas, B.T., Troscianko, T.: Segmentation of natural images using self-organising feature maps. In: Proceedings of the British Machine Vision Conference, pp. 223–232. University of Bristol, Bristol (1996), http://www.cs.bris.ac.uk/Publications/Papers/1000140.pdf (cited February 11, 2012)
9. Davydov, M.V., Nikolski, I.V.: Real-time video object classification using neural network classifier. Herald of National University "Lvivska Polytechnica" (549), 82–92 (2005) (Lviv, Ukraine, in Ukrainian)
10. Davydov, M.V., Nikolski, I.V.: Automatic identification of sign language gestures by means on dactyl matching. Herald of National University "Lvivska Polytechnica" (589), 174–198 (2007) (Lviv, Ukraine, in Ukrainian)
11. Davydov, M.V., Nikolski, I.V., Pasichnyk, V.V.: Software training simulator for sign language learning. Connection, 98–106 (2007) (Kyiv, Ukraine, in Ukrainian)
12. Davydov, M.V., Nikolski, I.V., Pasichnyk, V.V.: Selection of an effective method for image processing based on dactyl matching for identification of sign language gestures. Herald of Kharkiv National University of Radio-Electronics (139), 59–68 (2008) (Kharkiv, Ukraine, in Ukrainian)
13. Davydov, M.V., Nikolski, I.V., Pasichnyk, V.V.: Real-time Ukrainian sign language recognition system. In: Proceedings of the IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS), pp. 875–879 (2010)
14. Dreuw, P., Rybach, D., Deselaers, T., Zahedi, M.: Ney. H.: Speech recognition techniques for a sign language recognition system. In: ISCA Best Student Paper Award Interspeech 2007, Antwerp, Belgium, pp. 2513–2516 (2007)
15. Dreuw, P., Stein, D., Ney, H.: Enhancing a Sign Language Translation System with Vision-Based Features. In: Sales Dias, M., Gibet, S., Wanderley, M.M., Bastos, R. (eds.) GW 2007. LNCS (LNAI), vol. 5085, pp. 108–113. Springer, Heidelberg (2009)
16. Dreuw, P., Forster, J., Deselaers, T., Ney, H.: Efficient approximations to model-based joint tracking and recognition of continuous sign language. In: Proceedings of the 8th IEEE International Conference Automatic Face and Gesture Recognition, Amsterdam, The Netherlands, September 17-19, pp. 1–6 (2008), http://thomas.deselaers.de/publications/papers/dreuw_fg08.pdf (cited February 11, 2011)

17. Ford, A., Roberts, A.: Colour Space Conversions (1998), http://www.poynton.com/PDFs/coloureq.pdf (cited April 15, 2011)
18. García, C., Prieto, M., Pascual-Montano, A.D.: A speculative parallel algorithm for self-organizing maps. In: PARCO, pp. 615–622 (2005), http://www2.fz-juelich.de/nic-series/volume33/615.pdf (Cited April 15, 2011)
19. Hämäläinen, T.D.: Parallel implementation of self-organizing maps. In: Seiffert, U., Jain, L.C. (eds.) Self-Organizing Neural Networks: Recent Advances and Applications, pp. 245–278. Springer, New York (2002)
20. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann Publishers (2001)
21. Hodych, O., Nikolskyi, Y., Shcherbyna, Y.: Application of self-organising maps in medical diagnostics. Herald of National University "Lvivska Polytechnica" (464), 31–43 (2002) (Lviv, Ukraine, in Ukrainian)
22. Hodych, O., Nikolskyi, Y., Pasichnyk, V., Shcherbyna, Y.: Analysis and comparison of SOM-based training algorithms. Control Systems and Machines (2), 63–80 (2006) (Kyiv, Ukraine, in Ukrainian)
23. Hodych, O., Nikolskyi, Y., Pasichnyk, V., Shcherbyna, Y.: High-dimensional data structure analysis using self-organising maps. In: Proceedings of the 9th International Conference on CAD Systems in Microelectronics (CADSM 2007), pp. 218–221 (2007)
24. Hodych, O., Hushchyn, K., Shcherbyna, Y., Nikolski, I., Pasichnyk, V.: SOM-Based Dynamic Image Segmentation for Sign Language Training Simulator. In: Yang, J., Ginige, A., Mayr, H.C., Kutsche, R.-D. (eds.) Information Systems: Modeling, Development, and Integration. LNBIP, vol. 20, pp. 29–40. Springer, Heidelberg (2009)
25. Hodych, O.V., Davydov, M.V., Nikolski, I.V., Pasichnyk, V.V., Scherbyna, Y.M.: Ukrainian Sign Language: the aspects of computational linguistics. Piramida, Lviv (2009) (in Ukrainian)
26. Hoffmann, G.: CIE color space (2000), http://www.fho-emden.de/~hoffmann/ciexyz29082000.pdf (cited April 15, 2011)
27. Hoffmann, G.: CIELab color space (2003), http://www.fho-emden.de/~hoffmann/cielab03022003.pdf (cited April 15, 2011)
28. Hunt, R.W.G.: Measuring Colour, 3rd edn. Fountain Pr. Ltd (2001)
29. Jacox, E.H., Hanan, S.: Metric space similarity joins. ACM Trans. Database Syst. 33(2), 7:1–7:38 (2008)
30. Jander, M., Luciano, F.: Neural-based color image segmentation and classification using self-organizing maps (1996), http://mirror.impa.br/sibgrapi96/trabs/pdf/a19.pdf (cited April 15, 2011)
31. Jiang, Y., Chen, K.-J., Zhou, Z.-H.: SOM Based Image Segmentation. In: Wang, G., Liu, Q., Yao, Y., Skowron, A. (eds.) RSFDGrC 2003. LNCS (LNAI), vol. 2639, pp. 640–643. Springer, Heidelberg (2003)
32. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer, Berlin (2001)
33. Lawrence, R. D., Almasi, G.S., Rushmeier, H.E.: A scalable parallel algorithm for self-organizing maps with applications to sparse data mining problmes, http://www.research.ibm.com/dar/papers/pdf/scalableSOM.pdf (cited April 15, 2011)
34. Rauber, A., Tomsich, P., Merkl, D.: parSOM: A parallel implementation of the self-organizing map exploiting cache effects: making the som fit for interactive high-performance data analysis. Neural Networks 6, 61–77 (2000)
35. Reyes-Aldasoro, C.C.: Image segmentation with Kohonen neural network self-organising maps (2004) http://www.cs.jhu.edu/~cis/cista/446/papers/SegmentationWithSOM.pdf (cited April 15, 2011)
36. Tominaga, S., Takahashi, E., Tanaka, N.: Parameter estimation of a reflection model from a multi-band image. In: Proceedings of the 1999 IEEE Workshop on Photometric Modeling for Computer Vision and Graphics, pp. 56–63 (1999)

37. Wu, Y., Liu, Q., Huang, T.S.: An adaptive self-organizing color segmentation algorithm with application to robust real-time human hand localization. In: Proceedings of the Asian Conference on Computer Vision, Taiwan, pp. 1106–1111 (2000)
38. Zahedi, M., Keysers, D., Deselaers, T., Ney, H.: Combination of Tangent Distance and an Image Distortion Model for Appearance-Based Sign Language Recognition. In: Kropatsch, W.G., Sablatnig, R., Hanbury, A. (eds.) DAGM 2005. LNCS, vol. 3663, pp. 401–408. Springer, Heidelberg (2005)
39. Zieren, J., Kraiss, K.-F.: Non-intrusive sign language recognition for human-computer interaction. In: Proceedings of the 9th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design, and Evaluation of Human-Machine Systems, Atlanta, Georgia, p.CD-paper 49 (2004)
40. Zieren, J., Kraiss, K.-F.: Robust Person-Independent Visual Sign Language Recognition. In: Marques, J.S., Pérez de la Blanca, N., Pina, P. (eds.) IbPRIA 2005. LNCS, vol. 3522, pp. 520–528. Springer, Heidelberg (2005)
41. IBM research demonstrates innovative 'Speech to Sign Language' translation system. Pressrelease (September 12 2007), http://www-03.ibm.com/press/us/en/pressrelease/22316.wss (cited April 15, 2011)
42. The iCommunicator User's Guide (2005), http://www.myicommunicator.com/downloads/iCommunicator-UserGuide-v40.pdf (cited April 15, 2011)
43. Senthilkumaran, N., Rajesh, R.: A study on rough set theory for medical image segmentation. International Journal of Recent Trends in Engineering 2(2), 236–238 (2009)

Chapter 14
Hybrid Methods in Data Classification and Reduction

Paweł Delimata and Zbigniew Suraj

Abstract. In this paper, summary of many experiments for various hybrid methods will be presented. Hybrid methods combine various methodologies from Data Mining such as data reduction, multiple classifiers systems, feature selection with rough sets methods. In the paper, three algorithms will be presented which will use the notion of surroundings and a k-NN method for data reduction. The paper also describes one multiple classifier system which uses several algorithms such as k-NN, decomposition trees and neural network. The rest of the paper focuses on five algorithms which use reducts and deterministic or inhibitory decision rules for feature selection. All the algorithms presented in the paper were tested on well known data sets from the UCI Repository of Machine Learning Databases. The algorithms presented in the paper have been implemented and can be tested in the DMES system.

Keywords: Data reduction, multiple classifiers, feature selection, hybrid methods.

14.1 Introduction

In many cases, for real-life data we want to have fast classifying algorithms. Fast decision making is very important in many situations. The objective of this study is to summarize results obtained in three aspects of data analysis: data reduction, feature subset selection and construction of the classifier.

In the section with data reduction methods we introduce a model of data classification based on preliminary reduction of the training set of examples. Its purpose is to facilitate the use of NN techniques in near real-time applications. The extraordinary progress in the computer field has made NN techniques, once considered impractical from a computational viewpoint, feasible for consideration in

Paweł Delimata · Zbigniew Suraj
Institute of Computer Science, University of Rzeszów, Dekerta 2, 35-030 Rzeszów, Poland
e-mail: pdelimata@wp.pl, zsuraj@univ.rzeszow.pl

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 263–291.
springerlink.com

time-constrained, real-world applications. This study accordingly addresses the issue of minimising the computational resource requirements of NN techniques, memory as well as time, through the use of preliminary reduction techniques preserving the relatively high classification accuracy.

Generally, the training set reduction methods described in the paper eliminate the examples that do not satisfy some global assumptions about data. Two of the presented reduction methods eliminate the objects that do not match their neighbors. In the third one, the elimination of training examples is based on the relation between the examples and some prototypes representing the centers of the decision classes. These global assumptions about data seem to be rather in opposition to the idea of local models because the method with the local model assumes that some training examples represent the local properties of some specific subsets of the data set. Eliminating their elements on the basis of global criteria can lead to loss of information about these local properties. Sometimes this fact can cause the decrease of the classification accuracy of the local model based on the method with the training set reduction. Nevertheless, the results of experiments conducted on well known data sets found in the literature [2] with various combinations of reduction methods [16], [9] demonstrate the potential benefits of such reduction methods also for local models. They also highlight the desirability of a more thorough exploration of combinations of other alternative reduction methods that have been reported in the literature over the past few decades.

The section with feature selection methods focuses on five methods. The main one is the *RBFS* (*Reduct Based Feature Selection*) algorithm [18]. In the paper we discuss three ways of improving *RBFS* algorithm: by reducing the size of the set of decision-relative reducts on which algorithms work, by applying simplified version of the bagging algorithm [3] to the feature selection algorithm and by a combination of these two approaches.

By applying such methods to the *RBFS* algorithm we want to decrease execution time of the algorithm by decreasing the number of decision-relative reducts on which algorithm works and we want to increase the classification accuracy of the multiple classifier which is created from subsets of the decision table obtained from the *RBFS* algorithm. To achieve these aims we propose to use the *ARS* (*Algorithm for Reducts Selection*) algorithm. *ARS* selects reducts from the set of decision-relative reducts of a given decision table. The second aim (to increase the classification accuracy) is achieved by increasing the number of classifiers/subsets created by the *RBFS* algorithm, which is formulated by applying the simplified bagging approach to the classifier ensemble. The *RBFS* algorithm requires a lot of calculations; consequently, the computational complexity is very high. In our research the combination of the *ARS* algorithm with the *RBFS* algorithm allowed us to greatly decrease the computational time of the *RBFS* algorithm. Subsets created by this method, after using the *ARS* algorithm, were tested with a simple multiple classifier (k-NN classifiers combined with a simple voting scheme). The results were almost identical to the results obtained with the multiple classifier which worked with subsets created by *RBFS* algorithms without *ARS*. Applying the simplified bagging algorithm both to the plain *RBFS* algorithm and the *RBFS* algorithm combined with the *ARS*

algorithm allowed us to slightly increase the classification accuracy of the classifier ensemble. We compared these results with the *FASBIR* algorithm [25] which is a multiple classifier system. The comparison of the *RBFS* methods includes the classification accuracy and classification time of the multiple classifier which uses the *RBFS* algorithm for feature selection. The comparison between *FASBIR* and *RBFS* includes the comparison of the classification accuracy of the *FASBIR* algorithm with the classification accuracy of the multiple classifier created with the use of the *RBFS* algorithm.

In the feature selection section we also describe two algorithms for reducts evaluation. Presented algorithms use lazy deterministic and inhibitory rules [6] to calculate reduct usefulness. The purpose of these methods is to present the way of estimating the value of the reduct or the subset of the attribute set. This value can be used later in the reduct selection algorithm or in preprocessing stages for various classifiers.

The presented algorithms calculate the number of deterministic and inhibitory decision rules for a given reduct or subset of attributes. Deterministic and inhibitory rules calculation was performed with the use of the lazy algorithms. These algorithms do not calculate the whole rules set. They only provide information if certain rules exist or not.

Estimation of the value of the reduct or the attribute set obtained from algorithms, can be used to select reducts or the attribute set for which a given classifier (in this case k-NN and naive bayesian classifier) will have the best classification accuracy.

Estimation of the value of reducts and classification accuracy obtained with the use of the classifier have been compared. Experiments showed that a number of deterministic and inhibitory decision rules calculated by algorithms presented in this paper can be a good estimate of the accuracy of k-NN and naive bayesian classifier.

The last algorithm for feature selection presented in this section is *RedBoost*. The algorithm uses a set of decision relative or decision and object relative reducts (not necessarily a set of all reducts), training decision table, classifier (in this case k-NN) and percentage parameter. The algorithm from a given set of reducts creates feature subsets which have better, or equal, classification accuracy than reducts before the use of the *RedBoost* algorithm (i.e. a classifier that uses decision subtable created on the basis of reduct or feature subset that has better classification accuracy).

The algorithm reduces the number of created feature subsets by removing duplicates that are constructed during their creation by *RedBoost* and leaves only a percentage of created subsets (indicated by the percentage parameter).

Several experiments were performed using the *RedBoost* algorithm on the benchmark data tables from the *UCI* Machine Learning Databases [2]. A simple multiple classifier system was used for feature subsets testing. The results were compared with the results of other feature selection algorithms combined with the same multiple classifier system, and with results obtained with a single classifier k-NN and *FASBIR* multiple classifier system.

In the section with construction of the classifier we describe *MC*2 classifier. It is a multiple classifier system which uses multiple classifiers to reduce the size

of training and test sets. These new training and test sets are used during the classification. *MC*2 method allowed us to reduce almost 91% of the training set with a small loss on classification accuracy. *MC*2 method presented in this paper is closely related to the *stack generalization* [23]. Our *MC*2 method can be also interpreted as a *layered learning* presented in [15]. In that case we must treat the same classifiers, but defined on more generalized data as classifiers with relaxed conditions.

14.2 Basic Notions

14.2.1 Information Systems

An *information system* [11,12] is a pair $S = (U, A)$, where U is a non – empty, finite set of *objects* and A is a non–empty, finite set of *attributes*. Each attribute $a \in A$ corresponds to a function $a: U \to V_a$, where V_a is called the value set of a.

In supervised learning problems, objects (examples) from a training set are pre-classified into several categories or classes. To deal with such type of data we use the *decision systems* of the form $S = (U, A, dec)$, where $dec \notin A$ is a distinguished attribute called *decision* and elements of an attribute set A are called *conditions*. In practice, the decision systems contain a description of a finite sample U of objects from larger (maybe infinite) universe $\boldsymbol{U}$. Conditions are such attributes that their values are known for all objects from $\boldsymbol{U}$, but decision is a function defined on the objects from the sample U only. Without loss of generality one can assume that the domain V_{dec} of the decision dec is equal to $\{1, \ldots, d\}$. The decision dec determines a partition $\{CLASS_1, \ldots, CLASS_d\}$ of the universe U, where $CLASS_k = \{x \in U : dec(x) = k\}$ is called the k-th decision class of S for $1 \leq k \leq d$. By class distribution of any set $X \subseteq U$ we denote the vector $ClassDist(X) = < n_1, \ldots, n_d >$, where $n_k = card(X \cap CLASS_k)$ is the number of objects from X belonging to the k-th decision class [1].

14.2.2 Classical k-NN Method

In the k-NN method, it is necessary to define a *distance function* ρ between objects, $\rho: U \times U \to \boldsymbol{R}^+$, where $\boldsymbol{R}^+$ denotes the set of positive real numbers. The problem of searching for a relevant distance function for the given data set is not trivial. In the following, we assume that such function has already been defined.

In the k-NN method, the decision for a new object $x \in \boldsymbol{U} \setminus U$ is made on the basis of the set $NN(x;k) := \{x_1, \ldots, x_k\} \subseteq U$ with k objects from U which are nearest to x with respect to the distance function ρ. Usually, k is a parameter which can be set up by an expert or constructed from an experimental data. The k-NN classifiers

often use the voting algorithm for decision making, i.e., the decision value for a new object x can be predicted by:

$$dec(x) = Voting(<n_1, \ldots, n_d>),$$

where $ClassDist(NN(x;k))=< n_1, \ldots, n_d >$ is the class distribution of the set $NN(x;k)$ satisfying $n_1 + \ldots + n_d = k$. The voting function can return the most frequent decision value occurring in $NN(x;k)$. In case of imbalanced data, the vector $< n_1, \ldots, n_d >$ can be scaled, and after that the voting algorithm can be employed [1].

14.2.3 Reducts

In the reduction of knowledge the basic role is played by the fundamental concept of a *reduct*. Intuitively, a *reduct* of knowledge is its essential part, which suffices to define all basic concepts occurring in the considered knowledge, whereas the *core* is, in a certain sense, its most important part. We could say that a *reduct* is a minimal subset of the attribute set, such that it gives the same set of elementary concepts, and thus the same ability to express properties of the objects as the original set of attributes. The remaining attributes are redundant since their removal does not worsen the classification. In the rough set theory, many different types of reducts are considered. In the present paper, we use the so called *decision-relative reducts*. Reducts of this type are minimal conditional attribute subsets that, for all objects, enable us to make the same classifications as the full set of attributes does. Further information can be found in [11,12], [13,14].

14.2.4 Leave-One-Out Cross Validation Method

The *leave-one-out cross validation* method is performed as follows. From a given set, we take one or more objects. The remaining objects act as a training set, and object(s) that has/have been taken act as a test set. Next we put back the object(s) that we have previously taken and we take another object(s). The procedure continues for every object in a given set. Further information can be found in [5].

14.2.5 Bagging Algorithm

The *Bagging* approach (Bootstrap aggregating) was introduced by Breiman [3]. It aggregates by voting classifiers generated from different bootstrap samples. The *bootstrap sample* is obtained by sampling objects uniformly from the training set with replacement. Each sample has the same size as the original set. Although some examples do not appear in it, others may appear more than once.

14.2.6 Metric and Single Classifier

In all experiments the classical k-NN algorithm was used. The value of the k parameter was set to $\{1, 3, 5, 7, 9\}$. The distance used by k-NN classifier is called *MinkoVDM* [25] which combines *Minkowsky* and *VDM* distance. Here x_1 and x_2 are two objects described by d-dimensional continuous attribute vectors. *Minkowsky* distance is obtained from the formula Eq. 14.1

$$Minkowsky_p(x_1, x_2) = (\sum_{n=1}^{d} |x_{1,n} - x_{2,n}|^p)^{1/p} \tag{14.1}$$

By setting different values to p, different distance metrics can be obtained. In our case we use $p = 2$. In general, the smaller the value of p, the more robust the resulting distance metric to data variations; whereas the bigger the value of p, the more sensitive the resulting distance metrics to variations. *Minkowsky* distance can hardly deal with categorical attributes. Fortunately, *VDM Value Difference Metric* [17], [21] can be a good complement. Let $N_{a,u}$ denote the number of training examples holding value u on categorical attribute a, let $N_{a,u,c}$ denote the number of training examples belonging to the c-th class holding value u on a, and let n denote the number of decision classes. The distance between the two values u and v on a can be computed by a simplified version of *VDM* shown in Eq. 14.2.

$$VDM_p(u, v) = \sum_{c=1}^{n} |\frac{N_{a,u,c}}{N_{a,u}} - \frac{N_{a,v,c}}{N_{a,v}}|^p \tag{14.2}$$

MinkoVDM metric can be obtained by combining these two metrics in the way shown by Eq. 14.3, where first j attributes are categorical while the remaining $(d - j)$ ones are continuous attributes normalized to [0, 1]. It is evident that such a distance can deal with both continuous and categorical attributes.

$$MinkoVDM_p(x_1, x_2) = (\sum_{h=1}^{j} VDM_p(x_{1,h}, x_{2,h}) + \sum_{h=j+1}^{d} |x_{1,h} - x_{2,h}|^p)^{1/p} \tag{14.3}$$

14.2.7 Measures of Diversity

We use Eq. 14.5 and Eq. 14.6 previously proposed by Yule in [24] and the new measure Eq. 14.7, proposed in this paper, as measures of diversity between classifiers D_i and D_k, where D_i and D_k are classifiers created from subsets of attributes and the k - *NN* classifier. The subsets are evaluated with the use of the leave-one-out cross validation method.

Values a, b, c, d are the percentage of the respective pair of correct/incorect classifiers outputs. Table 14.1 summarizes these outputs.

Table 14.1 The 2x2 relationship table with percentages

	D_k correct(1)	D_k wrong(0)
D_i correct(1)	a	b
D_i wrong(0)	c	d

Total,

$$a+b+c+d=1 \tag{14.4}$$

Using these four values we can construct three diversity measures:

$$\alpha_{i,k} = \frac{ad-bc}{\sqrt{(a+b)(c+d)(a+c)(b+d)}} \tag{14.5}$$

$$\beta_{i,k} = \frac{ad-bc}{ad+bc} \tag{14.6}$$

$$\gamma_{i,k} = -(b+c)+d \tag{14.7}$$

Eq. 14.5, Eq. 14.6, and Eq. 14.7 measure diversity between two binary classifier outputs (correct/incorrect). When classifier outputs are correlated (many objects are classified correctly/incorrectly by both classifiers) the value of $\alpha_{i,k}, \beta_{i,k}, \gamma_{i,k}$ is close or equal to 1. In the other case when classifier outputs are not correlated (many objects are classified differently (correct/incorrect) by both classifiers), the value of this equation is close to -1. $\gamma_{i,k}$ diversity measure does not use a-value from Table 14.1 directly. This value is connected with other values by Eq. 14.4. Measure γ gives good values when $b+c$ is relatively high and there is a small percentage of objects classified improperly by both classifiers.

The required value for each of these measures should be as close to -1 as possible. Values close to 1 are not wanted because they indicate that classifiers gave similar results for the same objects. Our goal is to obtain diverse classifiers.

14.2.8 Decision Rules

Decision rules make it possible to classify objects, i.e., assign the value of the decision attribute. Having a collection of *rules* pointing at different decision, we may perform a voting, in this way obtaining a simple *rule*-based decision support system. The *decision rules* are a basic tool for data classification. Each *decision rule* consists of conditions and the decision. When a given object x satisfies all of these conditions, then the *decision rule* can tell us what the decision of the tested object

x is. One *decision rule* can classify objects only to one decision class. In order to classify new objects properly we must have a large set of *decision rules*. Algorithms that use *decision rules* for classification often use voting to obtain the decision made by the rules.

14.2.9 LTF-c Neural Network

$LTF-C$ (Local Transfer Function Classifier) (see [22]) is a classification-oriented artificial neural network model similar to that of Radial Basis Function network (RBF). It consists of two layers of computational neurons (plus an input layer). The first computational layer (hidden layer) consists of neurons that correspond to clusters of objects from the same decision class. Each of these neurons has a decision class assigned, and tries to construct a cluster of objects from this class. The second computational layer (output layer) consists of neurons which gather information from hidden (cluster-related) neurons, sum it up, and produce the final network's output.

14.2.10 Decomposition Tree

Decomposition trees are used to split a data set into fragments not larger than a predefined size. These fragments, after decomposition represented as leaves in the decomposition tree, are supposed to be more uniform and easier to cope with in a decision wise manner. For more information on underlying methods, please turn to [10]. Usually, the subsets of data in the leaves of the decomposition tree are used for calculation of decision rules.

14.2.11 Deterministic and Inhibitory Decision Rules

Let $T = (U, A, d)$ be a *decision table*, where $U = \{u_1, \ldots, u_n\}$ is a finite nonempty set of *objects*, $A = \{a_1, \ldots, a_m\}$ is a finite nonempty set of *conditional attributes*, and d is the *decision attribute*.

The set $\mathcal{U}(T)$ is called the *universe* for the decision table T. Besides objects from U we also consider objects from $\mathcal{U}(T)$. For any object (tuple) $v \in \mathcal{U}(T)$ and any attribute $a_j \in A$ the value $a_j(v)$ is equal to j-th integer component of v.

By the set $V_d(T)$ we denote the set of all decisions of table T. Let us consider a rules

$$a_{j_1}(x) = b_1 \wedge \ldots \wedge a_{j_t}(x) = b_t \Rightarrow d(x) = b, \tag{14.8}$$

$$a_{j_1}(x) = b_1 \wedge \ldots \wedge a_{j_t}(x) = b_t \Rightarrow d(x) \neq b, \tag{14.9}$$

where $t \geq 0$, $a_{j_1}, \ldots, a_{j_t} \in A$, $b_1, \ldots, b_t \in \omega$, $b \in V_d(T)$ and numbers $j_1, \ldots, j_t$ are pairwise different. Such rules are called *deterministic decision* rules (14.8) and *inhibitory decision* rules (14.9). The rules (14.8, 14.9) are called *realizable for an object* $u \in \mathcal{U}(T)$ if $a_{j_1}(u) = b_1, \ldots, a_{j_t}(u) = b_t$ or $t = 0$. The rule (14.8) is called *true for an object* $u_i \in U$ if $d(u_i) = b$ or (14.8) is not realizable for u. The rule (14.9) is called *true for an object* $u_i \in U$ if $d(u_i) \neq b$ or (14.9) is not realizable for u_i. The rule (14.8, 14.9) is called *true for* T if it is true for any object from U. The rule (14.8, 14.9) is called *realizable for* T if it is realizable for at least one object from U. By $Det(T)$ we denote the set of all deterministic decision rules and by $Inh(T)$ we denote the set of all inhibitory decision rules which are true for T and realizable for T.

Our aim is to recognize for given objects $u_i \in U$ and $v \in \mathcal{U}(T)$, and given value $b \in V_d(T)$ if there exists a rule from $Det(T)$ ($Inh(T)$) which is realizable for u_i and v and has $d(x) = b$ ($d(x) \neq b$) on the right hand side. Such a rule "supports" ("contradicts") the assignment of the decision b to the new object v.

Let $M(u_i, v) = \{a_j : a_j \in A, a_j(u_i) = a_j(v)\}$ and $P(u_i, v) = \{d(u) : u \in U, a_j(u) = a_j(v)$ for any $a_j \in M(u_i, v)\}$. Note that if $M(u_i, v) = \emptyset$, then $P(u_i, v) = \{d(u) : u \in U\} = V_d(T)$.

Proposition 14.1. *Let $T = (U, A, d)$ be a decision table, $u_i \in U$, $v \in \mathcal{U}(T)$, and $b \in V_d(T)$. Then in $Det(T)$ there exists a rule, which is realizable for u_i and v and has $d(x) = b$ on the right hand side, if and only if $P(u_i, v) = \{b\}$.*

Proposition 14.2. *Let $T = (U, A, d)$ be a decision table, $u_i \in U$, $v \in \mathcal{U}(T)$, and $b \in V_d(T)$. Then in $Inh(T)$ there exists a rule, which is realizable for u_i and v, and has $d(x) \neq b$ on the right hand side, if and only if $b \notin P(u_i, v)$.*

Proofs of the propositions 1 and 2 can be found in [6].

From Proposition 14.1 and 14.2 it follows that, there exists a polynomial algorithm recognizing, for a given decision table $T = (U, A, d)$, given objects $u_i \in U$ and $v \in \mathcal{U}(T)$, and a given value $b \in V_d(T)$, if there exists a rule from $Det(T)$ $Inh(T)$, which is realizable for u_i and v, and has $d(x) = b$ ($d(x) \neq b$) on the right hand side. This algorithm constructs the set $M(u_i, v)$ and the set $P(u_i, v)$. The considered rule exists if and only if $P(u_i, v) = \{b\}$ ($b \notin P(u_i, v)$).

14.3 Data Reduction Methods

In this section, we describe three reduction methods. These methods eliminate the objects that do not satisfy some global assumptions about data. We named these methods as follows: *Red1, Red2, Red3.* In each method we consider two sets of

objects. The first, U_{trn} is a training set. The second, set $\bar{U}_{trn}$, is obtained from U_{trn} by reduction ($\bar{U}_{trn} \subseteq U_{trn}$). All data sets obtained by reduction were tested with the classical k-NN. In most of the cases, the reduction gave us a good improvement of the execution time for the classical k-NN with a small loss of classification accuracy.

Algorithm $RED1$ presented below eliminates the objects that do not match their neighbours. It checks surroundings of objects with the use of the specified radius. The object which is currently tested is being thrown away if its decision is different from the most frequent decision among objects in surroundings. It is important that during the execution of the algorithm, objects are only marked. They are thrown away when the algorithm stops. This algorithm allows throwing away objects which are badly classified or cause noise.

Algorithm 14.1. Algorithm $RED1$

Input : U_{trn} – a training set of objects,
ρ – metric defined on set U_{trn},
$Par \in (0, 1)$ – a parameter.

Output: $\bar{U}_{trn}$ – a reduced set of training objects.

$\bar{U}_{trn} = \emptyset$ - an empty set of objects,

$Max\_dist = \max_{x,y \in U_{trn}} (\rho(x,y))$ maximal distance between training objects

$R = Max\_dist * Par;$ radius of spheres

forall the $(x \in U_{trn})$ **do**

$Rset(x) = \{y \in U_{trn} : \rho(x,y) < R\}$ a set of objects from the training set that are in radius R.

$d = Voting(ClassDist(Rset(x)))$ the most frequent decision value in $Rset(x)$.

if $(Dec(x) = d)$ **then**

$\bar{U}_{trn} = \bar{U}_{trn} \cup \{x\}$ if a object satisfies the condition, we add it to the $\bar{U}_{trn}$ set.

Algorithm $RED1$ starts with the empty set $\bar{U}_{trn}$ and a given training set U_{trn}. Parameter Par must also be specified. First, we compute *maximal distance* between training objects. We can make this process faster by random selection of a subset of a training set and finding the Max_dist of this subset. Then, we compute the radius by multiplying Max_dist by Par. Since $Par \in (0, 1)$, the radius will never be 0. Next, for each object from U_{trn} we find the sphere with radius R, after that, we find the most frequent value of the decision for objects in that sphere. Finally, we check if the decision of an object x is the same as the most frequent decision value that we have found. If so, we add the object x to the set $\bar{U}_{trn}$.

Parameter *Par* is very important in this method. Too low or too high value of this parameter can cause total reduction of the training set (all objects can be reduced). Different values of this parameter gave us different results. In our experiments we have checked many variants.

The time complexity of this algorithm is $O(mn^2)$, where n denotes the number of training objects in the set U_{trn} and m denotes the number of attributes.

Algorithm *RED*2 presented in the following is based on the concept of k nearest neighbours. It checks k nearest neighbours of objects. The object which is currently tested is being thrown away if its decision is different from all of its k nearest neighbours decisions. It is important that during the execution of the algorithm objects are only marked. They are thrown away when the algorithm stops. This algorithm allows throwing away objects which are badly classified or cause noise.

Algorithm 14.2. Algorithm *RED*2

Input : U_{trn} – a training set of objects,
ρ – metric defined on set U_{trn},
$K \in \mathbf{N}$ – a parameter. $\mathbf{N}$ - the set of positive integers.

Output: $\bar{U}_{trn}$ – a reduced set of training objects.

$\bar{U}_{trn} = \emptyset$ - an empty set of objects,

forall the $(x \in U_{trn})$ **do**
  if *(for all* $y \in NN(x; K)$*;* $dec(y) = dec(x)$*)* **then**
    $\bar{U}_{trn} = \bar{U}_{trn} \cup \{x\}$

The algorithm is a modification of *Edited Nearest Neighbour Rule* considered in [20]. We assume that U_{trn}, K are given and set $\bar{U}_{trn}$ is empty. For all elements x from U_{trn} we find the sets $NN(x;\ K)$. If all elements from set $NN(x;\ K)$ have the same decision as the examined object x, we add the object x to set $\bar{U}_{trn}$. A proper value of parameter K is important an thing in this algorithm. If it is too high, it can cause the reduction of too many objects. In our experiments, we have used many different values of this parameter. By $Num(d,\ SP)$ we denote the percentage of objects in sphere SP with decision different from d. $SP(x,\ R)$ denotes the sphere with the middle in x and the radius R.

The time complexity of this algorithm is $O(mn^3)$, where n denotes the number of training objects in U_{trn} and m denotes the number of attributes.

Algorithm *RED*3 presented below is based on the relation between the objects and some prototypes representing the centers of the decision classes. It checks surroundings of objects with a specified radius. We check surroundings but only for the central objects (These central objects are calculated by Algorithm 5 described further). Each central object represents one decision class. Iteratively increasing the radius, the algorithm checks the surroundings of the central objects. If percentage

of objects with different decision is higher than the parameter, all objects from the surroundings are marked. All objects that are not marked are thrown away when the algorithm stops. This algorithm allows throwing away objects which are badly classified or cause noise.

Algorithm 14.3. Algorithm *RED*3

Input : U_{trn} – a training set of objects,
ρ – metric defined on set U_{trn},
Central – a set of central objects from all decision classes,
$Par \in \boldsymbol{N}$ – a parameter,
Perc – a percentage parameter.

Output: $\bar{U}_{trn}$ – a reduced set of training objects.

$\bar{U}_{trn} = \emptyset$ - an empty set of objects,

$Max\_dist = \max_{x,y \in U_{trn}} (\rho(x,y))$ maximal distance between training objects

$Min\_dist = \min_{x,y \in U_{trn}} (\rho(x,y))$ minimal distance between training objects

Step 3. $Delta := (Max\_dist - Min\_dist)/Par$ the radius will be changed by this value during process

$R = Min\_dist$ initial radius of spheres

repeat
  forall the $(c \in Central)$ **do**
    if $(Num(dec(c), SP(c,R)) < Perc)$ **then**
      $\bar{U}_{trn} = \bar{U}_{trn} \cup SP(c,R)$
    $R = R + Delta$ – increase radius by Delta
until $(ForAll\ c \in Central; Num(dec(c), SP(c,R)) > Perc\ OR\ R \geq Max\_dist)$;

First, we find the *central object* for each decision class. Next, we compute the *maximal* and the *minimal distance* between training objects. We can make this process faster by a random selection of a subset of a training set and then finding the *Max_dist* and *Min_dist* on this subset. For these values we compute the parameter *Delta*. For each central object, Algorithm *RED*3 checks the decisions of objects from the sphere with the middle c and radius R. If the percentage of objects with different decision in this sphere is lower than the parameter *Perc*, then all objects from $SP(c, R)$ are added to set $\bar{U}_{trn}$, otherwise another central object is checked. We increase the radius of the spheres in each pass of the main loop.

Algorithm *RED*3 can become less/more sensitive when we, decrease/increase parameter *Par*. If parameter *Perc* is set at a too low level, we can cause total reduction of a training set. The meaning of parameter *Perc* is: the maximum allowed percentage of objects from different decision classes in a sphere.

For real values of attributes we can use an arithmetical average to find the central object for each decision class. In the case of nominal values of attributes, another method has to be applied.

The time complexity of this algorithm depends on the value of parameter *Par*. Let us assume that parameter *Par* has a value n where n is the number of training objects in set U_{trn} and m denotes number of attributes. The time complexity of Algorithm 4 is $O(mn^3)$. But for low (i.e., lower than 100) values of parameter *Par* the time complexity of the algorithm is $O(mn^2)$.

Let $ClassAttrib(X,\ a)\ =\ <k_{a_1},\ ...,\ k_{a_n}>$ denote a set with the numbers of occurrence. Each value $k_{a_1},\ ...,\ k_{a_n}$ is the number of objects with value a_i on attribute a in set X.

Algorithm 14.4. Algorithm *Finding central objects*

Input : U_{trn} – a training set of objects,
$Mod \in (0, 1)$– a parameter.

Output: *Central* – a set of central objects.

$Central = \emptyset$ –an empty set of central objects.

forall the $(d \in V_{dec})$ **do**
  forall the $(a \in A)$ **do**
    $Max\_occ = \max(ClassAttrib(CLASS_d, a))$
    $occ = max\_occ \cdot Mod$
    val_a = *Voting*(which element from $ClassAttrib(CLASS_d, a)$ is closest to occ)
  $Central = Central\ \cup\ <val_a,\ ...,\ val_s, d>$

Algorithm *Central* for each decision class finds one central object. It makes this object from nominal values which occur in the decision class. For each attribute, first, we find set $ClassAttrib(CLASS_d, a)$. Then, we find the value *max_occ* which is the maximal number of occurrence of the value in a decision class on attribute a. Next, we multiply this value by parameter *Mod*. After that, we find which attribute value with its occurrence is closest to the number *occ*. Finally, we make central object from values $val_a, \ldots, val_s$, decision d and add it to the central set. The value of parameter *Mod* was obtained during experiments, and it can be changed to the value which fits the experimental data best. In our experiments, we have used this parameter with the value 0.618.

The time complexity of this algorithm is $O(mn^2)$, where n denotes the number of training objects in U_{trn} and m - the number of attributes.

14.3.1 Methodology of Experiments and Results

The experiments were performed for seven large benchmark sets with nominal and real attributes from the UCI repository [2]. We used the following data Chess(36 attr., 2131 training obj., 1065 test obj.), Nursery(8 attributes, 8640 training obj., 4320 test obj.), Splice(60 attributes, 2000 training obj., 1186 test obj.), Mushrooms(23 attributes, 5461 training obj., 2708 test obj.), Abalone(9 attributes, 2785 training obj., 1392 test obj.), Thyroid(22 attributes, 4800 training obj., 2400 test obj.), Yeast(9 attributes, 990 training obj., 494 test obj.).

We have made two different comparisons. The first one concerns the comparison of reduction methods regarding the classification accuracy of the classification algorithms after reduction and reduction effectiveness. Results of these experiments are presented in Tables 2, 3 and 4. The second one concerns the new reduction methods which have been compared with the reduction methods described in [4]. The results of experiments are included in Table 14.5.

Tables 14.2, 14.3 and 14.4 consist of the comparison of testing accuracy for the classical k-NN on each data set before an after reduction (*k-NN org.* and *k-NN red.* columns). We can also see parameters used in the reduction methods (*Par, k, Perc* columns). Columns *No. of obj.* and *No. of red. obj.* denote respectively the number of objects before and after reduction. The last column in this table denotes how many objects (in percents) have not been removed from the original data set.

For the purpose of speeding up the k-NN algorithm, we prepared the data as follows: the data sets provided as a single file (*Chess, Nursery, Mushrooms, Abalone, Thyroid, Yeast*) have been randomly split into a training set and a test set with the ratio 2:1. The data set *Splice* has been tested with the originally provided partition.

For the purpose of comparing the quality of reduction methods we prepared the data as follows: the data sets provided as a single file (*Mushrooms, Abalone, Thyroid, Yeast*) have been randomly split into a training set and a test set with the ratio 4:1.

Table 14.2 Comparison of the results for *RED*1 algorithm

Data	*No. of obj.*	*No. of red. obj.*	*k-NN org.*	*k-NN red.*	*Par*	*% org.*
Chess	2131	1321	97.1	95.2	0.01	62.0
Nursery	8640	2858	99.0	94.6	0.004	33.1
Splice	2000	1401	94.0	90.4	0.1	70.0
Mushrooms	5416	3143	100.0	96.7	0.002	58.0
Abalone	2785	1328	55.0	55.5	0.003	47.7
Thyroid	4800	911	93.9	92.6	0.003	19.0
Yeast	990	502	59.5	56.9	0.091	50.7

For each data set, we have conduced several experiments with different parameters. We have used methods *Red*1, *Red*2, *Red*3 mentioned above to preprocess the

Table 14.3 Comparison of the results for *RED*2 algorithm

Data	*No. of obj.*	*No. of red. obj.*	*k-NN org.*	*k-NN red.*	*k*	*% org.*
Chess	2131	1469	97.1	94.7	25	68.9
Nursery	8640	7019	99.0	94.4	11	81.2
Splice	2000	509	94.0	90.2	63	25.5
Mushrooms	5416	1072	100.0	85.8	1800	19.8
Abalone	2785	535	55.0	54.2	3	19.2
Thyroid	4800	617	93.9	92.6	64	12.9
Yeast	990	301	59.5	57.1	2	30.4

Table 14.4 Comparison of the results for *RED*3 algorithm

Data	*No. of obj.*	*No. of red. obj.*	*k-NN org.*	*k-NN red.*	*Par*	*Perc*	*% org.*
Chess	2131	493	97.1	97.8	50	0.39	23.1
Nursery	8640	7013	99.0	95.7	50	0.48	81.2
Splice	2000	738	94.0	93.1	50	0.13	36.9
Mushrooms	5416	1954	100.0	99.7	50	0.05	36.1
Abalone	2785	501	55.0	54.8	50	0	18.0
Thyroid	4800	507	93.9	94.0	100	0	10.6
Yeast	990	743	59.5	57.7	50	0.58	75.1

data and apply them to the classical k-NN or local k-NN algorithms. All results were compared with the classical k-NN algorithm without preprocessing.

The loss of classification accuracy in all cases is smaller than 6% with the exception of data *Mushrooms* where the loss is greater. There are also a few cases where classification accuracy is greater than original one; see example results for data *Abalone*.

In Table 14.5 we can see comparison of six reduction methods. For each reduction method we can see two columns. The classification accuracy is in the first column. In the second column we can see how many objects (in percents) have not been removed from the original set. For example, if value *stor.* is 20 we have 20% of objects from the original set after reduction.

Table 14.5 Comparison of testing accuracy and storage requirements for each data set.

	Original		*Wilson*		*RT3*		*ICF*		*Red1*		*Red2*		*Red3*	
Data	Acc.	Stor.	Acc.	Stor.	Acc.	Stor.	Acc.	Stor.	Acc.	Stor.	Acc.	Stor.	Acc.	Stor.
Abalone	48.74	100	22.01	19.64	22.11	40.95	22.74	15.11	55.81	16.79	49.7	8.8	48.38	29.89
Thyroid	90.93	100	89.3	91.48	77.91	16.23	86.63	21.85	91.45	3.19	91.67	12.29	90.27	8.54
Yeast	52.7	100	55.39	52.97	55.32	27.03	52.25	16.62	50.68	25.5	53.38	10.19	47.3	31.73
Mushrooms	99.92	100	99.24	99.64	98.89	5.5	98.64	12.8	96.61	66.95	82.88	19.31	99.45	33.34

From the results given in Table 14.5 we can see that our results are at the same level or even at a higher one than the results obtained by the methods presented in [4]. In many cases we have increased the classification accuracy. If we look at column *stor.* it can be seen that our methods also give better results in many cases.

14.4 Feature Selection Methods

14.4.1 RBFS Algorithm

The *RBFS* algorithm [18] presented here is useful for selecting attributes for each classifier in multiple classifier methods. It can be applied to any kind of data with categorical or continuous values. The algorithm allows to create as many classifiers as needed, the only restriction being the number of decision-relative reducts.

For the simplicity of the algorithm, we use a few assumptions. Let $S = (U, A, d)$ be a decision table such that U is a set of objects of this table, A is a set of conditional attributes, d is a decision attribute and Tr is a training table of S. Moreover, let RED be a set of decision-relative reducts of the decision table Tr.

In the algorithm presented below we use the expressions “classification accuracy of the reduct" or “classification accuracy of the elements from set AT", where AT is the family of subsets of the attribute set A. These expressions mean that we evaluate decision subtables obtained from a given training table Tr with classifier CF. The subtables are decision tables with attributes from set $R \cup \{d\}$ where d is a decision attribute and R is a decision-relative reduct of Tr or an element from set AT. In [18] we used the leave-one-out cross validation method in the process of reduct evaluation (“classification"). Experiments showed that using the train and test method on a smaller subset of the decision table instead of the leave-one-out method on the whole decision table was much faster and gave almost identical results.

We can easily create a subtable of a given decision table using a reduct or an element from set AT. Thus, it is not important whether we talk about subtables of the decision table or subsets of the attribute set.

The algorithm as an input takes: Tr - a training set of a given decision table S, RED - a set of decision-relative reducts of Tr, CF - a classifier, $CNum$ - the number of classifiers for which we obtain the set of attribute subsets, Inc – a parameter telling us how many sets in **line with label 8** will receive a new reduct. As a result, the algorithm returns the set AT of the subsets which contain $CNum$ sets of attributes obtained by combining reducts.

In **line with label 1**, the algorithm from a given reduct set selects $CNum$ reducts which gave subtables with the highest classification accuracy; the remaining reducts are placed in set $RED1$ in **line with label 2**.

The subtables are obtained from set Tr and evaluated using classifier CF. In **line with label 4**, we take one unprocessed reduct R from the remaining set $RED1$,

which also gives the best classification accuracy. In the next line, we create a new set $TempAT$ by adding a selected reduct R to each element of set AT. The next line evaluates the new set $TempAT$ and the old one AT. We evaluate them by calculating classification accuracies. In **line with label 7**, we find Inc elements in set AT which gain most on classification accuracy after adding in **line with label 5**. In the next line, we replace the elements found in **line with label 7** by corresponding elements from set $TempAT$, so Inc elements in set AT are expanded by reduct R. Such action is performed only when all subsets gain on classification accuracy or it remains the same. Next, we repeat **line with label 4** to **line with label 8** until all reducts from set $RED1$ are processed.

The algorithm presented above allows to create subtables for classifiers from decision tables, using reducts. The number of created tables, and what is the equivalent number of classifiers, is dependent on the number of reducts in the decision table. Computational complexity of this algorithm is strongly related to the classifier used

---

Algorithm 14.5. RBFS Algorithm

---

input : Tr – a training set of S,
RED – the set of decision-relative reducts of Tr,
CF – a classifier,
$CNum$ – the number of subtables (classifiers),
Inc – an increment parameter,
$Inc<CNum$,
$TempAT$ – a temporary set of subsets of attribute set A.
output: AT – a family of subsets of attribute set A.
AT = a set of $CNum$ reducts from set RED with the highest classification accuracy.// **1**

$RED1 = RED \backslash AT$.// **2**

repeat

R = an unprocessed reduct from set $RED1$ with the highest classification accuracy. // **4**

$TempAT = \{P: P = B \cup R, B \in AT\}$. // **5**

Calculate classification accuracy of each element in sets AT and $TempAT$. // **6**

Compare accuracy of sets AT and $TempAT$ and find those Inc elements from set AT which gained most on classification accuracy. // **7**

Replace elements found in **line with label 7** with corresponding elements in set $TempAT$. (Only those which increased classification accuracy) // **8**

until *(all reducts from set RED1 are processed)*;

---

for reduct evaluation. Let n denote the number of objects in the training decision table and let r denote the number of reducts for this table and let us assume that the computational complexity of the classifier for one classifying object is $O(n^2)$, then *RBFS*, in the worst case, would have complexity $O(r^3n^3)$. The number of reducts for a decision table can be very large. It is possible to use a small subset of all reducts. One of the purposes of this paper is to present the algorithm which uses some heuristic to select a small subset of reducts.

Experiments showed that the value of the *Inc* parameter should be small and the best value is 1. Small values of the *Inc* parameter cause that *RBFS* creates good and diverse subsets of the original attribute set.

14.4.2 ARS Algorithm

Algorithm *ARS* was designed to shorten execution time of *RBFS* algorithms by decreasing the number of reducts for a given decision table. It is not required to use all available reducts. A sample of short reducts or reducts generating a small number of rules can be used however, in this paper, a set of all decision relative reducts was used for each data set.

Algorithm 14.6. Algorithm *ARS*

input : Tr – a training set,
RED – set of decision-relative reducts of Tr,
$RED2 = \emptyset$,
CF – classifier (i.e. k-NN or other classifier),
$Perc$ – percentage parameter,
dvr – diverse limit.
$\rho_{i,k} = \alpha_{i,k}$ or $\rho_{i,k} = \beta_{i,k}$ or $\rho_{i,k} = \gamma_{i,k}$
output: $RED2$ – a set of decision-relative reducts after selection.
Calculate classification accuracy of each reduct in set RED using classifier CF and table Tr.
Sort reducts in descending order using these classification accuracies.
Take first reduct r with the best classification accuracy, $RED2 = \{r\}$ and $RED = RED \setminus \{r\}$.// **2**
forall the $(r_i \in RED)$ **do**
    set counter=0
    forall the $(r_k \in RED2)$ **do**
        calculate $\rho_{i,k}$ for r_i and r_k;
        if $(\rho_{i,k} \leq dvr)$ $counter = counter + 1$;
    if $((counter/card(RED2)) \geq perc)$ **then**
        $RED2 = RED2 \cup \{r_i\}$
        $RED = RED \setminus \{r_i\}$

Algorithm *ARS* uses Eq. 14.5, Eq. 14.6, and Eq. 14.7 and a classifier to select reducts which are diverse to a certain degree. For this purpose two parameters are used: *dvr* and the percentage parameter *perc*. The parameters are used in the following way: a reduct is selected when already chosen *perc* reducts have values ρ lower than *dvr*. The first selected reduct is the reduct which provides the classifier with the best classification accuracy. ρ is the diversity measure α, β or γ.

The algorithm starts by calculating the classification accuracy of each reduct in set *RED* which is the classification accuracy of the decision table created from table *Tr* and attributes from a reduct. Classifier *CF* is used in this process. Next, reducts are sorted in the descending order, using these classification accuracies. In the third line we choose the first reduct from set *RED* which gives the best classification accuracy. We choose the best reduct because we want other selected reducts to be good as well. All reducts must differ only to a certain degree. In the loop below the **line with label 2** we check each remaining reduct in the *RED* set. If the classifier obtained from reduct r_i has ρ coefficient lower than the *dvr* parameter for more than *perc* reducts, it is added to the *RED*2 set. Parameter *dvr* causes that incoming reducts are more/less diverse. Parameter *perc* causes that we have more/less diverse reducts in the resulting set. Computational complexity of this algorithm is strongly related to the classifier used for reduct evaluation. Let n denote the number of objects in the training decision table and let r denote the number of reducts for this table, and let us assume that the computational complexity of the classifier for one classifying object is $O(n^2)$, then *ARS*, in the worst case, would have complexity $O(r^2n^3)$. However, this complexity is strongly related to the parameters *perc* and *dvr* and vary between $O(rn^3)$ and $O(r^2n^3)$.

14.4.3 Methodology of the Experiments and Results

In the experiments we used data from *UCI* repository of machine learning databases [2] to test the *RBFS* algorithm with a combination of $ARS(\alpha,\beta,\gamma)$ algorithm and the simplified bagging algorithm. The following data were used: *Glass* (214x10), *Diabetes* (768x9), *Autos* (205x26), $Breast-c$ (277x10). Objects with missing values were removed. In all experiments we used ten fold cross validation repeated five times. The average results of these experiments can be seen in Table 14.6.

Experiments for all data tables were performed by means of following schema. The *RBFS* algorithm in all experiments was tested with parameter $CNum = \{3,5,7\}$. Parameter *Inc* was set to 1 and the train and test method was used for reducts evaluation. In all cases 1, 3, 5, 7, 9-NN classifier was used with $MinkoVDM_2$ metric, as well simple multiple classifier using k-NN algorithm and majority voting was used.

Simplified bagging was performed in the following way. First, the training decision table was split randomly into 3 parts. Second, all possible pairs of decision tables from these 3 tables were created and combined. Finally, the remaining $\frac{1}{3}$ of all objects were randomly selected from the original decision table for each pair.

Table 14.6 Results for different decision tables and algorithms

Data	*Algorithm*	*Acc.*	*Red*	*/Red/*	*ExTm*	*Data*	*Acc*	*Red*	*/Red/*	*ExTm*
Diabetes	*kNN*	70.05	-	-	0.42	*Autos*	84.38	-	-	0.58
	RBFS	74.37	27	-	6.85		86.08	146.9	-	1.53
	RBFS1	**74.73**		-	20.73		**87.09**		-	7.15
	ARS α	-		16.4	3.60		-		40.3	1.57
	RBFS2 α	74.26			3.98		79.25			1.15
	RBFS3 α	**74.76**			8.99		**85.75**			3.61
	ARS β	-		12.2	3.37		-		39.5	1.52
	RBFS2 β	73.61			2.68		79.46			0.82
	RBFS3 β	**74.10**			9.41		**83.56**			2.18
	ARS γ	-		**11.5**	3.36		-		**34.5**	1.47
	RBFS2 γ	74.66			**2.45**		82.82			**0.54**
	RBFS3 γ	**75.03**			**7.41**		**85.64**			**1.77**
	FASBIR	**74.63**	-	-	-		**84.94**	-	-	-
Breast-c.	*kNN*	74.47	-	-	0.52	*Glass*	59.21	-	-	0.21
	RBFS	**75.65**	313.5	-	6.92		78.16	18.4	-	1.26
	RBFS1	75.42		-	14.68		**81.47**		-	1.72
	ARS α	-		72	3.75		-		**11.8**	0.05
	RBFS2 α	**76.97**			1.83		77.38			1.10
	RBFS3 α	76.66			6.50		**79.26**			**1.48**
	ARS β	-		81.5	3.54		-		13.5	0.04
	RBFS2 β	**76.47**			2.05		77.09			1.19
	RBFS3 β	76.04			7.46		**81.43**			1.71
	ARS γ	-		**27.8**	3.53		-		12.1	0.04
	RBFS2 γ	**76.63**			**0.72**		77.65			**1.00**
	RBFS3 γ	76.60			**1.44**		**80.78**			1.67
	FASBIR	**74.44**	-	-	-		**80.04**	-	-	-

As a result, we obtained 3 tables with the same number of objects as in the original table.

From all results obtained, the best were summarized in Table 14.6.

In Table 14.2 we use the following notation: *RBFS* - plain *RBFS* algorithm, *RBFS*1 - *RBFS* algorithm combined with simplified bagging algorithm, *RBFS*2 (α,β,γ) - *RBFS* algorithm combined with *ARS* (α,β,γ) algorithm, *RBFS*3 (α,β,γ) - *RBFS* algorithm combined with *ARS* (α,β,γ) and simplified bagging algorithms. "*Acc*" denotes classification accuracy of the algorithms, column "*Red*" denotes the average number of reducts in each approach. Column labelled as "*ExTm*" denotes the average execution time of the algorithm in seconds. */Red/* denotes the number of reducts after the use of the *ARS* algorithm.

In this table we can see that the execution time of the *RBFS* algorithm has improved in all cases after the use of the *ARS* algorithm. *ARS* allows to reduce the number of reducts to the value where classification accuracy is not drastically harmed.

The execution time of the *ARS* algorithm is also presented in Table 14.6. It is possible that the *ARS* algorithm will not be needed for data sets with a small number of reducts.

In some cases, the classification accuracy of the multiple classifier after the use of the *ARS* algorithm slightly decreased. However, the loss on classification accuracy is relatively small and can be accepted when we see an improvement of the execution time of the algorithms. This improvement could be many times greater than the execution time of the plain *RBFS* algorithm.

In Table 14.6 we can observe, that the classification accuracy of the multiple classifier which uses *RBFS* algorithms is almost always greater that the classification accuracy of the *FASBIR* algorithm. The results for the remaining methods are relatively close to the results obtained by the *FASBIR* algorithm.

From the results we can conclude that a simple bagging algorithm increased the classification accuracy in all tested data sets: however, in that case, the classification time increased.

The results showed that the best diversity measure is γ. It gives the highest reduction of the reducts set.

Initial experiments showed that for large data sets with 1000 and more objects a large number of reducts could present the main obstacle in using our methods. In such a case a classifier with small computational complexity should be used in both *RBFS* and *ARS* algorithms. Algorithm *RBFS* should also be used with small value of *CNum* parameter, algorithm *ARS* should use large values of the *Perc* parameter (close to 1). Moreover, methods which generate small sets of reducts (*e.g.* short reducts or reducts which generate a small number of rules) should be used for reduct generation.

14.4.4 Reducts Evaluation Using Lazy Algorithms

In this section we present two algorithms which use deterministic and inhibitory decision rules for reduct evaluation. In our case we actually did not need to generate all rules for a given reduct. We only needed the information if the rule existed or not. Our method used two algorithms to calculate the number of deterministic and inhibitory rules. To calculate the number of such rules for each reduct we used proposition 1 and 2 from section 2. The first was used to calculate the number of deterministic decision rules and the second was used to calculate the number of inhibitory decision rules.

The first algorithm use the value $D(T,b,v)$, $b \in V_d(T)$. This parameter is equal to the number of objects $u_i \in U$ for which there exists a rule from $Det(T)$, that is realizable for u_i and v, and has $d(x) = b$ on the right hand side. From Proposition 14.1 it follows that there exists a polynomial algorithm which for a given decision table $T = (U,A,d)$, a given object $v \in \mathcal{U}(T)$ and a given value $b \in V_d(T)$, computes the value $D(T,b,v) = |\{u_i : u_i \in U, P(u_i,v) = \{b\}\}|$.

Algorithm 14.7. Algorithm for Deterministic Rules Counting (*ADRC*)

input : T – a decision table.
output: *rullCnt* – number of deterministic decision rules.
rullCnt = 0 $\overline{U} = U$
forall the *objects v from U* **do**
  forall the *decisions b from* $V_d(T)$ **do**
    $\overline{U} = \overline{U} \setminus \{v\}$ calculate $D(T,b,v) = |\{u_i : u_i \in \overline{U}, P(u_i,v) = \{b\}\}|$ *rullCnt* = *rullCnt* + $D(T,b,v)$
  return *rullCnt*

The purpose of the algorithm is to calculate all deterministic decision rules for a given decision table. For each object $v \in U$ and decision $b \in V_d(T)$ we calculate the value of $D(T,b,v)$, which is equal to the number of rules from $Det(T)$ that are realizable for u_i and v, and has $d(x) = b$ on the right hand side. Before calculation of the D value, object v is subtracted from set $\overline{U}$. Such an action allows us to increase the speed of the algorithm and avoid double counting of the rules.

The second algorithm uses the value $I(T,b,v)$ [6], $b \in V_d(T)$. This parameter is equal to the number of objects $u_i \in U$ for which there exists a rule from $Inh(T)$, that is realizable for u_i and v, and has $d(x) \neq b$ on the right hand side. From Proposition 14.2 it follows that there exists a polynomial algorithm which, for a given decision table $T = (U,A,d)$, a given object $v \in \mathcal{U}(T)$ and a given value $b \in V_d(T)$, computes the value $I(T,b,v) = |\{u_i : u_i \in U, b \notin P(u_i,v)\}|$.

Algorithm 14.8. Algorithm for Inhibitory Rules Counting (*AIRC*)

input : T – a decision table.
output: *rullCnt* – number of inhibitory decision rules.
rullCnt = 0 $\overline{U} = U$
forall the *objects v from U* **do**
  forall the *decisions b from* $V_d(T)$ **do**
    $\overline{U} = \overline{U} \setminus \{v\}$ calculate $I(T,b,v) = |\{u_i : u_i \in \overline{U}, b \notin P(u_i,v)\}|$ *rullCnt* = *rullCnt* + $I(T,b,v)$
  return *rullCnt*

The purpose of the algorithm is to calculate all deterministic decision rules for a given decision table. For each object $v \in U$ and decision $b \in V_d(T)$ we calculate value of $I(T,b,v)$, which is equal to the number of rules from $Inh(T)$ that are realizable for u_i and v and has $d(x) = b$ on the right hand side. Before calculation of the I value object v is subtracted from set $\overline{U}$. Such an action allows us to increase the speed of the algorithm and avoid double counting of the rules.

Computation complexity of the *ADRC* and *AIRC* algorithms depends on the computational complexity of the algorithms which calculate $D(T,b,v)$ and $I(T,b,v)$

values. These values are computed in the polynomial time complexity, so the complexity of algorithms *ADRC* and *AIRC* are also polynomial.

14.4.5 Methodology of the Experiments and Results

Algorithms *ADRC* and *AIRC* were tested on several data sets from UCI Repository of Machine Learning Database Ballance-scale(625 obj., 5 attr., 6 reducts), Ecoli(336 obj., 8 attr., 25 reducts), Glass(214 obj., 9 attr., 178 reducts), Iris(150 obj., 5 attr., 9 reducts), Lymphography(148 obj., 19 attr, 3316 reducts), Zoo(101 obj., 17 attr., 590 reducts). (All reducts numbers are averages from all cross-validation folds).

The data with continuous attributes were discretized. For each data table 10-fold cross validation was performed. Each data set has three or more decision classes.

For each cross validation fold reducts were generated, and then train and test subtables were created out of those reducts. Each subtable consisted of attributes from the reduct and the decision attribute. Then each train subtable(reduct) was evaluated using *ADRC* and *AIRC* algorithms. After that algorithm *k*-NN and naive bayesian classifier were used. Algorithm *k*-NN was used with *VDM* metric and parameter $k \in \{1,3,5,7,9\}$.

Table 14.7 Results of the experiments

Classifier	*k*-NN				Naive Bayesian			
Data	SC_{det}	*SDEV*	SC_{inh}	*SDEV*	SC_{det}	*SDEV*	SC_{inh}	*SDEV*
Balance-scale	0.17	0.21	0.15	0.20	0.21	0.17	0.14	0.14
Ecoli	0.35	0.19	0.19	0.14	0.37	0.25	0.14	0.12
Glass	0.37	0.18	0.14	0.1	0.36	0.21	0.16	0.12
Iris	0.19	0.19	0.25	0.18	0.24	0.18	0.28	0.19
Lymphography	0.50	0.19	0.17	0.13	0.39	0.19	0.49	0.19
Zoo	0.31	0.24	0.14	0.11	0.29	0.21	0.47	0.23

Results obtained from algorithms *ADRC*, *AIRC*, *k*-NN, and bayesian classifier for each reduct in the data tables were scaled to the range [0, 1] using min-max normalization. After that, the results were compared using the following mean of the differences between values called *SC* (Similarity Coefficient).

$$SC = \frac{1}{n}\sum_{i=1}^{n} |e_i - f_i| \tag{14.10}$$

where $e_1,\ldots,e_i \in [0,1]$, and $f_1,\ldots,f_i \in [0,1]$ are two data sets. In this paper e_i denotes the number of deterministic or inhibitory decision rules obtained for reduct i of the particular data table (normalized to the $[0,1]$ range). f_i denotes classification accuracy obtained from classifier for reduct i (normalized to the $[0,1]$ range). Please

note that $SC \in [0,1]$. For each SC coefficient we calculated standard deviation of the differences.

All results can be seen in the Table 14.7. Where SC_{det} (SC_{inh}) denotes similarity coefficient calculated for the results obtained using the *ADRC* (*AIRC*) algorithm and classifier (k-NN or naive bayesian), *SDEV* denotes standard deviation for *SC* values.

From the results in Table 14.7 we can conclude that algorithm *AIDC* is more accurate in evaluating the usefulness of the reducts because it is more sensitive than the *ADRC* algorithm. This is caused by the fact that usually the number of inhibitory decision rules is much greater than the number of deterministic decision rules.

The most important values in Table 14.7 are standard deviations. Small values of these deviations indicate that values obtained from algorithms *AIRC* and *ADRC* are very well connected with the classification accuracy of the classifiers. Experiments showed that the large number of deterministic or inhibitory rules for decision table denotes that k-NN or naive bayesian classifier will have high classification accuracy for this table.

Experiments showed that decision tables evaluated by the *AIRC* and *ADRC* algorithms do not need to be created with the use of reducts. They can be created by a random selection of attributes from the attribute set.

14.4.6 RedBoost Algorithm

In this section we introduce an algorithm, which is used in the feature selection for multiple classifier systems. In the process the algorithm uses subsets of the attribute set, a classifier and a given training decision table. In this paper sets of decision relative and decision and object related reducts were used. The algorithm allows to decrease the number of given subsets (reducts) and increase the quality of these subsets (i.e. classifiers which use decision tables created on the basis of these subsets give better classification accuracies). The algorithm *RedBoost* takes a given reducts set, a given training decision table, a classifier and a percentage parameter that indicates how big the part of the created set of attribute subsets should remain.

The main loop of the algorithm is performed until the initial reduct set *RED* is not empty. In **line with label 3** every reduct from the reduct set *RED* is tested with the use of the classifier *CF* (from the training decision table we create subtables with conditional attributes like those in the reduct/atribute subsets and then test them with the use of the classifier *CF*). In the for loop starting below the **line with label 4** an algorithm seeks an attribute, which will be added to the created subsets of the attribute sets. In *for* loop in this loop for each conditional attribute the number of occurrences in the reducts set *RED* is calculated for each attribute (**line with label 8**) and for each attribute, the algorithm calculates the sum of classification accuracies of each reduct in which a given attribute is included (**line with label 9**). In **line with label 12** an algorithm calculates the mean value of the classification accuracy of each attribute. Using the $if \ldots then$ instruction the $maxAtr$ is found.

Algorithm 14.9. RedBoost

Input : T – training decision table with the m condition attributes.
RED - reducts set
CF - classifier
$Perc$ - $\in (0,1\rangle \subset R$.
Output: $NRED$ - set of attribute subsets of the training decision table T

```
NRED = ∅;
while (RED ≠ ∅) do
    accRED = Classif(RED, CF); //classification accuracies calculated for each
    reduct. // 3

    maxVal = 0; // 4
    forall the (attributes a ∈ A) do
        forall the (reducts R ∈ RED) do
            if (a ∈ R) then
                numAtr_a = numAtr_a + 1; // 8
                sumAcc_a = sumAcc_a + accRED_R; // 9
        avgAcc_a = sumAcc_a / numAtr_a // 12
        if (maxVal < avgAcc_a) then
            maxVal = avgAcc_a;
            maxAtr = a;
    forall the (R ∈ RED) do
        R = R ∪ maxAtr; // 20
    accRED2 = Classif(RED, CF); // 21
    forall the (R ∈ RED) do
        if (accRED2_R − accRED_R<=0) then
            NRED = NRED ∪ (R \ maxAtr);
            RED = RED \ R
NRED = RemoveWorst(NRED, 1 − Perc) // 29
```

This attribute has the biggest value of the mean classification accuracy. In the loop with **line with label 20** the best found attribute (*maxAtr*) is added to every subset from the reducts set *RED*. In **line with label 21** each new attribute subset is evaluated with the use of the classifier *CF*. In the loop below **line with label 21** the algorithm checks the difference between classification accuracy after adding attribute *maxAtr* and before adding it (see *if ... then* instruction below **line with label 21**). If the difference between these classification accuracies is less than or equal to zero (classification accuracy for such a subset decreases or remains unchanged) then the subset after removing attribute *maxAtr* is added to the result set *NRED* and it is removed from set *RED*. When the algorithm *RedBoost* finishes, the set of attribute subsets *NRED* contains subsets, which generally have higher classification

accuracies than reducts before using the *RedBoost* algorithm. The *NRED* set has a smaller number of subsets than set *RED*. This is because some of the subsets created by *RedBoost* are identical and they are removed. The algorithm also removes subsets with the smallest classification accuracy in **line 29**. Subsets which remain are percentage *Perc* of the set *NRED* (for example when $Perc = 0.2$ then 20% of the subsets will remain in set *NRED*).

If we assume that time complexity of the classifier used for reducts and subsets evaluation is $O(n^3)$ (for one reduct/subset) where n is the number of objects in the training decision table and r is the number of reducts, then the time complexity of the *RedBoost* algorithm in the worst case is $O(r^2n^3)$.

14.4.7 Methodology of the Experiments and Results

Many experiments were performed using the *RedBoost* algorithm. In the experiments we used decision relative reducts and object and decision relative reducts. In all the experiments a simple multiple classifier was used. The results of these experiments were compared with the results obtained with the same multiple classifier but different feature selection methods ($RBFS$, $RBFS1$, $RBFS2$, $RBFS3$ [8]), a single k-NN algorithm and the *FASBIR* multiple classifier. The experiments were performed on the data from *UCI* Machine Learning Databases [2]. Data tables that have been used are *Diabetes*, *Glass*, *Autos*, *Breast* $- c$. For each data table 10–fold cross–validation method was used. The results of the experiments are in Table 14.8. In the tables, column *Acc* denotes classification accuracy, $Redboost(1)$ denotes results for the *RedBoost* algorithm which uses decision relative reducts and $RedBoost(2)$ denotes results for the *RedBoost* algorithm which uses decision and object relative reducts, column *num. of red.* denotes the number of reducts used.

Table 14.8 Results obtained with *RedBoost* and other algorithms for all data sets

Data table	Diabetes		Glass		Autos		Breast-c	
Algorithm	*Acc*	*num. of red.*	*Acc*	*num. of red.*	*Acc*	*num. of red.*	*Acc*	*num. of red.*
k-NN	70.05	-	74.47	-	84.38	-	59.21	-
$RBFS$	74.37	27	75.65	313.5	86.08	146.9	78.16	18.4
$RBFS1$	74.73	27	75.42	313.5	87.08	146.9	81.47	18.4
$RBFS2$	74.66	12.2	76.97	72	82.82	34.5	77.65	12.1
$RBFS3$	75.03	11.5	76.66	72	85.75	40.3	81.47	11.8
FASBIR	74.63	-	74.44	-	84.94	-	80.04	-
$RedBoost(1)$	74.02	11	77.24	24.4	88.25	27.2	76.19	10.3
$RedBoost(2)$	75.32	11	76.90	17.4	89.75	17.4	81.81	11.2

We can conclude from the experiments results that in many cases algorithm *RedBoost* gave better or similar quality feature subsets to $RBFS$ algorithms. The

Redboost algorithm combined with a simple multiple classifier in many cases gave better results than the *FASBIR* algorithm. For all data tables, *RedBoost* combined with the simple multiple classifier gave better results than the single k-NN classifier.

The experiments also showed that parameter *Perc* should have a value such that the resulting feature subset generated by the *RedBoost* algorithm should have 10-20 feature subsets.

14.5 MC2 Multiple Classiffier System

For the *MC*2 algorithm we also have training and test tables split into n parts $\{Tr_1, \ldots, Tr_n\}$ and $\{Ts_1, \ldots, Ts_n\}$. This stage is critical to the classification. The parts of training and test tables must be chosen properly.

Algorithm 14.10. Algorithm *MC*2

Input : $\{Tr_1, \ldots, Tr_n\}$ and $\{Ts_1, \ldots, Ts_n\}$ are training and test subtables, respectively.
$CF_1, CF_2, \ldots, CF_n$ - classifiers (one of the algorithms: k-NN, decision rules, LTF-c neural network or decomposition tree),
n - the number of classifiers.

Output: *Test* - a test table classified by algorithm.

Step 1. Classify objects from training subtables $\{Tr_1, \ldots, Tr_n\}$ using leave-one-out cross validation method with one of the classifiers (k-NN, *decision rules*, $LTF-c$ neural network or *decomposition tree*). As a result, we obtain n subtables $\{Tr'_1, \ldots, Tr'_n\}$ with new decisions.

Step 2. Use $\{Tr_1, \ldots, Tr_n\}$ subtables to classify *Test* subtables $\{Ts_1, \ldots, Ts_n\}$ with one of the classifiers (k-NN, *decision rules*, $LTF-c$ neural network or *decomposition tree*). As a result, we obtain n subtables $\{Ts'_1, \ldots, Ts'_n\}$.

Step 3. Create a new training table by combining decision attributes from new training subtables and the original training set
$\{dec(Tr'_1), \ldots, dec(Tr'_n), dec(Original\ Training\ set)\}$.

Step 4. Create a new test table by combining decision attributes from new test subtables $\{dec(Ts'_1), \ldots, dec(Ts'_n)\}$.

Step 5. (Optional): Classify the new test table using (k-NN, *decision rules*, $LTF-c$ neural network or *decomposition tree*) and the new training table.

The next part of the *MC*2 algorithm is the creation of new training and test tables. We can use n classifiers (k-NN, *decision rules*, $LTF-c$ neural network or *decomposition tree*) to create these tables (in our experiments, we used k-NN to prepare training and test sets). The new training table is obtained from parts $\{Tr_1, \ldots, Tr_n\}$ through classification. We use the leave-one-out cross validation method to classify each object from training subtables. As a result, we obtain n

decision attributes (from each subtable). Test tables are obtained in a similar way with one exception. We use original subtables $\{Tr_1, \ldots, Tr_n\}$ to classify test subtables $\{Ts_1, \ldots, Ts_n\}$.

As a result of the *MC*2 algorithm we receive two new training and test sets. We can apply any classifier to such data. In the algorithm above, we included the optional **Step 4** to illustrate this idea.

We also use weights to increase the efficiency of the algorithm. Weights are applied to a new training set. Each conditional attribute has a weight which tells us how important this attribute is. The weighing procedure assigns weight to each conditional attribute. Attributes which are more compatible with the decision have higher weights. In our procedure, a weight is increased when the corresponding classifier gives a correct answer, otherwise the weight is decreased, taking into account the given class information of the object from the training set. Hence the weighing procedure uses this information to check whether this attribute is more or less compatible.

The *MC*2 Classifier method is a reduction method. It allows to reduce the size of the training and test data. Most reduction methods decrease the number of objects, see *e.g.*, [7]. By using this method we can reduce the number of conditional attributes instead of the number of objects by simply increasing the number of attributes taken to each classifier.

When we apply a classifier to algorithm **Step 4**, the *MC*2 method becomes closely related to the stacked generalization [23].

References

1. Bazan, J.G., Nguyen, H.S., Skowron, A., Szczuka, M.: A view on rough set concept approximation. In: Wang, M., et al. [19], pp. 181–188.
2. Blake, C., Merz, C.: UCI Repository of Machine Learning Data Bases, Department of Information and Computer Science, University of California, Irvine, CA (1998), `http://www.ics.uci.edu/mlearn/mlrepository.html`
3. Breiman, L.: Bagging Predictors. Machine Learning 24(2), 123–140 (1996)
4. Brighton, H., Mellish, C.: Advances in Instance Selection for Instance-Based Learning Algorithms. Data Mining and Knowledge Discovery 6, 153–172 (2002)
5. Cios, K., Pedrycz, W., Świniarski, R.: Data Mining. In: Methods for Knowledge Discovery. Kluwer, Dordrecht (1998)
6. Delimata, P., Moshkov, M., Skowron, A., Suraj, Z.: Inhibitory Rules in Data Analysis. A Rough Set Approach. SCI, vol. 163. Springer, Heidelberg (2008)
7. Delimata, P., Suraj, Z.: On *k*-NN Method with Preprocessing. Fundamenta Informaticae 69(3), 343–358 (2005)
8. Delimata, P., Suraj, Z.: Feature selection algorithm for multiple classifier systems: A hybrid approach. Fundamenta Informaticae 85(1-4), 97–110 (2008)
9. Góra, G., Wojna, A.: Riona: a new classification system combining rule induction and instance-based learning. Fundamenta Informaticae 51, 369–390 (2002)
10. Nguyen Hoa, S.: Data regularity analysis and applications in data mining. Ph.D. Thesis, Warsaw University, Warsaw, Poland (1999)
11. Pawlak, Z.: Rough sets. International Journal of Computer and Information Sciences 11, 341–356 (1982)

12. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning About Data. Kluwer, Dordrecht (1991)
13. Pawlak, Z., Skowron, A.: Rough sets and boolean reasoning. Information Sciences 177(1), 41–73 (2007)
14. Pawlak, Z., Skowron, A.: Rough sets: Some extensions. Information Sciences 177(1), 28–40 (2007)
15. Peters, J.F., Skowron, A., Grzymała-Busse, J.W., Kostek, B., Świniarski, R.W., Szczuka, M.S. (eds.): Transactions on Rough Sets I. LNCS, vol. 3100. Springer, Heidelberg (2004)
16. Randall, W., Martinez, T.: Reduction techniques for instance-based learning algorithms. Machine Learning 38(3), 257–286 (2000)
17. Stanfill, C., Waltz, D.: Toward memory - based reasonning. Communications of the ACM 29(12), 1213–1228 (1986)
18. Suraj, Z., Gayar, N., Delimata, P.: A rough set approach to multiple classifier systems. Fundamenta Informaticae 72(1-3), 393–406 (2006)
19. Wang, G., Liu, Q., Yao, Y., Skowron, A. (eds.): RSFDGrC 2003. LNCS (LNAI), vol. 2639. Springer, Heidelberg (2003)
20. Wilson, D.: Asymptotic properties of nearest neighbor rules using edited data. IEEE Transactions on Systems, Man and Cybernetics 2-3, 408–421 (1972)
21. Wojna, A.: Analogy-Based Reasoning in Classifier Construction. In: Peters, J.F., Skowron, A. (eds.) Transactions on Rough Sets IV. LNCS, vol. 3700, pp. 277–374. Springer, Heidelberg (2005)
22. Wojnarski, M.: LTF-c: Architecture, training algorithm and applications of new neural classifier. Fundamenta Informaticae 54(1), 89–105 (2003)
23. Wolpert, D.: Stacked generalisation. Neural Networks 5, 241–259 (1992)
24. Yule, G.: On the association of attributes in statistics. Phil. Trans. 194, 257–319 (1900)
25. Zhou, Z.H., Yu, Y.: Ensembling local learners through multimodal perturbation. IEEE Transactions on Systems, Man and Cybernetics Part B 35(4), 725–735 (2005)

Chapter 15
Uncertainty Problem Processing with Covering Generalized Rough Sets

Jun Hu and Guoyin Wang

Abstract. Rough set theory is useful in processing uncertainty problems. Because the limitation of classical rough set theory, many extensions have been developed. Covering generalized rough set model is an important one of them. Although many theoretical results have been achieved in the past years. However, the application of covering generalized rough set theory is discussed rarely. In this chapter, two typical uncertainty problems, incomplete information processing, and fuzzy decision making, are discussed from the view of covering generalized rough set theory.

Keywords: Uncertainty problem processing, covering generalized rough set, incomplete information system, knowledge reduction, fuzzy decision making, fuzzy rough set.

15.1 Introduction

Uncertainty is widely exist in real world. Randomicity and fuzziness are two kinds of uncertainties which are familiar to us [18]. To solve the problems with randomicity or fuzziness, probability theory and fuzzy set theory were proposed respectively. Different from these two uncertainties, roughness is another kind of uncertainty, which wins wide attention in recent years. Generally, roughness is induced by the incompleteness or imprecision of knowledge. To address the roughness in problems,

Jun Hu
Institute of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing, P. R. China
e-mail: hujun@cqupt.edu.cn

Guoyin Wang
Institute of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing, P. R. China
e-mail: wanggy@cqupt.edu.cn

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 293–308.
springerlink.com

rough set theory was developed by Pawlak in 1982 [24]. Since then, it has been widely applied in many fields, such as machine learning, data mining, and pattern recognition.

The classical rough set theory is based on indiscernibility relation. Given an indiscernibility relation, one can get a partition, where each element in the partition is called an equivalence class. And then, a subset of the universe which can be represented by a union of some equivalence classes is called a definable set. For an arbitrary subset of the universe, one can use two definable sets called lower approximation and upper approximation to approximate it. To be specific, the lower approximation is the maximal definable set included in it, and the upper approximation is the minimal definable set including it. This is the basic idea of classical rough sets.

Unfortunately, it might be impossible or cost too much to acquire indiscernibility relation in some cases. Thus, it is necessary to extend classical rough set theory to general cases. As we know, an equivalence relation determines a partition on a universe, and vice versa. However, there is no such one-to-one relationship between coverings and general binary relations on a universe. So, rough set model is extended in two different views. One is from the view of general relation [12] [15] [16] [26] [30] [27] [28], and the other is from the view of covering [36] [1] [22] [37] [38] [29] [11] [6] [31] [33] [19] [4] [39] [40] [34] [41] [42] [43] [7].

Although, covering generalized rough set theory was proposed by Zakowski in 1983 [36], it does not get much attention until recent years. Here we can classified all the works about covering generalized rough set theory into four aspects:

(1) The definition of rough operator. Bonikowski gave a concept called minimal description, and defined a pair of rough operators based on it [1]. Latter, other three different definitions were developed from different points [22] [29] [40].

(2) The reduction of covering. Zhu studied the reduction of covering, and found the condition in which two coverings generate the same covering lower and upper approximations [37] [38]. Hu proposed the definition of relative reduction of covering, and proofed that a covering and its relative reduction have the same approximation ability [7].

(3) The uncertainty measure of covering generalized rough sets. To characterize the uncertainty of covering generalized rough sets, many methods have been developed, such as rough entropy and fuzzy degree [11] [6] [34], which provide tools for quantitatively analyzing the approximation ability of covering.

(4) The combination of rough set theory with other theories. To address some complex problems, it is necessary to combine rough set theory with other theories. For example, covering generalized rough fuzzy set model has been developed to solve the problems with both roughness and fuzziness [31] [33] [4] [41].

This chapter focuses on how to use covering generalized rough set theory to address some uncertainty problems. The remainder of this chapter is arranged as follows. The preliminary of rough set theory is recalled in Section 15.2. In Section 15.3, the knowledge reduction model of covering is introduced firstly, and then is applied to process incomplete information system. A covering generalized rough fuzzy set model is developed to address problems with both roughness and fuzziness in Section 15.4. Section 15.5 concludes this chapter.

15.2 Preliminary of Rough Set Theory

U is a finite and non-empty set called the universe, and $R \subseteq U \times U$ is an equivalence relation on U. Then, the ordered pair (U,R) is called a Pawlak approximation space [25]. The equivalence relation R partitions the universe U into disjoint subsets, and each subset is called an equivalence class. Such a partition of the universe is denoted by U/R. If two elements x and y of U are in the same equivalence class, we say that x and y are indistinguishable. The equivalence classes induced by R, and the empty set $\emptyset$ are called the elementary sets in the approximation space (U,R).

The equivalence relation and the induced partition are regarded as the available information or knowledge about the objects under consideration. Given an arbitrary set $X \subseteq U$, if it is the union of one or more elementary sets, then it is definable. Otherwise, it is undefinable. That is, the available information is not sufficient to give a precise representation of X. In this case, one may characterize X using a pair of sets called the lower and upper approximations:

$$\underline{R}(X) = \cup\{Y \in U/R | Y \subseteq X\} \tag{15.1}$$

$$\overline{R}(X) = \cup\{Y \in U/R | Y \cap X \neq \emptyset\} \tag{15.2}$$

The lower approximation $\underline{R}(X)$ is the union of all the elementary sets which are subsets of X. It is the largest definable set contained in X. The upper approximation $\overline{R}(X)$ is the union of all the elementary sets which have a non-empty intersection with X. It is the smallest definable set containing X.

Let C be a family of subsets of U. If none subset of C is empty, and $\cup C = U$, C is called a covering of U. The ordered pair (U,C) is called a covering approximation space. Let x be an object of U, the set family $Ad(x) = \{K \in C | x \in K\}$ is called the description of x, and the set family $Md(x) = \{K \in C | x \in K \wedge \forall S \in C(x \in S \wedge S \subseteq K \Rightarrow K = S)\}$ is called the minimal description of x.

For a set $X (X \subseteq U)$, the same as classical rough set theory, one can characterize it by a pair of sets, called the covering lower and upper approximations. From different views, four types of definitions have been proposed [8]. They are:

$$CL(X) = \cup\{K | K \in C \wedge K \subseteq X\} \tag{15.3}$$

$$FH(X) = CL(X) \cup \{\cup Md(x) | x \in X - CL(X)\} \tag{15.4}$$

$$SH(X) = \cup\{K | K \in C \wedge K \cap X \neq \phi\} \tag{15.5}$$

$$TH(X) = \cup\{\cup Md(x) | x \in X\} \tag{15.6}$$

$$RH(X) = CL(X) \cup \{K | K \in C \wedge K \cap (X - CL(X)) \neq \phi\} \tag{15.7}$$

Where CL is the covering lower approximation, and FH, SH, TH and RH are the first, the second, the third, and the fourth types of covering upper approximation respectively.

For the convenience of discussion, we use $C_*(X)$ to represent the covering lower approximation, and $C^*(X)$ to represent the covering upper approximation, which could be any one of upper approximations introduced above.

15.3 Incomplete Information System Processing with Covering Generalized Rough Sets

In many problems, we cannot get all attribute values of every object for some reasons. This may result in the description of some objects as incomplete. We call the information systems, which have unknown attribute values, incomplete information systems. Incomplete information processing is a typical uncertainty problem.

Generally, there are two methods for incomplete information system processing. One is to transform an incomplete information system into a complete information system, then process it with general methods. The other is to extend general methods, then use it to process incomplete information system directly. Because the former method may result in change or reduction of information, people tend to use the later one to process incomplete information system.

In recent years, many extended models have been developed, such as tolerance relation based rough set model [15] [16], similarity relation based rough set model [27] [28], limited tolerance relation based rough set model [30], and characteristic relation based rough set model [5] [20]. We can find that all of them are from the view of relation. Here, we will discuss this problem in the view of covering. In the following, we will firstly develop a knowledge reduction model, and then use it to process incomplete information system.

15.3.1 Knowledge Reduction Model of Covering Approximation Space

In order to find the condition in which two different coverings generate the same covering lower and upper approximations, Zhu gave the definition of reducible element and irreducible element of a covering [38].

Definition 15.1. Let C be a covering of a universe U, K be an element of C. If K is a union of some sets of $C - \{K\}$, we say K is a reducible element of C, otherwise an irreducible element of C.

Definition 15.2. Let C be a covering of U. If every element of C is irreducible, we say C is irreducible, otherwise, C is reducible.

Proposition 15.1. *Let C be a covering of U, K be a reducible element of C, and K_1 be another element of C, then K_1 is a reducible element of C if and only if it is a reducible element of $C - \{K\}$.*

Definition 15.3. For a covering C of U, the new irreducible covering through eliminating the reducible elements of C is called the reduction of C, denoted by $red(C)$.

Example 15.1. Let $U = \{x_1, x_2, x_3, x_4, x_5\}$, $C = \{K_1, K_2, K_3, K_4\}$, where $K_1 = \{x_1, x_2, x_3\}$, $K_2 = \{x_1, x_2\}$, $K_3 = \{x_2, x_3\}$, and $K_4 = \{x_3, x_4, x_5\}$.

According to the definition above, K_1 is reducible because $K_1 = K_2 \cup K_3$, and other elements of C are not reducible. So, $red(C) = \{K_2, K_3, K_4\}$.

Because the reduction does not rely on other conditions, we call it absolute reduction.

Proposition 15.2. *Let C be a covering of U, $red(C)$ be the absolute reduction of C, then C and $red(C)$ generate the same covering lower approximations.*

Proposition 15.3. *Let C be a covering of U, $red(C)$ be the absolute reduction of C, then C and $red(C)$ generate the same first type of covering upper approximations.*

Proposition 15.4. *Let C be a covering of U, $red(C)$ be the absolute reduction of C, then C and $red(C)$ generate the same third type of covering upper approximations.*

However, a covering and its absolute reduction may generate different results for the second and the fourth types of covering upper approximations.

From the above introduction, we can find that the absolute reduction of a covering generates the same covering lower approximation, the same first and third types of covering upper approximations as the given covering. That is, it can simplify a covering by eliminating all the reducible elements of it, but does not decrease its approximation ability.

In many cases, we want covering to be as simple as possible. Then, for a given concept, there are other redundant elements in a covering besides the absolutely redundant elements? What is the condition in which, two coverings could generate the same covering lower and upper approximations for a given concept? In the following, we will discuss these problems.

Definition 15.4. Let C be a covering of U, X be a subset of U, and K be an element of C. If there is another element K' of C such that $K \subseteq K' \subseteq X$, then K is a relatively reducible element of C with respect to X, or a relatively reducible element for short, otherwise, a relatively irreducible element.

Definition 15.5. Let C be a covering of U, X be a subset of U. If every element of C is relatively irreducible, then C is irreducible with respect to X, otherwise, C is reducible with respect to X.

Proposition 15.5. *Let C be a covering of U, X be a subset of U. If K is a relatively reducible element of C with respect to X, $C - \{K\}$ is still a covering of U.*

Proposition 15.6. *Let C be a covering of U, X be a subset of U. If K is a reducible element of C with respect to X, and K' is another element of C, then K' is a relatively reducible element of C with respect to X if and only if it is a relatively reducible element of $C - \{K\}$ with respect to X.*

Proof. If $K^{'}$ is a relatively reducible element of $C-\{K\}$ with respect to X, then there should be an element $K_1 \in C-\{K,K^{'}\}$ such that $K^{'} \subseteq K_1 \subseteq X$. Obviously, $K_1 \in C-\{K\}$, so $K^{'}$ is also a reducible element of C with respect to X. On the other hand, if $K^{'}$ is a relatively reducible element of C with respect to X, then there should be an element $K_1 \in C-\{K^{'}\}$ such that $K^{'} \subseteq K_1 \subseteq X$. If $K_1 \neq K$, then $K_1 \in C-\{K,K^{'}\}$. So, $K^{'}$ is a relatively reducible element of $C-\{K\}$ with respect to X. Otherwise, if $K_1 = K$, since K is a reducible element of C with respect to X, then there should be an element $K_2 \in C-\{K\}$ such that $K \subseteq K_2 \subseteq X$. Thus, $K^{'} \subseteq K_2 \subseteq X$, so $K^{'}$ is also a relatively reducible element of $C-\{K\}$ with respect to X. □

Proposition 15.5 guarantees that it is still a covering after a relatively reducible element of a covering is eliminated. Proposition 15.6 indicates that it will not generate any new relatively reducible elements or make any other originally relatively reducible elements become relatively irreducible after a relatively reducible element of a covering is eliminated.

Definition 15.6. Let C be a covering of U, X be a subset of U. The new irreducible covering through eliminating all relatively reducible elements is called the relative reduction of C with respect to X, denoted by $red_X(C)$.

It can be proofed that each covering has only one relative reduction with respect to a given concept.

Example 15.2. Let $U = \{x_1,x_2,...,x_6\}$, and $C = \{K_1,K_2,K_3,K_4\}$, where $K_1 = \{x_1, x_2,x_3\}$, $K_2 = \{x_3,x_4\}$, $K_3 = \{x_3,x_4,x_5\}$, and $K_4 = \{x_1,x_6\}$.

For $X = \{x_3,x_4,x_5,x_6\}$, K_2 is relative reducible with respect to C because $K_2 \subseteq K_3 \subseteq X$, and other elements of C are not relatively reducible. So, $red_X(C) = \{K_1,K_3,K_4\}$.

Proposition 15.7. *Let C be a covering of U, X be a subset of U. If $K(K \in C)$ is a relatively reducible element, then the covering lower approximations of X generated by C and $C-\{K\}$ are the same.*

Proof. Suppose the covering lower approximations of X generated by C and $C-\{K\}$ are X_*^1 and X_*^2 respectively. From the definition of the covering lower approximation, it is evident $X_*^2 \subseteq X_*^1 \subseteq X$. If $x \in X_*^1$, then $\exists_{x \in K_2 \in C}(K_2 \subseteq X)$. If $K_2 \neq K$, then $x \in X_*^2$. If $K_2 = K$, since K is a relatively reducible element with respect to X, there should be an element $K_1 (K_1 \in C-\{K\})$ such that $K \subseteq K_1 \subseteq X$, then $x \in X_*^2$ too. Therefore, $X_*^1 \subseteq X_*^2$. So, $X_*^1 = X_*^2$, namely Proposition 15.7 holds. □

Corollary 15.1. *Let C be a covering of U, X be a subset of U, and $red_X(C)$ be a relative reduction of C with respect to X, then the covering lower approximations of X generated by C and $red_X(C)$ are the same.*

Proposition 15.8. *Let C be a covering of U, X be a subset of U. If $K(K \in C)$ is a relatively reducible element, then the covering upper approximations of X generated by C and $C-\{K\}$ are the same.*

Proof. Since K is a relatively reducible element with respect to X, there should be an element $K_1 (K_1 \in C - \{K\})$ such that $K \subseteq K_1 \subseteq X$. Therefore, we have $K \subseteq K_1 \subseteq C_*(X) \subseteq C^*(X)$ no matter which type of covering upper approximation it is. So, it does not reduce the covering upper approximation after a relatively reducible element of a covering is eliminated, namely Proposition 15.8 holds. □

Corollary 15.2. *Let C be a covering of U, X be a subset of U, and $red_X(C)$ be the relative reduction of C with respect to X, then the covering upper approximations of X generated by C and $red_X(C)$ are the same.*

Example 15.3. Let $U = \{x_1, x_2, x_3, x_4\}$, and $C = \{K_1, K_2, K_3\}$, where $K_1 = \{x_1\}$, $K_2 = \{x_1, x_2\}$, $K_3 = \{x_2, x_3, x_4\}$.

Let $X = \{x_1, x_2, x_3\}$. Since $K_1 \subseteq K_2 \subseteq X$, K_1 is a relatively reducible element of C with respect to X. So, $red_X(C) = \{K_2, K_3\}$. Then, $CL(X) = \{x_1, x_2\}$, $FH(X) = \{x_1, x_2, x_3, x_4\}$, and $TH(X) = \{x_1, x_2, x_3, x_4\}$ both in C and $red_X(C)$.

However, the covering lower approximation and upper approximations may be different for other concepts in Example 15.3. For example, let $Y = \{x_1\}$, the covering lower approximation of Y is $\{x_1\}$ in C, but the covering lower approximation of Y is $\emptyset$ in $red_X(C)$. So, the relative reduction of a covering with respect to a given concept has the same approximation ability for the given concept, but it maybe has different approximation ability for other concepts.

The purpose of data mining is to acquire knowledge (rules) from large data sets. In general, the more concise the rules are, the better the rules will be. Therefore, knowledge reduction is an important step in knowledge acquisition.

Let U be a universe of discourse, P be a family of nonempty subsets of U. If $\forall_{p_1, p_2 \in P}(p_1 \neq p_2 \Rightarrow p_1 \cap p_2 = \phi)$, and

$$\bigcup_{p_i \in P} p_i = U$$

then P is called a partition of U. Here, we also call it a decision.

Definition 15.7. Let C be a covering of U, and P be a decision. For an element $K (K \in C)$, if $\exists_{p_i \in P}(K \subseteq p_i)$, we call K a consistent granule with respect to P, or a consistent granule for short, otherwise, an inconsistent granule.

In knowledge acquisition, rules are often used to represent knowledge. For example, $p \rightarrow q$ is a rule, and $|p \cap q|/|U|$ (where $|.|$ is the cardinality of a set) is called its support degree. Generally speaking, a rule with higher support degree would be more useful than a lower one.

Let C be a covering of U, P be a decision. For two granules $K, K' (K, K' \in C)$, if $\exists_{p_i \in P}(K \subseteq K' \subseteq p_i)$, then both K and K' are consistent granules. However, we say rule $K' \rightarrow p_i$ is more useful than rule $K \rightarrow p_i$ for its higher support degree, while rule $K \rightarrow p_i$ is less useful. Therefore, for a consistent granule, if there is another consistent granule including it, then the rule generated by it is reducible. So, a consistent granule with lower support degree is a relatively reducible element.

In the other hand, for a granule $K(K \in C)$, suppose K is an inconsistent granule, and

$$K = \bigcup_{k_i \in C - \{K\}} k_i$$

where k_i is an absolute irreducible element of C. For a granule k_i and an element x of it, if $\exists_{p_i \in P}(k_i \subseteq p_i)$, then k_i is a consistent granule, and x can be classified into p_i definitely according to rule $k_i \to p_i$. Otherwise, k_i is an inconsistent granule. We know that $k_i \subseteq K$ and K is not in $Md(x)$. If $x \in p_i$, then x can be classified into p_i approximately by rule $k_i \to p_i$. Therefore, for an inconsistent granule, if it is a union of some other granules, the rule generated by it will also be reducible. Apparently, an inconsistent granule, which is a union of some other granules, is an absolute redundant element.

Definition 15.8. Let U be a universe, C be a covering of U, and P be a decision. If $K(K \in C)$ is relatively reducible with respect to any $p_i(p_i \in P)$, then K is relatively reducible with respect to P. Otherwise, K is relatively irreducible with respect to P.

Definition 15.9. Let U be a universe, C be a covering of U, and P be a decision. If every element of C is relatively irreducible with respect to P, C is relatively irreducible with respect to P. Otherwise, C is relatively reducible with respect to P.

Definition 15.10. Let U be a universe, C be a covering of U, and P be a decision. If any element of C is relatively reducible with respect to P, then delete it from C until, C is relatively irreducible with respect to P. Thus, one will get a relative reduction of C with respect to P, denoted by $red_P(C)$.

Definition 15.11. Let U be a universe, C be a covering of U, and P be a decision. The positive region and boundary region of C with respect to P are defined as follows:

$$Pos_C(P) = \bigcup_{p_i \in P} C_*(p_i) \tag{15.8}$$

$$BN_C(P) = \bigcup_{p_i \in P} C^*(p_i) - \bigcup_{p_i \in P} C_*(p_i) \tag{15.9}$$

Where $Pos_C(P)$ is a set composed by all elements which could be classified definitely in (U,C), and $BN_C(P)$ is the complement of $Pos_C(P)$. That is, $BN_C(P) = U - Pos_C(P)$. Since the definitions of covering lower approximation of four types of covering generalized rough set models are the same, the positive regions and boundary regions of C with respect to P are also the same in these models.

Proposition 15.9. *Let C be a covering of U, P be a decision, and $red(C)$ be the absolute reduction of C, then $red(C)$ and C generate the same positive regions and boundary regions.*

Proposition 15.10. *Let C be a covering of U, P be a decision, and $red_P(C)$ be the relative reduction of C, then $red_P(C)$ and C generate the same positive regions and boundary regions.*

Proposition 15.9 and Proposition 15.10 show that both absolute reduction and relative reduction do not reduce the positive region, or change the boundary region, namely the classification ability is not reduced.

From the above discussion, we can develop a diagram for knowledge reduction of covering approximation space shown in Fig. 15.1.

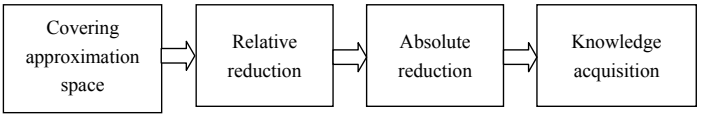


Fig. 15.1 knowledge reduction of covering approximation space

The algorithm for knowledge reduction of covering approximation space is as follows:

Algorithm 1 (Knowledge Reduction of Covering Approximation Space)
Input: An covering approximation space (U,C), and a decision P.
Output: $reduct(C|P)$.
Step 1: Let $reduct(C|P) = C$.
Step 2: If $C \neq \phi$, then choose an element $K(K \in C)$, and let $C = C - \{K\}$. Otherwise, let $T = reduct(C|P)$, and go to Step 4.
Step 3: If K is a relatively reducible element of $reduct(C|P)$ with respect to P, then let $reduct(C|P) = reduct(C|P) - \{K\}$. Go to Step 2.
Step 4: If $T \neq \phi$, then choose an element $K(K \in T)$, and let $T = T - \{K\}$. Otherwise, go to Step 6.
Step 5: If K is an absolutely reducible element of $reduct(C|P)$, then let $reduct(C|P) = reduct(C|P) - \{K\}$. Go to Step 4.
Step 6: Output $reduct(C|P)$.

Example 15.4. Let $U = \{x_1,x_2,x_3,x_4,x_5\}$, and $C = \{K_1,K_2,K_3,K_4,K_5,K_6,K_7\}$, $K_1 = \{x_1\}$, $K_2 = \{x_1,x_2,x_3\}$, $K_3 = \{x_3,x_4\}$, $K_4 = \{x_3\}$, $K_5 = \{x_4\}$, $K_6 = \{x_5\}$, $K_7 = \{x_1,x_5\}$. Suppose $P = \{p_1,p_2\}$, where $p_1 = \{x_1,x_2\}$, $p_2 = \{x_3,x_4,x_5\}$.

Since $K_4 \subseteq K_3 \subseteq p_2$, K_4 is relatively reducible with respect to P, and K_5 is also relatively reducible with respect to P for $K_5 \subseteq K_3 \subseteq p_2$. Moreover, because $K_7 = K_1 \cup K_6$, K_7 is absolutely reducible. Therefore, $reduct(C|P) = \{K_1,K_2,K_3,K_6\}$. Moreover, we have

$Pos_C(P) = Pos_{reduct(C|P)}(P) = \{x_1,x_3,x_4,x_5\}$,
$BN_C(P) = BN_{reduct(C|P)}(P) = \{x_2\}$.

In Algorithm 1, relatively redundant elements are reduced at first, and absolutely redundant elements are reduced then. If the absolutely redundant elements are reduced firstly, K_3 and K_7 will be reduced in Example 15.4. Then, the originally relatively redundant elements, K_4 and K_5, will become relatively irreducible. In that way, we need one more rule with lower support degree to cover the objects x_3 and x_4. So, these two steps can not be exchanged.

15.3.2 An Example

There are two kinds of semantics of unknown values in incomplete information system. One is missing, and the other is absent. Next, we will use the reduction model proposed above to address an incomplete information system with missing value and absent value. The incomplete information system is shown as Table 15.1, where "*" denotes missing value, and "♯" denotes absent value.

Table 15.1 An incomplete information table

O	a_1	a_2	a_3	a_4	d
x_1	1	*	1	1	φ
x_2	1	1	♯	1	ψ
x_3	1	2	2	2	φ
x_4	2	1	2	♯	ψ
x_5	2	1	3	3	ψ
x_6	3	2	*	2	φ
x_7	2	1	3	3	ψ

$a(x)$ denotes the value of an object x, on an attribute a, and $(a_1,1)$ denotes the set formed by all objects whose attribute values are 1 on attribute a_1. For a_1, there are no unknown values. We have $(a_1,1)=\{x_1,x_2,x_3\}$, $(a_1,2)=\{x_4,x_5,x_7\}$, and $(a_1,3)=\{x_6\}$. For a_2, there is a missing value. We have $(a_2,1)=\{x_1,x_2,x_4,x_5,x_7\}$ and $(a_2,2)=\{x_1,x_3,x_6\}$. For a_4, there is an absent value. We have $(a_4,1)=\{x_1,x_2\}$, $(a_4,2)=\{x_3,x_6\}$ and $(a_4,3)=\{x_5,x_7\}$. For a_3, there are both missing value and absent value. We have $(a_3,1)=\{x_1,x_6\}$, $(a_3,2)=\{x_3,x_4,x_6\}$ and $(a_3,3)=\{x_5,x_6,x_7\}$. In addition, for d, we have $(d,\varphi)=\{x_1,x_3,x_6\}$ and $(d,\psi)=\{x_2,x_4,x_5,x_7\}$.

Obviously, all the sets induced by $\{a_1,a_2,a_3,a_4\}$ form a covering $C=\{(a_1,1),(a_1,2),(a_1,3),(a_2,1),(a_2,2),(a_3,1),(a_3,2),(a_3,3),(a_4,1),(a_4,2),(a_4,3)\}$. In addition, all the sets induced by d form a partition $P=\{(d,\varphi),(d,\psi)\}$. Then, the given incomplete information system is transformed into a classification problem, based on a covering approximation space.

According to the definition of positive region and boundary region, we have
$Pos_C(P) = \{x_1, x_3, x_4, x_5, x_6, x_7\}$,
$BN_C(P) = \{x_2\}$.
That is, the decision P is inconsistent in (U, C). Since
$(a_1, 3) \subseteq (a_2, 2) \subseteq (d, \varphi)$,
$(a_3, 1) \subseteq (a_2, 2) \subseteq (d, \varphi)$,
$(a_4, 2) \subseteq (a_2, 2) \subseteq (d, \varphi)$,
$(a_4, 3) \subseteq (a_1, 2) \subseteq (d, \psi)$.
$(a_1, 3)$, $(a_3, 1)$, $(a_4, 2)$ and $(a_4, 3)$ are relatively reducible with respect to P. Then, we delete them from C. Because
$(a_1, 2) \cup (a_4, 1) = (a_2, 1)$,
$(a_2, 1)$ is absolutely reducible. Thus, we delete it from C. Then, we get the final reduction.
$reduct(C|P) = \{(a_1, 1), (a_1, 2), (a_2, 2), (a_3, 2), (a_3, 3), (a_4, 1)\}$.
Similarly, we have
$Pos_{reduct(C|P)}(P) = \{x_1, x_3, x_4, x_5, x_6, x_7\}$,
$BN_{reduct(C|P)}(P) = \{x_2\}$.
That is, the reduced approximation space has the same classification ability as the unreduced approximation space. Comparing with the unreduced approximation space, the number of elements in the reduced approximation space is 45% off.

15.4 Fuzzy Decision Making with Covering Generalized Rough Fuzzy Sets

To cope with the problems, both roughness and fuzziness, rough set theory and fuzzy set theory were put together. Rough fuzzy set and fuzzy rough set were first developed by Dubois and Prade [2] [3]. Since then, the combination of these two theories won wide attention [23] [32] [35] [21], and had been successfully applied in many complicated problems, such as numeric attribute information system processing, hybrid attribute information system processing, and so on [17] [13] [10] [9] [14].

Generally, classical rough set theory can only address nominal attribute information system. For the information system with numeric attribute, attribute discretization is a necessary step in data preprocessing. However, the nominal values gotten by discretization are often meaningless and disjoint. And yet, many concepts used in real world have no clear border, such as “young”, “middle” and “old”. So, the rules mined from data are not understandable or usable for people. In this section, we will combine rough set theory and fuzzy set theory, and develop a covering rough fuzzy set model, and then use it to solve the problem of fuzzy decision making.

15.4.1 Covering Generalized Rough Fuzzy Sets

Let (U,C) be a covering approximation space, and A be a fuzzy set on U, then A can be approximated by a pair of fuzzy sets $(\underline{C}(A),\overline{C}(A))$, where $\underline{C}(A)$ is called the covering lower approximation of A, and $\overline{C}(A)$ the covering upper approximation of A. Their membership functions are defined as follows:

$$\mu_{\underline{C}(A)}(x) = sup_{K\in Ad(x)}\{inf_{y\in K}\{\mu_A(y)\}\} \tag{15.10}$$
$$\mu_{\overline{C}(A)}(x) = inf_{K\in Ad(x)}\{sup_{y\in K}\{\mu_A(y)\}\} \tag{15.11}$$

Example 15.5. Let $U = \{x_1,x_2,x_3,x_4\}$, and $C = \{K_1,K_2,K_3\}$, where $K_1 = \{x_1,x_2\}$, $K_2 = \{x_2,x_3\}$,$K_3 = \{x_3,x_4\}$.

$A = \{1/x_1,1/x_2,0/x_3,0/x_4\}$ and $B = \{1/x_1,0/x_2,1/x_3,0/x_4\}$ are two fuzzy sets on U. According to the definition above, we have

$\underline{C}(A) = \{1/x_1,1/x_2,0/x_3,0/x_4\}$,$\overline{C}(A) = \{1/x_1,1/x_2,0/x_3,0/x_4\}$
$\overline{\underline{C}}(B) = \{1/x_1,0/x_2,1/x_3,0/x_4\}$,$\underline{C}(B) = \{1/x_1,1/x_2,1/x_3,1/x_4\}$

Let x be an arbitrary object of U. Suppose $K \in Ad(x)$, but $K \notin Md(x)$, then $\exists_{L\in Md(x)}(L \subseteq K)$. And then,

$\mu_{\underline{C}(A)}(x) \geq inf_{y\in L}\{\mu_A(y)\} \geq inf_{y\in K}\{\mu_A(y)\}$.
$\mu_{\overline{C}(A)}(x) \leq sup_{y\in L}\{\mu_A(y)\} \leq sup_{y\in K}\{\mu_A(y)\}$.

That means, the lower and upper approximations do not change after K is deleted. In other words, the minimal description set has the same approximation ability as the description set. So, the definition of covering rough fuzzy set can be equivalently written as:

$$\mu_{\underline{C}(A)}(x) = sup_{K\in Md(x)}\{inf_{y\in K}\{\mu_A(y)\}\} \tag{15.12}$$
$$\mu_{\overline{C}(A)}(x) = inf_{K\in Md(x)}\{sup_{y\in K}\{\mu_A(y)\}\} \tag{15.13}$$

Moreover, we can get the following propositions.

Proposition 15.11. *Let C be a covering of U, $red(C)$ be the absolute reduction of C, then C and $red(C)$ generate the same covering rough fuzzy sets.*

Proposition 15.12. *Let C be a covering of U, and $\emptyset$ be the empty set. A and B are two fuzzy sets, and $\sim A$ is the complement of A. We have*

(1) Co-normality: $\underline{C}(U) = \overline{C}(U) = U$
(2) Normality: $\underline{C}(\emptyset) = \overline{C}(\emptyset) = \emptyset$
(3) Contraction & Extension: $\underline{C}(A) \subseteq A \subseteq \overline{C}(A)$
(4) Duality: $\underline{C}(\sim A) = \sim \overline{C}(A)$, $\sim \overline{C}(\sim A) = \sim \underline{C}(A)$
(5) Monotone: $A \subseteq B \Rightarrow \underline{C}(A) \subseteq \underline{C}(B)$, $A \subseteq B \Rightarrow \overline{C}(A) \subseteq \overline{C}(B)$
(6) Idempotency: $\underline{C}(A) = \underline{C}(\underline{C}(A))$,$\overline{C}(A) = \overline{C}(\overline{C}(A))$

15.4.2 An Example

The approval of credit card is a complex problem. In the process of approval, the salary of applicant is a very important information for decision making. However, applicants are not often willing to tell their own salaries. To address this problem, we can forecast the income level of an applicant by other information. Next, we will use the model developed above to help us to address this problem.

To simplify the problem, we only use the educational level to forecast the income level of applicants. Suppose there are 9 applicants $U = \{x_1, x_2, ..., x_9\}$, and their educational levels labeled by 3 experts are as follows:

$E_1 : good = \{x_1, x_2, x_3\}, average = \{x_4, x_5, x_6\}, poor = \{x_7, x_8, x_9\}$;

$E_2 : good = \{x_1, x_2\}, average = \{x_3, x_4, x_5\}, poor = \{x_6, x_7, x_8, x_9\}$;

$E_3 : good = \{x_1, x_2\}, average = \{x_3, x_4, x_5, x_6, x_7\}, poor = \{x_8, x_9\}$.

According the evaluation of 3 experts, we can get a covering $C = \{good, average, poor\}$, where $good = \{x_1, x_2, x_3\}$, $average = \{x_3, x_4, x_5, x_6, x_7\}$, and $poor = \{x_6, x_7, x_8, x_9\}$.

The salaries of all applicants are as Table 15.2.

Table 15.2 Salaries of applicants

U	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
Salary	5000	3800	3500	3300	2500	2000	1600	1300	900

Then, we can get three fuzzy sets, *high*, *middle*, and *low*, according to the fuzzy membership function shown as Fig. 15.2. They are

$high = \{1/x_1, 0.8/x_2, 0.5/x_3, 0.3/x_4, 0/x_5, 0/x_6, 0/x_7, 0/x_8, 0/x_9\}$,

$middle = \{0/x_1, 0.2/x_2, 0.5/x_3, 0.7/x_4, 1/x_5, 1/x_6, 0.6/x_7, 0.3/x_8, 0/x_9\}$,

$low = \{0/x_1, 0/x_2, 0/x_3, 0/x_4, 0/x_5, 0/x_6, 0.4/x_7, 0.7/x_8, 1/x_9\}$.

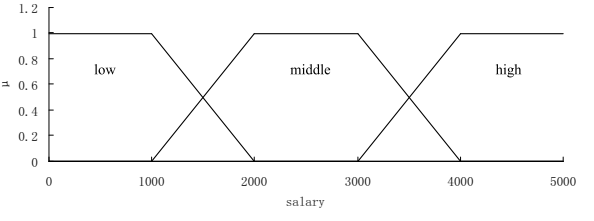


Fig. 15.2 Fuzzy membership function of salary

Taking fuzzy set *high* as example, we can get its lower and upper approximations:

$\underline{\underline{C}}(high) = \{0.5/x_1, 0.5/x_2, 0.5/x_3, 0/x_4, 0/x_5, 0/x_6, 0/x_7, 0/x_8, 0/x_9\}$

$\overline{C}(high) = \{1/x_1, 1/x_2, 0.5/x_3, 0.5/x_4, 0.5/x_5, 0/x_6, 0/x_7, 0/x_8, 0/x_9\}$

Let $\mu_{\overline{C}(high)}(good)$ and $\mu_{\underline{C}(high)}(good)$ be the maximum and minimum membership degrees of *good* in *high* respectively, then

$\mu_{\overline{C}(high)}(good) = sup\{u_{high}(x)|x \in good\} = 1,$

$\mu_{\underline{C}(high)}(good) = inf\{u_{high}(x)|x \in good\} = 0.5.$

That is, when we know the educational level of an applicant is *good*, we can forecast that the membership degree of he/she belonging to *high* is between 0.5 and 1. Similarly, we have

$\mu_{\overline{C}(middle)}(good) = 0.5$, $\mu_{\underline{C}(middle)}(good) = 0,$

$\mu_{\overline{C}(low)}(good) = 0$, $\mu_{\underline{C}(low)}(good) = 0,$

$\mu_{\overline{C}(high)}(average) = 0.5$, $\mu_{\underline{C}(high)}(average) = 0,$

$\mu_{\overline{C}(middle)}(average) = 1$, $\mu_{\underline{C}(middle)}(average) = 0.5,$

$\mu_{\overline{C}(low)}(average) = 0.4$, $\mu_{\underline{C}(low)}(average) = 0,$

$\mu_{\underline{C}(high)}(poor) = 0$, $\mu_{\overline{C}(high)}(poor) = 0,$

$\mu_{\overline{C}(middle)}(poor) = 1$, $\mu_{\underline{C}(middle)}(poor) = 0,$

$\mu_{\overline{C}(low)}(poor) = 1$, $\mu_{\underline{C}(middle)}(poor) = 0.$

Let $\mu_{\overline{C}(high)}(x_3)$ and $\mu_{\underline{C}(high)}(x_3)$ be the maximum and minimum membership degrees of x_3 in *high* respectively. Then

$\mu_{\overline{C}(high)}(x_3) = inf\{\mu_{\overline{C}(high)}(good), \mu_{\overline{C}(high)}(average)\} = 0.5,$

$\mu_{\underline{C}(high)}(x_3) = sup\{\mu_{\underline{C}(high)}(good), \mu_{\underline{C}(high)}(average)\} = 0.5,$

Therefore, for a new applicant, if we can get his/her educational level, then we can use the method proposed above to forecast his/her income level.

15.5 Conclusion

In this chapter, we discuss how to use covering generalized rough set theory to address some uncertainty problems. One is incomplete information processing, and the other is fuzzy decision making. The examples in this chapter give enlightenment for covering based uncertainty problem processing.

Acknowledgements. This paper is supported by the Science & Technology Research Program of Chongqing Education Committee of P. R. China (No.KJ090512), the Natural Science Foundation of Chongqing of P. R. China (No.2008BA2017), the National Natural Science Foundation of P. R. China (No.60773113, No.61073146), the Foundation of the College of Computer Sci. & Tech., Chongqing University of Posts and Telecommunications (JK-Y-2010004).

References

1. Bonikowski, Z., Bryniarski, E., Wybraniec, U.: Extensions and intentions in the rough set theory. Information Science 107, 149–167 (1998)
2. Dubois, D., Prade, H.: Rough fuzzy sets and fuzzy rough sets. International Journal of General Systems 17, 191–208 (1990)
3. Dubois, D., Prade, H.: Putting rough sets and fuzzy sets together. In: Słowiński, R. (ed.) Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory, pp. 203–222. Kluwer Academic Publishers, Boston (1992)
4. Feng, T., Mi, J.S., Wu, W.Z.: Covering-based generalized rough fuzzy sets. In: The First International Conference Rough Sets and Knowledge Technology, Chongqing, China, pp. 208–215 (2006)
5. Grzymala-Busse, J.W.: Characteristic Relations for Incomplete Data: A Generalization of the Indiscernibility Relation. In: Peters, J.F., Skowron, A. (eds.) Transactions on Rough Sets IV. LNCS, vol. 3700, pp. 58–68. Springer, Heidelberg (2005)
6. Hu, J., Wang, G.Y., Zhang, Q.H.: Uncertainty measure of covering generated rough set. In: 2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT 2006 Workshops) (WI-IATW 2006), pp. 498–504. IEEE Computer Society Press (2006)
7. Hu, J., Wang, G.Y.: Knowledge Reduction of Covering Approximation Space. In: Gavrilova, M.L., Tan, C.J.K., Wang, Y., Chan, K.C.C. (eds.) Transactions on Computational Science V. LNCS, vol. 5540, pp. 69–80. Springer, Heidelberg (2009)
8. Hu, J.: Covering based Granular Computing and Its Application. PhD thesis, Xidian University, Xi'an, Shanxi, China (2010)
9. Hu, Q.H., Xie, Z.X., Yu, D.R.: Hybrid attribute reduction based on a novel fuzzy-rough model and information granulation. Pattern Recognition 40, 3509–3521 (2007)
10. Hu, Q.H., Yu, D.R., Xie, Z.X.: Information-preserving hybrid data reduction based on fuzzy-rough techniques. Pattern Recognition Letters 27, 414–423 (2006)
11. Huang, B., He, X., Zhou, X.Z.: Rough entropy based on generalized rough sets covering reduction. Journal of Software 15(2), 215–220 (2004)
12. Huang, H., Wang, G.Y., Wu, Y.: An approach for Incomplete Information Systems. In: Dasarathy, B.V. (ed.) Data Mining and Knowledge Discovery: Theory, Tools, and Technology VI. Proceedings of SPIE, vol. 5433, pp. 114–121 (2004)
13. Jensen, R., Shen, Q.: Fuzzy-rough attribute reduction with application to web categorization. Fuzzy Sets and Systems 141(3), 469–485 (2004)
14. Jensen, R., Shen, Q.: Fuzzy-rough sets assisted attribute selection. IEEE Transactions on Fuzzy Systems 15(1), 73–89 (2007)
15. Kryszkiewicz, M.: Rough Set Approach to incomplete information system. Information Sciences 112, 39–49 (1998)
16. Kryszkiewicz, M.: Rules in incomplete information systems. Information Sciences 113, 271–292 (1999)
17. Kuncheva, L.I.: Fuzzy rough sets: Application to feature selection. Fuzzy Sets and Systems 51, 147–153 (1992)
18. Li, D.Y., Liu, C.Y., Du, Y., Han, X.: Artificial Intelligence with Uncertainty. Journal of Software 15(11), 1583–1593 (2004)
19. Li, T.J.: Rough approximation operators in covering approximation spaces. In: International Conference on Rough Sets and Current Trends in Computing, pp. 174–182 (2006)
20. Li, T.R., Ruan, D., Geert, W., Song, J., Xu, Y.: A Rough set based characteristic relation approach for dynamic attribute generalization in data mining. Knowledge-Based Systems 20(5), 485–494 (2007)
21. Mi, J.S., Leung, Y., Zhao, H.Y., Feng, T.: Generalized fuzzy rough sets determined by a triangular norm. Information Sciences 178(16), 3203–3213 (2008)
22. Mordeson, J.N.: Rough set theory applied to (fuzzy) ideal theory. Fuzzy Sets and Systems 121, 315–324 (2001)

23. Morsi, N.N., Yakout, M.M.: Axiomatics for fuzzy rough sets. Fuzzy Sets and Systems 100(1-3), 327–342 (1998)
24. Pawlak, Z.: Rough sets. Computer and Information Science 11, 341–356 (1982)
25. Pawlak, Z.: Rough Sets: Theoretical Aspects Of Reasoning About Data. Kluwer Academic Publishers, Boston (1991)
26. Słowiński, R., Vanderpooten, D.: A Generalized Definition of Rough Approximations Based on Similarity. IEEE Tansactions on Knowledge and Data Engineering 12(2), 331–326 (2000)
27. Stefanowski, J., Tsoukias, A.: On the extension of rough sets under incomplete information. In: Zhong, N., Skowron, A., Ohsuga, S. (eds.) The 7th International Workshop on New Directions in Rough Sets, Data Mining, and Granular Soft Computing, pp. 73–81 (1999)
28. Stefanowski, J., Tsoukias, A.: Incomplete information tables and rough classification. Computational Intelligence 17(3), 545–566 (2001)
29. Tsang, E.C.C., Chen, D., Lee, J.W.T., Yeung, D.S.: On the upper approximations of covering generalized rough sets. In: Proceedings of The Third International Conference on Machine Learning and Cybernetics, Shanghai, pp. 4200–4203 (2004)
30. Wang, G.Y.: Extension of rough set under incomplete information systems. In: IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1098–1103. IEEE Computer Society Press (2002)
31. Wei, L., Miao, D.Q., Xu, F.F., Xia, F.C.: Research on a covering rough fuzzy set model. Journal of Computer Research and Development 43(10), 1719–1723 (2006)
32. Wu, W.Z., Mi, J.S., Zhang, W.X.: Generalized fuzzy rough sets. Information Sciences 151, 263–282 (2003)
33. Xu, Z.Y., Liao, J.Q.: On the covering fuzzy rough sets model. Fuzzy Systems and Mathematics 20(3), 141–144 (2006)
34. Xu, W.H., Zhang, X.Y.: Fuzziness in covering generalized rough sets. In: The 26th Chinese Control Conference, Zhangjiajie, Hunan, China, pp. 386–390 (2007)
35. Yeung, D.S., Chen, D.G., Tsang, E.C.C., John, W.T.L.: On the generalization of fuzzy rough sets. IEEE Transactions on Fuzzy Systems 13(3), 343–361 (2005)
36. Żakowski, W.: Approximation in the Space (U, Π). Demonstratio Mathematica 16, 761–769 (1983)
37. Zhu, F.: On Covering Generalized Rough Sets, MS thesis, The University of Arizona, Tucson, Arizona, USA (2002)
38. Zhu, W., Wang, F.Y.: Reduction and axiomization of coving generalized rough sets. Information Science 152, 217–230 (2003)
39. Zhu, W., Wang, F.Y.: Covering based granular computing for conflict analysis. In: Proc. IEEE Intelligence and Security Informatics Conf. (ISI 2006), pp. 566–571. IEEE Computer Society Press (2006)
40. Zhu, W., Wang, F.Y.: A new type of covering rough sets. In: IEEE IS 2006, September 4-6, pp. 444–449. IEEE Computer Society Press (2006)
41. Zhu, W.: A class of covering-based fuzzy rough sets. In: The Fourth International Conference on Fuzzy Systems and Knowledge Discovery, Haikou, China, pp. 7–11 (2007)
42. Zhu, W.: Topological approaches to covering rough sets. Information Sciences 177, 1499–1508 (2007)
43. Zhu, W., Wang, F.Y.: On three types of covering-based rough sets. IEEE Transactions on Knowledge and Data Engineering 19(8), 1131–1144 (2007)

Chapter 16
Hardware Implementations of Rough Set Methods in Programmable Logic Devices

Maciej Kopczyński and Jarosław Stepaniuk

Abstract. This paper describes current achievements in hardware realization of rough sets algorithms in FPGA (**F**ield **P**rogrammable **G**ate **A**rray) logic devices. At the moment, only few ideas and hardware implementations have been created. This chapter is a survey of them.

Keywords: Rough sets, hardware implementation, FPGA, programmable logic devices

16.1 Introduction

Professor Zdzisław Pawlak introduced rough sets assuming that objects are perceived by values of some attributes (for review see *e.g.*, [9, 10, 12]). Existing implementations of the rough set methods are implemented using high-level programming languages. This type of implementation provides the ability to comply with any of the algorithms, but the biggest issue is the relative low speed of operation.

The computer processor is a versatile system that executes an arbitrary list of compiler-generated instructions dependent on the source code created by the programmer. For this reason, processors are not optimized to perform specific actions, such as simultaneous rapid execution of elementary logical operations on large amounts of data in a set of objects.

Creating hardware implementation allows us a huge acceleration of the calculation related to the chosen topic, but the disadvantage of this approach is the limit of the applicability of such system only to given issues. A good example of such solutions are GPU (**G**raphics **P**rocessing **U**nit), which are optimized for parallel

Department of Computer Science, Białystok University of Technology
Wiejska 45A, 15-351 Białystok, Poland
e-mail: {m.kopczynski,j.stepaniuk}@pb.edu.pl

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 309–321.
springerlink.com

execution of calculations related to computer graphics. Most of the mass-produced systems are ASIC-type systems (**A**pplication **S**pecific **I**ntegrated **C**ircuit) that do not allow changes defined in their logical function. Prototype implementations of specialized processors may be implemented in programmable logic devices (*e.g.*, CPLDs (**C**omplex **P**rogrammable **L**ogic **D**evice) and FPGAs) as sequential and combination systems.

Another possibility is a combination of both implementation techniques of the algorithms, which can take advantages of versatility known from software implementations and high-speed calculation of hardware implementations.

Rough sets theory concepts implementation in hardware device can significantly accelerate the execution time of algorithms compared to software implementation. Logic devices can execute the whole algorithm or just the most time-consuming parts of it.

The chapter is organized as follows. In Section 16.2.1, Pawlak's idea of rough set processor is discussed. In Section 16.2.2, application of cellular networks in rough set methods is shortly recalled. In Section 16.2.3, some investigations of Kanasugi are presented. In Conclusions, we summarize the results of the chapter and present some directions for further research.

16.2 Solutions Architecture

Designed and implemented rough sets hardware devices use the PC as an external data source and an element of executing all or part of the main control program. Hardware systems are used as mechanisms for performing complex calculations, so it is possible to significantly accelerate the calculation time of algorithms. This type of devices can be regarded as a kind of coprocessors. Block diagram of such solutions is shown in Fig. 16.1.

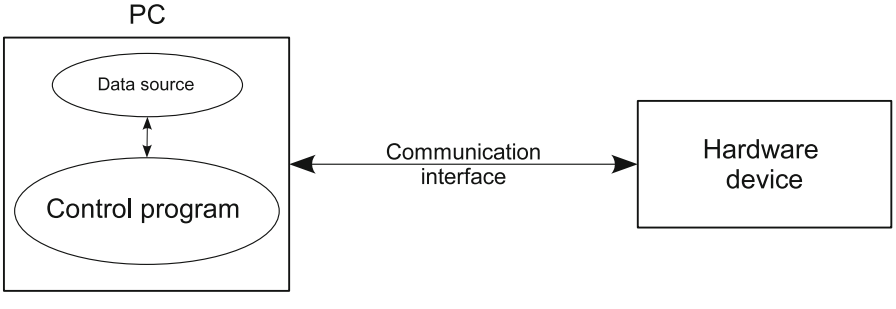


Fig. 16.1 Block diagram of the rough sets hardware implementation

More powerful FPGA development boards, in addition to the logic device, also contain RAM, flash memory, external flash memory connectors, communication interfaces (USB, Ethernet, etc...) and extensive microcontrollers, *e.g.*, on the ARM cores. Moreover, large FPGAs allow us the implementation of the soft-core processors [14] [15]. Development boards equipped within this type of devices can be used for the implementation of control software without the need for an external, large PC. Currently available versions of operating systems (*e.g.* Linux) support most common of soft-core processors, which makes it possible to install it on this type of device. The Computing power of such processors is satisfactory for supporting these tasks. This leads to minimization of the amount of space occupied by the resulting device, which finally becomes an independent unit.

Currently, most of the existing softwares implement rough set algorithms in a sequential manner. However, an important feature of these algorithms is the ability to perform their parts in parallel (see *e.g.* [13]), with an emphasis on hardware implementations. In hardware devices, it is easy to duplicate some blocks for parallel computational calculations. The only problem to be solved concerns the development of appropriate process control systems.

16.2.1 Pawlak's Idea of Rough Set Processor

In [8], Pawlak presented an outline of an exemplary RSP (**R**ough **S**et **P**rocessor) structure. The organization of a simple processor is based on elementary rough set granules and dependencies between them. A simplified *RSP* is shown in Fig. 16.2.

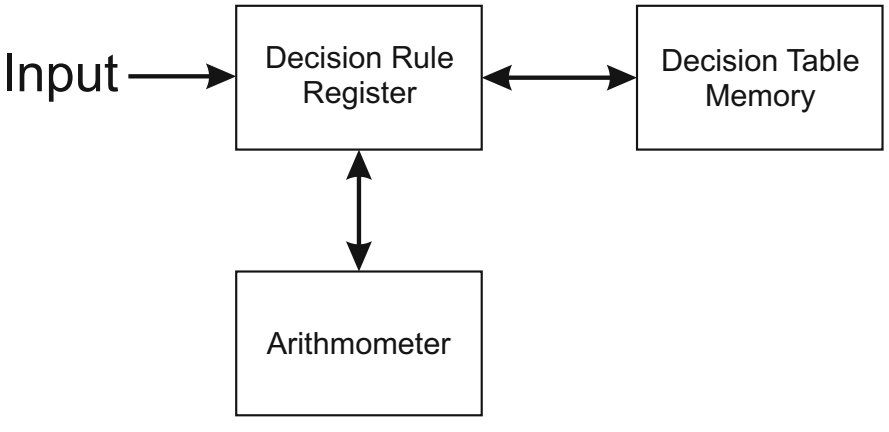


Fig. 16.2 Block diagram of *Rough Set Processor* [8]

RSP consists of the following units:

- *Decision table memory* – this unit keeps the data from the decision table. An ideal situation would be if the memory is large and fast enough to store the whole decision table.
- *Decision rule register* – the main purpose of this unit is to generate a final set of decision rules. This module cooperates with the arithmometer because of the need to perform some calculations.
- *Arithmometer* – is unit the used to perform arithmetic operations for the rest of the modules.

The idea of *RSP* design is as follows. At the beginning, only conditions, decisions and support of each decision rule are given. Condition and decision are parts of the decision rule. Support is the number of objects from the original decision table matching a given decision rule.

The next operation step is calculation of strength, certainty and coverage factors of each decision rule. These values will be used to find the most important decision rules.

The idea presented by Pawlak was not realized in the programmable logic device.

16.2.2 Cellular Networks

Rybiński and Muraszkiewicz in [6] created the concept of describing rough sets methods with usage of cellular networks. On this basis, the idea of the implementation of a device called *PRSComp* (**P**arallel **R**ough **S**ets **Comp**uter) was presented. This is a device for parallel processing of basic rough set operations. The description of cellular networks is contained in [7]. A single cell of the network is able to perform the following operations: write a single character, read the saved character, compare two characters, move the character to the southern neighbor, swap characters between two adjacent cells in the column, set the state and move the bit from the east input to the west output. Cellular network consists of a matrix of interconnected elements of the same type (cells that can be treated as simple, single processors) and a set of control registers. Block diagram of a cellular network with a set of registers is shown in Fig. 16.3.

Registers defined in *PRSComp* are as follows:

- *Column Mask Register (CM)* – used to inhibit the processing of matrix cells.
- *Comparand Register (C)* – used to transfer words from and to the processor array. Register also takes part in comparison operations.
- *Word Selection Register (E)* – contains the results of performed comparisons.

The use of cellular network with rough sets is based on the transformation of the input data set to the matrix and definition of the basic operations associated with rough sets using matrix notation. In [6], the following notions are presented along with their pseudocode:

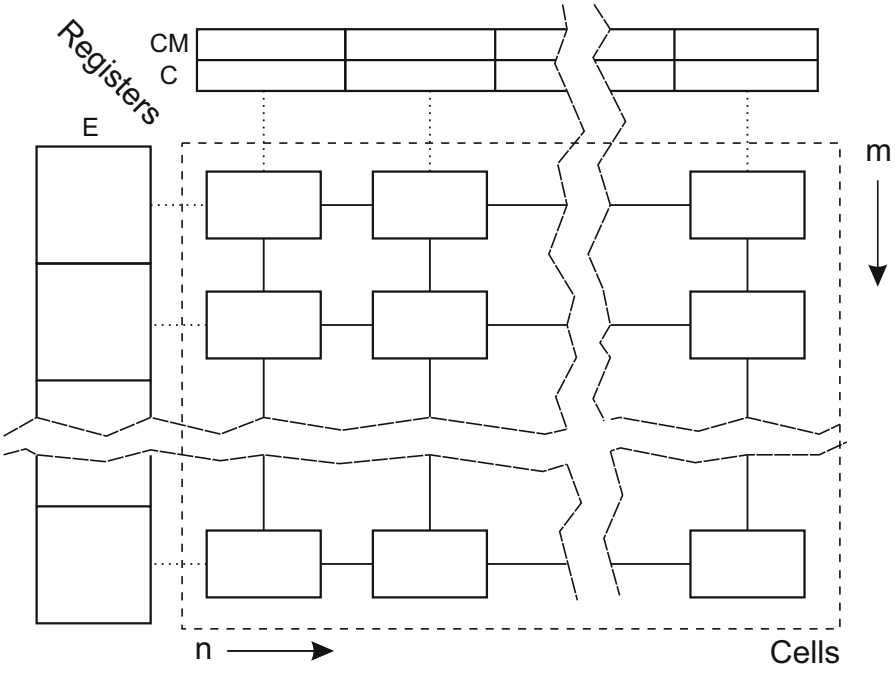


Fig. 16.3 Block diagram of sample cellular network

- indiscernibility relation,
- upper approximation,
- lower approximation,
- reducts calculation,
- core calculation.

The given pseudocode allows the implementation of the presented matrix notation in programmable logic devices. Pseudocode in conjunction with the homogenous nature of cellular networks makes it possible to encode algorithms in such a way that an operation (instruction) causes the simultaneous processing of many data elements. This mode of operation is called a *SIMD* (**S**ingle **I**nstruction **M**ultiple **D**ata).

Paper [6] provides a basis for the development and expanding *PRSComp* device for another, more complex operation associated with rough sets and matrix notation.

16.2.2.1 Self-Learning Cellular Network

Lewis, Perkowski and Jozwiak [5] described the idea of self-learning rough sets model representation in hardware device. The Model is based on cellular networks by Rybiński and Muraszkiewicz. They suggested the possibility of implementing

the solution on *DEC-PERLE-1* board, which is the matrix consisting of 23 type 3090 FPGAs from *Xilinx*. The general principle of device operation is to perform the learning process at the higher level (software), while the later results of this process are transferred to a lower level (hardware). A detailed description of the system working process is as follows:

1. Creation of cellular network logical structure based on an optimized decision matrix (the set of examples) and the requirements for the network construction.
2. Cellular network structure developed on the basis of data from the decision matrix mapped to the FPGA unit, where each device performs the functions described in the *PRSComp* system. Mapping is created using standard EDA software (*e.g.*, from *Xilinx*).
3. Device's knowledge is stored in the memory of the *DEC-PERLE-1* board as the patterns representing created cellular networks structures. Previously created cellular network patterns are multiplexed in order to choose the best one when working with different data sets. The Switching scheme is supervised by an external computer with appropriate control software.
4. During the network training phase, when solving new problems, decisions taken are stored in the memory. Based on this data, the network structure can be re-organized or built completely from scratch in order to avoid the impact of the previously created pattern.

Dharmadhikari, Ngo and Lewis in [1] developed a partial implementation of the *PRSComp* unit on the *DEC-PERLE-1* board. They have created a VHDL source code describing the operation of two elementary rough sets operations: the lower and upper approximation. The module performing the comparison of values (in [6] operation described as *EXTCOMP*) was also implemented. The authors did not test the cooperation of the modules; only single units were tested.

16.2.2.2 Didactic Example

Example of finding dispensable attributes will be presented. First, three of the algortihms known from *PRSComp* must be shown: *EXTCOMP*, *BasicCAT* and *Indispensable* [6]. Let m be the number of rows and n be the number of columns in cellular array.

```
Routine EXTCOMP (A[m; n]; C[n]; E[m])
/* C[n] is the comparand */
   MASK
   E[m] := 0
   for j = 0 to n if c[j] = a[i; j], then a couple is
      transparent by labeled by 't', otherwise it is opaquely
      labeled by 'o' logic value of 1 is propagated
      through rows of the array, if all cells in the
      row are transparent then 1 is injected into
      that position in the E register

Routine BasicCAT (A[m,n], i, E[m]) /* Basic category */
```

```
/* The E register contains the characteristic vector that
indicates the words belonging to the basic category
generated by a_i */
   MASK
   C[n] := a_i
   EXTCOMP(A[m; n]; i; E[m])

Routine Indispensable (A[m; n]; j) /* Indispensability */
/* j indicates the column number to be checked out */
   for i = 1 to m
   begin
      MASK
      BasicCAT (A[m; n]; i; B[m])
      (MASK - {j})
      BasicCAT (A[m; n]; i; C[m])
      if B[m] ≠ C[m] then {j} is indispensable
   end
```

Finding dispensable attribute can be realized by removing one input variable at a time and determining whether or not the resulting table is consistent. The provided example presents how this operation can be performed in *PRSComp*. The decision table used in the example is presented in Table 16.1.

Table 16.1 Decision table [5]

U	a	b	c	d	e	f
1	0	0	0	1	1	1
2	0	1	0	0	0	1
3	0	1	1	0	0	1
4	1	1	0	0	0	1
5	1	1	0	1	0	1
6	1	1	1	1	0	1
7	1	1	1	0	0	1
8	1	0	0	1	1	1
9	1	0	0	1	0	1
10	0	0	0	0	0	0

The initial state of the machine is shown in Table 16.2.

Every cell of this table represents one primitive processor in *PRSComp*. Initially, the contents of the *CM* (column mask register), *C* (comparand) and *E* (word selection) registers are undefined. The *U* column is shown in the tables for the clarity of example.

First, the *Indispensable* routine is called. Loop within the routine is executed for every row of Table 16.1. After the first step, the *CM* register is initialized to all

Table 16.2 Initial state of the *PRSComp* machine [5]

	CM						
	C						
E	*U*	*a*	*b*	*c*	*d*	*e*	*f*
	1	0	0	0	1	1	1
	2	0	1	0	0	0	1
	3	0	1	1	0	0	1
	4	1	1	0	0	0	1
	5	1	1	0	1	0	1
	6	1	1	1	1	0	1
	7	1	1	1	0	0	1
	8	1	0	0	1	1	1
	9	1	0	0	1	0	1
	10	0	0	0	0	0	0

1s (line $\overline{MASK}$), which means that no processors are masked. In the next line, the *BasicCAT* routine is called. The *BasicCAT* routine sets again the *CM* register with 1s. Then, the current row is copied to the comparand register *C*. In the next line, the *EXTCOMP* routine is called. It sets up the *CM* register in the first line. The *E* register is then initialized to all 0s. The result of the operations performed in the *PRSComp* machine is shown in Table 16.3, presenting intermediate state of the internal data.

Table 16.3 Intermediate state of the *PRSComp* machine [5]

	CM	1	1	1	1	1	1
	C	0	0	0	1	1	1
E	*U*	*a*	*b*	*c*	*d*	*e*	*f*
0	1	0	0	0	1	1	1
0	2	0	1	0	0	0	1
0	3	0	1	1	0	0	1
0	4	1	1	0	0	0	1
0	5	1	1	0	1	0	1
0	6	1	1	1	1	0	1
0	7	1	1	1	0	0	1
0	8	1	0	0	1	1	1
0	9	1	0	0	1	0	1
0	10	0	0	0	0	0	0

Then the *for* loop iterates through all cells of the current row and labels every cell with *"t"* or *"o"* depending on comparision result. *PRSComp* machine state after the first execution of *EXTCOMP* is shown in Table 16.4.

Table 16.4 State of the *PRSComp* machine after the first call of *EXTCOMP* [5]

	CM	1	1	1	1	1	1
	C	0	0	0	1	1	1
E	*U*	*a*	*b*	*c*	*d*	*e*	*f*
1	1	0 t	0 t	0 t	1 t	1 t	1 t
0	2	0 t	1 o	0 t	0 o	0 o	1 t
0	3	0 t	1 o	1 o	0 o	0 o	1 t
0	4	1 o	1 o	0 t	0 o	0 o	1 t
0	5	1 o	1 o	0 t	1 t	0 o	1 t
0	6	1 o	1 o	1 o	1 t	0 o	1 t
0	7	1 o	1 o	1 o	0 o	0 o	1 t
0	8	1 o	0 t	0 t	1 t	1 t	1 t
0	9	1 o	0 t	0 t	1 t	0 o	1 t
0	10	0 t	0 t	0 t	0 o	0 o	0 o

The current value of the *E* register is saved. Next, the bit in the *CM* register corresponding to the index of column *a* in decision table is set to 0 (the column is checked). The *EXTCOMP* routine is called again and Table 16.5 shows the *PRSComp* machine state after complete execution of the *EXTCOMP* routine.

Table 16.5 State of the *PRSComp* machine after second call of *EXTCOMP* [5]

	CM	0	1	1	1	1	1
	C	0	0	0	1	1	1
E	*U*	*a*	*b*	*c*	*d*	*e*	*f*
1	1	0 t	0 t	0 t	1 t	1 t	1 t
0	2	0 t	1 o	0 t	0 o	0 o	1 t
0	3	0 t	1 o	1 o	0 o	0 o	1 t
0	4	1 t	1 o	0 t	0 o	0 o	1 t
0	5	1 t	1 o	0 t	1 t	0 o	1 t
0	6	1 t	1 o	1 o	1 t	0 o	1 t
0	7	1 t	1 o	1 o	0 o	0 o	1 t
1	8	1 t	0 t	0 t	1 t	1 t	1 t
0	9	1 t	0 t	0 t	1 t	0 o	1 t
0	10	0 t	0 t	0 t	0 o	0 o	0 o

Previous and current values of the E register are compared. The values are not equal, which means that attribute a is indispensable — it must be kept in the final solution. In hardware implementation, all comparisions are done in parallel for every row. This speeds up all the necessary calculations.

16.2.3 Direct Solutions

Kanasugi and Yokoyama [2] developed a concept of logic device capable of minimizing the large logic functions created on the basis of discernibility matrix. System output are small logical functions representing important decision rules. Figure 16.4 shows the block diagram of a database knowledge discovery process with proposed location of the system.

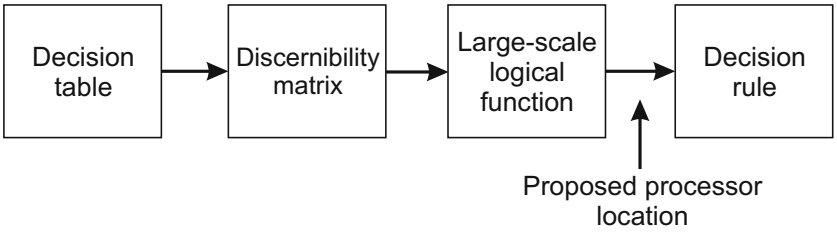


Fig. 16.4 Block diagram of the database knowledge discovery process with system location [4]

The presented system is not independent. It requires an external data source and the mechanisms for creating large logical functions from the database for correct operation. This system can be treated as a coprocessor supporting the central unit.

A block diagram of the logic device is shown in Fig. 16.5.

The project consists of the following functional elements:

- *Discernibility matrix maker* – the unit makes the conversion from the database that represents decision matrix to a binary decision matrix containing boolean values. This module is theoretical due to the implementation complexity in the hardware.
- *Core selector* – the main task of this unit is to select rows of binary decision matrix, which include the shortest logical formulas (which have the smallest number of variables with a true value).
- *Covering unit* – the goal of this unit is to check each row of a binary decision matrix and denote them as candidates to be removed. Selecting the rows is based on the data prepared by the *core selector*.
- *Reconstruction unit* – the role of this unit is to discover dominant variables in binary decision matrix, which helps to find most significant decision rules.

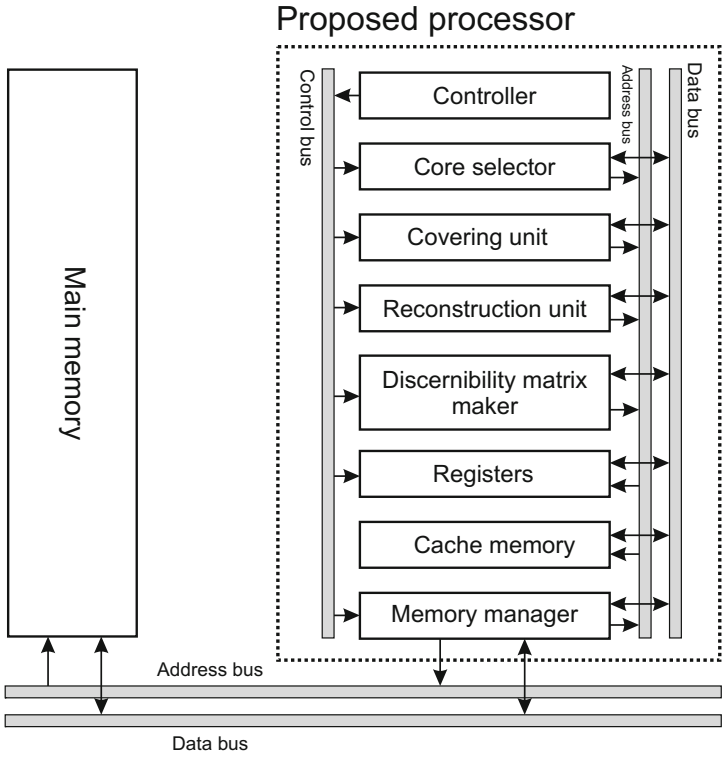


Fig. 16.5 Block diagram of the logic device [4]

System functionality can be divided into two parts: pre-process (data preparation) and main process (working with prepared data).
In the data pre-process, two units are used: *Core Selector* and *Covering Unit*. The purpose of the two mentioned units is to find the rows which contain the least amount of boolean variables (search for cores). The modules also select redundant binary decision matrix rows which can be deleted. Let us assume that the logical function is given in the following form:

$$F = (x_0 \vee x_2 \vee x_6) \wedge (x_0 \vee x_1 \vee x_2 \vee x_5 \vee x_6) \wedge (x_1 \vee x_2 \vee x_6 \vee x_7)$$

The shortest term is $(x_0 \vee x_2 \vee x_6)$. The second term can be removed because it satisfies the relationship $coreterm \subseteq otherterm$:

$$(x_0 \vee x_2 \vee x_6) \subseteq (x_0 \vee x_1 \vee x_2 \vee x_5 \vee x_6).$$

The purpose of the main process is to review the whole pre-processed binary decision matrix and select the most important rules (terms). The idea of the algorithm implemented in *Reconstruction Unit* is to find dominant variables in pre-processed

binary decision matrix and then create a new decision matrix. This matrix will contain some amount of important decision rules dependent on the algorithm parameter. It should be noted that the implemented algorithm uses an approximation technique. Kanasugi has decided on this solution because of the reduction in time of calculations and the size of the entire system in the FPGA structure.

The solution proposed by Kanasugi and Matsumoto in [4] allowed nearly 700 times increase in the speed of calculations in comparison to PC. Binary decision matrix containing 128 rows of data and 2032 attributes was used during the tests. Table 16.6 shows the results of the experiment.

Table 16.6 Comparison of calculation time between hardware and software solution [4]

Device type	Speed $[MHz]$	Time $[\mu s]$
Hardware solution	50	7.18
PC	3400	72.54

16.3 Conclusions and Future Research

The hardware implementation is the main direction of using scalable rough sets methods in real-time solutions. Software implementations are universal, but rather slow. Hardware realizations are deprived of this universality, however, the allow us to perform specific calculations in a substantially shorter time.

The authors are working on creating a rough sets hardware system, which has to be universal for any type of data. The goal of the system is to process the data in accordance with a set of rules, the rule generation from complex sets of data, fast reducts and approximation calculations and so on. An important part of the project is to create low-level input data transformation algorithms, which convert data from decision matrix or database to low-level form (boolean or binary). Inverse operation is also required — the conversion of results returned by the system to similar form found in software implementations. These operations can be performed by software executing on the main unit.

The system with hardware implementation of rough sets methods can be used in embedded systems, such as industrial controllers, or as an alternative and very fast method of process control and data classification. The field of potential usage of the system can be very wide due to its versatility.

Acknowledgements. We thank Professor Andrzej Skowron for the valuable comments and suggestions which helped us to improve the presentation. The research is supported by the Białystok University of Technology Rector's grant S/WI/5/08.

References

1. Dharmadhikari, M., Ngo, V., Lewis, T.: Design of a SIMD computer for rough set theory, web.cecs.pdx.edu/~mperkows/CLASS_VHDL/VHDL_CLASS_1999/project.ps (cited October 1, 2011)
2. Kanasugi, A., Yokoyama, A.: A basic design for rough set processor. In: The 15th Annual Conference of Japanese Society for Artificial Intelligence (2001)
3. Kanasugi, A.: A Design of Architecture for Rough Set Processor. In: Terano, T., Nishida, T., Namatame, A., Tsumoto, S., Ohsawa, Y., Washio, T. (eds.) JSAI-WS 2001. LNCS (LNAI), vol. 2253, pp. 406–412. Springer, Heidelberg (2001)
4. Kanasugi, A., Matsumoto, M.: Design and Implementation of Rough Rules Generation from Logical Rules on FPGA Board. In: Kryszkiewicz, M., Peters, J.F., Rybiński, H., Skowron, A. (eds.) RSEISP 2007. LNCS (LNAI), vol. 4585, pp. 594–602. Springer, Heidelberg (2007)
5. Lewis, T., Perkowski, M., Jozwiak, L.: Learning in hardware: Architecture and implementation of an FPGA-based rough set machine. In: 25th EUROMICRO 1999 Conference, vol. 1, pp. 1326–1334 (1999)
6. Muraszkiewicz, M., Rybiński, H.: Towards a parallel rough sets computer. In: Rough Sets, Fuzzy Sets and Knowledge Discovery, Workshops in Computing, pp. 434–443. Springer, Heidelberg (1994)
7. Muraszkiewicz, M.: Sieci komorkowe do przetwarzania danych nienumerycznych. Prace IINTE, no. 52 (1984) (in Polish)
8. Pawlak, Z.: Elementary rough set granules: Toward a rough set processor. In: Pal, S.K., Polkowski, L., Skowron, A. (eds.) Rough-Neurocomputing: Techniques for Computing with Words, Cognitive Technologies, pp. 5–14. Springer, Berlin (2004)
9. Pawlak, Z., Skowron, A.: Rudiments of rough sets. Information Sciences 177(1), 3–27 (2007)
10. Pedrycz, W., Skowron, A., Kreinovich, V. (eds.): Handbook of Granular Computing. John Wiley & Sons, New York (2008)
11. Skowron, A., Stepaniuk, J.: Tolerance approximation spaces. Fundamenta Informaticae 27(2-3), 245–253 (1996)
12. Stepaniuk, J.: Rough–Granular Computing in Knowledge Discovery and Data Mining. Springer, Heidelberg (2008)
13. Strąkowski, T., Rybiński, H.: A Distributed Decision Rules Calculation Using Apriori Algorithm. In: Peters, J.F., Skowron, A. (eds.) Transactions on Rough Sets XI. LNCS, vol. 5946, pp. 161–176. Springer, Heidelberg (2010)
14. Altera Corporation, www.altera.com (cited October 1, 2011)
15. Xilinx Corporation, www.xilinx.com (cited October 1, 2011)

Chapter 17
Determining Cosine Similarity Neighborhoods by Means of the Euclidean Distance

Marzena Kryszkiewicz

Abstract. Cosine similarity measure is often applied in the area of information retrieval, text classification, clustering, and ranking, where documents are usually represented as term frequency vectors or its variants such as tf-idf vectors. In these tasks, the most time-consuming operation is the calculation of most similar vectors or, alternatively, least dissimilar vectors. This operation has been commonly believed to be inefficient for large high-dimensional datasets. However, using the triangle inequality to determine neighborhoods based on a distance metric, offered recently, makes this operation feasible for such datasets. Although the cosine similarity measure is not a distance metric and, in particular, violates the triangle inequality, in this chapter, we present how to determine cosine similarity neighborhoods of vectors by means of the Euclidean distance applied to $(\alpha-)$normalized forms of these vectors and by using the triangle inequality. We address three types of sets of cosine similar vectors: all vectors, the similarity of which to a given vector is not less than an ε threshold value, and two variants of the k-nearest neighbors of a given vector.

Keywords: k-nearest neighbors, ε-neighborhood, the cosine similarity measure, the Euclidean distance, the triangle inequality, normalized vector, data clustering, text clustering, high-dimensional data.

17.1 Introduction

Cosine similarity measure is often applied in the area of information retrieval, text classification, clustering, and ranking, where documents are usually represented as

Marzena Kryszkiewicz
Institute of Computer Science, Warsaw University of Technology
Nowowiejska 15/19, 00-665 Warsaw, Poland
e-mail: mkr@ii.pw.edu.pl

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 323–345.
springerlink.com

term frequency vectors or its variants, such as tf-idf vectors. In these tasks, the most time-consuming operation is the calculation of most similar vectors or, alternatively, least dissimilar vectors. This operation has been commonly believed to be inefficient for large high-dimensional datasets. However, using the triangle inequality to determine neighborhoods based on a distance metric makes this operation feasible for such datasets [1,5–8,10,11].

Although the cosine similarity measure is not a distance metric and, in particular, violates the triangle inequality, recently we have proposed in the research report [9] how to determine cosine similarity neighborhoods of vectors by means of the Euclidean distance applied to normalized forms of these vectors and by using the triangle inequality. We have addressed there three types of sets of cosine similar vectors: all vectors, the similarity of which to a given vector is not less than an ε threshold value, and two variants of the k-nearest neighbors of a given vector. This chapter is an extendend version of report [9]. In particular, we show here that cosine similarity neighborhoods may be determined by means of the Euclidean distance applied to α-normalized forms of vectors, where $\alpha \neq 0$, and by using the triangle inequality. We also discuss here possible variants of the approach to calculating cosine similarity neighborhoods that was proposed in [9].

Our chapter has the following layout. Section 17.2 provides basic notions and properties used in the chapter. In particular, we examine properties and relationships among the three types of neighborhoods. In Section 17.3, we recall the methods offered in [5–8], which apply the triangle inequality property to efficiently calculate neighborhoods using a distance metric. The theoretical results which we derived in [9] for calculating cosine similarity neighborhoods based on the Euclidean distance, as well as their current generalizations and consequences, are presented in Section 17.4. In Section 17.5, we discuss and illustrate the ways of using these results for calculating the considered three types of cosine similarity neighborhoods. Section 17.6 concludes our work.

17.2 Basic Notions and Properties

17.2.1 Basic Operations on Vectors and Their Properties

In the chapter, we will consider vectors of the same dimensionality, say n. A vector u will be sometimes denoted as $[u_1, \ldots, u_n]$, where u_i is the value of the i-th dimension of u, $i = 1..n$. In Table 17.1, we recall definitions of basic operations on vectors. Table 17.2 presents their properties, which we will use in the chapter.

A *normalized form of a vector* u will be denoted by $NF(u)$ and will be defined as the ratio of u to its length $| u |$. A *vector* u will be called a *normalized vector* (or alternatively, a *unit vector*) if $u = NF(u)$. Clearly, the length of a normalized vector is equal to 1.

Table 17.1 Definitions of basic operations on vectors

Name of operation	Notation	Definition
Sum of vectors u and v	$u+v$	$[u_1+v_1,\ldots,u_n+v_n]$
Subtraction of vectors u and v	$u-v$	$[u_1-v_1,\ldots,u_n-v_n]$
Multiplication of vector u by scalar α	αv	$[\alpha u_1,\ldots,\alpha u_n]$
Division of vector u by scalar α	$\frac{u}{\alpha}$	$\frac{1}{\alpha}u$
Standard vector dot product of vectors u and v	$u\cdot v$	$\sum_{i=1..n} u_i v_i$
Length of vector u	$\lvert u\rvert$	$\sqrt{u\cdot u}$
Normalized form of a vector u	$NF(u)$	$\frac{u}{\lvert u\rvert}$

Table 17.2 Properties of operations on vectors

Properties of operations on vectors
$\lvert u\rvert^2 = u\cdot u = \sum_{i=1..n} u_i^2$
$(u+v)\cdot(u+v) = \sum_{i=1..n}(u_i+v_i)^2 = (u\cdot u)+(v\cdot v)+2(u\cdot v)$
$(u-v)\cdot(u-v) = \sum_{i=1..n}(u_i-v_i)^2 = (u\cdot u)+(v\cdot v)-2(u\cdot v)$
$NF(u)\ \cdot NF(u) = 1$
$\mid NF(u)\mid = 1$

In the chapter, we will also refer to notions of an α-*normalized form of a vector* and an α-*normalized vector*. An α-*normalized form of a vector u*, where $\alpha \neq 0$, is defined as $\alpha NF(u)$. A *vector u* is called an α-*normalized vector*, where $\alpha \neq 0$, if $u = \alpha NF(u)$. Clearly, the length of an α-normalized vector is equal to $\mid \alpha \mid$. If $\alpha = 1$, then obviously an α-normalized form of a vector is a normalized form of a vector and an α-normalized vector is a normalized vector.

17.2.2 Vector Dissimilarity and Similarity Measures

In the sequel, dissimilarity between two vectors p and q will be denoted by $dis(p, q)$. A vector q is considered as *less dissimilar* from vector p than vector r if $dis(q,p) < dis(r,p)$. In order to compare vectors, one may use a variety of dissimilarity measures among which an important class is *distance metrics*.

A *distance metric* (or shortly, *distance*) in a set of vectors D is defined as a dissimilarity measure $dis : D \times D \to [0, +\infty)$ that satisfies the following three conditions for all vectors p, q, and r in D:

1) $dis(p,q) = 0 \Leftrightarrow p = q$;
2) $dis(p,q) = dis(q,p)$;
3) $dis(p,r) \leq dis(p,q) + dis(q,r)$.

The third condition is known as the *the triangle inequality*. Often, an alternative form of this property, presented below, is more useful.

Property 17.2.2.1 (Alternative form of the triangle inequality). For any three vectors p, q, and r in a vector set D:

$$dis(p,q) \geq dis(p,r) - dis(q,r).$$

It was shown in [1, 5–8, 10, 11] how to use this property for efficient clustering of both low- and high-dimensional data.

The most popular distance metric is *the Euclidean distance*. *The Euclidean distance* between vectors u and v is denoted by $Euclidean(u, v)$ and is defined as follows:

$$Euclidean(u,v) = \sqrt{\sum_{i=1..n} (u_i - v_i)^2}.$$

Property 17.2.2.2. $Euclidean(u,v) = \sqrt{(u-v) \cdot (u-v)}$.

Sometimes, similarity measures are used rather than dissimilarity measures to compare vectors. In the following, the similarity between two vectors p and q will be denoted by $sim(p,q)$. A vector q is considered as *more similar* to vector p than vector r if $sim(q,p) > sim(r,p)$. Please note that, for example, $-sim(q,p)$ or $1 - sim(q,p)$ could be interpreted as a measure of dissimilarity between q and p.

In many applications, especially in text mining, *cosine similarity measure*, which is a function of the angle between two vectors, is applied. The *cosine similarity measure* between vectors u and v is denoted by $cosSim(u,v)$ and is defined as the cosine of the angle between them; that is,

$$cosSim(u,v) = \frac{u \cdot v}{|u| \cdot |v|}.$$

Example 17.2.2.1 (The Euclidean distance and the cosine similarity). Figure 17.1 presents three sample vectors p, q, and r. One may note that the Euclidean distance between p and q is greater than the Euclidean distance between r and q. On the other hand, in terms of the cosine similarity measure, p is more similar to q than r, since the cosine of the angle between p and q ($cosSim(p,q) = cos\alpha$) is greater than the cosine of the angle between r and q ($cosSim(r,q) = cos\beta$).

The cosine similarities of these vectors are presented in Table 17.3. One may note that neither $cosSim(p,q) \leq cosSim(p,r) + cosSim(r,q)$ nor $-cosSim(p,r) \leq -cosSim(p,q) + (-cosSim(q,r))$ nor $(1 - cosSim(p,r)) \leq (1 - cosSim(p,q)) + (1 - cosSim(q,r))$ (please, see Table 17.4). □

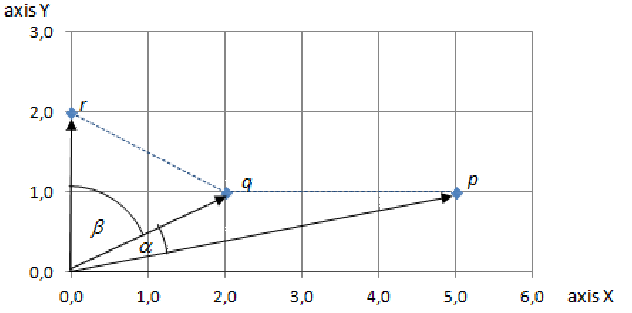


Fig. 17.1 The Euclidean distance and the cosine similarity

Table 17.3 The cosine similarity between vectors from Figure 17.1

(u,v)	$cosSim(u,v)$
(p,q)	0.965
(p,r)	0.196
(q,r)	0.447

Table 17.4 The cosine similarity versus the triangle inequality for vectors from Figure 17.1

The triangle inequality property	Fulfilled?
$\boldsymbol{cosSim(p,q) \leq cosSim(p,r) + cosSim(r,q)}$	**No**
$cosSim(p,r) \leq cosSim(p,q) + cosSim(q,r)$	Yes
$cosSim(q,r) \leq cosSim(q,p) + cosSim(p,r)$	Yes
$-cosSim(p,q) \leq -cosSim(p,r) + (-cosSim(r,q))$	Yes
$\boldsymbol{-cosSim(p,r) \leq -cosSim(p,q) + (-cosSim(q,r))}$	**No**
$\boldsymbol{-cosSim(q,r) \leq -cosSim(q,p) + (-cosSim(p,r))}$	**No**
$(1 - cosSim(p,q)) \leq (1 - cosSim(p,r)) + (1 - cosSim(r,q))$	Yes
$\boldsymbol{(1-cosSim(p,r)) \leq (1-cosSim(p,q)) + (1-cosSim(q,r)))}$	**No**
$(1 - cosSim(q,r)) \leq (1 - cosSim(q,p)) + (1 - cosSim(p,r))$	Yes

Corollary 17.2.2.1.

a) It is false that the triangle inequality holds for *cosSim* in each vector set.
b) It is false that the triangle inequality holds for $-cosSim$ in each vector set.
c) It is false that the triangle inequality holds for $(1-cosSim)$ in each vector set.

Clearly, the cosine similarity between any non-zero vectors u and v depends solely on the angle between them and does not depend on their lengths; hence the calculation of $cosSim(u,v)$ may be carried on $NF(u)$ and $NF(v)$ instead of u and v, respectively.

Property 17.2.2.3. Let u and v be non-zero vectors. Then:

a) $cosSim(NF(u),NF(v)) = NF(u) \cdot NF(v)$;
b) $cosSim(u,v) = cosSim(NF(u),NF(v))$;
c) $cosSim(u,v) = NF(u) \cdot NF(v)$.

17.2.3 Neighbourhoods Based on Dissimilarity Measures

Below, we provide definitions of neighborhoods in a given vector set D with respect to a given dissimilarity measure *dis*.

ε-neighborhood of a vector p in D is denoted by $\varepsilon\text{-}NB^D_{dis}(p)$ and is defined as any set of all vectors in $D\backslash\{p\}$ that are dissimilar to p by no more than ε; that is,

$$\varepsilon\text{-}NB^D_{dis}(p) = \{q \in D\backslash\{p\} \mid dis(p,q) \leq \varepsilon\}.$$

The set of all vectors in $D\backslash\{p\}$ that are less dissimilar to p than q will be denoted by $LessDissimilar^D_{dis}(p,q)$; that is,

$$LessDissimilar^D_{dis}(p,q) = \{s \in D\backslash\{p\} \mid dis(s,p) < dis(q,p)\}.$$

k-neighborhood of a vector p in D is denoted by $k\text{-}NB^D_{dis}(p)$ and is defined as the set of all vectors q in $D\backslash\{p\}$ such that the number of vectors in $D\backslash\{p\}$ that are less dissimilar to p than q is less than k; that is,

$$k\text{-}NB^D_{dis}(p) = \{q \in D\backslash\{p\} \mid \mid LessDissimilar^D_{dis}(p,q) \mid < k\}.$$

Please note that for any value k and for each vector p, one may determine a value of threshold ε in such a way that $\varepsilon\text{-}NB^D_{dis}(p) = k\text{-}NB^D_{dis}(p)$. In the following, the least value of ε such that $\varepsilon\text{-}NB^D_{dis}(p) = k\text{-}NB^D_{dis}(p)$ will be called the *radius* of $k\text{-}NB^D_{dis}(p)$.

Proposition 17.2.3.1 [8]. Let $\varepsilon = max(\{dis(q,p) \mid q \in k\text{-}NB^D_{dis}(p)\})$. Then $k\text{-}NB^D_{dis}(p) = \varepsilon\text{-}NB^D_{dis}(p)$ and ε is the radius of $k\text{-}NB^D_{dis}(p)$.

Proposition 17.2.3.2 [8]. If $\mid \varepsilon\text{-}NB^D_{dis}(p) \mid \geq k$, then $\varepsilon\text{-}NB^D_{dis}(p) \supseteq k\text{-}NB^D_{dis}(p)$.

Please note that $k\text{-}NB^D_{dis}(p)$ may contain more than k vectors. In some applications, it is of interest to determine a set of exactly k "nearest" vectors (neighbors) instead of $k\text{-}NB^D_{dis}(p)$.

k-nearest neighbors of a vector p in D are defined as any set of k vectors q in $D\backslash\{p\}$ such that the number of vectors in $D\backslash\{p\}$ that are less dissimilar to p than q is less than k.

Let $k\text{-}NN^D_{dis}(p)$ be a set of k-nearest neighbors of a vector p in D. Then the least value of ε such that $k\text{-}NN^D_{dis}(p) \subseteq \varepsilon\text{-}NB^D_{dis}(p)$ will be called the *radius of* $k\text{-}NN^D_{dis}(p)$.

Proposition 17.2.3.3 [9]. Let $k\text{-}NN^D_{dis}(p)$ be a set of k-nearest neighbors of a vector p in D, and ε be the radius of $k\text{-}NN^D_{dis}(p)$. Then:

a) $k\text{-}NN^D_{dis}(p) \subseteq k\text{-}NB^D_{dis}(p)$;
b) $\forall q \in k\text{-}NB^D_{dis}(p) \backslash\ k\text{-}NN^D_{dis}(p), dis(q,p) = \varepsilon$;
c) $k\text{-}NB^D_{dis}(p) = \varepsilon\text{-}NB^D_{dis}(p)$;
d) ε is the radius of $k\text{-}NB^D_{dis}(p)$;
e) ε is the radius of each set of k-nearest neighbors of vector p in D.

Corollary 17.2.3.1 [9]. Let $k\text{-}NN^D_{dis}(p)$ be a set of k-nearest neighbors of a vector p in D.

a) If $|\ \varepsilon\text{-}NB^D_{dis}(p)\ | \geq k$, then $\varepsilon\text{-}NB^D_{dis}(p) \supseteq k\text{-}NB^D_{dis}(p) \supseteq k\text{-}NN^D_{dis}(p)$;
b) If $|\ \varepsilon\text{-}NB^D_{dis}(p)\ | = k$, then $\varepsilon\text{-}NB^D_{dis}(p) = k\text{-}NB^D_{dis}(p) = k\text{-}NN^D_{dis}(p)$.

By Corollary 17.2.3.1a, if ε-neighborhood of a vector p contains at least k vectors in D, then ε-neighborhood of p in D contains k-neighborhood of p in D, which in turn contains k-nearest neighbors of p in D.

17.2.4 Neighbourhoods Based on Similarity Measures

In this subsection, we provide alternative definitions of neighborhoods in a given set D in terms of a similarity measure *sim*.

ε-similarity neighborhood of a vector p in D is denoted by $\varepsilon\text{-}SNB^D_{sim}(p)$ and is defined as the set of all vectors in $D\backslash\{p\}$ that are similar to p by no less than ε; that is,

$$\varepsilon\text{-}SNB^D_{sim}(p) = \{q \in D\backslash\{p\} \mid sim(p,q) \geq \varepsilon\}.$$

The set of all vectors in $D\backslash\{p\}$ that are less similar to p than q will be denoted by $MoreSimilar^D_{sim}(p,q)$; that is,

$$MoreSimilar^D_{sim}(p,q) = \{s \in D\backslash\{p\} \mid sim(s,p) > sim(q,p)\}.$$

k-similarity neighborhood of a vector p in D is denoted by $k\text{-}SNB^D_{sim}(p)$ and is defined as the set of all vectors q in $D\backslash\{p\}$ such that the number of vectors in $D\backslash\{p\}$ that are more similar to p than q is less than k; that is,

$$k\text{-}SNB^D_{sim}(p) = \{q \in D\backslash\{p\} \mid\ |\ MoreSimilar^D_{sim}(p,q)\ | < k\}.$$

Please note that for any value k and for each vector p, one may determine a value of threshold ε in such a way that ε-$SNB^D_{sim}(p) = k$-$SNB^D_{sim}(p)$. In the sequel, the greatest value of ε such that ε-$SNB^D_{sim}(p) = k$-$SNB^D_{sim}(p)$ will be called the *similarity radius of* k-$SNB^D_{sim}(p)$.

Proposition 17.2.4.1 [9]. Let $\varepsilon = \min(\{sim(q,p) \mid q \in k\text{-}SNB^D_{sim}(p)\})$. Then k-$SNB^D_{sim}(p) = \varepsilon$-$SNB^D_{sim}(p)$ and ε is the similarity radius of k-$SNB^D_{sim}(p)$.

Proposition 17.2.4.2 [9]. If $|\ \varepsilon\text{-}SNB^D_{sim}(p)| \geq k$, then ε-$SNB^D_{sim}(p) \supseteq k$-$SNB^D_{sim}(p)$.

k-similarity nearest neighbors of a vector p in D are defined as any set of k vectors q in $D\backslash\{p\}$ such that the number of vectors in $D\backslash\{p\}$ that are more similar to p than q is less than k.

Let k-$SNN^D_{sim}(p)$ be a set of k-similarity nearest neighbors of a vector p in D. Then the greatest value of ε such that k-$SNN^D_{sim}(p) \subseteq \varepsilon$-$SNB^D_{sim}(p)$ will be called the *similarity radius of* k-$SNN^D_{sim}(p)$.

Proposition 17.2.4.3 [9]. Let k-$SNN^D_{sim}(p)$ be a set of k-similarity nearest neighbors of a vector p in D and ε be the similarity radius of k-$SNN^D_{sim}(p)$. Then:

a) k-$SNN^D_{sim}(p) \subseteq k$-$SNB^D_{sim}(p)$;
b) $\forall q \in k\text{-}SNB^D_{sim}(p) \backslash\ k\text{-}SNN^D_{sim}(p), sim(q,p) = \varepsilon$;
c) k-$SNB^D_{sim}(p) = \varepsilon$-$SNB^D_{sim}(p)$;
d) ε is the similarity radius of k-$SNB^D_{sim}(p)$;
e) ε is the similarity radius of each set of k-similarity nearest neighbors of vector p in D.

Corollary 17.2.4.1 [9]. Let k-$SNN^D_{sim}(p)$ be a set of k-similarity nearest neighbors of a vector p in D.

a) If $|\ \varepsilon\text{-}SNB^D_{sim}(p)\ | \geq k$, then ε-$SNB^D_{sim}(p) \supseteq k$-$SNB^D_{sim}(p) \supseteq k$-$SNN^D_{sim}(p)$;
b) If $|\ \varepsilon\text{-}SNB^D_{sim}(p)\ | = k$, then ε-$SNB^D_{sim}(p) = k$-$SNB^D_{sim}(p) = k$-$SNN^D_{sim}(p)$.

By Corollary 17.2.4.1a, if ε-similarity neighborhood of a vector p contains at least k vectors in D, then ε-similarity neighborhood of p in D contains k- similarity neighborhood of p in D, which in turn contains k-similarity nearest neighbors of p in D.

17.3 The Triangle Inequality as a Mean for Efficient Determination of Neighborhoods Based on a Distance Metric

In this section, we present rudiments of the methods offered in [5–9] for speeding up the determination of the three types of neighboorhods based on a distance metric

by employing the triangle inequality to efficiently prune non-promising candidates for neighbors. The methods guarantee that the pruning will not eliminate any true neighbor.

17.3.1 Efficient Determination of ε-Neighborhoods Based on a Distance Metric

In this subsection, we recall the basic method of determining ε-neighborhoods based on a distance metric efficiently, as proposed in [5, 6].

Lemma 17.3.1.1 [5, 6]. Let dis be a distance metric and D be a set of vectors. For any two vectors p, q in D and any vector r:

$$dis(p,r) - dis(q,r) > \varepsilon \Rightarrow q \notin \varepsilon\text{-}NB^D_{dis}(p) \wedge p \notin \varepsilon\text{-}NB^D_{dis}(q).$$

Lemma 17.3.1.1 comes from the fact that $dis(p,r) - dis(q,r) > \varepsilon$ (by assumption) and $dis(p,q) \geq dis(p,r) - dis(q,r)$ (by the triangle inequality). Hence, $dis(p,q) > \varepsilon$. Thus, the fact that the difference of distances from two vectors p and q to any vector r is greater than ε implies that $q \notin \varepsilon\text{-}NB^D_{dis}(p)$ and $p \notin \varepsilon\text{-}NB^D_{dis}(q)$.

Now, let us consider a vector v such that $dis(v,r) > dis(p,r)$. If we know that $dis(p,r) - dis(q,r) > \varepsilon$, we may conclude that $dis(v,r) - dis(q,r) > \varepsilon$, and thus $q \notin \varepsilon\text{-}NB^D_{dis}(v)$ and $v \notin \varepsilon\text{-}NB^D_{dis}(q)$ without calculating the real distance between v and q. This observation provides intuition behind Theorem 17.3.1.1, as offered in [5, 6].

Theorem 17.3.1.1 [5, 6]. Let dis be a distance metric, r be any vector and D be a set of vectors ordered in a non-decreasing way with respect to their distances to r. Let p be any vector in D, q_f be a vector following vector p in D such that $dis(q_f,r) - dis(p,r) > \varepsilon$, and q_b be a vector preceding vector p in D such that $dis(p,r) - dis(q_b,r) > \varepsilon$. Then:

a) q_f and all vectors following q_f in D do not belong to $\varepsilon\text{-}NB^D_{dis}(p)$;
b) q_b and all vectors preceding q_b in D do not belong to $\varepsilon\text{-}NB^D_{dis}(p)$.

As follows from Theorem 17.3.1.1, it makes sense to order all vectors in a given vector set D with respect to their distances to a reference vector, say r, as this enables simple elimination of a potentially large subset of vectors that certainly do not belong to an ε-neighborhood of an analyzed vector.

Example 17.3.1.1. Let us consider a sample set D of two-dimensional vectors presented in Figure 17.2. We will illustrate the usefulness of Theorem 17.3.1.1 for determining the ε-neighborhood based on the Euclidean distance for vector $p =$ F in vector set D, given $\varepsilon = 0.5$. As a reference vector r, we will apply $[0,0]$. Table 17.5 illustrates the considered set D ordered in a non-decreasing way with

respect to the distances of its vectors to vector r. We note that $Euclidean(\mathrm{F},r) = 3.2$, the first vector q_f following vector F in D such that $Euclidean(q_f,r) - Euclidean(\mathrm{F},r) > \varepsilon$ is vector C ($Euclidean(\mathrm{C},r) - Euclidean(\mathrm{F},r) = 4.5 - 3.2 = 1.3 > \varepsilon$), and the first vector q_b preceding vector p in D such that $Euclidean(\mathrm{F},r) - Euclidean(q_b,r) > \varepsilon$ is G ($Euclidean(\mathrm{F},r) - Euclidean(\mathrm{G},r) = 3.2 - 2.4 = 0.8 > \varepsilon$). By Theorem 17.3.1.1, vectors C and G as well as each vector which either follows C or precedes G in D certainly do not belong to ε-$NB^D_{Euclidean}(F)$. As a result, only vector H out of eight vectors in D has a chance to belong to ε-$NB^D_{Euclidean}(\mathrm{F})$. So, H is the only vector for which it is necessary to calculate its actual distance to F in order to determine ε-$NB^D_{Euclidean}(\mathrm{F})$ properly. □

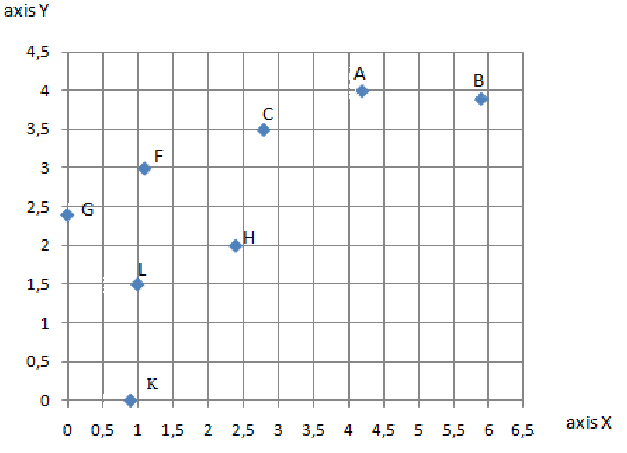


Fig. 17.2 Set of vectors D

The experimental evaluation of the usefulness of Theorem 17.3.1.1 was carried out in [5,6] by comparing the performance of density-based clustering carried out with the *TI-DBSCAN* algorithm and its variants, which used this theorem, and the *DBSCAN* algorithm [2] which used the *R-Tree* index [4]. As follows from the experiments reported in [5,6], the algorithms using the theorem were always faster, and in almost all cases speeded up the clustering process by at least an order of magnitude, also for high dimensional large vector sets consisting of hundreds of dimensions and tens of thousands of vectors.

17.3.2 *Efficient Determination of k-Neighborhoods Based on a Distance Metric*

In this subsection, we recall the basics of the methods of determining ε-neighborhoods based on a distance metric efficiently, as proposed in [7,8]. Also

Table 17.5 Ordered set of vectors D from Figure 17.2 with their Euclidean distances to reference vector r[0,0]

Q	X	Y	distance(Q,r)
K	0.9	0.0	0.9
L	1.0	1.5	1.8
G	0.0	2.4	2.4
H	2.4	2.0	3.1
F	**1.1**	**3.0**	**3.2**
C	2.8	3.5	4.5
A	4.2	4.0	5.8
B	5.9	3.9	7.1

in this case, all vectors in a given vector set D are assumed to be ordered with respect to a reference vector r. Then, for each vector p in D, its k-neighborhood can be determined in the following steps:

1) The radius, say ε, of k-neighborhood of p is estimated based on the real distances of k vectors located directly before and after p in the ordered set D.
2) Next the ε-neighborhood is determined in a way similar to the one described in Subsection 17.3.1. Clearly, the real distances to p from vectors considered in phase 1 do not need to be calculated again.
3) The k-neighborhood of p is determined as a subset of ε-neighborhood found in step 2.

The above description is a bit simplified. In [7, 8], steps 2 and 3 were not split, and the value of ε was adapted (narrowed) with each new candidate vector having a chance to belong to k-neighborhoods of p. Please see [7, 8] for a more detailed description. The presented approach was justified by Theorem 17.3.2.1, which has been offered.

Theorem 17.3.2.1 [7, 8]. Let dis be a distance metric, r be any vector and D be a set of vectors ordered in a non-decreasing way with respect to their distances to r. Let p be any vector in D and ε be a value such that $|\ \varepsilon\text{-}NB^{D}_{dis}(p)| \geq k, q_f$ be a vector following vector p in D such that $dis(q_f,r) - dis(p,r) > \varepsilon$, and q_b be a vector preceding vector p in D such that $dis(p,r) - dis(q_b,r) > \varepsilon$. Then:

a) q_f and all vectors following q_f in D do not belong to $k\text{-}NB^{D}_{dis}(p)$;
b) q_b and all vectors preceding q_b in D do not belong to $k\text{-}NB^{D}_{dis}(p)$.

Example 17.3.2.1. We will illustrate the usefulness of Theorem 17.3.2.1 for determining the k-neighborhood based on the Euclidean distance for vector p = F in

vector set D from Figure 17.2, given $k = 3$. As a reference vector r, we will apply [0, 0]. Table 17.5 illustrates the considered set D ordered in a non-decreasing way with respect to the distance of its vectors to vector r. Let us assume that we have calculated the distances between F and its two preceding vectors H, G, and one following vector C, respectively, and they are as follows: $Euclidean$(F, H) = 1.64, $Euclidean$(F, G) = 1.25, $Euclidean$(F, C) = 1.77. Now, we set the value of ε to max($Euclidean$(F, H), $Euclidean$(F, G), $Euclidean$(F, C)); that is, $\varepsilon = 1.77$. This means that vectors H,G,C $\in \varepsilon$-$NB^D_{Euclidean}(F)$ and $|\ \varepsilon$-$NB^D_{Euclidean}$(F) $| \geq k$. Thus, k-$NB^D_{Euclidean}$(F) will be found within the ε radius from F. Now, we note that the first vector q_f following vector F in D such that $Euclidean(q_f, r) - Euclidean(\mathrm{F}, r) > \varepsilon$ is vector A ($Euclidean(\mathrm{A}, r) - Euclidean(\mathrm{F}, r) = 5.8 - 3.2 = 2.6 > \varepsilon$), and the first vector q_b preceding vector F in D such that $Euclidean(\mathrm{F}, r) - Euclidean(q_b, r) > \varepsilon$ is K ($Euclidean(\mathrm{F}, r) - distance(\mathrm{K}, r) = 3.2 - 0.9 = 2.3 > \varepsilon$). By Theorem 17.3.2.1, vectors A, K as well as each vector that either follows A (that is, vector B) or precedes K in D (here, no vector precedes K) do not belong to k-$NB^D_{Euclidean}(F)$. □

The experimental evaluation of the usefulness of Theorem 17.3.2.1 was carried out in [7, 8] by comparing the performance of the *NBC* density based clustering [13] carried out with differently calculated k-neighborhoods; namely, by means of the index created by the *TI-k-Neighboorhood-Index* algorithm and its variants [7, 8], which used this theorem, as well as by means of the *VA-File* index [3] and the *R-Tree* index [4]. As follows from the experiments reported in [7, 8], the algorithms determining k-neighborhoods by means of the theorem were always faster, and in almost all cases speeded up the clustering process by at least an order of magnitude, also for high-dimensional large vector sets consisting of hundreds of dimensions and tens of thousands of vectors.

17.3.3 Efficient Determination of k-Nearest Neighbors Based on a Distance Metric

Let k-$NN^D_{dis}(p)$ be a set of k-similarity nearest neighbors of a vector p in D. Since k-$NB^D_{dis}(p) \supseteq k$-$NN^D_{dis}(p)$, then the vectors that do not belong to k-$NB^D_{dis}(p)$ do not belong to k-$NN^D_{dis}(p)$ either. This observation allows us to derive Proposition 17.3.3.1 from Theorem 17.3.2.1.

Proposition 17.3.3.1 [9]. Let dis be a distance metric, r be any vector and D be a set of vectors ordered in a non-decreasing way with respect to their distances to r. Let p be any vector in D, k-$NN^D_{dis}(p)$ be a set of k-similarity nearest neighbors of vector p in D, and ε be a value such that $|\ \varepsilon$-$NB^D_{dis}(p)\ | \geq k$, q_f be a vector following vector p in D such that $dis(q_f, r) - dis(p, r) > \varepsilon$, and q_b be a vector preceding vector p in D such that $dis(p, r) - dis(q_b, r) > \varepsilon$. Then:

a) q_f and all vectors following q_f in D do not belong to k-$NN^D_{dis}(p)$;
b) q_b and all vectors preceding q_b in D do not belong to k-$NN^D_{dis}(p)$.

Clearly, if $|k\text{-}NB^D_{dis}(p)| = k$, then $k\text{-}NN^D_{dis}(p) = k\text{-}NB^D_{dis}(p)$. Otherwise, $k\text{-}NN^D_{dis}(p)$ can be obtained by calculating $k\text{-}NB^D_{dis}(p)$, for instance, as presented in Section 17.3.2, and then removing all but one of the vectors from $k\text{-}NB^D_{dis}(p)$ that are distant from p by the radius of $k\text{-}NB^D_{dis}(p)$.

17.4 The Cosine Similarity Measure and Neighborhoods versus the Euclidean Distance and Neighborhoods

In Section 17.3, we described how neighborhoods expressed in terms of a distance metric can be calculated efficiently by using the triangle inequality property for skipping vectors that do not have a chance to belong to these neighborhoods. On the other hand, as we showed in Example 17.2.2.1, the cosine similarity measure is not a distance metric as the triangle inequality is not guaranteed to hold for it. However, in this section, we will show that cosine similarity neighborhoods are equivalent to corresponding neighborhoods based on the Euclidean distance.

17.4.1 Relationship between the Cosine Similarity and the Euclidean Distance

We start with formulating and proving two lemmas: one showing that the cosine similarity between two non-zero vectors can be expressed as a function of their lengths and the Euclidean distance between them (Lemma 17.4.1.1), and the next one showing that the cosine similarity between two normalized (forms of) non-zero vectors can be expressed as a function of solely their Euclidean distance (Lemma 17.4.1.2).

Lemma 17.4.1.1 [9]. Let u and v be non-zero vectors. Then:

$$\text{a) } cosSim(u,v) = \frac{(u\cdot u)+(v\cdot v)-(u-v)\cdot(u-v)}{2\mid u\parallel v\mid};$$

$$\text{b) } cosSim(u,v) = \frac{\mid u\mid^2+\mid v\mid^2-Euclidean^2(u,v)}{2\mid u\parallel v\mid}.$$

Proof. Ad a) Since $(u-v)\cdot(u-v) = (u\cdot u)+(v\cdot v)-2(u\cdot v)$ and $cosSim(u,v) = \frac{u\cdot v}{|u||v|}$, then $(u-v)\cdot(u-v) = (u\cdot u)+(v\cdot v)-2(cosSim(u,v)\mid u\parallel v\mid)$. Hence, $cosSim(u,v) = \frac{(u\cdot u)+(v\cdot v)-(u-v)\cdot(u-v)}{2|u||v|}$.
Ad b) Follows immediately from Lemma 17.4.1.1a, the fact that $u\cdot u = \mid u\mid^2$, $v\cdot v = \mid v\mid^2$ and $Euclidean(u,v) = \sqrt{(u-v)\cdot(u-v)}$. □

Lemma 17.4.1.2 [9]. Let u, v be non-zero vectors. Then:

$$cosSim(NF(u),NF(v)) = \frac{2 - Euclidean^2(NF(u),NF(v))}{2}.$$

Proof. $cosSim(NF(u),NF(v)) = \frac{|NF(u)|^2+|NF(v)|^2-Euclidean^2(NF(u),NF(v))}{2|NF(u)||NF(v)|}$ by Lemma 17.4.1.1b. Hence, since $|\ NF(u)\ | = |\ NF(v)\ | = 1$, we conclude that $cosSim(NF(u),NF(v)) = \frac{2-Euclidean^2(NF(u),NF(v))}{2}$. □

Now, as the cosine similarity of two non-zero vectors is equal to the cosine similarity of their normalized forms, we conclude from Lemma 17.4.1.2 that the cosine similarity between two non-zero vectors can be expressed as a function of solely the Euclidean distance of their normalized forms (Theorem 17.4.1.1).

Theorem 17.4.1.1 [9]. Let u,v be non-zero vectors. Then:

$$cosSim(u,v) = \frac{2 - Euclidean^2(NF(u),NF(v))}{2}.$$

Also, based on Theorem 17.4.1.1, we may conclude further that the cosine similarity between two non-zero vectors can be expressed as a function of solely α and the Euclidean distance of their α-normalized forms (Proposition 17.4.1.1).

Proposition 17.4.1.1. Let $\alpha \neq 0$ and u,v be non-zero vectors. Then:

$$cosSim(u,v) = \frac{2 - \frac{1}{\alpha^2}Euclidean^2(\alpha NF(u),\alpha NF(v))}{2}.$$

Proof. By Theorem 17.4.1.1, $cosSim(u,v) = \frac{2-Euclidean^2(NF(u),NF(v))}{2} = \frac{2-Euclidean^2(\frac{\alpha NF(u)}{\alpha},\frac{\alpha NF(v)}{\alpha})}{2} = \frac{2-\frac{1}{\alpha^2}Euclidean^2(\alpha NF(u),\alpha NF(v))}{2}$. □

Corollary 17.4.1.1. Let $\alpha \neq 0$ and u,v be α-normalized non-zero vectors. Then:

$$cosSim(u,v) = \frac{2 - \frac{1}{\alpha^2}Euclidean^2((u),(v))}{2}.$$

17.4.2 Vector Cosine Similarity Neighborhoods and Normalized Vector Neighborhoods based on the Euclidean Distance

In this subsection, we will use Theorem 17.4.1.1 to derive relationships between vector cosine similarity neighborhoods and corresponding normalized vector neighborhoods based on the Euclidean distance [9]. First, we start with Lemma 17.4.2.1a, in which we formulate and prove that a comparison of the cosine similarity between two non-zero vectors with an ε threshold is equivalent to a comparison of the Euclidean distance between their normalized forms with an ε' threshold being a function of ε. In Lemma 17.4.2.1b, we formulate and prove that a comparison of

the cosine similarity between any two non-zero vectors s and p with the cosine similarity between vector p and any vector q is equivalent to a comparison of the Euclidean distances between their normalized forms.

Lemma 17.4.2.1 [9]. Let p, q and s be non-zero vectors, $\varepsilon \in [-1,1]$ and $\varepsilon' = \sqrt{2-2\varepsilon}$. Then:

a) $cosSim(p,q) \geq \varepsilon$ iff $Euclidean(NF(p),NF(q)) \leq \varepsilon'$;
b) $cosSim(s,p) > cosSim(q,p)$ iff $Euclidean(NF(s),NF(p)) < Euclidean(NF(q),NF(p))$.

Proof. Ad a) $cosSim(p,q) \geq \varepsilon$ iff (by Theorem 17.4.1.1)
$\frac{2-Euclidean^2(NF(p),NF(q))}{2} \geq \varepsilon$ iff $Euclidean(NF(p),NF(q)) \leq \sqrt{2-2\varepsilon} = \varepsilon'$;
Ad b) $cosSim(s,p) > cosSim(q,p)$ iff (by Theorem 17.4.1.1)
$\frac{2-Euclidean^2(NF(s),NF(p))}{2} > \frac{2-Euclidean^2(NF(q),NF(p))}{2}$ iff
$Euclidean(NF(s),NF(p)) < Euclidean(NF(q),NF(p))$. □

Lemma 17.4.2.1 enables us to formulate and prove Lemma 17.4.2.2, in which we show that the problem of determining vectors in D that are more cosine similar to a given vector $p_{(i)}$ than another vector $p_{(j)}$ can be treated as the problem of determining the normalized forms of vectors from D that are less distant in the Euclidean sense from $NF(p_{(i)})$ than $NF(p_{(j)})$.

Lemma 17.4.2.2 [9]. Let D be an ordered set of m non-zero vectors $(p_{(1)},\ldots,p_{(m)})$, D' be the ordered set of m vectors $(u_{(1)},\ldots,u_{(m)})$ such that $u_{(i)} = NF(p_{(i)})$, $i = 1..m$, $\varepsilon \in$ [-1, 1] and $\varepsilon' = \sqrt{2-2\varepsilon}$. Then for any vectors $p_{(i)}, p_{(j)}$ in D:

a) $MoreSimilar^{D}_{cosSim}(p_{(i)},p_{(j)}) =$
$\{p_{(l)} \in D \backslash \{p_{(i)}\} \mid u_{(l)} \in LessDissimilar^{D'}_{Euclidean}(u_{(i)},u_{(j)})\}$;
b) $\mid MoreSimilar^{D}_{cosSim}(p_{(i)},p_{(j)}) \mid < k$ iff
$\mid LessDissimilar^{D'}_{Euclidean}(u_{(i)},u_{(j)}) \mid < k$.

Proof. Ad a) $MoreSimilar^{D}_{cosSim}(p_{(i)},p_{(j)}) =$
$\{p_{(l)} \in D \backslash \{p_{(i)}\} \mid cosSim(p_{(l)},p_{(i)}) > cosSim(p_{(j)},p_{(i)})\} =$
(by Lemma 17.4.2.1b)
$\{p_{(l)} \in D \backslash \{p_{(i)}\} \mid u_{(l)} \in D' \backslash \{u_{(i)}\} \wedge Euclidean(u_{(l)},u_{(i)})) <$
$Euclidean(u_{(j)},u_{(i)})\} =$
$\{p_{(l)} \in D \backslash \{p_{(i)}\} \mid u_{(l)} \in LessDissimilar^{D'}_{Euclidean}(u_{(i)},u_{(j)})\}$.
Ad b) Follows immediately from Lemma 17.4.2.2a. □

Now, we are ready to formulate and prove the equivalence of cosine similarity neighborhoods and corresponding neighborhoods based on the Euclidean distance.

Theorem 17.4.2.1 [9]. Let D be an ordered set of m non-zero vectors $(p_{(1)},\ldots,p_{(m)})$, D' be the ordered set of m vectors $(u_{(1)},\ldots,u_{(m)})$ such that $u_{(i)} = NF(p_{(i)})$, $i = 1..m$, $\varepsilon \in$[-1,1] and $\varepsilon' = \sqrt{2-2\varepsilon}$. Then:

a) $\varepsilon\text{-}SNB^{D}_{cosSim}(p_{(i)}) = \{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in \varepsilon'\text{-}NB^{D'}_{Euclidean}(u_{(i)})\}$;
b) $k\text{-}SNB^{D}_{cosSim}(p_{(i)}) = \{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in k\text{-}NB^{D'}_{Euclidean}(u_{(i)})\}$;
c) If $k\text{-}NN^{D'}_{Euclidean}(u_{(i)})$ is a set of k-nearest neighbours of $u_{(i)}$ in D', then $\{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in k\text{-}NN^{D'}_{Euclidean}(u_{(i)})\}$ is a set of k-similarity nearest neighbors of $p_{(i)}$ in D.

Proof. Ad a) $\varepsilon\text{-}SNB^{D}_{cosSim}(p_{(i)}) =$
$\{p_{(j)} \in D \backslash \{p_{(i)}\} \mid cosSim(p_{(i)}, p_{(j)}) \geq \varepsilon\} =$ (by Lemma 17.4.2.1a) $=$
$\{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in D' \backslash \{u_{(i)}\} \wedge Euclidean(u_{(i)}, u_{(j)}) \leq \varepsilon'\} =$
$\{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in \varepsilon'\text{-}NB^{D'}_{Euclidean}(u_{(i)})\}$.
Ad b) $k\text{-}SNB^{D}_{cosSim}(p_{(i)}) =$
$\{p_{(j)} \in D \backslash \{p_{(i)}\} \mid\mid MoreSimilar^{D}_{cosSim}(p_{(i)}, p_{(j)}) \mid < k\} =$
(by Lemma 17.4.2.2b) $= \{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in D' \backslash \{u_{(i)}\} \wedge$
$\mid LessDissimilar^{D'}_{Euclidean}(u_{(i)}, u_{(j)}) \mid < k\} =$
$\{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in k\text{-}NB^{D'}_{Euclidean}(u_{(i)})\}$.
Ad c) Let $k\text{-}NN^{D'}_{Euclidean}(u_{(i)})$ be a set of k-nearest neighbours of $u_{(i)}$ in D'.
Then, $k\text{-}NN^{D'}_{Euclidean}(u_{(i)})\}$ contains k vectors and thus
$\{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in k\text{-}NN^{D'}_{Euclidean}(u_{(i)})\}$ also contains k vectors.
In addition, the fact that $\{k\text{-}NN^{D'}_{Euclidean}(u_{(i)})\} \subseteq \{k\text{-}NB^{D'}_{Euclidean}(u_{(i)})\}$
implies that $\{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in k\text{-}NN^{D'}_{Euclidean}(u_{(i)})\} \subseteq$
$\{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in k\text{-}NB^{D'}_{Euclidean}(u_{(i)})\} =$
(by Theorem 17.4.2.1b) $= k\text{-}SNB^{D}_{cosSim}(p_{(i)})$.
Therefore, $\{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in k\text{-}NN^{D'}_{Euclidean}(u_{(i)})\}$ is a set of k-cosine similarity nearest neighbors of $p_{(i)}$ in D. □

One may easily observe that the equivalence of cosine similarity neighborhoods and neighborhoods based on the Euclidean distance that was stated in Theorem 17.4.2.1 becomes the equality in the case of normalized vectors.

Corollary 17.4.2.1. Let D be a set of normalized non-zero vectors, $p \in D$, $\varepsilon \in$[-1,1] and $\varepsilon' = \sqrt{2 - 2\varepsilon}$. Then:

a) $\varepsilon\text{-}SNB^{D}_{cosSim}(p) = \varepsilon'\text{-}NB^{D}_{Euclidean}(p)$;
b) $k\text{-}SNB^{D}_{cosSim}(p) = k\text{-}NB^{D}_{Euclidean}(p)$;
c) NN is a set of k-cosine similarity nearest neighbours of p in D iff NN is a set of k-nearest neighbours of p in D with regard to the Euclidean distance.

17.4.3 Vector Cosine Similarity Neighborhoods and α-Normalized Vector Neighborhoods based on the Euclidean Distance

In this subsection, we will generalize the results from Section 17.4.2, in that we will derive relationships between vector cosine similarity neighborhoods

and corresponding α-normalized vector neighborhoods based on the Euclidean distance.

Lemma 17.4.3.1. Let $\alpha \neq 0$, p, q and s be non-zero vectors, $\varepsilon \in [-1,1]$ and $\varepsilon' = |\alpha| \sqrt{2-2\varepsilon}$. Then:

a) $cosSim(p,q) \geq \varepsilon$ iff $Euclidean(\alpha NF(p), \alpha NF(q)) \leq \varepsilon'$;
b) $cosSim(s,p) > cosSim(q,p)$ iff $Euclidean(\alpha NF(s), \alpha NF(p)) < Euclidean(\alpha NF(q), \alpha NF(p))$.

Proof. Follows from Proposition 17.4.1.1. It can be proved in an analogous way as Lemma 17.4.2.1. □

Lemma 17.4.3.2. Let $\alpha \neq 0$, D be an ordered set of m non-zero vectors $(p_{(1)}, \ldots, p_{(m)})$, D' be the ordered set of m vectors $(u_{(1)}, \ldots, u_{(m)})$ such that $u_{(i)} = \alpha NF(p_{(i)})$, $i = 1..m$, $\varepsilon \in$ [-1, 1] and $\varepsilon' = |\alpha| \sqrt{2-2\varepsilon}$. Then for any vectors $p_{(i)}, p_{(j)}$ in D:

a) $MoreSimilar^{D}_{cosSim}(p_{(i)}, p_{(j)}) = \{p_{(l)} \in D \backslash \{p_{(i)}\} \mid u_{(l)} \in LessDissimilar^{D'}_{Euclidean}(u_{(i)}, u_{(j)})\}$;
b) $|MoreSimilar^{D}_{cosSim}(p_{(i)}, p_{(j)})| < k$ iff $|LessDissimilar^{D'}_{Euclidean}(u_{(i)}, u_{(j)})| < k$.

Proof. Ad a) Follows from Lemma 17.4.3.1b.
Ad b) Follows immediately from Lemma 17.4.3.2a. □

Theorem 17.4.3.1. Let $\alpha \neq 0$, D be an ordered set of m non-zero vectors $(p_{(1)}, \ldots, p_{(m)})$, D' be the ordered set of m vectors $(u_{(1)}, \ldots, u_{(m)})$ such that $u_{(i)} = \alpha NF(p_{(i)})$, $i = 1..m$, $\varepsilon \in$[-1,1] and $\varepsilon' = |\alpha| \sqrt{2-2\varepsilon}$. Then:

a) $\varepsilon\text{-}SNB^{D}_{cosSim}(p_{(i)}) = \{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in \varepsilon'\text{-}NB^{D'}_{Euclidean}(u_{(i)})\}$;
b) $k\text{-}SNB^{D}_{cosSim}(p_{(i)}) = \{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in k\text{-}NB^{D'}_{Euclidean}(u_{(i)})\}$;
c) If $k\text{-}NN^{D'}_{Euclidean}(u_{(i)})$ is a set of k-nearest neighbours of $u_{(i)}$ in D', then $\{p_{(j)} \in D \backslash \{p_{(i)}\} \mid u_{(j)} \in k\text{-}NN^{D'}_{Euclidean}(u_{(i)})\}$ is a set of k-similarity nearest neighbors of $p_{(i)}$ in D.

Proof. Analogous to the proof of Theorem 17.4.2.1.
Ad a) Follows from Lemma 17.4.3.1a.
Ad b) Follows from Lemma 17.4.3.2b.
Ad c) Follows from Theorem 17.4.3.1b. □

Corollary 17.4.3.1. Let D be a set of α-normalized non-zero vectors, $p \in D$, $\varepsilon \in$[-1,1] and $\varepsilon' = |\alpha| \sqrt{2-2\varepsilon}$. Then:

a) $\varepsilon\text{-}SNB^{D}_{cosSim}(p) = \varepsilon'\text{-}NB^{D}_{Euclidean}(p)$;
b) $k\text{-}SNB^{D}_{cosSim}(p) = k\text{-}NB^{D}_{Euclidean}(p)$;
c) NN is a set of k-cosine similarity nearest neighbours of p in D iff NN is a set of k-nearest neighbours of p in D with regard to the Euclidean distance.

17.5 Determination of Cosine Similarity Neighborhoods as Determination of Neighborhoods Based on the Euclidean Distance

Theorem 17.4.3.1 (Theorem 17.4.2.1) tells us that cosine similarity neighborhoods in vector set D can be determined as respective neighborhoods based on the Euclidean distance in vector set D' consisting of α-normalized (normalized) forms of the vectors from D. Thus, we propose the following approach for the determination of cosine similarity neighborhoods:

First, a set of original vectors, say $D = (p_{(1)}, \ldots, p_{(m)})$, should be transformed to the set $D' = (u_{(1)}, \ldots, u_{(m)})$ of their $\alpha-$normalized forms. Then, ε-cosine similarity neighborhoods (or alternatively, k-cosine similarity neighborhoods or k-cosine similarity nearest neighbors, respectively) in vector set D should be found as ε'-neighborhoods, where $\varepsilon' = \mid \alpha \mid \sqrt{2-2\varepsilon}$, (or alternatively, k-neighborhoods or k-nearest neighbors) in vector set D' with regard to the Euclidean distance by means of the triangle inequality property.

Examples 17.5.1 and 17.5.2 illustrate this approach for the determining cosine similarity neigboorhoods.

Example 17.5.1 (Determination of an ε-cosine similarity neighborhood). In this example, we will consider determination of ε-cosine similarity neighborhood of vector $p_{(3)}$ in vector set $D = (p_{(1)}, \ldots, p_{(8)})$ from Fig. 17.3 (and Table 17.6) applying, for instance, $\alpha = 1$. We assume that the cosine similarity threshold $\varepsilon = 0.9856$, which roughly corresponds to the angle 9.74°. Figure 17.4 shows set $D' = (u_{(1)}, \ldots, u_{(8)})$ that contains α-normalized forms of vectors from D. Clearly, the lengths of all of them are equal to $\mid \alpha \mid$; that is, 1. Now, we will determine the corresponding Euclidean distance threshold ε' as $\mid \alpha \mid \sqrt{2-2\varepsilon}$ (according to Theorem 17.4.3.1a). Hence, $\varepsilon' \approx 0.1697 \leq 0.17$. At this moment, we may start the procedure of determining ε-cosine similarity neighborhood for vector $p_{(3)}$ in vector set D as the procedure of determining ε'-neighborhood for vector $u_{(3)}$ in vector set D' of α-normalized vectors with regard to the Euclidean distance.

The vectors in D' need to be sorted with regard to their Euclidean distances to a same reference vector. For the sake of the example, we choose $r = [1,0]$ as a reference vector. Table 17.7 shows set D' ordered in a non-decreasing way with regard to the Euclidean distances of its vectors to vector r.

As follows from Table 17.7, the first vector q_f following vector $u_{(3)}$ in D' for which $Euclidean(q_f,r) - Euclidean(u_{(3)},r) > \varepsilon'$ is vector $u_{(4)}$ ($Euclidean(u_{(4)},r) - Euclidean(u_{(3)},r) = 1.15 - 0.87 = 0.28 > \varepsilon'$), and the first vector q_b preceding vector $u_{(3)}$ in D', for which $Euclidean(u_{(3)},r) - Euclidean(q_b,r) > \varepsilon'$ is vector $u_{(6)}$($Euclidean(u_{(3)},r) - Euclidean(u_{(6)},r) = 0.87 - 0.68 = 0.19 > \varepsilon'$). Thus, by Theorem 17.3.1.1, neither vector $u_{(4)}$ nor the vectors following $u_{(4)}$ in D' as well as neither vector $u_{(6)}$ nor the vectors preceding $u_{(6)}$ in D' belong to $\varepsilon'\text{-}NB^{D'}_{Euclidean}(u_{(3)})$. Hence, only vectors $u_{(1)}$ and $u_{(8)}$ may belong to $\varepsilon'\text{-}NB^{D'}_{Euclidean}(u_{(3)})$, and only for these vectors it is necessary to

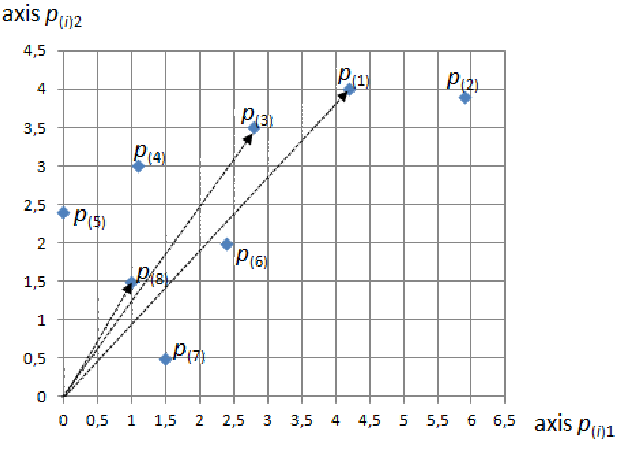


Fig. 17.3 Sample set D of vectors

Table 17.6 Sample set D

Vector $p_{(i)}$	$p_{(i)1}$	$p_{(i)2}$
$p_{(1)}$	4.20	4.00
$p_{(2)}$	5.90	3.90
$\boldsymbol{p_{(3)}}$	**2.80**	**3.50**
$p_{(4)}$	1.10	3.00
$p_{(5)}$	0.00	2.40
$p_{(6)}$	2.40	2.00
$p_{(7)}$	1.50	0.50
$p_{(8)}$	1.00	1.50

calculate their real Euclidean distances to $u_{(3)}$. These distances are as follows, $Euclidean(u_{(1)}, u_{(3)}) = 0.13$ and $Euclidean(u_{(8)}, u_{(3)}) = 0.07$. Since both values are less than ε', then $\varepsilon'\text{-}NB^{D'}_{Euclidean}(u_{(3)}) = \{u_{(1)}, u_{(8)}\}$, and by Theorem 17.4.3.1a, $\varepsilon\text{-}SNB^{D}_{cosSim}(p_{(3)}) = \{p_{(1)}, p_{(8)}\}$. Similarly, one may determine ε-cosine similarity neighborhood for the remaining vectors in D using already sorted set D'. □

Example 17.5.2 (Determination of a k-cosine similarity neighborhood and k-cosine similarity nearest neighbors). In this example, we will first consider determination of a k-cosine similarity neighborhood, where $k = 2$, of vector $p_{(3)}$ in the vector set $D = (p_{(1)}, \ldots, p_{(8)})$ from Figure 17.3 (and Table 17.6). Then we will determine k-cosine similarity nearest neigbors $k\text{-}SNN^{D}_{cosSim}(p_{(3)})$. In the example, we will apply $\alpha = 1$.

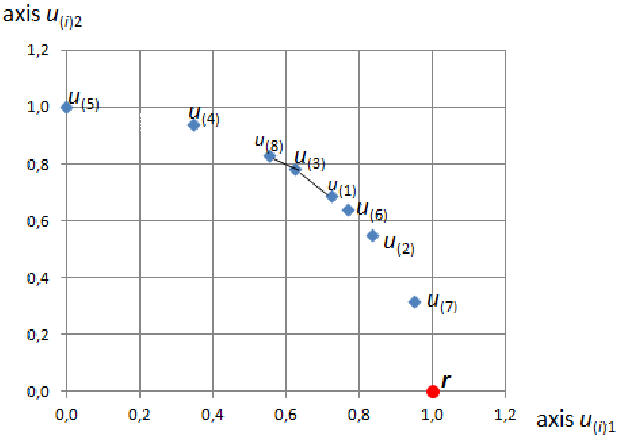


Fig. 17.4 Set D' containing normalized forms of vectors from D

Table 17.7 Normalized vectors in set D' sorted w.r.t. their distances to vector $r = [1,0]$

Vector $u_{(i)}$	$u_{(i)1}$	$u_{(i)2}$	$Euclidean(u_{(i)}, r)$
$u_{(7)}$	0.95	0.32	0.32
$u_{(2)}$	0.83	0.55	0.58
$u_{(6)}$	0.77	0.64	0.68
$u_{(1)}$	0.72	0.69	0.74
$\boldsymbol{u_{(3)}}$	**0.62**	**0.78**	**0.87**
$u_{(8)}$	0.55	0.83	0.94
$u_{(4)}$	0.34	0.94	1.15
$u_{(5)}$	0.00	1.00	1.41

We start with the calculation of set $D' = (u_{(1)}, \ldots, u_{(8)})$ of α-normalized forms of vectors from D. Figure 17.4 presents D'. Now, the determination of k-cosine similarity neighborhood for vector $p_{(3)}$ in vector set D can be performed as the determination of k-neighborhood $k\text{-}NB^{D'}_{Euclidean}(u_{(3)})$ of vector $u_{(3)}$ in set D' of α-normalized vectors with regard to the Euclidean distance. This procedure starts with ordering D' with regard to the Euclidean distances of its vectors to a same reference vector r. In the example, we assume $r = [1,0]$. Table 17.7 shows the result of this sorting.

Now, we need to estimate the radius within which k-nearest neighbors of $u_{(3)}$ occur. Let us assume that we have calculated the distances between $u_{(3)}$ and its directly preceding and following vectors in D'; that is, $u_{(1)}$ and $u_{(8)}$, respectively. These

distances are as follows: $Euclidean(u_{(1)},u_{(3)}) = 0.13$ and $Euclidean(u_{(8)},u_{(3)}) = 0.07$. Let $\varepsilon' = \max(Euclidean(u_{(1)},u_{(3)}), Euclidean(u_{(8)},u_{(3)}))$; that is, $\varepsilon' = 0.13$. Please note that $u_{(1)},u_{(8)} \in \varepsilon'\text{-}NB^{D'}_{Euclidean}(u_{(3)})$ and $\mid \varepsilon'\text{-}NB^{D'}_{Euclidean}(u_{(3)}) \mid \geq k$. The latter fact implies that $\varepsilon'\text{-}NB^{D'}_{Euclidean}(u_{(3)})$ contains k-neighborhood of $u_{(3)}$ in D' (by Corollary 17.2.3.1). Nevertheless, it is yet not certain if $u_{(1)}$ and/or $u_{(8)}$ belong to this neigborhood of $u_{(3)}$ in D'.

As follows from Table 17.7, the first vector q_f following vector $u_{(3)}$ in D' for which $Euclidean(q_f,r) - Euclidean(u_{(3)},r) > \varepsilon'$ is vector $u_{(4)}$ ($Euclidean(u_{(4)},r) - Euclidean(u_{(3)},r) = 1.15 - 0.87 = 0.28 > \varepsilon'$), and the first vector q_b preceding vector $u_{(3)}$ in D', for which $Euclidean(u_{(3)},r) - Euclidean(q_b,r) > \varepsilon'$ is vector $u_{(6)}$ ($Euclidean(u_{(3)},r) - Euclidean(u_{(6)},r) = 0.87 - 0.68 = 0.19 > \varepsilon'$). Thus, by Theorem 17.3.2.1, neither vector $u_{(4)}$ nor the vectors following $u_{(4)}$ in D' as well as neither vector $u_{(6)}$ nor the vectors preceding $u_{(6)}$ in D' belong to $k\text{-}NB^{D'}_{Euclidean}(u_{(3)})$. Hence, only vectors $u_{(1)}$ and $u_{(8)}$ may belong to $k\text{-}NB^{D'}_{Euclidean}(u_{(3)})$ and only for these vectors it is necessary to calculate their real Euclidean distances to $u_{(3)}$. As $k = 2$ and only two vectors: $u_{(1)}$ and $u_{(8)}$ were not eliminated, they constitute $k\text{-}NB^{D'}_{Euclidean}(u_{(3)})$. Thus, by Theorem 17.4.3.1b, $k\text{-}SNB^{D}_{cosSim}(p_{(3)}) = \{p_{(1)},p_{(8)}\}$.

Please note that in our example we had to calculate the Euclidean distance to vector $u_{(3)}$ only from two out of eight vectors in D'.

Now, let us consider the determination of $k\text{-}SNN^{D}_{cosSim}(p_{(3)})$. In our example, $k\text{-}NB^{D'}_{Euclidean}(u_{(3)})$, and consequently $k\text{-}SNB^{D}_{cosSim}(p_{(3)})$, have exactly k vectors each. Hence, $k\text{-}SNN^{D}_{cosSim}(p_{(3)}) = k\text{-}SNB^{D}_{cosSim}(p_{(3)}) = \{p_{(1)},p_{(8)}\}$. In the following, however, we present how $k\text{-}SNN^{D}_{cosSim}(p_{(3)})$ could be determined if we ignored the fact that the cardinality of $k\text{-}NB^{D'}_{Euclidean}(u_{(3)})$ (and the cardinality of $k\text{-}SNB^{D}_{cosSim}(p_{(3)})\}$) is k. First, one could determine k-nearest neighbors of $(u_{(3)})$ in D' based on $k\text{-}NB^{D'}_{Euclidean}(u_{(3)})$, the radius of which is known and equals the maximum of the Euclidean distances between $u_{(3)}$ and the vectors in $k\text{-}NB^{D'}_{Euclidean}(u_{(3)})$; that is, 0.13. Thus, k-nearest neighbors of $u_{(3)}$ in D' would contain all vectors from $k\text{-}NB^{D'}_{Euclidean}(u_{(3)})$ that are less distant from $u_{(3)}$ than 0.13 (here: only vector $u_{(8)}$) and exactly one arbitrary vector in $k\text{-}NB^{D'}_{Euclidean}(u_{(3)})$ that is distant from $u_{(3)}$ by 0.13 (here: vector $(u_{(1)})$. Next, by Theorem 17.4.3.1c, $k\text{-}SNN^{D}_{cosSim}(p_{(3)})$ would be found as $\{p_{(1)},p_{(8)}\}$. □

Please note that it is not necessary to verify if an α-normalized vector $u_{(l)}$ is a true neighbor of an analyzed α-normalized vector $u_{(i)}$ in D' by calculating the Euclidean distance between them. The verification can be carried out by calculating $cosSim(u_{(l)},u_{(i)})$ as $\frac{1}{\alpha^2}(u_{(l)} \cdot u_{(i)})$ (or simply $u_{(l)} \cdot u_{(i)}$ if vectors $u_{(l)},u_{(i)}$ are normalized) and, eventually, employing the cosine threshold ε provided $\varepsilon' = \mid \alpha \mid \sqrt{2-2\varepsilon}$. In fact, it is not even necessary to store α-normalized forms of vectors from D. It is sufficient to store only a sorted index containing the Euclidean distances of α-normalized forms of vectors from D to a reference vector. When the α-normalized forms are not stored, the verification for a (non-stored) α-normalized vector $u_{(l)}$ can

be carried on the corresponding original vectors in D as follows: $cosSim(u_{(l)}, u_{(i)}) = cosSim(p_{(l)}, p_{(i)}) = \frac{p_{(l)} \cdot p_{(i)}}{|p_{(l)}||p_{(i)}|}$.

17.6 Conclusions

In this chapter, we have offered a new solution to determining vector cosine similarity neighborhoods that consists in transforming the original problem into the problem of determining neighborhoods of (α-)normalized forms of the original vectors with regard to the Euclidean distance. We have discussed possible variants of the approach to calculating cosine similar neighborhoods that was proposed in [9] and based on applying normalized forms of vectors. The fact that the the problem of determining cosine similarity neighborhoods is transformable to the problem of determining neighborhoods based on the Euclidean distance allows applying the triangle inequality, which was proved in [1, 5–8, 10, 11] to be a powerful tool for making the neighboorhod determination efficient even in the case of high dimensional large vector sets consisting of hundreds of dimensions and tens of thousands of vectors. As a consequence, our solution helps to surpass the curse of dimensionality in the case of determining cosine similarity neighborhoods.

Acknowledgments. This work was supported by the National Centre for Research and Development (NCBiR) under Grant No. SP/I/1/77065/10 devoted to the Strategic scientific research and experimental development program: “Interdisciplinary System for Interactive Scientific and Scientific-Technical Information".

References

1. Elkan, C.: Using the triangle inequality to accelerate k-means. In: Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003), Washington, DC, USA, August 21-24, pp. 147–153. AAAI Press (2003)
2. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial database with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD 1996), Portland, Oregon, USA, August 2-4, pp. 226–231. AAAI Press (1996)
3. Blott, S., Weber, R.: A simple vector approximation file for similarity search in high-dimensional vector spaces. Technical Report 19, ESPRIT project HERMES, vol. 9141 (1997)
4. Guttman, A.: R-Trees: A dynamic index structure for spatial searching. In: SIGMOD 1984, Proceedings of Annual Meeting, Boston, Massachusetts, USA, June 18-21. ACM SIGMOD, pp. 47–57 (1984)
5. Kryszkiewicz, M., Lasek, P.: TI-DBSCAN: Clustering with DBSCAN by means of the triangle inequality. ICS Research Report 3, Institute of Computer Science, Warsaw University of Technology, Warsaw (2010)
6. Kryszkiewicz, M., Lasek, P.: TI-DBSCAN: Clustering with DBSCAN by Means of the Triangle Inequality. In: Szczuka, M., Kryszkiewicz, M., Ramanna, S., Jensen, R., Hu, Q. (eds.) RSCTC 2010. LNCS, vol. 6086, pp. 60–69. Springer, Heidelberg (2010)

7. Kryszkiewicz, M., Lasek, P.: A Neighborhood-Based Clustering by Means of the Triangle Inequality. In: Fyfe, C., Tino, P., Charles, D., Garcia-Osorio, C., Yin, H. (eds.) IDEAL 2010. LNCS, vol. 6283, pp. 284–291. Springer, Heidelberg (2010)
8. Kryszkiewicz, M., Lasek, P.: A neighborhood-based clustering by means of the triangle inequality and reference points. ICS Research Report 3, Institute of Computer Science, Warsaw University of Technology, Warsaw (2011)
9. Kryszkiewicz, M.: Efficient determination of neighborhoods defined in terms of cosine similarity measure. ICS Research Report 4, Institute of Computer Science, Warsaw University of Technology, Warsaw (2011)
10. Moore, A.W.: The anchors hierarchy: Using the triangle inequality to survive high dimensional data. In: Proceedings of the 16th Conference in Uncertainty in Artificial Intelligence (UAI 2000), Stanford, California, USA, June 30-July 3, pp. 397–405. Morgan Kaufmann, San Francisco (2000)
11. Patra, B.K., Hubballi, N., Biswas, S., Nandi, S.: Distance Based Fast Hierarchical Clustering Method for Large Datasets. In: Szczuka, M., Kryszkiewicz, M., Ramanna, S., Jensen, R., Hu, Q. (eds.) RSCTC 2010. LNCS, vol. 6086, pp. 50–59. Springer, Heidelberg (2010)
12. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Communications of the ACM 18(11), 613–620 (1975)
13. Zhou, S., Zhao, Y., Guan, J., Huang, J.Z.: A Neighborhood-Based Clustering Algorithm. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 361–371. Springer, Heidelberg (2005)

Chapter 18
Time Variability-Based Hierarchic Recognition of Multiple Musical Instruments in Recordings

Elżbieta Kubera, Alicja A. Wieczorkowska, and Zbigniew W. Raś

Abstract. The research reported in this chapter is focused on automatic identification of musical instruments in polyphonic audio recordings. Random forests have been used as a classification tool, pre-trained as binary classifiers to indicate presence or absence of a target instrument. Feature set includes parameters describing frame-based properties of a sound. Moreover, in order to capture the patterns which emerge on the time scale, new temporal parameters are introduced to supply additional temporal information for the timbre recognition. In order to achieve higher estimation rate, we investigated a feature-driven hierarchical classification of musical instruments built using agglomerative clustering strategy. Experiments showed that the performance of classifiers based on this new classification of instruments schema is better than performance of the traditional flat classifiers, which directly estimate the instrument. Also, they outperform the classifiers based on the classical Hornbostel-Sachs schema.

Keywords: Music information retrieval, automatic indexing, timbre recognition, pitch tracking, Hornbostel-Sachs system, temporal data mining, random forest, agglomerative clustering.

Elżbieta Kubera
University of Life Sciences in Lublin, Akademicka 13, 20-950 Lublin, Poland
e-mail: elzbieta.kubera@up.lublin.pl

Alicja A. Wieczorkowska
Polish-Japanese Institute of Information Technology, Koszykowa 86, 02-008 Warsaw, Poland
e-mail: alicja@poljap.edu.pl

Zbigniew W. Raś
University of North Carolina, Dept. of Computer Science, Charlotte, NC 28223, USA &
Warsaw University of Technology, Institute of Computer Science, 00-665 Warsaw, Poland &
Polish Academy of Sciences, Institute of Computer Science, 01-237 Warsaw, Poland
e-mail: ras@uncc.edu

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 347–363.
springerlink.com

18.1 Introduction

In recent years, rapid advances in digital music creation, collection and storage technology have enabled various organizations to accumulate vast amounts of musical audio data. The booming of multimedia resources in the Internet brought a tremendous need to provide new, more advanced tools for querying and processing vast quantities of musical data. Many multimedia resources provide data which are manually labeled with some description information, such as title, author, company, and so on. However, in most cases those labels are insufficient for content-based searching. This problem attracted the attention of academia and industry, and initiated research in Music Information Retrieval (MIR) some years ago. As the outcome of this research, various MIR systems emerged, addressing diverse needs of the users of audio data, including audio identification (finding a title and a performer of a given excerpt, re-played or even hummed), identification of style or music genre, or audio alignment (*e.g.*, score following), etc.; examples of systems available at commercial web sites can be found at [15], [23], and systems being part of research are described in [16], [17], see also papers in [21], [22], and so forth.

Extraction of pitch, so-called pitch tracking, is performed in some of the MIR systems, and it is quite accurate in the case of melodies when only one sound is played at a time. Clearly, multi-pitch extraction (for chords) is more challenging and the problem of assigning each pitch to appropriate part of the score has to be tackled. Automatic assignment of notes to particular voices would be facilitated if instruments participating in each chord were automatically identified. The research presented in this chapter addresses automatic identification of instruments in polyphonic multi-instrumental recordings.

Timbre recognition is one of the subtasks in MIR, and it has proven to be extremely challenging especially in multi-timbre sounds, where multiple instruments are playing at the same time. Compared with this, automatic recognition of an instrument in the case of single sounds (no chords) is relatively easy, and it has been investigated, starting in the twentieth century, by many researchers. The obtained accuracy depends on the number of sounds and instruments taken into account, a feature set used, and a classifier applied, as well as the validation method utilized. Even 100% can be achieved for a small number of sounds/instruments classified with an artificial neural network, but usually is lower, and generally decreases with increasing number of instruments, even below 40% when the number of instruments approaches thirty and full range of each instrument is taken into account. We should also notice that audio data, represented as a long sequence of amplitude values (44100 samples per second per channel is a standard for CD), may vary significantly, depending on many factors, *e.g.*, recording conditions, playing method, the player and his or her particular instrument, etc. Therefore, audio data are usually parameterized before applying classifiers, and the extracted feature vector also strongly influences the obtained results. The feature set can be based on the time-domain representation describing the sound amplitude or the spectrum obtained from the sound analysis describing frequency contents derived from short audio frames and

we also believe that temporal changes of various sound features can be beneficial as the sound may undergo substantial changes in time (see Figure 18.1). Spectral features are most often extracted using Fourier transform but other analyses are applied as well, *e.g.*, wavelet transform yielding time-frequency representation.

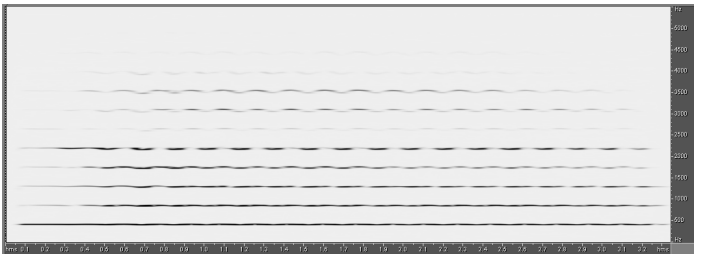


Fig. 18.1 Spectrogram (sonogram) for A4 (440 Hz) sound of violin, played vibrato. The spectrogram shows temporal changes of the sound spectrum. Horizontal axis represents time, and vertical axis represents frequency. The darker the shade of gray, the higher the magnitude.

Feature sets vary depending on the researcher; there is no standard feature set. However, many low-level audio descriptors from the MPEG-7 standard of multimedia content description [8] are often used. Mel-Frequency Cepstral Coefficients (MFCC), originating from speech recognition, can also be applied for MIR purposes [4], including recognition of musical instruments [2]. In our research, we apply various short-time sound features describing properties of the sound in time domain and its spectrum; besides, we add temporal features to this basic set in order to capture time-variability of the sound features. Detailed description of the feature set used in this research is presented in Section 18.3.

As it was mentioned before, the accuracy of instrument identification also depends on the classifier. The algorithms applied in experiments on instrument recognition include k-nearest neighbors (k-NN), artificial neural networks (ANN), rough-set-based classifiers, support vector machines (SVM), Gaussian mixture models (GMM), decision trees and random forests, and so on. The review of the outcomes of this research is given in [6](see also [9]). Although the obtained accuracies are far from being perfect when the number of instruments to be recognized is big, simple algorithm as k-NN may still yield good results. However, algorithms successfully identifying instruments playing single and isolated sounds can be prone to errors when executed on continuous polyphonic data (multi-instrumental chords), as happens in recordings, even when tried on duets [14]. Identification of instruments in the case of chords is much more challenging, and more sophisticated algorithms are advised to be used. For instance, ANN yielded over 80% accuracy for several four-instrument sets [10]; GMM classifier yielded about 60% accuracy for duets from five-instrument set [3]; random forests produced about 75% accuracy on average [11] for 2–5 instruments from 10-instrument sets, with variable accuracy obtained for particular instruments. Since random forests are quite robust with

respect to noise [1], and already proved to be rather successful in the instrument identification task, we decided to apply this classification technique in the reported research.

18.1.1 Random Forests

A random forest (RF) is an ensemble of classification trees, constructed using procedure minimizing bias and correlations between individual trees. Each tree is built using different N-element bootstrap sample of the training N-element set. The elements of the sample are drawn with replacement from the original set, so roughly one-third of the training data is not used in the bootstrap sample for any given tree.

Let us assume that objects are described by a vector of P attributes (features). At each stage of tree building, i.e., for each node of any particular tree in RF, p attributes out of all P attributes are randomly selected ($p \ll P$, often $p = \sqrt{P}$). The best split on these p attributes is used to split the data in the node. It is determined as minimizing the Gini impurity criterion, which is a measure how often an element would be incorrectly labeled if labeled randomly, according to the distribution of labels in the subset.

Each tree is grown to the largest extent possible (without pruning). By repeating this randomized procedure M times one obtains a collection of M trees, which constitute a random forest. Classification of each object is made by simple voting of all trees [1].

18.1.2 Outline of the Paper

The experiments presented in this chapter concern identification of multiple instruments in polyphonic multi-instrumental recordings. Feature sets used here contain both frame-based audio parameters, as well as new parameters describing temporal variability of the frame-based features. The training audio data were taken from two repositories, commonly used in similar research worldwide. Testing data represent audio recordings of classical music, as we decided to focus our research on this music genre. The testing data were manually labeled in a careful way in order to create ground-truth data. Random forests have been applied as classifiers, also for hierarchical classification, including feature-driven hierarchy. The details of this research are presented in the next sections of our chapter; audio data are described in Section 18.2, features for sound parameterization are shown in Section 18.3, and the experiments are presented and discussed in Section 18.4. The chapter is summarized and concluded in Section 18.5.

18.2 Audio Data

Music we listen to can be played by numerous instruments; in various music genres, typical sets of instruments are usually used. For instance, electric guitars and drums etc. are commonly used in rock music; violins, violas etc. are commonly used in classical music; and so on. The music collections available worldwide are often labelled with these categories, so we can assume that this information is given. In the research presented in this chapter, we decided to focus on classical music, and therefore limit the set of investigated instruments to ones which are typical for this type of music. If someone would like to investigate a different music genre, the same methodology can be applied.

The audio data we decided to use in the experiments represent the following 10 instruments: B-flat clarinet, cello, double bass, flute, French horn, oboe, piano, tenor trombone, viola, and violin. Obviously, this set is not comprehensive and could be extended; still, it is sufficient for the purpose of illustrating the task we are dealing with, i.e., recognition of multiple instruments in polyphonic recordings.

Our experiments included training and testing of random forests. Therefore, we needed recordings for training RFs to be used to recognize selected instruments. We used single sounds played in various ways: *vibrato* (with vibration), *pizzicato* (plucking the strings), *forte* (loud), *piano* (soft), etc.; techniques of playing are called articulation. Also, we used all available pitches for every instrument.

The training data were taken from two commonly used repositories:

- MUMS [19]: all available articulation versions for our 10 instruments;
- IOWA [25]: *fortissimo* (very loud) for piano, and *mezzo forte* (medium loud) for other instruments;
    - cello, viola, and violin: *arco* (bowing) and *pizzicato*;
    - flute: *vibrato* and *non-vibrato* (no vibration);
    - French horn: *fortissimo* for notes within C3–B3 (MIDI notation used, i.e., A4=440 Hz) and *mezzo forte* for the remaining notes.

Some of the sounds were recorded *vibrato* (*e.g.*, strings – violin, viola, cello, and double bass from MUMS), and others with no vibration (strings in IOWA repository). Sounds of strings and tenor trombone were also chosen played muted and not muted. Flute is represented by vibrato and flutter sounds. Piano is represented by soft, plucked, and loud sounds. For each instrument, all articulation versions of sounds of this instrument represent the same class, i.e., the given instrument.

Testing data were taken from RWC Classical Music Database [5], so they were utterly different from the training data. Since we planned to evaluate temporal features, describing evolution of a sound in time (whether this would be a single sound, or a chord), we needed pieces with long sounds, i.e., long enough to observe time variability of these sounds in non-transitory parts. Such long-lasting sounds were manually selected from RWC Classical Music Database. We also wanted our test set to represent various composers and music styles. Therefore, the following pieces were used (number of test sounds selected for each piece is shown in parentheses):

- No. 4: P.I. Tchaikovsky, Symphony no. 6 in B minor, op. 74 ‘Pathétique’, 4th movement (10 sounds);
- No. 9: R. Wagner, “Tristan und Isolde": Prelude and ‘Liebestod’ (9 sounds);
- No. 12: J.S. Bach, “The Musical Offering", BWV. 1079, ‘Ricercare à 6’ (14 sounds);
- No. 16: W.A. Mozart, Clarinet Quintet in A major, K. 581, 1st movement (15 sounds);
- No. 18: J. Brahms, Horn Trio in E♭ major, op. 40, 2nd movement (4 sounds).

Test sounds represent homogenous chords (i.e., the instruments playing and the notes played remain constant throughout the whole sound), played by two to five instruments. These sounds were manually selected in a careful way and then labelled, thus creating ground-truth data for further experiments.

Both training and testing data were recorded with 44.1 kHz sampling rate and 16-bit resolution. If the audio data were recorded stereo, then the left channel was arbitrarily chosen for processing. Also, as a preprocessing step, the silence before and after each isolated sound was removed. To do this, a smoothed version of amplitude was calculated starting from the beginning of the file, as moving average of 5 consequent amplitude values, and when this value increased by more than a threshold (experimentally set to 0.0001), this point was considered to be the end of the initial silence. Similarly, the ending silence was removed.

18.2.1 Hornbostel-Sachs System of Musical Instrument Classification

Instruments we investigate in the reported research represent various families of instruments, according to Hornbostel-Sachs system of musical instrument classification [7], which is the most commonly used system describing the taxonomy of instruments. This system classifies instruments of classical music into the following groups: aerophones (wind instruments), chordophones (stringed instruments), membranophones (mostly drums), and idiophones (basically, other percussive instruments, where a solid is a source of vibration). Since Hornbostel-Sachs system provides a hierarchical classification of musical instruments, these categories are further subdivided into subcategories. According to Hornbostel-Sachs system, the investigated instruments are classified as follows:

- aerophones
  - flutes
    - ⋆ (transverse) flute,
  - reed instruments
    - ⋆ single reed: B-flat clarinet,
    - ⋆ double reed: oboe,

  - brass
    - ⋆ French horn,
    - ⋆ tenor trombone,

- chordophones
  - bowed: cello, double bass, viola, and violin; these instruments can be played *pizzicato* (and this articulation was also investigated), but bowing is a primary articulation here, this is why these instruments are classified as bowed;
  - piano.

We decided to investigate sounds of definite pitch, with harmonic spectra, as we planned to monitor harmonic structure of the spectrum, among other sound features. Therefore, percussive instruments (membranophones and idiophones) are not investigated here.

The timbre of a sound may also differ depending on articulation. However, our goal was to identify musical instruments without taking this property into account. Therefore, all sounds of each particular instrument represented the same class, i.e., this instrument, and no classification according to articulation was investigated in the reported research.

18.3 Feature Set

Our feature set consists of the main, basic set of features, calculated for a 40-ms Hamming-windowed frame of the analyzed sound, which is then used twofold: to calculate average values, constituting the main representation of this sound, and to observe temporal behavior of the analyzed sound. To start with, average values of the main features are calculated for a sliding analysis frame with 10 ms hop size. In order to make sure that long-term behavior is captured, 430 ms are taken for this calculation. This may not cover the entire sound, but it is sufficient to cover the onset and a good portion of the steady state, which are usually sufficient to recognize an instrument by human listeners, so we also follow this philosophy. Next, we calculate *Fits* – this proposed feature represents the type of the function which best describes the temporal behavior of the main feature set; consecutive (and overlapping) parts of the sound can be described by different functions. Finally, we calculate *Peaks*; this multidimensional feature describes relationships between three greatest temporal local maxima, representing time variability of the given feature throughout the entire sound. The obtained temporal features are then added to the feature set. The details of calculations of the above-mentioned features are described below.

The basic feature set consists of the following parameters:

- *SpectralCentroid* of the spectrum obtained through the discrete Fourier transform (DFT), calculated as Fast Fourier Transform (FFT). In this case, the frame length must equal to the power of 2. Since 40 ms equals to 1764 audio

samples in the case of 44.1 kHz sampling rate, this frame is zero-padded to 2048 samples, and next *SpectralCentroid* C_i is calculated as follows:

$$C_i = \frac{\sum_{k=1}^{N/2} f(k)\,|X_i(k)|}{\sum_{k=1}^{N/2} |X_i(k)|} \tag{18.1}$$

where: N - number of available elements of the (symmetrical) discrete spectrum, i.e., frame length, so $N = 2048$;
$X_i(k)$ - k^{th} element of FFT for i^{th} frame;
$f(k)$ - frequency corresponding to k^{th} element of the spectrum;

- *SpectralSpread* S_i - a deviation of the power spectrum with respect to Spectral Centroid C_i in a frame, calculated as

$$S_i = \sqrt{\frac{\sum_{k=1}^{N/2} (f(k) - C_i)^2\,|X_i(k)|}{\sum_{k=1}^{N/2} |X_i(k)|}} \tag{18.2}$$

- *AudioSpectrumFlatness*, $Flat_1, \ldots, Flat_{25}$ - multidimensional parameter describing the flatness property of the power spectrum within a frequency bin for selected bins; 25 out of 32 frequency bands were used for a given frame, starting from 250 Hz, as recommended in MPEG-7. This feature is calculated as follows:

$$Flat_b = \frac{\sqrt[hi(b)-lo(b)+1]{\prod_{k=lo(b)}^{hi(b)} P_g(k)}}{\frac{1}{hi(k)-lo(k)+1} \sum_{k=lo(b)}^{hi(b)} P_g(k)} \tag{18.3}$$

where: b - band number, $1 \leq b \leq 25$,
$lo(b)$ and $hi(b)$ - lower and upper limits of the band b, respectively,
$P_g(k)$ - grouped coefficients of the power spectrum within the band b; grouping speeds up the calculations;

- *RollOff* - the frequency below which an experimentally chosen percentage of the accumulated magnitudes of the spectrum is concentrated (equal to 85%, which is the most often used setting). *RollOff* is a measure of spectral shape, used in speech recognition to distinguish between voiced and unvoiced speech;
- $Flux$ - sum of squared differences between the magnitudes of the FFT points in a given frame and its preceding frame. This value is usually very small, and it was multiplied by 10^7 in our research. For the starting frame, $Flux = 0$ by definition;
- *Energy* - energy (in logarithmic scale) of the spectrum of the parameterized sound;
- *MFCC* - multidimensional feature, consisting of 13 Mel frequency cepstral coefficients. The cepstrum was calculated as a logarithm of the magnitude of the spectral coefficients and then transformed to the mel scale. Mel scale is used instead of the Hz scale, in order to better reflect properties of the human perception of frequency. Twenty-four mel filters were applied, and the obtained results

were transformed to twelve coefficients. The thirteenth coefficient is the 0-order coefficient of MFCC, corresponding to the logarithm of the energy [12], [18];

- *ZeroCrossingRate*; zero-crossing is a point where the sign of time-domain representation of the sound wave changes;
- *FundamentalFrequency* - pitch; maximum likelihood algorithm was applied for pitch estimation [26];
- *HarmonicSpectralCentroid*, *HSC* - mean of the harmonic peaks of the spectrum, weighted by the amplitude in linear scale [8];
- *HarmonicSpectralSpread*, *HSS* - represents the standard deviation of the harmonic peaks of the spectrum with respect to *HarmonicSpectralCentroid*, weighted by the amplitude [8];
- *HarmonicSpectralVariation*, *HSV* - normalized correlation between amplitudes of harmonic peaks of each two adjacent frames, calculated as:

$$HSV = 1 - \frac{\sum_{n=1}^{N} A_n(i-1) \cdot A_n(i)}{\sqrt{\sum_{n=1}^{N} A_n^2(i-1)} \cdot \sqrt{\sum_{n=1}^{N} A_n^2(i)}}$$

where $A_n(i)$ - amplitude of n^{th} harmonic partial in i^{th} frame [8]. For the starting frame, $HSV = 1$ by definition.

- *HarmonicSpectralDeviation*, *HSD*, calculated as:

$$HSD = \frac{\sum_{n=1}^{N} |\log(A_n) - \log(SE_n)|}{\sum_{n=1}^{N} \log(A_n)}$$

where SE_n - n^{th} component from a spectral envelope,
A_n - amplitude of n^{th} harmonic partial.
This feature represents the spectral deviation of the log amplitude components from a global spectral envelope, where the global spectral envelope of the n^{th} harmonic partial is calculated as the average value of the neighboring harmonic partials: no. $n-1$, n, and $n+1$, calculated as [8]:

$$SE_n = \frac{\sum_{i=-1}^{1} A_{n+i}}{3}$$

- $r_1, \ldots, r_{11}$ - various ratios of harmonic partials in spectrum: r_1 – energy of the fundamental to the total energy of all harmonics, r_2: amplitude difference [dB] between 1^{st} and 2^{nd} partial, r_3: ratio of the sum of partials 3-4 to all harmonics, r_4: partials 5-7 to all, r_5: partials 8-10 to all, r_6: remaining partials to all, r_7: brightness – gravity center of spectrum, r_8, r_9: contents of even/odd harmonics in the spectrum, respectively.

For these basic features, we calculated:

- *Averages* - vector representing averaged (through 430 ms) values for all features; this is our basic feature set;

- *Fits* - type of function (from 7 predefined function types) which best describes the manner of feature values' variation in time. Analysis was performed in 4 parts of the sound, each described by 10 consecutive 40 ms frames 75% overlapped (altogether 280 ms); each of these 4 parts can be assigned to any of these 7 function types. Hop size between parts was equal to 5 frames. Predefined function types were as follows: linear, quadratic, logarithmic, power, hyperbolic, exponential, and sinusoidal with linear trend. Original feature vector was treated as a function of time. Functions of each predefined type were fitted into each feature function within a given part of the sound. Linear and quadratic functions were fitted using the method of least squares. In other cases, linearization was performed before applying the least squares method. R^2 value was calculated for each fit, where R is a Pearson's correlation coefficient. A function with the highest R^2 value was supposed to fit the data best. If the highest R^2 was lower than 0.8, then it was assumed that none of proposed functions fits data well, and "no fit" was assigned as a feature value;
- *Peaks* (new temporal features) - distances and proportions between maximal peaks in temporal evolution of feature values throughout the entire sound, defined as follows. Let us name original feature vector as p and treat p as a function of time. We searched for 3 maximal peaks of this function. Maximum $M_i(p)$, $i = 1,2,3$, was described by k - the consecutive number of the frame where the extremum appeared, and the value of feature p in the frame k:

  $$M_i(p) = (k_i, p[k_i]) \quad k_1 < k_2 < k_3.$$

  The temporal variation of each feature can be then represented as a vector $T = [T_1, \ldots, T_6]$ of temporal parameters, built as follows:

  $$T_1 = k_2 - k_1\ ,\ T_2 = k_3 - k_2\ ,\ T_3 = k_3 - k_1\ ,$$
  $$T_4 = p[k_2]/p[k_1]\ ,\ T_5 = p[k_3]/p[k_2]\ ,\ T_6 = p[k_3]/p[k_1]\ .$$

  These parameters reflect relative positions and changes of values representing maximal peaks in the temporal evolution of each feature [11].

18.4 Experiments and Results

The purpose of this chapter was to investigate automatic identification of musical instruments in polyphonic recordings, and to verify if new temporal features can be helpful to better recognize instruments in recordings. Another aim was to check if hierarchical classifiers yield better results than non-hierarchical ones.

18.4.1 Training and Testing of Random Forests

Training of the battery of RFs was performed on single isolated sounds of musical instruments, taken from IOWA and MUMS repositories, and on sound mixes of up to three instruments. This way we created a set of multi-instrumental audio samples, in order to train RF to identify the target instrument, even when accompanied by another instrument or instruments. Instrumental sounds added in mixes were randomly chosen in such a way that the obtained sounds constitute unisons or chords (major or minor), and the distribution of instruments in the obtained set of mixes reflects the distribution of instruments playing together in RWC Classical Music Database. One-label training of binary RFs was performed on these data, aiming at identification of a target instrument, i.e., whether it is playing in a sound, or not.

Tests of the obtained battery of RFs were performed on RWC Classical Music data. Predictions were based on the results obtained for all forests (for all instruments). Polytimbral music samples should produce multiple labels. To obtain such multi-label predictions from our classification system, we derived them in a following way. For each binary classifier we got a percentage of votes of trees in the forest on “yes" class (presence of an instrument corresponding to a given classifier), and this percentage was treated as the rate of each corresponding label (instrument name). Labels were sorted in decreasing order with respect to the corresponding rates. If the first label on a list had the rate exceeding 80% and next label had the rate below 20%, then we assumed that this sound was recognized as monotimbral and prediction contained only one label – an instrument name of the highest rate. Otherwise, the differences of rates of consecutive labels in the list were calculated, and the prediction list of labels was truncated where the highest difference was found.

In the case of hierarchical classification, binary RFs were similarly trained to recognize groups of instruments in a given node.

In this case predictions were obtained in a similar way, but rates for labels in leaves of a tree were calculated by multiplying rates from all nodes in a path from the root to a given leaf.

In this work we used the RF implementation from the *R* package *randomForest* [13], [20].

18.4.2 Feature-Driven Hierarchic Classifications of Musical Instruments

In our experiments, we aimed at identifying instruments playing in a given snippet of an audio recording, using several strategies of classification. To start with, we performed non-hierarchical classification using a battery of binary RFs, where each RF was trained to indicate whether a target instrument was playing in the investigated audio snippet or not. These classification results are shown in Table 18.1, together

Table 18.1 Results of the recognition of musical instruments in RWC Classical Music Database, for the basic feature set

Classification system	Precision	Recall	F-measure
Non-hierarchical	71.63%	58.43%	64.36%
Hierarchical (Hornbostel-Sachs)	70.74%	60.06%	64.97%

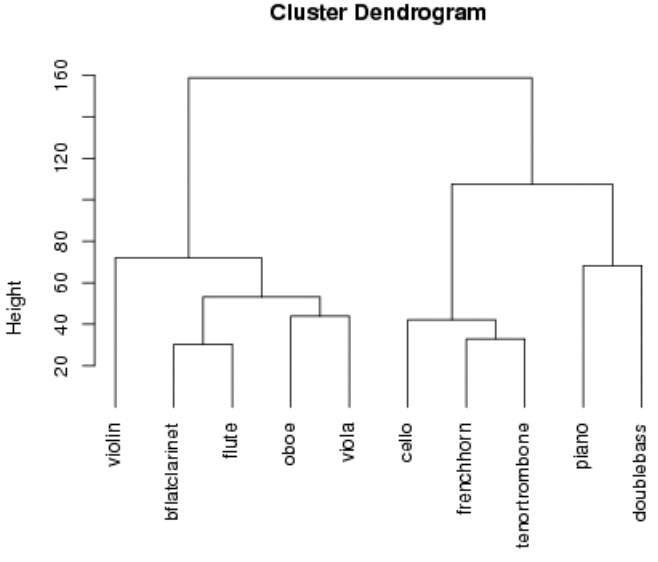


Fig. 18.2 Cluster dendrogram for *Averages*.

with the results obtained for hierarchical classification based on Hornbostel-Sachs taxonomy of musical instruments, for the basic feature set, i.e., *Averages*.

Apart from Hornbostel-Sachs hierarchical classification, feature-driven hierarchical classification of musical instruments in recordings was performed. Hierarchies were obtained through clustering.

Hierarchical clustering was performed by means of Ward's method, appropriate for quantitative variable as ours [24]. This method uses an analysis of variance approach to evaluate the distances between clusters. Ward's method attempts to minimize the sum of squares of any two hypothetical clusters that can be formed at each step. It finds compact, spherical clusters, although it tends to create clusters of small size. This method implements an agglomerative clustering algorithm, starting at the leaves, regarded as n clusters of size 1. It looks for groups of leaves, forms them into branches, and continues to the root of the resulting dendrogram. Distances between clusters were calculated using Manhattan distance, as it performed best in the conducted experiments.

Hierarchical clustering of instrument sounds was performed using *R*, an environment for statistical computing [20]. The clustering based on feature vectors rep-

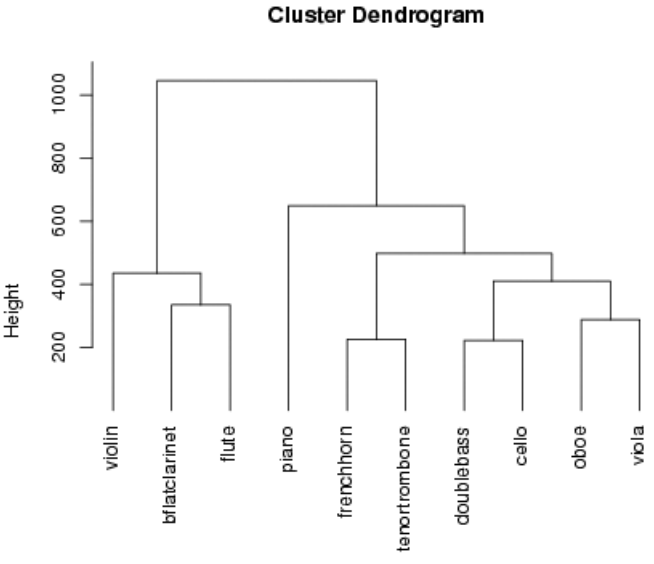


Fig. 18.3 Cluster dendrogram for *Averages* + *Peaks*.

resenting only average values of our basic features (*Averages*), and with addition of temporal observations of these features (*Fits* and *Peaks*) are shown in Figures 18.2, 18.4, and 18.3, respectively. Each dendrogram was built on the basis of single instrumental sounds only, without mixes, thus no foreign sounds distorted representation of each target instrument. Every instrument was represented by one artificial object, calculated as averaged value of all objects, i.e., parameterized sounds of this instrument.

As we can see, the taxonomies of musical instruments obtained through clustering shown in Figures 18.2, 18.4, and 18.3, differ significantly from classic Hornbostel-Sachs system, in all cases of the feature-driven hierarchical trees.

The results obtained for hierarchical classification in various settings of hierarchies are given in Table 18.2. As we can see, precision is almost constant, around 70-72%, so it is practically independent of the hierarchy. However, the obtained recall changes significantly. For each feature set, the recall improves when feature-driven hierarchy is used as a classification basis. The best overall results (reflected in F-measure) are obtained for feature-driven classification, and for *Fits* added to the feature set. The trade-off between precision and recall can be observed in some cases, but it is rather small. In general, adding temporal features improves the obtained results, comparing to the results obtained for *Averages*; adding *Peaks* improves accuracy, and adding *Fits* improves recall.

One can be interested in seeing the details of misclassification. Since we have multiple instruments labeling both the input and output data, a regular confusion matrix cannot be produced, since we cannot show which instrument was mistaken

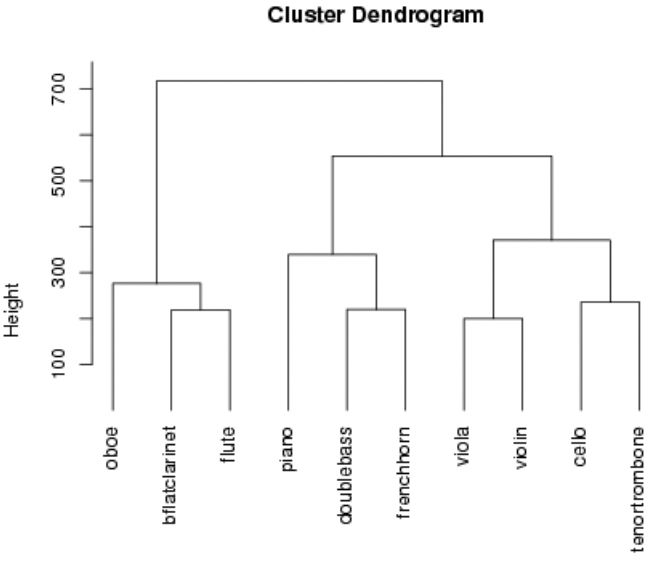


Fig. 18.4 Cluster dendrogram for *Averages + Fits*.

Table 18.2 Results of the recognition of musical instruments in RWC Classical Music Database for different feature sets and hierarchic classification systems

Instruments hierarchy	Feature set	Precision	Recall	F-measure
Hornbostel-Sachs	Averages	70.74%	60.06%	64.97%
Feature-driven	Averages	70.24%	65.74%	67.91%
Hornbostel-Sachs	Avg+Peaks	72.67%	60.42%	65.98%
Feature-driven	Avg+Peaks	72.35%	62.47%	67.04%
Hornbostel-Sachs	Avg+Fits	70.91%	64.74%	67.69%
Feature-driven	Avg+Fits	71.88%	70.67%	71.27%

for which one. Still, in order to illustrate the details of RF-based classification, exemplary results are presented in Figures 18.5 and 18.6, showing what types of classification errors we encountered.

Let us analyze the graphs presented in Figure 18.5. In the 1st graph, violin and cello were identified correctly, but double bass and viola were additionally indicated by the battery of RFs classifiers. Since double bass sound is similar to cello, and viola sound is similar to violin, it is not surprising that the corresponding RFs fired. In the case of the 2nd graph, the errors are more serious, since the violin and viola duo, although indicated correctly, was also accompanied by additional indication of cello and double bass. Even though cello and viola are relatively closely related instruments, the indication of double bass is considered to be a serious error here.

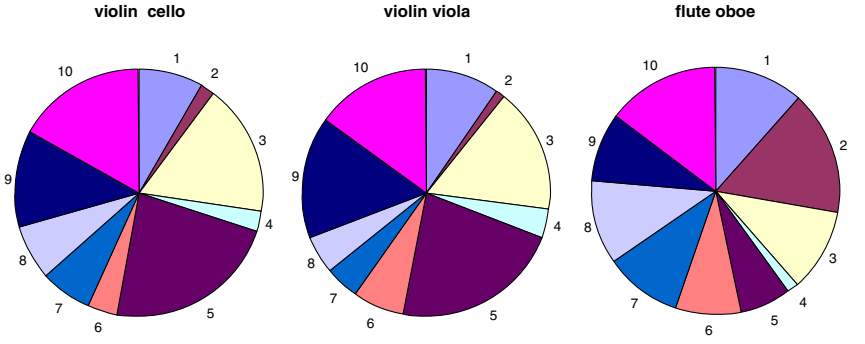


Fig. 18.5 Exemplary results of RF-based recognition of duo sounds. The numbers correspond to the instruments in the following order: 1. piano, 2. oboe, 3. cello, 4. trombone, 5. double bass, 6. French horn, 7. clarinet, 8. flute, 9. viola, 10. violin. The values shown represent outputs for each RF representing the given instrument

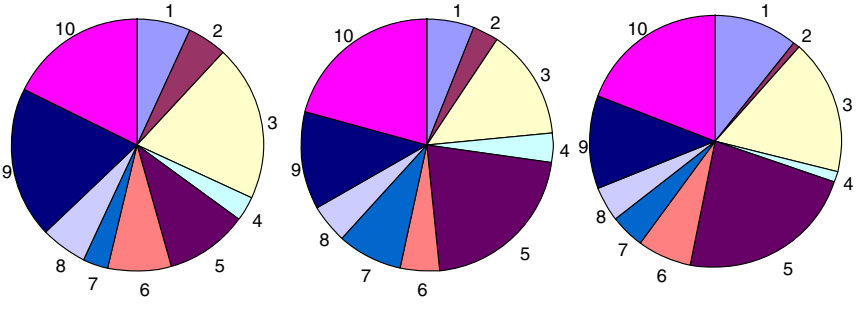


Fig. 18.6 Exemplary results of RF-based recognition of instruments in polyphonic recordings. Each input sound represented a chord played by violin, viola, and cello.

In the case of the 3rd diagram, oboe and flute were recognized correctly, but additionally violin (higher rate than flute), piano, cello, clarinet, viola, French horn and double bass were listed by our battery of RFs. This indicates that by adjusting the way of outputting the recognition list we may improve precision, but most probably at the expense of lower recall. Since recall is generally lower than precision in this research, we believe that cutting of more instruments listed by the RFs classifiers can deteriorate the overall results.

The graphs presented in Figure 18.6 show the results for three sounds, all representing violin, viola, and cello playing together. The 1st diagram shows correct identification of these three instruments, without errors. In the case of the other two diagrams, besides of recognizing the target instruments, our battery of RFs classifiers additionally indicated double bass. Again, double bass is similar to cello, so it is not considered to be a serious mistake.

18.5 Summary and Conclusions

In this chapter, we presented automatic hierarchical identification of musical instruments in recordings. The Sachs-Hornbostel classification is the most common hierarchic classification of musical instruments, but feature-driven classification yields better results in automatic recognition of instruments in recordings. The audio data are described here by means of various sound features, automatically calculated for short audio frames. These features are then used to calculate the main feature vector (*Averages*), as well as two additional feature types, *Peaks* and *Fits*, describing temporal changes of the basic features. Automatic recognition of instruments in polyphonic recordings was performed using Random Forests, for ten instruments commonly found in classical music pieces. Training of RFs classifiers was based on 2 repositories of instrumental sounds. Single sounds and sound mixes were used in this training; probability of adding an instrument to the training mix reflected the distribution of instruments playing together in classical music recordings, taken from RWC Classical Music Database.

Our experiments showed that hierarchical classification yields better results than non-hierarchical one. Feature-driven hierarchic classification always improves recall, which tends to be lower than precision (since identification of all instruments in a chord is difficult even for a human), so the increase of recall is valuable, and we consider it to be a success. Also, we observed that adding *Peaks* improves accuracy of instrument recognition, and adding the proposed feature *Fits* improves recall. We plan to continue experiments, with an extended feature vector, including both *Peaks* and *Fits* added to *Averages*. We also plan to add more detailed temporal features, and conduct experiments for more instruments.

Acknowledgments. This project was partially supported by the Research Center of PJIIT (supported by the Polish National Committee for Scientific Research (KBN)) and also by the National Science Foundation under Grant Number IIS-0968647. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

1. Breiman, L.: Random Forests. Machine Learning 45, 5–32 (2001)
2. Brown, J.C.: Computer identification of musical instruments using pattern recognition with cepstral coefficients as features. J. Acoust. Soc. Am. 105, 1933–1941 (1999)
3. Eggink, J., Brown, G.J.: Application of missing feature theory to the recognition of musical instruments in polyphonic audio. In: ISMIR 2003 (2003)
4. Foote, J.: An Overview of Audio Information Retrieval. Multimedia Systems 7, 2–11 (1999)
5. Goto, M., Hashiguchi, H., Nishimura, T., Oka, R.: RWC Music Database: Popular, Classical, and Jazz Music Databases. In: Proceedings of the 3rd International Conference on Music Information Retrieval, pp. 287–288 (2002)

6. Herrera-Boyer, P., Klapuri, A., Davy, M.: Automatic Classification of Pitched Musical Instrument Sounds. In: Klapuri, A., Davy, M. (eds.) Signal Processing Methods for Music Transcription. Springer Science & Business Media, LLC (2006)
7. Hornbostel, E.M., von Sachs, C.: Systematik der Musikinstrumente. Zeitschrift für Ethnologie 46, 553–590 (1914)
8. ISO MPEG-7 Overview, http://www.chiariglione.org/mpeg/
9. Klapuri, A., Davy, M. (eds.): Signal Processing Methods for Music Transcription. Springer, New York (2006)
10. Kostek, B.: Musical instrument classification and duet analysis employing music information retrieval techniques. Proc. IEEE 92(4), 712–729 (2004)
11. Kubera, E., Wieczorkowska, A., Raś, Z., Skrzypiec, M.: Recognition of Instrument Timbres in Real Polytimbral Audio Recordings. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010. LNCS (LNAI), vol. 6322, pp. 97–110. Springer, Heidelberg (2010)
12. Kubera, E.: The role of temporal attributes in identifying instruments in polytimbral music recordings (in Polish). Ph.D. Dissertation, Polish-Japanese Institute of Information Technology (2010)
13. Liaw, A., Wiener, M.: Classification and regression by random Forest. R News 2(3), 18–22 (2002)
14. Livshin, A.A., Rodet, X.: Musical Instrument Identification in Continuous Recordings. In: Proc. of the 7th Int. Conference on Digital Audio Effects (DAFX 2004), Naples, Italy (2004)
15. MIDOMI, http://www.midomi.com/
16. Mierswa, I., Morik, K., Wurst, M.: Collaborative Use of Features in a Distributed System for the Organization of Music Collections. In: Shen, J., Shephard, J., Cui, B., Liu, L. (eds.) Intelligent Music Information Systems: Tools and Methodologies, pp. 147–176. IGI Global (2008)
17. Müller, M.: Information retrieval for music and motion. Springer, Heidelberg (2007)
18. Niewiadomy, D., Pelikant, A.: Implementation of MFCC vector generation in classification context. J. Applied Computer Science 16, 55–65 (2008)
19. Opolko, F., Wapnick, J.: MUMS – McGill University Master Samples. CD's (1987)
20. R Development Core Team A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria, http://www.R-project.org
21. Raś, Z.W., Wieczorkowska, A.A. (eds.): Advances in Music Information Retrieval. SCI, vol. 274. Springer, Heidelberg (2010)
22. Shen, J., Shepherd, J., Cui, B., Liu, L. (eds.): Intelligent Music Information Systems: Tools and Methodologies. Information Science Reference, Hershey (2008)
23. Sony Ericsson TrackID, http://www.sonyericsson.com/trackid
24. The Pennsylvania State University Cluster Analysis - Ward's Method, http://www.stat.psu.edu/online/courses/stat505/18_cluster/09_cluster_wards.html
25. The University of IOWA Electronic Music Studios Musical Instrument Samples, http://theremin.music.uiowa.edu/MIS.html
26. Zhang, X., Marasek, K., Raś, Z.W.: Maximum Likelihood Study for Sound Pattern Separation and Recognition. In: 2007 International Conference on Multimedia and Ubiquitous Engineering, MUE 2007, pp. 807–812. IEEE (2007)

Chapter 19
Unifying Variable Precision and Classical Rough Sets: Granular Approach

Tsau Young Lin* and Yu Ru Syau*

Abstract. The primary goal of this paper is to show that neighborhood systems (NS) can integrate Ziarko's variable precision and Pawlak 's classical rough sets into one concept. NS was introduced by T.Y. Lin in 1989 to capture the concepts of "near" (in generalized topology) and "conflicts" (studied using non-reflexive and symmetric binary relation). Currently, NS's are widely used in granular computing.

Keywords: Rough sets, granular computing, variable precision rough set model (VPRS), neighborhood systems, binary neighborhood systems.

19.1 Introduction

In 1996, Professor Lotfi Zadeh explained the concept of Granular Mathematics (GrM) as follows:

> *If we view classical mathematics as the mathematics of points, then granular mathematics (GrM) is a new mathematics, in which points are replaced by granules.*

To limit the scope, T.Y. Lin proposed the label granular computing [37]. Since then, many significant theories and applications have appeared under this label (see, *e.g.*, [1–3, 12–18, 20–23, 28]). In particular, Lin used the neighborhood system (NS) to

Tsau Young Lin
Department of Computer Science, San Jose State University
San Jose, CA 95192-0249
e-mail: ty.lin@sjsu.edu

Yu Ru Syau
Department of Information Management, National Formosa University
Huwei 63201, Yunlin, Taiwan
e-mail: yrsyau@nfu.edu.tw

* Tsau Young Lin and Yu Ru Syau contributed equally to this work.

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 365–373.
springerlink.com

model Zadeh's granule, NS was called Local GrC Model (or First GrC model) in the Encyclopedia [21,22] and relationships with general topology [7,11,31] were studied. Moreover, T.Y. Lin in his keynote talk at GrC2011 proposed the granular framework based on the point free concept. The framework is related to studies of pointless topologies (see [9,31]).

Note that the concept of NS was used in late 80-ties for totally different considerations. In [13] are considered approximate retrieval and approximate reasoning in generalized topological spaces, in which a neighborhood is a granule of uncertainty (a list of indistinguishable objects). While in [14], the concept of conflicts is considered. A neighborhood is an enemy list which forms an elementary granule of knowledge in computer security. These investigations were stimulated by [5,6,8]. Let us also note that neighborhood systems (of clopen spaces) were used in rough sets since the beginning of rough sets in direct or indirect form (see Section 19.2) and recently are used more complex structures (see, *e.g.*, (see, *e.g.*, [34]).

The primary goal of this paper is to show that neighborhood system (NS) can integrate Ziarko's variable precision rough set model [41] and Pawlak's classical rough sets [24–27] into one concept.

19.2 Neighborhood Systems (NS)

Let us recall the neighborhood system definition.

Definition 19.1. A neighborhood system (NS) is a mapping

$$NS : U \longrightarrow 2^{P(U)}, \tag{19.1}$$

where $P(U)$ is the family of all crisp/fuzzy subsets of U and $2^{(P(U)}$ is the family of all crisp subsets of Y, where $Y = P(U)$.

In the discussed framework, the granule corresponding to a point p is defined by $NP(p)$, i.e., this granule is a family of fuzzy/crisp subsets of U (a mathematical object in the powerset of $2^{(}P(U))$).

Example 19.1. Consider the set $U = \{p_0, p_1, p_2, p_3\}$ to be the universe of discourse and let us assume

1. $NS(p_0) = \emptyset$, i.e., there are no neighborhoods at p_0.
2. $NS(p_1) = \{\emptyset\}$, i.e., there is a neighborhood at p_1 which is an empty set.
3. $NS(p_2) = \{\{p_1, p_2\}\}$, i.e., there is a neighborhood that is not an empty set at p_2.
4. $NS(p_3) = \{\{p_0, p_1\}, \{p_0, p_3\}\}$, i.e., there are more than one neighborhood at p_3.
5. $\{p_0, p_3\}$ is called a reflexive neighborhood of p_3.
6. $\{p_0, p_1\}$ is called an anti-reflexive or punctured neighborhood of p_3.

Remark 19.1.

1. The NS given in Definition 19.1 is actually a base of NS. Such a NS will become the largest one, denoted by (LNS), if we add the following axiom:
   • Super Set Condition (SSC): If $M \supseteq N(p)$ for some nonempty $N(p) \in NS(p)$, then $M \in NS(p)$. Observe that NS and LNS are topologically equivalent.
2. A topological neighborhood system (TNS) is a special LNS that requires three more axioms [21].
3. The approximation space of rough set theory with indiscernibility defined by equivalence relation is a clopen topological space whose NS consists of equivalence classes [23–26]. Its LNS is denoted by RNS.
4. A covering $\mathcal{C}$ is a special NS, in which the NS(p) consists of all the members of $\mathcal{C}$ that contain p. Its LNS is denoted by CNS. $\mathcal{C}$ is an open NS.

Let us observe that in rough set theory some neighborhood systems were used starting form the beginning [24–27]. For example, let us consider the language L of boolean combinations of descriptors, i.e., formulas of the form (a,v), where a is an attribute and v is its value. Any such formula α defines in the information system (U,A) a set $\|\alpha\|_A \subseteq U$ of all objects from U satisfying α. Then for a given x we define $NS(x) = \{\|\alpha\|_A \mid x \in \|\alpha\|_A \ \& \ \alpha \in L\}$. In the construction of the rule-based classifiers are often used neighborhood systems defined by a subset of L consisting conjunction of descriptors only. In this case, quite often instead of crisp membership of $x \in \|\alpha\|_A$ are used some conditions of expressing closeness (or partial inclusion) of granule defined by A-signature of x (i.e., $Inf_A(x) = \{(a,a(x)) \mid a \in A\}$) to the granule defined by α. This leads to neighborhood systems such that for a given object its neighborhoods not necessarily are including the object. Such neighborhoods are used in matching of decision rules by objects in inductive extension (generalization) of decision rule extensions.

For a neighborhood system NS over U and $X \subseteq U$ we define the NS-lower approximation $\underline{NS}X$ of X and the NS-upper approximation $\overline{NS}X$ of X by

$$\underline{NS}X = \{x \in U :| \ \exists Y \in NS(x)\ Y \subseteq X\}, \tag{19.2}$$

and

$$\overline{NS}X = \{x \in U \mid \ \forall Y \in NS(x)\ Y \cap X \neq \emptyset\}. \tag{19.3}$$

Remark 19.2.

1. Observe that (19.2) and (19.3) are topological concepts.
2. To obtain Pawlak's knowledge approximation, observe the followings:
   a. NS induces a Derived Partition (Equivalence Relation): $p \equiv q$ if and only if LNS(p)=LNS(q).
   b. Observe that the given π is the Derived Partition [17, 18].
   c. These equivalence classes are called center sets and the knowledge represented by each center set is called a central knowledge.

3. Theorem 19.1 implies that these topological concepts are approximations of central knowledge.

In a special case of $NS(U) = \{NS(p) \mid p \in U\}$, where $NS(p)$ is a singleton $\{B_p\}$ we replace NS by

$$B : U \longrightarrow 2^U. \tag{19.4}$$

where $B(p) = B_p$ for $p \in U$.

Such special neighborhood systems are called *binary neighborhood systems* (BNS). (U, B) is called *binary granular model* [21,22]. Each set $B(p)$ can be treated as an example of (elementary) granule. One can easily observe that BNS is equivalent to binary relation. If $B(U)$ is a partition, then B is an equivalence relation.

Let us note that binary neighborhoods were considered in rough set theory since the beginning. The basic binary neighborhood system is defined by $B(x) = [x]_A$, where $[x]_A$ is the indiscernibility relation defined by the set of attributes A [24–27]. This was the main research direction at the beginning of rough set theory. However, next rough set theory was extended to more general cases such as tolerance (or similarity) rough sets, where we deal with coverings of the universe of objects which are not necessarily partitions (see, *e.g.*, [32,33]). For tolerance approximation spaces are defined approximation spaces [32,33]. They consists together with the covering defined by a binary neighborhood system some rough inclusion measures. The simplest case is when the rough inclusion measure is identical with crisp set theoretical inclusion. However, one may consider more general cases. One of these cases is related to variable precision rough set model (VPRS) [41]. Some more general are related to function approximation or to cases investigated in rough mereology [29,30].

Approximation spaces defined by Professor Zdzisław Pawlak in 80-ties, are based on partitions and they lead to clo-open topologies. The further research on rough sets, *e.g.*, based on coverings instead of partitions resulted in generalized approximation spaces related to more general topologies on the universe of objects. The neighborhood systems proposed by T.Y. Lin create a general framework for studying such topologies and their relationships to concept approximation.

19.3 Variable Precision Rough Sets

Professor Skowron has communicated the following to us:

> *In [27,32,33] it was shown that the variable precision rough set model (VPRS) model can be defined by approximation spaces with binary neighboorhood systems together with some specific rough inclusion measure. In this section, we show that for any approximation space AS defined by the VPRS model one can define a new approximation space determined by a neighborhood system and a rough inclusion measure identical with crisp set theoretical inclusion in which the lower and upper approximations of X are the same as in the original approximation space AS. In this way, the approximations in VPRS model for a given concept X can be defined by neighborhood systems.*

However, one should note that this fact cannot be generalized for approximation spaces with arbitrary rough inclusion measures. Moreover, please note that when we move from partitions to coverings then one can consider many different approximations of concepts.

We thank Professor Skowron for his comments. In addition, we would like to observe that VPRS can be extended to the cases that have different degrees of rough inclusions for different equivalence classes; and in all these cases VPRS can all be modeled by NS-theory.

Let us recall the definition of concept approximation in VPRS model. Any VPRS model over U is defined by a partition $\pi = \{\pi_1, \pi_2, \cdots, \pi_m\}$ of U and a parameter β such that $0 \leq \beta < 0.5$.

Let X be a subset, we define that π_i is approximately included in X with error β, if

$$\pi_i \subseteq^{\beta} X \text{ iff } e(\pi_i, X) \leq \beta$$

where

$$e(\pi_i, X) = 1 - \frac{|\pi_i \cap X|}{|\pi_i|}.$$

$|\cdot|$ is the set cardinality and $e(\pi_i, X)$ is the inclusion error of π_i in X; the equation can be re-written as

$$|\pi_i - \pi_i \cap X| = e(\pi_i, X) \cdot |\pi_i|.$$

Definition 19.2. The upper and lower approximations are defined by (Ziarko, 1993):

$$\begin{aligned}
&\underline{R}^{\beta}(X) = \cup\{\pi_i \mid \pi_i \subseteq^{\beta} X\} \\
&= \cup\{\pi_i \mid e(\pi_i, X) \leq \beta\} \\
&= \cup\{\pi_i \mid |\pi_i - \pi_i \cap X| \leq \beta \cdot |\pi_i|\}.
\end{aligned}$$

$$\begin{aligned}
&\overline{R}^{\beta}(X) = \cup\{\pi_i \mid \pi_i \cap^{\beta} X \neq \emptyset\} \\
&= \cup\{\pi_i \mid e(\pi_i, X) < 1 - \beta\} \\
&= \cup\{\pi_i \mid |\pi_i \cap X| > \beta \cdot |\pi_i|\}.
\end{aligned}$$

Rough set theory with such approximations is called the *variable precision rough set theory* (VPRS) (Ziarko, 1993).

We we define define a NS for it.

Definition 19.3. The Neighborhood System for VPRS.
Let π and β be fixed. Let p be an arbitrary point in U, and $p \in \pi_i$. Let m_i be the maximal positive integer such that $m_i \leq \beta \cdot |\pi_i|$. Let

$$\begin{aligned}
&\mathcal{T}_i = \{T_i^j \subseteq \pi_i \mid |T_i^j| \leq m_i, j = 1, 2, \ldots\}. \\
&N^{\beta}_{T_i^j}(p) \equiv \pi_i - T_i^j. \\
&NS^{\beta}(p) = \{N^{\beta}_{T_i^j}(p) \mid \pi_i \in \pi, T_i^j \in \mathcal{T}_i\}.
\end{aligned}$$

$NS^{\beta}(U) = \{NS^{\beta}(p) \mid p \in U\}$ is referred to as the neighborhood systems of U under the given threshold β.

The idea of the neighborhood system construction for VPRS is illustrated in Figure 19.1.

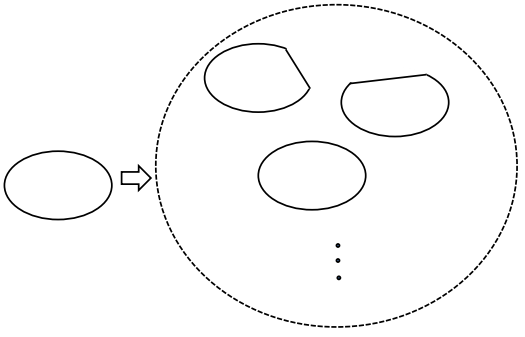


Fig. 19.1 The idea of neighborhood system constructing for VPRS: The idea of neighborhood system constructing for VPRS: each equivalence class is substituted by a family of neighborhoods obtained by cutting all possible 'small' parts from it.

One can prove that the approximations of concepts defined in VPRS model in the corresponding to it approximation space are identical with approximations defined by the defined above neighborhood system. More precisely, we have the following theorem:

Theorem 19.1. *For any VPRS model over U one can define a neighboorhood system NS over U such that the (lower) upper approximation of any subset of U in NS is equal to the (lower) upper approximation of this subset in the VPRS model.*

19.4 Conclusions

In this paper, we focuses on integrating variable precision rough sets with neighborhood systems. More works will be reported. Here we shall present some motivation of this approach.

1. In 1979, Zadeh observed that

   [...] the assumption that real numbers can be characterized and manipulated with infinite precision.

   situations, [...] cannot be dealt with through an appeal to continuity. In such cases, the information may be said to be granular [...]

2. We shall present a mathematical example:
   a. Based on the axioms of real numbers, the only solution of the inequalities $\{$ negative reals $\} < Z < \{$ positive reals $\}$ is $Z = 0$.
   b. If the axioms are expressed in First Order Logic (FOL), then we lost the infinite precision power and granules appear [10].
   c. The topological version of this granule is $TNS(p)$, for $p = 0$.
3. So the NS(p), is the model of granule. By abuse of language, we may also use granule to mean the neighborhood.

References

1. Bargiela, A., Pedrycz, W. (eds.): Granular Computing: An Introduction. Kluwer Academic Publishers (2003)
2. Baliga, P., Lin, T.Y.: Kolmogorov complexity based automata modeling for intrusion detection. In: GrC 2005, pp. 387–392. IEEE Computer Society Press (2005)
3. Chiang, I.J., Lin, T.Y., Liu, Y.: Table representations of granulations revisited. In: Ślęzak, D., Wang, G., Szczuka, M.S., Düntsch, I., Yao, Y. (eds.) RSFDGrC 2005, Part I. LNCS (LNAI), vol. 3641, pp. 728–737. Springer, Heidelberg (2005)
4. Dubois, D., Prade, H.: Putting rough sets and fuzzy sets together. In: Słowiński, R. (ed.) Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory, pp. 203–232. Kluwer Academic Publishers, Boston (1992)
5. Ginsburg, S., Hull, R.: Ordered attribute domains in the relational model. In: XP2 Workshop on Relational Database Theory, June 22-24, The Pennsylvania State University (1981)
6. Ginsburg, S., Hull, R.: Order dependency in the relational model. Theor. Comput. Sci. 26, 149–195 (1983)
7. Halmos, P.: Measure Theory. Van Nostrand (1950)
8. Hsiao, D., Harary, F.: A formal system for information retrieval from files. Communications of the ACM 13(2), 67–73 (1970); Corrigenda 13(4) (April 1970)
9. Johnstone, P.T.: The point of pointless topology. Bulletin (New Series) of the American Mathematical Society 8(1), 41–53 (1983)
10. Keisler, H.J.: Foundation of Infinitesimal Calculus. Prindle, Weber & Schmidt, Inc., Boston (1976)
11. Kelley, J.L.: General Topology. Van Nostrand, Princeton (1955)
12. Lin, T.Y.: Neighborhood systems and relational database. In: Proceedings of CSC 1988, p. 725 (1988)
13. Lin, T.Y.: Neighborhood systems and approximation in database and knowledge base systems. In: Proceedings of the Fourth International Symposium on Methodologies of Intelligent Systems, Poster Session, October 12-15, pp. 75–86 (1989a)
14. Lin, T.Y.: Chinese Wall security policy - an aggressive model. In: Proceedings of the Fifth Aerospace Computer Security Application Conference, December 4-8, pp. 286–293 (1989)
15. Lin, T.Y.: Topological and fuzzy rough sets. In: Słowiński, R. (ed.) Decision Support by Experience - Application of the Rough Sets Theory, pp. 287–304. Kluwer Academic Publishers, Amsterdam (1992)

16. Lin, T.Y., Liu, Q.: Rough approximate operators-axiomatic rough set theory. In: Ziarko, W. (ed.) Rough Sets, Fuzzy Sets and Knowledge Discovery, Workshop in Computing, pp. 256–260. Springer, Berlin & London (1994)
17. Lin, T.Y.: Neighborhood systems - A qualitative theory for fuzzy and rough sets. In: Wang, P. (ed.) Advances in Machine Intelligence and Soft Computing, vol. IV, pp. 132–155. Duke University, North Carolina (1997)
18. Lin, T.Y.: Granular computing on binary relations II: Rough set representations and belief function. In: Skowron, A., Polkowski, L. (eds.) Rough Sets and Knowledge Discovery, pp. 121–140. Physica-Verlag, Heidelberg (1998)
19. Lin, T.Y.: Granular computing on binary relations I: Data mining and neighborhood systems. In: Skowron, A., Polkowski, L. (eds.) Rough Sets and Knowledge Discovery, pp. 107–121. Physica-Verlag, Heidelberg (1998)
20. Lin, T.Y.: Granular Computing: Fuzzy logic and rough sets. In: Zadeh, L., Kacprzyk, J. (eds.) Computing with Words in Information/Intelligent Systems, pp. 183–200. Physica-Verlag, Heidelberg (1999)
21. Lin, T.Y.: Granular computing, Introduction to. Encyclopedia of Complexity and Systems Science, pp. 4313–4317. Springer, Berlin (2009)
22. Lin, T.Y.: Granular computing: Practices, theories, and future directions. In: Encyclopedia of Complexity and Systems Science, pp. 4339–4355. Springer, Berlin (2009)
23. Lin, T.Y.: Granular computing I: the concept of granulation and its formal model. International Journal of Granular Computing, Rough Sets and Intelligent Systems 1(1), 21–42 (2009)
24. Pawlak, Z.: Classification of Objects by Means of Attributes, Reports, vol. 429. Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland (1981)
25. Pawlak, Z.: Rough sets. International Journal of Computer and Information Sciences 11, 341–356 (1982)
26. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Dordrecht (1991)
27. Pawlak, Z., Skowron, A.: Rudiments of rough sets. Information Sciences 177(1), 3–27 (2007); Rough sets: Some extensions. Information Sciences 177(28-40), 1 (2007)
28. Pedrycz, W., Skowron, S., Kreinovich, V. (eds.): Handbook of Granular Computing. John Wiley & Sons, Hoboken (2008)
29. Polkowski, L., Skowron, A.: Rough mereology: A new paradigm for approximate reasoning. International Journal of Approximate Reasoning 15(4), 333–365 (1996)
30. Polkowski, L. (ed.): Approximate Reasoning by Parts. An Introduction to Rough Mereology. ISRL, vol. 20. Springer, Heidelberg (2011)
31. Sierpiński, W. (trans. by Krieger, C.): General Topology. Dover Publication Inc., Mineola (2000)
32. Skowron, A., Stepaniuk, J.: Generalized approximation spaces. In: The Third International Workshop on Rough Sets and Soft Computing Proceedings (RSSC 1994), San Jose, California, USA, November 10-12, pp. 156–163. San Jose University, San Jose (1994)
33. Skowron, A., Stepaniuk, J.: Tolerance approximation spaces. Fundamenta Informaticae 27(2-3), 245–253 (1996)
34. Skowron, A., Stepaniuk, J., Swiniarski, R.: Modeling rough granular computing based on approximation spaces. Information Sciences 184, 20–43 (2012)
35. Wong, E., Chiang, T.C.: Canonical structure in attribute based file organization. Communications of the ACM 14(9), 593–597 (1971)
36. Zadeh, L.A.: Toward extended fuzzy logic - A first step. Fuzzy Sets and Systems 160(21), 3175–3181 (2009)
37. Zadeh, L.A.: Some reflections on soft computing, granular computing and their roles in the conception, design and utilization of information intelligent systems. Soft Computing 2, 23–25 (1998)
38. Zadeh, L.A.: Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic. Fuzzy Sets and Systems 90, 111–127 (1997)

39. Zadeh, L.A.: The key roles of information granulation and fuzzy logic in human reasoning. In: 1996 IEEE International Conference on Fuzzy Systems, September 8-11, p. 1 (1996)
40. Zadeh, L.A.: Fuzzy sets and information granularity. In: Gupta, M., Ragade, R., Yager, R. (eds.) Advances in Fuzzy Set Theory and Applications, pp. 3–18. North-Holland, Amsterdam (1979)
41. Ziarko, W.: Variable precision rough set model. Journal of Computer and System Sciences 46, 39–59 (1993)

Chapter 20
Fuzzy Hybrid MCDM for Building Strategy Forces

Mei-Chen Lo and Gwo-Hshiung Tzeng

Abstract. The nature and degree of competition in an industry hinge on four basic forces: Production, Technology, Marketing and Research & Development (R&D). To establish a strategic agenda for dealing with these contending currents and to grow despite them, a company must understand how they work in its industry and how they affect the company in its particular situation. This study adopts Fuzzy MCDM methods and details how these forces operate and suggests ways of adjusting to them, and, where possible, of taking advantage of them. Knowledge of these underlying sources of competitive pressure provides the groundwork for a strategic agenda of action. The result highlights the critical strengths and weaknesses of the company, animate the positioning of the company in its industry, clarify the areas where strategic changes may yield the greatest payoff, and highlight the places where industry trends promise to hold the greatest significance as either opportunities or threats.

Keywords: Multiple Criteria Decision Making (MCDM), DEMATEL technique, Strategy Forces.

Mei-Chen Lo
Department of Business Management, National United University, No.1, Lienda Rd., Miaoli 36003, Taiwan *and* Institute of Project Management, Kainan University, No. 1, Kainan Road, Luchu, Taoyuan County 338, Taiwan
e-mail: meichen_lo@yahoo.com

Gwo-Hshiung Tzeng
Institute of Project Management, Kainan University, No. 1, Kainan Road, Luchu, Taoyuan County 338, Taiwan
e-mail: ghtzeng@mail.knu.edu.tw

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 375–392.
springerlink.com

20.1 Introduction

Every industry has an underlying structure, or a set of fundamental economic and technical characteristics, that gives rise to these competitive forces. Decision maker, wanting to position his company to cope best with its industry environment or to influence that environment in the company's favor, must learn what makes the environment tick. The essence of strategy formulation is coping with competition. Yet it is easy to view competition too narrowly and too pessimistically. Awareness of these forces can help a company stake out a position in its industry that is less vulnerable to attack.

The nature and degree of competition in an industry hinge on four forces: Production, Technology, Marketing and R&D [4]. To establish a strategic agenda for dealing with these contending currents and to grow despite them, a company must understand how they work in its industry and how they affect the company in its particular situation. This study details how these forces operate and suggests ways of adjusting to them, and, where possible, of taking advantage of them. Knowledge of these underlying sources of competitive pressure provides the groundwork for a strategic agenda of action [9]. They highlight the critical strengths and weaknesses of the company, animate the positioning of the company in its industry, clarify the areas where strategic changes may yield the greatest payoff, and highlight the places where industry trends promise to hold the greatest significance as either opportunities or threats. Understanding these sources also proves to be of help in considering areas for diversification.

The remainder of this paper is organized as follows. In Section 20.2, about strategy forces for business development is reviewed and the framework of dynamic strategy forces for efficiency business is built. In Section 20.3, the methodology of fuzzy hybrid MCDM model based on DEMATEL is proposed for applying strategy forces. An empirical case and result are illustrated in Section 20.4. Discussions and implications are presented in Section 20.5. Conclusion is offered in Section 20.6 and Future study in Section 20.7.

20.2 About Strategy Forces (SF)

The essence of strategy formulation is coping with competition. For a company in the fight for market share, competition is not manifested only in the other players. Rather, competitive strategies in an industry are rooted in its underlying economics, and competitive forces exist not only from internal considerations but also that go well beyond the established combatants in the high-tech industry. The forces sources may gestate from customers, suppliers, potential entrants, and substitute products which are all competitors that may be more or less prominent or active depending on a particular industry.

The nature and degree of competition in a company within the high-tech industry hinge on strategy forces [4, 18], such as Production Capability (PC), Technology Competence (TC), R&D (RD), Marketing & Service (MS).

1. Production Capability (PC): Manufacturing flexibility and production control.
2. Technology Competence (TC): Applications compatibility, core technology, system integration.
3. R&D (RD): Innovative projects, science research, technology milestone.
4. Marketing & Service (MS): Customer satisfaction, channels, promotion, one-stop-shopping.

Strategy-Forces based on answers to four generic questions about the strategy to be pursued by the organization which also assume the financial support and organizational deployment is available for high-tech competition game. Therefore, as Fig. 20.1 these four questions, one about production, one about marketing, one about Technology, and one about R&D evolved quickly into a standard set of "perspectives". The weaker the forces collectively, however, the greater the opportunity for superior performance. Whatever their collective strength, the corporate strategist's goal is to find a position between company and industry where can best defend itself against these forces or can influence them in its favor [9].

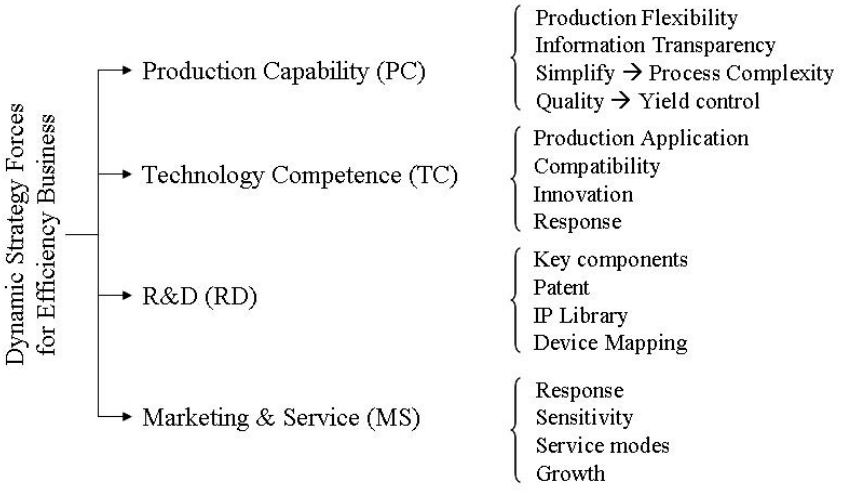


Fig. 20.1 The Hierarchy Structure

Understanding these sources also proves to be of help in considering areas for diversification. A strategy map is a diagram that is used to document the primary strategic goals being pursued by an organization or management team and to help managers focus their attention more closely on the interventions necessary to ensure the strategy is effectively and efficiently executed. By providing managers with the direct feedback on whether the required actions are being carried out, and whether

they are working. A performance management system is deciding what activities and outcomes to monitor. By providing a simple visual representation of the strategic objectives to be focused on, along with additional visual cues in the form of the perspectives and causal arrows, the strategy map has been found useful in enabling discussion within a management team about what objectives to choose, and subsequently to support discussion of the actual performance achieved.

20.3 Measuring the Forces Track

This section is based on what factors the past scholars in the study of strategy forces considerations indicated and to further discuss the result of the study; hence, identify the flow of facts. This study uses the four main aspects indicated by four-factor model as PC, TC, RD and MS factors to implement the strategies can be evaluated in dynamical considerations in time; identify the sub-factors that would affect the main factors, thus firmly develop evaluating criteria and its dynamic track of mapping.

20.3.1 Methodologies

The field of multiple criteria decision making (MCDM) concerns the problems that how decision makers should ideally do when facing multiple conflicting criteria. Group decision-making solve problems through a group of experts form their specific knowledge domain and help on business strategies decision-making [2, 19]. This study adopt a Decision-Making Trial and Evaluation Laboratory (DEMATEL) method which was developed by the Battelle Geneva Institute to analyze complex 'world problem' dealing mainly with interactive man-model techniques, and to evaluate qualitative and factor-linked aspects of societal problems. DEMATEL analyses the complicated problems in the real world via building a network interrelation. Therefore, the considerable methods and models have been proposed for various MCDM problems (as Fig. 20.2) with respect to different perspectives and theories.

A fuzzy hybrid MCDM model is proposed using expert groups. According to Fig. 20.2, the hierarchy structure is build by literature and experts' input. Each of methodology that applied on this study can be independently discussed its unique function as well as its characteristics. Although, the hybrid concept in methodologies application on problem solving is widely accepted while handle multiple stages on decision making process, combined those of methodologies can bring the case study serious and deep to look inside the problem itself by solving techniques in fitting the real life. The DEMATEL is used to detect complex relationships and to build a network relation map (NRM), including the calculation of dimension and criteria for direct/indirect influence forces and ranking. Then, the DANP (DEMATEL-based ANP approach) can be used to calculate the influential weights (W_D, W_C as of global weights and local weights) of criteria to overcome problems of

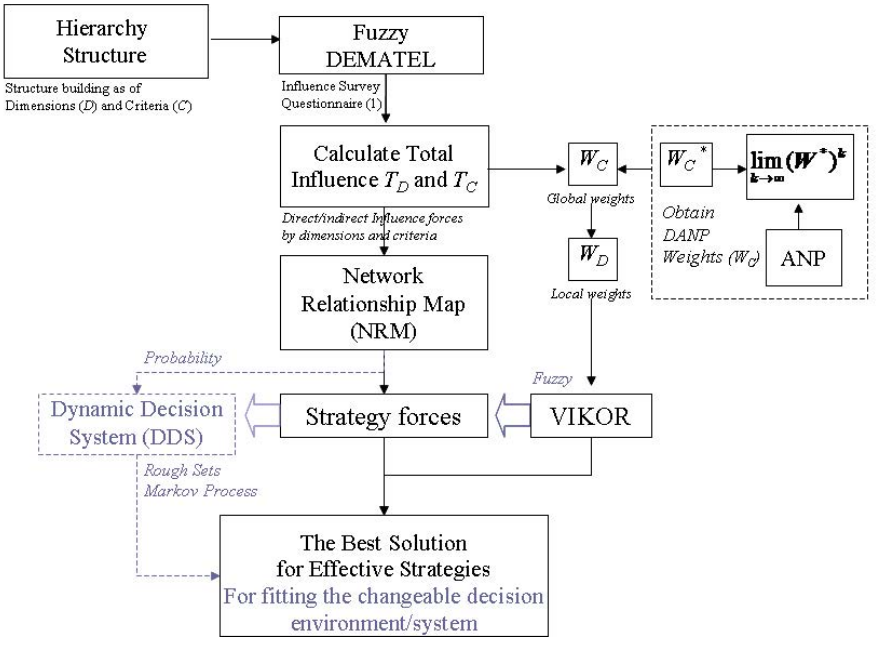


Fig. 20.2 Framework of Hybrid Methodologies Application

dependence and feedback among criteria, which we adopt the concept of ANP (analytic network of process) theory by Saaty [14]. Finally, VIKOR is used to evaluate the total performance to discover the performance scores and gaps. This study attempts to bridge this gap, using an empirical case of an improvement plan, and hopefully contributes to a complex strategic system with a useful evaluation model based on a hybrid MCDM method. Some future related work also is presented with grey print.

20.3.2 Fuzzy Theory with AHP and ANP

Because of the subjective of the attributes used in the evaluation, Analytic Hierarchy Process (AHP) [11–14, 16] is the most appropriate method for this problem. However, since the experts prefer natural language expressions rather than sharp numerical values in their assessments, the classical AHP may not yield satisfactory results. Expressions such as “not very clear”, “probably so”, and “very likely”, are used often in daily life, and more or less represent some degree of uncertainty of human thought. The fuzzy set theory proposed by Zadeh (1965) [20], an important concept applied in the scientific environment, has been available to other fields as well. Consequently, the fuzzy theory has become a useful tool for automating

human activities with uncertainty-based information. This study proposes a fuzzy hierarchy framework to implement factors analysis. This study helps organizations identify the differences between the desired and current situations, and then identify the improvement areas and develop the strategies for the business operation and implementation.

In addition, the company can also use the results as a benchmark with competitors and other "best-in-class" organizations. To manage strategy deployment successfully, its measurement indicators must be defined and prioritized. Because of the subjectivity of the attributes used in the evaluation, ANP [15] considers the relationship of mutually interactive and self-feedback as the most appropriate method for this problem. The ANP is an extension of AHP by Saaty to overcome the problem of interdependence and feedback between criteria or alternatives. Although the AHP and the ANP derive ratio scale priorities by making paired comparisons of elements of a criterion, there are differences between them. The first is that the AHP is a special version of the ANP; the ANP handles dependence within a cluster (inner dependence) and among different clusters (outer dependence). Secondly, the ANP is a nonlinear structure, while the AHP is hierarchical and linear, with the goal at the top and the alternatives in the lower levels [14]. The initial step of the ANP is to compare the criteria in the entire system to form a supermatrix through pairwise comparisons by asking "How much importance does one criterion have compared to another criterion, with respect to our interests or preferences?" The relative importance is determined using a scale of 1–5 representing equal importance to extreme importance.

The ANP is used to transform qualitative judgments into quantitative values and obtain the weights of perspectives. ANP was published by Saaty in 1999 [15]; the purpose is to solve the relaying and feedback problems of criteria. ANP overcomes one of the AHP limitations; in other words, ANP generalizes AHP. The biggest difference between AHP and ANP is that ANP has decision problem when applied to cases and criteria. However, AHP neglects the problem and presumes its independent relationship. Thus, when feedback occurs to exclude the cases and criteria, decision making might be affected. ANP will yield a more practical result. Moreover, the ANP is valuable for MCDM involving intangible attributes that are associated with strategies factors.

The ANP is a coupling of two parts. The first consists of a control hierarchy or network of criteria and sub-criteria that control the interactions. The second is a network of influences among the elements and clusters. The network varies from criterion to criterion and a different supermatrix of limiting influence is computed for each control criterion. Finally, each of these supermatrices is weighted by the priority of its control criterion and the results are synthesized through the addition for all the control criteria [16]. The ANP method [14] provides a general framework to deal with decisions without making assumptions about the independence of higher-level elements from lower-level elements and about the independence of the elements within a level as in a hierarchy. Compared with traditional MCDM methods, e.g., AHP, TOPSIS, ELECTRE, etc., which usually assume the independence between criteria, ANP, a theory that extends AHP to deal with dependence in

feedback and utilizes the supermatrix approach [10], is a more reasonable tool for dealing with complex MCDM problems in the real world. In this section, concepts of the ANP are summarized based on Saaty's earlier works [14,16].

20.3.3 Fuzzy Decision-Making

When people encounter uncertain or vague decision-making problems in the real world, they often express their thinking and subjective perception in words instead of probability and statistics. But the problem with words is that their meanings are often vague. Furthermore, even when people use the same words, individual judgment of events is invariably subjective and may differ. Moreover, even if the meaning of a word is well defined (e.g., the linguistic comparison labels in the standard AHP questionnaire responses), when we use the word to define a set, the boundary that separates whether an object does or does not belong to the set is often fuzzy or vague. This is why fuzzy numbers and fuzzy sets have been introduced to characterize linguistic variables. The preferences in AHP are essentially human judgments based on one's perception (this is especially true for intangibles), and we believe the fuzzy approach allows for a more accurate description of the decision-making process.

The Fuzzy ANP (FANP) approach applies a network, which connects the components of a decision system presented to prioritize strategy implementation factors for a manufacturing business to implement one of its most critical parts, used in management decision process and under multiple criteria decision-making in the fuzzy environment [16]. The main criteria and attributes have been decided which are based on the current business scenario and experience of the experts in the respective fields. The large number of criteria and attributes demonstrated the complexities involved in the strategy implementation factors. Each affecting factor [10], as strategy forces and implementation, has been analyzed and discussed. This paper has made a significant contribution to identifying the important criteria and paves the way to consider these practically relevant and interesting issues of strategy forces factors in the fuzzy environment.

We emphasize that the study contributes to build up and apply the theoretical model of strategy implementation priority. This study takes a step in the direction of setting priorities of clarifying, guiding, and integrating terms and concepts relevant to strategy implementation and management aspects in the firm process.

Fuzzy sets or fuzzy numbers is a way to cope with uncertain judgments and to incorporate the vagueness that typifies human thinking in expressing preferences. Fuzzy idea can be used in a wide range including possible related factors, which cover or overlap the concerns from one to another. Therefore, the defuzzification skill needs to be clarified for making sure that the final result can lead and help precisely in the right direction for decision-making.

For future studies, this study combines with fuzzy concept and uses the rough sets theory (first introduced by Pawlak in 1982 [8], is a valuable mathematical tool) for

dealing with vagueness and uncertainty. The use of the rough set approach (RSA) as a data mining techniques in general has been restricted to classification problems where the data is either non-ordered or the ordinal nature of the preference data is ignored. Therefore, that the data mining technique called Dominance-based Rough Set Approach (DRSA) analyzes a survey on strategy forces for dynamic decision space (DDS) is suggested. A set of "if ***antecedent***, then ***consequent***" decision rules are induced from the preference data that express the relationships between attribute values and the overall performance ratings. There are several advantages in using DRSA. First, the strategy decision rules are formulated in natural language and are easy to understand. Second, strategy decision associated with changeable attributes could be eliminated without affecting the company's overall performance rating. Third, decision rules that combine both technology development and innovation attributes could be used for mass customization.

20.3.4 Mapping Tools

A network connects the components of a decision system. According to size, there will be a system that is made up of subsystems, with each subsystem made up of components, and each component made up of elements. The elements in each component interact or have an influence on some or all of the elements of another component with respect to a property governing the interactions of the entire system.

20.3.4.1 DANP

DANP is the combined method of DEMATEL and ANP. This method is effective is solving the complicated problems between communities by the hierarchical structure, and to understand the complicated causalities. ANP relaxes the restriction of hierarchical structure [14] from AHP and is applied to solve interdependence problems and complicated network relations. We use the basic concept of ANP combining the total influential matrix of DEMATEL to build super-matrix of ANP for finding the influential weights among criteria.

The applicability of the method is widespread, ranging from industrial planning and decision-making to urban planning and design, regional environmental assessment, analysis of world problems, and so forth. It has also been successfully applied to many situations, such as marketing strategies, control systems, safety problems, developing the competencies of global managers, and group decision-making. Furthermore, a hybrid model combining the two methods has been widely used in various fields, for example, e-learning evaluation, airline safety measurement, and handset design for next-generation handset. Therefore, in this paper, we use DEMATEL not only to detect complex relationships and build an NRM of the criteria, but also to obtain the influence levels of each element over others; we then

adopt these influence level values as the basis of the normalization supermatrix for determining ANP weights to obtain the relative importance.

The DEMATEL method is based upon graph theory, enabling us to plan and solve problems visually, so that the relationship between the causes and effects of criteria is converted into an intelligible structural model of the system, to better understand causal–effect relationships. Directed graphs (also called digraphs) are more useful than directionless graphs, because digraphs will demonstrate the directed relationships of sub-systems. A digraph typically represents a communication network, or a domination relationship between individuals, etc. [3,17].

Suppose a system contains a set of elements, $S = \{s_1, s_2, \ldots, s_n\}$, and particular pair-wise relationships are determined for modeling, with respect to a mathematical relationship. Next, portray the relationship as a direct-relation matrix that is indexed equally in both dimensions by elements from the set S. Then, extract the case for which the number 0 appears in the cell (i, j), if the entry is a positive integer it has the meaning of:

1. the ordered pair (s_i, s_j) is in the mathematical relationship;
2. it has the kind of relationship where element s_i causes element s_j.

The digraph portrays a contextual relationship between the elements of the system, in which a numeral represents the strength of influence. The elements s_1, s_2, s_3, and s_4 represent the factors that have relationships in Fig. 20.3. The number between factors is influence or influenced degree. For example, an arrow from s_1 to s_2 represents the fact that s_1 influences s_2 and its influenced degree is two. The DEMATEL method can convert the relationship between the causes and effects of criteria into an intelligible structural model of the system.

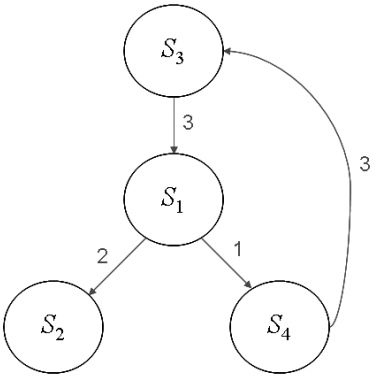


Fig. 20.3 An example of the directed influence graph

The influences of a given set of elements (determinants) in a component on any element in the decision system are represented by a ratio scale priority vector derived from paired comparisons of the comparative importance of one criterion and

another criterion with respect to the interests or preferences of the decision makers. This relative importance value can be determined using a scale of 1–9 to represent equal importance to extreme importance.

20.3.4.2 VIKOR Method

The VIKOR (Vlsekriterijumska Optimizacija I Kompromisno Resenje in Serbian, means Multicriteria Optimization and Compromise Solution) method introduced the multicriteria ranking index based on the particular measure of *closeness to the ideal/aspired level* solution and was introduced as one applicable technique to implement within MCDM [6,7]. The compromise solution is a feasible solution closest to the ideal/aspired level, and a compromise means an agreement established by mutual concessions. The VIKOR method was developed as a multicriteria decision-making method to solve discrete decision problems with non-commensurable and conflicting criteria. This method focuses on ranking and selecting from a set of alternatives in the presence of conflicting criteria, which could help the decision makers to reach a final decision. These methods rank and select alternatives based on all established criteria, using the same criteria for each alternative. However, in practice the decision maker often simultaneously manages or improves the achieved rate of progress in one or several projects (plans); therefore we needs to know the unimproved gaps of the projects or aspects of a project ('projects or aspects of a project' is abbreviated to 'Alternatives') so as to improve them to achieve the minimum/zero gaps.

The compromise-ranking method (VIKOR method) determines the compromise solution [5–7]; the obtained compromise solution is acceptable to the decision makers because it provides a maximum group utility of the majority (represented by min S), and a minimum individual maximal regret of the opponent (represented by min Q).

The VIKOR method began with the form of $L_p - metric$, which was used as an aggregating function in a compromise programming method and developed into the multicriteria measure for compromise ranking.

We assume the alternatives are denoted as $A_1, A_2, ..., A_i, ..., A_m$. w_j is the weight of the jth criterion, expressing the relative importance of the criteria, where $j = 1, 2, ..., n$, and n is the number of criteria. The rating (performance score) of the jth criterion is denoted by f_{ij} for alternative A_i. The form of $L_p - metric$ was introduced by Duckstein and Opricovic [1] and is formulated as follows:

$$L_i^p = \left(\sum_{j=1}^{n} [w_j \frac{(|f_j^* - f_{ij}|)}{(|f_j^* - f_j^-|)}]^p \right)^{\frac{1}{p}}, \tag{20.1}$$

$$1 \leq p \leq \infty; i = 1, 2, ..., m.$$

The VIKOR method is not only generated with the above form of $L_p - metric$, but it also uses $L_i^{p=1}$ (as S_i in Eq. 20.2) and $L_i^{p=\infty}$(as Q_i in Eq. 20.3) to formulate the ranking measure.[16,18,19,23]

$$S_i = L_i^{p=1} = \sum_{j=1}^{n} [w_j \frac{(|f_j^* - f_{ij}|)}{(|f_j^* - f_j^-|)}] \tag{20.2}$$

$$Q_i = L_i^{p=\infty} = \max_j \{w_j \frac{(|f_j^* - f_{ij}|)}{(|f_j^* - f_j^-|)}]; j = 1,2,...,n. \tag{20.3}$$

When p is small, the group utility is emphasized (such as $p = 1$) and as p increases, the individual regrets/gaps receive more weight. In addition, the compromise solution $\min_i L_i^p$ will be chosen because its value is closest to the ideal/aspired level. Therefore, in $\min_i S_i$ and $\min_i Q_i$, $\min_i S_i$ expresses the minimum of the sum of the individual regrets/gaps and $\min_i Q_i$ expresses the minimum of the maximum from individual regret. In other words, $\min_i S_i$ emphasizes the maximum group utility, whereas $\min_i Q_i$ emphasizes selecting minimum among the maximum individual regrets. Based on the above concepts, the compromise ranking algorithm VIKOR consists of the following steps.

Step 1. Determine the best f_j^*, and the worst f_j^- values of all criterion functions, $j = 1,2,...,n$. If we assume the jth function represents a benefit, then $f_j^* = \max_i f_{ij}$ (or setting an aspired level) and $f_j^- = \min_i f_{ij}$(or setting a tolerable level). Alternatively, if we assume the jth function represents a cost/risk, then $f_j^* = \min_i f_{ij}$ (or setting an aspired level) and $f_j^- = \max_i f_{ij}$ (or setting a tolerable level). Moreover, we propose an original rating matrix and a normalized weight-rating matrix of risk as follows:

$$\begin{array}{c} \\ alternatives \end{array} \begin{array}{c} \begin{array}{cccccc} & & criteria & & & \\ & c_1 & \cdots & c_j & \cdots & c_n \end{array} \\ \begin{array}{c} A_1 \\ \vdots \\ A_i \\ \vdots \\ A_m \end{array} \begin{bmatrix} f_{11} & \cdots & f_{1j} & \cdots & f_{1n} \\ \vdots & & \vdots & & \vdots \\ f_{i1} & \cdots & f_{ij} & \cdots & f_{in} \\ \vdots & & \vdots & & \vdots \\ f_{m1} & \cdots & f_{mj} & \cdots & f_{mn} \end{bmatrix} \\ \begin{array}{cccccc} f_1^* & \cdots & f_j^* & \cdots & f_n^* \\ f_1^- & \cdots & f_j^- & \cdots & f_n^- \end{array} \end{array} \quad \begin{array}{c} normalized \\ \Rightarrow \\ \times w_j \end{array} \quad \begin{array}{c} \begin{array}{cccccc} & & criteria & & & \\ & c_1 & \cdots & c_j & \cdots & c_n \end{array} \\ \begin{array}{c} A_1 \\ \vdots \\ A_i \\ \vdots \\ A_m \end{array} \begin{bmatrix} w_1 r_{11} & \cdots & w_j r_{1j} & \cdots & w_n r_{1n} \\ \vdots & & \vdots & & \vdots \\ w_1 r_{i1} & \cdots & w_j r_{ij} & \cdots & w_n r_{in} \\ \vdots & & \vdots & & \vdots \\ w_1 r_{m1} & \cdots & w_j r_{mj} & \cdots & w_n r_{mn} \end{bmatrix} \end{array}$$

(Original data) (Normalized data)

where, $r_{ij} = \frac{(|f_j^* - f_{ij}|)}{(|f_j^* - f_j^-|)}$, f_j^* is the aspired/desired level and f_j^- is tolerable level for each criterion.

Step 2. Compute the values S_i and Q_i, $i = 1, 2, \cdots, m$, using the relations

$$S_i = \sum_{j=1}^{n} w_j r_{ij}, \tag{20.4}$$

$$Q_i = \max_j \{w_j r_{ij} | j = 1, 2, ..., n\}, \tag{20.5}$$

Step 3. Compute the index values R_i, $i = 1, 2, \cdots, m$, using the relation

$$R_i = v\frac{(S_i - S^*)}{(S^- - S^*)} + (1 - v)\frac{(Q_i - Q^*)}{(Q^- - Q^*)} \tag{20.6}$$

where $S^* = \min_i S_i$ (or setting the best $S^* = 0$), $S^- = \max_i S_i$ (or setting the worst $S^- = 1$), $Q^* = \min_i Q_i$ (or setting the best $Q^* = 0$), $Q^- = \max_i Q_i$ (or setting the worst $Q^- = 1$), and $0 \le v \le 1$, where v is introduced as a weight for the strategy of maximum group utility, whereas $1 - v$ is the weight of the individual regret. In other words, when $v > 0.5$, this represents a decision-making process that could use the strategy of maximum group utility (i.e., if v is big, group utility is emphasized), or by consensus when $v \approx 0.5$, or with veto when $v < 0.5$.

Step 4. Rank the alternatives, sorting by the value of $\{S_i, Q_i$ and $R_i | i = 1, 2, ..., m\}$, in decreasing order. Propose as a compromise the alternative $(A^{(1)})$ which is ranked first by the measure $\min\{R_i | i = 1, 2, ..., m\}$ if the following two conditions are satisfied:

C1. *Acceptable advantage*: $R(A^{(2)}) - R(A^{(1)}) \ge \frac{1}{m-1}$, where $A^{(2)}$ is the alternative with second position in the ranking list by R; m is the number of alternatives.
C2. *Acceptable stability in decision-making*: Alternative $A^{(1)}$ must also be the best ranked by $\{S_i$ or/and $Q_i | i = 1, 2, ..., m\}$.

If one of the conditions is not satisfied, then a set of compromise solutions is proposed, which consists of:

1. Alternatives $A^{(1)}$ and $A^{(2)}$ if only condition C2 is not satisfied.
2. Alternatives $A^{(1)}, A^{(2)}, ..., A^{(M)}$ if condition C1 is not satisfied. $A^{(M)}$ is determined by the relation $R(A^{(M)}) - R(A^{(1)}) < \frac{1}{m-1}$ for maximum M (the positions of these alternatives are close.

The compromise solution is determined by the compromise-ranking method; the obtained compromise solution could be accepted by the decision makers because it provides maximum group utility of the majority (represented by min S, Eq. 20.4), and minimum individual regret of the opponent (represented by min Q, Eq. 20.5).

The VIKOR algorithm determines the weight stability intervals for the obtained compromise solution with the input weights given by the experts.

The model uses the DEMATEL and ANP procedures to obtain the weights of criteria with dependence and feedback and uses the VIKOR method to obtain the compromise solution.

20.4 Empirical Case and Results

An empirical case of a high-tech company, as the example illustrates, shows the proposed method. The empirical findings these four factors have the relationship of mutually interactive and self-feedback. Therefore, this study provides a set of patterns, and take major player of High-Tech company, explained that manufacturer of using the way of business strategies in a dynamic industry nature. The result presents a comprehensive framework for making judgments and to move forward in business strategies development that is essential. The function is refers to completes this position or duty group of important abilities, therefore the application function examines whether matches is a very good concept.

Empirical case of high-tech company as an example is illustrated to show the proposed method. Empirical findings of these four factors (through DANP calculation) have a mutually interactive and self-feedback relationship (as Table 20.1 and Table 20.2).

Table 20.1 The distance of influences by aspects

Aspect	A	B	C	D	R	R+S	R-S	Rank
A PC	0.297	0.286	0.310	0.301	1.193	2.414	-0.029	3
B TC	0.321	0.284	0.321	0.289	1.216	2.342	0.090	1
C RD	0.298	0.275	0.284	0.289	1.146	2.367	-0.075	4
D MS	0.306	0.282	0.306	0.284	1.178	2.341	0.014	2
S	1.222	1.126	1.221	1.164	-	-	-	-

Table 20.1 "Technology Competence" will be important to fit the change of the outside environment close to the real requirement and get the real necessity of customer needs. Besides, it shows the "Marketing and Service" can be a potential for challenge and to complete with rivals in a speedy growth industry. Since Fig. 20.4 demonstrates the strategy forces from the calculation of influences degree and demonstrates a strategic mapping for the decision maker to clarify the moment of strategy implementation, it may provide a picture to simulate the situations.

However, NRM may also provide a guide as a strategic consideration base for forecasting and simulating, leading those dimensions in dynamic concerns (one

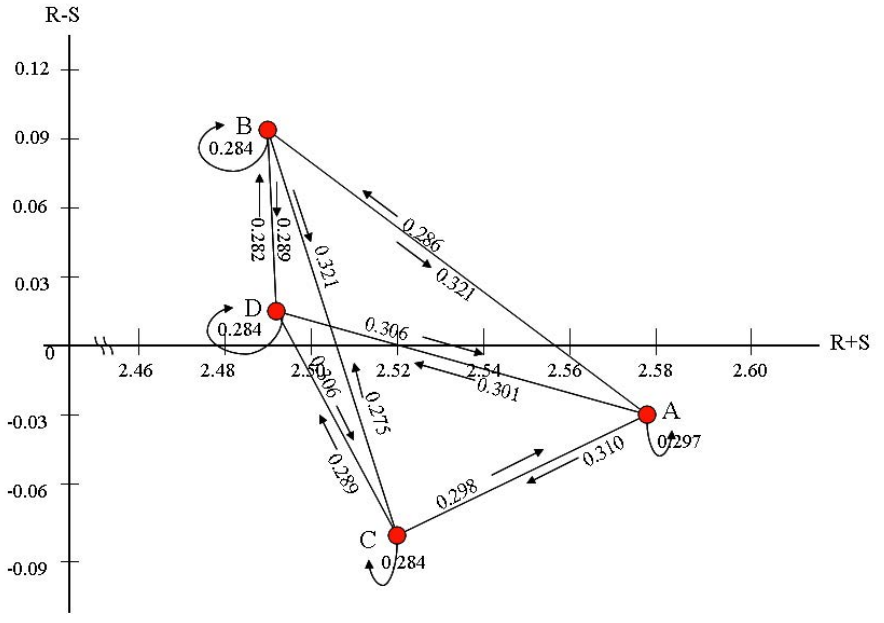


Fig. 20.4 NRM as of the Strategy Forces by Dimensions

factor change may cause the others' change).Then, this static NRM is expected to turn into dynamic NRM, be re-shaped with pre-defaults influence forces, and to be planed for the future study.

As the rank result (Table 20.2), "b4" and "a1" both locate at higher position of the company's strengths. Instantly, for the manufacturing business, the "Response" tasks have been always discussed based on high-tech work environment. Due to the sensitivity of market and economic changes, globalized changes may lead the change of corporate business strategies to fit the requirement of market (customers) needs. In contrast, "a3" and "c4" both locate at a lower position as the company's weaknesses. Therefore, this study suggests that a continuous and monitored "Production Flexibility" is required for deployment of internal resources and providing a well mechanism for feedback to end-users in time.

The essential alternatives are provided for problems solving. According to Table 20.3, "R7: Financial support/ incentive program" and "R6: Competitiveness Evaluation System" are the most favorable and satisfying issues, while the performance evaluation is measured. It is not surprising that "R1: Innovation/Intelligent Property" reveals the lowest credit which shows the importance of knowledge sharing and accumulation that can possibly create a new hope of ideas generated toward a new challenge on the next business strategic move. The definitions of those alternatives are as follows:

Table 20.2 The distance of influences by criteria

	Criteria	R	S	R+S	R-S	Rank
A	a1 Production Flexibility	5.222	4.241	9.463	0.981	2
	a2 Information Transparency	4.517	5.329	9.846	-0.812	14
	a3 Simplify operation – Process Complexity	4.222	5.676	9.898	-1.454	16
	a4 Quality – Yield control	5.120	4.299	9.419	0.821	3
B	b1 Production Application	4.387	5.140	9.527	-0.753	13
	b2 Compatibility	5.102	4.336	9.438	0.766	5
	b3 Innovation	5.104	4.308	9.412	0.796	4
	b4 Response	5.224	4.235	9.459	0.989	1
C	c1 Key components	4.691	4.726	9.417	-0.035	10
	c2 Patent	4.795	4.669	9.464	0.126	7
	c3 IP Library	4.627	4.747	9.374	-0.120	11
	c4 Device Mapping	4.227	5.392	9.619	-1.165	15
D	d1 Response	4.524	4.924	9.448	-0.400	12
	d2 Sensitivity	4.725	4.691	9.416	0.034	8
	d3 Service modes	4.773	4.795	9.568	-0.022	9
	d4 Growth	4.820	4.572	9.392	0.248	6

Table 20.3 Performance Evaluation by VIKOR

Criteria / Alternatives	Local weight	Global weight (ANP)	Aspire level	R1	R2	R3	R4	R5	R6	R7
A. Production capability (PC)	0.257	-	-	7.267	7.504	8.006	6.996	6.743	7.496	8.251
a1 Production Flexibility	0.251	0.0644	10	8	7	9	8	7	8	9
a2 Information Transparency	0.253	0.0650	10	9	8	8	5	4	7	8
a3 Simplify operation – Process Complexity	0.245	0.0629	10	5	7	7	7	7	8	8
a4 Quality – Yield control	0.251	0.0645	10	7	8	8	8	9	7	8
B. Technology competence (TC)	0.257	-	-	4.765	5.214	5.509	5.692	5.481	5.959	6.226
b1 Production Application	0.239	0.0614	10	5	6	6	8	7	9	8
b2 Compatibility	0.255	0.0655	10	6	8	8	9	9	8	7
b3 Innovation	0.239	0.0613	10	6	8	7	8	7	7	8
b4 Response	0.267	0.0686	10	8	7	9	7	8	8	9
C. R&D (RD)	0.257	-	-	5.852	5.756	5.726	8.000	7.364	7.480	7.852
c1 Key components	0.209	0.0538	10	4	6	7	8	6	8	6
c2 Patent	0.271	0.0695	10	7	6	5	8	9	8	9
c3 IP Library	0.276	0.0708	10	6	6	7	8	8	7	8
c4 Device Mapping	0.244	0.0627	10	6	5	4	8	6	7	8
D. Marketing & service (MS)	0.257	-	-	7.202	7.735	6.786	6.743	6.980	7.471	6.284
d1 Response	0.265	0.0679	10	7	7	8	7	7	6	7
d2 Sensitivity	0.276	0.0710	10	6	8	7	6	7	8	6
d3 Service modes	0.239	0.0615	10	9	8	5	8	6	8	7
d4 Growth	0.220	0.0564	10	7	8	7	6	8	8	5
TOTAL	5.027	1.0272	10	6.272	6.552	6.507	6.858	6.642	7.101	7.153
Performance Ranking for Alternative (Priority)		-	-	7	5	6	3	4	2	1

R1 : Innovation/Intelligent Property: Build an innovative environment for continuous technology development achievement.
R2 : Knowledge Platform: provide a knowledge platform for knowledge accumulation, problem solving, lesson learned, and information sharing.
R3 : Response System: External environmental change to cause Market/Operation strategies to be adjusted.
R4 : Communication System: a wide channel for customer services in viewing the progress of the online production and 24-h online service.
R5 : Efficiency Evaluation System: Internal and external factors to move the operation work effectively.
R6 : Competitiveness Evaluation System: Evaluate the way of technology adoption and shorten the time of technology development into the production phase.
R7 : Financial support/ incentive program: Budget plan and financial support.

20.5 Discussions and Implications

Where there was most evidence of improvements in team performance attributed to management development, these were typically related to process issues such as innovative activities, cost control, knowledge management, quality of service, better focus on customer needs, and improved customer relations, etc. Managers in some organizations cited evidence of more strategic behavior and identified better procedures and monitoring of actions. Technology improvement from cost reductions, efficiency gains, and more effective quality control were also reported. To face external change and to fit the requirement of business opportunities, appropriate dynamical strategies give a challenged team performance arising from strategy.

Major difficulties of measurement of strategy forces and of attributing improvements to strategy in management development were encountered, especially where the same measures were used for function and business performance, or where the performance was regarded as a function of a manager's performance. Equally, business performance was difficult to separate from organizational performance and was affected by extraneous factors.

The alternatives which were ranked highest in terms of organizational performance improvements displayed quantified, written, and corroborated evidence and respondents were unequivocal that strategic management development had contributed to all prime measures of business efficiency. The framework appears to be intuitive to managers and applicants who quickly adopt the terminology of the strategy forces. It offers a means of guiding easily recognized environmental components and provides a framework for combining these components to describe possible scenarios. In addition, it clearly distinguishes between the perspectives of different industry nature and increases the likelihood that decision makers will incorporate these perspectives into their decisions.

The decision-making and influence processes operating in decision events are likely to be advanced through a closer alignment with the problem solving and decision-making in the real world.

20.6 Conclusion

The conclusions and learning points to be drawn from the research outlined above relate both to the substantive findings, which contribute to the debate on the role of business in improving performance, and the procedural aspects of the methodology developed to undertake the study.

A case-study approach was necessary in order to capture qualitative information, including such 'soft' measures as perceptions of individuals, as well as exploring the relevant development and performance processes within organizations at different levels. In adopting such a methodology, however, there was a risk of marginalizing the findings because policy makers invariably seek 'hard' measures, with quantitative data that can be subject more easily to statistical tests. The challenge, therefore, was to devise a methodology that could permit hard conclusions from soft data.

The strategy force describes the structure of business systems and the industry nature of the influence mechanisms that determine outcomes. The strategy force's reliance on business decisions rather than operation activities as the key structural component of the framework provides a flexible and generalizable framework around which existing strategy force research can be arrayed. It also provides a means of aligning technology research with the problem solving and decision-making literatures. More work remains to be done to fully identify and understand influence mechanisms in strategy force systems. The strategy force can assist these efforts by offering a simple and easily recognizable structure for organizing research and practice in a complex field where a great deal is yet to be learned.

20.7 Future Study

The future study is subject to dynamic decision space (DDS), with the efficient consideration on improving hybrid MCDM method which this study proposed; the rough set concept [8] and Markov process are both suggested to be applied to DDS system (see Figure 20.2 in gray print) in further study for conducting effective and feasible strategies deployment.

References

1. Duckstein, L., Opricovic, S.: Multiobjective optimization in river basin development. Water Resources Research 16(1), 14–20 (1980)

2. Freimer, M., Yu, P.: Some new results on compromise solutions for group decision problems. Management Science, 688–693 (1976)
3. Hsu, C., Chen, K., Tzeng, G.H.: Fmcdm with fuzzy dematel approach for customers choice behavior model. International Journal of Fuzzy Systems, 236–246 (2007)
4. Lo, M.C., Michnik, J., Cheng, L.P.: Technology assessment as a guidance to business management of new technologies. In: 2007 IEEE International Conference on Industrial Engineering and Engineering Management, pp. 75–79 (2007), doi:10.1109/IEEM.2007.4419154
5. Opricovic, S., Tzeng, G.H.: Multicriteria planning of post-earthquake sustainable reconstruction. Computer-Aided Civil and Infrastructure Engineering 17(3), 211–220 (2002)
6. Opricovic, S., Tzeng, G.H.: Compromise solution by mcdm methods: A comparative analysis of vikor and topsis. European Journal of Operational Research 156(2), 445–455 (2004)
7. Opricovic, S., Tzeng, G.H.: Extended vikor method in comparison with outranking methods. European Journal of Operational Research 178(2), 514–529 (2007)
8. Pawlak, Z.: Rough sets. International Journal of Parallel Programming 11(5), 341–356 (1982)
9. Porter, M.: Towards a dynamic theory of strategy. Strategic Management Journal 12(S2), 95–117 (1991)
10. Saaty, R.: The Analytic Hierarchy Process (AHP) for Decision Making and The Analytic Network Process (ANP) for Decision Making with Dependence and Feedback. Creative Decisions Foundation, Pittsburgh (2003)
11. Saaty, T.: Axiomatic foundation of the analytic hierarchy process. Management Science, 841–855 (1986)
12. Saaty, T.: Rank generation, preservation, and reversal in the analytic hierarchy decision process. Decision Sciences 18(2), 157–177 (1987)
13. Saaty, T.: Decision making for leaders: the analytic hierarchy process for decisions in a complex world, vol. 2. RWS publications (1988)
14. Saaty, T.: The analytic network process: decision making with dependence and feedback. RWS Publications (1996)
15. Saaty, T.: Fundamentals of the analytic network process. In: Proceedings of the 5th International Symposium on the Analytic Hierarchy Process, pp. 12–14 (1999)
16. Saaty, T.: Fundamentals of the analytic network processxmultiple networks with benefits, costs, opportunities and risks. Journal of Systems Science and Systems Engineering 13(3), 348–379 (2004)
17. Tzeng, G.H., Chiang, C.H., Li, C.W.: Evaluating intertwined effects in e-learning programs: A novel hybrid mcdm model based on factor analysis and dematel. Expert Systems with Applications 32(4), 1028–1044 (2007)
18. Velosa, A.: Semiconductor manufacturing: Boom busts, and globalization. The Bridge 35(1) (2005)
19. Yu, P.L.: A class of solutions for group decision problems. Management Science, 936–946 (1973)
20. Zadeh, L.: Fuzzy sets. Information and Control 8(3), 338–353 (1965)

Chapter 21
Rough Set-Based Feature Selection: Criteria of Max-Dependency, Max-Relevance, and Max-Significance

Pradipta Maji and Sushmita Paul*

Abstract. Feature selection is an important data pre-processing step in pattern recognition and data mining. It is effective in reducing dimensionality and redundancy among the selected features, and increasing the performance of learning algorithm and generating information-rich features subset. In this regard, the chapter reports on a rough set-based feature selection algorithm called maximum relevance-maximum significance (MRMS), and its applications on quantitative structure activity relationship (QSAR) and gene expression data. It selects a set of features from a high-dimensional data set by maximizing the relevance and significance of the selected features. A theoretical analysis is reported to justify the use of both relevance and significance criteria for selecting a reduced feature set with high predictive accuracy. The importance of rough set theory for computing both relevance and significance of the features is also established. The performance of the MRMS algorithm, along with a comparison with other related methods, is studied on three QSAR data sets using the R^2 statistic of support vector regression method, and on five cancer and two arthritis microarray data sets by using the predictive accuracy of the K-nearest neighbor rule and support vector machine.

Keywords: Rough sets, feature selection, dependency, relevance, significance, classification, regression, quantitative structure activity relationship, microarray data, gene selection.

21.1 Introduction

Feature selection or dimensionality reduction of a data set is an essential preprocessing step used for pattern recognition, data mining, and machine learning [10,11]. It

Pradipta Maji and Sushmita Paul
Machine Intelligence Unit, Indian Statistical Institute
203 B. T. Road, Kolkata, 700 108, India
e-mail: {pmaji,sushmita_t}@isical.ac.in

* The work was done when one of the authors, S. Paul, was a Senior Research Fellow of the Council of Scientific and Industrial Research, Government of India.

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 393–418.
springerlink.com

is an important problem related to mining large data sets, both in dimension and size. Prior to analysis of the data set, preprocessing the data to obtain a smaller set of representative features and retaining the optimal salient characteristics of the data not only decrease the processing time, but also lead to more compactness of the models learned and better generalization. Hence, the general criterion for reducing the dimension is to preserve the most relevant information of the original data according to some optimality criteria [10,11].

In feature selection process, an optimal feature subset is always relative to a certain criterion. In general, different criteria may lead to different optimal feature subsets. However, every criterion tries to measure the discriminating ability of a feature or a subset of features to distinguish different class labels. To measure the feature-class relevance of a feature, different statistical and information theoretic measures such as the F test, t test [28], entropy, information gain, mutual information [41], normalized mutual information [31], and f-information [32] are typically used, and the same or a different metric like mutual information, f-information, the L_1 distance, Euclidean distance, and Pearson's correlation coefficient [20,41] is employed to calculate the redundancy among different features. However, as the F test, t test, Euclidean distance, and Pearson's correlation depend on the actual values of the data set, they are very much sensitive to noise or outlier of the data set [20,41]. On the other hand, as information measures depend only on the probability distribution of a random variable rather than on its actual values, they are more effective to evaluate both relevance and redundancy of the features [15,31,41,49].

Rough set theory, proposed by Pawlak [40,47], is a powerful paradigm to deal with uncertainty, vagueness, and incompleteness. It has been applied to fuzzy rule extraction [9], reasoning with uncertainty, fuzzy modeling, feature selection [26], microarray data analysis [12,49,53], and so forth. It is proposed for indiscernibility in classification according to some similarity [40]. The rough set theory has been applied successfully to feature selection of discrete valued data [19]. Given a data set with discretized attribute values, it is possible to find a subset of the original attributes using rough set theory that are the most informative; all other attributes can be removed from the data set with minimal information loss. From the dimensionality reduction perspective, informative features are those that are most useful in determining classifications from their values [7].

One of the popular rough set-based feature selection algorithms is the quick reduct algorithm [8,9] in which the dependency or quality of approximation of single attribute is first calculated with respect to the class labels or decision attribute. After selecting the best attribute, other attributes are added to it to produce better quality. Additions of attributes are stopped when the final subset of attributes has the same quality as that of maximum possible quality of the data set or the quality of the selected attributes remains same. Other notable algorithms include discernibility matrix-based method [25,46], and dynamic reducts [3]. However, all these approaches are computationally very costly. The variable precision rough set model [17,56], tolerance rough sets [22,39], and probabilistic rough sets [57] are the extensions of the original rough set-based knowledge representation. Different heuristic approaches based on rough set theory are also developed for feature

selection [35]. Combining rough sets and genetic algorithms, different algorithms have been proposed in [4, 48] to discover optimal subset of features.

In this chapter, a feature selection method called maximum relevance–maximum significance (MRMS) is reported, which selects a set of features from a high-dimensional data set by maximizing both relevance and significance of the selected features. It employs rough set theory to compute the relevance and significance of the features. Hence, the only information required in the MRMS algorithm is in the form of equivalence partitions for each feature, which can be automatically derived from the given data set. This avoids the need for domain experts to provide information on the data involved and ties in with the advantage of rough sets is that it requires no information other than the data set itself. The use of both relevance and significance criteria for selecting features with high predictive accuracy is theoretically justified based on the rough set theory. The effectiveness of the MRMS algorithm, along with a comparison with other feature selection algorithms, is demonstrated on two different types of problems. The first problem comprises quantitative structure activity relationship (QSAR) data and the second one constitutes gene expression data. The performance of the MRMS method on three QSAR data sets is assessed by using the R^2 statistic of support vector regression method, while for evaluating effectiveness of the MRMS algorithm over gene expression data sets, the predictive accuracy of the K-nearest neighbor rule and support vector machine is used.

The structure of the rest of this chapter is as follows: Section 21.2 introduces the necessary notions of rough sets. The theoretical analysis on the relationships of dependency, relevance, and significance is presented in Section 21.3 using rough sets. The rough set-based feature selection method is described in Section 21.4 for selecting relevant and significant features from high-dimensional data sets. Section 21.5 presents the application of rough set-based feature selection algorithm on QSAR data sets, while Section 21.6 describes the effectiveness of the algorithm on cancer and arthritis gene expression data sets. Finally, the concluding remarks are given in Section 21.7.

21.2 Rough Sets

The theory of rough sets, proposed by Pawlak [40, 47], begins with the notion of an approximation space, which is a pair $< \mathbb{U}, \mathbb{A} >$, where $\mathbb{U} = \{x_1, \cdots, x_i, \cdots, x_n\}$ is a non-empty set, the universe of discourse, and $\mathbb{A}$ is a family of attributes, also called knowledge in the universe. V is the value domain of $\mathbb{A}$ and f is an information function $f : \mathbb{U} \times \mathbb{A} \rightarrow V$. An approximation space is also called an information system [40]. Any subset $\mathbb{P}$ of knowledge $\mathbb{A}$ defines an equivalence, also called indiscernibility, relation $IND(\mathbb{P})$ on $\mathbb{U}$

$$IND(\mathbb{P}) = \{(x_i, x_j) \in \mathbb{U} \times \mathbb{U} | \forall a \in \mathbb{P}, f(x_i, a) = f(x_j, a)\}.$$

If $(x_i, x_j) \in IND(\mathbb{P})$, then x_i and x_j are indiscernible by attributes from $\mathbb{P}$. The partition of $\mathbb{U}$ generated by $IND(\mathbb{P})$ is denoted as

$$\mathbb{U}/IND(\mathbb{P}) = \{[x_i]_{\mathbb{P}} : x_i \in \mathbb{U}\} \tag{21.1}$$

where $[x_i]_{\mathbb{P}}$ is the equivalence class containing x_i. The elements in $[x_i]_{\mathbb{P}}$ are indiscernible or equivalent with respect to knowledge $\mathbb{P}$. Equivalence classes, also termed as information granules, are used to characterize arbitrary subsets of $\mathbb{U}$. The equivalence classes of $IND(\mathbb{P})$ and the empty set $\emptyset$ are the elementary sets in the approximation space $< \mathbb{U}, \mathbb{A} >$.

Given an arbitrary set $X \subseteq \mathbb{U}$, in general, it may not be possible to describe X precisely in $< \mathbb{U}, \mathbb{A} >$. One may characterize X by a pair of lower and upper approximations defined as follows [40]:

$$\underline{\mathbb{P}}(X) = \bigcup\{[x_i]_{\mathbb{P}} \mid [x_i]_{\mathbb{P}} \subseteq X\} \text{ and } \overline{\mathbb{P}}(X) = \bigcup\{[x_i]_{\mathbb{P}} \mid [x_i]_{\mathbb{P}} \cap X \neq \emptyset\}. \tag{21.2}$$

Hence, the lower approximation $\underline{\mathbb{P}}(X)$ is the union of all the elementary sets which are subsets of X, and the upper approximation $\overline{\mathbb{P}}(X)$ is the union of all the elementary sets which have a non-empty intersection with X. The tuple $< \underline{\mathbb{P}}(X), \overline{\mathbb{P}}(X) >$ is the representation of an ordinary set X in the approximation space $< \mathbb{U}, \mathbb{A} >$ or simply called the rough set of X. The lower (respectively, upper) approximation $\underline{\mathbb{P}}(X)$ (respectively, $\overline{\mathbb{P}}(X)$) is interpreted as the collection of those elements of $\mathbb{U}$ that definitely (respectively, possibly) belong to X. The lower approximation is also called positive region, sometimes denoted as $POS_{\mathbb{P}}(X)$. A set X is said to be definable or exact in $< \mathbb{U}, \mathbb{A} >$ iff $\underline{\mathbb{P}}(X) = \overline{\mathbb{P}}(X)$. Otherwise, X is indefinable and termed as a rough set. $BN_{\mathbb{P}}(X) = \overline{\mathbb{P}}(X) \setminus \underline{\mathbb{P}}(X)$ is called a boundary set.

An information system $< \mathbb{U}, \mathbb{A} >$ is called a decision table if the attribute set $\mathbb{A} = \mathbb{C} \cup \mathbb{D}$, where $\mathbb{C}$ is the condition attribute set and $\mathbb{D}$ is the decision attribute set. An important issue in data analysis is discovering dependency between attributes. Intuitively, a set of attributes $\mathbb{D}$ depends totally on a set of attributes $\mathbb{C}$, denoted as $\mathbb{C} \Rightarrow \mathbb{D}$, if all attribute values from $\mathbb{D}$ are uniquely determined by values of attributes from $\mathbb{C}$. If there exists a functional dependency between values of $\mathbb{D}$ and $\mathbb{C}$, then $\mathbb{D}$ depends totally on $\mathbb{C}$. The dependency can be defined in the following way:

Definition 21.1. Given $\mathbb{C}, \mathbb{D} \subseteq \mathbb{A}$, it is said that $\mathbb{D}$ depends on $\mathbb{C}$ in a degree κ, denoted as $\mathbb{C} \Rightarrow_{\kappa} \mathbb{D}$, if

$$\kappa = \gamma_{\mathbb{C}}(\mathbb{D}) = \frac{|POS_{\mathbb{C}}(\mathbb{D})|}{|\mathbb{U}|}, \text{ where } 0 \leq \kappa \leq 1. \tag{21.3}$$

where $POS_{\mathbb{C}}(\mathbb{D}) = \cup \underline{\mathbb{C}} X_i$, X_i is the ith equivalence class induced by $\mathbb{D}$ and $|\cdot|$ denotes the cardinality of a set. If $\kappa = 1$, $\mathbb{D}$ depends totally on $\mathbb{C}$, if $0 < \kappa < 1$, $\mathbb{D}$ depends partially (in a degree κ) on $\mathbb{C}$, and if $\kappa = 0$, then $\mathbb{D}$ does not depend on $\mathbb{C}$ [40].

The change in dependency when an attribute is removed from the set of condition attributes is a measure of the significance of the attribute. The higher the change in dependency, the more significant the attribute is. If the significance is 0, then the attribute is dispensable.

Definition 21.2. Given $\mathbb{C}, \mathbb{D}$ and an attribute $\mathcal{A} \in \mathbb{C}$, the significance of the attribute $\mathcal{A}$ is defined as [40]:

$$\sigma_{\mathbb{C}}(\mathbb{D}, \mathcal{A}) = \gamma_{\mathbb{C}}(\mathbb{D}) - \gamma_{\mathbb{C}-\{\mathcal{A}\}}(\mathbb{D}). \tag{21.4}$$

21.3 Relationships of Max-Dependency, Max-Relevance, and Max-Significance

This section presents the relationships among Max-Dependency, Max-Relevance, and Max-Significance using the rough set theory.

21.3.1 Max-Dependency

Let $\mathbb{C} = \{\mathcal{A}_1, \cdots, \mathcal{A}_i, \cdots, \mathcal{A}_j, \cdots, \mathcal{A}_m\}$ denote the set of m condition attributes or features of a given data set. In terms of rough sets, the task of attribute or feature selection is to find a feature subset $\mathbb{S} \subseteq \mathbb{C}$ with $d < m$ features $\{\mathcal{A}_i\}$, which jointly have the largest dependency on the target class or decision attribute set $\mathbb{D}$. This scheme, called Max-Dependency, has the following form:

$$\max \mathcal{D}(\mathbb{S}, \mathbb{D}), \qquad \mathcal{D} = \gamma_{\{\mathcal{A}_i, i=1,\cdots,d\}}(\mathbb{D}), \tag{21.5}$$

where $\gamma_{\{\mathcal{A}_i, i=1,\cdots,d\}}(\mathbb{D})$ represents the dependency between the feature subset $\mathbb{S} = \{\mathcal{A}_i, i = 1, \cdots, d\}$ and target class label $\mathbb{D}$ and is given by (21.3).

Obviously, when d equals 1, the solution is the feature that maximizes $\gamma_{\mathcal{A}_j}(\mathbb{D})$; $(1 \leq j \leq m)$. When $d > 1$, a simple incremental search scheme is to add one feature at one time. This type of selection is called the first-order incremental search. By definition of first-order search, it is assumed that $\mathbb{S}_{d-1}$, that is, the set of $d-1$ features has already been obtained. The task is to select the optimal dth feature $\mathcal{A}_d$ from the set $\{\mathbb{C} - \mathbb{S}_{d-1}\}$ that contributes to the largest increase of $\gamma_{\mathbb{S}}(\mathbb{D})$. The quick reduct algorithm of Chouchoulas and Shen [8] is based on the principle of Max-Dependency, which is explained in Figure 21.1.

The dependency $\mathcal{D}$ in (21.5) is represented by the dependency of (21.3), that is, $\mathcal{D} = \gamma_{\mathbb{S}_d}(\mathbb{D})$, where $\mathbb{S}_d = \{\mathbb{S}_{d-1}, \mathcal{A}_d\}$. Hence, from the definition of dependency in rough sets, the first-order incremental search algorithm optimizes the following condition to select dth feature from the set $\{\mathbb{C} - \mathbb{S}_{d-1}\}$:

1. $\mathbb{S} \leftarrow \{\}$
2. do
3. $T \leftarrow \mathbb{S}$
4. $\forall_{\mathcal{A}_i} \in (\mathbb{C} - \mathbb{S})$
5. if $\gamma_{\mathbb{S}\cup\{\mathcal{A}_i\}}(\mathbb{D}) > \gamma_T(\mathbb{D})$
6. $T \leftarrow \mathbb{S} \cup \{\mathcal{A}_i\}$
7. $\mathbb{S} \leftarrow T$
8. until $\gamma_{\mathbb{S}}(\mathbb{D}) == \gamma_{\mathbb{C}}(\mathbb{D})$
9. return $\mathbb{S}$

Fig. 21.1 Quick Reduct Algorithm

$$\max_{\mathcal{A}_j \in \{\mathbb{C}-\mathbb{S}_{d-1}\}} \left\{ \gamma_{\{\mathbb{S}_{d-1},\mathcal{A}_j\}}(\mathbb{D}) \right\}, \tag{21.6}$$

which is equivalent to optimize the following condition given the set of selected features $\mathbb{S}_{d-1}$:

$$\max_{\mathcal{A}_j \in \{\mathbb{C}-\mathbb{S}_{d-1}\}} \{\gamma_{\{\mathbb{S}_{d-1},\mathcal{A}_j\}}(\mathbb{D}) - \gamma_{\mathbb{S}_{d-1}}(\mathbb{D})\} = \max_{\mathcal{A}_j \in \{\mathbb{C}-\mathbb{S}_{d-1}\}} \{\sigma_{\mathbb{S}_d}(\mathbb{D}, \mathcal{A}_j)\}. \tag{21.7}$$

Obviously, the Max-Dependency is equivalent to either maximizing the joint dependency between selected feature set and the target class label or maximizing the significance of the candidate feature with respect to the already-selected features.

Despite the theoretical value of Max-Dependency, it is often hard to generate the resultant equivalence classes due to two difficulties in the high-dimensional space: the number of samples is often insufficient and the generation of resultant equivalence classes is usually an ill-posed problem. Another drawback of Max-Dependency is the slow computational speed. These problems are most pronounced for real-life applications. If each feature has c categorical or discrete states and n samples, then d features could have a maximum $\min\{c^d, n\}$ equivalence classes. When the number of equivalence classes increases very quickly and gets comparable to the number of samples n, the joint dependency of these features cannot be estimated correctly. Hence, although Max-Dependency feature selection might be useful to select a very small number of features when n is large, it is not appropriate for real-life applications where the aim is to achieve high classification accuracy with a reasonably compact set of features.

21.3.2 Max-Relevance and Max-Significance

As Max-Dependency criterion is hard to implement, an alternative is to select features based on maximal relevance criterion (Max-Relevance). Max-Relevance is to search features satisfying (21.8), which approximates $\mathcal{D}(\mathbb{S}, \mathbb{D})$ in (21.5) with the mean value of all dependency values between individual feature $\mathcal{A}_i$ and class label $\mathbb{D}$:

$$\max \mathcal{R}(\mathbb{S},\mathbb{D}), \qquad \mathcal{R} = \frac{1}{|\mathbb{S}|}\sum_{\mathcal{A}_i\in\mathbb{S}} \gamma_{\mathcal{A}_i}(\mathbb{D}). \tag{21.8}$$

It is likely that features selected according to Max-Relevance could have rich redundancy, that is, the dependency among these features could be large. When two features highly depend on each other, the respective class discriminative power would not change much if one of them were removed. Therefore, the following maximal significance (Max-Significance) condition can be added to select mutually exclusive features:

$$\max \mathcal{S}(\mathbb{S},\mathbb{D}),\ \mathcal{S} = \frac{1}{|\mathbb{S}|(|\mathbb{S}|-1)}\sum_{\mathcal{A}_i\neq\mathcal{A}_j\in\mathbb{S}} \{\sigma_{\{\mathcal{A}_i,\mathcal{A}_j\}}(\mathbb{D},\mathcal{A}_i)+\sigma_{\{\mathcal{A}_i,\mathcal{A}_j\}}(\mathbb{D},\mathcal{A}_j)\}. \tag{21.9}$$

The criterion combining the above two constraints is called "maximum-relevance–maximum-significance" (MRMS) [33,34]. The operator $\Phi(\mathcal{R},\mathcal{S})$ is defined to combine $\mathcal{R}$ and $\mathcal{S}$, and the following simplest form is considered to optimize $\mathcal{R}$ and $\mathcal{S}$ simultaneously:

$$\max \Phi(\mathcal{R},\mathcal{S}), \qquad \Phi = \mathcal{R}+\mathcal{S}. \tag{21.10}$$

In practice [32,41], incremental search methods can be used to find the near-optimal features defined by $\Phi(.)$. Given the feature set $\mathbb{S}_{d-1}$ with $d-1$ features, the task is to select the dth feature from the set $\{\mathbb{C}-\mathbb{S}_{d-1}\}$. This is done by selecting the feature that maximizes $\Phi(.)$. The respective incremental algorithm optimizes the following condition:

$$\max_{\mathcal{A}_j\in\{\mathbb{C}-\mathbb{S}_{d-1}\}}\left[\gamma_{\mathcal{A}_j}(\mathbb{D})+\frac{1}{d-1}\sum_{\mathcal{A}_i\in\mathbb{S}_{d-1}}\sigma_{\{\mathcal{A}_i,\mathcal{A}_j\}}(\mathbb{D},\mathcal{A}_j)\right]. \tag{21.11}$$

Hence, the combination of Max-Relevance and Max-Significance, that is, the MRMS criterion, is equivalent to maximizing the dependency between the candidate feature $\mathcal{A}_d$ and class label $\mathbb{D}$ as well as maximizing the average value of all significance values of the candidate feature $\mathcal{A}_d$ with respect to the already-selected feature $\mathcal{A}_i\in\mathbb{S}_{d-1}$. The above discussions lead to the following conclusions:

1. Maximizing the first term of (21.11), that is, maximizing $\mathcal{R}(\mathbb{S},\mathbb{D})$ of (21.8), only leads to Max-Relevance. Clearly, the difference between Max-Relevance and Max-Dependency of (21.5) is rooted in the different definitions of dependency in terms of rough set theory. Equation (21.8) does not consider the joint effect of features on the target class $\mathbb{D}$. On the contrary, Max-Dependency of (21.5) considers the dependency between the data distribution in multi-dimensional space and the target class $\mathbb{D}$. This difference is critical in many circumstances.
2. Maximizing the second term of (21.11) only, that is, maximizing $\mathcal{S}(\mathbb{S},\mathbb{D})$ of (21.9), is equivalent to searching mutually exclusive or independent features. This is not sufficient for selecting highly discriminative features.

3. The equivalence between Max-Dependency and Max-Significance indicates that Max-Significance is an optimal first order implementation of Max-Dependency.
4. Compared to Max-Dependency, the MRMS criterion avoids the estimation of resultant equivalence classes for multiple features. Instead, computing the resultant equivalence classes for two features could be much easier and more accurate. This also leads to a more efficient feature selection algorithm.

21.4 Maximum Relevance-Maximum Significance Algorithm

In real data analysis such as the QSAR and microarray data, the data set may contain a number of insignificant features. The presence of such irrelevant and insignificant features may lead to a reduction in the useful information. Ideally, the selected features should have high relevance with the classes and high significance in the feature set. The features with high relevance are expected to be able to predict the classes of the samples. However, if insignificant features are present in the subset, they may reduce the prediction capability. A feature set with high relevance and high significance enhances the predictive capability. Accordingly, a measure is required that can enhance the effectiveness of feature set.

21.4.1 The Algorithm

In the MRMS algorithm, the rough set theory is used to select the relevant and significant features from high-dimensional data sets. Let $\mathbb{C} = \{\mathcal{A}_1, \cdots, \mathcal{A}_i, \cdots, \mathcal{A}_j, \cdots, \mathcal{A}_m\}$ denote the set of m features of a given data set and $\mathbb{S}$ the set of features. Define $\hat{f}(\mathcal{A}_i, \mathbb{D})$ as the relevance of the feature $\mathcal{A}_i$ with respect to the class labels $\mathbb{D}$, while $\tilde{f}(\mathcal{A}_i, \mathcal{A}_j)$ as the significance of the feature $\mathcal{A}_j$ with respect to the feature $\mathcal{A}_i$. The total relevance of all selected features is, therefore, given by

$$\mathcal{J}_{\text{relev}} = \sum_{\mathcal{A}_i \in \mathbb{S}} \hat{f}(\mathcal{A}_i, \mathbb{D}) \tag{21.12}$$

while the total significance among the selected features is

$$\mathcal{J}_{\text{signf}} = \sum_{\mathcal{A}_i \neq \mathcal{A}_j \in \mathbb{S}} \tilde{f}(\mathcal{A}_i, \mathcal{A}_j). \tag{21.13}$$

Therefore, the problem of selecting a set $\mathbb{S}$ of relevant and significant features from the whole set $\mathbb{C}$ of m features is equivalent to maximize both $\mathcal{J}_{\text{relev}}$ and $\mathcal{J}_{\text{signf}}$, that is, to maximize the objective function $\mathcal{J}$, where

$$\mathcal{J} = \mathcal{J}_{\text{relev}} + \beta \mathcal{J}_{\text{signf}} = \sum_{\mathcal{A}_i \in \mathbb{S}} \hat{f}(\mathcal{A}_i, \mathbb{D}) + \beta \sum_{\mathcal{A}_i \neq \mathcal{A}_j \in \mathbb{S}} \tilde{f}(\mathcal{A}_i, \mathcal{A}_j) \tag{21.14}$$

with β a weight parameter. To solve the above problem, the following greedy algorithm can be used.

1. Initialize $\mathbb{C} \leftarrow \{\mathcal{A}_1, \cdots, \mathcal{A}_i, \cdots, \mathcal{A}_j, \cdots, \mathcal{A}_m\}, \mathbb{S} \leftarrow \emptyset$.
2. Calculate the relevance $\hat{f}(\mathcal{A}_i, \mathbb{D})$ of each feature $\mathcal{A}_i \in \mathbb{C}$.
3. Select the feature $\mathcal{A}_i$ as the most relevant feature that has the highest relevance value $\hat{f}(\mathcal{A}_i, \mathbb{D})$. In effect, $\mathcal{A}_i \in \mathbb{S}$ and $\mathbb{C} = \mathbb{C} \setminus \mathcal{A}_i$.
4. Repeat the following two steps until the desired number of features is selected.
5. Calculate the significance of each of the remaining features of $\mathbb{C}$ with respect to the selected features of $\mathbb{S}$ and remove it from $\mathbb{C}$ if it has zero significance value with respect to any one of the selected features.
6. From the remaining features of $\mathbb{C}$, select features $\mathcal{A}_j$ that maximize the following condition:

$$\hat{f}(\mathcal{A}_j, \mathbb{D}) + \frac{\beta}{|\mathbb{S}|} \sum_{\mathcal{A}_i \in \mathbb{S}} \tilde{f}(\mathcal{A}_i, \mathcal{A}_j). \tag{21.15}$$

   As a result, $\mathcal{A}_j \in \mathbb{S}$ and $\mathbb{C} = \mathbb{C} \setminus \mathcal{A}_j$.

Both the relevance and significance of a feature are calculated based on the rough set theory. The relevance $\hat{f}(\mathcal{A}_i, \mathbb{D})$ of a feature $\mathcal{A}_i$ with respect to the class labels $\mathbb{D}$ is calculated using (21.3), while significance $\tilde{f}(\mathcal{A}_i, \mathcal{A}_j)$ of the feature $\mathcal{A}_j$ with respect to the already-selected feature $\mathcal{A}_i$ is computed using (21.4).

21.4.2 Computational Complexity

The rough set-based feature selection method has low computational complexity with respect to the number of features in the original data set.

1. The computation of the relevance of m features is carried out in step 2 of the MRMS algorithm, which has $\mathcal{O}(m)$ time complexity.
2. The selection of the most relevant feature from the set of m features, which is carried out in step 3, has also a complexity $\mathcal{O}(m)$.
3. There is only one loop in step 4 of the MRMS feature selection method, which is executed $(d-1)$ times, where d represents the number of selected features.
   a. The computation of significance of a candidate feature with respect to the already-selected features takes only a constant amount of time. If $\acute{m}$ represents the cardinality of the already-selected feature set, the total complexity to compute the significance of $(m - \acute{m})$ candidate features, which is carried out in step 5, is $\mathcal{O}(m - \acute{m})$.
   b. The selection of a feature from $(m - \acute{m})$ candidate features by maximizing both relevance and significance, which is carried out in step 6, has also a complexity $\mathcal{O}(m - \acute{m})$.

Hence, the total complexity to execute the loop $(d-1)$ times is $(\mathcal{O}((d-1)((m-\acute{m})+(m-\acute{m})))) = \mathcal{O}(d(m-\acute{m}))$.

In effect, the selection of a set of d relevant and significant features from the whole set of m features using rough set-based first-order incremental search method has an overall computational complexity of $(\mathcal{O}(m)+\mathcal{O}(m)+\mathcal{O}(d(m-\acute{m}))) = \mathcal{O}(m)$ as $d, \acute{m} << m$.

21.4.3 Generation of Equivalence Classes

Different discretization methods, such as discretization based on equal frequency binning [16], and roughfication method [49] can be employed to discretize the continuous values. In real data set, the feature values of samples are generally continuous. In many cases, class labels are also continuous. Hence, to measure both relevance and significance of features using rough set theory, the continuous feature values are usually divided into several discrete partitions to generate equivalence classes. The discretization method reported in [32] is employed to discretize the continuous feature values. The values of an attribute or feature are discretized using mean μ and standard deviation σ computed over n values of that attribute: any value larger than $(\mu + \frac{\sigma}{2})$ is transformed to state 1; any value between $(\mu - \frac{\sigma}{2})$ and $(\mu + \frac{\sigma}{2})$ is transformed to state 0; any value smaller than $(\mu - \frac{\sigma}{2})$ is transformed to state -1 [32]. The equivalence classes are then generated to compute both relevance and significance of features.

21.5 Molecular Descriptor Selection for Fitting QSAR Model

The quantitative structure activity relationship (QSAR) is the process by which chemical structure is quantitatively correlated with a well-defined process such as biological activity or other molecular property. Biological activity can be expressed quantitatively as in the concentration of a substance required to give a certain biological response. Additionally, when physiochemical properties or structures are expressed by numbers, one can form a mathematical relationship or quantitative structure activity relationship between the two. The mathematical expression can then be used to predict the biological response of other unknown chemical structures. The properties that describe the molecule quantitatively are known as molecular descriptors. Molecular descriptors can be obtained by calculated methods or experimental methods. In the calculated method, a mathematical procedure is used that transforms chemical information into a number such as surface areas (polar, nonpolar), dipole moment, and volume. On the other hand, in the experimental method, some standardized experiments are conducted to measure a molecular descriptor such as melting point, partition coefficients, and refractive index. The molecular descriptors describe different aspects of a molecule, compare different molecular

structures, different conformations of the same molecule, and database storage, and relate structure to activity [21,27,29].

However, among the large amount of descriptors, only a small fraction is effective for performing the predictive modeling task. Also, a small subset of descriptors is desirable in developing the QSAR data-based predicting tools for delivering precise, reliable, and interpretable results. With the descriptor selection results, the cost of biological experiment and decision can be greatly reduced by analyzing only the effective descriptors. Hence, identifying a reduced set of most relevant descriptors is the goal of descriptor selection. The small number of molecules and a large number of descriptors make this a more relevant and challenging problem in the QSAR method. This is an important problem in machine learning and referred to as feature selection [10].

The performance of rough set-based MRMS method on the QSAR data sets is extensively studied and compared with that of some existing algorithms [33]. All the algorithms are implemented in C language and run in LINUX environment having machine configuration Pentium IV, 2.8 GHz, 1 MB cache, and 512 MB RAM. To analyze the performance of different algorithms, the experimentation is done on three QSAR data sets. The major metric for evaluating the performance of different algorithms is the R^2 statistic of support vector regression method. The weight parameter β of the MRMS algorithm is set to 1.0 for the QSAR data sets [33].

21.5.1 QSAR Data Sets

In this chapter, the following three QSAR data sets are used that are available at http://www.cheminformatics.org.

1. *Steroid Data Set:* This data set contains 31 steroid molecules presented in MOL format, which is used in cheminformatics applications for storing atomic coordinates, chemical bond information, and metadata of the 3D structure of a single chemical compound in plain text tabular format. The $\log(1/k)$ values of these molecules are also given. All these molecules are categorized into three activity classes. Among them, 11 are reported as high activity molecules, 9 moderate and rest 11 as lowest activity molecules.
2. *Small Dopamine Data Set:* It contains 26 dopamine molecules given in MOL format. The biological activity of these molecules is also available.
3. *Large Dopamine Data Set:* This data set consists of 116 dopamine molecules that are given along with their molecular descriptors in binary form. The continuous valued biological activity of each molecule is also given.

Both steroid and small dopamine data sets are available in MOL format. The molecular descriptors of these data sets are obtained using MODEL software [29], which calculates approximately 4000 molecular descriptors for each molecule. The calculated descriptors cover different aspects of the molecular structure including topological, electronic, constitutional, geometrical, and physical descriptors.

21.5.2 Support Vector Machine

The support vector machine (SVM) [54] is a margin classifier that draws an optimal hyperplane in the feature vector space; this defines a boundary that maximizes the margin between data samples in different classes, therefore leading to good generalization properties. A key factor in the SVM is to use kernels to construct nonlinear decision boundary. In the present work, radial basis function kernels are used.

The performance of the SVM is analyzed using R^2 statistic or coefficient of determination value. The R^2 statistic tells about the goodness of fit of a model and how well a regression approximates its attributes. The value of R^2 statistic ranges from 0 to 1. The nearer the value reaches to 1, the better is the approximation. The R^2 statistic can be calculated as follows:

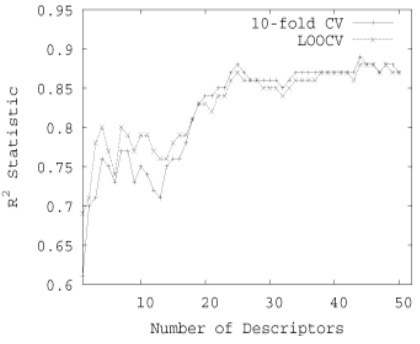
$$R^2 = 1 - \frac{SS_{\text{err}}}{SS_{\text{tot}}} \tag{21.16}$$

where $SS_{\text{tot}} = \Sigma_i (y_i - \bar{y})^2$ represents the total sum of squares, which is proportional to the sample variance, and $SS_{\text{err}} = \Sigma_i (y_i - f_i)^2$ is the sum of squared errors, also called the residual sum of squares. Here, $\bar{y}$ represents the mean of the observed data, while y_i and f_i are the ith observed and modeled or predicted values, respectively.

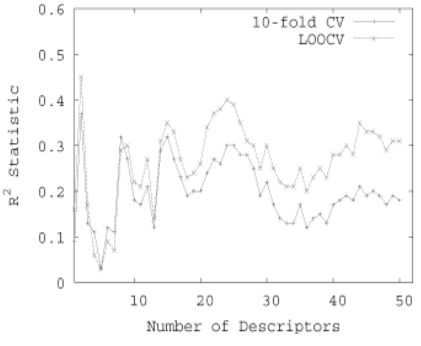
21.5.3 Performance Analysis

The experimental results on three QSAR data sets are presented in Fig. 21.2-21.6. Subsequent discussions analyze the results with respect to the R^2 statistic of the SVM. To compute the R^2 statistic of the SVM, both leave-one-out cross-validation (LOOCV) and 10-fold cross-validation (10-fold CV) are performed on each QSAR data set. The number of molecular descriptors selected ranges from 1 to 50.

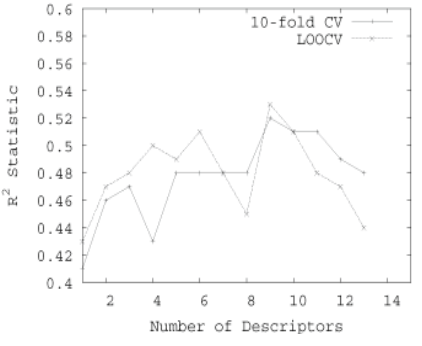
Fig. 21.2(a) presents the performance of the MRMS method on steroid molecules obtained by both tenfold CV and LOOCV, while Fig. 21.2(b) and 21.2(c) depicts that for small and large dopamine molecules, respectively. From Fig. 21.2(a), it is seen that as the number of selected descriptors of steroid molecules ranges from 1 to 15, the R^2 statistic of the SVM fluctuates in case of both tenfold CV and LOOCV. It indicates that the MRMS method gets stuck into local minima of the search space for this range. However, the R^2 statistic continuously increases with the increase in the number of selected descriptors for more than 15. Finally, the MRMS method attains its maximum R^2 statistic of 0.88 and 0.89 using only 44 descriptors for the LOOCV and tenfold CV, respectively. That is, the MRMS method is able to find out an optimum or near to optimum solution using 44 descriptors for both tenfold CV and LOOCV. On the other hand, from Fig. 21.2(b), it can be seen that in case of small dopamine molecules, two most relevant and significant descriptors are sufficient to achieve the maximum R^2 statistic values of 0.45 and 0.37 of the MRMS method for the LOOCV and tenfold CV, respectively. Finally, Fig. 21.2(c) depicts



(a) Steroid molecules



(b) Small dopamine



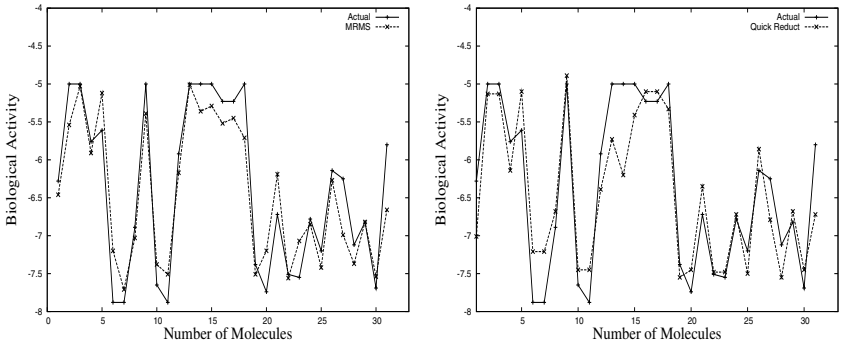
(c) Large dopamine

Fig. 21.2 Results obtained by tenfold cross-validation and leave-one-out cross-validation

the results for large dopamine molecules. From the results presented in Fig. 21.2(c), it is seen that the MRMS method attains maximum R^2 statistic of 0.53 with nine descriptors using the LOOCV, while for tenfold CV, the best R^2 statistic is 0.52 with the same number of descriptors. In other words, the MRMS method is able to find out optimum or near to optimum solutions using two and nine molecular descriptors for small and large dopamine molecules, respectively.

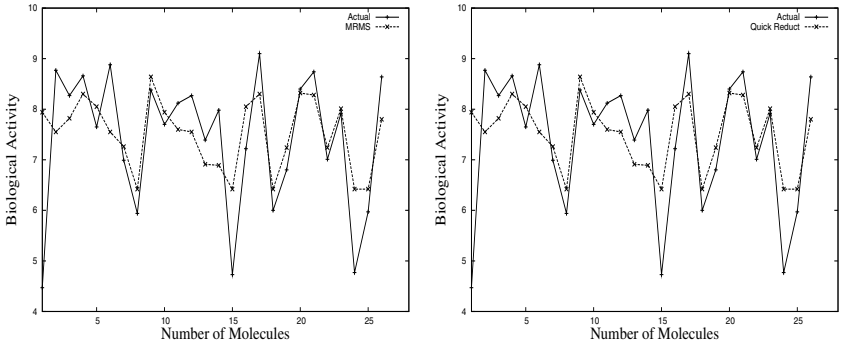
Figs. 21.3-21.5 present the comparative performance analysis of the MRMS method and one of the most popular rough set-based algorithm, called quick reduct algorithm [8]. All the results are reported for three QSAR data sets based on the LOOCV. The actual and obtained biological activity values of different molecules for three QSAR data sets are reported for comparison. The R^2 statistic values of quick reduct algorithm are 0.82, 0.45, and 0.56 for steroid, small dopamine, and large dopamine molecules, respectively. For tenfold CV, the R^2 statistic values of quick reduct algorithm are 0.83, 0.37, and 0.52 on steroid, small dopamine, and large dopamine, respectively. From the results reported in Figs. 21.3-21.5, it is seen that the performance of the MRMS method is better than quick reduct algorithm in case of steroid data set and comparable with quick reduct algorithm for both

small and large dopamine molecules. In this regard, it should be noted that another rough set-based algorithm, called discernibility matrix-based method [46], attains the R^2 statistic values of 0.79, 0.43, and 0.39 for steroid, small dopamine, and large dopamine molecules, respectively, using tenfold CV,



(a) MRMS method (b) Quick reduct algorithm

Fig. 21.3 Results for steroid molecules obtained by leave-one-out cross-validation

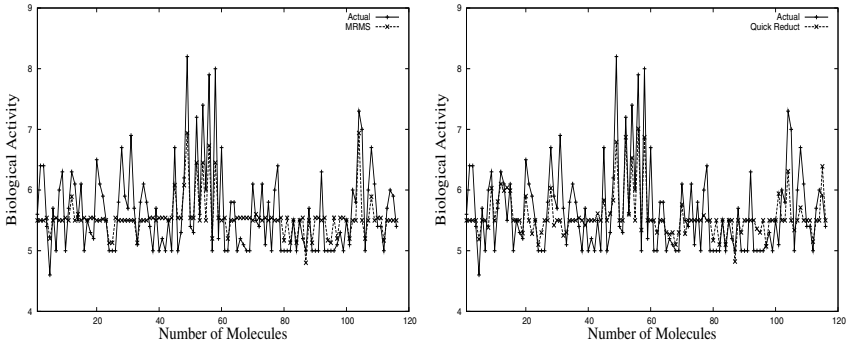


(a) MRMS method (b) Quick reduct algorithm

Fig. 21.4 Results for small dopamine molecules obtained by leave-one-out cross-validation

while the corresponding values for the LOOCV are 0.79, 0.61, and 0.41, respectively. However, as the computational complexity of both quick reduct method [8] and discernibility matrix-based method [46] is very high, they require significantly higher execution time compared to that of the MRMS algorithm.

Table 21.1 compares the execution time (in millisecond) of the MRMS algorithm and that of quick reduct algorithm [8] and discernibility matrix-based method [46] for three QSAR data sets. From the results reported in Table 21.1, it is seen that the execution time required for the MRMS algorithm is significantly lower than that of the other two algorithms, irrespective of the data sets used. As the computational



(a) MRMS method (b) Quick reduct algorithm

Fig. 21.5 Results for large dopamine molecules obtained by leave-one-out cross-validation

Table 21.1 Execution Time of Different Algorithms

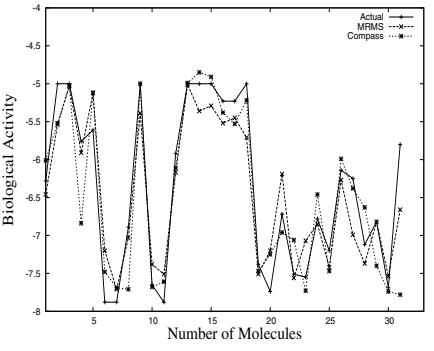
Data Set	Quick Reduct	Discern. Matrix	MRMS
Steroid	383253	55061	3498
Small Dopamine	350015	54044	4299
Large Dopamine	487735	35027	1755

complexity of both quick reduct algorithm and discernibility matrix-based method is polynomial in nature [8,46], they require significantly higher execution time compared to that of the MRMS algorithm. The significantly lesser execution time of the MRMS algorithm is achieved due to its linear computational complexity.

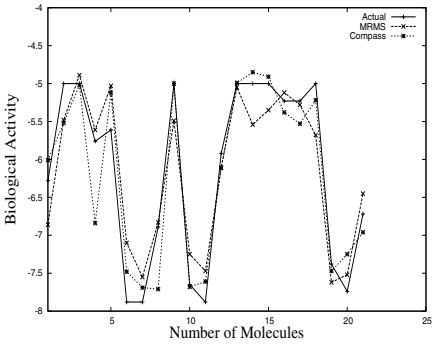
21.5.4 Comparative Performance Analysis

The MRMS method performs significantly better than different existing QSAR methods. To establish the superiority of the MRMS method, extensive experimentation is carried out on different QSAR data sets. Fig. 21.6(a) presents the predicted biological activity values of the MRMS method and Compass [18], an well-known existing QSAR model, along with the actual activity values. Results are reported based on the LOOCV. The R^2 statistic values corresponding to the MRMS method and Compass are 0.89 and 0.79, respectively. Next, the steroid data set is divided into two sets: training set of 21 molecules and test set of 10 molecules. The LOOCV results of 21 molecules obtained by the MRMS method as well as two well-known existing approaches, namely, Compass [18] and CoMFA [52], are reported in Table 21.2.

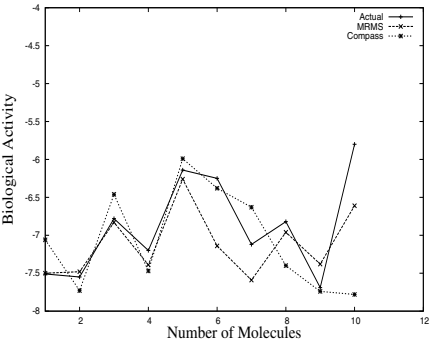
Fig. 21.6(b) and 21.6(c) depicts the actual and predicted values of the MRMS method and Compass [18] for 21 training and 10 test steroid molecules, respectively.



(a) LOOCV on 31 molecules



(b) 21 training molecules



(c) 10 testing molecules

Fig. 21.6 Results of MRMS and Compass on steroid molecules

Table 21.2 Result on Training Set of Steroid Data

	Methods	R^2 statistic
Existing Models	CoMFA	0.69
	Compass	0.89
Rough Sets	MRMS	0.97

A detailed comparison of the MRMS method with other existing 3D QSAR methods, namely, QS-SM [2], MS-WHIM [5], PARM [6], Compass [18], MEDV [30], COMSA [42], TQSAR [43], SOMFA [44], EEVA [51], EVA [52], and CoMFA [52] is presented in Table 21.3 on test set of steroid data, that is, molecules 22 to 31.

From the R^2 statistic reported in Tables 21.2 and 21.3, along with the results reported in Fig. 21.6(a)-21.6(c), it can be seen that the MRMS method outperforms different existing QSAR approaches in case of steroid data set. Also, the MRMS method predicts biological activity of 21 training and 10 test molecules significantly

Table 21.3 Result on Test Set of Steroid Data

	Methods	R^2 statistic
Existing Models	Compass	0.16
	MS-WHIM	0.28
	PARM	0.33
	TQSAR	0.16
	SOMFA	0.20
	EVA	0.36
	CoMFA	0.25
	COMSA	0.09
	MEDV	0.45
	QS-SM	0.36
	EEVA	0.36
Rough Sets	MRMS	0.67

better than the Compass [18]. Moreover, the model building phase of Compass takes about 1 min per molecule for steroid data set [18], which is significantly higher than that of the MRMS method.

Finally, the tenfold CV result of the MRMS method for large dopamine data is compared with the existing approach Boosting of Sventik et al. [50]. While the MRMS method achieves the R^2 value of 0.52 with nine attributes, the best result obtained by the Boosting method is 0.48, that is, the MRMS method performs significantly better than the existing method.

21.6 Gene Selection from Microarray Data

Recent advancement and wide use of high-throughput technology are producing an explosion in using gene expression phenotype for identification and classification in a variety of diagnostic areas. An important application of gene expression data in functional genomics is to classify samples according to their gene expression profiles such as to classify cancer versus normal samples or to classify different types or subtypes of cancer [13].

A microarray gene expression data set can be represented by an expression table, where each row in the expression table corresponds to one particular gene and each column to a sample [13]. However, for most gene expression data, the number of training samples is still very small compared to the large number of genes involved in the experiments. The number of samples is likely to remain small for many areas of investigation, especially for human data, due to the difficulty of collecting and processing microarray samples [13]. When the number of genes is significantly

greater than the number of samples, it is possible to find biologically relevant correlations of gene behavior with the sample categories [36].

However, among the large amount of genes, only a small fraction is effective in performing a certain task. Also, a small subset of genes is desirable in developing gene expression-based diagnostic tools for delivering precise, reliable, and interpretable results. With the gene selection results, the cost of biological experiment and decision can be greatly reduced by analyzing only the marker genes. Hence, identifying a reduced set of most relevant and significant genes is the goal of gene selection. The small number of training samples and a large number of genes make gene selection a more relevant and challenging problem in gene expression-based classification. This is an important problem in machine learning and referred to as feature selection [10, 23, 24].

The performance of rough set-based MRMS method is extensively studied and compared with that of some existing algorithms, namely, quick reduct algorithm [8], discernibility matrix-based approach [46], roughfication [49], and the methods proposed by Valdes and Barton [53] and Fang et al. [12]. The performance of the MRMS method is also compared with that of Max-Dependency and Max-Relevance criteria.

To analyze the performance of different algorithms, the experimentation is done on five cancer and two arthritis microarray data sets. For each data set, 50 top-ranked genes are selected for analysis, and each data set is preprocessed by standarizing each sample to zero mean and unit variance. The major metrics for evaluating the performance of different algorithms are the classification accuracy of the K-nearest neighbor (K-NN) rule and support vector machine (SVM).

In the present work, linear kernels are used in the SVM to construct nonlinear decision boundary. The K-nearest neighbor (K-NN) rule [11] is used for evaluating the effectiveness of the reduced gene set for classification. It classifies samples based on closest training samples in the feature space. A sample is classified by a majority vote of its K-neighbors, with the sample being assigned to the class most common among its K-nearest neighbors. The value of K, chosen for the K-NN, is the square root of the number of samples in the training set. To compute the prediction accuracy of both the SVM and K-NN rule, leave-one-out cross-validation (LOOCV) is performed on each gene expression data set.

21.6.1 Gene Expression Data Sets

In this chapter, publicly available five cancer and two arthritis data sets are used. Since binary classification is a typical and fundamental issue in diagnostic and prognostic prediction of both cancer and arthritis, different methods are compared using the following binary class data sets.

1. *Breast Cancer:* The breast cancer data set contains expression levels of 7129 genes in 49 breast tumor samples [55]. The samples are classified according to

their estrogen receptor (ER) status: 25 samples are ER positive, while the other 24 samples are ER negative.
2. *Leukemia:* It is an affymetrix high-density oligonucleotide array that contains 7070 genes and 72 samples from two classes of leukemia [13]: 47 acute lymphoblastic leukemia and 25 acute myeloid leukemia.
3. *Colon Cancer:* The colon cancer data set contains expression levels of 2000 genes and 62 samples from two classes [1]: 40 tumor and 22 normal colon tissues.
4. *Lung Cancer:* This data set contains 181 tissue samples: among them 31 are malignant pleural mesothelioma and the rest 150 adenocarcinoma of the lung [14]. Each sample is described by the expression levels of 12533 genes.
5. *Prostate Cancer:* In this data set, 136 samples are grouped into two classes: 77 prostate tumor and 59 prostate normal samples [45]. Each sample contains 12600 genes.
6. *Rheumatoid Arthritis versus Osteoarthritis (RAOA):* The RAOA data set consists of gene expression profiles of 30 patients: 21 with RA and 9 with OA [37]. The Cy5-labeled experimental cDNA and the Cy3-labeled common reference sample were pooled and hybridized to the lymphochips containing $\sim$18,000 cDNA spots representing genes of relevance in immunology [37].
7. *Rheumatoid Arthritis versus Healthy Controls (RAHC):* The RAHC data set consists of gene expression profiling of peripheral blood cells from 32 patients with RA, 3 patients with probable RA, and 15 age and sex matched healthy controls performed on microarrays with a complexity of $\sim$26K unique genes (43K elements) [38].

21.6.2 Importance of Rough Sets

In the MRMS method, both the relevance and significance of a gene are calculated based on the rough set theory. The relevance of a gene with respect to the class labels is calculated using (21.3), while significance of a gene with respect to the already-selected gene is computed using (21.4). However, other measures such as mutual information can also be used to compute both relevance and significance of a gene. In order to establish the importance of rough sets over mutual information with respect to the classification accuracy of both SVM and K-NN rule, extensive experimental results are reported in Table 21.4 for seven microarray data sets. The value of β is set to 1.0 for the MRMS method [34].

From the results reported in Table 21.4, it is seen that the performance of rough sets is better than that of mutual information in most of the cases. Out of total 14 cases, the MRMS criterion achieves significantly better results for rough sets in 9 cases. However, the mutual information provides better accuracy of the SVM for leukemia, prostate cancer, and RAHC data sets and that of the K-NN for colon and prostate cancer data sets.

Table 21.4 Comparative Performance of Rough Sets and Mutual Information Using LOOCV

Microarray Data Set	Quantitative Measures	Rough Sets		Mutual Information	
		Accuracy	Gene	Accuracy	Gene
Breast	SVM	100	18	97.96	11
	K-NN	100	45	93.88	6
Leukemia	SVM	97.2	22	98.61	19
	K-NN	98.6	47	95.83	25
Colon	SVM	87.1	5	87.1	5
	K-NN	83.9	3	88.71	40
Lung	SVM	100	34	99.45	2
	K-NN	100	38	99.45	2
Prostate	SVM	89.7	44	96.32	47
	K-NN	88.2	7	92.65	27
RAOA	SVM	100	5	100	7
	K-NN	100	3	100	11
RAHC	SVM	90	20	98	10
	K-NN	100	11	100	16

21.6.3 Effectiveness of MRMS Criterion

To establish the effectiveness of the MRMS criterion-based feature selection method over Max-Dependency and Max-Relevance criteria, extensive experimental results are reported in Table 21.5 for seven microarray data sets.

Table 21.5 Performance of Max-Dependency, Max-Relevance, and MRMS Using LOOCV

Microarray Data Set	Quantitative Measures	Max-Dependency		Max-Relevance		MRMS ($\beta = 1.0$)		MRMS ($0.0 < \beta < 1.0$)		
		Accuracy	Gene	Accuracy	Gene	Accuracy	Gene	Accuracy	Gene	Value of β
Breast	SVM	85.7	3	98	11	100	18	100	18	0.6-0.9
	K-NN	83.7	2	98	17	100	45	100	45	0.8-0.9
Leukemia	SVM	100	3	97.2	32	97.2	22	98.6	36	0.1
	K-NN	98.6	2	98.6	43	98.6	47	100	50	0.1-0.3
Colon	SVM	80.7	2	80.7	23	87.1	5	87.1	5	0.9
	K-NN	80.7	3	82.3	50	83.9	3	85.5	9	0.9
Lung	SVM	99.5	3	99.5	7	100	34	100	34	0.6-0.9
	K-NN	99.5	3	99.5	42	100	38	100	39	0.9
Prostate	SVM	84.6	4	81.6	47	89.7	44	89.7	44	0.9
	K-NN	88.2	4	91.2	5	88.2	7	88.2	7	0.1-0.9
RAOA	SVM	86.7	1	90	50	100	5	100	3	0.5-0.6
	K-NN	90	2	90	2	100	3	100	3	0.7-0.9
RAHC	SVM	70	1	94	16	90	20	94	36	0.1-0.4
	K-NN	84	3	90	11	100	11	100	11	0.5-0.9

Subsequent discussions analyze the results with respect to the classification accuracy of both SVM and K-NN rule. The best results obtained using Max-Dependency and Max-Relevance criteria on these data sets are also presented in this table for the

sake of comparison. The value of β varies from 0.0 to 1.0 for the MRMS criterion. In this context, it should be noted that the Max-Relevance criterion is equivalent to the MRMS criterion with $\beta = 0.0$, while the quick reduct algorithm of Chouchoulas and Shen [8] follows the Max-Dependency criterion.

21.6.3.1 Optimum Value of β

The parameter β regulates the relative importance of the significance of the candidate gene with respect to the already-selected genes and the relevance with the output class. If β is zero, only the relevance with the output class is considered for each gene selection. If β increases, this measure is incremented by a quantity proportional to the total significance with respect to the already-selected genes. The presence of a β value larger than zero is crucial to obtain good results. If the significance between genes is not taken into account, selecting the genes with the highest relevance with respect to the output class may tend to produce a set of redundant genes that may leave out useful complementary information.

The values of β for which the MRMS criterion-based feature selection algorithm achieves its best performance are reported in Table 21.5 on seven microarray data sets. From the results reported in this table, it is seen that the MRMS criterion attains its best performance at $\beta = 0.9$ for breast, colon, lung, and prostate cancer data sets using both SVM and K-NN rule, and for RAOA and RAHC data sets using only the K-NN rule. On the other hand, the MRMS algorithm provides its best results at $\beta = 0.1$ for leukemia data set using both SVM and K-NN rule and for RAHC data set using only the SVM. Hence, the MRMS criterion achieves its best performance for $0.1 \leq \beta \leq 0.9$ irrespective of the data sets and classifiers used.

21.6.3.2 Comparative Performance Analysis

From the results reported in Table 21.5, it is seen that the performance of MRMS criterion is better than that of Max-Dependency and Max-Relevance criteria in most of the cases. Out of 14 cases, the MRMS criterion achieves significantly better results than Max-Dependency or Max-Relevance in 11 cases. However, the Max-Dependency criterion provides better accuracy of the SVM for leukemia data set and the same accuracy of the K-NN rule with lower number of genes for prostate cancer data set than the MRMS criterion. Also, the Max-Relevance criterion achieves better accuracy of the K-NN rule for prostate cancer data set and the same accuracy of the SVM with lower number of genes for RAHC data set than the MRMS criterion. That is, both Max-Dependency and Max-Relevance criteria are useful to select a very small number of genes, but not appropriate to achieve high classification accuracy. Hence, the MRMS criterion must be used to get a reduced set of genes with high classification accuracy.

21.6.4 Performance of Different Rough Set Based Algorithms

Finally, the best results of different algorithms on seven microarray data sets are presented in Table 21.6. Subsequent discussions analyze the results with respect to the prediction accuracy of the SVM and K-NN rule. The best performance of some existing algorithms such as roughfication [49], quick reduct algorithm [8], discernibility matrix-based approach [46], and the methods proposed by Fang and Busse [12] and Valdes and Barton [53] is provided on same data sets for the sake of comparison.

Table 21.6 Comparative Performance Analysis of Different Algorithms

Microarray Data Set	Quantitative Measures	Fang & Busse		Roughfication		Valdes & Barton		Quick Reduct		Discern. Matrix		MRMS	
		Accuracy	Gene	Accuracy	Gene	Accuracy	Gene	Accuracy	Gene	Accuracy	Gene	Accuracy	Gene
Breast	SVM	73.5	7	77.6	7	81.7	1	85.7	3	71.4	5	100	18
	K-NN	71.4	6	79.6	49	89.8	1	83.7	2	73.5	3	100	45
Leukemia	SVM	86.1	6	84.7	16	93.1	1	100	3	95.8	4	98.6	36
	K-NN	79.2	6	80.6	7	93.1	1	98.6	2	91.7	1	100	50
Colon	SVM	64.5	1	85.5	241	85.5	1	80.7	2	83.9	4	87.1	5
	K-NN	61.1	2	80.7	6	85.5	1	80.7	3	82.3	5	85.5	9
Lung	SVM	99.5	4	*	*	97.2	1	99.5	3	99.5	5	100	34
	K-NN	98.9	3	*	*	97.2	1	99.5	3	93.9	5	100	38
Prostate	SVM	56.6	3	*	*	74.3	7	84.6	4	75.0	10	89.7	44
	K-NN	78.7	4	*	*	84.6	1	88.2	4	75.0	10	88.2	7
RAOA	SVM	70.0	1	86.7	1	83.3	1	86.7	1	76.7	4	100	3
	K-NN	73.3	1	93.3	3	90.0	1	90.0	2	86.7	3	100	3
RAHC	SVM	70.0	1	82.0	6	86.0	1	70.0	1	*	*	94.0	36
	K-NN	80.0	1	84.0	8	84.0	1	84.0	3	*	*	100	11

From the results reported in Table 21.6, it is seen that out of a total of 14 cases, the MRMS method achieves 100% classification accuracy in 8 cases, while quick reduct algorithm attains this accuracy in 1 case. However, the method proposed by Valdes and Barton attains the same K-NN accuracy for colon cancer and quick reduct algorithm attains the same K-NN accuracy for prostate cancer as that of the MRMS method with lesser number of genes. The better performance of the MRMS feature selection algorithm is achieved due to the fact that it can identify relevant and significant genes from microarray data sets more accurately than the existing algorithms.

21.7 Conclusion

In this chapter, a rough set-based feature selection algorithm is presented. It selects a set of features from a high-dimensional data set by maximizing the relevance and significance of the selected features. A theoretical analysis is reported to justify the use of both relevance and significance criteria for selecting a reduced feature set with high predictive accuracy. The importance of rough set theory for computing both relevance and significance of the features is also established.

The effectiveness of the algorithm, along with a comparison with other related methods, is demonstrated on three QSAR data sets using the R^2 statistic of support vector regression method, and on five cancer and two arthritis microarray data sets by using the predictive accuracy of the K-nearest neighbor rule and support vector machine. All the results reported in this chapter establish the fact that an optimal subset of features can be obtained using the rough set-based feature selection algorithm by maximizing both relevance and significance of the selected features, which has a significant relation with the class label as well as low redundancy among the selected features, and thus has a property of generating more generalized classifiers.

References

1. Alon, U., Barkai, N., Notterman, D.A., Gish, K., Ybarra, S., Mack, D., Levine, A.J.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proceedings of the National Academy of Science, USA 96(12), 6745–6750 (1999)
2. Amat, L., Besalu, E., Dorca, R.C.: Identification of active molecular sites using quantum-self-similarity matrices. Journal of Chemical Information and Computer Science 41, 978–991 (2001)
3. Bazan, J., Skowron, A., Synak, P.: Dynamic Reducts as a Tool for Extracting Laws from Decision Tables. In: Raś, Z.W., Zemankova, M. (eds.) ISMIS 1994. LNCS (LNAI), vol. 869, pp. 346–355. Springer, Heidelberg (1994)
4. Bjorvand, A., Komorowski, J.: Practical applications of genetic algorithms for efficient reduct computation. In: Proceedings of the 15th IMACS World Congress on Scientific Computation, Modeling and Applied Mathematics, vol. 4, pp. 601–606 (1997)
5. Bravi, G., Gancia, E., Mascagni, P., Pegna, M., Todeschini, R., Zaliani, A.: MS-WHIM: New 3D theoretical descriptors derived from molecular surface properties: A comparative 3D QSAR study in a series of steroids. Journal of Computer-Aided Molecular Design 11, 79–92 (1997)
6. Chen, H., Zhou, J., Xie, G.: PARM: A genetic algorithm to predict bioactivity. Journal of Chemical Information and Computer Science 38, 243–250 (1998)
7. Chen, K.H., Raś, Z.W., Skowron, A.: Attributes and rough properties in information systems. International Journal of Approximate Reasoning 2, 365–376 (1988)
8. Chouchoulas, A., Shen, Q.: Rough set-aided keyword reduction for text categorisation. Applied Artificial Intelligence 15, 843–873 (2001)
9. Cornelis, C., Jensen, R., Martin, G.H., Ślęzak, D.: Attribute selection with fuzzy decision reducts. Information Sciences 180, 209–224 (2010)
10. Devijver, P.A., Kittler, J.: Pattern Recognition: A Statistical Approach. Prentice-Hall, Englewood Cliffs (1982)
11. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification and Scene Analysis. John Wiley and Sons, New York (1999)
12. Fang, J., Busse, J.W.G.: Mining of MicroRNA Expression Data—A Rough Set Approach. In: Wang, G.-Y., Peters, J.F., Skowron, A., Yao, Y. (eds.) RSKT 2006. LNCS (LNAI), vol. 4062, pp. 758–765. Springer, Heidelberg (2006)
13. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science 286(5439), 531–537 (1999)

14. Gordon, G.J., Jensen, R.V., Hsiao, L.L., Gullans, S.R., Blumenstock, J.E., Ramaswamy, S., Richards, W.G., Sugarbaker, D.J., Bueno, R.: Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. Cancer Research 62, 4963–4967 (2002)
15. Gruzdz, A., Ihnatowicz, A., Ślęzak, D.: Interactive gene clustering - A case study of breast cancer microarray data. Information Systems Frontiers 8, 21–27 (2006)
16. Han, J., Kamber, M.: Data Mining, Concepts and Techniques. Morgan Kaufmann Publishers (2001)
17. Inuiguchi, M., Yoshioka, Y., Kusunoki, Y.: Variable-precision dominance-based rough set approach and attribute reduction. International Journal of Approximate Reasoning 50, 1199–1214 (2009)
18. Jain, A.N., Koile, K., Chapman, D.: Compass: Predicting biological activities from molecular surface properties. Performance comparisons on a steroid benchmark. Journal of Medicinal Chemistry 37, 2315–2327 (1994)
19. Jensen, R., Shen, Q.: Semantics-preserving dimensionality reduction: Rough and fuzzy-rough-based approach. IEEE Transactions on Knowledge and Data Engineering 16(12), 1457–1471 (2004)
20. Jiang, D., Tang, C., Zhang, A.: Cluster analysis for gene expression data: A survey. IEEE Transactions on Knowledge and Data Engineering 16(11), 1370–1386 (2004)
21. Katritzky, A.R., Lobanov, V., Karelson, M.: Comprehensive descriptors for structural and statistical analysis version 1.1. University of Florida (1994)
22. Kim, D.: Data classification based on tolerant rough set. Pattern Recognition 34(8), 1613–1624 (2001)
23. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artificial Intelligence 97(1-2), 273–324 (1997)
24. Koller, D., Sahami, M.: Toward optimal feature selection. In: Proceedings of the International Conference on Machine Learning, pp. 284–292 (1996)
25. Komorowski, J., Pawlak, Z., Polkowski, L., Skowron, A.: Rough sets: A tutorial. In: Pal, S., Skowron, A. (eds.) Rough-Fuzzy Hybridization: A New Trend in Decision Making, pp. 3–98. Springer, Singapore (1999)
26. Kudo, Y., Murai, T., Akama, S.: A Granularity-based framework of deduction, induction, and abduction. International Journal of Approximate Reasoning 50(8), 1215–1226 (2009)
27. Leach, A.R.: Molecular Modelling: Principles and Applications, vol. 2. Prentice-Hall (2001)
28. Li, J., Su, H., Chen, H., Futscher, B.W.: Optimal search-based gene subset selection for gene array cancer classification. IEEE Transactions on Information Technology in Biomedicine 11(4), 398–405 (2007)
29. Li, Z.R., Han, L.Y., Xue, Y., Yap, C.W., Li, H., Jiang, L., Chen, Y.Z.: MODEL – Molecular descriptor lab: A Web-based server for computing structural and physicochemical features of compounds. Biotechnology and Bioengineering 97, 389–396 (2007)
30. Liu, S.S., Yin, C.S., Li, Z.L., Cai, S.X.: QSAR study of steroid benchmark and dipeptides based on MEDV-13. Journal of Chemical Information and Computer Science 41, 321–329 (2001)
31. Liu, X., Krishnan, A., Mondry, A.: An entropy based gene selection method for cancer classification using microarray data. BMC Bioinformatics 6(76), 1–14 (2005)
32. Maji, P.: f-Information measures for efficient selection of discriminative genes from microarray data. IEEE Transactions on Biomedical Engineering 56(4), 1063–1069 (2009)
33. Maji, P., Paul, S.: Rough sets for selection of molecular descriptors to predict biological activity of molecules. IEEE Transactions on System, Man and Cybernetics, Part C, Applications and Reviews 40(6), 639–648 (2010)
34. Maji, P., Paul, S.: Rough set based maximum relevance-maximum significance criterion and gene selection from microarray data. International Journal of Approximate Reasoning 52(3), 408–426 (2011)

35. Modrzejewski, M.: Feature selection using rough sets theory. In: Proceedings of the 11th International Conference on Machine Learning, pp. 213–226 (1993)
36. Napolitano, F., Raiconi, G., Tagliaferri, R., Ciaramella, A., Staiano, A., Miele, G.: Clustering and visualization approaches for human cell cycle gene expression data analysis. International Journal of Approximate Reasoning 47, 70–84 (2008)
37. van der Pouw Kraan, T.C.T.M., Kraan, T.C.T.M., van Gaalen, F.A., Kasperkovitz, P.V., Verbeet, N.L., Smeets, T.J.M., Kraan, M.C., Fero, M., Tak, P.P., Huizinga, T.W.J., Pieterman, E., Breedveld, F.C., Alizadeh, A.A., Verweij, C.L.: Rheumatoid arthritis is a heterogeneous disease: Evidence for differences in the activation of the STAT-1 pathway between rheumatoid tissues. Arthritis and Rheumatism 48(8), 2132–2145 (2003)
38. van der Pouw Kraan, T.C.T.M., Wijbrandts, C.A., van Baarsen, L.G.M., Voskuyl, A.E., Rustenburg, F., Baggen, J.M., Ibrahim, S.M., Fero, M., Dijkmans, B.A.C., Tak, P.P., Verweij, C.L.: Rheumatoid arthritis subtypes identified by genomic profiling of peripheral blood cells: Assignment of a type I interferon signature in a subpopulation of pateints. Annals of the Rheumatic Diseases 66, 1008–1014 (2007)
39. Parthalain, N.M., Shen, Q.: Exploring the boundary region of tolerance rough sets for feature selection. Pattern Recognition 42(5), 655–667 (2009)
40. Pawlak, Z.: Rough Sets, Theoretical Aspects of Resoning About Data. Kluwer, Dordrecht (1991)
41. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(8), 1226–1238 (2005)
42. Polanski, J., Walczak, B.: The comparative molecular surface analysis (COMSA): a novel tool for molecular design. Computers and Chemistry 24, 615–625 (2000)
43. Robert, D., Amat, L., Carbo-Dorca, R.: Three-dimensional quantitative structure-activity relationships from tuned molecular quantum similarity measures: Prediction of the corticosteroid-binding globulin binding affinity for a steroid family. Journal of Chemical Information and Computer Sciences 39, 333–344 (1999)
44. Robinson, D.D., Winn, P., Lyne, P., Richards, W.: Self-organizing molecular field analysis: A tool for structure-activity studies. Journal of Medicinal Chemistry 42, 573–583 (1999)
45. Singh, D., Febbo, P.G., Ross, K., Jackson, D.G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A.A., D'Amico, A.V., Richie, J.P., Lander, E.S., Loda, M., Kantoff, P.W., Golub, T.R., Sellers, W.R.: Gene expression correlates of clinical prostate cancer behavior. Cancer Research 1, 203–209 (2002)
46. Skowron, A., Rauszer, C.: The discernibility matrices and functions in information systems. In: Słowiński, R. (ed.) Intelligent Decision Support, pp. 331–362. Kluwer Academic Publishers, Dordrecht (1992)
47. Skowron, A., Świniarski, R.W., Synak, P.: Approximation Spaces and Information Granulation. In: Peters, J.F., Skowron, A. (eds.) Transactions on Rough Sets III. LNCS, vol. 3400, pp. 175–189. Springer, Heidelberg (2005)
48. Ślęzak, D.: Approximate reducts in decision tables. In: Proceedings of the 6th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU 1996), pp. 1159–1164 (1996)
49. Ślęzak, D., Wróblewski, J.: Roughfication of Numeric Decision Tables: The Case Study of Gene Expression Data. In: Yao, J., Lingras, P., Wu, W.-Z., Szczuka, M.S., Cercone, N.J., Ślęzak, D. (eds.) RSKT 2007. LNCS (LNAI), vol. 4481, pp. 316–323. Springer, Heidelberg (2007)
50. Sventik, V., Wang, T., Tong, C., Liaw, A., Sheridan, R.P., Song, Q.: Boosting: An ensemble learning tool for compound classification and QSAR modeling. Journal of Chemical Information and Modeling 45(3), 786–799 (2005)
51. Tuppurainen, K., Viisas, M., Laatikainen, R., Peräkylä, M.: Evaluation of a novel electronic eigenvalue (EEVA) molecular descriptor for QSAR/QSPR studies: Validation using a benchmark steroid data set. Journal of Chemical Information and Computer Sciences 42, 607–613 (2002)

52. Turner, D.B., Willett, P., Ferguson, A.M., Heritage, T.W.: Evaluation of a novel molecular vibration-based descriptor (EVA) for QSAR studies: 2. Model validation using a benchmark steroid dataset. Journal of Computer-Aided Molecular Design 13, 271–296 (1999)
53. Valdés, J.J., Barton, A.J.: Relevant Attribute Discovery in High Dimensional Data: Application to Breast Cancer Gene Expressions. In: Wang, G.-Y., Peters, J.F., Skowron, A., Yao, Y. (eds.) RSKT 2006. LNCS (LNAI), vol. 4062, pp. 482–489. Springer, Heidelberg (2006)
54. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, New York (1995)
55. West, M., Blanchette, C., Dressman, H., Huang, E., Ishida, S., Spang, R., Zuzan, H., Olson, J.A., Marks, J.R., Nevins, J.R.: Predicting the clinical status of human breast cancer by using gene expression profiles. Proceedings of the National Academy of Science, USA 98(20), 11,462–11,467 (2001)
56. Xie, G., Zhang, J., Lai, K., Yu, L.: Variable precision rough set for group decision-making: an application. International Journal of Approximate Reasoning 49, 331–343 (2008)
57. Yao, Y.: Probabilistic rough set approximations. International Journal of Approximate Reasoning 49(2), 255–271 (2008)

Chapter 22
Towards Logics of Some Rough Perspectives of Knowledge

A. Mani

Abstract. Pawlak had introduced a concept of knowledge as a state of relative exactness in classical rough set theory (RST) [30]. From a theory of knowledge and application perspective, it is of much interest to study concepts of relative consistency of knowledge, correspondences between evolvents of knowledges and problems of conflict representation and resolution. Semantic frameworks for dealing with these are introduced and developed in this research paper by the present author. New measures that deal with different levels of contamination are also proposed. Further, it is shown that the algebraic semantics are computationally very amenable. The proposed semantics would also be of interest for multi-agent systems, dynamic spaces and collections of general approximation spaces. Part of the literature on related areas is also critically surveyed.

Keywords: Knowledge interpretation, concepts, rough sets, algebraic semantics, contamination problem, granular measures of knowledge consistency, power knowledge algebras.

22.1 Introduction

In Pawlak's concept of knowledge in classical RST [30], if $\underline{S}$ is a set of attributes and P an indiscernibility relation on it, then sets of the form A^l and A^u

A. Mani
Department of Pure Mathematics
University of Calcutta
9/1B, Jatin Bagchi Road, Kolkata (Calcutta)-700029, India
e-mail: a.mani.cms@gmail.com
Homepage: http://www.logicamani.co.cc

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 419–444.
springerlink.com

represent clear and definite concepts. If Q is another stronger equivalence ($Q \subseteq P$) on $\underline{S}$, then the state of the knowledge encoded by $\langle \underline{S}, Q \rangle$ is a *refinement* of that of $S = \langle \underline{S}, P \rangle$. Subsequent work on logics and semantics for comparing different types of knowledge and measures of relative consistency have been very limited.

The knowledge interpretation has been extended in a natural way to another general approximation space-based RST (including TAS) [24]. Relative this interpretation any non-degenerate use of choice operations over granules in the more general cases for the construction of *definite* objects (in any of the rough senses) can be seen to result in clear concepts or beliefs with ontology. In general, the distinction between belief and knowledge cannot be made without additional information about the evolution of the context and applicable concepts of justification. So this applies to knowledge as well.

Measures for consistency between two knowledges have been introduced in [5] and a related problem of forming a logic for the same has been proposed. In [25], these have been generalised from a different paradigm. The resulting measures contain more information than the measures of [5] if viewed from a classical perspective. These are surveyed in some detail and the latter is improved in this paper. Algebraic logics for all of the situations are introduced and it is also shown that the algebras are also very amenable from the computation point of view.

Some of the general aspects that may qualify different approaches related to knowledge interpretation of RST are as below.

1. Knowledge of multiple agents (real or abstract).
2. Comparability of knowledges.
3. Consistency of two or more knowledges.
4. Knowledges with special emphasis on properties generated by derived indiscernibility predicates.
5. Knowledges that have special relationship with subsets of attributes
6. Algebraic logics for the context.
7. Modal/multi-modal logics for the context.
8. Relationship of knowledge with multiple concepts of rough equalities.
9. Contamination problem.

The first, second and seventh along with application domain-based aspects have been the main motivations for the divergent approaches of [10, 12], [17, 31, 32] and [29]. Corresponding to a set τ of 'terms', multi-agent systems of the form $\langle S, (R_t)_{t\in\tau} \rangle$ are considered (with S being a set and R_t being an equivalence on S for each $t \in \tau$) in [31]. If R_a, R_b are, respectively, the 'knowledge bases' of agents a, b , then weak and strong distributed knowledge bases ($R_{a\vee b}$, $R_{a\wedge b}$, respectively) derived from them are defined via

- $(\forall a,b \in \tau) U|R_{a\vee b} = \{[x]_{R_a} \cap [y]_{R_b}; [x]_{R_a} \cap [y]_{R_b} \neq \emptyset\}$
- $(\forall a,b \in \tau) U|R_{a\wedge b} = \{[x]_{R_a} \cup [y]_{R_b}; [x]_{R_a} \cap [y]_{R_b} \neq \emptyset\}$

In these, it is assumed that the set of agents is finite. In [17] and related papers,the primary focus is on four different approximations generated by the set of agents. In [10,12], the focus is on collections of approximation spaces. Concepts of consistency of knowledges in different senses and contexts concerning conflicting information would be still relevant in applications of collections of approximation spaces and multi-agent distributive systems. In all of these approaches, the relation between possible rough equalities and indiscernibility relations involved does not play a major role.

Preliminary connections between the knowledge interpretation and set of all rough equalities on a given set can, in retrospect, be traced to [27] where it is shown that an equivalence on a power set is a rough equality precisely when the union of the singleton classes contained in its quotient forms a Boolean subalgebra under induced operations of the natural Boolean algebra on the power set. Section 10 of [15] has more information on the relation between rough equalities and equivalences. The nature of rough equalities in partially ordered approximation spaces [19] will be relevant in the context of the sixth class of problems mentioned above.

Apart from the fourth and seventh, all of the nine aspects mentioned are among the primary motivations of the present paper. The contamination problem is particularly relevant in applications to human reasoning contexts. The primary goal of the present paper is towards forming logics that can represent any concept that can be expressed in terms of granules in a rough perspective of knowledge. This includes concepts of relative consistency of knowledge, correspondences between granules and conflict representation. Eventually, we intend to deal with knowledge generated by arbitrary subsets of relations subject to attribute-level constraints. Another goal is to deal with inverse problems of forming *rough evolutions of knowledge* in applications to human learning contexts – where applications of RST are hindered by its more common measures. The connection between classical RST and concept lattices has been examined by different authors in the literature. But, this will not be taken up in the present paper for reasons of space.

In the next section, some background, a summary of part of earlier work and basic results is presented. In the third section, the nature of semantic domains, meta levels, granules and granulations is clarified. In the following section, the contamination problem and a method of counting is presented. These are used in the introduction of new generalised measures in the fifth section. In the sixth section, basic knowledge algebras are introduced and characterized. Power knowledge algebras are introduced in the next section. Computational aspects also taken up in the same section. Abstract aspects of knowledge granularity are taken up in the penultimate section.

22.2 Some Background and Terminology

The set of all equivalence, tolerance and reflexive relations on a set S will be denoted by $EQ(S)$, $Tol(S)$ and $Ref(S)$, respectively. For any $\rho, \sigma \in EQ(S)$, if we define the operations $\wedge, \vee$ as per

$$\sigma \wedge \rho = \sigma \cap \rho \text{ and } \sigma \vee \rho = Cl(\sigma \cup \rho),$$

where Cl is the transitive closure operator, then we have:

Proposition 22.1. *$\langle EQ(S), \vee, \wedge, \Delta, 1\rangle$ is an algebraic lattice with the least element Δ being the diagonal of S and the greatest element $1 = S^2$. If S is finite, then $EQ(S)$ is atomistic (A lattice is atomistic if it is generated by its atoms).*

Definition 22.1. A non-empty subset H of a poset P is said to be *directed* if and only if $(\forall x, y \in H)(\exists z \in P)x \leq z \& y \leq z$. If every directed subset of a poset has a least upper bound, then it is said to be a *directed complete partial order*.

It is well known that:

Theorem 22.1. *If P is an algebraic lattice, then*

1. *the set of compact elements $K(P)$ is a join semilattice with least element in the induced order;*
2. *P is order-isomorphic to the lattice of order ideals of the compact elements, that is $P \cong \mathcal{I}(K(P))$. The associated map is $: x \mapsto x\downarrow \cap K(P)$ ($x\downarrow$ being the principal order ideal generated by x).*

Theorem 22.2. *If L is a join-semilattice with least element, then*

1. *the collection $\mathcal{I}(L)$ of all order-ideals of L is an algebraic lattice (the join corresponds to set union);*
2. *L is order-isomorphic to $K(\mathcal{I}(L))$. The associated map is $: x \mapsto x\downarrow$.*

Definition 22.2. A collection $\mathcal{C}$ of subsets of a set A that is closed under the intersection of arbitrary subcollections is called a *closure system* over A. If $\mathcal{C}$ is also closed under the union of subcollections that are (upward) directed (under set inclusion), then it is said to be an *algebraic closure system* (or an algebraic closed-set system).

An algebraic closure system $\mathcal{C}$ forms an algebraic lattice under the set-theoretic inclusion order. The elements of $\mathcal{C}$ are also called filters. If $\mathcal{C}$ is an algebraic closure system over a non-empty set A and if B is a non-empty subset of A, then $\{F \cap B : F \in \mathcal{C}\}$ is also an algebraic closure system over B. If $h : B \longmapsto A$ is a map, then $h^{-1}(C) := \{h^{-1}(F) : F \in \mathcal{C}\}$ is an algebraic closure system over B.

Definition 22.3. A cardinal κ is *inaccessible* if and only if all of the following hold:

1. $\aleph_0 < \kappa$
2. $(n < \kappa \longrightarrow 2^n < \kappa)$
3. If L is a set of cardinals such that $Card(L) < \kappa$ and $(\forall n \in L)n < \kappa$, then $Sup\{n; n \in L\} < \kappa$.

Definition 22.4. A *Partial Algebra P* is a tuple of the form

$$\langle \underline{P}, f_1, f_2, \ldots, f_n, (r_1, \ldots, r_n) \rangle$$

with $\underline{P}$ being a set, f_i's being partial function symbols of arity r_i. The interpretation of f_i on the set $\underline{P}$ should be denoted by $f_i^{\underline{P}}$, but the superscript will be dropped in this paper as the application contexts are simple enough. If predicate symbols enter into the signature, then P is termed a *Partial Algebraic System*. See [4,18] for the basic theory.

In a partial algebra, for term functions p, q, the *weak* equality is defined via,

$$p \stackrel{\omega}{=} q \text{ iff } (\forall x \in dom(p) \cap dom(q))\, p(x) = q(x).$$

The *weak-strong* equality is defined via,

$$dom(p) = dom(q), \& p \stackrel{\omega^*}{=} q \text{ iff } (\forall x \in dom(p))\, p(x) = q(x).$$

For two terms s, t, $s \stackrel{\omega}{=} t$ shall mean, if both sides are defined, then the two terms are equal (the quantification is implicit). $s \stackrel{\omega^*}{=} t$ shall mean if either side is defined, then the other is and the two sides are equal (the quantification is implicit).

Definition 22.5. A *t-norm* is a function $t : [0,1]^2 \mapsto [0,1]$ that satisfies all of (Note that we omit initial quantifiers uniformly):

1. $t(a,1) = a$ (this is the same as $(\forall a)t(a,1) = a$).
2. $(b \leq c \longrightarrow t(a,b) \leq t(a,c)); t(a,b) = t(b,a)$
3. $t(a,t(b,c)) = t(t(a,b),c); t(a,0) = 0$

An *s-norm* $s : [0,1]^2 \mapsto [0,1]$ is a function for which there exists a t-norm t such that:

$$(\forall a,b)s(a,b) = 1 - t(1-a, 1-b).$$

22.2.1 $EQ(S)$ and Approximations

The behaviour of rough approximations relative to the order structure of $EQ(S)$ is considered in this subsection. Subscripts as in $l_\sigma, l_{\sigma\vee\rho}, l_{\sigma\wedge\rho}$ will be used to indicate lower approximations relative σ, $\sigma \vee \rho$ and $\sigma \wedge \rho$, respectively, and so on.

Theorem 22.3. $(\forall A \subseteq S)(\forall \sigma, \rho \in EQ(S))$

$$A^{l_\rho} \cap A^{l_\sigma} \subseteq A^{l_\rho} \cup A^{l_\sigma} \subseteq A^{l_{\rho\wedge\sigma}} \subseteq A^{u_{\rho\wedge\sigma}} \subseteq A^{u_\sigma} \cap A^{u_\rho} \subseteq A^{u_\sigma} \cup A^{u_\rho}.$$

Proof. For each $x \in S$, the granule $[x]_{\rho\wedge\sigma}$ generated relative $\rho \wedge \sigma$ is the same as the intersections of the granules generated relative ρ and σ separately, that is

$$[x]_{\rho\wedge\sigma} = [x]_\rho \cap [x]_\sigma.$$

So, $A^{l_{\rho\wedge\sigma}} = \bigcup\{[x]_\rho \cap [x]_\sigma : [x]_\rho \cap [x]_\sigma \subseteq A\}$. This set contains $A^{l_\rho} \cup A^{l_\sigma}$. Similarly, the other parts can be checked.

Proposition 22.2. *For each $x \in S$,*

$$[x]_\rho \cup [x]_\sigma \subseteq [x]_{\rho\vee\sigma}.$$

Proof. If $y \in [x]_\rho$, then $y \in [x]_{\rho\vee\sigma}$, but as the transitive closure of $\rho \cup \sigma$ is $\rho \vee \sigma$, it is possible that $z \in [x]_{\rho\vee\sigma}$ and $z \notin [x]_\rho \cup [x]_\sigma$. So $[x]_\rho \cup [x]_\sigma \subseteq [x]_{\rho\vee\sigma}$.

Because of the above results, it is not easy to reduce the study of all equivalences to a study of the approximations without taking recourse to higher order constructions. Naturally for other kinds of relations, granules and granulations, the situation becomes more complex.

22.2.2 Relative Consistency of Knowledge

In this subsection, the approach of [5, 6] on consistency of knowledges is summarized. The authors assume that:

1. S is finite ([5, 30]).
2. Knowledge is characterized by granules ([5, 30]).
3. Knowledge K_1 is *fully consistent* with another knowledge K_2 if and only if both generate the same granules [5].
4. Knowledge K_1 is *fully inconsistent* with another knowledge K_2 if and only if no granule of one is included in a granule of the other [5].
5. (Apparently) granules need not be restricted in any special way.

The finiteness assumption amounts to permitting only a finite number of relevant attributes, granules and finite number of viewpoints. Such a position may not always be useful from practical points of view especially in contexts where such cannot be predicted because of apparent randomness or indeterminacy. From a semantic point of view, the finite and infinite situations are very different – this is explained in the section on algebraic semantics.

According to Pawlak's approach [30] to theories of knowledge from a classical rough perspective, if $\underline{S}$ is a set of attributes and R an indiscernibility relation on it, then sets of the form A^l and A^u represent clear and definite concepts. If Q is another stronger equivalence ($Q \subseteq R$) on $\underline{S}$, then the state of the knowledge encoded by $\langle \underline{S}, Q\rangle$ is a *refinement* of that of $S = \langle \underline{S}, R\rangle$. The R-positive region of Q is defined to be

$$POS_R(Q) = \bigcup_{X \in S|Q} X^{l_R} : \ X^{l_R} = \bigcup \{[y]_R; [y]_R \subseteq X\}.$$

The degree of dependence of knowledge Q on R, $\delta(Q,R)$, is defined by

$$\delta(Q,R) = \frac{Card(POS_R(Q))}{Card(S)}.$$

As the set S is common to all of the considerations, knowledges of the form $\langle S, R\rangle$ can be abbreviated by R.

Definition 22.6. In the notation, $P \preceq Q$ if and only if $(\forall x)\,[x]_P \subseteq [x]_Q$.

Definition 22.7. In the above context, P depends on Q to a degree k or in symbols $P \multimap_k Q$ if and only if $k = \delta(P,Q)$.

Proposition 22.3. *In the above context, all of the following hold:*

1. $(P \multimap Q \,\&\, P \preceq R \longrightarrow R \multimap Q)$
2. $(P \multimap Q \,\&\, R \preceq Q \longrightarrow P \multimap R)$
3. $(P \multimap Q \,\&\, Q \multimap R \longrightarrow P \multimap R)$
4. $(P \multimap R \,\&\, Q \multimap R \longrightarrow P \cap Q = R)$
5. $(P \multimap R \cap Q \longrightarrow P \multimap R \,\&\, P \multimap Q)$
6. $(P \multimap Q \,\&\, R \multimap T \longrightarrow P \cap R \multimap Q \cap T)$

Proposition 22.4. *In the above context, all of the following hold:*

1. $(Q \preceq P \longrightarrow X^{l_P} \subseteq X^{l_Q})$
2. $(P \multimap_a Q \,\&\, R \preceq P \longrightarrow (\forall a \leq b) R \multimap_b Q)$
3. $(P \multimap_a Q \,\&\, P \preceq R \longrightarrow (\forall b \leq a) R \multimap_b Q)$
4. $(P \multimap_a Q \,\&\, R \preceq Q \longrightarrow (\forall b \leq a) P \multimap_b R)$
5. $(P \multimap_a Q \,\&\, Q \preceq R \longrightarrow (\forall a \leq b) P \multimap_b R)$.

Proposition 22.5. *In the above context, all of the following hold:*

1. $(R \multimap_a P \,\&\, Q \multimap_b P, \longrightarrow (\exists c \max(a,b) \leq c) R \cap Q \multimap_c P)$
2. $(R \cap P \multimap_a Q \longrightarrow (\exists b, c \leq a) R \multimap_b Q \,\&\, P \multimap_c Q)$
3. $(R \multimap_a Q \,\&\, R \multimap_b P \longrightarrow (\exists c \leq \min(a,b)) R \multimap_c Q \cap P)$
4. $(R \multimap_a Q \cap P \longrightarrow (\exists a \leq b, c) R \multimap_b Q \,\&\, R \multimap_c P)$
5. $(R \multimap_a P \,\&\, P \multimap_b Q \longrightarrow (\exists c \geq a + b - 1) R \multimap_c Q)$.

Definition 22.8. In [5], the *consistency degree* between two knowledges P, Q, when $P \multimap_a Q$, $Q \multimap_b P$ is defined by

$$Cons(P,Q) = \frac{a + b + nab}{n + 2}$$

for a constant $n \in Z_+$.

Proposition 22.6. *If*

$$C(a,b) = \frac{a+b+nab}{n+2},$$

then the function $\delta(a,b) = 1 - C(1-a, 1-b)$ *is an s-norm.*

The above concept is extended to four types of cover-based RST in the same way without any constraints on the granules in [6] and the preliminary nature of the study is admitted. Based on the perspective of the contamination problem, more general measures for the purpose have been proposed by the present author in [25]. Disjointness of granules at a local level has been an important requirement in the knowledge interpretation in [23, 24]. These aspects including the contamination problem are taken up in subsequent sections.

22.3 Semantic Domains and Granules

In classical RST (see [30]), an approximation space is a pair of the form $\langle S, R\rangle$, with R being an equivalence on the set S. On the power set $\wp(S)$, lower and upper approximation operators, apart from the usual Boolean operations, are definable. The resulting structure constitutes a semantics for RST (though not satisfactory) in a classicalist perspective. This continues to be true even when R is some other type of binary relation. More generally (see fourth section), it is possible to replace $\wp(S)$ by some set with a part-hood relation and some approximation operators defined on it. The associated semantic domain in the sense of a collection of restrictions on possible objects, predicates, constants, functions and low level operations on those will be referred to as the classical semantic domain for general RST. In contrast, the semantics associated with sets of roughly equivalent or relatively indiscernible objects with respect to this domain will be called the rough semantic domain. It is well known that various types of rough equalities can be defined using different number of approximation operators. The concept of bottom and top equalities (see [28, 35]) can also be relevant from a semantic perspective. Many other semantic domains including hybrid semantic domains can be generated (see [20–22]) for different types of rough semantics, but these two broad domains will always be - though not necessarily with a nice correspondence between the two. In one of the semantics developed in [21], the reasoning is within the power set of the set of possible order-compatible partitions of the set of roughly equivalent elements. The concept of *semantic domain* used is, therefore, similar to the sense in which it is used in general abstract model theory [26] (though one can object to formalisation on different philosophical grounds).

The term *object level* will mean a description that can be used to directly interface with fragments (sufficient for the theories or observations under consideration) of the concrete real world. Meta levels concern fragments of theories that address aspects of dynamics at lower meta levels or the object level. Importantly, we permit meta level aspects to filter down to object levels relative a different object level

specification. So it is always possible to construct more meta levels and expressions constructed in these can be said to carry intentions.

Despite all this, two particular meta levels namely Meta-C (or Meta Classical), Meta-R (or Meta Rough) and an object level will be used for comparing key notions introduced with the more common approaches in the literature. Meta-R is the meta level corresponding to the observer or agent experiencing the vagueness or reasoning in vague terms (but without rough inclusion functions and degrees), while Meta-C will be the more usual higher order classical meta level at which the semantics is formulated. It should be noted that rough membership functions and similar measures are defined at Meta-C, but they do not exist at Meta-R. A number of meta levels placed between Meta-R and Meta-C can possibly be defined and some of these will be implicit in the section on rough naturals.

Many logics have been developed with the intent of formalising 'rough sets' as 'well-formed formulae' in a fixed language. They do not have a uniform domain of discourse and even ones with category theoretically equivalent models do not necessarily see the domain in the same way (though most meanings can be mapped in a bijective sense). For example, the regular double Stone algebra semantics and complete rough algebra semantics correspond to different logical systems of classical RST (see [2,3]). The super rough algebra semantics in [20] actually adds more to the rough algebra semantics of [1]. It is possible to express the ability of objects to approximate in the former, while this is not possible in the latter. This is the result of a higher order construction used for generating the former.

22.3.1 Granules

Granules are essentially the objects that can be seen to construct approximations at Meta-C. At Meta-R they may or may not be perceived as objects. Other features like atomicity, which are more commonly associated with granules may or may not be true in general. In [24], a brief version of the axiomatic theory of granules by the present author along with related philosophical aspects may be found. Here we mention a few concepts.

Granulations can be seen as collections of granules that contain every object of the universe in question. The concept of *granules* actually evolves across the temporal space of the construction of a theory and may be essentially *a priori* or *a posteriori* (relative the theory or semantics) in nature. Possible concepts of granules are also naturally dependent on the choice of semantic domain in the contexts of RST and 'a priori' granules may even be identified at some stage after the identification of approximations.

A particular tolerance space (for example) can have multiple admissible granulations — the same approximation can be viewed as being constructed out of different sets of granules (see [21] for example). We state this to stress that the definition of knowledge should be based on the granulation and not on that of the relation alone. Finer aspects of the definition of knowledge with respect to granulation are taken up in the penultimate section.

22.4 Contamination Problem and IPC

Suppose the problem at hand is to model vague reasoning in a particular context and relative the agents involved in the context. It is natural for the model to become *contaminated* with additional inputs from the classicalist perspective imposed on the context by the person doing the modelling. In other words, meta level aspects can contaminate the perception of object level features. From an algebraic perspective, if the model concerns objects of specific types like 'roughly equivalent objects in some sense' then the situation is relatively better than a model that involves all types of objects. But the operations used in the algebra or algebraic system can still be viewed with suspicion.

By the contamination problem, I mean the problem of minimising or eliminating the influences of the classicist perspective imposed on the context. In other words, the problem is of minimising the contamination of models of meta-R fragments by meta-C aspects. One aspect of the problem is solved partially in [25] by the present author. In the paper, a more realistic conception of rough membership functions and other measures of RST have been advanced from a minimalist perspective avoiding the real numbers or rationals based rough measures that dominate the rough literature. Most of the rough measures based on cardinalities are of course known to lack mathematical rigour and have the potential to distort analysis.

In the mathematics of exact phenomena, the natural numbers arise in the way they do precisely because it is assumed that things being counted are well defined and have exact existence. When a concrete collection of identical balls on a table is being counted, then it is their relative position on the table that helps in the process. But there are situations in real life, where

- such identification may not be feasible,
- the number assigned to one may be forgotten while counting subsequent objects,
- the concept of identification by way of attributes may not be stable,
- the entire process of counting may be 'lazy' in some sense,
- the mereology necessary for counting may be insufficient.

To count a collection of objects in the usual sense, it is necessary for them to be distinct and well defined. So a collection of well-defined distinct objects and indiscernible objects can be counted in the usual sense from a higher meta level of perception. Relative to this meta level, the collection must appear as a set. In the counting procedures developed, the use of this meta level is minimal and certainly far lesser than in other approaches. It is dialectical as two different interpretations are used simultaneously to complete the process. These two interpretations cannot be merged as they have contradictory information about relative discernibility. Though the classical interpretation is at a relatively higher meta level, it is still being used in the counting process and the formulated counting is not completely independent of the classical interpretation.

A strictly formal approach to these aspects and a system of rough naturals will be part of a forthcoming paper. The concept of *Rough Y-System*(RYS) was introduced by the present author in [24]. As the technical aspects are not explicitly used in the present paper, it suffices to consider RYS *as some subset of a power set along with multiple lower and upper approximation operators, inclusion operation and a granule indicating operation. The concept of a* rough object *is intended to be expressed by sets of approximations that cover the inexact object in a order-theoretic sense*.

Counting of a set of objects of an approximation space and that of its power set is very different as they have very different kinds of indiscernibility inherent in them. The latter possess a complete evolution for all of the indiscernibility present in it while the former does not. Counting of elements of a RYS is essentially a generalisation of the latter. In general, any lower level structure like an approximation space corresponding to a 1-neighbourhood system [36] or a cover need not exist in any unique sense. The axiomatic theory of granules developed in the previous sections provides a formal exactification of these aspects.

Let S be a RYS, with R being a derived binary relation (interpreted as a weak indiscernibility/ weak rough equality relation) on it. As the other structure on S will not be explicitly used in the following, it suffices to take S to be an at most countable list of elements : $\{x_1, x_2, \ldots, x_k, \ldots, \}$ (actually the set of axioms required to define lists is a consequence of the set of ZF axioms). Taking $(a,b) \in R$ to mean 'a is weakly indiscernible from b' concepts of *primitive counting* regulated by different types of meta level assumptions are defined below. The adjective *primitive* is intended to express the minimal use of granularity and related axioms.

Indiscernible Predecessor Based Primitive Counting (IPC)

In this form of 'counting', the relation with the immediate preceding step of counting matters crucially. s is the successor operation, while f is the operation that implements the count.

1. Assign $f(x_1) = 1_1 = s^0(1_1)$
2. If $f(x_i) = s^r(1_j)$ and $(x_i, x_{i+1}) \in R$, then assign $f(x_{i+1}) = 1_{j+1}$
3. If $f(x_i) = s^r(1_j)$ and $(x_i, x_{i+1}) \notin R$, then assign $f(x_{i+1}) = s^{r+1}(1_j)$

The 2-types of the expression $s^{r+1}(1_j)$ will be j. Successors will be denoted by the natural numbers indexed by 2-types.

The following example illustrates IPC:

Let $S = \{f,b,c,a,k,i,n,h,e,l,g,m\}$ and let R, Q be the reflexive and transitive closure of the relation

$$\{(a,b), (b,c), (e,f), (i,k), (l,m), (m,n), (g,h)\}$$

and

$$\{(a,b),(e,f),(i,k),(l,m),(m,n)\}$$

respectively. Then, $\langle S,R\rangle$ and $\langle S,Q\rangle$ are approximation spaces. The IPC procedure on the set in the given order proceeds as follows: assign 1_1 to f, then as $(b,f)\notin R$ assign 2_1 to b, next as $(c,b)\in R$ we assign 1_2 to c. Continuing in this way, we get the IPC-count (relative R) as :

$$\{1_1,2_1,1_2,1_3,2_3,1_4,2_4,3_4,1_5,2_5,1_6,2_6\}$$

Apart from examples in [25], the most glaring examples for avoiding the measures come from attempts to apply rough sets to modelling conceptual development. The 'same' concept X may depend on ten other concepts in one perspective and nine other concepts in another perspective and concepts of knowing the concept X and gradation does not admit a linear measure in general. Using one in fields like education or education research would only encourage scepticism. The quality of measures like 'impact factor' of journals provides a supportive example.

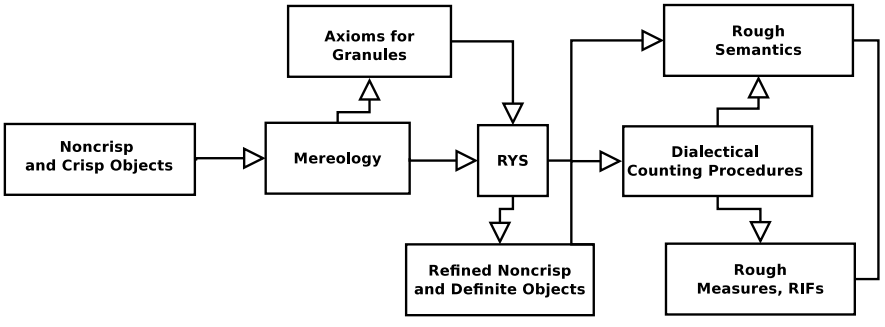
The underlying assumptions behind rough measures are much less than in a corresponding statistical approach (subject to being possible in the first place in the application context in question) and do not make presumptions of the form -'relative errors should have some regularity'. Still the contamination problem is relevant in other domains of application of RST and more so when the context is developed enough to permit an evaluation of semantic levels.

There may be differences of the semantic approach of proceeding from algebraic models to logics in sequent calculus form in comparison to the approach of directly forming the logic as a sequent calculus, or the approach of forming the logic in Kripke-like or frame-related terminology but one can expect one to feed the other. It should also be noted that this has nothing to do with supervaluationary perspectives, where the goal is to reduce vagueness by improving the language. Moreover, the primary concerns in the contamination problem are not truth values or gaps in them. The contamination problem is analogous to the problem of intuitionist philosophers to find a perfect logic free from problematic classicist assumptions. A difficult approach to the latter problem can be found in [11]. The important thing to note in [11] is the suggestion that it is better to reason within a natural deduction system to generate the 'pure logic'. In case of the contamination problem, the general understanding stems from model theoretic interpretations and so should be more appropriate.

If a model-theoretic perspective is better, then it should be expected to provide justification for the problem. The problems happen most explicitly in attempts to model human reasoning, in conceptual modelling, especially (in learning contexts) in attempts to model counting processes in the presence of vagueness and others. In applications to machine intelligence, an expression of contamination would be 'are you blaming machines without reason?'

From a blunt practical view point, I prefer to avoid using rough-theoretical measures (like membership functions), excess of meta-levels, and questionable valuations in semantic and syntactic approach till may be the last step. This modifies

the basic approach to rough semantics and is explained in detail in a forthcoming research paper. The following figure gives an idea about the orientation of components.



Dependencies Between Different Components

The link between 'Rough Semantics' and 'Rough Measures' should be read as 'possibly related'.

22.5 Generalized Measures

In [25], the following concepts were introduced:

Definition 22.9. The *granular dependence degree of knowledge Q on R*, $gk(Q,R)$ will be the tuple $(k_1,\ldots,k_r)$, with the k_i's being the ratio of the number of granules of type i included in $POS_R(Q)$ to $card(S)$.

Note that the order on S induces a natural order on the granules (classes) generated by R, Q respectively. This vector cannot be extracted from a single IPC count in general (the example in the last section should be suggestive). However, if the granulation is taken into account, then much more information (apart from the measure) can be extracted.

Proposition 22.7. *If* $gk(Q,R) = (k_1,k_2,\ldots,k_r)$, *then* $\delta(Q,R) = \sum k_i$.

The concepts of consistency degrees of multiple models of knowledge introduced in [5] can also be improved by a similar strategy:

If $\delta(Q,R) = a$ and $\delta(R,Q) = b$, then the consistency degree of Q and R, $Cons(Q,R)$ is defined by $Cons(Q,R) = \dfrac{a+b+nab}{n+2}$, where n is the consistency constant. With larger values of n, the value of the consistency degree becomes smaller.

Definition 22.10. If $gk(Q,R) = (k_1,k_2,\ldots,k_r)$ and $gk(R,Q) = (l_1,l_2,\ldots,l_p)$, then the *granular consistency degree* $gCons(Q,R)$ of the knowledge represented by Q,R will be

$$gCons(Q,R) = (k_1^*,\ldots,k_r^*,l_1^*,\ldots,l_p^*,nk_1^*l_1,\ldots nk_r^*l_p),$$

where $k_i^* = \frac{k_i}{n+2}$ for $i = 1,\ldots,r$ and $l_j^* = \frac{l_j}{n+2}$ for $j = 1,\ldots,p$.

The knowledge interpretation can be extended in a natural way to other general RST (including TAS) and also to choice inclusive rough semantics [24]. With respect to the counting procedures defined, these two general measures are relatively constructive, provided granules can be extracted. This is possible in many of the cases though not always. They can be replaced by a technique of defining atomic sub-measures based on counts and subsequently combining them. These aspects will be taken up in a future paper.

Though the measures above are proper generalisations and avoid contamination to some extent, the essence of the avoidance has not been fully achieved by $gCons(Q,R)$. $Cons(Q,S)$ is of course worse. To improve the situation, I propose a more minimalistic version.

Definition 22.11. For two knowledges R, Q, let $\mathcal{G}_R$, $\mathcal{G}_Q$ be the associated granulations (sets of granulations). In the granular perspective, $\mathcal{G}_R$ and $\mathcal{G}_Q$ can be regarded as the knowledge generators in a general rough structure like RYS. The *dependence* of Q on R will be the map : $\nabla_{RQ} : \mathcal{G}_R \longmapsto \wp(\mathcal{G}_Q)$ subject to:

$$\nabla_{RQ}(x) = \text{Max}\{y : (\forall a \in y)x \subseteq a\}.$$

Max is to be computed over the collections of collections of sets. In a RYS, $\subseteq$ can be replaced by the part-hood relation **P**.

Definition 22.12. The pair $(\nabla_{RQ}, \nabla_{QR})$ shall be called the *general consistency pair*.

These two definitions do not force the introduction of 'suspect' numbers and permit wide variety of local variations. Further, they allow a more coherent internalisation of the degree in the semantics developed in the following sections. In case R, Q are equivalences, then ∇_{RQ} would be the set of granules in the positive region.

22.6 Algebraic Semantics-1

In this section, an algebraic semantics using all of the equivalences, the knowledge order between them and approximation operators is developed through a higher order operation (*Rough equalities can be easily incorporated into both the algebras, but have not been done in the light of the contamination problem*). The basic idea of the semantic approach is an extension of the k-dimensional deductive systems of [9,13,14] to partial algebras. The main steps are:

1. Form a partial algebra S corresponding to the desired semantic processes with the intent of expressing all relevant well-formed formulas.
2. Form a set of closed endomorphisms $En(S)$ of S.
3. On S^k for a suitable integer k, define an algebraic closure system $\tau(S)$.
4. If $(\forall \sigma \in En(S))(\forall (A_1, \ldots, A_k) \in \tau(S))\,(\sigma^{-1}(A_1), \ldots, \sigma^{-1}(A_k)) \in \tau(S)$, then $\langle S, TH(S)\rangle$ is a *general k-deductive system*.
5. The consequence operator will correspond to the algebraic closure system $\tau(S)$.
6. In the present paper $k \leq 2$ will suffice, but I will not be dealing with the question of weak algebraizability in any of the general senses in the present paper because of reasons of space. The interpretation also does not fall within known versions.

It may seem that the algebraic structure of the set of all equivalences on a set should be able to express the basic aspects of the relation of knowledge consistency, but this is

Construction:

For each $\sigma \in EQ(S)$, let

$$x \cdot y = \begin{cases} x, \text{ if } (x,y) \in \sigma \\ y, \text{ otherwise} \end{cases}$$

Relative to this operation, the following theorem [16] holds:

Theorem 22.4. *$\langle S, \cdot \rangle$ is a groupoid that satisfies the following axioms (braces are omitted when the binding is to the left,* e.g. *'abc' is the same as '$(ab)c$'):*

E1 $xx = x$
E2 $x(yz) = (xy)(xz)$
E3 $xyx = x$
E4 $yzxyuz = yuz$
E5 $u(yzxy)z = uyz$

Let $\mathbb{E}$ be the variety of groupoids defined by the above five equations and let $\mathbb{E}_0$ be the variety of groupoids defined by the first three equations in the above and $xyzyx = xzyx$. All the parts of the following theorem can be found in [16]:

Theorem 22.5. *1. The **HSP** closure of the class of equivalence algebras is $\mathbb{E}$.*
2. The variety $\mathbb{E}$ is contained in the non locally finite variety $\mathbb{E}_0$.
3. The five equations mentioned above form an equational base for the variety $\mathbb{E}$.

Theorem 22.6. *The following are consequences of the defining equations of $\mathbb{E}_0$:*

1. $x(yx) = x$; $x(xy) = xy$; $(xy)y = xy$ (from **E1**, **E2**, **E3***).*
2. $x(xyz) = x(yz)$; $(xz)(yz) = xz$; $(xy)(zx) = xyzx$ (from **E1**, **E2**, **E3***).*
3. $xyzxy = xy$; $xyzyz = xyz$; $xcyzyxy = xyzy$.
4. $(xyzx)(zy) = x(zy)$; $x(yz)y = xyzy$; $(xyz)(yx) = (xzy)(zx)$; $xyzxz = xzyz$.
5. $(\forall x)(ex = ey \longrightarrow x = y)$ is equivalent to $(\forall x)xe = e$ (from **E1**, **E2**, **E3***).*

The set of all equivalence operations on a set S will be denoted by $E(S)$. Taking $\underline{K} = E(S) \times S^2$, the following partial operations can be defined on the product $\underline{K}$ (we abbreviate the quantifier $(\forall \alpha, \beta \in E(S))(\forall x, y, a, b \in S)$ by $\mathfrak{Q}$):

$$(\forall \alpha \in E(S))(\forall x, y \in S)\, u(\alpha, x, y) = (\alpha, xy, xy)$$

$$(\forall \alpha \in E(S))(\forall x, y, z, a \in S)\, (\alpha, x, y) \otimes (\alpha, z, a) = (\alpha, xz, ya)$$

$$(\mathfrak{Q})\, (\alpha, x, y) \curlywedge (\beta, a, b) = \begin{cases} (\alpha \wedge \beta, x, y) \text{ if } x = a,\ y = b \\ \text{undefined}, \quad \text{else} \end{cases}$$

$$(\mathfrak{Q})\, (\alpha, x, y) \curlyvee (\beta, a, b) = \begin{cases} (\alpha \vee \beta, x, y) \text{ if } x = a,\ y = b \\ \text{undefined}, \quad \text{else} \end{cases}$$

$$(\mathfrak{Q})\, (\alpha, x, y) \sqcup (\beta, a, b) = (\alpha \vee \beta, xa, yb)$$

$$(\mathfrak{Q})\, (\alpha, x, y) \sqcap (\beta, a, b) = (\alpha \wedge \beta, xa, yb)$$

In the definition of the operations u and $\otimes$, xy means the product corresponding to α. $(\alpha, x, y) \otimes (\beta, x, y)$ will be taken to be undefined if $\alpha \neq \beta$. In the definition of $\sqcap, \sqcup$, the products are with respect to $\alpha \wedge \beta$ and $\alpha \vee \beta$, respectively.

Definition 22.13. By a *basic knowledge algebra* will be meant a partial algebra of the form $K = \langle \underline{K}, \otimes, \curlywedge, \curlyvee, \sqcap, \sqcup, u, (2, 2, 2, 2, 2, 1) \rangle$ with $\underline{K}$ and operations being as in the above.

Proposition 22.8. *The operations above are well defined.*

Theorem 22.7. *A basic knowledge algebra K satisfies all of the following:*

1. $(\forall a \in K)\, u^2(a) = u(u(a)) = u(a)$
2. $(\forall (\alpha, x, x) \in K)\, u(\alpha, x, x) = (\alpha, x, x)$
3. $(\forall a \in K)\, a \otimes a = a$
4. $(\forall a, b \in K)\, (a \otimes b) \otimes a \overset{\omega}{=} a$
5. $(\forall a, b, c \in K)\, a \otimes (b \otimes c) \overset{\omega^*}{=} (a \otimes b) \otimes (a \otimes c)$
6. $(\forall a, b, c, e \in K)\, b \otimes c \otimes a \otimes b \otimes e \otimes c \overset{\omega^*}{=} b \otimes e \otimes c$
7. $(\forall a, b, c, e \in K)\, (e \otimes (b \otimes c \otimes a \otimes b)) \otimes c \overset{\omega^*}{=} e \otimes b \otimes c$
8. $(\forall a \in K)\, a \curlyvee a = a \,\&\, a \curlywedge a = a$
9. $(\forall a, b \in K) a \curlyvee b \overset{\omega^*}{=} b \curlyvee a \,\&\, a \curlywedge b \overset{\omega^*}{=} b \curlywedge a$
10. $(\forall a, b, c \in K)(a \curlyvee b) \curlyvee c \overset{\omega^*}{=} a \curlyvee (b \curlyvee c) \,\&\, (a \curlywedge b) \curlywedge c \overset{\omega^*}{=} a \curlywedge (b \curlywedge c)$
11. $(\forall a, b \in K)(a \curlyvee b) \curlywedge a \overset{\omega}{=} a \,\&\, (a \curlywedge b) \curlyvee a \overset{\omega}{=} a$
12. $(\forall (\alpha, x, y) \in K)(\alpha, x, y) \curlyvee (1, x, y) = (1, x, y)$, *1 corresponding to the largest equivalence on S.*
13. $(\forall (\alpha, x, y) \in K)(\alpha, x, y) \curlyvee (\Delta, x, y) = (\alpha, x, y)$, *$\Delta$ corresponding to the diagonal on S.*

14. $(\forall a,b \in K)a \sqcup a = a, a \sqcap a = a(a \sqcup b) \sqcap a = a, (a \sqcap b) \sqcup a = a$
15. $(\forall a,b,c \in K)(a \sqcup b) \sqcup (a \sqcup c) = (a \sqcup b) \sqcup c, (a \sqcap b) \sqcap (a \sqcap c) = (a \sqcap b) \sqcap c$

Proof. 1. $u(u(\alpha,x,y) = (\alpha,(xy((xy),(xy)(xy)) = (\alpha,xy,xy$ (since $xx = x$).
2. $u(\alpha,x,x) = (\alpha,xx,xx) = (\alpha,x,x)$
3. If $a = (\alpha,x,y)$, then $a \otimes a = (\alpha,xx,yy) = (\alpha,x,y)$.
4. If $(a \otimes b) \otimes b$ is defined then the first component of a and b must be the same. So the term will have the form

$$((\alpha,x,y) \otimes (\alpha,z,h)) \otimes (\alpha,x,y) = (\alpha,xzx,yhy) = (\alpha,x,y) = a.$$

5. If $(a \otimes b) \otimes c$ is defined then the first component of a, b and c must be the same. So the product will have the general form $(\alpha,x,y) \otimes ((\alpha,v,w) \otimes (\alpha,g,h)) = (\alpha,x(vg),y(wh)) = (\alpha,(xv)(xg),(yw)(yh)) = (\alpha,xv,yw) \otimes (\alpha,xg,yh)$. Similarly if $(a \otimes b) \otimes (a \otimes c)$ is defined, then we have a reversed sequence of equalities.
6. This corresponds to $xyzxwy = xwy$ in an equivalence algebra.
7. This corresponds to $u(yzxy))z = uyz$ in an equivalence algebra.
8. $a \curlyvee a = (\alpha \vee \alpha,x,y) = (\alpha,x,y) = a$.
9. If $a \curlyvee b$ is defined, then $a \curlyvee b = (\alpha \vee \beta,x,y) = (\beta \vee \alpha,x,y) = (\beta,x,y) \curlyvee (\alpha,x,y) = b \curlyvee a$. Note the symmetry of the argument in the above.
10. Similar to the above.
11. Follows from the absorptivity property of the lattice operations in $E(S)$.
12. The proof is obvious.
13. The proof is obvious.
14. If $a = (\alpha,x,y)$ and $b = (\beta,g,h)$, then $((\alpha,x,y) \sqcup (\beta,g,h)) \sqcap (\alpha,x,y) = ((\alpha \vee \beta) \wedge \alpha,xgx,yhy) = (\alpha,x,y) = a$.
15. See proof of the 5th item for the essential part.

Theorem 22.8. *The following hold in a basic knowledge algebra K:*

1. *For each $x,y \in S$, $\langle \{(\alpha,x,y) : \alpha \in E(S)\}, \curlyvee, \curlywedge \rangle$ is an algebraic lattice.*
2. $(\forall a,b \in K)(a \curlyvee b = a \curlyvee b \longrightarrow a \curlywedge b = a \sqcap b = b \sqcap a = b \curlywedge a, \& a \curlyvee b = a \sqcup b = b \sqcup a = b \curlyvee a)$
3. $(\forall a,b \in K)(a \curlywedge b = a \curlywedge b \longrightarrow a \curlywedge b = a \sqcap b = b \sqcap a = b \curlywedge a, \& a \curlyvee b = a \sqcup b = b \sqcup a = b \curlyvee a)$
4. *For each $x,y \in S$, $\rangle \{(\alpha,x,y) : \alpha \in E(S)\}, \sqcup, \sqcap \rangle$ is an algebraic lattice isomorphic to $\langle \{(\alpha,x,y) : \alpha \in E(S)\}, \curlyvee, \curlywedge \rangle$.*
5. $a \sqcup ((a \sqcup b) \sqcup c) = a \sqcup (b \sqcup c)$.

Proof. 1. $\langle E(S), \curlyvee, \curlywedge \rangle$ is isomorphic to the lattice of equivalence relations and the lattice.
2. If $a \curlyvee b = a \curlyvee b$, then $a \curlyvee b$ is defined and so the second and third components of a and b must be correspondingly equal. If $a = (\alpha,x,y)$ and $b = (\beta,g,h)$, then it is easy to check $a \wedge b = a \sqcap b = b \sqcap a$ and others.
3. Proof is similar to the above.
4. Follows from 1, 2 and 3.

5. Since $x((xy)z) = (x(xy))(xz) = (xy)(xz) = x(yz)$, therefore $a \sqcup ((a \sqcup b) \sqcup c) = a \sqcup (b \sqcup c)$.

The last property may be called *demi-associativity*. It follows that $\sqcap, \sqcup$ are almost lattice like operations. One motivation for using $E(S) \times S^2$ instead of $E(S)$ as the base of the algebra is to express S-related features of knowledge consistency. If the goal was to base it on equivalences, then a suitable deductive system would be

$$\langle E(S), \mathcal{E} \rangle,$$

where $\mathcal{E}$ is the algebraic closure system constructed out of order-filters in $E(S)$. The main problem then would be the excessive generality of the structure (an abstract representation theorem would be an issue) and its inability to express very little beyond the predicate "is consistent with". It is, however, possible to extend $\langle E(S), \mathcal{E} \rangle$ using a product with the rationals Q to form a two-valued logic.

22.6.1 Topology

Topological aspects have not been explicitly considered in the above and cannot be found in the literature on the structure of set of all equivalences or other relations on a set.

Theorem 22.9. *The algebraic closure operator α associated with the algebraic lattice $EQ(S)$ satisfies:*

$$\alpha(\emptyset) = \emptyset;\ (\forall A, B \subseteq S^2)\, \alpha(A \cup B) = \alpha(A) \cup \alpha(B).$$

That is, the operator is also topological.

Proof. $EQ(S)$ is an algebraic lattice, such that all elements are representable as joins of compact elements that are also join-irreducible. This happens because the compact elements are the atoms of EQ(S) that are obtainable by adjoining exactly one non-trivial pair to the diagonal of S. These are obviously join-irreducible.

That this much is necessary and sufficient for ensuring the topological property of α was proved recently in [34]. In our terminology, *the algebraic operator associated with an algebraic lattice L is topological if and only if every element of L is representable as the join of compact elements that are also join-irreducible*.

So the result holds.

22.7 Algebraic Semantics at Meta-C

One problem with a basic knowledge algebra is that the algebraic lattice structure of $EQ(S)$ is hidden, and $\curlyvee$ and $\curlywedge$ are not explicit lattice operations. Further approximation operations cannot be easily expressed in K. To improve the situation, it

makes sense to take the base of the algebra as $\underline{H} = E(S) \times (\wp(S))^2$. This way even approximation operators can be accommodated into the discourse.

Definition 22.14. By a *power knowledge algebra* will be meant a partial algebra of the form

$$H = \langle \underline{H}, \veebar, \overline{\wedge}, \otimes, \curlywedge, \curlyvee, \sqcap, \sqcup, \star, J, G, L, U, \theta, \top, \bot \rangle$$

of type $(2,2,2,2,2,2,2,1,1,1,1,1,1,0,0)$ with $\underline{H}$ being as in the above and the other operations being defined as below ($\mathfrak{Q}$ being as in the previous section):

1. $J(\sigma,x,y) = (\sigma,xy,xy)$; Here, xy should be understood in the global sense with respect to σ, that is $xy = \{ab : a \in x, b \in y\}$.
2. $$\theta(\sigma,x,y) = \begin{cases} (\sigma,x,y), & \text{if } x,y \text{ are singletons} \\ \text{undefined}, & \text{otherwise} \end{cases}$$
3. $$G(\sigma,x,y) = \begin{cases} (\sigma,x,y), & \text{if } x,y \text{ are granules relative } \sigma \\ \text{undefined}, & \text{otherwise} \end{cases}$$
4. $L(\sigma,x,y) = (\sigma,x^l,y^l)$, the approximations x^l, y^l being with respect to σ.
5. $U(\sigma,x,y) = (\sigma,x^u,y^u)$, the approximations x^u, y^u being with respect to σ.
6. $$(\mathfrak{Q})(\alpha,x,y) \curlywedge (\beta,a,b) = \begin{cases} (\alpha \wedge \beta,x,y) & \text{if } x = a, y = b \\ \text{undefined}, & \text{else} \end{cases}$$
7. $$(\mathfrak{Q})(\alpha,x,y) \curlyvee (\beta,a,b) = \begin{cases} (\alpha \vee \beta,x,y) & \text{if } x = a, y = b \\ \text{undefined}, & \text{else} \end{cases}$$
8. $$(\mathfrak{Q})(\alpha,x,y) \veebar (\beta,a,b) = (\alpha \vee \beta, x \cup a, y \cup b)$$
9. $$(\mathfrak{Q})(\alpha,x,y) \overline{\wedge} (\beta,a,b) = (\alpha \wedge \beta, x \cap a, y \cap b)$$
10. $(\forall \alpha \in E(S))(\forall x,y \in \wp(S))\ (\alpha,x,y)^\star = (\alpha, x^c, y^c)$
11. $$(\mathfrak{Q})(\alpha,x,y) \otimes (\beta,a,b) = \begin{cases} (\alpha,xa,yb) & \text{if } \alpha = \beta \\ \text{undefined}, & \text{else} \end{cases}$$
12. $$(\mathfrak{Q})(\alpha,x,y) \sqcup (\beta,a,b) = (\alpha \vee \beta, xa, yb)$$
13. $$(\mathfrak{Q})(\alpha,x,y) \sqcap (\beta,a,b) = (\alpha \wedge \beta, xa, yb)$$
14. $$\top = (1,S,S), \quad \bot = (0,\emptyset,\emptyset).$$

(Here, 1 corresponds to the equivalence S^2 and 0 corresponds to the smallest equivalence Δ_S).

In comparison to the basic knowledge algebra, a power knowledge algebra has a very rich structure. This is shown in the following theorems:

Theorem 22.10. *All of the following hold in a power knowledge algebra of the form* $H = \langle \underline{H}, \veebar, \overline{\wedge}, \otimes, \curlywedge, \curlyvee, \sqcap, \sqcup, \star, J, G, L, U, \theta \rangle$ *:*

1. $\langle \underline{H}, \veebar, \overline{\wedge}, \top, \bot \rangle$ *is a bounded lattice.*
2. *Given* $\alpha \in E(S)$, $\langle \{(\alpha, x, y) : x, y \in \wp(S)\}, \veebar, \overline{\wedge}, \star, \top, \bot \rangle$ *is a Boolean algebra.*
3. *Given* $(x, y) \in (\wp(S))^2$, $\langle \{(\alpha, x, y) : \alpha \in E(S)\}, \veebar, \overline{\wedge}, \top, \bot \rangle$ *is an algebraic lattice.*
4. *For each* $\alpha \in E(S)$, $\langle dom(\theta), \otimes \rangle$ *is a groupoid with the finite equational base of equivalence algebras.*
5. $(\forall a \in \underline{H})\, a^{\star\star} = a \,\&\, (L(a^\star))^\star = U(a) \,\&\, (U(a^\star))^\star = L(a)$
6. $(\forall a, b, c \in \underline{H})\, (a \curlyvee b = c \longrightarrow a \veebar b = c)$
7. $(\forall a, b, c \in \underline{H})\, (a \curlywedge b = c \longrightarrow a \overline{\wedge} b = c)$
8. $(\forall a, b \in \underline{H})\, (\theta b = b \longrightarrow G(U(b)) = U(b)$
9. $(\forall a, b \in \underline{H})\, (a \veebar b = b \longrightarrow L(a) \veebar L(b) = L(b))$
10. $(\forall a \in \underline{H})\, L(a) \veebar a = a \,\&\, L(L(a)) = L(a) \,\&\, U(a) \veebar a = U(a) \,\&\, U(U(a)) = U(a)$
11. $(\forall a \in \underline{H})\, (L(a) \curlywedge \bot = \bot \longrightarrow L(a) = a)$
12. $(\forall a \in \underline{H})\, (U(a) \curlywedge \bot = \bot \longrightarrow U(a) = a)$
13. $(\forall a, b \in \underline{H})\, L(a) \veebar L(b) \subseteq L(a \veebar b) \,\&\, L(a) \overline{\wedge} L(b) = L(a \overline{\wedge} b)$
14. $(\forall a, b \in \underline{H})\, U(a) \veebar U(b) = U(a \veebar b) \,\&\, U(a \overline{\wedge} b) \subseteq U(a) \overline{\wedge} U(b)$.

Proof. 1. The structure $\langle \underline{H}, \overline{\wedge}, \veebar, \bot, \top \rangle$ is a product of bounded lattices, so it is a bounded lattice relative the induced operations $\overline{\wedge}$, $\veebar$, $\bot$ and $\top$.

2. $\alpha \in E(S)$, $\langle \{(\alpha, x, y) : x, y \in \wp(S)\}, \veebar, \overline{\wedge}, \star, \top, \bot \rangle$ is a product of two Boolean algebras and so is a Boolean algebra.

3. The collection of all equivalence relations on a set forms an algebraic lattice with respect to lattice operations defined before. These operations are induced on the collection of all equivalence operations. So the result follows.

4. For each $\alpha \in E(S)$, the structure of $\langle dom(\theta), \otimes \rangle$ is equivalent to the product of two equivalence algebras on S. So it is an equivalence algebra with its usual equational base.

5. $\star$ acts on the last two components of elements of $\underline{H}$ to form their complements. So the property follows from known properties of complementation and approximations.

6. If $a \curlyvee b = c$, then it is necessarily defined for the pair and so the conclusion follows.

7. Proof is similar to that of the previous assertion.

8. $\theta(b) = b$ means the last two components of b are singletons. The upper approximation of singletons must be granules. So $G(U(b)) = U(b)$ follows.

9–14. The rest of the assertions follows from the basic properties of rough approximation operators.

Theorem 22.11. *The following hold in a power knowledge algebra of the form*

$$H = \langle \underline{H}, \veebar, \overline{\wedge}, \otimes, \curlywedge, \curlyvee, \sqcap, \sqcup, \star, J, G, L, U, \theta, \top, \bot \rangle,$$

1. $(\forall(\sigma,x,x), b \in \underline{H})\,(J(J(b)) = J(b)\,\&\,J(\sigma,x,x) = (\sigma,x,x))$
2. $(\forall b \in \underline{H})(G(b) = b \longrightarrow U(b) = b\,\&\,L(b) = b)$
3. $(\forall(\sigma,x,y), (\alpha,g,h) \in \underline{H})$
   $(G(\sigma,x,y) = (\sigma,x,y), G(\alpha,g,h) = (\alpha,g,h) \longrightarrow G(U((\sigma,x.y) \sqcup (\alpha,g,h))) = U(\sigma \vee \alpha, xg, yh))$
4. $(\forall t, p \in \underline{H})\,(G(t) = t\,\&\,t \veebar p = t \longrightarrow G(U(p)) = U(p) = t)$.
5. $(\forall t, p \in \underline{H})(G(t) = t\,\&\,t \barwedge p = p \longrightarrow G(U(p)) = U(p) = t)$.

Proof. 1. $J(\sigma,x,x) = (\sigma,xx,xx) = (\sigma,x,x)$ as x is the range of the global groupoidal operation.

$$J(J(\sigma,x,y)) = J(\sigma,xy,xy) = (\sigma,(xy(xy),(xy)(xy)).$$

Now $(xy)(xy)$ includes the groupoidal operation on the diagonal of xy. So $(xy)(xy) = xy$ and $J(J(\sigma,x,y)) = J(\sigma,x,y)$.

2. If $G(\sigma,x,y) = (\sigma,x,y)$, then x, y must be granules. So the upper and lower approximations of these must, respectively, be x, y and $U(\sigma,x,y) = (\sigma,x,y)$, $L(\sigma,x,y) = (\sigma,x,y)$.
3. If x, y are granules relative to σ, α, respectively, then $(xg)^u$, $(yh)^u$ will be granules relative $\sigma \vee \alpha$. So the result follows.
4. If the last two components of an element are granules and p is an element below it, then the upper approximations of the last two components of p must coincide with the granules x, y, respectively.
5. The proof is similar to that of the above.

Definition 22.15. A subset K of $\underline{H}$, H being a power knowledge algebra, will be said to be a *bar filter* of H if the following conditions hold:

- $(\forall x \in K)(\forall y \in H)(x \barwedge y = x \longrightarrow y \in K)$
- K is closed under $\barwedge$.

Obviously, bar filters are lattice filters with respect to the forgetful structure. Further, the following properties hold:

Theorem 22.12. *If K is a bar filter of H, then*

1. *K is closed under $\sqcap, \sqcup$,*
2. *the collection of all bar filters forms an algebraic closure system.*

For expressing relative consistency of knowledge using the contaminated approach of [5], the basic knowledge algebra can be combined with a 'depends to a degree k' operator or the expression can be externalised through valuations. Power knowledge algebras would be overkill for the purpose as the 'depends to a degree k' operators are actually present in such an algebra.

For the granule based approach and the granular correspondence based measures, valuations /extensions over the power knowledge algebras would be suitable. Importantly, the valuations can be internalised as granules that are expressible within such algebras.

22.7.1 Algebraic Computational Aspects

Power knowledge algebras are computationally very amenable. This stems from the structure of the equivalence lattice $EQ(S)$ (and $E(S)$). A brief reformulation of fairly recent results on the generation of $EQ(S)$ is provided below to clarify this.

Theorem 22.13 ([33]).

1. *If S is finite, then $EQ(S)$ is atomistic.*
2. *If $Card(S) \geq 4$ and is finite, then $EQ(S)$ is generated by four elements, but it cannot be generated by three elements.*
3. *If S is finite and $Card(S) \geq 7$, then $EQ(S)$ would be $1+1+2$-generable, that is it would be generable by four equivalences with exactly two of them being comparable.*

If completeness of the lattice is assumed, and it is required that there is no inaccessible cardinal less than the cardinality of $EQ(S)$, then also it is generated by four elements [7]. But, this result is not particularly relevant in computational contexts. In [8], it is shown that completeness is not required when S is countable (long proof). Moreover $EQ(S)$ would be $1+1+2$-generated.

The generation process is as below: ($\langle a,b\rangle$ means the smallest equivalence containing a,b, while x,y are variables to be selected as per range of sub or superscript)

1. Let $S = \{a_i^k; k \in \aleph_0, 0 \leq i \leq k+12\} \cup \{b_i^k; k \in \aleph_0, 0 \leq i \leq k+11\}$
2. Let $\{B^k; k \in \aleph_0\}$ be a partition of S with each $B^k = \{a_i^k; i \leq k+12\} \cup \{b_i^k; i \leq k+11\}$
3. Define $\alpha = \langle a_y^x, a_{y+1}^x\rangle + \langle b_y^x, b_{y+1}^x\rangle$
4. Define $\beta = \langle a_y^x, b_y^x\rangle + \langle b_6^x, a_6^{x+1}\rangle$
5. Define $\gamma = \langle a_{y+1}^x, b_y^x\rangle + \langle b_4^x, a_2^{x+1}\rangle + \langle b_3^x, a_3^{x+1}\rangle + \langle b_{x+7}^x, a_{x+11}^{x+1}\rangle + \langle b_{x+8}^x, a_{x+12}^{x+1}\rangle$
6. Define $\delta = \langle a_0^x, a_{x+12}^x\rangle + \langle b_0^x, b_{x+11}^x\rangle$

α, β, γ and δ would be the generators of $EQ(S)$.

22.8 Abstraction

A set of granular axioms for knowledge are proposed in this section. The essential content is clarified in the light of fundamentally distinct approaches adopted in [5,6] and by the present author in [25].

Consider the following principles:

1. Individual granules are atomic units of knowledge.
2. If collections of granules combine subject to a concept of mutual independence, then the result would be a concept of knowledge. The 'result' may be a single entity or a collection of granules depending on how one understands the concept of *fusion* in the underlying mereology. In set theoretic (ZF) setting, the fusion operation reduces to set-theoretic union and so would result in a single entity.

3. Maximal collections of granules subject to a concept of mutual independence are admissible concepts of knowledge.
4. Parts common to subcollections of maximal collections are also knowledge.
5. All stable concepts of knowledge consistency should reduce to correspondences between granular components of knowledges. Two knowledges are *consistent* if and only if the granules generating the two have 'reasonable' correspondence.
6. Knowledge A is consistent with knowledge B if and only if the granules generating knowledge B are part of some of the granules generating A.
7. Knowledge A is consistent with knowledge B if and only if each of the granules generating knowledge A is part of some of the granules generating B.
8. Knowledge B is consistent with knowledge A if and only if the granules generating knowledge B are part of some of the granules generating A.
9. Knowledge B is n-consistent with knowledge A if and only if each of the granules generating knowledge A is union of at most n of the granules generating B.

The last four axioms are intended to give a feel of possible non-equivalent definitions of 'knowledge consistency' among the many possible. In a general context, interesting choices of axioms can be $\{1,2\}$, $\{1,3,4\}$, $\{1,3,5\}$ and $\{1,23,4,5\}$ among others. Choice functions have been used in [24], for imposing independence of granules at a local level. This leads to more complicated concepts of 'knowledge consistency'. The correspondence-based measure would have to depend on the granules chosen.

Among the principles, the second one should be compulsory for any concept of knowledge as knowledge is also a question of involving clear concepts alone and clarity is usually understood with respect to forms of independence. In [6], the authors do not follow this principle and so the quality of the extension should be doubted. Pawlak's concept was of course formulated in relation to approximation spaces, where the second principle would be trivially valid. The main issue that I see is that far too many things would qualify as knowledge if the second principle is violated.

Definition 22.16. Let

1. X, Y be two RYS,
2. $\mathcal{G}(X)$, $\mathcal{G}(Y)$ be two sets of granules associated with them respectively,
3. Z be a RYS that has surjective relational morphisms into both X and Y,
4. $\mathcal{G}(XZ)$, $\mathcal{G}(YZ)$ be the sets of granules induced by the relational morphisms,
5. ξ be an injective map of $\mathcal{G}(XZ)$ into $\mathcal{G}(YZ)$.

If ξ is such that

$$(\forall x \in \mathcal{G}(XZ))(\exists y \in \mathcal{G}(YZ))\, \mathbf{P}\xi(x)y,$$

P being the part-hood relation on Z, then the knowledge expressed by the pair $\langle Y, \mathcal{G}(Y)\rangle$ will be said to be consistent with the knowledge expressed by $\langle X, \mathcal{G}(X)\rangle$ relative Z and ξ.

It is possible to deal with knowledge consistency between two granulations on the same RYS ($\langle X, \mathcal{G}\rangle$ and $\langle X, \mathcal{H}\rangle$ say) as a special case of this definition.

22.9 Further Directions

In this research paper, the use of RST for modelling theories of knowledge has been critically examined from the usual viewpoint and in the light of the contamination problem. The relevance of the contamination problem in modelling human knowledge has also been explained. Two different semantics capable of expressing most of the desired features including consistency, cumulation, rough approximation, commonality and granularity of knowledge have been proposed. From a computational viewpoint the structures are directly applicable in practical contexts provided the set of attributes is at most countable. From the point of view of algebraic logic, a new concept of algebraizability using partial algebras is implicit in the presented proposal. These will be clarified in a subsequent paper for reasons of space. Abstract representation theorems for power knowledge algebras have also been omitted for the same reason.

The compatibility of measures of knowledge consistency proposed in [5, 6, 25] with the proposed semantics have also been considered in detail. The entire approach would also be relevant in the study of multi-agent systems and collections of approximation spaces. A concept of knowledge consistency based on granular correspondences has also been introduced. It can be viewed as a proper extension of the view that 'positive regions reflect dependence'.

The definition of knowledge from a granular perspective has also been considered in the light of the axiomatic approach to granules [24] from a philosophical perspective. It is argued that knowledge should be different from 'collections of objects or objects representing approximations'.

Acknowledgements. I would like to thank the referee for many useful remarks that led to improvement of the presentation of this paper.

References

1. Banerjee, M., Chakraborty, M.K.: Rough sets through algebraic logic. Fundamenta Informaticae 28, 211–221 (1996)
2. Banerjee, M., Chakraborty, M.K.: Algebras from rough sets – an overview. In: Pal, S.K., et al. (eds.) Rough-Neural Computing, pp. 157–184. Springer, Heidelberg (2004)
3. Banerjee, M., Khan, M. A.: Propositional Logics from Rough Set Theory. In: Peters, J.F., Skowron, A., Düntsch, I., Grzymała-Busse, J.W., Orłowska, E., Polkowski, L. (eds.) Transactions on Rough Sets VI. LNCS, vol. 4374, pp. 1–25. Springer, Heidelberg (2007)
4. Burmeister, P.: A Model-Theoretic Oriented Approach to Partial Algebras. Akademie-Verlag (1986, 2002)
5. Chakraborty, M.K., Samanta, P.: Consistency-Degree Between Knowledges. In: Kryszkiewicz, M., Peters, J.F., Rybiński, H., Skowron, A. (eds.) RSEISP 2007. LNCS (LNAI), vol. 4585, pp. 133–141. Springer, Heidelberg (2007)

6. Samanta, P., Chakraborty, M.K.: On Extension of Dependency and Consistency Degrees of Two Knowledges Represented by Covering. In: Peters, J.F., Skowron, A., Rybiński, H. (eds.) Transactions on Rough Sets IX. LNCS, vol. 5390, pp. 351–364. Springer, Heidelberg (2008)
7. Czedli, G.: Four generated large equivalence lattices. Acta Sci. Math. Szeged 62, 47–69 (1996)
8. Czedli, G.: 1+1+2-generated equivalence lattices. J. Alg. 221, 439–462 (1999)
9. Czelakowski, J., Pigozzi, D.: Fregean logics, pp. 1–72 (2003) (preprint)
10. Demri, S., Orlowska, E.: Incomplete Information: Structures, Inference, Complexity. Springer (2002)
11. Dummet, M.: The Logical Basis of Metaphysics. Harvard University Press (1991)
12. Farinas Del Cerro, L., Orlowska, E.: Dal – a logic for data analysis. Theoretical Computer Science 36, 251–264 (1997)
13. Font, J.M., Jansana, R.: A General Algebraic Semantics for Sentential Logics, vol. 7. Association of Symbolic Logic (2009)
14. Font, J.M., Jansana, R., Pigozzi, D.: A survey of abstract algebraic logic. Studia Logica 74(1/2), 13–97 (2003)
15. Järvinen, J.: Lattice Theory for Rough Sets. In: Peters, J.F., Skowron, A., Düntsch, I., Grzymała-Busse, J.W., Orłowska, E., Polkowski, L. (eds.) Transactions on Rough Sets VI. LNCS, vol. 4374, pp. 400–498. Springer, Heidelberg (2007)
16. Jezek, J., Mcenzie, R.: Variety of equivalence algebras. Algebra Universalis 45, 211–219 (2001)
17. Khan, M.A., Banerjee, M.: Formal reasoning with rough sets in multiple-source approximation spaces. Internat. J. Approximate Reasoning 49, 466–477 (2008)
18. Ljapin, E.S.: Partial Algebras and their Applications. Kluwer, Dordrecht (1996)
19. Mani, A.: Rough equalities from posets and rough difference orders. Fundamenta Informaticae 53(3,4), 321–333 (2002)
20. Mani, A.: Super rough semantics. Fundamenta Informaticae 65(3), 249–261 (2005)
21. Mani, A.: Algebraic semantics of similarity-based bitten rough set theory. Fundamenta Informaticae 97(1-2), 177–197 (2009)
22. Mani, A.: Integrated dialectical logics for relativised general rough set theory. In: Internat. Conf. on Rough Sets, Fuzzy Sets and Soft Computing, Agartala, India, 6 p. (Refereed) (2009), http://arxiv.org/abs/0909.4876
23. Mani, A.: Meaning, choice and similarity based rough set theory. In: Internat. Conf. Logic and Appl., Chennai, pp. 1–12 (Refereed) (January 2009), http://arxiv.org/abs/0905.1352
24. Mani, A.: Choice inclusive general rough semantics. Information Sciences 181(6), 1097–1115 (2011), http://dx.doi.org/10.1016/j.ins.2010.11.016, doi:10.1016/j.ins.2010.11.016
25. Mani, A.: Dialectics of counting and measures of rough set theory. In: IEEE Proceedings of NCESCT 2011, Pune, February 1-3, pp. 1–17. Arxiv:1102.2558 (2011), http://arxiv.org/abs/1102.2558
26. Mundici, D.: Generalization of abstract model theory. Fundamenta Math. 124, 1–25 (1984)
27. Novotny, M., Pawlak, Z.: Characterization of rough top and bottom equalities. Bull. Pol. Acad. Sci (Math.) 33(1-2), 99–104 (1985)
28. Novotny, M., Pawlak, Z.: Characterization of rough top and bottom equalities. Bull. Pol. Acad. Sci. (Math.) 33(1-2), 91–97 (1985)
29. Pagliani, P.: Pretopologies and dynamic spaces. Fundamenta Informaticae 59(2-3), 221–239 (2004)
30. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning About Data. Kluwer Academic Publishers, Dodrecht (1991)
31. Rauszer, C.: Rough logic for multi-agent systems. In: Masuch, M., Polos, L. (eds.) Logic at Work 1992. LNCS, vol. 808, pp. 161–181. Springer, Heidelberg (1994)

32. Skowron, A.: Approximate reasoning in mas - a rough set approach. In: Proc. Int. Conf. on Intelligent Agent Technology, pp. 12–18. IEEE Computer Society (2006)
33. Streitz, H.: Finite partition lattices are four generated. In: Proc. Conf. Lattice Theory, Ulm, pp. 257–259 (1975)
34. Swamy, U.M., Rao, S.R.: Algebraic topological closure operators. SEA Bull. Math. 26, 669–679 (2002)
35. Tripathy, B.K.: On Approximation of classifications, rough equalities and equivalences. In: Abraham, A., Falcón, R., Bello, R. (eds.) Rough Set Theory. SCI, vol. 174, pp. 85–133. Springer, Heidelberg (2009)
36. Yao, Y.: Relational interpretation of neighbourhood operators and rough set approximation operators. Information Sciences 111(1-4), 239–259 (1998)

Chapter 23
Classifiers Based on Nondeterministic Decision Rules

Barbara Marszał-Paszek and Piotr Paszek

Abstract. In the chapter, we discuss classifiers based on rough set theory and nondeterministic decision rules. We used two kinds of nondeterministic rules called the first and second type. These rules have a few decision values but the rules of the second type can have on the left-hand side one generalized descriptor. i.e., a condition of the form $a \in V$, where V is a two-element subset of the attribute value set V_a. We show that these kinds of rules can be used for improving the quality of classification and we propose classifications algorithms based on nondeterministic (first and second type) rules. These algorithms are using not only nondeterministic rules but also minimal rules in the sense of rough sets. In the chapter, these classifiers were tested on several data sets from the UCI Machine Learning Repository and the results were compared. The reported results of experiments show that the proposed classifiers based on nondeterministic rules can improve the classification quality but it requires tuning some of their parameters relative to analyzed data.

Keywords: Nondeterministic decision rules, conflict resolution, rough sets, rule-based classifiers.

23.1 Introduction

The knowledge background of this article is the rough set theory (RST) which was created by Professor Zdzisław Pawlak in 1982 [10]. Over the years many methods based on rule induction and rough sets [2,5,13,14,16] were developed.

At this time rule-based classification systems [6,8,15] are very useful in real-life data analysis. The chapter presents methods for improving the rule-based classification systems. We propose two methods for rule inducing. These methods are based

Barbara Marszał-Paszek · Piotr Paszek
Institute of Computer Science, University of Silesia
41-200 Sosnowiec, Poland
e-mail: {bpaszek,paszek}@us.edu.pl

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 445–454.
springerlink.com

on two types of nondeterministic decision rules. In both cases, we are searching for strong rules for a union of a few relevant decision classes — first type. For rules of the second type, one of the conditions of the left-hand side of such rules can be generalized, i.e., it can be of the form $a \in V$, where V is a two-element subset of the value set V_a of attribute a.

In the chapter, an application of first and second type nondeterministic rules in construction of rule-based classifier is presented.

We include the results of experiments showing that by combining rule-based classifiers based on minimal decision rules [11] with the classifier based on first and second type nondeterministic decision rules, it is possible to improve the classification quality and reduce classification error.

The following classification problem is considered: for a given decision table T [10, 11] and a new object x generate a value of the decision attribute on x using values of conditional attributes on x.

The chapter consists of five sections. In Section 23.2, we recall the basic rough set notions and we describe notions of first and second type nondeterministic decision rules. Section 23.3 contains a description of classification algorithm. Results of experiments are discussed in Section 23.4. Section 23.5 contains short conclusions.

23.2 Basic Notions

In 1982, Pawlak proposed the rough set theory as an innovative mathematical tool for describing knowledge, including the uncertain and inexact knowledge [10]. In this theory, knowledge is based on possibility (capability) of classifying objects. The objects may be for instance real objects, statements, abstract concepts and processes.

Let $T = (U, A, d)$ be a *decision table*, where $U = \{u_1, \ldots, u_n\}$ is a finite nonempty set of *objects*, $A = \{a_1, \ldots, a_m\}$ is a finite nonempty set of *conditional attributes* (functions defined on U), and d is the *decision attribute* (function defined on U).

We assume that for each $u_i \in U$ and each $a_j \in A$ the value $a_j(u_i)$ belong to $V_{a_j}(T)$ and the value $d(u_i)$ belong to $V_d(T)$, where $V_d(T)$ denotes the set of values of the decision attribute d on objects from U.

Every decision system T is associated formal language $\mathbf{L}(T)$. The *alphabet* of $\mathbf{L}(T)$ is a set [10]:

$$A \cup \{d\} \cup \bigcup_{a_j \in A} V_{a_j} \cup V_d \cup \{\neg, \vee, \wedge, \rightarrow, \equiv\} \cup \{), (\},$$

where $\{\neg, \vee, \wedge, \rightarrow, \equiv\}$ is the set of propositional connectives.

An *elementary formula* is an expression of the form $(a_j = v)$ (called descriptor) where $a_j \in A$ and $v \in V_{a_j}(T)$. A generalized descriptor is a formula of the form $a \in V$, where $V \subseteq V_a$.

The set of *condition formulas* ($F(A)$) of $\mathbf{L}(T)$ is the least set satisfying the following conditions:

1. Elementary formulas belong to the set $F(A)$;
2. Set $F(A)$ is closed with respect to propositional connectives.

The set of *decision formulas* ($F(d)$) of $\mathbf{L}(T)$ consists of formulas $d = v$ or $d = v_1 \vee \cdots \vee d = v_k$, where $v, v_1, \ldots, v_k \in V_d$.

In general, the *deterministic decision rule* in T has the following form:

$$(a_{j_1} \in V_1) \wedge \cdots \wedge (a_{j_k} \in V_k) \to (d = v),$$

where $a_{j_1}, \ldots, a_{j_k} \in A$, $V_j \subseteq V_{a_j}$, for $j \in \{1, \ldots, k\}$ and $v \in V_d(T)$.

The predecessor of this rule is a conjunction of some formulas from $F(A)$. The successor of this rule is an elementary formula from $F(d)$.

In this paper, we also consider nondeterministic decision rules. A *nondeterministic decision rule* in a given decision table T is of the form:

$$(a_{j_1} \in V_1) \wedge \cdots \wedge (a_{j_k} \in V_k) \Rightarrow d = (c_1 \vee \ldots \vee c_s), \tag{23.1}$$

where $a_{j_1}, \ldots, a_{j_k} \in A$, $V_j \subseteq V_{a_j}$, for $j \in \{1, \ldots, k\}$, numbers $j_1, \ldots, j_k$ are pairwise different, and $\emptyset \neq \{c_1, \ldots, c_s\} \subseteq V_d(T)$. Some notation about rules of the form (23.1) are introduced in [3].

Let us introduce some notation. If r is the nondeterministic rule (23.1) then by $lh(r)$ we denote its left-hand side, i.e., the formula $(a_{j_1} \in V_1) \wedge \cdots \wedge (a_{j_k} \in V_k)$, and by $rh(r)$ its right-hand side, i.e., the formula $d = (c_1 \vee \cdots \vee c_s)$.

By $||lh(r)||_T$ (or $||lh(r)||$, for short), we denote all objects from U satisfying $lh(r)$ [11]. To measure the quality of such rules, we use coefficients called the support and the confidence [1]. They are defined as follows. If r is a nondeterministic rule of the form (23.1), then the support of this rule in the decision system T is defined by

$$supp(r) = \frac{|||lh(r)|| \cap ||rh(r)|||}{|U|}, \tag{23.2}$$

and the confidence of r in T is defined by

$$conf(r) = \frac{|||lh(r)|| \cap ||rh(r)|||}{|||lh(r)|||}. \tag{23.3}$$

We also use a normalized support of r in T defined by

$$norm\_supp(r) = \frac{supp(r)}{\sqrt{|V(r)|}}, \tag{23.4}$$

where $V(r) \subseteq V_d(T)$ is a decision values set from right-hand side of the rule ($rh(r)$).

23.2.1 First Type Nondeterministic Rules

Now we can define a parameterized set of first type nondeterministic decision rules that are used in Section 23.3 for enhancing the quality of classification of rule-based classifiers.

This parameterized set is defined as the set of all nondeterministic rules r (over attributes in T, see (23.1)) such that:

1. On the left-hand sides of such rules are only conditions of the form $a \in \{v\}$, where $v \in V_a$. We write $a = v$ instead of $a \in \{v\}$;
2. $conf(r) \geq \alpha$, where $\alpha \in [0.5, 1]$ is a threshold;
3. $norm\_supp(r) \geq \beta$, where $\beta \in (0, 1]$ is a threshold;
4. $|V(r)| \leq k < |V_d(T)|$, where k is a threshold used as an upper bound on the number of decision values on the right-hand sides of rules — k is assumed to be small.

Hence, the first type nondeterministic decision rules are of the form:

$$(a_{j_1} = b_1) \wedge \ldots \wedge (a_{j_k} = b_k) \Rightarrow d = (c_1 \vee \ldots \vee c_s), \tag{23.5}$$

where $a_{j_1}, \ldots, a_{j_k} \in A$, for $j \in \{1, \ldots, k\}$, $b_j \in V_{b_j}(T)$, numbers $j_1, \ldots, j_k$ are pairwise different, and $\emptyset \neq \{c_1, \ldots, c_s\} \subseteq V_d(T)$.

The algorithm searching for the first type nondeterministic rules with sufficiently large support and relatively small sets of decisions defined by the right-hand sides of such rules for the decision table T was proposed in [9]. This algorithm is based on greedy strategy which is used to minimize the length of rules.

23.2.2 Second Type Nondeterministic Rules

A nondeterministic rule is of the second type if on its left-hand side all but one conditions are descriptors and the generalized descriptor on its left-hand side is of the form $a \in V$, where $V \subseteq V_a$ is a two-element set. Hence, the second type nondeterministic decision rules are of the form:

$$(a_{j_1} = b_1) \wedge \ldots \wedge (a_{j_i} = (b_{i_1} \vee b_{i_2})) \wedge \cdots \wedge (a_{j_k} = b_k) \rightarrow d = (c_1 \vee \ldots \vee c_s), \tag{23.6}$$

where $a_{j_1}, \ldots, a_{j_k} \in A$, $b_j \in V_{a_j}(T)$, for $j \in \{1, \ldots, k\}$, numbers $j_1, \ldots, j_k$ are pairwise different, $\emptyset \neq \{c_1, \ldots, c_s\} = V(r) \subseteq V_d(T)$ and $b_{i_1} \neq b_{i_2}$. The rule of the form (23.6) has nondeterminism on one condition attribute beside nondeterminism on decision part of rule. This attribute has two values but it is different from other attributes which have exactly one value. This type of nondeterministic rules appears as a result of shortening rules according to the principle MDL (*minimum description length*) [12].

The rule r of the form (23.6) can be represented by distribute on two nondeterministic rules of the form (23.5) such as:

$$r_1 : (a_{j_1} = b_1) \wedge \ldots \wedge (a_{j_i} = b_{i_1}) \wedge \cdots \wedge (a_{j_k} = b_k) \rightarrow d = (c_1 \vee \ldots \vee c_s),$$

$$r_2 : (a_{j_1} = b_1) \wedge \ldots \wedge (a_{j_i} = b_{i_2}) \wedge \cdots \wedge (a_{j_k} = b_k) \rightarrow d = (c_1 \vee \ldots \vee c_s).$$

The support of the second type nondeterministic rule in the decision table T is defined by

$$supp(r) = supp(r_1) + supp(r_2)$$

where for $i = 1, 2$ the $supp(r_i)$ looks like (23.2). We also use a normalized support of r in T. The confidence of r in T is defined by

$$conf(r) = conf(r_1) + conf(r_2)$$

where for $i = 1, 2$ the $conf(r_i)$ looks like (23.3).

By $RULE_{ND}(T, \alpha)$ we denote the set of all second type nondeterministic rules r, over attributes in T such that

$$1 \geq conf(r_1) + conf_T(r_2) = conf_T(r) \geq \alpha,$$

where parameter $\alpha \in [0.5, 1]$.

The main steps of the algorithm which was developed for generation of second type nondeterministic rules from $RULE_{ND}(T, \alpha)$ are as follows.

Input: T — decision table, parameter $\alpha \in [0.5, 1]$;
Output: $RULE_{ND}(T, \alpha)$ — the set of second type nondeterministic decision rules.
Step 1. $RULE_{ND}(T, \alpha)$ is empty set.
Step 2. For all condition attributes of T do the following:

- Find often appearing two values for this attribute;
- Generate subtable with restriction to these attribute values;
- Delete an attribute which was chosen;
- Generate the set of rules of type (23.5);
- Add to these rules attribute which was deleted (now rules have the form (23.6));
- Add these generated rules to the set $RULE_{ND}(T, \alpha)$;

23.3 Classifiers

In this section, we present an application of first and second type nondeterministic rules for classification process. We constructed two classifier C_1 and C_2. The set of first type nondeterministic rules and the set of minimal rules generated by the system RSES [14] were used to induce our first classifier (C_1). The set of second type nondeterministic rules and the set of minimal rules generated by the system RSES are used in inducing of our second classifier (C_2).

Because we have two groups of rules in each classifier we should negotiate between them.

For any new object, using C_1 or C_2 classifier, the decision value set is generated as follows.

First, for any new object, all nondeterministic rules matching the object are extracted. Next, from these matched rules, a rule with the largest (normalized) support is selected. In the case when several rules have the same support, the decision value set $V(r)$ of the nondeterministic rule r with the smallest set of decision value set ($|V(r)|$) is selected. If still several nondeterministic rules with the above property exist then first of them is selected.

Next, for this object, all minimal rules matching the object are extracted. We obtain a single decision value using standard voting procedure.

In this way, for any new object we obtain a decision value $v \in V_d(T)$ and a decision value set $V(r)$, where r is the rule selected from the set of nondeterministic rules.

The final decision for a given new object is obtained from the decision value v and decision value set $V(r)$ by the following strategy for resolving conflicts [7].

1. If for a given new object the standard voting based on minimal rules predicts the decision value v and $v \in V(r)$, (i.e., no conflict arises) then we take as the final decision the single decision v.
2. If for a given new object the standard voting based on minimal rules predicts the decision value v and $v \notin V(r)$ (i.e., conflict arises) then we take as the final decision value the single decision value v if support of the minimal (deterministic) rule (i.e., the support of the rule $lh(r_1) \vee \ldots \vee lh(r_k) \Rightarrow d = v$ in the considered decision table T, where $r_1, \ldots, r_k$ are all minimal rules matched by the new object) is larger than the normalized support of nondeterministic decision rule r and selected for the given new object. In the opposite case, we take as the final decision a single decision value from the set $V(r)$, with the largest support in T among decisions from $V(r)$.
3. If for a new object, the standard voting based on minimal rules predicts the decision value v and this object does not match any nondeterministic rule then we assign the decision v as the final decision.
4. If a given new object does not match any of the minimal rules then we assign as the final decision the single decision from $V(r)$ with the largest support among decisions from $V(r)$, where r is the rule selected by voting on nondeterministic rules.
5. In the remaining cases, a given new object is not classified.

23.4 Experiments

We have performed experiments on decision tables from UCI Machine Learning Repository [4] using proposed classification algorithms C_1 and C_2. The classification algorithm C_1 is based on all minimal decision rules and the first type

nondeterministic rules. The classification algorithm C_2 is based on all minimal decision rules and the second type nondeterministic rules.

The data sets selected for the experiments included the following: Balance Scale, Dermatology, Ecoli, Iris, Lymphography, Postoperative, Primary Tumor and Zoo. Decision table *BalanceScale* was generated in 1976 to model psychological experimental results. Decision table *Dermatology* contains information about differential diagnosis of erythemato-squamous diseases. Decision table *Ecoli* concerns the protein localization sites in Escherichia coli bacteria. *Iris* is the best known database to be found in the pattern recognition literature. *Lymphography* and *PrimaryTumor* data are two of three domains provided by the University Medical Center, Institute of Oncology from Ljubljana. The classification task of decision table *Postoperative* is to determine when patients in a postoperative recovery area should be sent to the next one. Decision table *Zoo* is a simple database containing information about animals from the zoo.

Table 23.1 Accuracy of classifiers based on first type nondeterministic decision rules — cross-validation method

Decision table			Classification	Classification algorithm						
				Alg^a	C_1, α^b					
name	# atr.	# obj.	factor		1.0	0.9	0.8	0.7	0.6	0.5
Balance Scale	5	625	acc × cover	78.30	80.74	**81.86**	81.83	80.79	79.78	76.83
			mrd	0.020	0.032	0.024	0.024	0.023	0.026	0.024
Dermatology	35	366	acc × cover	84.62	85.04	84.97	**85.27**	85.21	84.62	84.59
			mrd	0.012	0.008	0.014	0.011	0.014	0.012	0.009
Ecoli	8	336	acc × cover	54.76	55.45	**56.01**	54.52	54.40	50.63	50.63
			mrd	0.036	0.040	0.033	0.026	0.027	0.020	0.020
Lymphography	19	148	acc × cover	37.47	37.47	37.47	37.47	**37.50**	37.47	37.47
			mrd	0.038	0.038	0.038	0.038	0.037	0.038	0.038
Post-Operative	9	90	acc × cover	65.00	65.00	65.00	64.72	66.39	68.28	**68.83**
			mrd	0.061	0.061	0.061	0.058	0.064	0.072	0.033
Primary Tumor	18	339	acc × cover	59.71	**60.09**	60.09	60.09	60.09	60.09	60.09
			mrd	0.016	0.020	0.020	0.020	0.020	0.020	0.020
Iris	5	150	acc × cover	**90.47**	90.10	89.93	88.87	87.16	87.17	86.17
			mrd	0.018	0.028	0.026	0.031	0.048	0.0783	0.0583
Zoo	17	101	acc × cover	**92.48**	89.01	82.38	83.76	84.95	86.04	86.34
			mrd	0.044	0.0485	0.022	0.034	0.032	0.029	0.032

[a] In the column marked by *Alg* the classification is defined by the classification algorithm based on deterministic rules. In the column marked by C_1 the classification is defined by the algorithm based on first type nondeterministic rules and deterministic rules.

[b] Confidence of nondeterministic rules generated by the algorithm is not smaller than the parameter α.

Table 23.2 Accuracy of classifiers based on second type nondeterministic decision rules — cross-validation method

Decision table			Classification	Classification algorithm						
				Alg^a	C_2, α^b					
name	# atr.	# obj.	factor		1.0	0.9	0.8	0.7	0.6	0.5
Balance Scale	5	625	acc × cover	78.61	81.14	**81.33**	81.10	80.61	79.13	76.64
			mrd	0.020	0.026	0.026	0.022	0.028	0.023	0.040
Ecoli	8	336	acc × cover	54.99	55.51	**56.01**	54.52	54.40	50.63	50.63
			mrd	0.038	0.037	0.033	0.026	0.027	0.020	0.020
Lympho-graphy	19	148	acc × cover	38.06	38.06	38.06	38.06	38.06	38.06	38.06
			mrd	0.022	0.022	0.022	0.022	0.022	0.022	0.022
Post-Operative	9	90	acc × cover	65.00	65.44	65.44	65.44	65.67	67.89	**69.11**
			mrd	0.061	0.066	0.066	0.034	0.034	0.034	0.024
Primary Tumor	18	339	acc × cover	59.71	**60.09**	60.09	60.09	60.09	60.09	60.09
			mrd	0.016	0.020	0.020	0.020	0.020	0.020	0.020
Iris	5	150	acc × cover	90.47	90.47	90.47	90.47	90.47	90.47	90.47
			mrd	0.018	0.018	0.018	0.018	0.018	0.018	0.018
Zoo	17	101	acc × cover	**92.48**	88.22	83.96	85.74	86.63	87.33	87.43
			mrd	0.044	0.051	0.042	0.026	0.035	0.022	0.022

[a] In the column marked by *Alg* the classification is defined by the classification algorithm based on deterministic rules. In the column marked by C_2 the classification is defined by the algorithm based on second type nondeterministic rules and deterministic rules.

[b] Confidence of nondeterministic rules generated by the algorithm is not smaller than the parameter α.

In evaluation of the accuracy of classification algorithms on a decision tables (i.e., the percentage of correctly classified objects) the fivefold cross-validation method was used. For any considered data table, we used the classification algorithms C_1 and C_2 (for different values of parameter α). On testing sets the accuracy and the coverage factor were calculated. Also the maximal relative deviation (mrd) was calculated.

Table 23.1 and Table 23.2 contain the results of our experiments.

Classifier C_1 was compared with classifier from *RSESlib* (Rough Set Exploration System library) [2] based on all minimal decision rules and standard voting (indicated in the Table 23.1 as *Alg*), and results are included in Table 23.1. For six decision tables — Balance Scale, Dermatology, Ecoli, Lymphography, Post-Operative, Primary Tumor — the classification quality measured by *accuracy* × *coverage* was better for the classification algorithm C_1 than in the case of the classification algorithm from *RSESlib* based only on minimal rules with standard voting. For two data sets (Iris, Zoo), using only deterministic rules in the classification process, result was better than in case of the classification algorithm C_1.

Classifier C_2 was compared with classifier from *RSESlib* based on all minimal decision rules and standard voting (indicated in the Table 23.2 as Alg) and these results are shown in Table 23.2. For four decision tables — Balance Scale, Ecoli, Post-Operative, Primary Tumor — the classification quality measured by $accuracy \times coverage$ was better for the classification algorithm C_2 than in the case of the classification algorithm from *RSESlib* based only on minimal rules with standard voting (indicated in the Table 23.2 as Alg). For two data sets (Lymphography, Iris), the classification quality for both classifiers C_2 and Alg was equal. For data set Zoo, using only deterministic rules in classification process (classifier Alg), result was better than in case of the classification algorithm C_2.

For obtaining those better results, it was necessary to optimize the threshold for each data table. This means that the parameter α should be tuned for each data set.

23.5 Conclusions

Results of experiments with nondeterministic rules are showing that these rules can improve the classification quality. We have demonstrated this by using classification algorithms based on minimal decision rules and nondeterministic rules. Experiments have shown that proposed classifiers can improve classification accuracy, in our experiments the improvement was for the most decision tables. The second type of nondeterministic rules are very similar to the first type. One can say that it is another searching method for the first type of nondeterministic rules.

Proposed classifiers C_1 and C_2 are comparable in case of the classification quality. For Balance Scale data set, we got better result for the classifier C_1 then for the classifier C_2. For post-operative data set, we got better result for the classifier C_2 then for the classifier C_1. For the rest data sets, the classification quality for both classifiers C_1 and C_2 is the same.

Our algorithms for first and second type nondeterministic rules generation has polynomial computational complexity, which depends on number of objects and number of attributes.

At this moment, the proposed classification algorithm uses nondeterministic rules and minimal rules (from *RSESlib*). Since the algorithm for constructing minimal rules has exponential computational complexity, then we plan to use others classifiers (*e.g.* based on subsets of minimal decision rules or decision trees).

Acknowledgements. We wish to express our thanks to Professor Andrzej Skowron for his helpful comments during the processing of this work.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules Between Sets of Items in Large Databases. In: Buneman, P., Jajodia, S. (eds.) Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, pp. 207–216. ACM Press, New York (1993)
2. Bazan, J.G., Szczuka, M.S., Wojna, A., Wojnarski, M.: On the Evolution of Rough Set Exploration System. In: Tsumoto, S., Słowiński, R., Komorowski, J., Grzymała-Busse, J.W. (eds.) RSCTC 2004. LNCS (LNAI), vol. 3066, pp. 592–601. Springer, Heidelberg (2004)
3. Delimata, P., Marszał-Paszek, B., Moshkov, M., Paszek, P., Skowron, A., Suraj, Z.: Comparison of Some Classification Algorithms Based on Deterministic and Nondeterministic Decision Rules. In: Peters, J.F., Skowron, A., Słowiński, R., Lingras, P., Miao, D., Tsumoto, S. (eds.) Transactions on Rough Sets XII. LNCS, vol. 6190, pp. 90–105. Springer, Heidelberg (2010)
4. Frank, A., Asuncion, A.: UCI Machine Learning Repository, University of California, Irvine (2010), `http://archive.ics.uci.edu/ml` (cited September 1, 2011)
5. Grzymała-Busse, J.W.: LERS — A Data Mining System. In: Maimon, O., Rokach, L. (eds.) The Data Mining and Knowledge Discovery Handbook, pp. 1347–1351. Springer, New York (2005)
6. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. SIGKDD, Explorations 11(1), 10–18 (2009)
7. Marszał–Paszek, B., Paszek, P.: Minimal Templates and Knowledge Discovery. In: Kryszkiewicz, M., Peters, J.F., Rybiński, H., Skowron, A. (eds.) RSEISP 2007. LNCS (LNAI), vol. 4585, pp. 411–416. Springer, Heidelberg (2007)
8. Michalski Ryszard, `http://www.mli.gmu.edu/michalski` (cited September 1, 2011)
9. Paszek, P., Marszał-Paszek, B.: Deterministic and Nondeterministic Decision Rules in Classification Process. Journal of Medical Informatics and Technologies 15, 87–92 (2010)
10. Pawlak, Z.: Rough Sets — Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Dordrecht (1991)
11. Pawlak, Z., Skowron, A.: Rudiments of Rough Sets. Information Sciences 177, 3–27 (2007); Rough Sets: Some Extensions. Information Sciences 177, 28–40 (2007); Rough Sets and Boolean Reasoning. Information Sciences 177, 41–73 (2007)
12. Rissanen, J.: Modeling by Shortest Data Description. Automatica 14, 465–471 (1978)
13. Rosetta, `http://www.lcb.uu.se/tools/rosetta` (cited September 1, 2011)
14. Rough Set Exploration System, `http://logic.mimuw.edu.pl/~rses` (cited September 1, 2011)
15. Triantaphyllou, E., Felici, G. (eds.): Data Mining and Knowledge Discovery Approaches Based on Rule Induction Techniques. Springer Science and Business Media, LLC (2006)
16. Tsumoto, S.: Modelling Medical Diagnostic Rules Based on Rough Sets. In: Polkowski, L., Skowron, A. (eds.) RSCTC 1998. LNCS (LNAI), vol. 1424, pp. 475–482. Springer, Heidelberg (1998)

Chapter 24
Approximation and Rough Classification of Letter-Like Polygon Shapes

Elisabeth Rakus-Andersson

Abstract. It is a privilege for the author to be involved in composing a book chapter in the anthology devoted to the life and scientific occupation of Professor Zdzisław Pawlak. The author made a personal acquaintance with the outstanding scientist Professor Pawlak and still remembers him as a warm and gentle human being. Professor Pawlak's theory of rough sets was taught to students during the courses in Computational Intelligence established at Blekinge Institute of Technology in Karlskrona, Sweden. In some Master of Science theses, the principles of rough set theory were discussed in the aspects of technical applications. In this context, we can feel that the theory is still alive and very useful.

In this work, we recall again the basics of rough sets to apply them to the classification of discrete two-dimensional point sets, which form the shapes resembling some letters. These possess very irregular patterns and cannot be approximated by standard curves without committing large errors. Since the approximation of letter-like point sets is required by the latter classification of their shapes then we, due to own model, wish to find a continuous curve which fits best for each distribution of points. To accomplish the thorough approximation of finite point sets, we test parametric s-truncated functions piecewise, which warrants a high accuracy of approximating. By operating on the functions, replacing samples of points obtained during experiments carried out, we are able to adopt the rough set technique to verify decisions about the primary recognitions of the curves' appearance as letter shapes. Even if the curves are stretched and shaped differently in the plane, we will divide them in classes gathering similar objects. Our investigations have not a character of pure art — on the contrary — their results are utilized in the classifications of internet packet streams or the analysis of wave signals typical of, e.g., medical examinations.

Elisabeth Rakus-Andersson
School of Engineering, Blekinge Institute of Technology
S-37179 Karlskrona, Sweden
e-mail: Elisabeth.Andersson@bth.se

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 455–474.
springerlink.com

Keywords: *s*-truncated parametric function, approximation of letter-like point sets, rough classification, indiscernibility relation, lower approximation, upper approximation.

24.1 Introduction

Some observations of the behavior of two variables X and Y provide us with sequences of values x and y, which can be included in the pairs (x,y), $x \in X$, $y \in Y$, treated further as the coordinates of points in the two-dimensional system. We suppose that a finite set A consists of the points (x,y); therefore, A can be illustrated as a polygon with the nodes joined by segments of straight lines.

Certain experiments, in which $y \in [y_{\min}, y_{\max}], y_{\min} < 0, y_{\max} > 0$, provide us with the polygon (the set A) composed of parts looking like bells, e.g., like A sketched in Fig. 24.1.

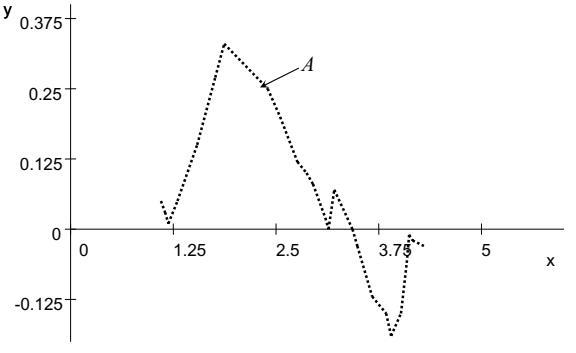


Fig. 24.1 The example of the multi-shaped polygon reflecting $A = \{(x,y)\}$

The polygon that ties a lot of straight-line segments cannot constitute a good interpolation of the points since a number of first degree polynomial equations are too large for further efficient analysis and, moreover, such interpolation is not smooth enough.

The most popular classical method of approximating applied to a set of points is known as the least-square regression that inspires scientists to develop plenty of modern variants [2]. Other algorithms of approximating adopt such technical tools as cubic polynomials based on four points [8], tangent curves [1], free algebras [6], weighted approximations [32] or 1-corner polygonal chains [4]. Lately, the methods of constrained optimization [3], the human perception approaches [7] and the observations of localities [9] have appeared to widen a spectrum of approximation tools.

We consider an approximation of multi-shapes from Fig. 24.1 by s-truncated functions used piecewise as another model assisting the numerical curve fitting to point sets. The y-coordinates of the points, constituting the vertices of A, belong to the interval $[0, y_{\max}]$ or $[y_{\min}, 0]$; therefore, it will be desirable to approximate pieces of the polygon A by the clock-like s-functions after adjusting their heights from $[0,1]$ to $[0,\varepsilon]$, $\varepsilon \in [y_{\min}, y_{\max}]$. The procedure of forming the approximation of A by truncated s-functions tied by pieces of straight lines, if needed, is developed in the second section of the chapter. "The sampled truncated s", as we call an entire approximation curve, consists of first and second degree polynomials. It follows the polygon's shape very closely and it cumulates a very low error measuring deviations between the approximating curve and the polygon. This should be regarded as an important advantage of the sampled truncated approximation method when comparing it to other procedures. We should add that the approach proposed as the approximation of non-standard curve shapes contributes an original own solution in numerical mathematics. We conduct the discussion concerning "sampled truncated s" in Section 24.2.

Let us assume that changes in the y values of curves, similar in shape to the set depicted in Fig. 24.1, are important indicators in the further classification process of these curves. They can resemble some letters, *e.g.*, N, W, or M, and can occur in different places along the x axis. The shapes of the letters mentioned can point out important states of the processes considered, like in the internet protocol pattern recognition.

The internet protocol patterns involving the shapes of N, W and M appear, *e.g.*, in a problem of complex bottleneck recognition [5]. A shape of the bottleneck contains information of its nature. We assign the letter of "M" to the shared bottleneck, the "W" letter to a shaping bottleneck and, finally, "N" stands for an overloaded bottleneck. The patterns sometimes are mixed and they do not resemble basic letters, but we still want to find for them an appropriate class, which has most of their attributes.

Therefore the reliable classification of patterns N, W and M is very important. To assign the curves to adequate classes denoted by N, W or M, we should compare their y-coordinate values. It is not possible if the curves under consideration are scattered in different segments of the x axis. To be able to make the curves comparable, which aims at estimating their deviations in y values stated at the same x, we should move the curves over the interval $[0,1] \subset X$. The procedure developed by the author [25-26] transfers the curves over a common interval and preserves their original shapes. In Section 24.3, we outline all transformations derived for this purpose.

When remembering that the letters correspond to certain states, like in the example of bottleneck [5], we accomplish the letter classifications by a very thorough approximation of point sets combined with rough set techniques [11-22, 33]. The internet protocol shape recognition has stimulated us to develop the final solution, which appears in Section 24.4.

24.2 Sampled Truncated *S*-Functions in the Approximation of Letter-Like Polygons

The approach to approximation of irregular polygons presented below constitutes an own original solution [23-26], which differs from other procedures of seeking approximation curves [1, 2, 3, 6, 7, 8, 32].

We discover that the x values of pairs included in A belong to the interval $[x_{\min}(A), x_{\max}(A)]$, in which $x_{\min}(A)$ is the smallest and $x_{\max}(A)$ is the largest x-value in A. Next, we divide $[x_{\min}(A), x_{\max}(A)]$ in subintervals $\lfloor x_{\min(A_j)}, x_{\max(A_j)} \rfloor$ where $A_j,\ j = 1, \ldots, Q$, are parts of A. In the parts A_j, we can experience either the growth or the decrease of the y values corresponding to these x that are placed between the borders $x_{\min(A_j)}$, and $x_{\max(A_j)}$ standing for the smallest and, respectively, the largest value of x in A_j. *S*-functions or segments of straight lines attached to two adjacent s-curves approximate the A_j components.

Example 1
The pairs, which create the polygon depicted in Fig. 24.1, are the members of

$$\begin{aligned} A = \{&(1.1,\ 0.05), (1.15,\ 0.03), (1.19,\ 0.01), (1.3,\ 0.05),\\ &(1.54,\ 0.15), (1.76,\ 0.27), (1.87,\ 0.33), (2.4,\ 0.25),\\ &(2.55,\ 0.2), (2.76,\ 0.12), (2.87,\ 0.1), (2.96,\ 0.08),\\ &(3.1,\ 0.02), (3.14,\ 0), (3.21,\ 0.07), (3.48,\ 0),\\ &(3.49,\ -0.03), (3.67,\ -0.12), (3.84,\ -0.15), (3.9,\ -0.19),\\ &(4.02,\ -0.15), (4.09,\ -0.06), (4.12,\ -0.01), (4.16,\ -0.02),\\ &(4.3,\ -0.03)\}. \end{aligned}$$

By measuring the direction of changes in the y values, which point out extreme nodes in A's shape, we consider the subintervals [1.1, 1.19], [1.19, 1.3], [1.3, 1.87], [1.87, 3.14], [3.14, 3.21], [3.21, 3.43], [3.43, 3.9], [3.9, 4.12], [4.12, 4.16], [4.16, 4.3]. Over the intervals either s-functions or straight lines will be applied as approximation tools.

The s-function with the standard parameters α, β, γ introduced by [10, 11, 34, 35] as

$$y = s(x, \alpha, \beta, \gamma) = \begin{cases} (1) \quad \varepsilon \left(2 \left(\frac{x-\alpha}{\gamma-\alpha} \right)^2 \right) & \text{for } \alpha \le x \le \beta, \\ (2) \quad \varepsilon \left(1 - 2 \left(\frac{x-\gamma}{\gamma-\alpha} \right)^2 \right) & \text{for } \beta \le x \le \gamma, \end{cases} \tag{24.1}$$

is fitted best for the appearances of the "half-bells" A_j. Since the y values of the classical s belong to the interval $[0, 1]$ ($-s$ has its y values in $[-1, 0]$) then we should insert an additional parameter ε in (24.1) to accommodate a height of the function to the real data existing in the set $A_j,\ j = 1, \ldots, Q$. The parameter β is estimated as the arithmetic mean of α and γ. We have already introduced the partition of A

by means of the subsets A_j, looking like "half-bells", then we should denote each s-function that approximates A_j by $s_{A_j}(x, \alpha_{A_j}, \beta_{A_j}, \gamma_{A_j}, \varepsilon_{A_j})$.

Example 2
We intend to recall in mind how the s-function (24.1) looks like. If we choose, *e.g.*, $\alpha = 0$, $\gamma = 2.1$, $\beta = 1.05$, and $\varepsilon = 0.26$ as some casual numbers, then the function will have a graph drawn in Fig. 24.2.

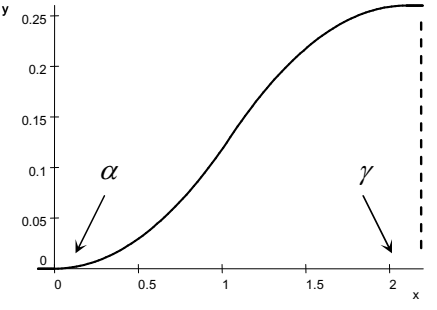


Fig. 24.2 The s-function for $\alpha = 0, \gamma = 2.1$ and $\varepsilon = 0.26$

In the next part of the section, we discuss different cases of A_j's approximation that is dependent on the sizes of y-coordinates in the set A_j.

Let us assume that the values of the y-coordinates in A_j associated with the x-values belonging to $[x_{\min(A_j)}, x_{\max(A_j)}]$ appear in the ascending order, and let us notice that no y-coordinate is equal to zero. The pair $\left(x_{\min(A_j)}, y(x_{\min(A_j)})\right)$ $(y(x_{\min(A_j)})$ corresponds to $x_{\min(A_j)})$ begins the set A_j, but we cannot identify $x_{\min(A_j)}$ as α_{A_j}. Thus, the value of α_{A_j} in the s_{A_j}-function, which is expected to approximate A_j, is unknown. To find α_{A_j} we, at first, accept the value of ε_{A_j} as the largest y-coordinate in A_j, assigned to the x-coordinate $x_{\max(A_j)} = \gamma_{A_j}$. We can now reconstruct the value of the remaining parameter α_{A_j} according to the following patterns:

a) $\alpha_{A_j} = \dfrac{X_{\min(A_j)} - \gamma_{A_j}\sqrt{\frac{y(x_{\min(A_j)})}{2\cdot\varepsilon_{A_j}}}}{1-\sqrt{\frac{y(x_{\min(A_j)})}{2\cdot\varepsilon_{A_j}}}}$ for $y(x_{\min(A_j)}) < \frac{\varepsilon}{2}$ - this modifies (24.1) as

$$y = \begin{cases} (1) \quad \varepsilon_{A_j}\left(2\left(\dfrac{x-\alpha_{A_j}}{\gamma_{A_j}-\alpha_{A_j}}\right)^2\right) & \text{for } x_{\min(A_j)} \le x < \beta_{A_j}, \\ (2) \quad \varepsilon_{A_j}\left(1-2\left(\dfrac{x-\gamma_{A_j}}{\gamma_{A_j}-\alpha_{A_j}}\right)^2\right) & \text{for } \beta_{A_j} \le x < \gamma_{A_j}; \end{cases} \tag{24.2}$$

b) $\alpha_{A_j} = \gamma_{A_j} - \frac{\gamma_{A_j} - x_{\min(A_j)}}{\sqrt{\frac{\varepsilon_{A_j} - y(x_{\min(A_j)})}{2 \cdot \varepsilon_{A_j}}}}$ for $y(x_{\min(A_j)}) \geq \frac{\varepsilon}{2}$ - then the $s_{A_j}(x)$ formula, yielded by (24.1), appears as

$$y = \begin{cases} (1) & 0 & \text{for } x < x_{\min(A_j)}, \\ (2) & \varepsilon_{A_j} \left(1 - 2 \left(\frac{x - \gamma_{A_j}}{\gamma_{A_j} - \alpha_{A_j}}\right)^2\right) & \text{for } x_{\min(A_j)} \leq x \leq \gamma_{A_j}. \end{cases} \tag{24.3}$$

It happens that the positions of pairs in the set A_j introduce the descending order among points with respect to the y-coordinate values. We assume that none of them is equal to zero. The pair $\left(x_{\max(A_j)}, y(x_{\max(A_j)})\right)$ will end the set A_j, but $x_{\max(A_j)} \neq \gamma_{A_j}$. Let us assign the largest value of y in A_j, regarded as ε_{A_j}, to the x-coordinate $x_{\max(A_j)} = \alpha_{A_j}$. Then, it is possible to restore the missing value of γ_{A_j} that is one of the parameters included in a function $1 - s_{A_j}(x, \alpha_{A_j}, \beta_{A_j}, \gamma_{A_j}, \varepsilon_{A_j})$ applied to approximate A_j.

We distinct between two cases of adjusting the parameter γ_{A_j} to the data set A_j:

c) $\gamma_{A_j} = \frac{x_{\max(A_j)} - \alpha_{A_j} \sqrt{\frac{y(x_{\max(A_j)})}{2 \cdot \varepsilon_{A_j}}}}{1 - \sqrt{\frac{y(x_{\max(A_j)})}{2 \cdot \varepsilon_{A_j}}}}$ for $y(x_{\max(A_j)}) < \frac{\varepsilon}{2}$ - thus we suggest the changes in (24.1) due to

$$y = \begin{cases} (1) & \varepsilon_{A_j} \left(1 - 2 \left(\frac{x - \alpha_{A_j}}{\gamma_{A_j} - \alpha_{A_j}}\right)^2\right) & \text{for } \alpha_{A_j} \leq x < \beta_{A_j}, \\ (2) & \varepsilon_{A_j} \left(2 \left(\frac{x - \gamma_{A_j}}{\gamma_{A_j} - \alpha_{A_j}}\right)^2\right) & \text{for } \beta_{A_j} \leq x \leq x_{\max(A_j)}; \end{cases} \tag{24.4}$$

d) $\gamma_{A_j} = \alpha_{A_j} + \frac{x_{\max(A_j)} - \alpha_{A_j}}{\sqrt{\frac{\varepsilon_{A_j} - y(x_{\max(A_j)})}{2 \cdot \varepsilon_{A_j}}}}$ for $y(x_{\max(A_j)}) \geq \frac{\varepsilon}{2}$ - and we convert formula (24.1) of $s_{A_j}(x)$ to

$$y = \begin{cases} (1) & \varepsilon_{A_j} \left(1 - 2 \left(\frac{x - \alpha_{A_j}}{\gamma_{A_j} - \alpha_{A_j}}\right)^2\right) & \text{for } \alpha_{A_j} \leq x \leq x_{\max(A_j)}, \\ (2) & 0 & \text{for } x > x_{\max(A_j)}. \end{cases} \tag{24.5}$$

The modified s_{A_j} constitutes a section of the classical s-function. Because of that we will name it a truncated s-function. By selecting the minimal and the maximal x value and the maximal y value, which exist in the set A_j, we prepare the mathematical apparatus with (24.2)-(24.5) for computing unknown parameters α_{A_j} or γ_{A_j}. The point, in which the y-coordinate takes the ε_{A_j} value and the x-coordinate

is equal to the γ_{A_j} value for $s_{A_j}(x,\alpha_{A_j},\beta_{A_j},\gamma_{A_j},\varepsilon_{A_j})$ (respectively the α_{A_j} value for $1-s_{A_j}(x,\alpha_{A_j},\beta_{A_j},\gamma_{A_j},\varepsilon_{A_j})$), is one of the vertices in A. The total approximation s_A of A consists of all s_{A_j}, $j=1,\ldots,Q$, and is called "sampled truncated s".

To preserve the right shape of the approximating curve, it is advisable to tie two functions $s_{A_j}, s_{A_{j+1}}$, between the points $\left(x_{\max(A_j)}, y(x_{\max(A_j)})\right)$, $\left(x_{\min(A_{j+1})}, y(x_{\min(A_{j+1})})\right)$ by the equation of a straight line in the form

$$y = line_{A_j}(x) = k_{A_j}x + l_{A_j} \quad \text{for} \quad x_{\max(A_j)} \le x < x_{\min(A_{j+1})}. \tag{24.6}$$

Example 3
The "sampled truncated s", accommodated to the data from Ex.1, is shown in Fig. 24.3.

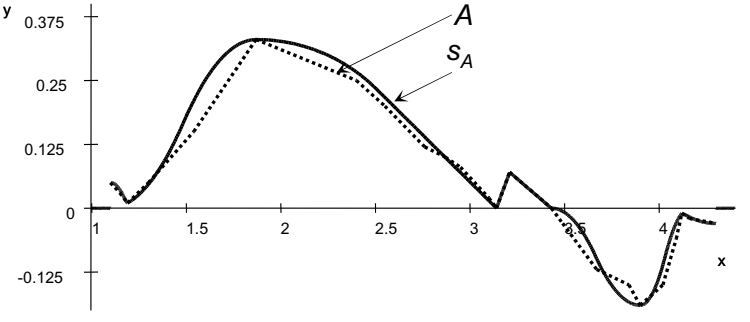


Fig. 24.3 The approximation of A by "sampled truncated s"

The first subset of points $A_1 \subset A$, in which the y coordinates establish the descending order, is placed over $[1.1, 1.19]$. Since no y value is equal to zero we will reconstruct

$$\gamma_{A_1} = \frac{1.19 - 1.1\sqrt{\frac{0.01}{2\cdot 0.05}}}{1-\sqrt{\frac{0.01}{2\cdot 0.05}}} = 1.2316$$

for $\varepsilon_{A_1} = 0.05, \alpha_{A_1} = 1.1, x_{\max(A_1)} = 1.19$, and $y(x_{\max(A_1)}) = 0.01$ in accordance with c).

In the next interval $A_2 = [1.19, 1.87]$, the value of α_{A_2} should be estimated. If we request the values of $\varepsilon_{A_2} = 0.33$, $\gamma_{A_2} = 1.87$, $x_{\min(A_2)} = 1.19$ and $y(x_{\min(A_2)}) = 0.01$ then

$$\alpha_{A_2} = \frac{1.19 - 1.87\sqrt{\frac{0.01}{2\cdot 0.33}}}{1 - \sqrt{\frac{0.01}{2\cdot 0.33}}} = 1.0945$$

due to a).

The formula of s_A for A is expanded as the split definition

$$y = s_A(x) = \begin{cases} 0.05\left(1 - 2\left(\frac{x-1.1}{1.1-1.2316}\right)^2\right) & \text{for } 1.1 \le x < 1.1658, \\ 0.05\left(2\left(\frac{x-1.1968}{1.1-1.2316}\right)^2\right) & \text{for } 1.1658 \le x < 1.19, \\ \quad\vdots & \quad\vdots \quad\vdots \\ (-0.31818)x + 1.0914 & \text{for } 3.21 \le x < 3.43, \\ \quad\vdots & \quad\vdots \quad\vdots \\ -0.03\left(1 - 2\left(\frac{x-4.3}{3.9958-4.3}\right)^2\right) & \text{for } 4.1479 \le x < 4.3. \end{cases}$$

We can prove some additional operations on the s-function values, *e.g.*, $y = (s(x))^2$ or $y = (s(x))^{\frac{1}{2}}$ to match a shape of the function to the given polygon in the best way.

It is worth noticing that the total error that collects the deviations of $s_A(x)$ from A is very small. This is entailed in further analysis of curves, where even small differences between approximating curves and point sets can lead to wrong conclusions.

24.3 Sampled *S*-Functions over the *X*-Interval [0,1]

The curve created for A has a particular pattern as it resembles the letter N. In some technical problems, we obtain the sets of points looking like three letters, namely, N, M and W. The shapes of mentioned letters can be disturbed or vague, which makes difficult to classify them properly, i.e., we do not know exactly how to include the curves in classes determined by N, M and W. To ensure if a vague or unknown object can belong to the considered class or not, we accomplish a classification according to the rules of rough set theory. Definitely, we can model some other shapes as well but we need the mentioned letters in the unsolved bottleneck problem [5], which inspires us to lead the further discussion.

If we are given several polygons then we should wish to collect all approximated objects over a common interval [0, 1] in the X-space to measure their deviations in y-values with respect to the same x value.

Example 4

Suppose that we have obtained different shapes of the curves originating from the point sets $A^1 - A^5$. Each of them is approximated by a continuous function, composed of s-sections and pieces of straight lines that link the parts of s-functions if it is necessary. Figure 24.4 reveals the polygons and the approximating functions over their original intervals along the x axis.

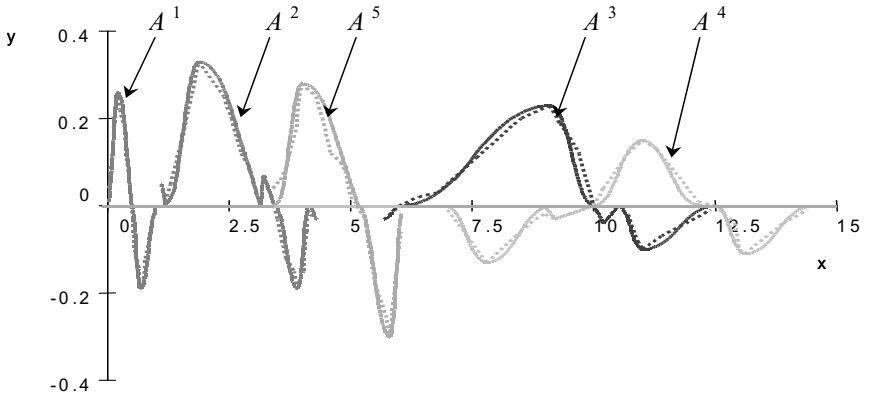


Fig. 24.4 The approximated polygons $A^1 - A^5$

We suggest the polygon memberships in the following way: A^1, A^3 and A^5 belong to the "N" class, A^4 is a member of the "W" class while the origin of A^2 is unknown. The hypothesis is primary and should be verified. This information will be introduced in the decision table constructed for curves in the next section.

In the further analysis, we use only the continuous curves, also named $A^1 - A^5$.

To move all curves, generally denoted by $A^1, \ldots, A^p$, to the same start point settled as the origin of the xy coordinate system we suggest the following transformations.

Suppose that the A^i-curve, $i = 1, \ldots, p$, is placed in the x-interval $\left[x_{\min}(A^i), x_{\max}(A^i)\right]$. We move the jth segment $s_{A^i_j}$, approximating the subset A^i_j, of A^i, $i = 1, \ldots, p, j = 1, \ldots, Q$, to a position close to the origin by introducing the formula

$$y = \tag{24.7}$$

$$\begin{cases} (1)\ \varepsilon_{A^i_j}\left(2\left(\frac{x-(\alpha_{A^i_j}-x_{\min}(A^i))}{\gamma_{A^i_j}-\alpha_{A^i_j}}\right)^2\right) & \text{for } x_{\min(A^i_j)} - x_{\min}(A^i) \le x < \beta_{A^i_j} - x_{\min}(A^i), \\ (2)\ \varepsilon_{A^i_j}\left(1-2\left(\frac{x-(\gamma_{A^i_j}-x_{\min}(A^i))}{\gamma_{A^i_j}-\alpha_{A^i_j}}\right)^2\right) & \text{for } \beta_{A^i_j} - x_{\min}(A^i) \le x \le \gamma_{A^i_j} - x_{\min}(A^i). \end{cases}$$

The function (24.7), with $x_{\min(A^i_j)}$ being the least x-value in the set of pairs A^i_j, should be previously suited to A^i_j by applying (24.2) (or (24.3)). The choice of the $(1-s)$-function induces the application of (24.4) (or (24.5)).

The straight line (24.6) is transferred nearby the origin by the action of an equation

$$\begin{aligned} y = line_{A^i_j}(x) = \\ k_{A^i_j} x + l_{A^i_j} + k_{A^i_j} \cdot x_{\min}(A^i) \quad \text{for } x_{\max(A^i_j)} - x_{\min}(A^i) \le x < x_{\min(A^i_{j+1})} - x_{\min}(A^i) \\ = K_{A^i_j} x + L_{A^i_j}. \end{aligned} \tag{24.8}$$

Example 5
Figure 24.5 shows $A^1 - A^5$ attached to the origin after performing (24.7) and (24.8).

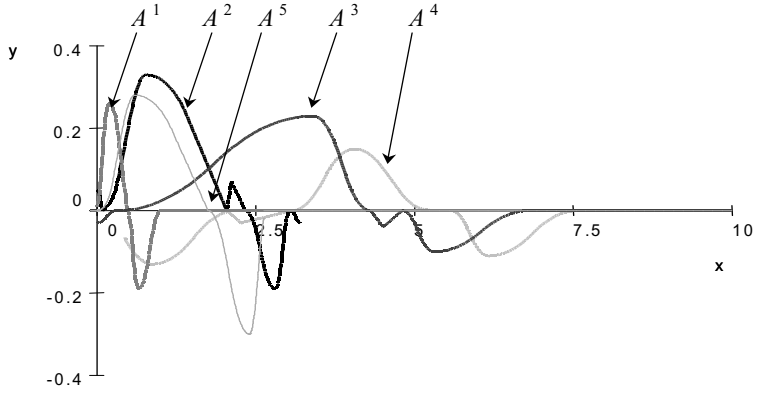


Fig. 24.5 The curves $A^1 - A^5$ with their start points at the origin

In Fig. 24.5, we recognize A_2 as A from Ex. 1. We decide and modify "sampled truncated s" for $A_2 = A$, as a function

$$y = s_{A^2}(x) = \begin{cases} 0.05\left(1 - 2\left(\frac{x-(1.1-1.1)}{1.1-1.2316}\right)^2\right) & \text{for } 1.1 - 1.1 \le x < 1.1658 - 1.1, \\ 0.05\left(2\left(\frac{x-(1.2316-1.1)}{1.1-1.2316}\right)^2\right) & \text{for } 1.1658 - 1.1 \le x < 1.19 - 1.1, \\ \vdots & \vdots \ \vdots \\ (-0.31818)x + 0.7414 & \text{for } 3.21 - 1.1 \le x < 3.43 - 1.1, \\ \vdots & \vdots \ \vdots \\ -0.03\left(1 - 2\left(\frac{x-(1.43-1.1)}{3.9958-4.3}\right)^2\right) & \text{for } 4.1479 - 1.1 \le x < 4.3 - 1.1. \end{cases}$$

which displaces A_2's beginning to the origin.

The comparison of all curves will be successful if we can observe them at a common interval of the X-space. Let us determine the interval [0, 1] as a new domain for all split-functions $A^1 - A^p$. Each piece $s_{A^i_j}$ or $line_{A^i_j}$, $i = 1,\ldots,p,\ j = 1,\ldots,Q$, should be shrunk or widened proportionally to fit it for the interval [0, 1] together with other pieces.

To achieve the required movements of s^i_j over [0,1], we initiate the parameter $\delta_{A^i} = \frac{1}{x_{\max}(A^i) - x_{\min}(A^i)}$. After inserting δ_{A^i} in (24.7), we generate a new formula [10]

$$y = \begin{cases} (1) \quad \varepsilon_{A^i_j}\left(2\left(\frac{x-(\alpha_{A^i_j}-x_{\min}(A^i))\delta_{A^i}}{(\gamma_{A^i_j}-\alpha_{A^i_j})\delta_{A^i}}\right)^2\right) \\ \quad \text{for } (x_{\min(A^i_j)} - x_{\min}(A^i))\delta_{A^i} \le x < (\beta_{A^i_j} - x_{\min}(A^i))\delta_{A^i}, \\ (2) \quad \varepsilon_{A^i_j}\left(1-2\left(\frac{x-(\gamma_{A^i_j}-x_{\min}(A^i))\delta_{A^i}}{(\gamma_{A^i_j}-\alpha_{A^i_j})\delta_{A^i}}\right)^2\right) \\ \quad \text{for } (\beta_{A^i_j} - x_{\min}(A^i))\delta_{A^i} \le x \le (\gamma_{A^i_j} - x_{\min}(A^i))\delta_{A^i}. \end{cases} \tag{24.9}$$

Before equipping Eq. (24.8) with the parameter δ_{A^i}, we should find another form of (24.8), adapted to the range [0, 1]. We suggest a new appearance of (24.8) in the form of

$$y = K_{A^i_j}x + L_{A^i_j} = \frac{x + \frac{L_{A^i_j}}{K_{A^i_j}}}{\frac{1}{K_{A^i_j}}} \quad \text{for } x_{\max(A^i_j)} - x_{\min}(A^i) \le x < x_{\min(A^i_{j+1})} - x_{\min}(A^i). \tag{24.10}$$

We can now place δ_{A^i} in (24.10) according to a pattern

$$y = \frac{x + \frac{L_{A^i_j}\delta_{A^i}}{K_{A^i_j}}}{\frac{\delta_{A^i}}{K_{A^i_j}}} \quad \text{for } (x_{\max(A^i_j)} - x_{\min}(A^i))\delta_{A^i} \le x < (x_{\min(A^i_{j+1})} - x_{\min}(A^i))\delta_{A^i}. \tag{24.11}$$

Example 6

The applications of (24.9) and (24.11) to every s-section and every line segment of $A^1 - A^5$ yield the effects of collecting all curves over the x-domain [0, 1] as plotted in Fig. 24.6. The new formula of A^2 is a function

$$y = s_{A^2}(x) =$$

$$\begin{cases} 0.05\left(1-2\left(\frac{x-(1.1-1.1)0.31}{(1.1-1.2316)0.31}\right)^2\right) & \text{for } (1.1-1.1)0.31 \le x < (1.1658-1.1)0.31, \\ 0.05\left(2\left(\frac{x-(1.2316-1.1)0.31}{(1.1-1.2316)0.31}\right)^2\right) & \text{for } (1.1658-1.1)0.31 \le x < (1.19-1.1)0.31, \\ \vdots & \vdots \quad \vdots \\ \frac{x+\frac{0.7414\cdot 0.31}{-0.31818}}{\frac{0.31}{-0.31818}} & \text{for } (3.21-1.1)0.31 \le x < (3.34-1.1)0.31, \\ \vdots & \vdots \quad \vdots \\ -0.03\left(1-2\left(\frac{x-(1.43-1.1)0.31}{(3.9958-4.3)0.31}\right)^2\right) & \text{for } (3.9958-1.1)0.31 \le x < (4.3-1.1)0.31, \end{cases}$$

in which we have added $\delta_{A^2} = \frac{1}{4.3-1.1} \approx 0.31$.

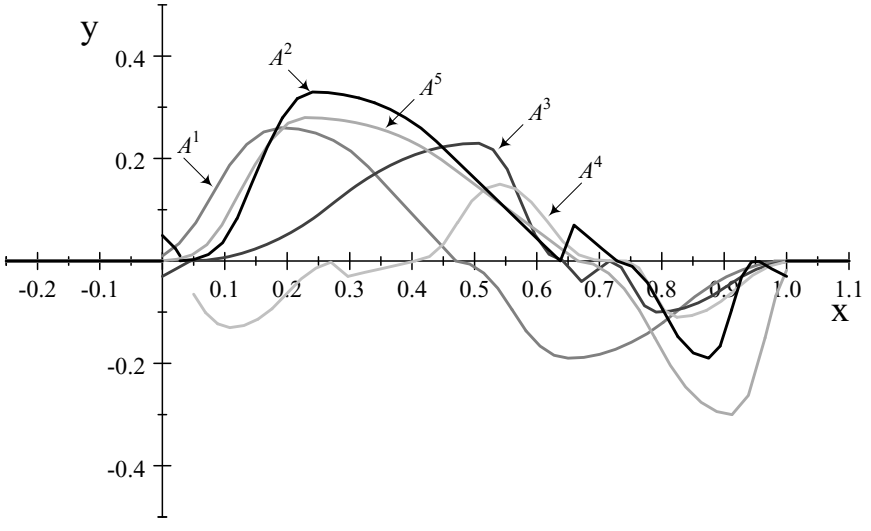


Fig. 24.6 The curves $A^1 - A^5$ over the common x-interval [0, 1]

The mathematical tools used for polygons result in the creation of a common collection of curves over a substantial part of the x axis determined as the interval [0, 1]. By researching the y values related to selected x values, we can prepare data to a decision table, being a crucial part of further empirical examinations in which the selected elements of rough set theory constitute a foundation.

24.4 Rough Set Theory in Polygon Classification

As a method of classifying the different letter shapes, we have selected a classification made by means of rough set theory. Let us insert a general description of that efficient classification rule [11-22, 33].

Rough set philosophy is founded on the assumptions that with every object of the universe of discourse, we associate some information in the form of data or knowledge. Objects characterized by the same information are indiscernible (similar) in view of the available information about them. The indiscernible objects are called elementary sets, and form basic granules of knowledge about the universe.

The classical rough set analysis is based on the indiscernibility relation that describes indistinguishbility of objects. Each rough set has boundary line cases, i.e., the objects which cannot be with certainty classified as the members of the set or its complement. With any rough set, a pair of precise sets called its lower and upper approximation is associated. The lower approximation consists of all objects, which surely belong to the set and the upper approximation contains all objects which possibly are members of the set. The difference between the upper and the lower approximation constitutes the boundary region of the rough set.

The main purpose of rough set analysis is the induction of approximation of concepts from the acquired data. Data are often represented as information tables. The information systems are data tables or decision tables whose columns are labeled by attributes, rows are labeled by objects and entries of the tables are attribute values. The decision tables or the data tables describe decisions in terms of conditions that must be specified in the decision tables.

Roughness can be seen as another approach to vagueness similar to fuzzy set theory.

Advanced Internet Protocol network applications, such as IP video conferences, Voice-over-IP or on-line games, involve IP network operations. These generate data streams, which are sensitive to specific delays and throughput requirements. The streams resemble letters N, M and W, like bottleneck shapes already mentioned in Introduction.

To include unknown sets defined as "internet protocols" within classes N, M and W we apply some elements of rough set theory [11-22, 33], which have proven useful in the process of a polygon classification [23-26].

The comparison of y values, characterizing different curves, is now regarded as the most essential moment of classification. To build a technique of their differentiation let us first divide interval $[y_{\min}, y_{\max}]$ in n levels Y_t, $t = 0, \ldots, n-1$, verbally defined. The name of the Y_t^{th} level assists a fuzzy set given by two s-class functions aggregated as a membership function

$$\mu_{Y_t}(y) = \begin{cases} \text{left } \mu_{Y_t}(y) \\ \text{right } \mu_{Y_t}(y), \end{cases} \tag{24.12}$$

for [27-31, 36]

$$\text{left } \mu_{Y_t}(y) = \tag{24.13}$$
$$\begin{cases} 2\left(\frac{y-((y_{\min}-h_\gamma)+h_\gamma\cdot t)}{h_\gamma}\right)^2 & \text{for } (y_{\min}-h_\gamma)+h_\gamma\cdot t \le y \le \left(y_{\min}-\frac{h_\gamma}{2}\right)+h_\gamma\cdot t, \\ 1-2\left(\frac{y-(y_{\min}+h_\gamma\cdot t)}{h_\gamma}\right)^2 & \text{for } \left(y_{\min}-\frac{h_\gamma}{2}\right)+h_\gamma\cdot t \le y \le (y_{\min})+h_\gamma\cdot t. \end{cases}$$

and

$$\text{right } \mu_{Y_t}(y) = \tag{24.14}$$
$$\begin{cases} 1-2\left(\frac{y-(y_{\min}+h_\gamma\cdot t)}{h_\gamma}\right)^2 & \text{for } (y_{\min})+h_\gamma\cdot t \le y \le \left(y_{\min}+\frac{h_\gamma}{2}\right)+h_\gamma\cdot t, \\ 2\left(\frac{y-((y_{\min}+h_\gamma)+h_\gamma\cdot t)}{h_\gamma}\right)^2 & \text{for } \left(y_{\min}+\frac{h_\gamma}{2}\right)+h_\gamma\cdot t \le y \le (y_{\min}+h_\gamma)+h_\gamma\cdot t. \end{cases}$$

The formula depends on the minimal y value $y_{\min}$, the number of the level t and the parameter $h_\gamma = \frac{y_{\max}-y_{\min}}{n-1}$. The parameter h_Y is equal to a distance between the beginnings of two adjacent membership functions of Y_t and Y_{t+1}.

Example 7
Since the y values of the curves from Examples 1 to 6 are located in $[y_{\min}, y_{\max}] = [-0.35, 0.35]$ then we can distinguish the list "*Changes in y-values*" $= \{Y_0 =$ "*negative*", $Y_1 =$ "*zero*", $Y_2 =$ "*positive*". For $y_{\min} = -0.35, t = 0, 1, 2$ and $h_Y = 0.35$, the membership functions of sets Y_t, $t = 0, 1, 2$ are drawn in Fig. 24.7.

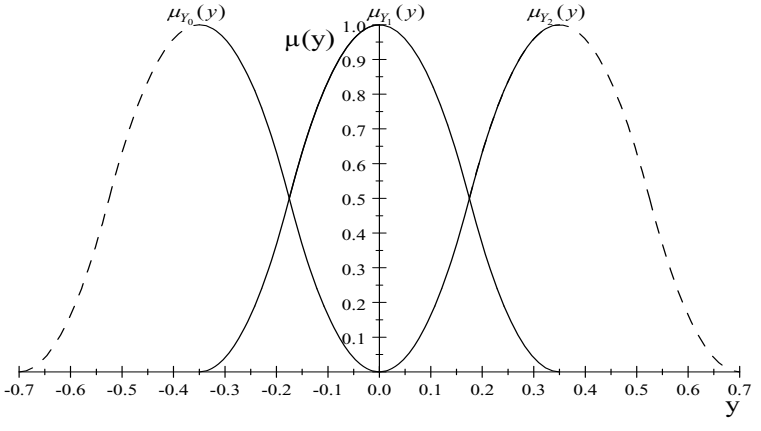


Fig. 24.7 The fuzzy sets $Y_0 - Y_2$

If we regard the membership values greater than 0.5 as the most essential, then we should select the appropriate intervals corresponding to them in $[y_{\min}, y_{\max}] = [-0.35, 0.35]$. These intervals represent fuzzy sets from the list "*Changes in*

y-values". We thus differentiate $[-0.35,-0.175]$ as the representative support of $Y_0 =$ "*negative*", $[-0.175,0.175]$ - as the interval typical of $Y_1 =$ "*zero*" and [0.175, 0.35] as the substantial piece of the set $Y_2 =$ "*positive*". The changes in y values accompanying the patterns of $A^1 - A^5$ will be treated as characteristic signals that help us to recognize the curves' letter shapes. The three differentiated intervals of y values will be coded in the further procedure of classification. We assign code -1 to $[-0.35,-0.175]$, code 0 to $[-0.175,0.175]$ and code 1 to [0.175, 0.35] as designed in Fig. 24.8.

Each considered polygon has an envelope created by a continuous function A^i, $i = 1,\ldots,5$, that approximates it over [0, 1] (see Ex. 6). For every value x belonging to [0, 1], we can establish the association between x and one of the codes by utilizing (24.9) and (24.11).

Let us state a universe set $U = \{A^1,\ldots,A^p\}$ composed of the polygons resembling the letters M, N and W. The objects of U are determined by two groups of attributes, the so-called condition and decision attributes included in sets B and D, respectively. We assume that the set B consists of sizes $x_k \in [0,1]$, $k = 1,\ldots,m$, associated with values $code_{A^i}(x_k)$, $i = 1,\ldots,p$, which are equal to the integers -1, 0 and 1.

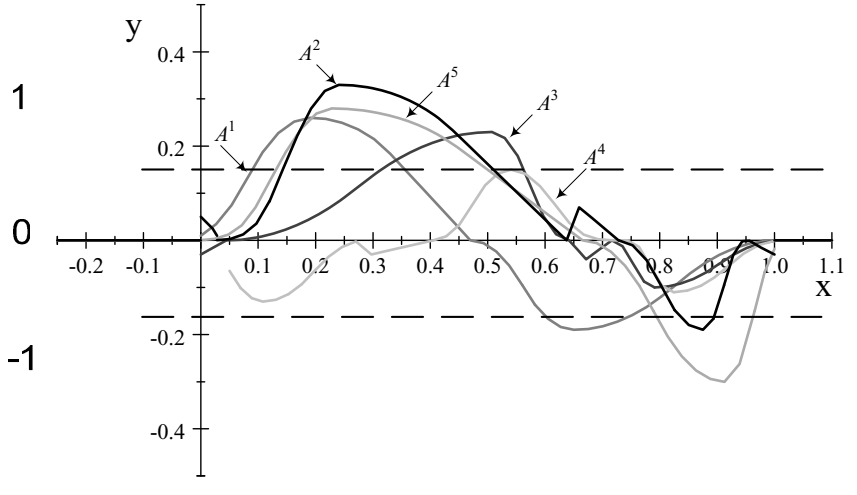


Fig. 24.8 Code values in the partition of the interval $[y_{\min}, y_{\max}]$

Assume that we wish to assign some members to the "*N*" class. Then, the set D obtains an attribute stated as "the membership of a polygon in "*N*"", where the membership is expressed as "*yes*", "*no*" and "*unknown*".

The triple $I = (U, B, D)$ forms the decision table, which is treated as the data basis for an equivalence relation $I(B)$ called the indiscernibility relation and defined by a relationship

$$I(B) = \left\{(A^i, A^s) : code_{A^i}(x_k) = code_{A^s}(x_k)\right\} \text{ for each size } x_k\,, \tag{24.15}$$

where $k = 1, \ldots, m;\ i, s = 1, \ldots, p$.

We find the equivalence classes of the relation $I(B)$, i.e., the blocks $IB(A^i)$ as the sets

$$IB(A^i) = \left\{A^s : (A^i, A^s) \in I(B)\right\}. \tag{24.16}$$

The values of decision D constitute the entries of the last column in Table I.

By following a general rough set procedure [11-22], we create a set $X = \{A^i$: which have the decision "*yes*" assigned in the last column with respect to the shape "N"$\}$.

The lower approximation set of X

$$B_*(X) = \left\{A^i : IB(A^i) \subseteq X\right\} \tag{24.17}$$

contains the curves (thus polygons), which match the "N" class without doubts (they are sure members of "N").

The upper approximation of X

$$B^*(X) = \left\{A^i : IB(A^i) \cap X \neq 0\right\} \tag{24.18}$$

samples the members of U, which may belong to the considered class "N".

The elements of a boundary set

$$B_{border}(X) = B^*(X) - B_*(X) \tag{24.19}$$

are the members of "N" in a certain grade.

The membership degree of A^i, interpreted as a degree of being a member in "N", is computed as

$$\mu_{\text{“}N''}(A^i) = \frac{|X \cap IB(A^i)|}{|IB(A^i)|}. \tag{24.20}$$

Example 8

We refer to the data concerning $A^1 - A^5$ and sampled after transformations over [0, 1] due to Fig. 24.8. $A^1 - A^5$ are pictures of different letter patterns. We state $U = \{A^1, A^2, A^3, A^4, A^5\}$. The decision triple $I = (U, B, D)$, prepared for sizes $x_k = 0.125, 0.250, 0.375, 0.500, 0.625, 0.750, 0.875$ is expanded in Table 24.1. As the example has rather a didactic character then we have selected only a few sizes from [0, 1] to match the table width to the breadth of the page. In real-life applications, we can prepare a dense set of sizes by selecting many of them from the continuous set [0, 1]. We are also able to add more shapes A^i to make them the objects of

classification but, unfortunately, the transmitted data from technicians concerned only five patterns to compare.

Table 24.1 The decision table $I = (U, B, D)$

$A^i \backslash x_k$	0.125	0.250	0.375	0.500	0.625	0.750	0.875	Class"N"
A^1	1	1	1	0	−1	−1	0	*Yes*
A^2	1	1	1	1	0	0	−1	*unknown*
A^3	0	1	1	1	0	0	0	*yes*
A^4	−1	0	0	1	0	0	0	*no*
A^5	1	1	1	1	0	0	−1	*yes*

The code values -1, 0 and 1 have been determined in conformity with the membership function values of A^i in x_k, $k = 1, \ldots, 7$.

The equivalence relation $I(B)$, provided in accordance with (24.15), is formed by a set of pairs

$$I(B) = \{(A^1,A^1),(A^2,A^2),(A^3,A^3),(A^4,A^4),(A^5,A^5),(A^2,A^5),(A^5,A^2)\}.$$

The equivalence classes of $I(B)$ are decided as the sets

$$IB(A^1) = \{(A^1\}, \; IB(A^2) = \{A^2,A^5\}, \; IB(A^3) = \{A^3\},$$
$$IB(A^4) = \{A^4\}, \; IB(A^5) = \{A^2,A^5\}.$$

The value of the decision attribute "N" = "*yes*" generates the set $X = \{A^1,A^3,A^5\}$, which, in turn, is an essential factor implementing

$$B_*(X) = \{A^1,A^3\}, \; B^*(X) = \{A^1,A^2,A^3,A^5\} \text{ and } B_{border}(X) = \{A^2,A^5\}.$$

The curve membership degrees, whose sizes confirm the membership in the "N" class, are obtained as

$$\mu_{\text{"}N\text{"}}(A^1) = 1, \;\; \mu_{\text{"}N\text{"}}(A^2) = \frac{1}{2}, \;\; \mu_{\text{"}N\text{"}}(A^3) = 1, \;\; \mu_{\text{"}N\text{"}}(A^4) = 0, \;\; \mu_{\text{"}N\text{"}}(A^5) = \frac{1}{2}.$$

When comparing the results above to the primary decision, concerning the membership of curves in class "N" due to the decision table (see Table 24.1), we interpret the obtained results as a verification constituting the secondary decision. We can now conclude that A^1 and A^3 are the true members of "N"-class in U, whereas A^2 and A^5 may belong to the investigated class to certain grades. We can also notice that A^2 affects a status of A^5 negatively and, on the contrary, we can see that A^5 upgrades an importance of A^2 in the "N"-class. A_2, which has not been recognized at the first stage of classification, has joined the group of possible members of "N". The recognition of the "N" shapes like, *e.g.*, overloaded bottleneck, is very important in some internet processes when we want to eliminate these shapes as disturbances.

The effects of rough classification have been compared to results of fuzzy clustering. The fuzzy cluster method concerns sampling points (x,y), with intuitively predetermined membership degrees, in three clusters M, W and N. After performing an iterative process of clustering procedure, we can only improve thoroughness of the membership degrees of strings (x,y) in different clusters. In that way, we are furnished with a hint to prioritise some clusters to be more suitable than the others, when moving (x,y) to them.

Rough classification allows accomplishing a very subtle analysis of deviations in y values, which should be regarded as an advantage of the methodology. Moreover, we could find exact objects shaped like N, something, which clustering cannot offer.

24.5 Conclusions

Some finite sets of pairs are often interpolated by polygons, which seldom have convenient equations mathematically expanded. Although there exists a large number of approximation methods applied to point sets — especially the different variations of least square regressions — we suggest applying a new procedure of approximation. This originates from the standard s-functions in truncated forms that approximate the irregular parts of the polygons very smoothly.

The functions, called by us "the sampled, truncated s" are composed of the first and second degree-polynomials in the form of split definitions. The low degrees of approximating functions make further operations on them rather easy. One truncated s-segment can approximate many nodes belonging to the point set. In this manner, we reduce a number of piecewise functions involved in the general definition of an approximating collection. But most of all we notice that "the sampled, truncated s" follows the changes of the polygon's pattern very sensitively to guarantee the high thoroughness of approximation results.

A new process of approximation is sometimes invented in mathematics as an interesting theoretical item without greater practical validity. To prove the empirical aspect of "sampled truncated s" we want to consider praxis of recognizing signals forming point sets, which resemble some letters. These are approximated by s-functions to be later moved over the interval [0, 1]. The set [0, 1] is approved as an appropriate domain for all functions to compare them. The operations on functions, transferring the curves over [0, 1], are the own contributions in the solution.

The accomplishment of a successful classification of unknown objects, possessing only some features typical of the considered class, is not an easy task. By applying the rough set theory combined with earlier achievements in approximation, we could classify polygons within the same class even if they had an unknown origin. We could also state sure and possible members of the group under control with the degrees of membership.

Rough set technique is thus a very powerful and easy tool of a successful classification of objects. It is worth emphasizing once again that rough set theory provides

us with the appealing approach to dividing members of a certain universe in clearly interpretable groups.

References

1. Agadi, A., Penot, J.-P.: A comparative study of various notions of approximation of sets. Journal of Approximation Theory 134(1), 80–101 (2005)
2. Cherkassky, V., Gehring, D., Mulier, F.: Comparison of adaptive methods for function estimation for samples. IEEE Transactions on Neural Networks 7(4), 969–984 (1996)
3. Claisse, A., Frey, P.: Level set driven smooth curve approximation from unorganized or noisy point set. In: ESAIM: Proc., 2009, vol. 27, pp. 254–271 (2009)
4. Díaz-Bánez, J.-M., Gómez, F., Hurtado, F.: Approximation of point sets by 1-corner polygonal chains. INFORMS Journal of Computing 12(4), 317–323 (2000)
5. Fiedler, M., Tutschku, K., Carlsson, P.: Identification of performance degradation in ip net-works using throughput statistics. In: Charzinski, J., Lehnert, R., Tran-Gia, P. (eds.) Providing Quality of Service in Heterogeneous Environments. Proceedings of the 18th International Teletraffic Congress - ITC-18, Berlin, Germany, August 31-September 5, pp. 399–408. Elsevier, Amsterdam (2003)
6. Haan, O.: A free algebraic solution for the planar approximation. Nuclear Physics B 705, 563–575 (2005)
7. Latecki, L.J., Lakämper, R., Sobel, M.J.: Polygonal Approximation of Point Sets. In: Reulke, R., Eckardt, U., Flach, B., Knauer, U., Polthier, K. (eds.) IWCIA 2006. LNCS, vol. 4040, pp. 159–173. Springer, Heidelberg (2006)
8. Lavoue, G., Dupont, F., Baskurt, A.: A new subdivision based approach for piecewise smooth approximation of 3d polygonal curves. Journal of Pattern Recognition Society 38(8), 1139–1151 (2005)
9. Liu, Y., Yuan, Y., Xiao, D., Zhang, Y., Hu, J.: A point-set-based approximation for areal objects: A case study of representing localities. Computers, Environment and Urban Systems 34(1), 28–39 (2010)
10. Novák, V., Perfilieva, I.: Evaluating of linguistic expressions and functional fuzzy theories in fuzzy logic. In: Zadeh, L.A., Kacprzyk, J. (eds.) Computing with Words in Information - Intelligent Systems 2. STUDFUZZ, vol. 33, pp. 383–406. Physica-Verlag, Heidelberg (1999)
11. Pal, S.K., Mitra, P.: Case generation using rough sets with fuzzy representation. IEEE Transactions on Knowledge and Data Engineering 16(3), 292–300 (2004)
12. Pawlak, Z.: Rough sets. International Journal of Computer and Information Sciences 11(5), 341–356 (1982)
13. Pawlak, Z.: On rough sets. Bulletin of the EATCS 24, 94–108 (1984)
14. Pawlak, Z.: Rough sets: Theoretical Aspects of Reasoning about Data. Kluwer, Dordrecht (1991)
15. Pawlak, Z.: Vagueness - a Rough Set View. In: Mycielski, J., Rozenberg, G., Salomaa, A. (eds.) Structures in Logic and Computer Science. LNCS, vol. 1261, pp. 106–117. Springer, Heidelberg (1997)
16. Pawlak, Z.: Rough set approach to knowledge-based decision support. European Journal of Operational Research 99(1), 48–57 (1997)
17. Pawlak, Z.: Rough sets and intelligent data analysis. Information Sciences 147(1-4), 1–12 (2002)
18. Pawlak, Z.: Decision Networks. In: Tsumoto, S., Słowiński, R., Komorowski, J., Grzymała-Busse, J.W. (eds.) RSCTC 2004. LNCS (LNAI), vol. 3066, pp. 1–7. Springer, Heidelberg (2004)

19. Pawlak, Z., Polkowski, L., Skowron, A.: Rough Set Theory. In: Wah, B. (ed.) Wiley Encyclopedia of Computer Science and Engineering. Wiley & Sons, Inc., New York (2007), doi:10.1002/9780470050118.ecse466
20. Pawlak, Z., Skowron, A.: Rudiments of rough sets. Information Sciences 177(1), 3–27 (2007)
21. Polkowski, L., Lin, T.Y., Tsumoto, S. (eds.): Rough Set Methods and Applications: New Developments in Knowledge Discovery in Information Systems. STUDFUZZ, vol. 56. Springer-Verlag/Physica-Verlag, Heidelberg (2000)
22. Polkowski, L.: Rough Sets: Theoretical Foundations. Springer, Heidelberg (2002)
23. Rakus-Andersson, E., Salomonsson, M.: The truncated p-functions in approximation of multi-shaped polygons. In: De Baets, B., Caluwe, R., de Tré, G., Fodor, J., Kacprzyk, J., Zadrożny, S. (eds.) Current Issues in Data and Knowledge Engineering, Proceedings of EUROFUSE 2004 - EURO EWG on Fuzzy Sets, pp. 444–452. EXIT, Warszawa (2004)
24. Rakus-Andersson, E., Salomonsson, M.: P-truncated functions and rough sets in the classification of internet protocols. In: Liu, Y., Chen, G., Ying, M. (eds.) Proceedings of Eleventh International Fuzzy Systems Association World Congress - IFSA 2005, Beijing, pp. 1487–1492. Tsinghua University Press - Springer (2005)
25. Rakus-Andersson, E.: S-truncated functions and rough sets in approximation and classification of bottleneck polygons. In: Proceedings of Modeling Decisions for Artificial Intelligence - MDAI 2005, Tsukuba-Japan, CD-ROM, Consejo Superior de Investigaciones Cientificas (2005), paper nr 049
26. Rakus-Andersson, E.: Fuzzy and Rough Techniques in Medical Diagnosis and Medication. Springer, Heidelberg (2007)
27. Rakus-Andersson, E., Salomonsson, M., Zettervall, H.: Two-player Games with Fuzzy Entries of the Payoff Matrix. In: Computational Intelligence in Decision and Control - Proceedings of FLINS 2008, Madrid, pp. 593–598. World Scientific (2008)
28. Rakus-Andersson, E., Salomonsson, M., Zettervall, H.: Ranking of weighted strategies in the two-player games with fuzzy entries of the payoff matrix. In: Xhafa, F., Herrera, F., Abraham, A., et al. (eds.) Proceedings of the 8th International Conference on Hybrid Intelligent Systems, Barcelona, CDR by Universitat Polytecnica de Catalunya (2008)
29. Rakus-Andersson, E., Jain, L.C.: Computational intelligence in medical decisions making. In: Rakus-Andersson, E., Yager, R.R., Ichalkaranje, N., Jain, L.C. (eds.) Recent Advances in Decision Making. SCI, vol. 222, pp. 145–159. Springer, Heidelberg (2009)
30. Rakus-Andersson, E.: Approximate reasoning in surgical decisions. In: Proceedings of the International Fuzzy Systems Association World Congress - IFSA 2009, Lisbon, pp. 225–230. Instituto Superior Technico, Lisbon (2009)
31. Rakus-Andersson, E., Zettervall, H., Erman, M.: Prioritization of weighted strategies in the multi-player games with fuzzy entries of the payoff matrix. Int. J. of General Systems 39(3), 291–304 (2010)
32. Salmeri, M., Mencattini, A., Rovatti, R.: Function approximation using non-normalized siso fuzzy systems. International Journal of Approximate Reasoning 26, 211–231 (2001)
33. Słowiński, R.: Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory. Kluwer, Dordrecht (1992)
34. Zadeh, L.A.: Outline of a new approach to the analysis of complex systems and decision processes. IEEE Trans. Systems 3, 28–44 (1973)
35. Zadeh, L.A.: Calculus of fuzzy restrictions. In: Zadeh, L.A., Fu, K.S., Tanaka, K., Shimura, M. (eds.) Fuzzy Sets and their Applications to Cognitive and Decision Processes, pp. 1–39. Academic Press, New York (1975)
36. Zettervall, H., Rakus-Andersson, E., Forssell, H.: The Mamdani controller in prediction of the survival length in elderly gastric patients. In: Pellegrini, M., Fred, A.L.N., Filipe, J., Gamboa, H. (eds.) BIOINFORMATICS 2011 - Proceedings of the International Conference on Bioinformatics Models, Methods and Algorithms, Rome, Italy, January 26-29, pp. 283–286. SciTePress (2011)

Chapter 25
Rough Set-Based Identification of Heart Valve Diseases Using Heart Sounds

Mostafa A. Salama, Omar S. Soliman, Ilias Maglogiannis, Aboul Ella Hassanien, and Aly A. Fahmy

Abstract. Taking into account that heart auscultation remains the dominant method for heart examination in the small health centers of the rural areas and generally in primary healthcare set-ups, the enhancement of this technique would aid significantly in the diagnosis of heart diseases. In this context, this chapter introduces the ability of rough set methodology to successfully classify heart sound diseases without the need applying feature selection. Heart sound data sets represents real life data that contains continuous attributes and a large number of features that could be hardly classified by most of classification techniques. Discretizing the raw heart sound data and applying a feature reduction approach should be applied prior any classifier to increase the classification accuracy results. The capabilities of rough set in discrimination, feature reduction classification have proved their superior in classification of objects with very excellent accuracy results. The experimental results obtained, show that the overall classification accuracy offered by the employed rough set approach is high compared with other machine learning techniques including Support Vector Machine (SVM), Hidden Naive Bayesian network (HNB), Bayesian network (BN), Naive Bayesian tree (NBT) [9], Decision tree (DT), Sequential minimal optimization (SMO).

Keywords: Heart sound disease, rough set, dimensional reduction, classification.

Mostafa A. Salama
Department of Computer Science, British University in Egypt, Cairo, Egypt
e-mail: mostafa.salama@gmail.com

Omar S. Soliman,Aboul Ella Hassanien, and Aly A. Fahmy
Faculty of Computers and Information, Cairo University, Egypt
e-mail: dr.omar.soliman@gmail.com, aboitcairo@gmail.com, aly.fahmy@gmail.com

Ilias Maglogiannis
University of Central Greece, Department of Computer Science and Biomedical Informatics, Lamia, Greece
e-mail: imaglo@ucg.gr

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 475–491.
springerlink.com

25.1 Introduction

Heart auscultation, defined as listening and interpretation of the sound produced by the heart, has been a very important method for diagnosing heart diseases from the early stages of medicine, since most heart diseases are reflected to the sound that the heart produces [4, 15]. It is an operationally simple, low cost and non-invasive method of high sensitivity to most heart diseases. Although some new methods, such as Echocardiography, and Medical Imaging modalities (i.e. Ultra sound Imaging (US); Computed Tomography (CT); Magnetic Resonance Imaging, (MRI), etc.), can provide more direct and accurate evidence of heart disease than heart auscultation, these methods require sophisticated and expensive equipment and specialized personnel, being costly and operationally complex [4]. These methods are suitable for use in well organized healthcare environments, but not in small health centers of the rural areas and generally in primary healthcare set-ups. In these healthcare establishments, the heart auscultation remains the basic tool for a first screening of patients and deciding which of them should be referred to more costly medical examinations and tests (e.g. based on advanced imaging techniques) and/or specialized cardiologists. Also, many heart diseases cause differentiations of heart sound in much earlier stages before they can be observed in other comparable techniques, such as the Electrocardiogram(ECG) [15]. Therefore increasing the accuracy and the whole effectiveness of heart auscultation is of critical importance for improving both the health level of the populations (by diagnosing heart diseases in their early stages) and also the economics of the health systems (by avoiding unnecessary costly medical examinations and tests due to incorrect screening). Furthermore, it should be taken into account that in some circumstances, such as in the developing countries, the auscultation is the only available tool for diagnosis of heart diseases for most of their population.

Related work on automated SW tools for diagnosis of heart disease using sound signals may be found in the literature. Maglogiannis et al in [15] have applied different classification algorithms including support vector machine with different parameters to find the best classification accuracy. Due to the high number of features, 100 features, the classification accuracy could be enhanced if non-informative features are removed. Another study have been applied on heart sound data set by Kumar in [10]. In this work, 17 features are extracted including time, frequency and in the state space domain. It have selected only 10 features for performing the classification. The importance of such techniques appears in an application that implements heart sound analysis software that can run in real time on a standard cellphone connected to a hands-free kit [3]. In most of the surveyed systems of the classification staged is preceded by discretization and/or feature selection reducing occasionally the classification accuracy and performance [25]. The main conditions that the selected discretization method should be a supervised based method is to fit the nature of the classification problem. Also the feature selection method should be applied after the discretization method, which shows the dependence on such method for producing appropriate results. If any of these two pre-processing stages fails to

act probably, the results will be deteriorated accordingly. Finally, a classification technique is applied to perform the class label ,disease type, prediction. Numerous machine learning-based classification techniques, especially artificial neural network (ANN) and support vector machine (SVM) [20] have received lots of attentions. But both of the ANN and SVM techniques are black-box models and their generated results are difficult to interpret [15]. Decision tree-based method suffers from the problem that some attributes may be redundant and this could affect negatively on the classification accuracy [18]. Bayesian network assumes the attributes to be independent which not the case in many real life cases [22].

Applying machine learning such as support vector machine or rough sets for the heart disease detection is a very important direction in electronic healthcare as it simplifies and decrease the cost of both, the early diagnosis of the heart diseases and the detection of its types [3]. In this study, the heart diseases under consideration are heart valve diseases categorized into four classes corresponding to the four most usual heart valve diseases: aortic stenosis (AS), aortic regurgitation (AR), mitral stenosis (MS) and mitral regurgitation (MR) [7]. The heart sound is represented by 100 extracted features that is divided into six parts. The implemented feature extraction methodology is presented in Section 25.2.

On the other hand, rough set theory [19] provides the tools that could successfully produce high classification accuracy and generate an interpretable rules. The main goal of the rough set analysis is the induction of approximations of concepts, as it is based on the premise that lowering the degree of precision in the data makes the data pattern more visible. In order to applying classification, first the input data will be discretized using a rough set and boolean reasoning discretization method [16], then rules are generated, and finally classification is applied based on the generated rules. Also the reducts concept in rough set theory allows to keep only the attributes that are not redundant and their removal could not worsen the classification, where this could be a privilege for rough set over decision trees [24]. Rough set put into account the relation among attributes. [23, 26]

On applying rough set discretization and disease prediction on the heart disease data sets, the need of applying feature selection is not required due to the reducts concept, the rough set produces the highest classification accuracy. Also the rules generated that are used in classification are also useful to detect some of the knowledge and facts that exist in the input data set. It is also shown from the resulted rules that the rules generate from the rough set are dependents on the attributes selected by features like chimerge technique and information gain methods. The experimental study shows the results of applying rough set in classification with and without feature selection, also includes comparison between the classification results of different machine learning methods and rough set as a classifier. Finally, it shows the rules generated by rough set and how these rules depends, partially, on features selected by feature selection methods [8].

The rest of this paper is organized as follows: Section 25.2 reviews the basic concepts of the heart sound valve diseases and rough sets. Section 25.3 discusses in details the proposed system and its phases including the pre-processing, analysis

and rule generating and identification and prediction. The experimental results and conclusions are presented in Section 25.4.

25.2 Background Information

25.2.1 The Heart Valve Diseases

A lot of research have been applied on heart sound for the detection of heart valve disease. Features are extracted from the heart sound signal into a data set that is composed of a number of features. Then a classification algorithm is applied on such data set for detection of heart valve disease. Features are extracted in three phases:

- **Phase 1:** In the first phase of the segmentation of the heart sound signals is performed, i.e. the cardiac cycles in every signal are detected by locating the *S*1 and *S*2 peaks. For this purpose, the collected heart sound samples were analyzed with the Wavelet decomposition method described in [11,12], with the only difference being that the 4th and 5th level detail was kept (i.e. frequencies from 34 to 138 Hz), followed by calculation of the normalized average Shannon Energy. Then a morphological transform was applied aiming at the amplification of the sharp peaks and the attenuation of the broad ones [5]. The method described in Ref. [11,12] is used next to locate the peaks corresponding to S1 and S2 and reject the others. Heart sound segmentation was completed with an algorithm that determines the boundaries of *S*1 and *S*2 in each heart cycle, while a method, similar to the one described in Ref. [6], was used to distinguish *S*1 from *S*2 peaks.
- **Phase 2:** In a second phase, for each of the transformed heart sounds that were produced in the first phasewere calculated the standard deviation of the duration of all the heart cycles it includes, the standard deviation of the *S*1 peak values of all heart cycles, the standard deviation of the *S*2 peak values of all heart cycles and the average heart rate. These values are the first four scalar features (F1ıF4) of the feature vector of each heart sound signal.
- **Phase 3:** In a third phase, the rest of the features used for classification are extracted. For this purpose, we calculated for each transformed heart sound signal two mean signals for each of the four structural components of the heart cycle, namely two signals for the *S*1, two for the systolic phase, two for the *S*2 and two for the diastolic phase. The first of these mean signals focused on the frequency characteristics of the heart sound, while the second mean signal focused on the morphological time characteristics of the heart sound. In particular, the first signal is calculated as the mean value of each component, after segmenting and extracting the heart cycle components, time warping them and aligning them. The second signal is calculated as the mean value of the normalized

average Shannon Energy Envelope of each component, after segmenting and extracting the heart cycles components, time warping them and aligning them. The second S1 mean signal is then divided into 8 equal parts, for each part the mean square value is calculated and the resulting 8 values are used as features (F5 - F12). Similarly 24 scalar features for the systolic period (F13 - F36), 8 scalar features for *S*2 (F37 - F44) and 48 scalar features for the diastolic period (F45 - F92) were calculated. Finally, the systolic and diastolic phase components of the above first mean signal were passed from four band-pass filters:

- a 50ı250 Hz filter giving its low frequency content,
- a 100 - 300 Hz filter giving its medium frequency content,
- a 150 - 350 Hz filter giving its mediumŰhigh frequency content and
- a 200 - 400 Hz filter giving its high frequency content. For each of these 8 outputs, the total energy was calculated and was used as a feature in the heart sound vector (F93 - F100).

The above three processing phases result in a heart sound feature vector consisting of 100 components for each signal. These extracted feature are 100 features that represents the four stages of a heart signal which are *S*1 signal, systolic period, *S*2 signal and diastolic period as shown in Figure 25.1. These features are divided into six groups as follows:

- F1:F4 are the standard deviation of all heart sounds, *S*1, *S*2 and average heart rate.
- F5:F12 represents signal *S*1.
- F13:F36 represents the systolic period.
- F37:F44 represents signal *S*2.
- F45:F92 represents the diastolic period.
- F93:F100 the four stage of a heart signal are passed from four band-pass frequency filters. The energy of each output is calculated to form these last 8 features.

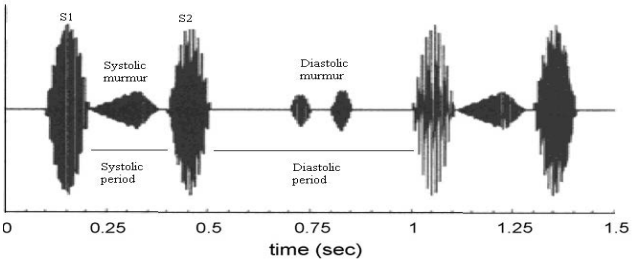


Fig. 25.1 Heart signal: systolic period and diastolic period [15]

25.2.2 Rough Sets: Basics

Rough sets theory proposed by Pawlak [19] is a new intelligent mathematical tool. It is based on the concept of approximation spaces and models of the sets and concepts. In rough sets theory, feature values of sample objects are collected in what are known as information tables. Rows of a such a table correspond to objects and columns correspond to object features.

Let $\mathcal{O}, \mathcal{F}$ denote a set of sample objects and a set of functions representing object features, respectively. Assume that $B \subseteq \mathcal{F}, x \in \mathcal{O}$. Further, let $x_{\sim_B}$ denote $x_{/\sim_B} = \{y \in \mathcal{O} \mid \forall \phi \in B, \phi(x) = \phi(y)\}$, *i.e.*, $x_{/\sim_B}$ (description of x matches the description of y). Rough sets theory defines three regions based on the equivalent classes induced by the feature values: lower approximation $\underline{B}X$, upper approximation $\overline{B}X$ and boundary $BND_B(X)$. A lower approximation of a set X contains all equivalence classes $x_{/\sim_B}$ that are proper subsets of X, and upper approximation $\overline{B}X$ contains all equivalence classes $x_{/\sim_B}$ that have objects in common with X, while the boundary $BND_B(X)$ is the set $\overline{B}X \setminus \underline{B}X$, *i.e.*, the set of all objects in $\overline{B}X$ that are not contained in $\underline{B}X$.

Some strategies for discrimination of real-valued features must be used when we need to apply learning strategies for data classification (*e.g.*, equal width and equal frequency intervals). It has been shown that the quality of learning algorithm is dependent on this strategy, which has been used for real-valued data discritization. In the context of supervised learning, an important task is the discovery of classification rules from the data provided in the decision tables. The decision rules not only capture patterns hidden in the data as they can also be used to classify new unseen objects. Rules represent dependencies in the data set, and represent extracted knowledge which can be used when classifying new objects not in the original information system. When the reducts were found, the job of creating definite rules for the value of the decision feature of the information system was practically done. To transform reduct into a rule, one only has to bind the condition feature values of the object class from which the reduct originated to the corresponding features of the reduct. Then, to complete the rule, a decision part comprising the resulting part of the rule is added. This is done in the same way as for the condition features. To classify objects, which has never been seen before, rules generated from a training set will be used. These rules represent the actual classifier. This classifier is used to predict to which classes new objects are attached. The nearest matching rule is determined as the one whose condition part differs from the feature vector of re-image by the minimum number of features. When there is more than one matching rule, we use a voting mechanism to choose the decision value. Every matched rule contributes votes to its decision value, which are equal to the times number of objects matched by the rule. The votes are added and the decision with the largest number of votes is chosen as the correct class. Quality measures associated with decision rules can be used to eliminate some of the decision rules.

25.3 The Proposed Rough Set-Based Identification of Heart Valve Diseases System

Figure 25.2 illustrates the overall steps of the proposed identification of heart valve diseases system. It is composed of three consecutive phases, which are elaborated in the following subsections. These phases are:

- **Pre-processing phase.** This phase includes tasks such as extra variables addition and computation, decision classes assignments, data cleansing, completeness, correctness, feature creation, feature selection, feature evaluation and discretization.
- **Analysis and rule generating phase**. This phase includes the generation of preliminary knowledge, such as computation of object reducts from data, derivation of rules from reducts, rule evaluation and prediction processes.
- **Identification and prediction phase**. This phase utilizes the rules generated from the previous phase to predict the stock price movement.

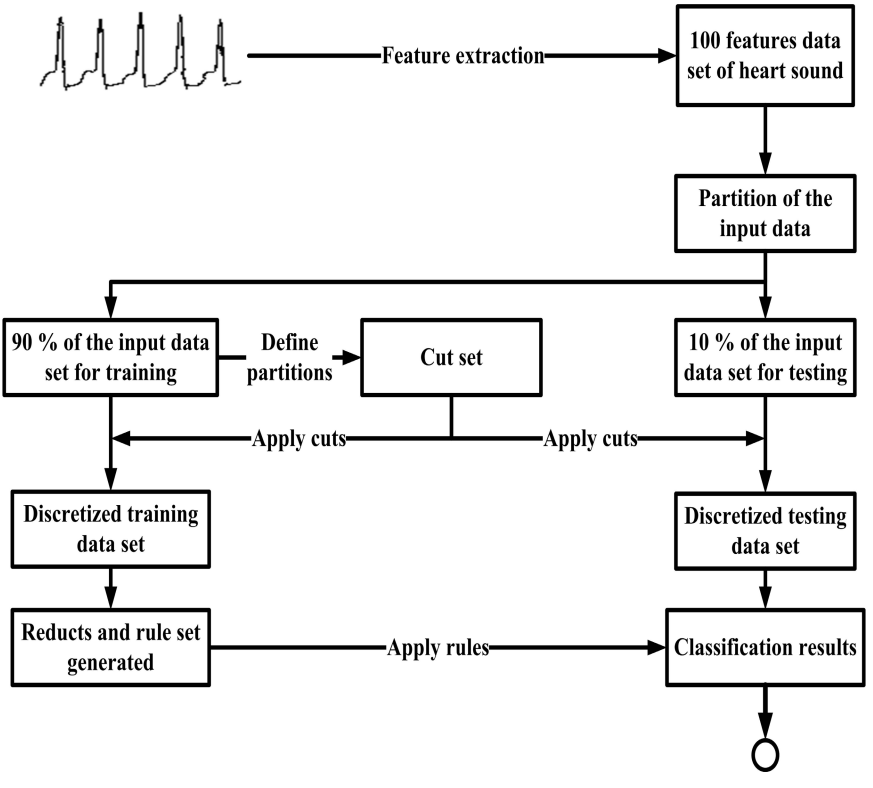


Fig. 25.2 Rough set-based identification of heart valve diseases system

25.3.1 Pre-processing Phase

25.3.1.1 Feature Reduction

The reducts resulted in rough set technique is compared to a feature selection technique. One of the most popular feature selection methods is the chi-Square χ^2 method, which measures the lack of independence between each feature A and the target class c. ChiMerge or Chi2-Square [13, 14] is a χ^2-based discretization method. The reason of using this method is due to the nature of extracted features of heart sound, which represents a continuous feature. Nearly most of the feature selection method discretize the continuous feature which could leads to the distortion of data and loose of its characteristics [22] and hence the decrease feature classification. Chimerge technique determines the Chi-Square χ^2 value while perform the discrimination of features which leads to more accurate results. It uses the χ^2 statistic to discretize numeric features repeatedly until some inconsistencies are found in the data that achieves feature selection via discretization. The χ^2 value is calculated for each continuous feature as follows: Initially, each distinct value of a numeric feature A is considered to be one interval. The values,intervals, of feature A are sorted and the χ^2 is applied for every pair of adjacent intervals as follows:

$$\chi^2 = \sum_{i:1..2} \sum_{j:1..k} \frac{(A_{ij} - E_{ij})^2}{E_{ij}} \tag{25.1}$$

Where:

- A_{ij} is the number of values in the ith interval and jth class,
- R_{ij} is the number of values in the jth class = $\sum_{j:1..k} A_{ij}$,
- C_{ij} is the number of values in the ith interval = $\sum_{i:1..2} A_{ij}$,
- N is the total number of values = $\sum_{i:1..2} R_{ij}$,and
- E_{ij} is the expected frequency of $A_{ij} = \frac{R_{ij} * C_{ij}}{N}$

Adjacent intervals with the least χ^2 values are merged together, because low χ^2 values for a pair indicates similar class distributions. This merging process proceeds recursively until all χ^2 values of all pairs exceeds a parameter *signlevel* (initially 0.5). Then repeat the previous steps with a decreasing *signlevel* until an inconsistency rate is exceeded, where two patterns are the same but classified into different categories.

25.3.1.2 Discretization Based on RSBR

A real world data set, like medical data sets, contains mixed types of data including continuous and discrete valued data sets. The discretization process divides the attributeŠs value into intervals [2]. The discretization based on RS and Boolean Reasoning (RSBR) proposed in [17], [16] shows the best results in the case of heart disease data set. In the discretization of a decision table $S = (U, A \bigcap \{d\})$, where U is a non-empty finite set of objects and A is a non-empty finite set of attributes. And $V_a = [x_a, x_a)$ is an interval of real values x_a, w_a in attribute a. The required is to a partition P_a of V_a for any $a \in A$. Any partition of V_a is defined by a sequence of the so-called cuts $x_1 < x_2 < .. < x_k$ from V_a. The main steps of the RSBR discretization algorithm are provided in algorithm 25.1.

Algorithm 25.1. RSBR discretization algorithm [1]

Input: Information system table (S) with real valued attributes A_{ij} and n is the number of intervals for each attribute.
Output: Information table (ST) with discretized real valued attribute

1: **for** $A_{ij} \in S$ **do**
2: Define a set of boolean variables as follows:

$$B = \{\sum_{i=1}^{n} C_{ai}, \sum_{i=1}^{n} C_{bi} \sum_{i=1}^{n} C_{ci}, ..., \sum_{i=1}^{n} C_{ni}\} \tag{25.2}$$

3: **end for**
4: Where $\sum_{i=1}^{n} C_{ai}$ correspond to a set of intervals defined on the variables of attributes a
5: Create a new information table S_{new} by using the set of intervals C_{ai}
6: Find the minimal subset of C_{ai} that discerns all the objects in the decision class D using the following formula:

$$\Upsilon^u = \wedge\{\Phi(i,j) : d(x_i \neq d(x_j)\} \tag{25.3}$$

7: Where $\Phi(i,j)$ is the number of minimal cuts that must be used to discern two different instances x_i and x_j in the information table.

25.3.2 *Analysis and Rule Generating Phase*

Unseen instances are considered in the discovery process, and the uncertainty of a rule, including its ability to predict possible instances, can be explicitly represented in the strength of the rule. The rule generation from the extracted reducts is

maintained in a method named in [27]. The quality of rules is related to the corresponding reduct(s). We are especially interested in generating rules which cover largest parts of the universe U. Covering U with more general rules implies smaller size of a rule set. The algorithm in 25.2 presents the soft hybrid induction system GDT-RS constituting the core in the discovery of classification rules from databases with uncertain and incomplete data.

Algorithm 25.2. Rule generation and classification [2]

Input: reduct sets $R_{final} = \{r_1 \cup r_2 \cup .... \cup r_n\}$
Output: Set of rules

for each reduct r **do**
  for each correspondence object x **do**
    Contract the decision rule $(c_1 = v_1 \wedge c_2 = v_2 \wedge .... \wedge c_n = v_n) \longrightarrow d = u$
    Scan the reduct r over an object x
    Construct $(c_i, 1 \leq i \leq n)$
    for every $c \in C$ **do**
      Assign the value v to the correspondence attribute a
    end for
    Construct a decision attribute d
    Assign the value u to the correspondence decision attribute d
  end for
end for

25.3.2.1 Identification and Prediction Phase

Classification and prediction are the last phase of our proposed approach. To transform a reduct into a rule, one only has to bind the condition feature values of the object class from which the reduct originated to the corresponding features of the reduct. Then, to complete the rule, a decision part comprising the resulting part of the rule is added. This is done in the same way as for the condition features. To classify objects which have never been seen before, the rules generated from a training set will be used. These rules represent the actual classifier. This classifier is used to predict to which classes the new objects are attached. The nearest matching rule is determined as the one whose condition part differs from the feature vector of the re-object by the minimum number of features. When there is more than one matching rule, we use a voting mechanism to choose the decision value. Every matched rule contributes votes to its decision value, which are equal to the t times number of the objects matched by the rule. The votes are added and the decision with the largest number of votes is chosen as correct class. Quality measures associated with decision rules can be used to eliminate some of the decision rules. The global strength

defined in [1] for rule negotiation is a rational number in [0, 1] representing the importance of the sets of decision rules relative to the considered tested object.

Let us assume that $T = (U; A \bigcup d)$ is a given decision table, u_t is a test object, $Rul(X_j)$ is the set of all calculated basic decision rules for T, classifying objects to the decision class $X_j(v_d^j = v_d), MRul(X_j; u_t) \subseteq Rul(X_j)$ is the set of all decision rules from $Rul(X_j)$ matching tested object u_t. The global strength of the decision rule set $MRul(X_j; u_t)$ is defined as given in [1]. The measure of the strengths of the rules defined above is applied in constructing classification algorithm. To classify a new case, rules are first selected matching the new object. The strength of the selected rule sets is calculated for any decision class, and then the decision class with maximal strength is selected with the new object being classified to this class.

25.4 Experimental Results and Discussion

25.4.1 The Heart Sound: Data Sets Declaration

The identification of heart valve diseases proposed system were applied on three different data sets of heart sound signals with the same number of instances in every class. The first data set "*HS_AS_MR*" is about systolic diseases where it contains 37 instances of aortic stenosis *AS* cases and 37 instances of mitral regurgitation *MR* cases. The second data set "*HS_AR_MS*" is about diastolic diseases where it contains 37 instances of aortic regurgitation *AR* cases and 37 instances of mitral stenosis *MS* cases. The third data set "*HS_N_S*" contains 64 instance, where 32 instances represent healthy patients and the other 32 represents unhealthy, murmur diseased patients. The tool that has been used in this chapter is the rough set exploration system tool [21]

25.4.2 Analysis, Results and Discussion

25.4.2.1 The Set of Reducts in Comparison to the Chimerge Feature Selection Technique

For each data set, we reach the minimal number of reducts that contains a combination of attributes which have the same discrimination factor for each data set. Table 25.1 shows the final generated reducts for each data set, which are used to generate the list of the rules for the classification.

Table 25.1 Rough reducts sets of the three data sets

Data set	Reduct sets
HS_AR_MS	3, 8, 31, 38,82
HS_AS_MR	3, 6, 36,39
HS_N_S	1, 9, 87, 94, 97

In order to evaluate the proposed rough sets classifier, we will study the lets discuss the reducts in comparison to the feature selection using chimerge approach and find the results of the classifiers after feature selection for each data set.

For the first data set *HS_AR_MS*, the selected number of features by the chimerge technique are:

{F32, {**F31**}, F30, F29, F100, F28, F33, F27, {**F3**}, F5, F4, F49, F48, F45, F46, F50, F25, F26}.

It noticed that the selected features by the chimerge technique includes feature $F31$ and feature $F3$, where feature $F31$ is the *second* most important feature.

In regard to the second data set *HS_AS_MR*, the selected features using the chimerge technique are:

{{**F36**}, F13, F12, F14, F15, F35, F4, F2, F18, F17, F11, {**F3**}, F92, F5, F16, F20, F19, F23, F21, F93, F54, F22, F95, F10, F1, F94, F24, F28, F37, F25, F41, F29, F100, F40, F26, F88, F55, F96}.

It noticed that the selected features by the chimerge technique includes feature $F36$ and feature $F3$, where feature $F36$ is the *second* most important feature. Also it is noticed that in both data sets, *HS_AR_MS* and *HS_AS_MR* data sets, feature $F3$ is selected and it is common in chimerge selected features and reducts.

For the third data set "*HS_N_S*", the selected features using the chimerge technique are as follows:

{F54, F58, F53, F65, F64, F67, F59, {**F97**}, F61, F70, F96, F98, F55, F60, F66, F51, F52, F49, F62, F63, F23, F22, F45, F50, F2, F56, F57, F71, F28, F27, F24, F26, F25, F47, F46, F99, F21, F72, F68, F69, F3, F75, F73, F92, F48, {**F87**}, F7, F76}.

It noticed that the selected features by the chimerge technique includes feature $F97$ and feature $F87$, where feature $F97$ is the $eight^{th}$ most important feature. Also it is noticed that feature $F3$ is appears in all the reducts in the three data sets.

25.4.2.2 The Set of Extracted Rules

The extracted rules from *HS_AR_MS*, *HS_AS_MR* and *HS_N_S* data sets are shown in Tables 25.2, 25.3 and 25.4, respectively.

Table 25.2 Generated rules for the *HS_AR_MS* data set

Matches	Decision rules
22	(3="(0.590,Inf)")&(31="(0.004,Inf)")&(38="(0.01,Inf)")=>(D=1[22])
22	(31="(0.00,Inf)")&(38="(0.01,Inf)")&(82="(-Inf,0.00)")=>(D=1[22])
21	(3="(-Inf,0.59)")&(31="(-Inf,0.004)")=>(D=0[21])
20	(8="(0.04,Inf)")&(31="(0.004,Inf)")&(38="(0.012,Inf)")=>(D=1[20])
12	(8="(0.04,Inf)")&(31="(-Inf,0.004)")=>(D=0[12])
11	(3="(0.59,Inf)")&(8="(-Inf,0.04)")&(82="(-Inf,0.00)")=>(D=1[11])
11	(8="(-Inf,0.04)")&(31="(0.004,Inf)")&(82="(-Inf,0.00)")=>(D=1[11])
9	(3="(-Inf,0.59)")&(8="(-Inf,0.04)")&(82="(0.00,Inf)")=>(D=0[9])
9	(3="(0.59,Inf)")&(8="(-Inf,0.04)")&(31="(0.004,Inf)")=>(D=1[9])
8	(3="(-Inf,0.59)")&(38="(-Inf,0.01)")&(82="(0.00,Inf)")=>(D=0[8])
7	(3="(-Inf,0.59)")&(8="(0.04,Inf)")&(38="(-Inf,0.01)")=>(D=0[7])
6	(3="(0.59,Inf)")&(31="(0.004,Inf)")&(82="(0.00,Inf)")=>(D=1[6])
5	(31="(-Inf,0.004)")&(38="(0.01,Inf)")&(82="(0.00,Inf)")=>(D=0[5])
5	(3="(0.59,Inf)")&(8="(0.04,Inf)")&(82="(0.00,Inf)")=>(D=1[5])
4	(8="(0.04,Inf)")&(38="(-Inf,0.01)")&(82="(-Inf,0.00)")=>(D=0[4])
3	(3="(0.59,Inf)")&(8="(-Inf,0.04)")&(38="(-Inf,0.01)")=>(D=1[3])
3	(3="(0.59,Inf)")&(31="(-Inf,0.004)")&(38="(-Inf,0.01)")=>(D=1[3])
2	(3="(0.59,Inf)")&(38="(-Inf,0.01)")&(82="(0.00,Inf)")=>(D=1[2])

Table 25.3 Generated rules for the *HS_AS_MR* data set

Matches	Decision rules
26	(3="(-Inf,0.31)")&(36="(0.047,Inf)")=>(D=0[26])
25	(6="(0.05,Inf)")&(36="(-Inf,0.04)")=>(D=1[25])
18	(3="(-Inf,0.31)")&(6="(-Inf,0.05)")=>(D=0[18])
14	(3="(0.31)")&(39="(0.001,Inf)")=>(D=1[14])
13	(3="(0.31,0.76)")&(36="(-Inf,0.04)")=>(D=1[13])
11	(3="(0.76,Inf)")&(36="(-Inf,0.04)")=>(D=1[11])
6	(3="(0.31)")&(6="(-Inf,0.05)")=>(D=1[6])
5	(3="(0.76,Inf)")&(6="(0.05,Inf)")&(36="(0.04,Inf)")=>(D=0[5])
3	(3="(0.76,Inf)")&(6="(-Inf,0.05)")&(39="(0.001,Inf)")=>(D=1[3])
2	(6="(-Inf,0.05)")&(36="(-Inf,0.04)")&(39="(-Inf,0.001)")=>(D=0[2])
2	(3="(0.76,Inf)")&(39="(-Inf,0.001)")=>(D=0[2])
2	(6="(0.05,Inf)")&(36="(0.04,Inf)")&(39="(-Inf,0.001)")=>(D=0[2])

25.4.2.3 Classification Accuracy of the Proposed Model in Comparison to the Other Classification Techniques

The comparison shown in table 25.5 contains a comparative study between the proposed model of rough set classifier and other conventional and known classifiers. Feature selection will be applied before the the conventional classifier in order to

Table 25.4 Generated rules for the *HS_N_S* data set

Matches	Decision rules
20	(94="(0.16,Inf)")&(97="(0.04,Inf)")=>(D=1[20])
12	(94="(-Inf,0.16)")&(97="(-Inf,0.04)")=>(D=0[12])
9	(1="(0.12,Inf)")&(9="(0.00,Inf)")&(97="(-Inf,0.04)")=>(D=0[9])
8	(9="(0.00,Inf)")&(87="(-Inf,0.00)")&(97="(-Inf,0.04)")=>(D=0[8])
8	(1="(0.12,Inf)")&(87="(0.00,Inf)")&(94="(-Inf,0.16)")=>(D=0[8])
8	(9="(-Inf,0.00)")&(87="(-Inf,0.00)")&(97="(0.04,Inf)")=>(D=1[8])
7	(1="(0.12,Inf)")&(87="(0.00,Inf)")&(97="(-Inf,0.04)")=>(D=0[7])
6	(9="(-Inf,0.00)")&(87="(0.00,Inf)")&(94="(-Inf,0.16)")=>(D=0[6])
4	(1="(0.12,Inf)")&(87="(-Inf,0.00)")&(97="(0.04,Inf)")=>(D=1[4])
3	(9="(-Inf,0.00)")&(87="(0.00,Inf)")&(97="(-Inf,0.04)")=>(D=0[3])
3	(9="(-Inf,0.00)")&(87="(-Inf,0.00)")&(94="(0.16,Inf)")=>(D=1[3])

enhance the corresponding percentage of classification accuracy. On the other hand, rough set model will not preceded by feature selection, where it will depends only the set of generated reducts. For example, in the case of *HS_AS_MR* data set, the classification accuracy of DT and SVM after feature selection are 89.0 and 94.5 respectively, where these results are still less than that of rough set accuracy results. Although, Sequential minimal optimization, (*SMO*) has the percentage of classification accuracy slightly greater than the Rough set system in the case of *HS_AS_MR* data set, rough set model has a greater percentage of classification accuracy in *HS_AR_MS* and *HS_N_S* data sets.

Table 25.5 Accuracy results: Comparative analysis among Hidden Naive Bayesian network (HNB), Bayesian network (BN), Naive Bayesian tree (NBT), Decision tree (DT), Sequential minimal optimization (SMO)

Classifier	*HS_AS_MR*	*HS_AR_MS*	*HS_N_S*
SVM	90.54	90.54	85.93
HNB	90.54	91.89	**90.625**
BN	86.48	83.78	84.37
DT	89.18	83.78	82.81
SMO	**93.24**	94.59	87.50
NBT	87.83	89.18	89.06
RS	**92.90**	**97.1**	**90.0**

Figure 25.3 illustrates the overall rough sets classification accuracy in terms of sensitivity and specificity compared with Hidden Naive Bayesian network (HNB), Bayesian network (BN), Naive Bayesian tree (NBT), Decision tree (DT), Sequential minimal optimization (SMO). Empirical results reveal that the proposed rough set approach performs better than the other classifiers.

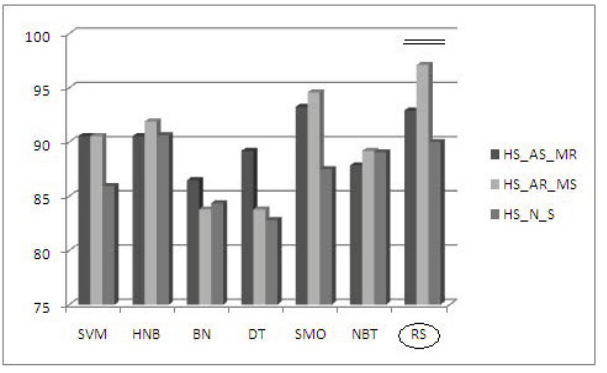


Fig. 25.3 Classification accuracy: Comparative analysis among Support Vector Machine (SVM), Hidden Naive Bayesian network (HNB), Bayesian network (BN), Naive Bayesian tree (NBT), Decision tree (DT), Sequential minimal optimization (SMO)

25.5 Conclusions

Heart sound data sets represents a real life data that contains continuous attributes and a large number of features that could be hardly classified by most of classification techniques. This chapter introduces the ability of rough set theory to successfully classify heart sound diseases without the need for applying feature selection. Discretizing the raw heart sound data and applying a feature reduction approach should be applied prior any classifier to increase the classification and prediction accuracy results. The experimental results obtained, show that the overall classification accuracy offered by the employed rough set approach is high compared with other machine learning techniques including Support Vector Machine (SVM), Hidden Naive Bayesian network (HNB), Bayesian network (BN), Naive Bayesian tree (NBT), Decision tree (DT), Sequential minimal optimization (SMO).

References

1. Al-Qaheri, H., Hassanien, A.E., Abraham, A.: A Generic Scheme for Generating Prediction Rules Using Rough Sets. In: Abraham, A., Falcón, R., Bello, R. (eds.) Rough Set Theory. SCI, vol. 174, pp. 163–186. Springer, Heidelberg (2009)
2. Chao, S., Li, Y.: Multivariate interdependent discretization for continuous attribute. In: Proceeding of the 3rd International Conference on Information Technology and Applications, vol. 1, pp. 167–172. IEEE Computer Society Press (2005)
3. Chen, T., Kuan, K., Celi, L., Clifford, G.: Intelligent heartsound diagnostics on a cell phone using a hands-free kit. In: Proceeding of AAAI Artificial Intelligence for Development, AI-D 2010, pp. 26–31. Association for the Advancement of Artificial Intelligence (2010)
4. Ericson, B.: Heart sounds and murmurs: A practical guide. Mosby-Year Book Inc. (1997)
5. Haghighi-Mood, A., Torry, J.N.: A sub-band energy tracking algorithm for heart sound segmentation. In: Computers in Cardiology, pp. 501–504. Computer Society Press (1995)

6. Hebden, J.E., Torry, J.N.: Neural network and conventional classifiers to distinguish between first and second heart sounds. Artificial Intelligence Methods for Biomedical Data Processing IEE Colloquium (Digest) 3, 1–6 (1996)
7. Higuchi, K., Sato, K., Makuuchi, H., Furuse, A., Takamoto, S., Takeda, H.: Automated diagnosis of heart disease in patients with heart murmurs: application of a neural network technique. Journal of Medical Engineering and Technology 30, 61–68 (2006)
8. Janecek, A.G.K., Gansterer, W.N., Demel, M., Ecker, G.F.: On the relationship between feature selection and classification accuracy. Journal of Machine Learning Research - Proceedings Track 4, 90–105 (2008)
9. Kohavi, R.: Scaling up the accuracy of Naive-bayes classifiers: a decision-tree hybrid. In: Simoudis, E., Han, J., Fayyad, U.M. (eds.) Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD 1996), pp. 202–207. AAAI Press (1996)
10. Kumar, D., Carvalho, P., Antunes, M., Paiva, R.P., Henriques, J.: Heart murmur classification with feature selection. In: Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, vol. 1, pp. 4566–4569. Computer Society Press (2010)
11. Liang, H., Lukkarinen, S., Hartimo, I.: A heart sound segmentation algorithm using wavelet decomposition and reconstruction. In: Proceedings of the 19th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, vol. 4, pp. 1630–1633. Computer Society Press (1997)
12. Liang, H., Lukkarinen, S., Hartimo, I.: Heart sound segmentation algorithm based on heart sound envelogram. In: Computers in Cardiology, pp. 105–108. IEEE Computer Society Press (1997)
13. Liu, H., Setiono, R.: Chi2: Feature selection and discretization of numeric attributes. In: Proceedings of the Seventh International Conference on Tools with Artificial Intelligence, Virginia, USA, November 8, pp. 388–391. IEEE Computer Society Press (1995)
14. Liu, H., Setiono, R.: Feature selection via discretization. IEEE Transactions on Knowledge and Data Engineering 9, 642–645 (1997)
15. Maglogiannis, I., Loukis, E., Zafiropoulos, E., Stasis, A.: Support vectors machine-based identification of heart valve diseases using heart sounds. Comput. Methods Programs Biomed 95(1), 47–61 (2009)
16. Nguyen, H.S., Nguyen, S.H.: Discretization methods in data mining. In: Polkowski, L., Skowron, A. (eds.) Rough Sets in Knowledge Discovery, vol. 1, pp. 451–482. Physica-Verlag, Heidelberg (1998)
17. Nguyen, H.S., Skowron, A.: Quantization of real value attributes. In: Proceedings of the 2nd Joint Conference of Information Sciences (JCIS 1995), Durham, NC, pp. 34–37 (1995)
18. Pavlopoulos, S., Stasis, A., Loukis, E.: A decision tree-based method for the differential diagnosis of aortic stenosis from mitral regurgitation using heart sounds. BioMedical Engineering OnLine 3(21), 1–15 (2004)
19. Pawlak, Z.: Rough set approach to knowledge-based decision support. European Journal of Operational Research 99, 48–57 (1997)
20. Platt, J.C.: Sequential minimal optimization: A fast algorithm for training support vector machines. In: Schölkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods - Support Vector Learning, pp. 185–208. MIT Press (1999)
21. Rough Set Exploration System (RSES), Group of Logic, Institute of Mathematics, Warsaw University, Poland, `http://logic.mimuw.edu.pl/~rses/`
22. Salama, M.A., Hassanien, A.E., Fahmy, A.A.: Uni-class pattern-based classification model. In: Proceeding of the 10th IEEE International Conference on Intelligent Systems Design and Applications, Cairo, Egypt, pp. 1293–1297. IEEE Computer Society Press (December 2010)

23. Soliman, O.S., Hassanien, A.E., El-Bendary, N.: A rough clustering algorithm based on entropy information. In: Corchado, E., Snášel, V., Sedano, J., Hassanien, A.E., Calvo, J.L., Ślęzak, D. (eds.) SOCO 2011. AISC, vol. 87, pp. 213–222. Springer, Heidelberg (2011)
24. Swiniarski, R., Skowron, A.: Rough set methods in feature selection and extraction. Pattern Recognition Letters 24(6), 833–849 (2003)
25. Vyas, O.P., Das, K.: A suitability study of discretization methods for associative classifiers. International Journal of Computer Applications 5(10), 46–51 (2010)
26. Zhang, M., Yao, J.T.: A rough sets based approach to feature selection. In: Proceedings of The 23rd International Conference of NAFIPS, Banff, Canada, June 27-30, pp. 434–439 (2004)
27. Zhong, N., Dong, J.Z., Ohsuga, S.: Data mining: A probabilistic rough set approach. In: Rough Sets in Knowledge Discovery, vol. 2, pp. 127–146. Physica-Verlag, Heidelberg (1998)

Chapter 26
Rough Sets and Neuroscience

Tomasz G. Smolinski and Astrid A. Prinz

Abstract. It has been almost exactly 10 years since the publication of the Neurocomputing Special Volume on Rough-Neuro Computing, and nearly 8 years since the seminal book "Rough-Neural Computing" came out. Rough-Neuro (or Neural) Computing (RNC) generalizes traditional artificial neural networks by incorporating the concepts of information granularity and computing with words. It provides solid theoretical foundations for hybridization of neural computing with the theory of rough sets, as well as rough mereology, and has many interesting practical applications. Interestingly, while the RNC paradigms directly or indirectly draw extensively from the field of neuroscience, not many applications of the theory of rough sets (in the form of RNC or otherwise) to solve problems in that field exist. This is somewhat surprising as many problems in neuroscience are inherently vague and/or ill-defined and could potentially significantly benefit from the rough sets' ability to deal with imprecise data, and those applications that have been proposed, have been very successful. In this chapter, we describe a few examples of the existing applications of the theory of rough sets (and its hybridizations) in the field of neuroscience and its clinical "sister," neurology. We also provide a discussion of other potential applications of rough sets in those areas. Finally, we speculate on how the new insights into the field of neuroscience derived with the help of rough sets may help improve RNC, thus closing the loop between the two fields.

Keywords: Rough sets, rough-neuro computing, neuroscience, neurology, classificatory decomposition, neuronal modeling.

Tomasz G. Smolinski
Department of Computer and Information Sciences, Delaware State University
1200 N. DuPont Highway, Dover, DE 19901, USA
e-mail: tsmolinski@desu.edu

Astrid A. Prinz
Department of Biology, Emory University
1510 Clifton Rd. NE, Atlanta, GA 30322, USA
e-mail: aprinz@emory.edu

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 493–514.
springerlink.com

26.1 Introduction

Rough-Neuro (or Neural) Computing (RNC) generalizes traditional artificial neural networks by incorporating the concepts of information granularity and computing with words. It provides solid theoretical foundations for hybridization of neural computing with the theory of rough sets [33], as well as rough mereology [34], and has many interesting practical applications. The applications include, for example, acquisition of audio signals [4], analysis of dyslexia of Kanji characters [53], financial time series prediction [43], or even XML prefetching for accessing Facebook from mobile environments [51]. Interestingly, while the RNC paradigms directly or indirectly draw extensively from the field of neuroscience, not many applications of the theory of rough sets (in the form of RNC or otherwise) to solving problems in that field exist. This is somewhat surprising as many problems in neuroscience are inherently vague and/or ill-defined and could potentially significantly benefit from the rough sets' ability to deal with imprecise data. Furthermore, those applications that have been proposed, have been very successful. For example, Szczuka and Wojdyłło implemented a rough sets-based neuro-wavelet classifier for analysis of EEG signals [52], Smolinski, *et al.*, designed a methodology for classificatory decomposition of cortical evoked potentials based on hybridization of rough sets and multi-objective evolutionary algorithms [42,45,46] and a system to classify neuronal models using rough sets [49], and Przybyszewski developed a framework for cognitive computation using the rough set theory [36].

This chapter is organized as follows: first, we provide some background information on the theory of rough sets and selected branches of the field of neuroscience. Then, we review a few existing applications of rough sets in the field of neuroscience and its clinical "sister," neurology. This is followed by a discussion of potential applications of rough sets in those areas. Finally, we speculate on how the new insights into the field of neuroscience derived with the help of rough sets may help improve RNC, thus closing the loop between the two fields.

26.2 Background

26.2.1 Theory of Rough Sets

The theory of rough sets (RS) deals with the classificatory analysis of data tables [31]. The main idea behind it is the so-called indiscernibility relation that describes objects indistinguishable from one another [23,32]. The main goal of rough set analysis is to synthesize approximation of concepts from acquired data. The concepts are represented by lower and upper approximations [18].

26.2.1.1 Information Systems and Decision Tables

In the theory of rough sets, a dataset is in a form of a table, in which each row represents an object, and every column represents an attribute that can be observed or measured for that object. Such a table is called an information system, which can be represented formally as a pair: $IS = (U,\ A)$, where U is a non-empty finite set of objects (universe) and A is a non-empty finite set of attributes such that $a : U \Rightarrow Va$ for every $a \in A$. The V_a is called the value set of a.

In many situations, some classification of objects is given *a priori*. This knowledge is expressed by one or more distinguished attributes called the decision attributes. Information systems of this kind are called decision tables. Such a system can be formally represented by $DT = (U,\ A \cup d)$, where $d \notin A$ is the decision attribute and the elements of A are the conditional attributes. An example of a decision table actually utilized in one of the projects described later in this chapter is shown in Fig. 26.5.

26.2.1.2 Indiscernibility

A decision table represents the entire available knowledge about a given problem. However, the table may be unnecessarily large because it is redundant in at least two ways: "vertical" (*i.e.*, object-wise) and "horizontal" (*i.e.*, attribute-wise). In other words, the same, or indistinguishable from one another, objects may be represented several times, or some of the attributes may be overabundant.

In order to take a closer look at those issues, let us first recall the notion of equivalence. A binary relation $R \subseteq X \times X$ that is reflexive (*i.e.*, an object is in relation with itself xRx), symmetric (if xRy, then yRx) and transitive (if xRy and yRz, then xRz) is called an equivalence relation. This relation divides a given set of elements (objects) into a certain number of disjoint equivalence classes. The equivalence class of an element $x \in X$ consists of all objects $y \in X$ such that xRy.

Let $IS = (U,\ A)$ be an information system, then for any $B \in A$ there exists an equivalence relation $IND_{IS}(B)$, as shown in Eq. 26.1.

$$IND_{IS}(B) = \left\{(x,x') \subset U^2 \middle| \forall a \in B, a(x) = a(x')\right\}. \tag{26.1}$$

If $(x,x') \in IND_{IS}(B)$, the so-called B-indiscernibility relation, then objects x and x' are indistinguishable from each other, in light of the attributes in B.

26.2.1.3 Set Approximation

The indiscernibility relation induces a partitioning of the universe, by dividing it into disjoint equivalence classes, denoted as $[x]_B$. These partitions can be used to build new subsets of the universe. Usually, those subsets that contain objects that

belong to the same decision class are of interest. However, it is possible that a decision class (or concept) cannot be defined in a crisp manner. The main goal of rough set analysis is to synthesize approximations of such concepts from acquired data. The concepts are represented by lower and upper approximations. Although it may be impossible to precisely define some concept X, it may be possible to approximate it using the information contained in B by constructing the B-lower and B-upper approximations of X, denoted by $\underline{B}X$ and $\overline{B}X$ respectively, where, $\underline{B}X = \left\{x \middle| [x]_B \subseteq X\right\}$ and $\overline{B}X = \left\{x \middle| [x]_B \cap X \neq \emptyset\right\}$. Only the objects in $\underline{B}X$ can be classified as members of X with certainty, based on the knowledge conveyed by B. Consequently, only the objects located in $U - \overline{B}X$ (*i.e.*, B-outside region), can be classified with certainty as those not belonging to the concept X. Figure 26.1 illustrates the idea of rough sets-based approximation with a synthetic, two-dimensional example.

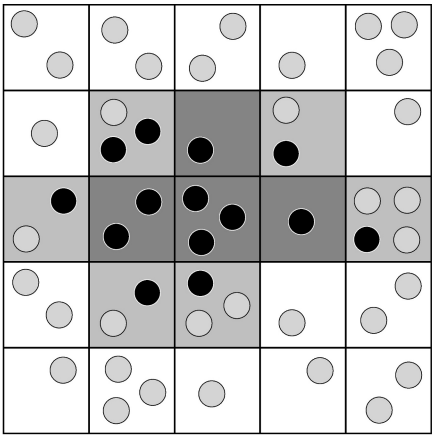


Fig. 26.1 An illustration of rough sets-based set approximation using two fictitious attributes (*i.e.*, dimensions) B. The squares represent disjoint equivalence classes, *i.e.*, circles in each square are considered indiscernible from one another in light of the two attributes. The black circles represent data points belonging to some concept X. The concept is approximated by its B-lower (dark gray squares) and B-upper (light gray squares and dark gray squares combined) approximations. The white squares represent equivalence classes constituting the outside region, *i.e.*, objects that certainly do not belong to the concept X.

A rough set can be characterized numerically by the accuracy of approximation coefficient:

$$\alpha_B(X) = \frac{card(\underline{B}X)}{card(\overline{B}X)}, \tag{26.2}$$

or the quality of classification coefficient:

$$\gamma_B(X) = \frac{card(\underline{B}X \cup \underline{B}\neg X)}{card(U)}, \tag{26.3}$$

where $\underline{B}X$ is the lower approximation of X, $\overline{B}X$ is the upper approximation of X, $\underline{B}\neg X$ is the lower approximation of the set of objects that do not belong to X, U is the set of all objects, and *card* stands for cardinality of a set.

26.2.1.4 Reducts

All the considerations discussed so far represent one way of reducing data by identifying equivalence classes, *i.e.*, objects that are indistinguishable using the available attributes. That is a much more efficient representation, since only one element of the equivalence class is needed to characterize the entire class. The other consideration in terms of data reduction is to keep only those attributes that preserve the indiscernibility relation and, consequently, the set approximation. The rejected attributes are redundant since their removal cannot worsen the classification.

There are usually several such subsets of attributes and those that are minimal are called reducts. The following presents a formal definition of a reduct: a "minimal" $R \subseteq A$, such that:

$$\forall k,n \; \forall a_i \in R : \big((a_i(o_k) = a_i(o_n)) \Rightarrow (d(o_k) = d(o_n))\big), \tag{26.4}$$

where $k,n = 1..N$ (N is the number of objects in the decision table), $o_{k|n}$ is the $k^{th}|n^{th}$ object, $d(o_{k|n})$ is the value of the decision attribute for the $k^{th}|n^{th}$ object, and $i = 1..M$ (M is the number of conditional attributes). The reduct is "minimal" if $\neg\exists P \subset R$: (26.4) is satisfied for P.

Computing equivalence classes is straightforward. Finding a global minimal reduct, on the other hand (*i.e.*, reduct with a minimal cardinality among all reducts), is NP-hard. There are many heuristics designed to deal with this problem. For more details and examples of reduct-finding algorithms see *e.g.*, [27, 59].

26.2.1.5 Extensions of the Theory of Rough Sets

Since its proposal in the early 1980s, the theory of rough sets has been continuously adapted and expanded. For example, in order to better represent imprecise information, nondeterministic information systems have been proposed [30]. In other works, the original indiscernibility relation has been relaxed to a tolerance relation, where the transitivity property is not required [41]. However, although all those extensions are unquestionably important and can be potentially used in future applications of rough sets in the field of neuroscience, they are beyond the scope of this chapter.

26.2.2 Neuroscience

Neuroscience (or neural science) is the study of the nervous system, which is a complex organ system comprising a network of nerve cells called neurons [14].

Neurons can be studied at various levels: molecular, cellular, structural, functional, developmental, *etc*. While all those aspects are very important in themselves, as knowing the specific characteristics of nerve cells is critical for diagnosis and treatment of many neurological disorders, the ultimate goal of neuroscience is to understand the biological basis of consciousness and the mental processes through which we perceive, reason, act, learn, and remember. Modern neuroscience represents an amalgam of cell and molecular biology, neurophysiology, anatomy, embryology, and psychology. Furthermore, areas that have not been traditionally equated with ground-breaking discoveries in the biological sciences, such as mathematics or computer science, play an increasing role in the field. Due to the recent technological advances in both experimental instrumentation and data collection methods in various branches of neural science, there is an ever-growing need for automated analysis tools and database systems capable of handling such vast amounts of data. Since many problems in neuroscience are very complex in nature, data mining and knowledge discovery methods, including rough sets, lend themselves well to solving those problems. Below, we present a few selected sub-areas of neuroscience, where such techniques may be, or already have been, utilized.

26.2.2.1 Neurophysiology

Neurophysiology is a part of physiology that deals specifically with the study of the nervous system. The fundamental neurophysiological techniques include intra- and extra-cellular electrophysiological recordings, patch clamp recordings, calcium imaging, as well as messenger-RNA (mRNA) expression analysis and other tools used in molecular biology [14]. Another very important technique in neurophysiology is the dynamic clamp, which uses a real-time interface between one or several living cells and a computer or analog device to simulate dynamic processes such as membrane or synaptic currents in the living cells [35]. Regardless of the specific technique, as recording quality improves, neurophysiological experiments result in ever growing amounts of data, making it harder for conventional analysis methods to keep pace. Therefore, the need for automated analysis tools, with emphasis on computational intelligence techniques, and database systems has become widespread in neurophysiology [9].

Spike sorting is one of the most important problems in neurophysiology. It is the process of identifying and classifying spikes recorded by one or more electrodes as having been produced by particular neurons, and thus distinguishing their activity from the background noise, by only using the recorded data [7]. Another closely related problem is that of detection and characterization of bursts of neuronal electrical activity. Figure 26.2 presents an example of bursting neural activity, along with an illustration of the attributes that could be used for its characterization.

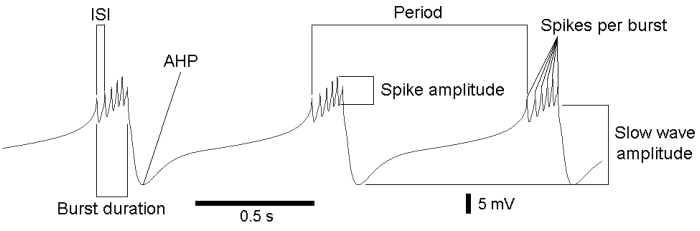


Fig. 26.2 An example of bursting electrical activity of a single neuron (generated by a model). ISI stands for Inter-Spiking Interval (*i.e.*, time between spikes in a burst) and AHP stands for After-Hyperpolarization Potential (*i.e.*, trough voltage between bursts) (source: [49]).

In both these problems, as well as many other tasks in neurophysiology, rough sets can be applied after some preprocessing and/or transformation of the recorded signals has been performed.

26.2.2.2 Behavioral and Cognitive Neuroscience

Behavioral neuroscience, also referred to as biological psychology, biopsychology, or psychobiology, is the study of physiological, genetic, and developmental mechanisms underlying the relationships between the brain and behavior [16]. Cognitive neuroscience is concerned with the scientific study of biological structures and mechanisms underlying cognition and mental processes in general [3]. Obviously, both those fields are very closely related and often studied in parallel. They are also highly inter-disciplinary drawing from such diverse fields as neurobiology and physiology, psychology and psychiatry, philosophy, linguistics, as well as physics, bioengineering, computer science, and mathematics.

Typically, experiments in behavioral neuroscience involve non-human animal models (*e.g.*, rats, mice, insects, and non-human primates). Therefore, most of the research in the field deals with behaviors that are shared by various animal models including sensation and perception, movement control, learning and memory, sleep and other biological rhythms, emotion, *etc*. The application of intricate research methods in behavioral neuroscience, such as disabling, decreasing, or enhancing neural function, via genetic manipulation or otherwise (*e.g.*, lesion induction, electrical stimulation, *etc*.), and the utilization of various neurophysiological methods for measuring neural activity, facilitates the collection of huge amounts of data describing the processes underlying the behaviors in question. Such data are usually in desperate need of efficient data mining techniques capable of explaining the phenomena responsible for the differences between normal and abnormal behaviors.

Researchers in cognitive science often employ imaging and visualization methods such as functional magnetic resonance imaging (fMRI) to study various topics including consciousness, attention, language, decision-making, learning, memory, *etc*. Yet again, such experiments yield vast amounts of multi-modal data that are well suited for rough sets-driven analysis.

26.2.2.3 Computational Neuroscience

Computational modeling of neurons is a very important aspect of today's neuroscience research. It allows for exploration of many parameter combinations and various types of neuronal activity, without requiring a prohibitively large number of "wet-lab" experiments. This is especially important in light of recent discoveries suggesting that functional neuronal electrical activity can be produced on the basis of widely varying cellular parameter combinations [22].

There exist various types of neuronal models, such as integrate-and-fire [1], Hodgkin-Huxley [11], Morris-Lecar [26], and others. In a multi-compartmental model, each part of the neuron (*e.g.*, soma–the neuron's cell body, neurites–branched projections of a neuron that conduct the electrical stimulation received from other cells, axon–the nerve fiber that conducts electrical impulses away from the cell body, *etc.*) is represented by a compartment, or a collection of compartments, each described by appropriate differential equations with a set of parameters [5].

As an example, let us consider one of the best-characterized neural networks in biology and a popular subject for studies of rhythmic activity in the central nervous system: the pyloric network in crustaceans (*e.g.*, lobster, crab) [25, 39]. Rhythmic activity is crucial for any living organism as it is responsible for such critical functions as breathing, chewing, running, *etc*. The pyloric network consists of up to 14 neurons of 6 distinct types. The AB (anterior burster) neuron is one of the three neurons forming the pacemaker kernel which drives the rhythmic activity of the pyloric neural network, which is responsible for filtering of food in the animal. The AB neuron produces rhythmic bursts of electrical activity of a specific profile, even when isolated from other cells in the network. The two-compartment model shown in Fig. 26.3 represents the AB neuron in the conductance parameter space [50], meaning that the model is described by a set of parameters that represent the maximum membrane conductances for different ions in the neuron. The first compartment in the model represents the soma and the neurites (S/N), and the second compartment corresponds to the axon (A). The figure also shows the ionic currents determined by the membrane conductances used in the model (arrows indicate the directionality of the currents–inward vs. outward).

The parameters for such a neuronal model can be varied independently yielding many possible combinations of the parameter values. The parameter space can be explored by simulating all of those parameter value combinations and checking which of the solutions match the behavior of their biological counterparts. Not only can a thorough analysis of the characteristics of physiologically realistic models help determine the ranges for the neuronal models' parameter values (and thus the cellular properties of the real cells), but it can also shed some light on the relationships between those parameters, which is a tremendously "hot topic" in today's field of neuroscience, as it relates to co-regulations of ionic currents in a cell and thus provides some insights into how biology responds to the changing environment and other perturbations.

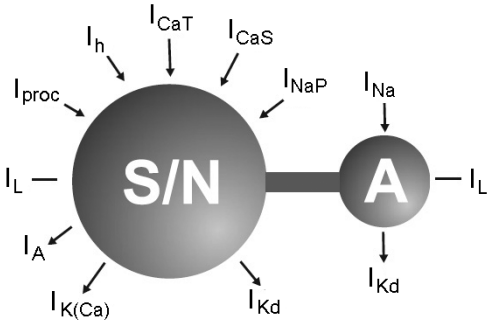


Fig. 26.3 A model of the AB (anterior burster) neuron of the pacemaker kernel in the crustacean pyloric neural network. S/N: Soma/Neurite compartment. A: Axon compartment. Arrows indicate inward and outward ionic currents as marked by labels (source: [50]).

26.2.2.4 Neurology

Neurology is a medical specialty concerned with the diagnosis and treatment of diseases involving the central, peripheral, and autonomic nervous systems, including blood vessels and all effector tissue, such as muscle [40]. It is arguably one of the most difficult and exciting medical specialties. Practitioners in that field must have solid foundations in neuroscience, including neuroanatomy, neurophysiology, neuropathology, and many other aspects spanning the complexity of the nervous system. Moreover, neurologists have to be familiar with numerous medical tests used in neurologic diagnosis, such as lumbar puncture, electroencephalography (EEG), computer tomography (CT), magnetic resonance imaging (MRI), and many more [37]. Neurologic examination includes numerous procedures that are designed to test the patients' level of consciousness, cognitive functioning, orientation, common knowledge, memory, insight and judgment, concentration, calculations, verbal fluency, *etc*. Furthermore, the data resulting from such examinations can be supplemented by a whole plethora of other types of neurophysiological data, as described in Sections 26.2.2.1-26.2.2.2. However, in addition to the inherent complexity of the neurophysiology in itself, analysis of neurological data becomes even more difficult when all the aspects of the urgency and criticality of medical diagnoses are considered. Thankfully, many medical databases, including neurological ones, can be relatively easily transformed into decision tables and thus lend themselves to rough sets-based data mining. This is especially true when the underlying task is a classification problem. For example, based on the neurological attributes collected from a sample of previously diagnosed and treated patients, a classifier capable of detecting a specific disease (or a stage of a disease) can be built. Obviously, not only can such a classifier be tremendously useful as a diagnosis support tool, but it may also help determine which tests are really needed for a given diagnosis (*e.g.*, by discovering reducts in the dataset), which in turn can save patients from having to go though potentially invasive, painful, and costly, yet unnecessary, procedures.

26.3 Rough Sets in Neuroscience: Selected Applications

There have been some successful applications of the theory of rough sets implemented in the "neuro world." Here, we present some of them. Since most of those applications concern neurology, we begin our overview with those pertaining to that field. We conclude the overview with the description of applications devoted to more basic neuroscience research, and describe two of those in more detail. Although the works described in this chapter are obviously not the only ones, we believe that they are quite representative of the current state of the area of applications of rough sets in neuroscience.

26.3.1 Clinical Neurology

Mitochondrial encephalomyopathies (MEM) are a group of neurological disorders caused by dysfunctional mitochondria, the organelles that convert the energy of food molecules into the ATP that powers most cell functions [54, 55]. The diseases affect many children around the world and are tainted by bleak prognosis. Wakulicz-Deja and Paszek [57] proposed an application of rough sets to diagnosis of MEM in children using data obtained from the Department of Pediatrics of the Silesian Academy of Medicine, Bytom, Poland. The decision support system significantly limits the need for invasive diagnostic procedures such as lumbar puncture or muscle/nerve biopsies. The authors achieved such a high performance by applying rough sets-driven attribute reduction and classification independently at three stages of the overall process of diagnosis, with increasing levels of invasiveness: 1) classification based on the clinical symptoms; 2) classification on the basis of biochemical data (*i.e.*, levels of lactic and formic acids in the blood serum and cerebrospinal fluid); and 3) classification on the basis of enzyme levels (*i.e.*, enzymatic activity in biopsied segments of muscles or nerves).

Epilepsy is a chronic neurological disorder characterized by seizures [2]. It affects about 50 million people worldwide. EEG is one of the most important tools used in diagnosis and treatment of epilepsy. Szczuka and Wojdyłło proposed a rough sets-based neuro-wavelet system, called WaRS, for noise-resistant classification of EEG signals [52]. The WaRS method provides a significant reduction in the dimensionality of the original problem, while keeping the essential information needed for accurate classification. The goal is achieved by first applying wavelet transformation to the original signals to extract descriptive features, which are then supplied to the rough sets-based algorithm for best discretization cuts selection and rough set rule generation. Experiments performed by the authors showed that the proposed method provides extended robustness and generalization, and allows a direct interpretation of the analysis results.

As mentioned in Section 26.2.2, magnetic resonance imaging is a very important tool in both basic neuroscience research, as well as neurology. Kobashi, *et al.*, [17] implemented a rough sets-based system to analysis of human brain MRI images.

The authors proposed a clustering method that extracts features of each pixel in an image using thresholding and labeling algorithms. Thus, the image features are transformed into nominal values, which are well-suited for rough set analysis. Widz, *et al*., [58] introduced an automated MRI segmentation technique based on approximate reducts derived from the theory of rough sets. They utilized multi-spectral MRI images from a simulated brain database as a gold standard to train and test the segmentation algorithm. The results suggested that approximate reducts, used alone or in combination with other classification methods, may provide a novel and efficient approach to the segmentation of volumetric MRI data sets.

26.3.2 Cognitive Computation

Humans can effortlessly recognize complex objects, such as faces, even if they have never seen them in a given context before. Przybyszewski [36] proposed to look into anatomical and neurophysiological basis of how object shapes are classified by the brain and to describe its computational properties by the means of the theory of rough sets. Many psychophysical experiments suggest that not only do we perform object classifications based on partial information about that object, but that the information about variations in the object's context, such as its rotation, is insignificant (illustrated, among others, by the so-called Thatcher effect, in which it is difficult to detect local feature changes in an upside-down face, while identical changes can be immediately spotted in an upright face). This suggests that the brain may be utilizing vague concepts to process approximate information about perceived objects to classify them. This assumption allows for an application of the rough sets theory in which, as described in Section 26.2.1, concepts are approximated via their lower and upper approximations. Here, we describe some aspects of the experiments and analyses performed by Przybyszewski.

In the experiments done on macaques, the brain's "expertise" in classifying objects' components was estimated by analyzing single cell responses to visual stimuli collected from multiple neurons in the area responsible for simple shape recognition–the visual area V4. The stimuli were described by a set of attributes such as spatial frequency, spatial frequency bandwidth, x and y axis position, stimulus shape, *etc*. The neuron responses were classified into three categories: 0 – activity below the threshold of 10–20 spikes/second; 1 – activity above the threshold labeled; 2 – activity above 30–40 spikes/second. Given this representation of the data, the results of the experiment were represented by a decision table, where each analyzed neuron was characterized by the stimuli properties (*i.e.*, conditional attributes) and the characteristics of its response to the stimuli (*i.e.*, decision attribute), for each of the applied stimuli.

Such representation of the data allowed for a direct application of rough set-based analysis through calculation of equivalence classes, concept approximations, and classification rules. This analysis facilitated discovery of interesting rules that seem to corroborate previous studies which associated area V4 with shape processing.

This, in turn, may indicate that the brain indeed uses rough sets-like reasoning for object classification.

26.3.3 Classificatory Decomposition of Cortical Evoked Potentials

As described in Section 26.2.2, many experiments in neuroscience produce traces of neuronal electrical activity. Often, such signals need to be classified into one or more predefined classes (*e.g.*, specific neuron type, presence vs. absence of a particular disease, *etc.*). Classification of such signals is possible when some measure D, which describes distances (*e.g.*, Euclidean, L2, *etc.*) between particular signals in the dataset, is introduced. However, it is often very difficult to come up with a distance measure that is not only accurate and insensitive to measurement errors, but also efficient computationally. Thus, in the problem of signal classification, extracting particular features that distinguish one process from another (*i.e.*, preserve the original distances) is crucial. Extraction of such feature vectors can be based upon a determination of a set of simple characteristics of a given signal (*e.g.*, minimum, maximum, average, *etc.*) or can be in the form of a more complex transformation or signal decomposition technique. The general idea behind these more complex transformations is to represent the original time series $\mathbf{x}$ in terms of some fixed (for all series in the database) basis functions $\mathbf{M}$ and a set of coefficients (possibly different for each series) $\mathbf{a}$, with an addition of some error $\mathbf{e}$:

$$\mathbf{x} = \mathbf{M}\mathbf{a} + \mathbf{e}. \tag{26.5}$$

Various transformation/decomposition techniques, such as principal component analysis (PCA) [8] or independent component analysis (ICA) [12], have been applied to signal classification with relative success. However, statistical criteria utilized in those methodologies are often insufficient to build a reliable classifier.

The main concept of classificatory decomposition was motivated by the hybridization of evolutionary algorithms (EA) with sparse coding with overcomplete bases (SCOB) introduced in [24]. Using this approach, the basis functions as well as the coefficients are being evolved by a genetic algorithm optimizing a fitness function that minimizes the reconstruction error and at the same time maximizes the sparseness of the basis function coding. This methodology produces a set of basis functions and a set of sparse (*i.e.*, "as few as possible") coefficients. This may significantly reduce dimensionality of a given problem but, as any other traditional decomposition technique, does not assure the classificatory usefulness of the generated model.

Classificatory decomposition (CD) is a general term that describes attempts to improve the effectiveness of signal decomposition techniques by providing them with "classification-awareness" [44]. In CD, the sparseness term is replaced by a rough sets-derived data reduction-driven classification accuracy measure. This should assure that the result will be both "valid" (*i.e.*, via the reconstruction

constraint) and useful for the classification task. Furthermore, since the classification-related constituent also searches for a reduct, the classification is done with as few as possible basis functions. Finally, the single-objective EA utilized in the aforementioned technique is replaced by a multi-objective optimization approach, in which the EA deals with the reconstruction error and classification accuracy, both at the same time. Since the approach utilized in CD is based upon finding a solution satisfying two potentially conflicting goals (*i.e.*, component-based reconstruction accuracy vs. classification accuracy), an application of multi-objective evolutionary algorithms (MOEA) seems natural.

The implementation of CD described in this section utilizes an extension of the Vector Evaluated Genetic Algorithm (VEGA), called end-VEGA (elitist-non-dominated-VEGA), which improves and enhances the original algorithm by supplying it with considerations of elitism and non-dominance [46]. To address non-dominance, the algorithm employs a simple approach based on multiplying the fitness of a given individual by the number of solutions that this individual is dominated by (+ 1 to ensure that the fitness function of a non-dominated solution is not multiplied by 0). Since all fitness functions in CD are set to be minimized (maximization of the classification accuracy can be easily translated into minimization of the classification error), the dominated solutions will be adequately penalized. To include elitism, end-VEGA utilizes the idea of an external sequential archive [20] to keep track of the best-so-far (*i.e.*, non-dominated) solutions, and to make sure that their genetic material is in the active gene pool.

The problem of minimization of the reconstruction error is intuitively simple. Once a particular distance measure has been decided upon, virtually any optimization algorithm can be used to minimize the distance between the original signal and the reconstructed one. The measure employed in CD is the well known 2-norm, referred to in signal processing as the signal energy-based measure [19]. In order to deal with raw signals which can be large (thus causing the energy-based distance measure to be large as well), a simple normalization of the energy-based measure by the energy of the original signal is proposed [44]:

$$D_{NORM} = \frac{\sum_{t=1}^{n} (\mathbf{x}_t - (\mathbf{Ma})_t)^2}{\sum_{t=1}^{n} (\mathbf{x}_t)^2}, \tag{26.6}$$

where $\mathbf{x}$ represents the original signal, $\mathbf{M}$ is the matrix of basis functions, $\mathbf{a}$ is a set of coefficients, and $t = 1..n$ where n is the number of samples in the signal.

Subsequently, the reconstruction error fitness function f_{REC} for a representative p takes the following form:

$$f_{REC}(p) = \frac{\sum_{i=1}^{N} D^i_{NORM}}{N}, \tag{26.7}$$

where D^i_{NORM} is the normalized reconstruction error for the i^{th} signal and N is the total number of the input signals.

The problem of maximizing the classificatory competence of the decomposition process, and at the same time reducing the number of computed basis functions, can be dealt with by the application of rough sets. In CD, the rough sets-based *quality of classification*, as introduced in Eq. (26.3), is used for the purpose of estimating the classificatory aptitude.

The quality of classification is estimated directly on the candidate reduct, which can be computed by any of the existing algorithms/heuristics. Note that the main objective that deals with the classificatory capability of decomposition can actually be considered a bi-objective optimization problem itself. On the one hand, we are looking for the best possible classification accuracy, but on the other, we want to use as few basis functions as possible. However, based on previous applications of EAs in the search for reducts, as described in [59], minimization of a single-objective fitness function that is simply a summation of the classification error and the relative length of the reduct was utilized, as shown in Eq. (26.8).

$$f_{CLASS}(p) = \left(1 - \gamma_R\right) + \frac{L(R)}{M}, \tag{26.8}$$

where p is a given representative (*i.e.*, chromosome), $L(R)$ is the length of the potential reduct R (*i.e.*, the number of attributes used in the representative), normalized by the total number of conditional attributes M, and γ_R is the *quality of classification* coefficient for the reduct R.

Classificatory decomposition has been successfully applied, at various stages of its development, in two projects involving cortical evoked potentials. First, the methodology was employed to investigate the influence of short cooling events applied to the surface of the cortex of the rat's brain, which temporarily disabled specific parts of the animal's brain [42]. Secondly, the approach was used to analyze the differences between rats that had been exposed to cigarette smoke *in utero* (*i.e.*, their mothers were exposed to cigarette smoke during pregnancy), while the others had not [45, 46]. The research problem was to investigate how treatments (like nicotine) could alter the brain's responses to discrete stimuli. In both cases, classificatory decomposition aided neuroscientists in their research pursuits and produced interesting insights into the science of the brain.

26.3.4 Classification of Functional and Non-functional Neuronal Models

As described earlier, computational modeling of neurons allows for exploration of many parameter combinations and various types of neuronal activity *in silico*. In this section, we describe an exploration and analysis of a large parameter space of the AB neuron introduced in section 26.2.2.3 [49].

To investigate the differences between functional and non-functional models of the AB cell, an extensive database of 21,600,000 of models was created by

systematically varying 12 parameters describing the model neurons (*i.e.*, maximal conductances of membrane currents, as shown in Fig. 26.3) from a "hand-tuned" AB model [50]. First, to determine the extent of the variations in the parameter values that the model can withstand and still produce functional activity, the maximal conductances of membrane currents were independently varied around their canonical values. To reduce the computational time and the size of the output database, parameters were first varied one at a time to determine physiologically reasonable value ranges and step sizes for each conductance separately. The "variation matrix" of all the explored values for the parameters in the AB model is shown in Fig. 26.4.

		variation in %					
soma							
g_{CaT}	4	-20	-10	0	+10		
g_{CaS}	5	-50	-25	0	+25	+50	
g_{NaP}	5	-100	-50	0	+50	+100	
g_h	3		-100	0	+150		
g_{Kd}	4		-10	0	+10	+20	
g_{KCa}	5		-50	0	+50	+100	+150
g_A	4		-100	0	+100	+200	
g_{Proc}	4	-20	-10	0	+10		
g_{leak}	3		-80	0	+80		
axon							
g_{Na}	5	-50	-25	0	+25	+50	
g_{Kd}	5		-50	0	+50	+100	+150
g_{leak}	3		-100	0	+200		
index		*1*	*2*	*3*	*4*	*5*	*6*

$3^3 \times 4^4 \times 5^5 = 21.6$ million combinations of AB models

Fig. 26.4 Explored parameter values for the AB neuron models, expressed as % deviation from the hand-tuned values (the quantity on the gray background shows the number of possible values for a given parameter) (source: [49]).

Each of the 21.6 million "candidate" model neurons was simulated and classified as functional if it produced biologically realistic activity under four scenarios: spontaneous activity, spontaneous activity with neuromodulator deprivation (*i.e.*, removal of the influence of neurotransmitters descending from other parts of the nervous system), activity with external current injections, and activity with neuromodulator deprivation and current injections. Whether the activity generated by the AB models was biologically realistic was judged based on experiments performed on their biological counterparts in isolation from the rest of the pyloric network and under each of the four conditions [48]. There were 353,208 (1.6352% of the the entire database) models meeting all the above criteria. The identified "good" models, as well as those which failed the test of functionality, were then subject to a rough sets-based rule-mining analysis in an attempt to explain the differences between the two groups via a set of concise and understandable IF/THEN classification rules.

Each model (*i.e.*, a particular combination of the parameter values) was coded in the database by integer numbers corresponding to the indices in the variation matrix (with 1 being the smallest possible index, and 3 always indicating the canonical value, as shown in Fig. 26.4). A binary classification attribute was also added to differentiate between functional and non-functional entries, thus transforming our model database into a full-fledged decision table. An example of a portion of the decision table is shown in Fig. 26.5.

g_{CaT} soma	g_{CaS} soma	g_{NaP} soma	g_{h} soma	g_{Kd} soma	g_{KCa} soma	g_{A} soma	g_{Proc} soma	g_{leak} soma	g_{Na} axon	g_{Kd} axon	g_{leak} axon	FUNCTIONAL?
2	1	1	1	2	3	2	4	4	2	3	3	NO
2	1	1	1	2	4	2	3	2	4	4	6	NO
2	1	1	1	3	2	3	2	2	2	2	2	YES
2	1	1	1	3	2	4	5	4	2	2	5	NO
2	1	1	1	3	4	2	3	1	3	3	2	NO
2	1	1	1	3	5	3	2	1	4	4	3	NO
2	1	1	1	3	5	4	2	3	3	5	3	NO
2	1	1	1	3	5	5	2	2	3	3	6	NO
2	1	1	1	3	5	5	4	3	2	3	3	NO
2	1	1	1	4	2	2	3	2	4	2	2	YES
2	1	1	1	4	5	2	4	3	3	1	2	NO
2	1	1	1	4	5	4	2	3	4	5	2	NO
2	1	1	1	4	5	5	2	4	3	4	2	NO
2	1	1	2	2	2	3	3	4	2	5	2	YES

Fig. 26.5 An example of a portion of the decision table constructed for the AB model neurons.

To reduce the computational complexity of the analysis, the approach was first tested on a sampled subset of the models. Based on previous experiments, it was determined that a 1% random sample adequately preserves the characteristics of the original dataset [49]. In addition, to deal with the problem of huge disproportion between the numbers of functional and non-functional models, the following sampling protocol was utilized: first, a random 1% sample of the "good" models was selected, and then 10 random samples of the same size of the "bad" models were drawn, thus creating 10 datasets with equal distributions of the two classes, which would be subject to further analyses in parallel. This was based on a quite well-known approach to balancing class distributions, especially useful in artificial neural network training [21], with existing applications in neuroscience [10].

One of the most natural ways to explain the differences between "good" and "bad" models could be via classification rules of the form "IF *some pattern within the parameter space*, THEN *functional model*" and "IF *some other pattern within the parameter space*, THEN *non-functional model*." The theory of rough sets lends itself naturally to this kind of analysis, especially since it is very well equipped to deal with imprecise and somewhat ambiguous data, which is a "part of life" in neuroscience. Not only can similar functional activity be produced by

neurons with disparate cellular characteristics, but also quite intricate interactions and relationships between the neurons' (and therefore models') parameters have been discovered [15, 38]. What this means is that it not only may be difficult to identify interesting and trustworthy IF/THEN rules, but that they also will most likely not be 100% accurate. In other words, even if a particular rule adequately explains the functional behavior of a subset of models, it may fail to elucidate the mechanisms governing a different subset, due to some hidden interactions characterizing that subset. The theory of rough sets by definition allows this kind of uncertainty in data, by the means of approximation of concepts via the indiscernibility relation and the equivalence classes determined upon it, as described in Section 26.2.1.

Since the AB models in the database are represented by sequences of integers (*i.e.*, indices in the variation matrix), which correspond to percentages of the hand-tuned values of the maximal membrane conductances, a direct application of rough sets-based analysis is possible. However, generating classification rules based on precise values of the specific membrane conductances makes for a difficult biological interpretation. Therefore, a discretization algorithm, the Equal Frequency Binning, which divides a sorted variable into k bins, where, given n instances, each bin contains m/k adjacent values [6] was applied. The k was purposely set to 3 to generate bins, which could be, without a loss of too much fidelity, referred to as "low," "intermediate" (always close to the hand-tuned value), and "high" conductance, independent of the actual value in μS.

A "by-product" of classification rules-based analysis is the ability to identify "important" attributes in a decision table. Obviously, if a given attribute is utilized in a trustworthy rule, it must be important from the standpoint of the underlying classification problem. The theory of rough sets provides a straightforward approach to the problem of selection of important features via the concept of reducts. In this particular project, the following two algorithms were utilized: 1) the well-known simple greedy Johnson's algorithm [13], which computes a single reduct only, and 2) a genetic algorithms-based implementation, which is capable of computing multiple reducts from a single dataset [56].

The methodology of genetic algorithms-driven pseudo-association rule mining [47, 49] was applied to the reduced and discretized data, and yielded 9 concise rules with support of between 1% and 20%, and confidence of at least 75% in the data. The rules provided very useful insights into the problem of analysis of how the activity of neurons depends on their cellular parameters. For instance, some of the rules described an intuitive dependence of the neural activity of the AB neuron on its axon's sodium (Na) current. The current is known to play a critical role in the process of spike generation, thus it makes sense that its corresponding conductance must be at least intermediate (*i.e.*, close to the hand-tuned value) to produce proper bursting. The understanding of these kinds of phenomena is extremely important to neuroscientists.

26.4 Rough Sets in Neuroscience: Open Problems

Although, as presented above, the theory of rough sets has found some successful applications in neuroscience and related areas, there is still plenty of untapped potential left. Since many problems in neuroscience are inherently complex, imprecise and/or ill-defined, research in the field could significantly benefit from the rough sets' ability to deal with vague or incomplete data. Potential applications in neurology are obvious–there is always going to be a need for efficient and accurate data mining and knowledge discovery methods in tackling such critical diseases as Parkinson's, Alzheimer's, Huntington's, Amyotrophic Lateral Sclerosis, to only name those few, for which there are clinical datasets available, but there is no cure.

There is also a tremendous potential for applications of rough sets in behavioral studies, where many experiments can be described in the form of decision tables and the task is to investigate the phenomena underlying the differences between normal and abnormal behaviors.

While rough sets have been successfully applied to analysis of computational neuronal models, as described in Section 26.3.4, the methodology could also be potentially utilized in the very process of the generation of physiologically realistic models. For example, combined with evolutionary algorithms that explore the parameter space in search for functional models, rough sets-based feature selection could be used to dynamically determine which of the model's parameters are critical at a given point for differentiation between functional and non-functional models, and therefore need additional tuning. The other parameters, deemed unimportant in the given segment of the parameter search space, may be temporarily ignored, thus simplifying the problem and speeding up the computations.

26.5 Conclusions

In this chapter, we reviewed selected applications of the theory of rough sets in the broadly defined field of neuroscience. We described problems ranging from clinical neurology to cognitive and computational neuroscience. As we hope to have shown, the existing applications in the neuro domain have been very successful, therefore, it is even more surprising that rough sets are not more commonly used in that field. With this chapter, we hope to promote the theory of rough sets amongst the researchers and practitioners in the "neuro world," but at the same time, we hope to foster a greater interest of the members of the rough sets community, and data mining and knowledge discovery as a whole, in the fascinating and rich field of neuroscience.

Importantly, we believe that by participating in the research efforts in the field of neuroscience, the rough sets community will benefit tremendously by finding inspiration and ideas for extensions of the now well-established field of rough-neural computing. After all, many RNC paradigms are directly or indirectly based on neuroscientific principles. By helping enhance the understanding of those

principles, we can improve the efficiency of RNC methods by incorporating new ideas for how humans perceive and process granules of knowledge; by aiding in the creation and analysis of realistic neuronal models and their networks, we can propose new architectures and training algorithms for artificial neural networks.

Acknowledgments. Support contributed to TGS through NSF grant EPS 0814251 to Delaware EPSCoR, the State of Delaware, and the Center for Integrated Biological and Environmental Research (CIBER) at Delaware State University, and to AAP through Burroughs-Wellcome Fund CASI Award.

The authors would like to thank Ms. Samantha McDaniel for her help with the preparation of some of the figures.

References

1. Abbott, L.F.: Lapique's introduction of the integrate-and-fire model neuron (1907). Brain Research Bulletin 50(5/6), 303–304 (1999)
2. Blume, W., Lüders, H., Mizrahi, E., Tassinari, C., van Emde Boas, W., Engel, J.: Glossary of descriptive terminology for ictal semiology: report of the ILAE task force on classification and terminology. Epilepsia 42(9), 1212–1218 (2001)
3. Breedlove, M., Watson, N.V., Rosenzweig, M.R.: Biological Psychology: An Introduction to Behavioral and Cognitive Neuroscience, 5th edn. Sinauer Associates, Inc. (2007)
4. Czyzewski, A.: Intelligent acquisition of audio signals employing neural networks and rough sets algotithms. In: Pal, S.K., Polkowski, L., Skowron, A. (eds.) Rough-Neural Computing, pp. 521–542. Springer, Heidelberg (2003)
5. Dayan, P., Abbott, L.F.: Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems. MIT Press (2001)
6. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and unsupervised discretization of continuous features. In: Proc. 12th International Conference on Machine Learning, Tahoe City, CA, pp. 194–202 (1995)
7. Fee, M.S., Mitra, P.P., Kleinfeld, D.: Automatic sorting of multiple unit neuronal signals in the presence of anisotropic and non-Gaussian variability. J. Neurosci. Methods 69, 175–188 (1996)
8. Flury, B.: Common Principal Components and Related Multivariate Models. John Wiley & Sons (1988)
9. Günay, C., et al.: Computational Intelligence in Electrophysiology: Trends and Open Problems. SCI, vol. 122, pp. 325–359 (2008)
10. Günay, C., Prinz, A.A.: Model calcium sensors for network homeostasis: Sensor and readout parameter analysis from a database of model neuronal networks. J. Neuroscience 30(5), 1686–1698 (2010)
11. Hodgkin, A., Huxley, A.: A quantitative description of membrane current and its application to conduction and excitation in nerve. J. Physiol. 117, 500–544 (1952)
12. Hyvarinen, A., Oja, E.: Independent component analysis: Algorithms and applications. Neural Networks 13, 411–430 (2000)
13. Johnson, D.S.: Approximation algorithms for combinatorial problems. J. of Computer and System Sciences 9, 256–278 (1974)
14. Kandel, E.R., Schwartz, J.H., Jessell, T.M.: Principles of Neural Science, 4th edn. McGraw-Hill (2000)
15. Khorkova, O., Golowasch, J.: Neuromodulators, not activity, control coordinated expression of ionic currents. J. Neuroscience 27(32), 8709–8718 (2007)
16. Kimble, D.P.: Biological Psychology. Holt, Rinehart, and Winston, Inc. (1988)

17. Kobashi, S., Kondo, K., Hata, Y.: Rough sets based medical image segmentation with connectedness. In: Proc. 5th Int. Forum on Multimedia and Image Processing, pp. 197–202 (2004)
18. Komorowski, J., Pawlak, Z., Polkowski, L., Skowron, A.: Rough sets: A tutorial. In: Pal, S.K., Skowron, A. (eds.) Rough Fuzzy Hybridization: A New Trend in Decision-Making, pp. 3–98 (1999)
19. Kreyszig, E.: Introductory Functional Analysis with Applications. Wiley, New York (1978)
20. Laumanns, M., Zitzler, E., Thiele, L.: A unified model for multi-objective evolutionary algorithms with elitism. In: Proc. Congress on Evolutionary Computation, pp. 46–53 (2000)
21. Lawrence, S., Burns, I., Back, A., Tsoi, A.C., Giles, C.L.: Neural Network Classification and Prior Class Probabilities. In: Orr, G.B., Müller, K.-R. (eds.) NIPS-WS 1996. LNCS, vol. 1524, pp. 299–314. Springer, Heidelberg (1998)
22. Marder, E., Goaillard, J.M.: Variability, compensation and homeostasis in neuron and network function. Nature Reviews Neuroscience 7(7), 563–574 (2006)
23. Marek, W., Pawlak, Z.: Rough Sets and Information Systems. Fundamenta Informaticae 17, 105–115 (1984)
24. Milanova, M.G., Smolinski, T.G., Boratyn, G.M., Żurada, J.M., Wrobel, A.: Sparse Correlation Kernel Analysis and Evolutionary Algorithm-Based Modeling of the Sensory Activity within the Rat's Barrel Cortex. In: Lee, S.-W., Verri, A. (eds.) SVM 2002. LNCS, vol. 2388, pp. 198–212. Springer, Heidelberg (2002)
25. Miller, J.P., Selverston, A.I.: Mechanisms underlying pattern generation in lobster stomatogastric ganglion as determined by selective inactivation of identified neurons. II. Oscillatory properties of pyloric neurons. J. Neurophysiology 48(6), 1378–1391 (1982)
26. Morris, C., Lecar, H.: Voltage oscillations in the barnacle giant muscle fiber. Biophys J. 35(1), 193–213 (1981)
27. Øhrn A.: ROSETTA Technical Reference Manual (2001) (retrieved May 6, 2011), `http://www.lcb.uu.se/tools/rosetta/materials/manual.pdf`
28. Pal, S.K, Pedrycz, W., Skowron, A., Swiniarski, R. (eds.): Special Volume: Rough-neuro Computing. Neurocomputing 36 (2001)
29. Pal, S.K., Polkowski, L., Skowron, A. (eds.): Rough-Neural Computing. Springer (2003)
30. Orłowska, E., Pawlak, Z.: Representation of nondeterministic information. Theoretical Computer Science 29, 27–39 (1984)
31. Pawlak, Z.: Rough Sets. International J. of Computer and Information Sciences 11, 341–356 (1982)
32. Pawlak, Z.: Rough sets - Theoretical aspects of reasoning about data. Kluwer (1991)
33. Peters, J.F., Skowron, A., Han, L., Ramanna, S.: Towards Rough Neural Computing Based on Rough Membership Functions: Theory and Application. In: Ziarko, W.P., Yao, Y. (eds.) RSCTC 2000. LNCS (LNAI), vol. 2005, pp. 611–618. Springer, Heidelberg (2001)
34. Polkowski, L., Skowron, A.: Rough-Neuro Computing. In: Ziarko, W.P., Yao, Y. (eds.) RSCTC 2000. LNCS (LNAI), vol. 2005, pp. 57–64. Springer, Heidelberg (2001)
35. Prinz, A.A., Abbott, L.F., Marder, E.: The Dynamic Clamp Comes of Age. Trends in Neuroscience 27, 218–224 (2004)
36. Przybyszewski, A.W.: The Neurophysiological Bases of Cognitive Computation Using Rough Set Theory. In: Peters, J.F., Skowron, A., Rybiński, H. (eds.) Transactions on Rough Sets IX. LNCS, vol. 5390, pp. 287–317. Springer, Heidelberg (2008)
37. Ropper, A., Samuels, M.: Adams and Victor's Principles of Neurology, 9th edn. McGraw-Hill Professional (2009)
38. Schulz, D.J., Goaillard, J.-M., Marder, E.: Quantitative expression profiling of identified neurons reveals cell-specific constraints on highly variable levels of gene expression. PNAS 104(32), 13187–13191 (2007)

39. Selverston, A.I., Miller, J.P.: Mechanisms underlying pattern generation in lobster stomatogastric ganglion as determined by selective inactivation of identified neurons. I. Pyloric system. J. Neurophysiology 44(6), 1102–1121 (1980)
40. Simon, R., Greenberg, D., Aminoff, M.: Clinical Neurology, 7th edn. McGraw-Hill Professional (2009)
41. Skowron, A., Stepaniuk, J.: Tolerance approximation spaces. Fundamenta Informaticae 27, 245–253 (1996)
42. Smolinski, T.G., Boratyn, G.M., Milanova, M.G., Żurada, J.M., Wrobel, A.: Evolutionary Algorithms and Rough Sets-Based Hybrid Approach to Classificatory Decomposition of Cortical Evoked Potentials. In: Alpigini, J.J., Peters, J.F., Skowron, A., Zhong, N. (eds.) RSCTC 2002. LNCS (LNAI), vol. 2475, pp. 621–628. Springer, Heidelberg (2002)
43. Smolinski, T.G., Chenoweth, D.L., Zurada, J.M.: Time Series Prediction Using Rough Sets and Neural Networks Hybrid Approach. In: Castillo, O. (ed.) Proc. IASTED International Conference on Neural Networks and Computational Intelligence (NCI 2003), pp. 108–111 (2003)
44. Smolinski, T.G.: Classificatory Decomposition for Time Series Classification and Clustering. PhD thesis, Univ. of Louisville, Louisville (2004)
45. Smolinski, T.G., Milanova, M.G., Boratyn, G.M., Buchanan, R., Prinz, A.A.: Multi-Objective Evolutionary Algorithms and Rough Sets for Decomposition and Analysis of Cortical Evoked Potentials. In: Proc. IEEE International Conference on Granular Computing (GrC 2006), pp. 635–638 (2006)
46. Smolinski, T.G., Boratyn, G.M., Milanova, M.G., Buchanan, R., Prinz, A.A.: Hybridization of Independent Component Analysis, Rough Sets, and Multi-Objective Evolutionary Algorithms for Classificatory Decomposition of Cortical Evoked Potentials. In: Rajapakse, J.C., Wong, L., Acharya, R. (eds.) PRIB 2006. LNCS (LNBI), vol. 4146, pp. 174–183. Springer, Heidelberg (2006)
47. Smolinski, T.G., Soto-Treviño, C., Rabbah, P., Nadim, F., Prinz, A.A.: Analysis of biological neurons via modeling and rule mining. International J. of Information Technology and Intelligent Computing 1(2), 293–302 (2006)
48. Smolinski, T.G., Soto-Treviño, C., Rabbah, P., Nadim, F., Prinz, A.A.: Computational exploration of a multi-compartment model of the AB neuron in the lobster pyloric pacemaker kernel. BMC Neuroscience 9(suppl. 1), P53 (2008)
49. Smolinski, T.G., Prinz, A.A.: Rough Sets for Solving Classification Problems in Computational Neuroscience. In: Szczuka, M., Kryszkiewicz, M., Ramanna, S., Jensen, R., Hu, Q. (eds.) RSCTC 2010. LNCS (LNAI), vol. 6086, pp. 620–629. Springer, Heidelberg (2010)
50. Soto-Treviño, C., Rabbah, P., Marder, E., Nadim, F.: Computational model of electrically coupled, intrinsically distinct pacemaker neurons. J. Neurophysiology 94(2), 590–604 (2005)
51. Sulaiman, S., Shamsuddin, S.M., Abraham, A.: Rough Neuro-PSO Web Caching and XML Prefetching for Accessing Facebook from Mobile Environment. In: Proc. 8th International Conference on Computer Information Systems and Industrial Management (CISIM 2009), pp. 884–889. IEEE Computer Society Press (2009)
52. Szczuka, M., Wojdyłło, P.: Neuro-Wavelet Classifiers for EEG Signals Based on Rough Set Methods. In: Pal, S.K., Pedrycz, W., Skowron, A., Swiniarski, R. (eds.) Special Volume: Rough-neuro Computing. Neurocomputing, vol. 36, pp. 103–122 (2001)
53. Tsumoto, S.: Computational Analysis of Acquired Dyslexia of Kanji Characters Based on Conventional and Rough Neural Networks. In: Pal, S.K., Polkowski, L., Skowron, A. (eds.) Rough-Neural Computing, pp. 637–648. Springer, Heidelberg (2003)
54. Tulinius, M.H., Holme, E., Kristianson, B.: Mitochondrial encephalomyopathies in childhood: 1. Biochemical and morphologic investigations. J. Pediatrics 119, 242–250 (1991)
55. Tulinius, M.H., Holme, E., Kristianson, B.: Mitochondrial encephalomyopathies in childhood: 2. Clinical manifestation and syndromes. J. Pediatrics 119, 251–259 (1991)

56. Vinterbo, S., Øhrn, A.: Minimal approximate hitting sets and rule templates. International J. of Approximate Reasoning 25(2), 123–143 (2000)
57. Wakulicz-Deja, A., Paszek, P.: Applying rough set theory to multi stage medical diagnosing. Fundamenta Informaticae 54(4), 387–408 (2003)
58. Widz, S., Revett, K., Ślęzak, D.: Application of rough set based dynamic parameter optimization to MRI segmentation. In: Proc. 23rd Int. Conference of the North American Fuzzy Information Processing Society, pp. 440–445 (2004)
59. Wróblewski, J.: Finding minimal reducts using genetic algorithms. In: Proc. 2nd Annual Joint Conference on Information Sciences, pp. 186–189 (1995)

Chapter 27
Knowledge Representation and Automated Methods of Searching for Information in Bibliographical Data Bases: A Rough Set Approach

Zbigniew Suraj, Piotr Grochowalski, and Krzysztof Pancerz

Abstract. In this paper, we present an approach to searching for information in bibliographical data bases founded on rough set theory and the domain knowledge. The additional knowledge of the information searched by the user is represented in the form of two kinds of ontologies: a general ontology and a specific ontology. The general ontology is built by domain experts. In research carried out, this ontology covers information about fundamental notions in the area of rough set theory and its applications as well as about significant relationships between these notions. The specific ontology delivers us the additional knowledge of a bibliographical description of a paper searched in a data base. This knowledge is extracted automatically from data gathered in the Rough Set Database System (RSDS). This system, besides a typical utility function, constitutes an environment for conducting research with a view to verify the validity of the proposed methods and algorithms devoted to searching for information in its data base.

Keywords: ontology, ontological graphs, rough sets, database systems, RSDS system.

Zbigniew Suraj
Institute of Computer Science, University of Rzeszów, Dekerta Str. 2, 35-030 Rzeszów, Poland
e-mail: zsuraj@univ.rzeszow.pl

Piotr Grochowalski
Institute of Computer Science, University of Rzeszów, Dekerta Str. 2, 35-030 Rzeszów, Poland
e-mail: piotrg@univ.rzeszow.pl

Krzysztof Pancerz
Institute of Biomedical Informatics, University of Information Technology and Management, Sucharskiego Str. 2, 35-225 Rzeszów, Poland
e-mail: kpancerz@wsiz.rzeszow.pl

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 515–538.
springerlink.com

27.1 Introduction

These days each of us is exposed to a lot of information. The man, being a thinking creature, is able to select the obtained information himself. In everyday life, we meet computers almost everywhere. They expose us to an uncountable amount of information which must be processed to select it properly. The greatest, and the most popular source of information for the computer users today, is the Internet. The Internet contains countless information concerning different fields of knowledge. In such immensity of information, it is really hard to find something that interests us in particular. Although a lot of more or less intelligent browsers have been constructed, there is still vast research conducted to find new methods of browsing broad resources of information (data bases). This is a scientific challenge in the field of equipment, programming, as well as conception. It also concerns such fields as: artificial intelligence, reading comprehension, processing and understanding images, speech recognition, etc.

In recent years, we take more and more attempts related to the development of tools for semantic searching for information [4, 7, 12, 17, 27]. Support for semantic searching for information occurs at very different levels. Some systems expect, from the end-user, knowledge of the technology used in creating and describing ontologies representing data [4, 7], or specialized knowledge of languages based on ontologies [36, 37]. Other techniques use natural language processing (NLP) [17]. Methodology developed by us and the semantic search system that uses it in the RSDS system may be classified into a group of Keyword-Based Search Engines. An example of a semantic search system that belongs to this group is the Aragog system [18]. In comparison to Aragog, our system does not require knowledge of the construction used in the ontology and specialized query languages, the whole search process is "invisible" to the user. The main mechanism of semantic searching is not based on any specialized search language, but on the data stored in a relational database using an ontology and theory of rough sets. Additionally, our system is based only on one ontology describing the domain of exploration, which is enlarged on the basis of the ontology automatically constructed from data in the RSDS system.

In the research that moves to create the mechanisms, which would allow machines (computers) to search for information (mainly text resources) in the way that the man does, the key role is the ability to understand the obtained information automatically. Until now, machines are not able to understand the obtained information. In order to eliminate that barrier, the intensive research is conducted. This research mainly concerns the methods of analyzing great reserves of information, often with an undefined structure. We can classify these methods into one of two groups [8, 9]: knowledge-rich methods or knowledge-poor methods. The knowledge-rich methods require the "initial knowledge" — semantic information on the basis of which they perform further operations. Unfortunately, they are often limited to a part of reality, which is represented by a reconsidered resource of information (a set of documents). At present, a big role in this research is played by ontologies [11] which are treated as a source of semantic knowledge. Ontologies are examined in different aspects,

e.g. building (learning) ontologies on the basis of the text [3], defining new terms for ontologies, defining various relationships between particular fragments of the text [8, 26, 28], building intelligent browsers of information [1, 14], etc.

The knowledge-poor methods are characterized mainly by the abstraction from the specification of the processed documents, and they are often based on the statistical approach to the text analysis [2, 6, 35]. Moreover, the research concerning both groups is conducted to try to join them [5].

The realized research aims at equipping the hitherto existing "data carriers" with additional data, called metadata [19], which help to perform various operations connected with generally defined information processing, *e.g.* the concept of the Semantic Internet [13], including semantic browsers [1, 14], semantic digital libraries [16, 39–43].

The structure of the remaining part of this paper is as follows. Section 27.2 includes basic definitions of the concepts used in further parts of the paper. The formulated methods and algorithms related to semantic searching for information have been described in Section 27.5. That chapter also contains general characteristics of the defined field ontology, i.e., concerning rough set theory and its applications. The description of the RSDS system and its basic applications have been presented in Section 27.6. Section 27.7 shows briefly the conducted experiments and the conclusions. The final conclusions have been presented in Section 27.8.

27.2 Basic Concepts

In this section, the definitions of basic terms concerning rough sets [21–24, 30], ontologies [15, 20], metrics and the angle between two vectors have been gathered.

27.2.1 Rough Sets

Rough set theory is mainly based on binary relation between objects of some universe. Let U be a nonempty set of objects and $u \in U$. Any subset $R \subseteq U \times U$ is called a binary relation on U. $R(u)$ denotes a set of all objects $v \in U$ such that $(u,v) \in R$. The pair $(u,v) \in U \times U$ will be read as "u is in the relation R with v". We say that the relation R is:

- reflexive, if and only if, for any $u \in U$, it is true that $(u,u) \in R$,
- symmetric, if and only if, for any $u,v \in U$, it is true that if $(u,v) \in R$, then $(v,u) \in R$,
- transitive, if and only if, for any $u,v,w \in U$, it is true that if $(u,v) \in R$ and $(v,w) \in R$, then $(u,w) \in R$.

If R is reflexive, symmetric and transitive, then R is called the equivalence relation.

Let U be a nonempty, finite set called the universe, $R \subseteq U \times U$ be a binary relation defined on U, and $X \subseteq U$ be a given subset of U.

Definition 27.1. The lower approximation $\underline{R}(X)$ of the set X with respect to the relation R is the set $\underline{R}(X) = \{u \in U : R(u) \subseteq X\}$.

The lower approximation $\underline{R}(X)$ is such a set of objects in U, which are only in relation R with objects from the set X.

Definition 27.2. The upper approximation $\overline{R}(X)$ of the set X with respect to the relation R is the set $\overline{R}(X) = \{u \in U : R(u) \cap X \neq \emptyset\}$.

The upper approximation $\overline{R}(X)$ is such a set of objects in U, which are in the relation R with the objects from the set X and which can be in the relation R with the objects from the set $U - X$.

Definition 27.3. The boundary region $BN_R(X)$ of the set X with respect to the relation R is the set $BN_R(X) = \overline{R}(X) - \underline{R}(X)$.

The boundary region $BN_R(X)$ is a set of objects in U, which are in the relation R with the objects from the set X as well as with the objects from the set $U - X$.

Definition 27.4. Accuracy of approximation $\alpha_R(X)$ of the set X with respect to the relation R is a ratio $\alpha_R(X) = \frac{card(\underline{R}(X))}{card(\overline{R}(X))}$, where $card(A)$ denotes the cardinality of the set A.

The values of the coefficient $\alpha_R(X)$ fulfill the dependence: $0 \leq \alpha_R(X) \leq 1$. If $\alpha_R(X) = 1$, then the set X is uniquely defined by the relation R. This coefficient is a dispersal measure of the relation R on different objects.

27.2.2 Ontologies

Definition 27.5. Let U be a nonempty, finite set called the universe of concepts, $\mathcal{C} \subseteq U$, and $\mathcal{R}$ be a family of relations defined on the set $\mathcal{C}$. An ontology O is a pair $O = (\mathcal{C}, \mathcal{R})$. $\mathcal{C}$ is called the set of concepts in the ontology O.

Definition 27.6. Let $O = (\mathcal{C}, \mathcal{R})$ be an ontology. An ontology graph OG is a directed graph: $OG = (N, E, \gamma, \rho)$, where N is a finite set of nodes, $E \subseteq N \times N$ is a set of directed edges, $\gamma : N \to \mathcal{C}$ is a function which attributes the concept to the graph nodes, and $\rho : E \to \mathcal{R}$ is a function which attributes relations to the graph edges.

In the graph OG nodes correspond to concepts whereas edges correspond to relationships between concepts.

Let $O = (\mathcal{C}, \mathcal{R})$ be an ontology. In the research, we distinguish five binary relations between concepts from the set $\mathcal{C}$, i.e., $\mathcal{R} = \{R_{\bowtie}, R_{\subset}, R_{\triangleright}, R_{\triangleleft}, R_{\sim}\}$. We call these relations adequately:

- $R_{\bowtie}$: a connection relation. We assume, that this relation is reflexive and symmetric. If $(x, y) \in R_{\bowtie}$, then we say that the concept x is "semantically connected" with the concept y.

- $R_{\subset}$: an inclusion relation. We assume, that this relation is reflexive and transitive. If $(x,y) \in R_{\subset}$, then we say that the concept x is "part of" the concept y.
- $R_{\lhd}$: a generalization relation. We assume, that this relation is reflexive and transitive. If $(x,y) \in R_{\lhd}$, then we say that the concept y is a "generalization" of the concept x.
- $R_{\rhd}$: a specification relation. We assume, that this relation is reflexive and transitive. If $(x,y) \in R_{\rhd}$, then we say that the concept y is a "specification" of the concept x.
- $R_{\sim}$: a synonym relation. We assume, that this relation is an equivalence relation. If $(x,y) \in R_{\sim}$, then we say that the concept x is a "synonym" of the concept y.

Using the definitions from Subsection 27.2.1, we can modify the definitions of the lower and the upper approximation as follows. Let $\mathcal{C}$ be a set of concepts in the ontology O, $D \subseteq \mathcal{C}$, $\mathcal{R}$ be a set of binary relations on $\mathcal{C}$. For every relation $R \in \mathcal{R}$, we define the lower and the upper approximation of the set D as $\underline{R}(D) = \{c \in \mathcal{C} : R(c) \subseteq D\}$ and $\overline{R}(D) = \{c \in \mathcal{C} : R(c) \cap D \neq \emptyset\}$, respectively. The boundary region of the set D is denoted as $BN_R(D) = \overline{R}(D) - \underline{R}(D)$.

We will further consider only a particular subset C of concepts from the whole ontology. That is why we define the the lower and the upper approximation of a given set D restricted to the subset C. Restricted approximations have the forms as follows: $\underline{R}(D)|_C = \{c \in C : R(c) \subseteq D\}$ and $\overline{R}(D)|_C = \{c \in C : R(c) \cap D \neq \emptyset\}$, respectively. Analogically, the boundary region of the set D restricted to the set C, is defined as: $BN_R(D)|_C = \overline{R}(D)|_C - \underline{R}(D)|_C$.

For every relation R from $\mathcal{R}$, we define accuracy of approximation of the set D in such a way, that $\alpha_R(D)|_C = \frac{card(\underline{R}(D)|_C)}{card(\overline{R}(D)|_C)}$.

We slightly modify accuracy of approximation for the use it in ontological searching (see Subsection 27.5.2).

27.2.3 Generators of Weights

In this subsection, we recall the formulas for generators of weights used in experiments described in Section 9.5.

Definition 27.7. Arithmetic generator
Let n be a fixed natural number $(n > 0)$. An individual weight a_i of the sequence of weights is defined as $a_i = \frac{2i}{n(n+1)}$ for $i = 1,2,\ldots,n$.

Definition 27.8. Pseudo-geometric generator
Let n and p be fixed natural numbers $(n > 0,\ p > 1)$. An individual weight a_i of the sequence of weights is defined as $a_i = (p-1)\frac{1}{p^i}$ for $i = 1,2,\ldots,n$ and the rest $r_n = \frac{1}{p^n}$.

Definition 27.9. Ordinary generator
Let n be a fixed natural number $(n > 0)$. An individual weight a_i of the sequence of weights is defined as $a_i = \frac{1}{i(i+1)}$ for $i = 1,2,\ldots,n$ and the rest $r_n = \frac{1}{n+1}$.

Definition 27.10. Golden generator
Let n be a fixed natural number $(n > 0)$. An individual weight a_i of the sequence of weights is recursively defined in the following way:

$$a_i = \begin{cases} \frac{\sqrt{5}-1}{2} \text{ for } i = 0, \\ \frac{\sqrt{5}-1}{2}(1 - \sum_{j=0}^{i-1} a_j) \text{ for } i > 0 \end{cases}$$

and the rest $r_n = 1 - \sum_{i=0}^{n-1} a_i$.
Individual weights a_i of the sequence of weights are created as a result of a golden division of number 1 as well as the "remaining" parts of this number.

27.2.4 Metrics

In this subsection, we recall definitions of selected metrics used in experiments. Let $x = (x_1, x_2, \ldots, x_n)$ and $y = (y_1, y_2, \ldots, y_n)$ be vectors with n real coordinates.

Definition 27.11. Euclidean metric
The Euclidean distance d_E between vectors x and y is defined by the following formula:

$$d_E(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}.$$

Definition 27.12. Minkowski metric
The Minkowski distance d_M between vectors x and y is defined by the following formula:

$$d_M(x,y) = (\sum_{i=1}^{n} |x_i - y_i|^p)^{1/p}, \ p \in [1,\infty).$$

Definition 27.13. City metric
The city distance d_C between vectors x and y is defined by the following formula:

$$d_C(x,y) = \sum_{i=1}^{n} |x_i - y_i|.$$

Definition 27.14. Hamming metric
The Hamming distance d_H between vectors x and y is defined by the following formula:

$$d_H(x,y) = \sum_{i=1}^{n} f(x_i,y_i) \text{ where } f(x_i,y_i) = \begin{cases} 0 \text{ if } x_i = y_i \\ 1 \text{ if } x_i \neq y_i \end{cases}.$$

Definition 27.15. Chebyschev (maximum) metric
The Chebyschev distance d_T between vectors x and y is defined by the following formula:

$$d_T(x,y) = \max_{i=1,2,\ldots,n} |x_i - y_i|.$$

27.2.5 Angle between Vectors

In order to define the similarity of two vectors, we can use the angle between vectors defining whether two vectors are near each other.

Definition 27.16. Let $x = (x_1, x_2, \ldots, x_n)$ and $y = (y_1, y_2, \ldots, y_n)$ be vectors with n real coefficients. The formula for the cosine of the angle α between vectors x and y has the form:

$$\cos\alpha = \frac{d^2(x,0) + d^2(y,0) - d^2(x-y,0)}{2d(x,0) \cdot d(y,0)}$$

where $d(z,0)$ denotes the length of the vector z according to a given metric d defined in Subsection 27.2.4, $0 = (0,0,\ldots,0)$ is the zero vector with n coordinates, while vector $x - y = (x_1 - y_1, x_2 - y_2, \ldots, x_n - y_n)$.

27.3 Main Aims of the Paper

The aim of the author's research is an attempt to formulate effective methods and mechanisms which would allow automatized administration of information, searching for information as well as representation of possessed information in bibliographical databases. An intelligent activity of the proposed mechanisms will depend on the ability to understand (at least partly) the possessed information, i.e., obtaining the knowledge which so far could have been read only by human beings. The experimental environment for the conducted research is the formulated system of the RSDS data base (*Rough Set Database System*). This system includes bibliographical descriptions of publications related to the methodology of rough set theory and its applications. Despite the fact, that this system does not include various kinds of information, it is still a good tool allowing to test the correctness of the proposed solutions.

27.4 The Results Obtained So Far

Within the conducted research, we have formulated the field (general) ontology, constituting an attempt to systematize rough set theory. The creation of such a formal description will allow to conduct different research related to rough sets. We have

also proposed the methodology allowing creation of an ontology of a bibliographical description for every publication, for the next step in the process of intelligent administration of information in the system. This ontology will represent the meaning of the content of a given publication. On the basis of the formulated kinds of ontologies, we have proposed the mechanism which allows to realize the intelligent information searching in bibliographical systems on the example of the RSDS system. The intelligent activity of this system can be observed through the ability to understand the possessed information. This mechanism has been implemented in the RSDS system which helped to verify it in practice. The conducted experiments demonstrate the effectiveness of the mechanism.

27.5 Methods and Algorithms Related to "Intelligent" Searching for Information

The existing techniques of searching for information, i.e., database engines, the web search engines (Google, etc.) are very dependent on the man, as they are not directed to try to understand expectations of the system's user to enhance the effectiveness of searching. In the approaches proposed by the authors, such an aspect of searching for information is treated as a key aspect. As a consequence, we propose to broaden the present RSDS system and add further "intelligent" mechanisms of searching for information. This system is a bibliographical database related to rough set theory and its applications, but it is also the experimental environment for the conducted research. The theoretical foundation to build such extensions for this system are *e.g.* the rough set methods and those ontologies, which are treated as the reserves of additional information about publications collected in the RSDS system. The ontologies used in the research allow to describe a set of publications and the dependencies between the main elements (concepts) existing in those ontologies, in a more detailed way. In this research, the descriptions of publications are formulated in English.

27.5.1 Methodology of Creating a Detailed Ontology

In the approach that we have presented, we take into consideration two kinds of ontologies: a general ontology and a detailed one. The first of them, i.e., the *general (field) ontology*, describes the concepts related to a considered field of applications, and it also describes covert and overt dependencies between such concepts. For the RSDS system, the general ontology deals with rough set theory and its applications. This ontology has been formulated and presented in [10]. A part of this ontology is shown in Figure 27.1.

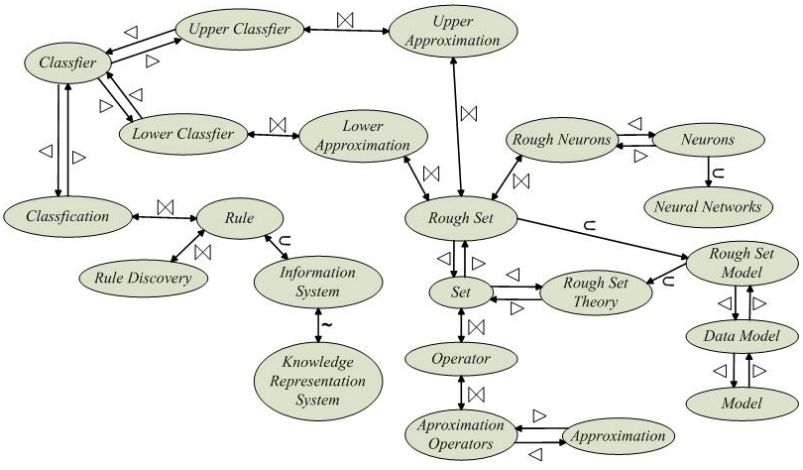


Fig. 27.1 Part of general (field) ontology.

Building the general ontology was the first step to construct an "intelligent" mechanism of searching for demanded information. In this case, it means more "apt" information finding in the RSDS system, i.e., finding one particular publication or if it is for some reason impossible, pointing out other publications covering similar topics. The second kind of ontology, called the *detailed (publication) ontology*, concerns the concepts and relations between the concepts in the descriptions of publications placed in the RSDS system. A description of a publication put into that system includes such elements as: a title of a publication, sometimes a subtitle, authors of a publication, summary, key words, etc. In those elements of a description of a publication, there is a vast reserve of overt semantic (meaningful) information, and also covert information which can be discovered using *e.g.* the rough set methods. The detailed ontology is built for every publication separately, by looking for the general ontology concepts in the publication. In case of finding such a concept, it is incorporated to the created detailed ontology for that particular publication. In order to make searching for the concepts in descriptions of publications more effective, first we submit the concepts of the general ontology and descriptions of publications to preprocessing. The general idea of preprocessing depends on:

1. Defining basic grammatical forms (roots) for particular words existing in descriptions and concepts of the main ontology. Thanks to that, we avoid the situations where every word, existing in different grammatical forms in a description of a publication, would define many new concepts. This process is realized with the use of the quite popular Porter stemming algorithm [25]. This algorithm contains, in its knowledge base, a defined list of well known, and the most often appearing, word endings. We try to obtain the root of a word by analyzing the endings of the processed word, and using a defined knowledge base. That process is repeated in several turns. In every further step it deals with a

different element of the word analysis, *e.g.* making the word singular, putting the word into the present tense, eliminating the ending, etc.
2. Eliminating words with little semantic value, *e.g.* a, about, etc. Such words bring very little essential information from the point of view of a considered problem, and they can introduce unnecessary interference to the process of searching for the demanded information.

After the preprocessing has been done, we start to identify the existence of the general ontology concepts in the elements of a description of publications available in the RSDS system. In case of finding a particular concept from the general ontology in a description of a given publication, it is added in the form of a node into a created graph, representing the detailed ontology for a given publication. However, if there is no concept from the main ontology identified, we start to search for "important" words in a description of a publication (in present research, we consider defining the measure of importance of particular words). In this case, the system awaits for an expert to help and to take a final decision for the importance of the concept on the basis of such information. The system tells the expert which words in a description of a publication appear most often (probably they have the greatest meaning in the considered text). Next, the expert decides whether the proposed concept is vital or not for the phase of building a particular ontology for a publication. During the research conducted at present, we consider only the nodes of the created graph in view of the further development of the created methodology, and we consider defining the relations between particular concepts. The exemplary nodes of the graph for a single detailed ontology are: *Knowledge Discovering, Data Reduction, Rough Set, Attribute, Characteristic Rule, Classification Rule, Decision Rule, Knowledge Rule*.

It is worth adding, that detailed ontologies, in the proposed approach, are automatically built for every publication, separately with the use of the general ontology and bibliographical descriptions made available through the database of the RSDS system. When the detailed ontologies are defined, they are recorded in the system to use them later in the created process.

27.5.2 The Mechanism of "Intelligent" Searching for Information

Another step to create a formulated mechanism was creating a searching mechanism on the basis of defined ontologies. For that purpose, we additionally used the rough sets methodology, in particular, the concepts of the lower and the upper approximation defined in it (see Section 27.2.1). The mechanism of that step looks as follows:

- At the beginning, one has to define a demanded concept c in relation to which information will be searched for, i.e., publications in the RSDS system. At present, the mechanism allows to define only individual concepts for searching, defined in the domain ontology. However, we are trying to work out how

to extend the abilities of the mechanism. The formulated extension will depend on a possibility of defining complex concepts, i.e., concepts consisting of some individual concepts connected with adequate operators.

- In the next step, a demanded concept c is searched for in the field ontology, particularly, in an ontology graph, defined on the basis of the field ontology. That graph is built according to Definition 27.6. The concept c has in such a graph its own nearest surrounding consisting of other concepts, i.e., it has its neighbors. Two concepts (the nodes of the ontology graph) are neighbors if there exists an edge between them. Taking into consideration the fact that in the considered graph, there are five different relations (connections) between the concepts, we can distinguish five different surroundings of the concept c according to the kind of the relation. For every surrounding of the concept c, we define its approximation on the basis of the definition of the lower and the upper approximation from rough set theory, which are defined in Subsection 27.2.2. In definitions of both approximations, we use some slightly modified relation R^*. If R is reflexive, then we exclude the trivial situation that $(u,u) \in R$. Therefore $R^* = R - \{(u,u)\}$. Hence $R^*(u) = R(u) - \{u\}$. Let C be a set of concepts included in a given publication and $D = \{c\}$, the lower and the upper approximation can be intuitively defined as follows:
  - concepts from C, which are connected by a given relation R^* only with a considered concept c, belong to the lower approximation $\underline{R^*}(D)|_C$ of D according to R^* restricted to C,
  - concepts from C, which are connected by a given relation R^* with a considered concept c, and through that relation they are connected with other concepts, belong to the upper approximation $\overline{R^*}(D)|_C$ of D according to R^* restricted to C.

  Taking into consideration the fact, that there is a need to define the extent to which the information (publication) approximates the concept c, the above approximations are defined for every considered publication. In order to achieve it, we take into account the detailed ontology of a publication. It is checked, whether the concepts from that ontology are in the nearest surrounding of the concept c, and whether they are connected with it by the considered relation R^*.
- On the basis of the assigned sets, we calculate the accuracy of approximation $\alpha_{R^*}(D)|_C$ of D, where $D = \{c\}$ for a given publication consisting of concepts included in C, with respect to the relation R^*:

$$\alpha_{R*}(D)|_C = \begin{cases} 1 & \text{if } c \in C, \\ 0 & \text{if } \overline{R^*}(D)|_C = \emptyset, \\ \frac{card(\underline{R^*}(D)|_C)}{card(\overline{R^*}(D)|_C)} & \text{otherwise.} \end{cases}$$

That coefficient is calculated according to the definition shown in Subsection 27.2.2 for each distinguished relation. In this way, we obtain five coefficients of approximations. In order to attribute only one value, describing a degree of approximation for the concept c, to a given publication,

we connect those values into one, using an aggregation operator. For that purpose, we have used the following operators: minimum, maximum, median, arithmetic average, average weighted with different ways of defining weights. The ways of defining weights, i.e., their generators, have been described in Subsection 27.2.3. The value obtained in this way is still modified ("improved") by taking into consideration the date and place of publication. There have been established the rankings for the year of publishing and the place of publication, i.e., for articles in journals - the ranking of journals, for books - the ranking of publishers, for conference articles - the ranking of conferences. For instance, the ranking for the year of publishing is computed as a ratio of a number of publications in a given year in the system to a number of all publications in the system, and it is established in an identical way for the remaining variants of the rankings. The final coefficient of the publication importance is computed as:

$$\begin{aligned} \textit{importance} = &\textit{importance of accuracy of approximation} \\ &+ \textit{importance of the place of publishing} \\ &+ \textit{importance of the year of publishing}, \end{aligned}$$

where

- the *importance of accuracy of approximation* is defined as a product of the value of the average accuracy of approximation of the concept c for a given publication and $weight_1$;
- the *importance of the place of publishing* is defined as a product of the value taken from the assigned ranking, if the place of publishing is there, and $weight_2$;
- the *importance of the year of publishing* is established as a product of $weight_3$ and the value taken from the assigned ranking, if a given publication is not older than three years;
- the values of weights: $weight_1$, $weight_2$, $weight_3$ are assigned experimentally, i.e., they are set up manually, *e.g.* values $0.5; 0.3; 0.2$ or with the use of defined generators, i.e., arithmetic, pseudo-geometric, ordinary and golden generator. Their definitions have been given in Subsection 27.2.3.

The value of the coefficient of importance for the concept c assigned in this way for a considered publication, is attributed to it, and it causes setting its position in the result ranking, which is created for all publications, and which is presented as the result of searching. In order to take into consideration different kinds of publications and their importance depending on the type, and the convenience of looking through the results of searching, they have been divided into groups according to the type of publication, i.e., into 12 groups reflecting the types of publications defined in the BibTeX specification. They are *e.g. article*, *book*, *booklet*, *inbook*, *incollection*, *inproceedings*, *manual*, *mastersthesis*, *phdthesis*, *proceedings*, *techreport*, *unpublished* [38].

For the process of searching for information (publications) defined in this way, we have conducted the experiments which have been described briefly in Section 27.7.

The presented process of defining publications semantically suitable for the searched concept can be shown by means of the algorithms described below.

Algorithm 27.1. The algorithm describing the process of an "intelligent" searching for bibliographical data on the basis of semantic reflection of the searched concepts, by the summaries of publications (abstracts).

Input : $\mathcal{A}$ - a set of summaries (abstracts) from the RSDS system database, c - a searched concept, μ^{min} - an established minimum level of accuracy of approximation.

Output: $\mathcal{A}_c \subseteq \mathcal{A}$ - a subset of summaries of publications from $\mathcal{A}$, which semantically match the concept c.

$\mathcal{A}_c \leftarrow \emptyset$ **for** *every summary* $A \in \mathcal{A}$ **do**
  Calculate $\mu(A,\{c\})$ defining the importance of A for c using Algorithm 27.5.2
  if $\mu(A,\{c\}) > \mu^{min}$ **then**
    $\mathcal{A}_c \leftarrow \mathcal{A}_c \cup A$
Return $\mathcal{A}_c$

The aim of Algorithm 27.5.2 is to assign the result set which will contain selected publications, i.e., publications, whose coefficient of attribution to the concept fulfills a particular minimal level.

Algorithm 27.2. The algorithm assigning the importance of the summary of a publication for the searched concept.

Input : $\mathcal{O}$ - the field ontology, recorded in the format of an ontology graph, A - a summary of a publication (abstract), c - a searched concept, o - an aggregation operator, *e.g.* minimum, maximum, median, arithmetic average, average weighted with different kinds of defining weights.

Output: $imp(A,\{c\})$ - the importance of A for c.

Attribute neutral element for o to $\mu(A,\{c\})$ Create the set C of concepts from the field ontology $\mathcal{O}$ occurring in the summary A **for** *every relation* $R \in \mathcal{R}$ **do**

$$\mu_R(C,\{c\}) \leftarrow \begin{cases} 1 & \text{if } c \in C, \\ 0 & \text{if } \overline{R^*}(\{c\})|_C = \emptyset, \\ \frac{card(\underline{R^*}(\{c\})|_C)}{card(\overline{R^*}(\{c\})|_C)} & \text{otherwise.} \end{cases} \qquad \mu(A,\{c\}) \leftarrow o(\mu(A,\{c\}),\mu_R(C,\{c\}))$$

$imp(A,\{c\}) \leftarrow$*importance of* $\mu(A,\{c\})$ + *importance of the place of publishing* + *importance of the year of publishing* Return $\mu(A,\{c\})$

Algorithm 27.5.2 assigns accuracy of approximation of a given publication with respect to a searched concept, on the basis of the following steps:

1. Defining the approximations (upper and lower) of the searched concept c in relation to the concepts of the general ontology, existing in a given summary

for each type of relation (connection, inclusion, generalization, specification and synonym relation).
2. Assigning the coefficient to the concept c for every type of relation.
3. Aggregation of the obtained coefficients into one value reflecting the entire adhesion of the concept to a given summary with the use of *e.g.* minimum, maximum, arithmetic average, average weighted with different kinds of defining weights.

This algorithm is repeated for all publications (abstracts) present in the RSDS system.

27.5.3 The Outline of the General Ontology for Rough Set Theory and Its Applications

The ontology defined by an expert, constitutes a proposal of the concept model, concerning rough set theory and its applications. In order to create it, we have distinguished the most essential concepts, the definitions concerning rough sets, and we have defined the dependencies between them. This allowed to try to systematize and regulate widely considered rough set theory.

The model of the ontology consists of the concepts defined on a high level of abstraction, *e.g. System*, *Model*, which are detailed with the concepts of lower levels. Accepting such a way of defining the dependencies between the concepts, allows free introduction of the concept hierarchy. On the basis of Definition 27.6, the relations have been defined between particular concepts at different levels of hierarchy. Those relations enable us to realize different ways of connections. Every relation is a binary relation, and it connects two concepts together: the source concept and the target concept.

- *Connection relation* $R_{\bowtie}$ - that relation defines whether a given concept (source) depends on the other concept (target) as for meaning, and whether it is connected with it in any way, *e.g.* the concept *Lower Approximation* is connected with the concept *Rough Set*. For this relation, we can define the reverse relation between the concepts, i.e., for a reverse direction, *e.g. Rough Set* is connected with *Lower Approximation*.

The further relations (see paragraph 27.2.2) caused regulation of the defined concepts in the graph structure, representing the described model. The ontology editor *Protégé* works on such a structure. This editor is used at the initial phase of research. Creating such a structure in order to describe the designed model, allows easy and intuitive recording, interpretation and modification.

- *Inclusion relation* $R_{\subset}$ - it states that the extent of meaning of the source concept is included in the extent of meaning of the target concept, *e.g.* the extent of meaning of the concept *Model* includes the extent of meaning of the concept *Data Model*. For this relation, we cannot define a reverse relation, i.e., the

reverse statement saying that the extent of meaning of the concept *Model* is included in the extent of meaning of the concept *Data Model* is not true.
- *Specification (detailing) relation* $R_{\triangleright}$ - this relation states, that the source concept is defined more precisely than the target concept. The reverse relation for the specification relation is the generalization relation.
- *Generalization relation* $R_{\triangleleft}$ - it defines, that the source concept is the concept which is generalized by the target concept. The reverse relation for the generalization relation is the specification relation.
- *Synonym relation* $R_{\sim}$ - it defines two concepts: the source concept and target one, which are equivalent in meaning.

At the beginning of the research, we used the *Protégé* editor to create and define the ontology. However, as the processing and visualizing were difficult, the ontology has been written in the *GraphML* format, which made its further formulation independent from the *Protégé* editor. The further edition of the ontology is now possible in any tool which allows to use the *GraphML* format. In the conducted research, in order to achieve the aim, we used our own software and the *yED Graph Editor* for visualization.

The formal record of the definitions of the ontologies and the defined relations, together with their features, is presented in paragraph 27.2.2. The ontology proposed by an expert, was formally verified, i.e., it was checked, whether the specified relations fulfill the defined features. After such verification, the number of relations enlarged significantly. This process has been conducted automatically with the use of our own software.

The highest level of abstraction (first level) defines the general concepts of rough set theory and its applications. These are such concepts as: *System, Domain, Reasoning, Model, Lower Approximation, Upper Approximation, Discernibility Matrix, Covering, Decision Making, Decision Class, Attribute, Reduct, Operator, Rule, Reduct Set, Rough Set Theory, Classifier Evaluation, Classifier, Object, Neurons, Database*.

27.6 The Description of the RSDS System

The created and formulated RSDS bibliographical system is an experimental environment for the conducted research. This system contains bibliographical descriptions of publications related to rough set theory and its applications [21–24, 30]. The system is available at `http://rsds.univ.rzeszow.pl`. At present, in the database of the system, there are 3857 bibliographical descriptions of scientific publications. The RSDS system also contains:

- information related to the software connected to rough set theory and its applications,
- biographies of people meritorious for rough set theory and its applications,

- the contact data of the authors of those publications which are present in the data base of that system.

The system has been formulated in the client-server technology. This means that the data for the system as well as the mechanisms servicing these data are located on the server, while a user, by means of the client-Internet browser, uses the content of the system.

Fig. 27.2 The main window of the system.

27.6.1 The Logical Structure of the System

The construction of the RSDS system can be divided into four functional layers. Each of the layers fulfills particular tasks by means of the modules included in it:

- The presentation layer with the module of a graphical interface.
- The application layer with the modules of logging, adding / editing, searching, statistics-graph, downloading, supplementary (biographies of people, software, maps).
- The communication layer with the module of communication with the database.
- The physical layer with the database.

The aim of the modules from the presentation layer is to communicate with the user.

In the application layer, there are modules which realize the main functionality of the system. The module of logging is responsible for the correct service of the process of logging the users in and out, as well as storing the information about the users logged into the system. The module of adding and editing, operates the process of introducing new data into the system or editing the existing data. It checks the correctness of the data introduced into the system, and the correct attribution of the

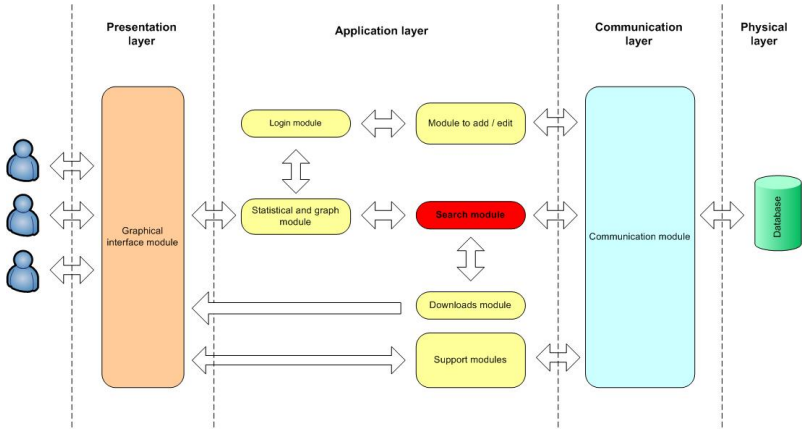


Fig. 27.3 The logical structure of the system.

data to the user - "their owner". The module of searching is a module, which realizes the process of searching for the descriptions of publications which fulfill the user's criteria. This is the module whose role is to facilitate the conducted research. The statistics-graph module is the module which analyzes the data included in the system as well as the users' activity . The data analysis takes place in different aspects, i.e., concerning publications, authors of publications, and the dependencies between them. The aim of this module is also to present the obtained results of analysis. The module of downloading gives the users access to different variants of downloading data from the system in the form of prepared bibliographical lists. The supplementary modules expand the basic functionality of the system by adding the biographies of people meritorious to the development of rough set theory and its applications, available software related to rough sets, the map of the world illustrating where in the world the theory mentioned above develops.

The communication layer has a module which is responsible for correct communication with the database, in which the data stored in the system are kept.

The physical layer includes the relational database where the data represented in the system are stored.

27.6.2 The Functional Capabilities of the System

The basic functionalities of the system are:

- Adding data – online or automatically.
- Editing the existing data.
- Searching for the data.
- Registration of the users in the system.

- Saving data in the files.
- Sending files with data to the administrator.
- Servicing the users' comments.
- Statistics, statistics-graph analysis, publication classifier.
- Help.

The scope and the content of the system is still being broadened.

The RSDS system is a bibliographical system . In order to store the possessed information in the simplest possible form, eliminating the overload (data redundancy), the data for the system are stored a relational data base. The data of the system are saved in the BibTeX format [38]. The reasons for that choice were well defined and unified structures of a description. In addition, there was one more capability of the system added to it, which is the ability to obtain, from the system, bibliographical descriptions in the BibTeX format. This allows for automatic generation of bibliographical lists, and adding them to the prepared publications.

The system had to be equipped with the data to give access to it. The introduction of data, as well as other operations allowing data modification require to confirm credibility of a user through logging into the system. New users, who want to get an access to the full functionality of the system, must register in it.
Introducing data can be done in two independent ways:

1. Introducing data online by means of predefined forms. According to this method, the data are introduced in two phases. In the first phase, one has to fill the form with the data describing a publication. In the second phase, one has to introduce information about authors or editors of a publication. The data of particular authors (editors) must be introduced separately, until the moment of introducing information about every person. Each of the introduced persons will be attributed to information about a publication, introduced at the beginning of the whole process. This process is illustrated in Figure 27.4.

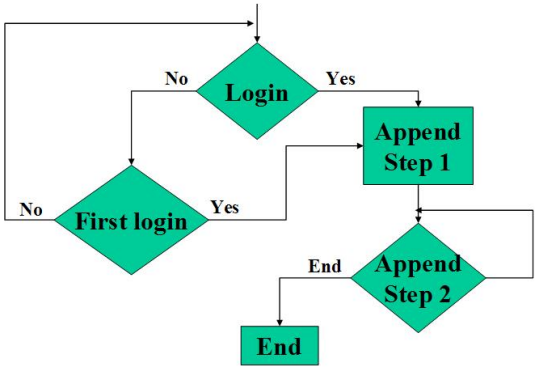


Fig. 27.4 A block diagram representing the process of adding data (publications) into the RSDS system.

2. Introducing data by means of a script, allowing to read the files in the BibTeX format and saving information in the system properly.

Until now, the available ways of searching allowed only searching for information by matching the result to a demanded template included in a query. So far, searching for information in the RSDS system has been done in two main ways (Figure 27.5):

- alphabetical searching, according to the defined keys, i.e., titles of publications, their authors, publishers, names of conferences, journals, or years of issue,
- advanced searching, based on defining the criteria which must be fulfilled by a demanded description of a publication.

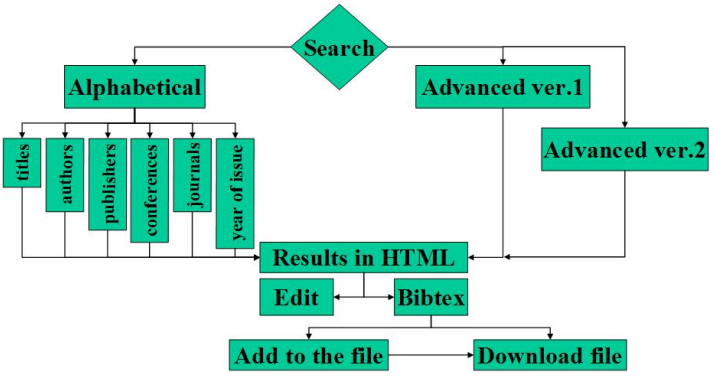


Fig. 27.5 A block diagram representing the process of searching for information in the RSDS system.

Each of the possibilities of searching for information in the RSDS system available now, has both, positive and negative aspects. Thus, alphabetical searching works well, when a user knows *e.g.* the author of the demanded publication, the title of the journal, where the publication appeared, the editor who issued the publication, or the year of issue. The weaker aspect of this kind of searching is, however, the fact, that with the lack of fairly precise information about a demanded publication, the system usually finds a great number of descriptions of publications which fulfill the criteria of searching. As a result, the criteria often must undergo a laborious selection. Whereas in advanced searching, the user defines the criteria which a demanded publication must fulfill, and depending on accuracy of choice of the criteria, we obtain more or less adequate results of searching. Still, the problem of further selection of results is, in many cases, unavoidable. The current course of the process of searching for information in the RSDS system is displayed in Figure 27.6.

Apart from bibliographical information (represented in a descriptive form or in the BibTeX format), the system gives access to a whole range of different statistics and the results of analysis of the included data.

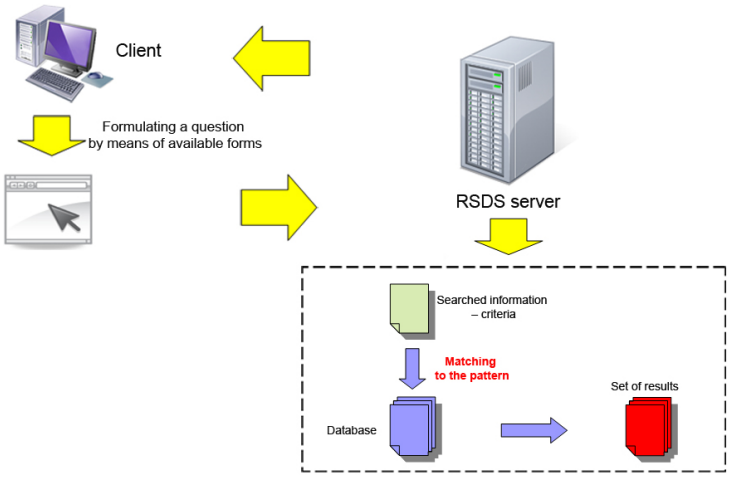


Fig. 27.6 The current course of the process of searching for information in the RSDS system.

For more information about the destination of the system and its functional capabilities, the reader can see papers [31–34].

27.7 Experiments

27.7.1 Methodology of Experiments

In order to verify the correctness of the proposed methodology for searching information in data bases we have performed a series of tests and experiments, especially the following ones:

- We have made automatic verification of the general ontology from the point of view of the correctness of relationships defined in it. A special software tool has been created to make this verification.
- We have evaluated a quality of searching information in the RSDS system using our approach. In this case, we have performed experiments consisting in sending queries to the system. Obtained results, semantically matched to the query, have been compared with results of similar experiments using the VSM (Vector Space Model) method [29]. First, this method has been used to represent data included in RSDS. Next, this method has been used to calculate a membership degree of a searched publication to the resulting set by means of determining the similarity of vectors describing a query and a publication. The similarity has been calculated using the cosine metric (the cosine of the angle between vectors). In our experiments, we have used only vectors with

non-negative coordinates. It causes that the cosine values are included in the interval $[0, 1]$. The value equal to 1 means that the similarity of two vectors is complete (vectors overlap, both methods give the same results). The value equal to 0 means that two vectors are completely dissimilar (vectors are orthogonal, methods give extremely different results). Resulting sets containing publications have been compared using several metrics (Euclidean, Minkowski, Hamming, City, Chebyschev) mentioned in Subsection 27.2.4 enabling us to determine the mutual orientation of the two vectors representing resulting sets.

27.7.2 Comments

Detailed results of experiments are omitted in this paper due to their large size. In this subsection, we give only some important comments. They are as follows:

- The resulting set for the ontological method is a superset of the resulting set for the VSM method, i.e., it includes additional descriptions of publications in comparison with the VSM method. Publications covered by both methods include a searched concept in their descriptions. The vector method gives only those publications. Searching based on the ontology enables us to add new descriptions semantically matched (i.e., not only exactly matched) to our query. It is worth noting that publications including exactly searched concepts sometimes do not satisfy well expectation of the user putting the query.
- Verification of the general ontology from the point of view of the formal correctness of relationships between concepts causes a complement the original ontology to missing relationships. It causes a significant growth of the number of relationships between concepts. An extension of the ontology to new relationships affects positively a quality of the process of information searching.
- Selection of the aggregation operator for accuracy of approximation for particular relations as well as a value of the first weight used in calculations have a big impact on the disparity of results for ontological searching. Simultaneously, we can observe a smaller impact of selection of the two remaining weights for calculation of the final value of accuracy of approximation. For selection of weights, we have used the special generators shown is Subsection 27.2.3.
- Using the general ontology in the experiments, after a formal verification of relationships, we can see a general trend toward increasing of the angle between vectors. It means that some additional information appears, it is not present in results obtained on the basis of the VSM method.

27.8 Summary and Final Conclusions

The aim of the conducted research is to create the mechanisms of automatic administration of information (bibliographical description) in the bibliographical system,

such as the already described RSDS system. The research is based on rough set theory, enriched with the field knowledge, represented by the ontology. This allows automatic - "intelligent" recognition of what a given publication concerns and describes. At the moment, the formulated mechanism is used during the process of searching for a demanded information (publication). The use of such an approach in operating the system which contains a large amount of information allows the improvement of its work during an interaction with the user. It seems, that generalization of that mechanism will be the origin of the "intelligence" of different kinds of systems and browsers available in the Internet.

27.8.1 Directions of Further Research

On the basis of the results achieved, we are planning the following progress of the research:

- formulating the mechanism of logical verification of the defined connections in the field ontology;
- improving the effectiveness of the process of searching for information;
- formulating the mechanism of automatic broadening of the field ontology;
- automatization of the procedure of processing the possessed information, limiting the interference of an expert.

References

1. Ambroszkiewicz, S., Mikułowski, D.: Web services and semantic Web, ideas and technologies. EXIT, Warsaw (2006) (in Polish)
2. Bennacer, N., Karoui, L.: A framework for retrieving conceptual knowledge from web pages. In: Tummarello, G., Bouquet, P. (eds.) SWAP 2005 - Semantic Web Applications and Perspectives, Proceedings of the 2nd Italian Semantic Web Workshop, December 14-16. University of Trento, Trento (2005)
3. Buitelaar, P., Cimiano, P., Magnini, B.: Ontology Learning From Text: Methods, Evaluation and Applications. IOS Press (2005)
4. Corby, O., Dieng-Kuntz, R., Faron-Zucker, C.: Querying the semantic Web with corese search engine. In: Proceedings of 16th European Conference on Artificial Intelligence (ECAI/PAIS), Valencia, Spain, August 22-27, pp. 705–709. IOS Press, Amsterdam (2004)
5. Dell Orletta, F., Lenci, A., Montemagni, S., Marchi, S., Pirrelli, V., Venturi, G.: Acquiring legal ontologies from domain-specific texts. In: Proccedings of LangTech 2008, Roma, Italy, February 28-29, pp. 98–101 (2008)
6. Englmeier, K., Murtagh, F., Mothe, J.: Domain ontology: automatically extracting and structuring community language from texts. In: Proceedings of International Conference Applied Computing (IADIS), Salamanca, Spain, February 18-20, pp. 59–66 (2007)
7. Helfin, J., Hendler, J.: Searching the Web with SHOE. In: Proceedings of the AAAI Workshop on AI for Web Search, Austin, Texas, USA, July 30-August 1, pp. 35–40. AAAI Press (2000)

8. Gasperin, C., Gamallo, P., Agustini, A., Lopes, G., Lima, V.: Using syntactic contexts for measuring word similarity. In: Proceedings of the Semantic Knowledge Acquisition and Categorisation Workshop, ESSLLI 2001, Helsinki, Finland (2001)
9. Grefenstette, G.: Evaluation techniques for automatic semantic extraction: Comparing syntatic and window based approaches. In: Pustejovsky, J., Boguraev, B. (eds.) Corpus Processing for Lexical Aquisition, pp. 205–216 (1995)
10. Grochowalski, P., Pancerz, K.: The outline of an ontology for the rough set theory and its applications. Fundamenta Informaticae 93(1-3), 143–154 (2009)
11. Gruber, T.R.: A translation approach to portable ontology specyfications. Knowledge Acquisition 5(2), 199–220 (1993)
12. Guha, R., McCool, R., Miller, E.: Semantic search. In: Proceedings of the 12th International Conference on World Wide Web, Budapest, Hungary, May 20-24, pp. 700–709 (2003)
13. Jacob, E.K.: Ontologies and the semantic web. Bullettin of ASIST, 19–22 (2003)
14. Kłopotek, M.A.: Intelligent Search Engines. EXIT, Warsaw (2001) (in Polish)
15. Köhler, J., Philippi, S., Specht, M., Rüegg, A.: Ontology based text indexing and querying for the semantic Web. Knowledge-Based Systems 19(8), 744–754 (2006)
16. Kruk, S.R., Synak, M., Zimmermann, K.: Marcont - Integration ontology for bibliographic description formats. In: Proceedings of the 2005 International Conference on Dublin Core and Metadata Applications: Vocabularies in Practice (DCMI 2005), Madrid, Spain, September 12-15, pp. 231–234 (2005)
17. Lopez, V., Pasin, M., Motta, E.: AquaLog: An Ontology-Portable Question Answering System for the Semantic Web. In: Gómez-Pérez, A., Euzenat, J. (eds.) ESWC 2005. LNCS, vol. 3532, pp. 546–562. Springer, Heidelberg (2005)
18. Mittal, H., Singh, J., Sachdeva, J.: ARAGOG Semantic Search Engine: Working, Implementation and Comparison with Keyword-Based Search Engines. In: Fred, A., Dietz, J.L.G., Liu, K., Filipe, J. (eds.) IC3K 2009. CCIS, vol. 128, pp. 177–186. Springer, Heidelberg (2011)
19. Nahotko, M.: Metadata, a way of sorting the Internet out. Jagiellonian University Press, Cracow (2004) (in Polish)
20. Neches, R., Fikes, R.E., Finin, T., Gruber, T.R., Patil, R., Senator, T., Swartout, W.R.: Enabling technology for knowledge sharing. AI Magazine 12(3), 36–56 (1991)
21. Pawlak, Z.: Rough Sets. International Journal of Computer and Information Sciences 11, 341–356 (1982)
22. Pawlak, Z., Grzymała-Busse, J.W., Słowiński, R., Ziarko, W.: Rough Sets. Communications of the ACM 38(11), 88–95 (1995)
23. Pawlak, Z., Skowron, A.: Rough sets and boolean reasoning. Information Sciences 177(1), 41–73 (2007)
24. Polkowski, L.T.: Rough Sets. In: Mathematical Foundations. ASC. Physica-Verlag, Heidelberg (2002)
25. Porter, M.F.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)
26. Protaziuk, G., Kryszkiewicz, M., Rybiński, H., Delteil, A.: Discovering Compound and Proper Nouns. In: Kryszkiewicz, M., Peters, J.F., Rybiński, H., Skowron, A. (eds.) RSEISP 2007. LNCS (LNAI), vol. 4585, pp. 505–515. Springer, Heidelberg (2007)
27. Rocha, C., Schwabe, D., de Aragao, M.P.: A hybrid approach for searching in the semantic Web. In: Proceedings of the 13th International Conference on World Wide Web, New York, USA, May 17-22, pp. 374–383 (2004)
28. Rybiński, H., Kryszkiewicz, M., Protaziuk, G., Jakubowski, A., Delteil, A.: Discovering Synonyms Based on Frequent Termsets. In: Kryszkiewicz, M., Peters, J.F., Rybiński, H., Skowron, A. (eds.) RSEISP 2007. LNCS (LNAI), vol. 4585, pp. 516–525. Springer, Heidelberg (2007)
29. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Communications of the ACM 18(11), 613–620 (1975)
30. Skowron, A., Pal, S.K. (eds.): Special Volume: Rough Sets, Pattern Recognition and Data Mining. Pattern Recognition Letters, vol. 24(6) (2003)

31. Suraj, Z., Grochowalski, P.: The Rough Set Database System: An Overview. In: Tsumoto, S., Słowiński, R., Komorowski, J., Grzymała-Busse, J.W. (eds.) RSCTC 2004. LNCS (LNAI), vol. 3066, pp. 841–849. Springer, Heidelberg (2004)
32. Suraj, Z., Grochowalski, P.: Functional Extension of the RSDS System. In: Greco, S., Hata, Y., Hirano, S., Inuiguchi, M., Miyamoto, S., Nguyen, H.S., Słowiński, R. (eds.) RSCTC 2006. LNCS (LNAI), vol. 4259, pp. 786–795. Springer, Heidelberg (2006)
33. Suraj, Z., Grochowalski, P.: Patterns of collaborations in rough set research. In: Gomez, Y., Bello, R., Falcon, R. (eds.) Proceedings of the International Symposium on Fuzzy and Rough Sets, ISFUROS 2006, Santa Clara, Cuba, December 5-8 (2006)
34. Suraj, Z., Grochowalski, P., Garwol, K., Pancerz, K.: Toward intelligent searching the rough set database system (RSDS): An ontological approach. In: Szczuka, M., Czaja, L. (eds.) Proceedings of the CS&P 2009 Workshop, CS&P 2009, Krakow, September 28-30, vol. 1-2, pp. 574–582. Warsaw University, Poland (2009)
35. Velardi, P., Fabriani, P., Missikoff, M.: Using text processing techniques to automatically enrich a domain ontology. In: Proceedings of the international conference on Formal Ontology in Information Systems (FOIS 2001), pp. 270–284. ACM, New York (2001)
36. Lei, Y., Uren, V.S., Motta, E.: SemSearch: A Search Engine for the Semantic Web. In: Staab, S., Svátek, V. (eds.) EKAW 2006. LNCS (LNAI), vol. 4248, pp. 238–245. Springer, Heidelberg (2006)
37. Zhou, Q., Wang, C., Xiong, M., Wang, H., Yu, Y.: SPARK: Adapting Keyword Query to Semantic Search. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 694–707. Springer, Heidelberg (2007)
38. BibTeX, http://www.bibtex.org/
39. Dublin Core Metadata Initiative: DC-2005: International Conference on Dublin Core and Metadata Applications "Metadata Vocabularies in Practice", Leganés (Madrid), Spain, September 12-15. Dublin Core Metadata Initiative (2005) http://dcpapers.dublincore.org/ojs/pubs/issue/view/28
40. Dublin Core Metadata Initiative: DC-2006: International Conference on Dublin Core and Metadata Applications "Metadata for Knowledge and Learning", Manzanillo, Colima, Mexico, October 3-6. Dublin Core Metadata Initiative (2006), http://dcpapers.dublincore.org/ojs/pubs/issue/view/29
41. Dublin Core Metadata Initiative: DC-2008: International Conference on Dublin Core and Metadata Applications "Metadata for Semantic and Social Applications", Berlin, Germany, September 22-26. Dublin Core Metadata Initiative (2008), http://dcpapers.dublincore.org/ojs/pubs/issue/view/32
42. Dublin Core Metadata Initiative: DC-2009: International Conference on Dublin Core and Metadata Applications "Semantic Interoperability of Linked Data", Seoul, Korea, October 12-16. Dublin Core Metadata Initiative (2009), http://dcpapers.dublincore.org/ojs/pubs/issue/view/33
43. JeromeDL - e-library with semantics, http://www.jeromedl.org/

Chapter 28
Design and Verification of Rule-Based Systems for Alvis Models

Marcin Szpyrka and Tomasz Szmuc

Abstract. Alvis is a modelling language designed for embedded systems that provides a possibility of a formal model verification. Because of the fact that many embedded systems contain a rule-based system as a part of them, it is necessary to provide a possibility to include such systems into Alvis models. Alvis combines flexible graphical modelling of interconnections among agents with a high level programming language used for the description of agents behaviour. The most natural way of including a rule-based system into an Alvis model is to encode it in the Haskell functional language. Some Haskell features like lazy evaluation, pattern matching, high level functions etc. make it a very attractive proposition from the rule-based systems' engineering point of view. The paper presents a method of encoding and verification of rule-based systems with Haskell to include them into Alvis models.

Keywords: Rule-based systems, railway traffic management system, Haskell, Alvis, formal modelling and verification.

28.1 Introduction

Rule-based systems are widely used in various kinds of computer systems, *e.g.* expert systems, decision support systems, monitoring and control systems, diagnostic systems, etc. They constitute an easy way of knowledge encoding and interpreta-

Marcin Szpyrka
AGH University of Science and Technology, Department of Applied Computer Science, Al. Mickiewicza 30, 30-059 Cracow, Poland
e-mail: mszpyrka@agh.edu.pl

Tomasz Szmuc
AGH University of Science and Technology, Department of Applied Computer Science, Al. Mickiewicza 30, 30-059 Cracow, Poland
e-mail: tsz@agh.edu.pl

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 539–558.
springerlink.com

tion, however, the design and verification of them are often a time-consuming and difficult task [7], [10].

Various approaches to rule-based systems analysis and verification are considered, and most of them is based on Prolog programming language ([5]). Prolog (PROgramming in LOGic) is a declarative programming language and seems to be perfect for building rule-based systems. However, some Prolog features like side-effects (assert, retract, cut operator) can seriously reduce confidence in the correctness of an implemented rule-based system. This chapter deals with a Haskell [11] approach to encoding rules and algorithms for rule-based system verification. One of the main advantages of this approach is the possibility of including such systems into Alvis models.

Alvis [17], [15] is a formal modelling language designed especially for embedded systems. It combines possibilities of formal models verification with flexibility and simplicity of practical programming languages. Moreover, Alvis combines a hierarchical graphical modelling with a high level programming language. An Alvis model consists of three layers:

Graphical layer is used to define data and control flow among distinguished parts of the system under consideration that are called *agents*. The layer takes the form of a hierarchical graph and supports both *top-down* and *bottom-up* approaches to systems development.

Code layer is used to describe the behaviour of individual agents. It uses both Haskell functional programming language and original Alvis statements (Alvis Code Language).

System layer depends on the model running environment i.e. the hardware and/or operating system. The layer is the predefined one and is necessary for a model simulation and verification.

Alvis uses Haskell to define data types for parameters and to define functions for data manipulation. Encoding a rule-based system as a Haskell function is the simpliest way to include the system into the corresponding Alvis model. As it has been presented in this chapter, Haskell can be also used to verify the most important properties of a rule-based system.

In the presented approach, rule-based systems take the form of a decision table [12], but non-atomic values of attributes are possible. Each cell of such a decision table should contain a formula, which evaluates to a Boolean value for condition attributes, and to a single value (that belongs to the corresponding domain) for decision attributes ([13]). It means that for any condition attribute, we can write a formula that describes suitable values of the attribute in the corresponding rule. On the other hand, for any decision attribute we can write a formula that contains names of condition attributes and evaluates to a single value belonging to the domain of the decision attribute.

The scope of the paper is as follows. A railway traffic management system used to illustrate the presented approach is described in Section 28.2. The proposed Haskell form of rule-based systems and Haskell verification algorithms are presented in Section 28.3. A short presentation of the Alvis modelling language is given in

Section 28.4. Section 28.5 contains parts of an Alvis model of the considered railway traffic management system. The model is used to point out the place of rule-based systems in Alvis models. A short summary is given in the final section.

28.2 Example

To illustrate the presented approach, an example of a railway traffic management system for a real train station has been choosen. Czarna Tarnowska is a small train station belonging to the Polish railway line no. 91 from Cracow to Medyka. The topology of the train station with original signs is shown in Fig. 28.1. The letters A, B, D, etc. stand for color light signals, the symbols Z3, Z4, Z5, etc. stand for turnouts and JTA, JTB, JT1, etc. stand for track segments. Some simplifications have been introduced to reduce the size of the model. We are not interested in controlling the local shunts so the track segment JT6 will not be considered. We assume that the light signals display only two signals: *stop* and *way free*. Moreover, outside the station, the trains can ride using the right track only.

The considered system is used to ensure safe riding of trains through the station. It collects some information about current railway traffic and uses a rule-based system to choose routes for trains. A train can ride through the station only if a suitable route has been prepared for it, i.e., suitable track segments must be free, we have to set turnouts and light signals and to guarantee exclusive rights to these elements for the train. Required positions of turnouts for all possible routes are shown in Table 28.1, where the used symbols stand for:

- + closed turnout (the straight route);
- – open turnout (the diverging route);
- o+ closed turnout (for safety reasons).

Let us focus on selected symbols from Table 28.1. The symbol B4 stands for the input route from the light signal B to the track no. 4. The symbol F2W stands for the output route from the track no. 2 (from the light signal F) to the right (to Wola Rzędzińska), etc. The route B4 can be used by a train only if: turnouts 7, 8, 15, 16 are closed, turnouts 3, 4, 6 are open, and the track segments JTB, JT4, JZ4/6 (a segment between turnouts 4 and 6), JZ7 (diagonal segment leading to the turnout 7) and JZ16 are free.

Some routes cannot be set at the same time because of different position of turnouts or for safety reasons. Mutually exclusive routes are presented in Table. 28.2, where the used symbols stand for:

- x – mutually exclusive (different position of turnouts);
- xx – mutually exclusive (safety reasons).

The system is expected to choose suitable routes for moving trains. It should take under consideration that some trains should stop at the platform, while others are only moving through the station and two routes (an input and an output one) should be prepared for them. In such a case, if it is not possible to prepare two routes, only an input one should be prepared.

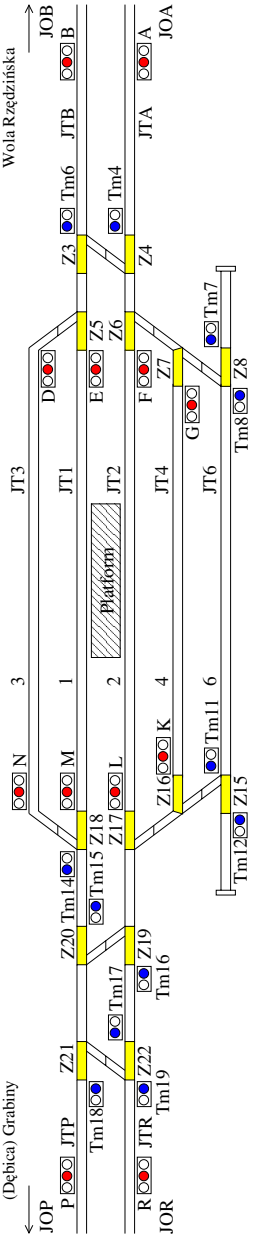


Fig. 28.1 Czarna Tarnowska – topology of the train station

Table 28.1 Required position of turnouts for all possible routes

	Turnouts								
Routes	3/4	5	6	7/8	15/16	17	18	19/20	21/22
B1	+	+							
B2	–		+	o+					
B3	+	–							
B4	–		–	+	o+				
R2				o+	o+	+		+	+
R4				o+	+	–		+	+
F2W	+		+	o+					
G2W	+		–	+					
K1D					+	–		–	+
L1D					o+	+		–	+
M1D							+	+	+
N1D							–	+	+

Table 28.2 Relationships between routes

	B1	B2	B3	B4	R2	R4	F2W	G2W	K1D	L1D	M1D	N1D
B1	–	x	x	x								xx
B2	x	–	x	x	xx		x	x				
B3	x	x	–	x								
B4	x	x	x	–	xx	xx	x	x				
R2		xx		xx	–	x		xx	x	x		
R4				xx	x	–			x	x		
F2W		x		x			–	x				
G2W		x		x	xx		x	–				
K1D					x	x			–	x	x	x
L1D					x	x			x	–	x	x
M1D									x	x	–	x
N1D	xx								x	x	x	–

The decision table for the considered system is presented in Table 28.3. The table contains 20 conditions and 2 decision attributes. The condition attributes provide information about:

- current position of the train (attribute JT) — before the light signal B, F, G, etc.;
- type of the train (attribute TT) — only moves through the station (1) or must stop at the platform (2);
- current status of track segments (attributes JT1, JT2, JT3, JT4, JOA, JOP) — a segment is free (0) or it is taken (1);

Table 28.3 Decision table

	JT	TT	JT1	JT2	JT3	JT4	JOA	JOP	B1	B2	B3	B4	R2	R4	F2W	G2W	K1D	L1D	M1D	N1D	In	Out
*R*1	B	1	0					0	0	0	0	0					0	0	0	0	B1	M1D
*R*2	B	1	1		0			0	0	0	0	0					0	0	0	0	B3	N1D
*R*3	B	1	1	0	1			0	0	0	0	0	0	0	0	0	0	0	0	0	B2	L1D
*R*4	B	1	1	1	1	0		0	0	0	0	0	0	0	0	0	0	0	0	0	B4	K1D
*R*5	R	1		0			0			0		0	0	0	0	0	0	0			R2	F2W
*R*6	R	1		1		0	0			0		0	0	0	0	0	0	0			R4	G2W
*R*7	B	2	0						0	0	0	0								0	B1	None
*R*8	B	2	1	0					0	0	0	0	0		0	0					B2	None
*R*9	B	2		0					0	0	0	0	0		0	0	0	0	0	1	B2	None
*R*10	R	2		0						0		0	0	0		0	0	0			R2	None
*R*11	B	1	0					1	0	0	0	0								0	B1	None
*R*12	B	1	0						0	0	0	0					1	0	0	0	B1	None
*R*13	B	1	0						0	0	0	0					0	1	0	0	B1	None
*R*14	B	1	0						0	0	0	0					0	0	1	0	B1	None
*R*15	B	1	1	0	1			1	0	0	0	0	0		0	0					B2	None
*R*16	B	1	1	0	1				0	0	0	0	0	1	0	0	0	0			B2	None
*R*17	B	1	1	0	1				0	0	0	0	0		0	0	1	0	0	0	B2	None
*R*18	B	1	1	0	1				0	0	0	0	0		0	0	0	1	0	0	B2	None
*R*19	B	1	1	0	1				0	0	0	0	0		0	0	0	0	1	0	B2	None
*R*20	B	1		0	1				0	0	0	0	0		0	0	0	0	0	1	B2	None
*R*21	B	1	1		0			1	0	0	0	0									B3	None
*R*22	B	1	1		0				0	0	0	0					1	0	0	0	B3	None
*R*23	B	1	1		0				0	0	0	0					0	1	0	0	B3	None
*R*24	B	1	1		0				0	0	0	0					0	0	1	0	B3	None
*R*25	B	1			0				0	0	0	0					0	0	0	1	B3	None
*R*26	B	1	1	1	1	0		1	0	0	0	0	0	0	0	0					B4	None
*R*27	B	1	1	1	1	0			0	0	0	0	0	0	0	0	1	0	0	0	B4	None
*R*28	B	1	1	1	1	0			0	0	0	0	0	0	0	0	0	1	0	0	B4	None
*R*29	B	1	1	1	1	0			0	0	0	0	0	0	0	0	0	0	1	0	B4	None
*R*30	B	1		1	1	0			0	0	0	0	0	0	0	0	0	0	0	1	B4	None
*R*31	R	1		0			1			0		0	0	0		0	0	0			R2	None
*R*32	R	1		0						0		0	0	0	1	0	0	0			R2	None
*R*33	R	1		1		0	1					0	0	0			0	0			R4	None
*R*34	R	1				0			0	1	0	0	0	0			0	0			R4	None
*R*35	R	1		1		0						0	0	0	1	0	0	0			R4	None
*R*36	R	1				0						0	0	0	0	1	0	0			R4	None
*R*37	K	1				1		0					0	0			0	0	0	0	None	K1D
*R*38	L			1				0					0	0			0	0	0	0	None	L1D
*R*39	M		1					0									0	0	0	0	None	M1D
*R*40	N	1			1			0	0								0	0	0	0	None	N1D
*R*41	F			1			0			0		0			0	0					None	F2W
*R*42	G	1				1	0			0		0	0		0	0					None	G2W

- already prepared routes (attributes B1, B2, B3, etc.) — a route is already set (1) or not (0).

The decision attributes In and Out represent the input and output routes (that will be prepared for the train), respectively. Domains for these attributes are defined as follows:

- $D_{\mathrm{JT}} = \{\mathrm{B}, \mathrm{F}, \mathrm{G}, \mathrm{K}, \mathrm{L}, \mathrm{M}, \mathrm{N}, \mathrm{R}\}$,

- $D_{\mathrm{TT}} = \{1,2\}$,
- $D_{\mathrm{JT1}} = D_{\mathrm{JT2}} = \cdots = D_{\mathrm{N1D}} = \{0,1\}$,
- $D_{\mathrm{In}} = \{\mathrm{B1},\mathrm{B2},\mathrm{B3},\mathrm{B4},\mathrm{R2},\mathrm{R4},\mathrm{None}\}$,
- $D_{\mathrm{Out}} = \{\mathrm{F2W},\mathrm{G2W},\mathrm{K1D},\mathrm{L1D},\mathrm{M1D},\mathrm{N1D},\mathrm{None}\}$.

28.3 Haskell Form of Rule-Based Systems

A decision table, from the Haskell point of view, can be treated as a function that takes values of condition attributes as its arguments and provides values of decision attributes as its result. It is convenient to define two composite data types (tuples) *e.g. Condition* and *Decision* to denote the Cartesian product if condition and decision attributes, respectively. Such data types for the rule-based system shown in Table 28.3 are defined as presented in Listing 28.1.

```
data JTDomain = B | F | G | K | L | M | N | R
data InDomain = B1 | B2 | B3 | B4 | R2 | R4 | INone
data OutDomain = F2W | G2W | K1D | L1D | M1D | N1D | ONone
type Condition = (JTDomain,Int,Int,Int,Int,Int,Int,Int,Int,Int,
                  Int,Int,Int,Int,Int,Int,Int,Int,Int,Int)
type Decision = (InDomain,OutDomain)
```

Listing 28.1 Data type used for the considered rule-based system

Because a construction function name must be unique, we used names *INone* and *ONone* instead of *None*.

Haskell functions can be defined piece-wise, meaning that we can write one version of a function for certain parameters and then another version for other parameters. This approach uses the so-called *pattern matching*, in which a sequence of syntactic expressions called *patterns* is used to choose between a sequence of results of the same type. If the first pattern is matched, then the first result is chosen; otherwise, the second pattern is checked, etc. The *wild card* pattern _ (underscore) can be also used that matches any value. Using the wild card pattern, we can indicate that we do not care what is present in a part of a pattern. Of course, more than one wild card can be used in a single pattern. Moreover, each piece of a function definition can take the form of the so-called *guarded equation*. In such a case, a Boolean expression is put after the | symbol (read as "such that") that must be satisfied beside the pattern matching.

The *rbs* function shown in Listing 28.2 is a Haskell implementation of the considered rule-based system. The listing contains the source code for the first five decision rules only. Other decision rules from the table have been implemented similarly. The function takes information about an input state and provides a single decision (the first matching decision rule is used).

```
rbs :: Condition -> Decision
rbs(B,1,0,_,_,_,_,0,0,0,0,0,_,_,_,_,0,0,0,0) = (B1,M1D)
rbs(B,1,1,_,0,_,_,0,0,0,0,0,_,_,_,_,0,0,0,0) = (B3,N1D)
rbs(B,1,1,0,1,_,_,0,0,0,0,0,0,0,0,0,0,0,0,0) = (B2,L1D)
rbs(B,1,1,1,1,0,_,0,0,0,0,0,0,0,0,0,0,0,0,0) = (B4,K1D)
rbs(R,1,_,0,_,_,0,_,_,0,_,0,0,0,0,0,0,0,_,_) = (R2,F2W)
```

Listing 28.2 Haskell implementation of the considered decision table

To verify a Haskell implementation of a decision table, we have to define another function that determines all possible decisions for an input state or generates an empty list, if none decision can be undertaken. The Haskell code for the *allDecisions* function is shown in Listing 28.3.

```
allDecisions' :: Int -> Condition -> [(Int, Decision)]

allDecisions' 1
  (B,1,0,jt2,jt3,jt4,joa,0,0,0,0,0,r2,r4,f2w,g2w,0,0,0,0)
  = [(1,(B1,M1D))] +!+ allDecisions' 2
  (B,1,0,jt2,jt3,jt4,joa,0,0,0,0,0,r2,r4,f2w,g2w,0,0,0,0)

allDecisions' 2
  (B,1,1,jt2,0,jt4,joa,0,0,0,0,0,r2,r4,f2w,g2w,0,0,0,0)
  = [(2,(B3,N1D))] +!+ allDecisions' 3
  (B,1,1,jt2,0,jt4,joa,0,0,0,0,0,r2,r4,f2w,g2w,0,0,0,0)

allDecisions' 3
  (B,1,1,0,1,jt4,joa,0,0,0,0,0,0,0,0,0,0,0,0,0)
  = [(3,(B2,L1D))] +!+ allDecisions' 4
  (B,1,1,0,1,jt4,joa,0,0,0,0,0,0,0,0,0,0,0,0,0)
-- ...
allDecisions' i _ | i > 42 = []
allDecisions' i s = allDecisions' (i + 1) s

allDecisions :: Condition -> [(Int, Decision)]
allDecisions = allDecisions' 1
```

Listing 28.3 *allDecisions* function

The `+!+` operator states for lists concatenation with dropping duplicates. The *allDecisions* function checks all rules and generates the list of all possible decisions for a given input state. The list contains pairs — a decision rule number and the corresponding result.

28.3.1 Input States

To be useful, a generalised decision table should satisfy some qualitative properties such as completeness, consistency (determinism) and optimality. A decision table is considered to be *complete* if for any possible input situation at least one rule can produce a decision. A decision table is *deterministic* if no two different rules can produce different results for the same input situation. The last property means that any dependent rules were removed.

Let $\mathcal{A}$ denote a set of attributes selected to describe important features of the system under consideration, i.e., conditions and actions, $\mathcal{A} = \{A_1, A_2, ..., A_n\}$. For any attribute $A_i \in \mathcal{A}$, let D_i denote the domain (finite set of possible values) of A_i. It can be assumed that D_i contains at least two different elements. The set $\mathcal{A}$ is divided into two parts. $\mathcal{A}_c = \{A_{c_1}, A_{c_2}, \ldots, A_{c_k}\}$ will denote the set of *conditional attributes*, and $\mathcal{A}_d = \{A_{d_1}, A_{d_2}, \ldots, A_{d_m}\}$ will denote the set of *decision attributes*. For the sake of simplicity, it will be assumed that $\mathcal{A}_c$ and $\mathcal{A}_d$ are non-empty, finite, and ordered sets.

Let $\mathcal{R} = \{R_1, R_2, \ldots, R_l\}$ denote the set of all decision rules in the considered rule-based system. A formula for an attribute $A_i \in \mathcal{A}$ in a rule $R_j \in \mathcal{R}$ will be denoted by $R_j(A_i)$. To every attribute $A_i \in \mathcal{A}_c$, there will be attached a *variable* A_i that may take any value belonging to the domain D_i.

Let us focus on the decision table show in Table 28.3. We applied a few simplification to reduce the table size. First of all, we have omitted attributes variables, *e.g.* $R2(\mathrm{JT1}) = \mathrm{JT1} > 0$ but it is written as "> 0". Moreover, an empty cell in row i and column j means that $R_i(A_j) = A_j \in D_j$. In other words, the value of the attribute A_j is not important in the rule R_i (any value is accepted).

Besides attributes domains, a rule-based system domain $D \subseteq D_{c_1} \times D_{c_2} \times \cdots \times D_{c_k}$ can be considered. The domain represents all permissible combinations of values of condition attributes. To calculate the domain D, we remove from the set $D_{c_1} \times D_{c_2} \times \cdots \times D_{c_k}$ states excluded by the so-called *domain contradictions* that state in an explicit way that the particular combination of input values is impossible or not allowed. In the case of the considered decision table 27 contradictions follow directly from Table 28.2. For example, the $C1$ contradiction $\mathrm{B1} = 1 \wedge \mathrm{B2} = 1$ means that these attributes cannot take the value 1 at the same time. Other contradictions are presented in Table 28.4. All these contradictions follow from the relationships presented in Tables 28.1 and 28.2. Elements of the set D will be called *input states*.

To check a decision table properties, we have to generate the set D for the rule-based system under consideration. An argumentless function *states* is used for this purpose. The function code is presented in Listing 28.4. The function takes under consideration the attributes domains and contradictions, and generates a list of all admissible input states. The set D for the considered example contains 6.920 input states.

Table 28.4 Domain contradictions for the considered decision table (part 1)

	JT	TT	JT1	JT2	JT3	JT4	JOA	JOP	B1	B2	B3	B4	R2	R4	F2W	G2W	K1D	L1D	M1D	N1D
$C28$	B	2		0									1							1
$C29$	B	2		0											1					1
$C30$	B	2		0												1				1
$C31$	B	2		1																1
$C32$	B	2	1										1							
$C33$	B	2	1												1					
$C34$	B	2	1													1				
$C35$	B	1			1										1					1
$C36$	B	1			1											1				1
$C37$	B	1			1								1							1
$C38$	B	1		1	1	0								1						1
$C39$	B	1		1	1	1														1
$C40$	B								1											
$C41$	B									1										
$C42$	B										1									
$C43$	B											1								
$C44$	B		1		1										1					
$C45$	B		1		1											1				
$C46$	B		1		1								1							
$C47$	B		1	1	1									1						
$C48$	B		1	1	1	1														
$C49$	$\in${G,K,N}	2																		
$C50$	B	2	1	1																
$C51$	R	2														1				
$C52$	R	2								1										
$C53$	R	2		1																
$C54$	R												1							
$C55$	R													1						
$C56$	R											1								
$C57$	R																1			
$C58$	R																	1		
$C59$	R					1				1										
$C60$	R					1										1				
$C61$	R			1		1														
$C62$	K					0														
$C63$	L			0																
$C64$	M		0																	
$C65$	N				0															
$C66$	$\in${K,L}												1							
$C67$	$\in${K,L}													1						
$C68$	$\in${K,L,M,N}																1			
$C69$	$\in${K,L,M,N}																	1		
$C70$	$\in${K,L,M,N}																		1	
$C71$	$\in${K,L,M,N}								0											1
$C72$	N								1											
$C73$	$\in${K,L,M,N}							1												
$C74$	F			0																
$C75$	G					0														
$C76$	$\in${F,G}									1										
$C77$	$\in${F,G}											1								
$C78$	$\in${F,G}														1					
$C79$	$\in${F,G}															1				
$C80$	$\in${F,G}						1													
$C81$	G												1							

```
states :: [Condition]
states = [((jt,tt,jt1,jt2,jt3,jt4,joa,jop,b1,b2,
  b3,b4,r2,r4,f2w,g2w,k1d,l1d,m1d,n1d)) |
  jt <- [B,F,G,K,L,M,N,R],
  tt <- [1,2],
  jt1 <- [0,1],
  jt2 <- [0,1],
  -- ... another attributes domains
  n1d <- [0,1],
  not (b1 == 1 && b2 == 1),
  not (b1 == 1 && b3 == 1),
  -- ... another contradictions
  not ((jt == F || jt == G) && joa == 1),
  not (jt == G && r2 == 1)]
```

Listing 28.4 *states* function

28.3.2 Completeness Verification

Definition 28.1. An input state $\varphi \in D$ is said to *satisfy* the conditional part of a rule $R_j \in \mathcal{R}$ ($\varphi \cong R_j|\mathcal{A}_c$) iff each formula $R_j(A_i)$, where $A_i \in \mathcal{A}_c$, evaluates to *true*, for values of attributes compatible with φ.

Definition 28.2. A set of decision rules $\mathcal{R}$ is *complete* iff for any input state $\varphi \in D$ there exists a rule $R_i \in \mathcal{R}$ such that φ satisfies the conditional part of the rule R_i.

```
notCovered :: Condition -> Bool
notCovered s = allDecisions s == []

notCoveredStates :: [Condition]
notCoveredStates = filter notCovered states
```

Listing 28.5 Completeness verification algorithms

The result of the completeness analysis is a list of input states that are not covered by decision rules. To check whether an input state is covered, the *notCovered* function is used (see Listing 28.5). The function is used by the *notCoveredStates* function, which filters not covered states from the list generated by the *states* function. The result provided by the function for the considered decision table is the empty list. It means that the table is complete.

28.3.3 Consistency Verification

Definition 28.3. A set of decision rules $\mathcal{R}$ is *consistent* (deterministic) iff for any input state $\varphi \in D$, and any two rules $R_i, R_j \in \mathcal{R}$ if φ satisfies the conditional parts of the rules R_i and R_j, then both rules provide the same decision for the input state φ.

```
notDeterministic :: Condition -> Bool
notDeterministic s = length (removeDuplicates
  (secondElements (allDecisions s))) > 1

notDeterministicStates :: [Condition]
notDeterministicStates = filter notDeterministic states

notDeterministicStates' :: [(Condition, [(Int, Decision)])]
notDeterministicStates' = [(state, allDecisions state) |
  state <- (filter notDeterministic states)]
```

Listing 28.6 Consistency verification algorithms

A rule-based system is indeterministic, if for at least one input state, two different decision rules provide two different results. To check whether an input state is deterministic, the *notDeterministic* function is used (see Listing 28.6). The function *secondElements* takes the list of pairs provided by the *allDecisions* function and provides a list of decisions (rule numbers are omitted). Then, duplicates from the list are removed and its length is checked. The *notDeterministicStates* function filters inconsistent states from the list of all admissible states. The result provided by the function for the considered decision table is the empty list. It means that the table is consistent.

From practical point of view, the *notDeterministicStates'* function presented is more convenient. It provides not only the list of indeterministic states, but also decisions produced for these input states and numbers of fired rules. Such a list can be used to improve the decision table and to eliminate the indeterminism if any.

28.3.4 Optimality Verification

Definition 28.4. Let $\mathcal{R}$ be a complete and consistent set of decision rules. A rule $R_i \in \mathcal{R}$ is *independent* iff the set $\mathcal{R} - \{R_i\}$ is not complete. A rule $R_i \in \mathcal{R}$ is *dependent* iff the rule is not independent. The set $\mathcal{R}$ is *semi-optimal* iff any rule belonging to the set $\mathcal{R}$ is independent.

The optimality analysis allows designer to remove some rules that are not necessary to take decisions. First of all, we can check whether each decision rule can be fired (see the *possiblyFired* shown in Listing 28.7). The *concat* function transforms a list of lists into one list, while the *firstElements* generates the list of first elements from

```
possiblyFired :: [Int]
possiblyFired = sort (removeDuplicates (firstElements
  (concat [allDecisions state | state <- states])))

oneRuleTrigger :: Condition -> Bool
oneRuleTrigger s = length (allDecisions s) == 1

independentRules = sort (removeDuplicates (firstElements
  (concat [allDecisions state |
  state <- (filter oneRuleTrigger states)])))
```

Listing 28.7 Optimality verification algorithms

a list of pairs. A decision rule is independent, if for at least one input state, it is the only fired rule. The *oneRuleTrigger* function checks whether for a given input state exactly one rule can be fired. The *independentRules* function generates the list of all independent rules. For the considered decision table, the function returns a list that contains numbers of all decision rules. It means that all rules are independent.

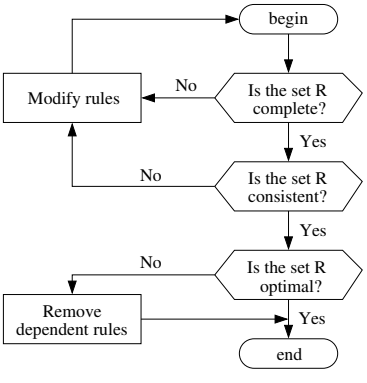


Fig. 28.2 Scheme block of the verification procedure

The semi-optimality should be verified after a set of rules is complete and consistent. The verification algorithm is presented in Fig. 28.2.

28.4 Alvis Modelling Language

Alvis is a successor of the XCCS modelling language [2], [8], which is an extension of the CCS process algebra [9], [1]. In contrast to process algebras, Alvis uses a high level programming language based on the Haskell syntax, instead of algebraic equations. Moreover, it combines hierarchical graphical modelling with high level programming language statements.

The key concept of Alvis is an *agent* that denotes any distinguished part of the system under consideration with a defined identity persisting in time. From the Alvis point of view, a system is seen as a set of agents that usually run concurrently, communicate one with another, compete for shared resources etc [17]. Each agent has assigned a set of ports used for a communication with other agents or, if embedded systems are considered, with the corresponding system environment. Ports are used both to collect data (*e.g.* sensors reading) and to provide results of the agent activity (*e.g.* control signals for external devices). The behaviour of an agent is defined with AlvisCL statements. If necessary, a rule-based system encoded in Haskell is used to make decisions. From the behaviour description points of view, agents are treated as independent individuals and defined components that can be used to compose a concurrent system. Communication diagrams are used to point out pairs of ports that make up communication channels used to exchange information between agents.

28.4.1 Code Layer

The *code layer* is used to define the behaviour of agents. Each agent is described with a piece of source code that may contain Alvis statements presented in Table 28.5. Moreover, Alvis uses the Haskell programming language [11] to define parameters, data types and data manipulation functions [15].

From the code layer point of view, agents are divided into *active* and *passive* ones. *Active agents* perform some activities and are similar to tasks in Ada programming language [3]. Each of them can be treated as a thread of control in a concurrent or distributed system. *Passive agents* do not perform any individual activity, and are similar to protected objects (shared variables). Passive agents provide mechanism for the mutual exclusion and data synchronisation. They provide a set of procedures that may be called by other agents.

28.4.2 Communication Diagrams

The *graphical layer* takes the form of a communication diagram [Szpyrla et al. (2011b)]. The layer is used to define interconnections (communication channels) among agents. A communication diagram is a hierarchical graph whose nodes may represent both kinds of agents (*active* or *passive*) and parts of the model from the lower level. The diagrams allow programmers to combine sets of agents into modules that are also represented as agents (called *hierarchical ones*). Active and hierarchical agents are drawn as rounded boxes, while passive ones as rectangles. Hierarchical agents are indicated by black triangles. Ports are drawn as circles placed at the edges of the corresponding rounded box or rectangle. Communication channels are drawn as lines (or broken lines). An arrowhead points out the input port for the particular connection. Communication channels without arrowheads represent pairs of connections with opposite directions. Elements of Alvis communication diagrams are shown in Fig. 28.3.

Table 28.5 Alvis statements

Statement	Description
`cli`	Turns off the interrupts handlers.
`critical {...}`	Define a set of statements that must be executed as a single one.
`delay ms`	Delays an agent execution for a given number of miliseconds.
`exec x = e`	Evaluates the expression and assign the result to the parameter; the *exec* keyword can be omitted.
`exit`	Terminates the agent that performs the statement.

`if (g1) {...}`
`elseif (g2) {...}`
`...`
`else {...}`
	Conditional statement.

`in p` `in p x`	Collects a signal via the port *p*. Collects a value via the port *p* and assigns it to the parameter *x*.
`jump label`	Transfers the control to the line of code identified with the *label*.
`jump far A`	Transfers the control to the agent *A*.

`loop (g) {...}`
`loop (every ms) {...}`
`loop {...}`
	Repeats execution of the contents while the guard if satisfied..
	Repeats execution every *ms* miliseconds.
	Infinite loop.

`null`	Empty statement.
`out p` `out p x`	Sends a signal via the port *p*. Sends a value of the parameter *x* via the port *p*; a literal value can be used instead of a parameter.
`proc (g) p {...}`	Defines the procedure for the port *p* of a passive agent (guard is optional).

`select {`
`  alt (g1) {...}`
`  alt (g2) {...}`
`  ... }`
	Selects one of the alternative choices.

`start A`	Starts the agent *A* if it is in the *Init* state, otherwise do nothing.
`sti`	Turns on the interrupts handlers.

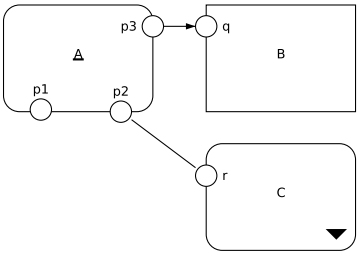


Fig. 28.3 Elements of Alvis communication diagrams

28.4.3 System Layer

The *system layer* depends on the model running environment, i.e. the hardware and/or operating system. The layer is necessary for a model simulation and verification. From the users point of view, the layer is the predefined one and it works in the read-only mode. Agents can retrieve some data from the layer, but they cannot directly change them. The system layer provides some functions that are useful for the implementation of scheduling algorithms or for retrieving information about other agents states. A user can choose one of a few versions of the layer and it affects the model semantic. System layers differ about the scheduling algorithm and system architecture mainly. The most universal one is the α^0 system layer. This layer makes Alvis a universal formal modelling language similar to Petri nets [14] or process algebras [4]. The α^0 system layer is based on the following assumptions.

- Each active agent has access to its own processor and performs its statements as soon as possible.
- The scheduler function is called after each statement automatically.
- In case of conflicts, agents priorities are taken under consideration. If two or more agents with the same highest priority compete for the same resources, the system works indeterministically.
  A *conflict* is a state when two or more active agents try to call a procedure of the same passive agent or two or more active agents try to communicate with the same active agent.

28.4.4 Communication in Alvis

Alvis uses two statements for the communication. The *in* statement for collecting data and *out* for sending. Each of them takes a port name as its first argument and optionally a parameter name as the second. A communication between two active agents can be initialised by any of them. The agent that initialises it performs the *out* statement to provide some information and waits for the second agent to take it, or performs the *in* statement to express its readiness to collect some information and waits until the second agent provides it.

A communication between an active and a passive agent can be initialised only by the former. Any procedure in Alvis uses only one either input or output parameter (or signal in case of parameterless communication). In case of an input procedure, an active agent calls the procedure using the *out* statement (and provides the parameter, if any, at the same time). In case of an output procedure, an active agent calls the procedure using the *in* statement and waits for the result. A procedure is finished after executing its last step.

Alvis agents may contain ports that are not used in any connection. Such ports are called *border ports* and are used for a communication with the considered system environment [16]. Border ports can be used both for collecting or sending some information to the embedded system environment. Properties of border ports are specified in the code layer preamble with the use of the *environment* statement. Each border port used as an input one is described with at least one *in* clause. Similarly, each border port used as an output one is described with at least one *out* clause. Each clause inside the *environment* statement contains the following pieces of information:

- *in* or *out* key word,
- the border port name,
- a type name or a list of permissible values to be sent through the port,
- a list of time points, when the port is accessible,
- optionally some modifiers: *durable, queue, signal*.

It should be underlined that only the *signal* modifier should be used in the final model of an embedded system. Other modifiers are defined mainly for the verification purposes, if reduced models are considered [16]. This modifier is used mainly for interrupt signals modelling.

28.4.5 Formal Verification

One of the main advantages of Alvis is a possibility of a formal model verification. States of an Alvis model and transitions among them are represented using a labelled transition system (LTS graph for short). An LTS graph is an ordered graph with nodes representing states of the considered system and edges representing transitions among states.

Due to practical reasons, such an LTS graph generated automatically for an Alvis model takes the textual form. Then, it is converted into the *Binary Coded Graphs* (BCG) format and used as input data for the CADP toolbox [6]. CADP offers a wide set of functionalities, ranging from step-by-step simulation to massively parallel model-checking.

28.5 Railway Traffic Management System – Case Study

The considered rule-based system is used as an element of an Alvis model of the railway traffic management system. Part of the communication diagram for the model is shown in Fig. 28.4.

```
-- ...
type Request = (JTDomain,Int);

environment {
  in jt1 [0,1] [] signal;
  in jt2 [0,1] [] signal;
  in jt3 [0,1] [] signal;
  in jt4 [0,1] [] signal;
  in joa [0,1] [] signal;
  in jop [0,1] [] signal;
  in request Request [] signal;
}

agent TC {
  jt1s :: Int = 0;
  jt2s :: Int = 0;
  -- ...
  n1ds :: Int = 0;
  requests :: [Requests] = [];
  route :: Decision = (INone, ONone);
  req :: Requests = (B, 1);

  select {
    alt (ready [in(jt1)]) { in jt1 jt1s; }
    alt (ready [in(jt2)]) { in jt2 jt2s; }
    alt (ready [in(jt3)]) { in jt3 jt3s; }
    alt (ready [in(jt4)]) { in jt4 jt4s; }
    alt (ready [in(joa)]) { in joa joas; }
    alt (ready [in(jop)]) { in jop jops; }
    alt (ready [in(request)]) {
      in request req;
      requests = requests ++ [req];
    }
    -- track unset if possible
    if(requests \= []) {
      req = head requests;
      route = rbs (fst req, snd req, jt1s, jt2s, jt3s, jt4s,
        joas, jops, b1s, b2s, b3s, b4s, r2s, r4s,
        f2ws, g2ws, k1ds, l1ds, m1ds, n1ds);
      if(route /= (INone, ONone)) {
        requests = tail requests;
        out set route;
      }
    }
    else {
      -- try to handle another request if any
    }
  }
}
```

Listing 28.8 Part of the code layer for the railway traffic management system

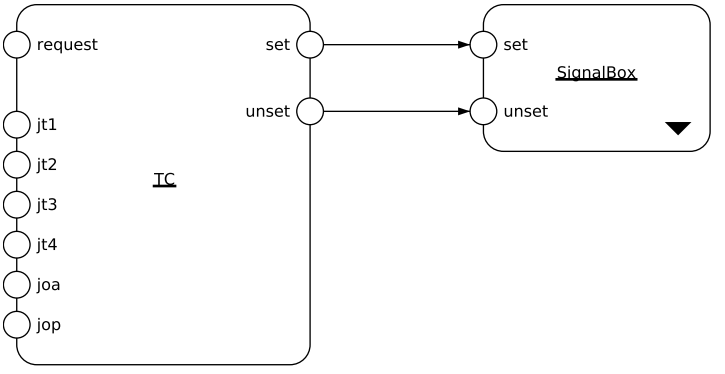


Fig. 28.4 Part of the communication diagrams for the railway traffic management system

The *SignalBox* agent is a hierarhical one. It represents the subsystem used for setting light signals and turnouts states. From this paper point of view, more interesting is the *TC* agent, which uses the decision table to make decisions. This is the main agent in the considered model. The most interesting parts of its implementation are presented in Listing 28.8. The agent uses seven border ports to collect information about the current state of the truck segments and about the requests sent from trains. The empty lists in the *environment* clauses mean that any of these signals may appear at any time. The *TC* agent contains parameters used to preserve information about:

- the current states of truck segments;
- currently set routes;
- requests waiting for handling.

After changing a value of at list one of these parameters, the *TC* agent try to handle one request from its requests queue. It calls the *rbs* function for this purpose.

28.6 Summary

A Haskell approach to rule-based systems implementation and verification has been presented in the paper. The approach has been worked out for the Alvis modelling language. Encoding of a decision table as a Haskell function allows the designer to include a rule-based system into the code layer of an Alvis model.

Moreover, it has been shown that Haskell can be useful from rule-based systems analysis point of view. Algorithms used to verify selected rule-based systems properties have been also presented in the paper. Their source code illustrates how short and intuitive can be Haskell. Thus, even a rule-based system is not implemented to be included into an Alvis model, Haskell can be used to verify its properties.

Acknowledgements. The paper is supported by the Alvis Project funded from 2009 to 2010 resources for science as a research project.

References

1. Aceto, L., Ingófsdóttir, A., Larsen, K., Srba, J.: Reactive Systems: Modelling, Specification and Verification. Cambridge University Press, Cambridge (2007)
2. Balicki, K., Szpyrka, M.: Formal definition of XCCS modelling language. Fundamenta Informaticae 93(1-3), 1–15 (2009)
3. Barnes, J.: Programming in Ada 2005. Addison-Wesley (2006)
4. Bergstra, J.A., Ponse, A., Smolka, S.A. (eds.): Handbook of Process Algebra. Elsevier Science, Upper Saddle River (2001)
5. Bratko, I.: Prolog Programming for Artificial Intelligence. Addison-Wesley (2000)
6. Garavel, H., Mateescu, R., Lang, F., Serwe, W.: CADP 2006: A Toolbox for the Construction and Analysis of Distributed Processes. In: Damm, W., Hermanns, H. (eds.) CAV 2007. LNCS, vol. 4590, pp. 158–163. Springer, Heidelberg (2007)
7. van Harmelen, F., Lifschitz, V., Porter, B. (eds.): Handbook of Knowledge Representation. Elsevier Science (2007)
8. Matyasik, P.: Design and analysis of embedded systems with XCCS process algebra. Ph.D. Thesis, AGH University of Science and Technology, Faculty of Electrical Engineering, Automatics, Computer Science and Electronics, Krakow, Poland (2009)
9. Milner, R.: Communication and Concurrency. Prentice-Hall (1989)
10. Nalepa, G.J.: Languages and tools for rule modeling. In: Giurca, A., Dragan Gasevic, K.T. (eds.) Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches, pp. 596–624. IGI Global, Hershey (2009)
11. O'Sullivan, B., Goerzen, J., Stewart, D.: Real World Haskell. O'Reilly Media, Sebastopol (2008)
12. Pawlak, Z.: Rough Sets. Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers (1991)
13. Szpyrka, M.: Design and analysis of rule-based systems with adder designer. In: Cotta, C., Reich, S., Schaefer, R., Ligęza, A. (eds.) Knowledge-Driven Computing: Knowledge Engineering and Intelligent Computations. SCI, vol. 102, pp. 255–271. Springer, Heidelberg (2008)
14. Szpyrka, M.: Petri nets for modelling and analysis of concurrent systems. WNT, Warsaw (2008) (in Polish)
15. Szpyrka, M.: Alvis On-line Manual. AGH University of Science and Technology (2011), `http://fm.ia.agh.edu.pl/alvis:manual`
16. Szpyrka, M., Kotulski, L., Matyasik, P.: Specification of embedded systems environment behaviour with Alvis modelling language. In: Proc. of the 2011 International Conference on Embedded Systems and Applications ESA 2011 (part of Worldcomp 2011), Las Vegas, Nevada, USA, pp. 79–85 (2011)
17. Szpyrka, M., Matyasik, P., Mrówka, R.: Alvis – modelling language for concurrent systems. In: Bouvry, P., González-Vélez, H., Kołodziej, J. (eds.) Intelligent Decision Systems in Large-Scale Distributed Environments. SCI, vol. 362, pp. 315–341. Springer, Heidelberg (2011)

Chapter 29
On Objective Measures of Actionability in Knowledge Discovery

Li-Shiang Tsay and Osman Gurdal

Abstract. One of the main goals of knowledge discovery is to find nuggets of useful knowledge that could influence or help users in a decision-making process. This task can be viewed as searching in an immense space for possible actionable concepts. Most of the KDD researchers believe that the task of finding actionable patterns is not easy and actionability is a purely subjective concept. Practitioners report that applying the KDD algorithms comprises not more than 20% of the knowledge discovery process and the remaining 80% relies on human experts to post-analyze the discovered patterns manually. To improve the effectiveness of the process, actionability can be defined as an objective measure via providing a well-defined strategy of pattern generations that allow guidance from domain experts at key stages in the search for useful patterns. The approach tightly integrates KDD and decision making by solving the decision-making problems directly on the core of KDD algorithms. In this paper, we present a granular computing-based method for generating a set of rules by utilizing the domain experts' prior knowledge to formulate its inputs and to evaluate the observed regularities it discovers. The generated rule overcomes the traditional data-centered pattern mining, resulting to bridge the gap and enhance real-world problem-solving capabilities.

Keywords: Actionable patterns, granular computing, reclassification model.

Li-Shiang Tsay
North Carolina A&T State University
Greensboro, NC USA 27411
e-mail: ltsay@ncat.edu

Osman Gurdal
Indiana University
Indianapolis, IN 46202
e-mail: ogurdal@iupui.edu

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 559–575.
springerlink.com

29.1 Introduction

The actionability in Knowledge Discovery (KD) refers to the discovered patterns that suggest workable, concrete, and profitable strategies to the decision-maker. Discovering actionable patterns is one of the greatest challenges in data mining [4, 7, 11, 20, 21]. Several approaches have been proposed in KDD literature to define various measures of actionable patterns with the goal of developing KDD tools that automatically discover only the patterns that will allow users to directly take action to gain advantage [17, 22, 23].

In the last decade, research in the field has made tremendous progress and KDD has greatly impacted the industry and society [8]. Nevertheless, there is a gap between the discovered patterns and the formulated solutions [6, 7]. Currently, this gap is filled by manual or semi-automatic analyses [6], which is time consuming, biased, and limits the efficiency of KDD's overall process and capabilities. Practitioners report that applying the KD algorithms comprises no more than 20% of the knowledge discovery process and the remaining 80% relies on human experts to post-analyze the discovered patterns manually [1]. The limitation of current tools is that they do not incorporate human user's knowledge and reuse the discovered knowledge. It is well known that purely empirical induction is infeasible to produce non-trivial results and some amount of the domain expert's knowledge must be considered at an early stage in the search for data patterns.

It is a remarkable fact that machine-learning algorithms incorporate very little knowledge of domain experts into its standard learning process and adequately obtain useful results in many domains [6]. To define good measures of actionability that would allow the KD system to discover the real workable plans is of vital importance. With such measures, the input of algorithms includes data as well as domain expert's knowledge and the output of algorithms is the expected benefit that could be realized by taking a specific action. Mined results are the potential solution to users and reduce the need of going through all of the discovered rules manually. The gap between data and action is reduced and the overall knowledge discovery process can be greatly improved.

In general, the evaluation of the actionability of discovered patterns has both an objective and subjective aspect. Objective measure is when the judgment is made by computer-derived methods or strategies, and subjective measure is when the judgment is made by people. Most researchers believe that the evaluation of the pattern's actionability is inherently subjective. There are several issues concerning subjective interestingness measure. First, users may not know or cannot precisely specify what interests them. Second, the users may not know enough about their domain. Third, the users may have difficulty to list all of their beliefs about the domain. Fourth, the result of analysis varies between individual users. Finally, it may require a lot of time to analyze the result in order to form a timely solution for making a better decision.

Conventionally, objective measure is defined as data-driven and domain-independent. It is mainly to remove irrelevant, spurious, and insignificant rules rather than to discover really interesting ones to human users. Because it does not

include domain knowledge and usually does not capture all the complexities of the pattern discovery process, the number of rules left after pruning can still be very large and most of the patterns left can still be considered redundant, obvious, and useless to users. That means, by itself alone, it is not sufficient in determining the actionability of extracted patterns; subjective concepts of actionability are also needed in order to define true actionability of patterns. Therefore, in this paper, we study the objective measure of actionability, but unlike others, we study them in a domain-dependent context.

29.2 Related Work

Actionability has long been identified as an important problem in the knowledge discovery area. There are two possible approaches in dealing with the actionability. One strategy is the employment of a post-analysis module at the back-end of the knowledge discovery system [12]. They do not use the domain experts' prior knowledge input to guide the rule generation process. The post-analysis module is purely subjective. Another strategy that is purely objective utilizes the users' input knowledge about the domain to guide the rule generation process and then discovers useful knowledge by comparing it with some forms of beliefs. Any mining results that will either support or contradict these beliefs are considered as interesting. This strategy can avoid generating uninteresting and useless rules, it reduces labor and time to evaluate the discovered patterns, and it improves the overall KD process.

We focus on objective approaches for actionability. Some recent work has specifically focused on the construction of workable plans by comparing the profiles of two sets of targeted objects - those that are desirable and those that are not. Each action plan was constructed from two groups, each with different desirable classes. It was assumed that the values of some attributes listed in both groups are unchangeable and had to be the same, *e.g.* for a customer database, the date of birth is an attribute that a decision maker cannot change. Other attributes, the ones that the decision maker does have influence on, are used to form workable strategies. Based on construction methods, the actionable patterns can be further divided into two types: rule-based and object-based.

Rule-based actionable patterns [14, 18, 19, 24, 25, 26, 28] are built on the foundations of previously discovered classification rules, so the quality and quantity of action rules is dependent on the adopted classification methods. Object-based actionable patterns [10, 27, 29], are built directly from the dataset and aimed to move the mined results to the final application, which does not rely on post-processing techniques and/or a great deal of a manual evaluation. In [10], authors explicitly formulated it as a search problem in a support-confidence-cost framework and next they presented an ID3-like algorithm for mining action rules. However, they did not take into account the dependencies between attributes which are especially important because the cost of rules decides as to whether to accept or reject a rule.

In [27, 29], authors presented an algorithm similar to Apriori [2] for constructing actionable rules directly from a decision table.

29.3 Actionable Rules

In this section, we introduce the theoretical background for actionability mining. First, the definition of an information system and its example are presented, Secondly, the definition of an object-based actionable rule jointly with the notions of their right support, left support, and confidence are given. Finally, the objectivity of object-based actionable rules will be further discussed in Section 29.4.

29.3.1 Information Systems

An information system $S = (U, A)$ is used for representing knowledge [16], where:

- U is a non-empty, finite set of objects,
- A is a non-empty, finite set of attributes, i.e. $a : U \to V_a$ is a function for any $a \in A$, where V_a is called the domain of a.

Elements of U are called objects. For the purpose of clarity, the objects can be viewed as customers, patients, students, etc. Attributes are interpreted as features such as, offers made by a bank, medical treatments for patients, characteristic conditions, etc. We only consider a special type of information systems called decision systems.

By a decision system we mean an information system $S = (U, A_S \cup A_F \cup \{d\})$, where $d \notin (A_S \cup A_F)$ is a distinguished attribute called the decision. Attributes in S are partitioned into stable conditions A_S and flexible conditions A_F. The number of elements in $d(U) = \{k : (\exists x \in U)[d(x) = k]\}$ is called the rank of d and it is denoted by $r(d)$. Clearly, the attribute d determines the partition $Part_S(d) =$

Table 29.1 Decision system Example

Objects	a	b	c	D
x_1	0	2	0	I
x_2	2	2	0	I
x_3	2	1	2	W
x_4	2	3	0	I
x_5	2	1	1	W
x_6	3	3	1	N
x_7	3	4	0	N

$\{X_1, X_2, \ldots, X_{r(d)}\}$ of objects in U into decision classes, where $X_k = d^{-1}(\{k\})$ for $1 \leq k \leq r(d)$.

Table 29.1 presents an example of a decision system, which consists of 7 objects described by 4 attributes. Attributes in $\{b, c\}$ are flexible, attribute a is stable, and d is the decision attribute.

29.3.2 Object-Based Action Rule, Left Support, Right Support, Confidence

Action rule mining constructs workable and useful strategies by comparing the profiles of two sets of targeted objects U: those that are desirable and those that are undesirable. This model goes beyond learning to predict likely outcomes, and learns to suggest preemptive actions that achieve the desired outcome. The discovered patterns provide an insight of how relationships should be managed so that the undesirable objects can be moved to a group of desirable ones. For example, in a school, one would like to not only identify students' learning barriers, but also to find a way to improve his or her academic performance. Another example can be found in the situation of modeling a mortgage bank. It is clearly helpful to predict which mortgagees are at high risk of failing to repay their loans; it is even more helpful if it would help the customer to learn which preventive actions might help to reduce their risk.

By an **object-based action rule** r in a decision system S, we mean an expression:

$$\begin{aligned} r = [[(a_1 = \omega_1) \wedge (a_2 = \omega_2) \wedge \ldots \\ \wedge (a_q = \omega_q)] \wedge (b_1, \alpha_1 \to \beta_1) \wedge (b_2, \alpha_2 \to \beta_2) \wedge \ldots \\ \wedge (b_p, \alpha_p \to \beta_p)] \Rightarrow [(d, k_1 \to k_2)], \end{aligned}$$

where $\{b_1, b_2, \ldots, b_p\}$ are flexible attributes and $\{a_1, a_2, \ldots, a_q\}$ are stable in S. In addition, we assume that $\omega_i \in Dom(a_i)$, $i = 1, 2, \ldots, q$ and $\alpha_i, \beta_i \in Dom(b_i)$, $i = 1, 2, \ldots, p$. The term $(a_i = \omega_i)$ states that the value of the attribute a_i is equal to ω_i, and $(b_j, \alpha_j \to \beta_j)$ means that value of the attribute b_j has been changed from α_j to β_j. We say that object $x \in U$ supports an object-based action rule r in S, if there is an object $y \in U$ such that:

$$\begin{aligned} &(\forall i \leq p)[[b_i(x) = \alpha_i] \wedge [b_i(y) = \beta_i]], \\ &(\forall i \leq q)[a_i(x) = a_i(y) = \omega_i], \\ &d(x) = k_1 \text{ and } d(y) = k_2. \end{aligned}$$

Action plans are constructed by comparing the profiles of two sets of targeted customers. So, we can assume that there are two patterns associated with each object-based action rule, a left-hand side pattern P_L and a right-hand side pattern P_R. There are three objective measures of rule interestingness including *Left Support, Right Support*, and *confidence*.

The *Left Support* defines the domain of an object-based action rule which identifies objects in U on which the rule can be applied. The larger its value is, the more interesting the rule will be for a user. The left-hand side pattern of an object-based action rule is defined as the set

$$P_L = V_L \cup \{k_1\}, \quad \text{where} \quad V_L = \{\omega_1, \omega_2, \ldots, \omega_q, \alpha_1, \alpha_2, \ldots, \alpha_p\}.$$

The domain $Dom_S(V_L)$ of the left pattern P_L is a set of objects in S that exactly match V_L. Card$[Dom_S(V_L)]$ is the number of objects in that domain. Card$[Dom_S(P_L)]$ is the number of objects in S that exactly match P_L and Card$[U]$ is the total number of objects in the decision system S. By the **left support** $supL$ of an object-based action rule r, we mean:

$$supL(r) = \text{Card}[Dom_S(P_L)]/\text{Card}[U].$$

The *RightSupport* shows how strongly the rule is supported by objects in S from the preferable decision class. The higher its value is, the stronger case of the reclassification effect will be. The pattern P_R of an object-based action rule r is defined as:

$$P_R = V_R \cup \{k_2\}, \quad \text{where } V_R = \{\omega_1, \omega_2, \ldots, \omega_q, \beta_1, \beta_2, \ldots, \beta_p\}.$$

By domain $Dom_S(V_R)$ we mean the set of objects in S matching V_R. Card$[Dom_S(P_R)]$ is the number of objects that exactly match P_R. By the **right support** $supR$ of action rule r, we mean:

$$supR(r) = \text{Card}[DomS(P_R)]/\text{Card}[U].$$

The *confidence* of rule r shows the success measure in transforming objects from a lower-preference decision class to a higher one. The *support* of an action rule r in S, denoted by $Sup_S(r)$, is the same as the left support $supL(r)$ of action rule r. This is the percentage of objects that need to be reclassified into more preferable class. By the **confidence** of an action rule r in S, denoted by $conf_S(r)$, we mean:

$$\begin{aligned} conf_S(r) =& (\text{Card}[Dom_S(P_L)]/\text{Card}[Dom_S(V_L)]) \cdot \\ & (\text{Card}[Dom_S(P_R)]/\text{Card}[Dom_S(V_R)]). \end{aligned}$$

29.4 Objectivity

The aim of this research is to look at the actionability in an objective way. Object-based action rules, as we already stated, provide a structure used to mathematically analyze a data set. Since mining object-based action rules do require information about domain knowledge in its initial state, we cannot get rid of some degree of subjectivity in determining which attributes should remain stable and how to take an action. Obviously, the partition of attributes into stable and flexible has to be done by users. This decision is a purely subjective one. A flexible attribute (occupation, interest rate, medical

treatment, etc.) means that its values can be changed and this change can be influenced and controlled by the users. A stable attribute (race, gender, etc.) is on the other end and its value cannot be changed. Stable attributes can also be used to define constraints (for example, "patient who does not smoke, can not start smoking") which objects in S have to satisfy in order to be processed by certain object-based action rules.

Basically, an object-based action rule shows that some selected objects can be reclassified from an undesired state to a desired one by changing some of the values of the corresponding flexible features. How to take action on those flexible attributes can be determined using either objective or subjective approach. It depends on the characteristic of the corresponding flexible attributes. If the attribute is an interest rate on the bank account, then banks can take action as the rule states (i.e., lower the mortgage rate to 4.75%). In this case, we have a purely objective step. However, if the attribute is a fever, then doctors have several options to follow to decrease the temperature. So, this will be a subjective step. Basically, object-based action rule mining cannot eliminate some amount of subjectivity in the process.

29.5 The Straightforward *StrategyGenerator* Approach

The *StrategyGenerator* algorithm is proposed to find the set of the most concise object-based action rules by considering a change of value within a single flexible attribute and each value of a stable attribute as atomic expressions from which more complex expressions are built. The algorithm follows a breath-first-type strategy and does not require prior extraction of classification rules. It guarantees that all discovered action rules are the shortest and the same, so they avoid unnecessary information.

There are two basic steps in the proposed approach.

(1) **Partition the decision table and select target sub-tables**:

The original decision table S is first partitioned into a number of sub-tables $S_1, S_2, \ldots, S_p$ according to the decision attribute in the decision table. Relevant sub-tables are selected based on the reclassification goal for forming workable strategies, most of the time only two sub-tables will be chosen. Usually, a practical data-mining application has a specific task, such as to find preventative actions to reduce the high-risk pregnancies, or those at high risk of requiring an emergency Cesarean-section delivery. Only some subsets of objects are important and need to be mined.

(2) **Form actionable plans**:

An object-based action rule is a structure of the form

$$[(A_S, \omega) \wedge (A_F, \alpha \rightarrow \beta)] \Rightarrow (d, \phi \rightarrow \psi),$$

where (A_S, ω) is a premise-type stable atom, $(A_F, \alpha \rightarrow \beta)$ is a premise-type flexible atom, and $(d, \phi \rightarrow \psi)$ is a decision-type atom.

First, a single candidate atom is formed for each attribute and a candidate atom is selected if its support meets a minimum criteria. The anti-monotonic property

is applied to filter candidate terms. When the support value of an atom is below the threshold, a negative mark is placed on the term and it is eliminated from the candidate list immediately.

Second, formation of an action rule assumes to concatenate a premise-type flexible atom with a decision-type atom. The rule is generated and a positive mark is placed on its premise-type atom, if its Left Support, Right Support, and confidence satisfy the thresholds $\alpha_1, \alpha_2, \alpha_3$, respectively. When an atom is marked with positive sign, it is removed from the candidate list as well. By doing this, we guarantee that the discovered rules are the most concise ones. At this step, stable atomic terms are not considered, because they cannot be solely used to construct action rules.

Third involves forming one element longer from the unmarked premise-type atoms. If its support is below the threshold value, then the negative mark is placed.

Fourth involves construction of the object-based action rule, which is a conjunction of newly-formed premise-type atoms with a decision-type atom, and then making the decision as to whether the rule can be accepted. The algorithm recursively takes unmarked candidate terms and extends them by one new unmarked atomic term till no new candidates are found.

The *StrategyGenerator* $SG(S, \alpha_1, \alpha_2, \alpha_3, R(r))$ algorithm for extracting object-based action rules from a complete decision system S is presented in Fig. 29.1.

```
SG( S, λ1, λ2, λ3, R(r) )
S = (X, A, V) – a complete decision system, where
      X is a set of objects, A is a set of attributes,
      and V is a set of attribute values.
      A is further divided into: A = (A_S ∪ A_F ∪ d)
                  A_S -- stable attribute
                  A_F -- flexible attribute (A_F)
                  d -- decision attribute
λ1 – threshold for a minimum Right Support
λ2 – threshold for a minimum Left Support
λ3 – threshold for a minimum confidence
R(r)– set of rules discovered from S by SG
Function main
   if S contains more than one decision value
      P = PartitionS()
      S_i = SelectTargetSub-tables(P)
      Display metadata of DS
      //Ask user to specify A_S, A_F, d, and
      //reclassification direction
      getInput ()
      //for Premise-type stable atom, flexible atom,
      // decision atom
      t:=GeneratedInitialTerms()
      Flag:=ture
      while(flag)
         if t.size > 0 then
            GenerateActionRules(t)
            GenerateCombinationTerms()
         else
            flag:=false
         end if-else
      end while
   end if
end main function
Function GenerateActionRules
   for ( i = 0; i < maxNumOfAtoms)
      r_i =conjunct premise atoms with decision atom
         and compute its supL, supR, and conf
      if (supL >= λ1)
         if (supR >= λ2)
            if (conf >= λ3 )
               R(r) = addRule( r_i )
               Remove atoms from the list
            end if
         else
            Remove right pattern's premise atoms
               from the list
         end if-else
      else
         Remove Left pattern's premise atoms from
         the list
   end if-else
end for loop
end function
```

Fig. 29.1 *StrategyGenerator* Algorithm

Now, referring back to Table 29.1, we illustrate the *StrategyGenerator* for the construction of object-based action rules step by step. We assume that W, I, and N denote the ranking of patients' health conditions as wellbeing, ill, and incurable status, respectively. The direction of reclassification is from I to W. The minimum support threshold for both *supR* and *supL* is 14%, and the minimum confidence threshold for rules is 75%.

Partition the decision table. In this example, the domain of the decision attribute is defined as $\{I, W, N\}$ and the reclassification direction is from I to W. That means the customers with decision value N are not the focus point in this case. Therefore, the decision table S can be divided into S_1 and S_2 according to the decision values I and W as represented in Fig. 29.2. Actionable strategies will be constructed based on sub-tables S_1 and S_2 only.

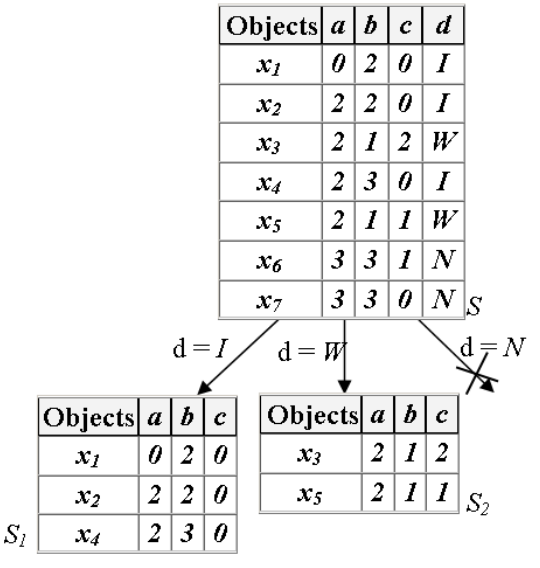


Fig. 29.2 Partition and Selection of Objects

Forming actionable strategies. The main idea of the reclassification goal is to move objects from an undesirable group into a more desirable one. Objects in S that have property P_L are denoted by L_S^* and objects in S that have property P_R are denoted by R_S^*. These two sets are also called granules. *StrategyGenerator* algorithm starts with atomic terms for S generated in its first loop. These terms are classified into two groups: premise-type and decision-type. Premise-type atomic terms are split into stable and flexible. As we mentioned before, any action rule without at least one flexible premise-type atomic term is meaningless. Stable atomic terms cannot be solely used to construct action rules, but they are important in boosting their confidence [25]. In this example, one valid candidate term which is a stable atom $(a, 2)$ is generated. In order to create atomic terms for a flexible attribute we check

its domain in both sub-tables. Referring back to Example 1, the values of attribute b are "2" and "3" in sub-table S_1 and "1" in sub-table S_2. It means that the action recommendations for attribute b say that its value should be changed from 2 to 1 or from 3 to 1. The corresponding atomic terms are presented as $(b,2 \rightarrow 1)$ and $(b,3 \rightarrow 1)$. Following the same procedure for attribute c, its corresponding atomic terms can be formed and they are listed below.

One-element term loop:
//Granules corresponding to values of a decision attribute
Decision-type atomic term: $(d,I \rightarrow W)$,
Granules: $L^* = \{x_1,x_2,x_4\}$, $\mathrm{R}^* = \{x_3,x_5\}$.
//Granules corresponding to values of condition attributes

Premise-type stable atomic expressions:
$(a,0)$, $L^* = \{x_1\}, \mathrm{R}^* = \emptyset$ Marked "-"
$(a,2)$, $L^* = \{x_2,x_4\}$, $\mathrm{R}^* = \{x_3,x_5\}$.

Premise-type flexible atomic expressions:
$(b,2 \rightarrow 1)$, $L^* = \{x_1,x_2\}, supL(r) = 2/7$; $R^* = \{x_3,x_5\}$,
$supR(r) = 2/7$; $Conf(r) = (2/2) \times (2/2) = 100\%$ Marked "+"

$(b,3 \rightarrow 1)$, $L^* = \{x_4\}$, $supL(r) = 1/7$; $R^* = \{x_3,x_5\}$,
$supR(r) = 2/7$; $Conf(r) = (1/3) \times (2/2) = 33\%$

$(c,0 \rightarrow 2)$, $L^* = \{x_1,x_2,x_4\}$, $supL(r) = 3/7$; $R^* = \{x_3\}$,
$supR(r) = 1/7$; $Conf(r) = (3/3) \times (1/1) = 100\%$ Marked "+"

$(c,0 \rightarrow 1)$, $L^* = \{x_1,x_2,x_4\}$, $supL(r) = 3/7$; $R^* = \{x_5\}$,
$supR(r) = 1/7$; $Conf(r) = (3/3) \times (1/2) = 50\%$.

The object-based action rule r linking each premise-type term and the decision-type term is acceptable when the values of the corresponding $supL(r), supR(r)$, and $Conf(r)$ meet the user-specified thresholds. The primary idea of the *StrategyGenerator* algorithm lies in the anti-monotonic property of the support. It is used to prune unqualified candidates. This is achieved by placing a "-" mark when a term does not have sufficient support. Going back to the example, the support of the atomic term $(a,0)$ does not satisfy the minimum support requirement, so it is marked with "-" symbol and it is not considered in later steps of the algorithm. The goal of this algorithm is to find the shortest actionable patterns. It means when a premise-type term t_1 jointly with a decision-type term form an acceptable action rule, then t_1 is not investigated any further.

In this example, the term $(b,2 \rightarrow 1)$ jointly with $(d,W \rightarrow I)$ meet all three thresholds, so the action rule $(b,2 \rightarrow 1) \Rightarrow (d,W \rightarrow I)$ is discovered and the term $(b,2 \rightarrow 1)$ is marked as "+".

Building two-element premise-type terms by concatenating any two unmarked premise-type terms that have different attributes. Below is the list of two-element terms. There are three action rules generated in this step.

Two-elements term loop:

$$(a,2) \wedge (b,3 \rightarrow 1),\ L^* = \{x_2\}, supL(r) = 1/7;\ R^* = \{x_3, x_5\},$$
$$supR(r) = 2/7;\ Conf(r) = (1/1) \times (2/2) = 100\%\ \text{Marked “+”}$$

$$(a,2) \wedge (c,0 \rightarrow 1),\ L^* = \{x_2, x_4\},\ supL(r) = 2/7;\ R^* = \{x_5\},$$
$$supR(r) = 1/7;\ Conf(r) = (2/2) \times (1/1) = 100\%\ \text{Marked “+”}$$

$$(b,3 \rightarrow 1) \wedge (c,0 \rightarrow 1),\ L^* = \{x_4\}, supL(r) = 1/7;\ R^* = \{x_5\},$$
$$supR(r) = 1/7;\ Conf(r) = (1/1) \times (1/1) = 100\%\ \text{Marked “+”}$$

Since all the premise-type atoms are marked, the algorithm is terminated and five object-based action rules are generated as follows:

$$(b,2 \rightarrow 1) \Rightarrow (d, I \rightarrow W),\ supL(r) = 2/7,\ supR(r) = 2/7,\ Conf(r) = 100\%$$

$$(c,0 \rightarrow 2) \Rightarrow (d, I \rightarrow W),\ supL(r) = 3/7,\ supR(r) = 1/7,\ Conf(r) = 75\%$$

$$((a,2) \wedge (b,3 \rightarrow 1)) \Rightarrow (d, I \rightarrow W),\ supL(r) = 1/7,\ supR(r) = 2/7,$$
$$Conf(r) = 100\%$$

$$((a,2) \wedge (c,0 \rightarrow 1)) \Rightarrow (d, I \rightarrow W),\ supL(r) = 2/7,\ supR(r) = 1/7,$$
$$Conf(r) = 100\%$$

$$((b,3 \rightarrow 1) \wedge (c,0 \rightarrow 1)) \Rightarrow (d, I \rightarrow W),\ supL(r) = 1/7,\ supR(r) = 1/7,$$
$$Conf(r) = 100\%.$$

The discovered rules presented to a decision maker should only consist of simple, understandable, and complete strategies that allow a reasonably easy identification

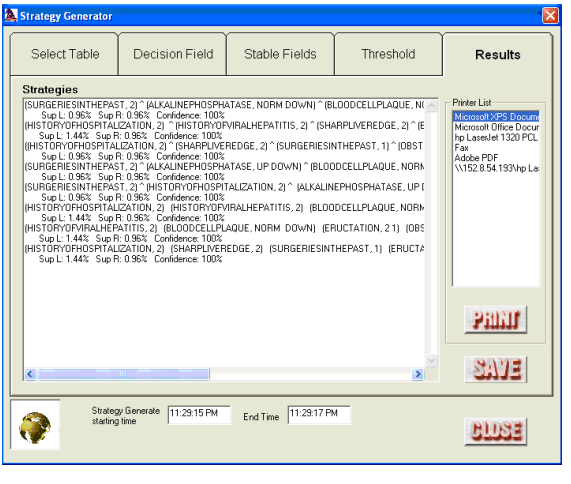


Fig. 29.3 *StrategyGenerator* Interface

of preferable rules. We claim that the new presented method guarantees that the actionable patterns are concise, general, and reliable. As we can see, the discovered action rules contain relatively few attribute-value pairs on the classification side and the number of these rules is also relatively small. Such rules are more readable, easier to understand and apply later on.

The algorithm, *StrategyGenerator*, was implemented in Windows XP (see Fig. 29.3).

It was tested on several public domain datasets and on the medical database HEPAR created in 1990, and thoroughly maintained in the Clinical Department of Gastroenterology and Metabolism at the Institute of Food and Feeding, Warsaw, Poland.

In all the cases, the recall of the new algorithm was higher than the recall of *DEAR* [18, 24, 25, 28].

29.5.1 Experiment I

Let us compare the object-based action rules generated by *StrategyGenerator* with action rules constructed by the tree-based algorithms *DEAR* [18, 24, 25, 28]. In the case of a decision table presented as Example 1 from the previous section, nine classification rules have been generated and they are listed below:

$$\begin{array}{lll} (a,0) \rightarrow (d,I); & (b,2) \rightarrow (d,I) & (c,0) \rightarrow (d,I) \\ (a,3) \rightarrow (d,N) & (b,1) \rightarrow (d,W) & (c,2) \rightarrow (d,W) \\ (a,2) \wedge (b,3) \rightarrow (d,I) & (b,3) \wedge (c,1) \rightarrow (d,N) & (a,2) \wedge (c,1) \rightarrow (d,W). \end{array}$$

For the reclassification direction from I to W, *DEAR* algorithm generates from them only 4 action rules listed below:

$$(b,2 \rightarrow 1) \Rightarrow (d,I \rightarrow W),\ supL(r) = 2/7,\ supR(r) = 2/7,\ Conf(r) = 100\%$$

$$(c,0 \rightarrow 2) \Rightarrow (d,I \rightarrow W),\ supL(r) = 3/7,\ supR(r) = 1/7,\ Conf(r) = 75\%$$

$$((a,2) \wedge (c,0 \rightarrow 1)) \Rightarrow (d,I \rightarrow W),\ supL(r) = 2/7,\ supR(r) = 1/7,\ Conf(r) = 100\%$$

$$(b,3 \rightarrow 1) \Rightarrow (d,I \rightarrow W),\ supL(r) = 1/7,\ supR(r) = 1/7,\ Conf(r) = 100\%.$$

The new method generates more action rules than *DEAR* as we have seen in the previous example. Because *DEAR* constructs actionable patterns from the classification rules, their quantity and quality is affected by the strategy chosen for classification rules extraction. *StrategyGenerator* is an object-based action rule mining while *DEAR* is a rule-based one. The comparison of these two approaches is presented in Fig. 29.4.

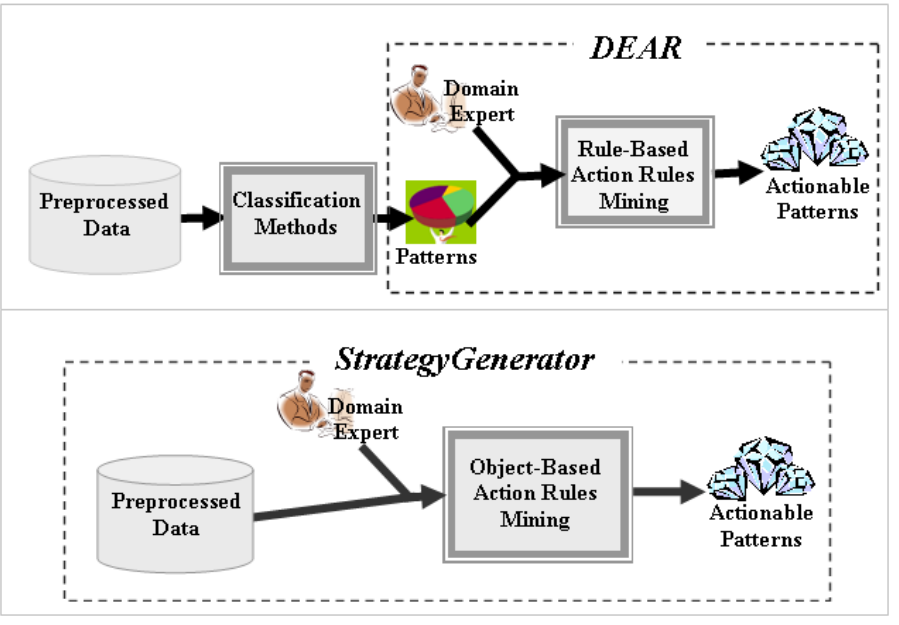


Fig. 29.4 *StrategyGenerator* v.s. *DEAR*

29.5.2 Experiment II - HEPAR Database

As the application domain for this research, we have chosen the HEPAR system built in collaboration between the Institute of Biocybernetics and Biomedical Engineering of the Polish Academy of Sciences and physicians at the Medical Center of Postgraduate.

Education in Warsaw, Poland [3, 15]. HEPAR was designed for gathering and processing clinical data about patients with liver disorders. It contains information on 758 patients described by 106 attributes (including 31 laboratory tests with values discretized to: "below normal", "normal", "above normal"). It has 14 stable attributes. Two laboratory tests are invasive: HBsAg (in tissue) and HBcAg (in tissue). The decision attribute has 7 values: I (acute hepatitis), IIa (subacute hepatitis [types B and C]), IIb (subacute hepatitis [alcohol-abuse]), IIIa (chronic hepatitis [curable]), IIIb (chronic hepatitis [non-curable]), IV (cirrhosis-hepatitis), V (liver-cancer).

The diagnosis of liver disease depends on a combination of the patient's history, physical examinations, laboratory tests, radiological tests, and frequently a liver biopsy. Blood tests play an important role in the diagnosis of liver diseases. However, their results should be analyzed along with the patient's history and physical examination. The most common radiological examinations used in the assessment of liver diseases are ultrasound and sonography. Ultrasound is a good test for

the detection of liver masses, assessment of bile ducts, and detection of gallstones present. However, it does not detect the degree of inflammation or fibrosis of the liver. Ultrasound is a non-invasive procedure and there are no risks associated with it. Liver biopsy enables the doctor to examine how much inflammation and how much scarring has occurred. Liver biopsy is an example of invasive procedure that carries certain risks to the patient. Therefore, despite the importance of its results to the diagnosis, clinicians try to avoid biopsy as often as possible. However, liver biopsy is often the only way to establish correct diagnosis in patients with chronic liver disorders.

A medical treatment is naturally associated with re-classification of patients from one decision class into another one. In this research we were mainly interested in the re-classification of patients from the class IIb into class I and from the class IIIa into class I, but without referring to any invasive test results in action rules.

Database HEPAR has many missing values. Following the approach proposed in [17], we removed all its attributes with more than 90% of null values assuming that these attributes are not related to invasive tests. Also, subjective attributes (like "history of alcohol abuse") have been removed. Next, we used one of the classical null value imputation techniques (provided in Rough Sets Exploration System (RSES)) to make the resulting database complete.

For testing purposes, we have chosen the same d-reduct R= $\{m,n,q,u,y,aa,ah,ai,am,an,aw,bb,bg,bm,by,cj,cm\}$ as in [17] because it does not contain any invasive tests. By d-reduct we mean a minimal subset of attributes which by itself can fully characterize the knowledge about attribute d in the database. The description of attributes (tests) listed in R is given below:

m - Bleeding	am - History of hospitalization (stable)
n - Subjaundice symptoms	ah - History of viral hepatitis (stable)
q - Eructation	bb - Cysts
u - Obstruction	bg - Sharp liver edge (stable)
y - Weight loss	bm - Blood cell plaque
aa - Smoking	by - Alkaline phosphatase
ai - Surgeries in the past (stable)	cj - Prothrombin index
an - Jaundice in pregnancy	cm - Total cholesterol
aw - Erythematous dermatitis	dd - Decision attribute

StrategyGenerator discovered eight action rules. Three of them have a very high confidence (close to 100) and they are given below:

$$[(am,2) \wedge (ah,2) \wedge (bg,2)] \wedge (q,2 \rightarrow 1) \wedge (cm,2 \rightarrow 1) \Rightarrow (dd, IIIA \rightarrow I).$$

The first rule is applicable to patients with a history of hospitalization, history of viral hepatitis, and with a sharp liver edge which is not normal. It says that if we get rid of eructation and decrease the cholesterol level to normal, then we should be able to reclassify such patients from the category IIIA to I.

$$[(am,2) \wedge (bg,2) \wedge (ai,1)] \wedge (u,2 \rightarrow 1) \wedge (y,2 \rightarrow 1) \Rightarrow (dd, IIIA \rightarrow I).$$

The second rule is applicable to patients with a history of hospitalization, with a sharp liver edge which is not normal, and with no surgeries in the past. It says that if we get rid of obstruction and change the weight loss level to normal, then we should be able to reclassify such patients from the category IIIA to I.

$$[(am,2) \wedge (bg,2) \wedge (ai,1)] \wedge (q,2 \rightarrow 1) \wedge (u,2 \rightarrow 1) \wedge (n,2 \rightarrow 1) \Rightarrow (dd, IIIA \rightarrow I).$$

The third rule is applicable to patients with a history of hospitalization, with a sharp liver edge which is not normal, and with no surgeries in the past. It says that if we get rid of eructation, get rid of obstruction, and change the subjaundice symptoms to normal, then we should be able to reclassify such patients from the category IIIA to I.

29.6 Conclusion

Knowledge Discovery and Data mining (KDD) has been widely used both in the government and the private sector for making better management decisions and improving services or performance. However, successful KDD still requires skilled technical and analytical specialists who can structure, analyze, and interpret the mined results. To address this issue, we proposed *StrategyGenerator* to discovery actionable patterns by utilizing the domain experts' prior knowledge to formulate its inputs and to evaluate the observed regularities. The mined results provide an insight of how relationships should be managed so the undesirable objects(patients) can be moved to a group of desirable ones. In our future work, we plan to investigate the notion of a cost associated with changes of values of an attribute. Our goal is to extract actionable patterns with the lowest cost to enlarge decision-support power in the real world. Related module will be added to the *StrategyGenerator* system.

References

1. Abramowicz, W., Zurada, J.M. (eds.): Knowledge Discovery for Business Information Systems. Kluwer, Dordrecht (2001)
2. Agrawal, R., Srikant, R.: Fast algorithm for mining association rules. In: Proceeding of the Twentieth International Conference on VLDB, pp. 487–499 (1994)
3. Bobrowski, L.: HEPAR: Computer system for diagnosis support and data analysis. In: Prace IBIB 31, Institute of Biocybernetics and Biomedical Engineering, Polish Academy of Sciences, Warsaw, Poland (1992)
4. Cao, L.: Domain-driven data mining: Challenges and prospects. IEEE Transactions on Knowledge and Data Engineering 22(6), 755–769 (2010)

5. Chmielewski, M.R., Grzymała-Busse, J., Peterson, N.W., Than, S.: The rule induction system LERS - a version for personal computers. Foundations of Computing and Decision Sciences 18(3-4), 181–121 (1993)
6. Domingos, P.: Toward knowledge-rich data mining. Data Mining Knowledge Discovery 15(1), 21–28 (2007)
7. Fayyad, U., Shapiro, G., Uthurusamy, R.: Summary from the KDD-03 panel Data Mining: The next 10 years. ACM SIGK Explorations Newsletter 5(2), 191–196 (2003)
8. Greco, S., Matarazzo, B., Pappalardo, N., Słowiński, R.: Measuring expected effects of interventions based on decision rules. Journal of Experimental and Theoretical Artificial Intelligence 17(1-2), 103–118 (2005)
9. He, Z., Xu, X., Deng, S., Ma, R.: Mining action rules from scratch. Expert Systems with Applications 29(3), 691–699 (2005)
10. Kriegel, H., Borgwardt, K., Kroger, P., Pryakhim, A., Schubert, M., Zimek, A.: Future trends in data mining. Data Mining and Knowledge Discovery 15(1), 87–97 (2007)
11. Liu, B., Hsu, W.: Post-analysis of learned rules. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI 1996), pp. 828–834. AAAI Press, Menlo Park (1996)
12. Major, J.A., Mangano, J.: Selecting among rules induced from a hurricane database. In: AAAI 1993 Workshop on Knowledge Discovery in Databases, pp. 24–44. AAAI Press, Menlo Park (1993)
13. Mitchell, T.: Machine learning and data mining. CACM 42(11), 31–36 (1999)
14. Onisko, A., Druzdzel, M., Wasyluk, H.: Extension of the HEPAR II model to multiple-disorder diagnosis. In: Intelligent Information Systems. ASC, pp. 303–313. Springer, Heidelberg (2000)
15. Pawlak, Z.: Information systems – theoretical foundations. Information Systems 6, 205–218 (1981)
16. Piatetsky-Shapiro, G., Matheus, C.J.: The interestingness of deviations. In: Proceedings of the AAAI 1994 Workshop on Knowledge Discovery in Databases, pp. 1–12 (1994)
17. Piatetsky-Shapiro, G.: Data Mining and Knowledge Discovery 1996 to 2005: Overcoming the hype and moving from "university" to "business" and "analystics". Data Mining and Knowledge Discovery 15(1), 99–105 (2007)
18. Raś, Z.W., Tsay, L.-S.: Discovering extended action-rules (System DEAR). In: Intelligent Information Systems, Proceedings of the IIS 2003 Symposium. ASC, pp. 293–300. Springer, Heidelberg (2003)
19. Raś, Z.W., Wieczorkowska, A.A.: Action-Rules: How to Increase Profit of a Company. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 587–592. Springer, Heidelberg (2000)
20. Shapiro, G.P., Djeraba, C., Getoor, L., Grossman, R., Feldman, R., Zaki, M.: What are the grant challenges for data mining? In: KDD-2006 Panel Report, ACM SIGKDD Explorations Newletter (2006)
21. Silberschatz, A., Tuzhilin, A.: On subjective measures of interestingness in knowledge discovery. In: Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD 1995), Montreal, Canada, August 20-21, pp. 275–281. AAAI Press, Menlo Park (1995)
22. Silberschatz, A., Tuzhilin, A.: What makes patterns interesting in knowledge discovery systems. IEEE Transactions on Knowledge and Data Engineering 8(6), 970–974 (1996)
23. Tsay, L.-S., Raś, Z.W.: Action rules discovery: system DEAR2, method and experiments. Journal of Experimental and Theoretical Artificial Intelligence 17(1-2), 119–128 (2005)
24. Tsay, L.-S., Raś, Z.W.: Action Rules Discovery System DEAR_3. In: Esposito, F., Raś, Z.W., Malerba, D., Semeraro, G. (eds.) ISMIS 2006. LNCS (LNAI), vol. 4203, pp. 483–492. Springer, Heidelberg (2006)
25. Tsay, L.-S., Raś, Z.W.: E-action rules. In: Lin, T.Y., et al. (eds.) Data Mining: Foundations and Practice. SCI, vol. 118, pp. 277–288. Springer, Heidelberg (2007)

26. Tsay, L.-S., Raś, Z.W.: Discovering the Concise Set of Actionable Patterns. In: An, A., Matwin, S., Raś, Z.W., Ślęzak, D. (eds.) Foundations of Intelligent Systems. LNCS (LNAI), vol. 4994, pp. 169–178. Springer, Heidelberg (2008)
27. Tsay, L.-S., Raś, Z., Wieczorkowska, A.: Tree-based algorithm for discovering extended action-rules (System DEAR2). In: Intelligent Information Processing and Web Mining, Proceedings of the IIS 2004 Symposium. ASC, pp. 459–464. Springer, Heidelberg (2004)
28. Tsay, L.S., Raś, Z.W., Im, S.: Reclassification Rules. In: Proceedings of 2008 IEEE International Conference on Data Mining Workshops, Pisa, Italy, pp. 619–627. IEEE Computer Society (2008)
29. Webb, G.I.: Editorial. Data Mining and Knowledge Discovery 15(1), 1–2 (2007)

Chapter 30
Pseudometric Spaces from Rough Sets Perspective

Piotr Wasilewski

Abstract. Pseudometric spaces are presented form the point of view of their connections with approximation spaces. A special way of determining equivalence relations by pseudometric spaces is considered and open sets in pseudometric spaces are studied. Investigations focus on the class of pseudometric spaces which are lower bounded in each point since open sets in these spaces coincide with definable sets of some prescribed approximation spaces. It is also shown that all equivalence and non transitive tolerance relations can be determined by pseudometric spaces in specified ways.

Keywords: Rough sets, pseudometric spaces, approximation spaces, information systems, indiscernibility relations, informational representability.

30.1 Introduction

We study pseudometric spaces from a perspective of their connections with approximation spaces from rough set theory [11–13] (see also *e.g.* [14–16]). A fundamental connection is based on the fact that every pseudometric space determines an equivalence relation identifying elements such that their distance with respect to a given pseudometric is equal to zero. We also show that every equivalence relation can be determined in that way. We call such relations atomizing, since they divides pseudometric spaces into atoms: a given pseudometric does not differentiate between elements of equivalence classes of its atomizing relation. This is closely related to the Pawlak's idea of information atoms [11–13]. We consider the form of open sets

Piotr Wasilewski
Institute of Mathematics,
University of Warsaw
Banacha 2, 02-097 Warsaw, Poland
e-mail: piotr@mimuw.edu.pl

A. Skowron and Z. Suraj (Eds.): Rough Sets and Intelligent Systems, ISRL 43, pp. 577–600.
springerlink.com

in pseudometric spaces and show that every open set in arbitrary pseudometric space is a union of equivalence classes of its atomizing relation, thus it is a definable set in an approximation space determined by this atomizing relation.

We study some properties of pseudometric spaces including a property of being lower bounded in each point characterizing pseudometric spaces where every atom is an open set. This implies that open sets in lower bounded in each point pseudometric spaces coincide with definable sets of approximation spaces determined by corresponding atomizing relations of these spaces. This makes this property interesting from the Rough Sets perspective applied in this paper so it is studied in sections 30.4–30.7. We characterize continuous functions between pseudometric spaces which are lower bounded in each point (Section 30.4). We discuss pseudometrics determined by families of sets, they are examples of pseudometric space which are lower bounded in each point and we show that any equivalence relation determines a pseudometric space such that it is atomized by this relation (Section 30.5). We show that clo-open topological spaces are exactly spaces pseudometrizable by pseudometric spaces which are lower bounded in each point (Section 30.6). Then we show that every lower bounded in each point pseudometric space is equivalent to double bounded pseudometric spaces determined by some partition of the space (Section 30.7). We finish our presentation by providing a topological characterization of attribute dependency. Then we generalizing this concept for information systems with different object domains and we provide a topological characterization of this generalized attribute dependency (Section 30.8).

Such study of pseudometric spaces is motivated by rough set theory and possible applications in analysis of incomplete information. However, in this paper we focus on theoretical aspects only, leaving discussion of applications for future work.

30.2 Rough Sets and Indiscernibility Relations

Rough sets were introduced by Zdzisław Pawlak as a tool for analyzing information systems – formal counterparts of information tables, where rows are labeled by names of objects and columns – by names of attributes. An *information system* is a triple $S = \langle Ob, At, Val_a \rangle$ where Ob is a set of objects, At is a set of attributes, and each Val_a is a value domain of an attribute $a \in At$, where $a : Ob \longrightarrow \mathcal{P}(Val_a)$ ($\mathcal{P}(Val_a)$ is a power set of Val_a). If $a(x) \neq \emptyset$ for all $x \in Ob$ and $a \in At$, then S is total. If $card(a(x)) = 1$ for every $x \in Ob$ and $a \in At$, then S is *deterministic*. Otherwise S is *indeterministic*. Referring to deterministic information systems we will use simply *information systems*.

According to Zdzisław Pawlak, knowledge is based on ability to discern objects by means of attributes [8, 10–13]. In information systems, this ability is presented by indiscernibility relation. This idea can lead also to introducing a notion of a distance based on distinguishability of objects where a distance between objects is, loosely speaking, equal to the number of attributes distinguishing them (Section 30.5). For analyzing indiscernibility relations determined by information

Table 30.1 Example of indeterministic information system.

	Height (cm)	Degrees		Height (cm)	Degrees
x_1	196	{BSc, MSc, PhD}	x_6	183	{BA, MA}
x_2	174	{BSc, BA, MA, PhD}	x_7	190	{BSc}
x_3	173	{BSc, MSc}	x_8	192	{BSc, MA, PhD}
x_4	179	{BA, MA, PhD}	x_9	187	{BA, MA}
x_5	178	{BSc}	x_{10}	184	{BSc, MSc, PhD}

systems Zdzisław Pawlak proposed approximation spaces and on their basis Pawlak introduced rough sets as tools for representing knowledge contained in information systems.

Let $S = \langle U, At, Val_a \rangle$ be an information system, $B \subseteq At$ and $x, y \in U$. *Indiscernibility relation* $ind(B)$ is a relation such that $(x,y) \in ind(B) \Leftrightarrow a(x) = a(y)$ for all $a \in B$. $ind(B)$ can be further analyzed from abstract perspective of approximation spaces [11, 12]. An *approximation space* is a pair (U,R) where U is a non-empty set and R is an equivalence relation on U. (the family of all equivalence relations on a set U we denote by $Eq(U)$). The equivalence classes of R are called *atoms* [12]. Subsets of U which are unions of atoms are called *definable* (or *composed*). Otherwise they are called *rough* [11–13]. $Def_R(U)$ denotes the family of all definable sets in (U,R). For (U,R) and $X \subseteq U$, *lower* and *upper approximations* of X in (U,R) are defined as follows

$$R_*(X) = \bigcup \{Y \in U_{/R} : Y \subseteq X\} \qquad R^*(X) = \bigcup \{Y \in U_{/R} : Y \cap X \neq \emptyset\}.$$

Approximation spaces can be investigated by means of set spaces (see [29–31] where they are called *general approximation spaces*). They appeared to be appropriate tools for general investigations into approximation spaces and indiscernibility relations and their connections with concept lattices [30, 31]. Set spaces can also serve as basic structures for in foundations of granular computing [28].

A pair $(U,\mathcal{C})$ is a *set space* if U is non-empty set and $\mathcal{C} \subseteq \mathcal{P}(U)$. For any $\mathcal{C} \subseteq \mathcal{P}(U)$, $Sg^c(\mathcal{C})$ denotes the least complete field of sets containing $\mathcal{C}$. Elements of $Sg^c(\mathcal{C})$ are called definable sets in the set space $(U,\mathcal{C})$. For any set space $(U,\mathcal{C})$ two operators can be defined:

$$\mathcal{C}_*(X) := \bigcup \{A \in Sg^c(\mathcal{C}) : A \subseteq X\} \qquad \mathcal{C}^*(X) := \bigcap \{A \in Sg^c(\mathcal{C}) : X \subseteq A\}.$$

Let $C \subseteq U$ then an *indiscernibility relation with respect to* C and *indiscernibility relation with respect to family* $\mathcal{C}$ are defined respectively as follows:

$$x \approx_C y \Leftrightarrow_{def} x \in C \Leftrightarrow y \in C \qquad (x,y) \in \approx_{\mathcal{C}} \Leftrightarrow_{def} (x,y) \in \bigcap_{C \in \mathcal{C}} \approx_C .$$

Observe that $\approx_C, \approx_{\mathcal{C}} \in Eq(U)$. Set spaces can represent information systems. Moreover, set spaces and information systems are mutually information representable

[2, 9] (with keeping the discernibility of objects, i.e. determining the same indiscernibility relation, for mutual informational representability of set spaces and information systems see [31]). For an information system $\langle Ob, At, \{V_a : a \in At\}, f\rangle$ there is a set space $(U, \mathcal{C})$ such that $ind(At) = \approx_{\mathcal{C}}$. To show this, for every $a \in AT$ and $v \in V_a$ one can define a set $C_a^v := \{x \in Ob : f(x,a) = v\}$ whereas a family $\mathcal{C}_{At}$ can be defined as follows $\mathcal{C}_{At} := \{C_a^v : a \in At, v \in V_a\}$. Thus $(OB, \mathcal{C}_{At})$ is a set space and now one can prove that $ind(At) = \approx_{\mathcal{C}_{At}}$. In the reverse direction, for any general approximation space $(U, \mathcal{C})$ there is an information system $\langle Ob, At, \{V_a : a \in At\}, f\rangle$ such that $\approx_{\mathcal{C}} = ind(At)$. To show this, for a set space $(U, \mathcal{C})$ one can consider an information system $\langle U, \mathcal{C}, \{V_C : C \in \mathcal{C}\}, f\rangle$, where $V_C = \{0, 1\}$ for each $C \in \mathcal{C}$ and a function f assigns each pair (a, C) the value of the characteristic function χ_C for the object a and prove that $\approx_{\mathcal{C}} = ind(\mathcal{C})$.

It can be shown also that for any equivalence relation $R \in Eq(U)$ there is a family $\mathcal{C} \subseteq \mathcal{P}(U)$ such that $R = \approx_{\mathcal{C}}$. To show this one can prove that $R = \approx_{U/R}$. Moreover, approximation spaces can be adequately represented by set spaces since the following conditions are equivalent: $R = \approx_{\mathcal{C}}$ iff for any $X \subseteq U$ $R_*(X) = \mathcal{C}_*(X)$ and $R^*(X) = \mathcal{C}^*(X)$ iff $Def_R(U) = Sg^c(\mathcal{C})$ [30, 31]. If $\mathcal{A} \subseteq U$ is a field of sets on U, then $At(\mathcal{A})$ denotes the family of all atoms of $\mathcal{A}$. For any family $\mathcal{C} \subseteq \mathcal{P}(U)$ it can be shown that $At(Sg^c(\mathcal{C})) = U_{/\approx_{\mathcal{C}}}$, i.e. atoms of $Sg^c(\mathcal{C})$ are precisely equivalence classes of $\approx_{\mathcal{C}}$. Thus $B \in Sg^c(\mathcal{C})$ if and only if B is a union of classes from $U_{/\approx_{\mathcal{C}}}$ [30, 32]. Rough set theory allows us also to investigate dependency between attributes in information systems: for information system $S = \langle U, At, Val_a\rangle$ and $B, D \subseteq At$, *D depends on B* in S, symbolically $B \Rightarrow_S D$, iff $ind(B) \subseteq ind(D)$.

30.3 Pseudometric Spaces: Definition, Examples and Basic Properties

Let us recall now definition of pseudometric space [3]. $\mathbb{R}_+$ denotes the set of non-negative reals, i.e. $\mathbb{R}_+ := [0, +\infty)$, $\mathbb{R}_+^*$ denotes the set of positive reals, i.e. $\mathbb{R}_+^* := (0, +\infty)$. Let U be any nonempty set. A function $p : U^2 \to \mathbb{R}_+$ is *pseudometric* iff the following conditions hold:

(PM1) $\forall x \in U :\ p(x,x) = 0$
(PM2) $\forall x, y \in U :\ p(x,y) = p(y,x)$
(PM3) $\forall x, y, z \in U :\ p(x,z) \leq p(x,y) + p(y,z)$

A pair of the form (U, p) is called a *pseudometric space* while a function p is called a *pseudometric on the set U*. We refer to the condition (PM2) as the *symmetry condition* and to the condition (PM3) as the *triangle inequality condition*.

Let us note that every metric is also pseudometric, so pseudometric which is not a metric we will call *proper pseudometrics*. Now we present some examples of pseudometrics ($|\cdot|$ denotes the absolute value of a real number).

Example 30.1. Let a function $p : \mathbb{R}^2 \to \mathbb{R}_+$ be given by formula: for any $x, y \in \mathbb{R}$

$$p(x,y) := |x^2 - y^2|.$$

Let us note that a function p pseudometric on the set $\mathbb{R}$. Notice that $p(-2,2) = 0$ and since $2 \neq -2$, then p is a proper pseudometric on $\mathbb{R}$.

In order to give the next example, let us mention that $\mathbb{Z}$ denotes the set of integral numbers.

Example 30.2. Let $a \in \mathbb{Z}$ and $b \in \mathbb{N}$. Let $r(a,b)$ denote a reminder of division of a by b. Let $n \in \mathbb{N}$ and let a function $p_n : \mathbb{Z}^2 \to \mathbb{R}_+$ be given by formula: for any $a, b \in \mathbb{Z}$

$$p_n(a,b) := |r(a,n) - r(b,n)|.$$

Let us note that $p_n(a,a) = 0$ for every $n \in \mathbb{N}$ while symmetry and triangle inequality conditions follow from properties of the absolute value operation. Thus a pair $(\mathbb{Z}, p_n)$ is pseudometric space. Let us note also that if numbers a i b are multiples of a number n, then $p_n(a,b) = 0$. Let k and l be mutually different multiples of a number n, so $p_n(k,l) = 0$, but $k \neq l$. Thus a function p_n is proper pseudometric on the set $\mathbb{Z}$.

Let us observe also that $n = 1$ if and only if $p_n(a,b) = 0$ for all $a, b \in \mathbb{Z}$.

Notice that every pseudometric is determined uniquely by some equivalence relation. This is relation which identify elements such that a distance between them, according to a given pseudometric, is equal to zero. Equivalence classes of this relation consist of elements which are not distinguish by a given pseudometric. Thus these classes play role of the basic components of a pseudometric space. Particularly, open sets in a given pseudometric space do not separate elements from equivalence classes of such relation. So, this remark justify call such equivalence classes *atoms*. Let us introduce the next definition:

Definition 30.1. Let (U,p) be a pseudometric space and let $x, y \in U$. Then we define a relation $\sim_p$ putting:

$$x \sim_p y := p(x,y) = 0.$$

We say that a relation $\sim_p$ *atomize* the space (U,p) or more generally, since $\sim_p \in Eq(U)$, a relation $\theta \in Eq(U)$ (an approximation space (U,θ)) *atomizes a space* (U,p) iff $\theta = \sim_p$. Equivalence classes of $\sim_p$ are called *atoms of a pseudometric space* (U,p), or shortly *atoms*.

Above definition directly entails the following corollary:

Corollary 30.1. *If a pseudometric space (U,p) is a metric space, then (U,p) is atomized by the identity relation on the space U, i.e. $\sim_p = \Delta_U$.*

Thus, for example, the relation atomizing of every metric space on $\mathbb{R}^n$ is the identity relation on the set $\mathbb{R}^n$. Here are next examples of atomizing relations:

Example 30.3. Let $(\mathbb{R}, p)$ be a pseudometric space from the example 30.1. Let us note that the relation atomizing this space has the following form:

$$\sim_p = \{(x,y) \in \mathbb{R}^2 : \ |x| = |y|\}.$$

Example 30.4. Let $(\mathbb{Z}, p_n)$, for $n \in \mathbb{N}$, be a pseudometric space from the example 30.2. Let $\equiv_n$ denotes a relation of congruence modulo n between integral numbers, i.e.

$$\equiv_n := \ \{(a,b) \in \mathbb{Z}^2 : r(a,n) = r(b,n)\}.$$

This relation is compatible with operation from the field of integral numbers or using universal algebraic terminology, it is a congruence in the field of integral numbers. Let us note that $\equiv_n = \sim_{p_n}$, that is for any $a,b \in \mathbb{Z}$, $p_n(a,b) = 0 \Leftrightarrow a$ and b are multiples of n and a congruence $\equiv_n$ atomizes the space $(\mathbb{Z}, p_n)$.

Definition 30.2. Let U be any non-empty set and $\theta \in Eq(U)$. We say that a function $p : U^2 \to \mathbb{R}_+$ and a relation θ *satisfy generalized determinacy condition* (GM), when for any $x,y \in U$:

$$p(x,y) = 0 \Leftrightarrow (x,y) \in \theta.$$

Proposition 30.1. *Let U be any set and let $p : U^2 \to \mathbb{R}_+$ be a function. p is pseudometric on U if and only if there is a relation $\theta \in Eq(U)$ such that the following conditions are satisfied:*

(GM) $p(x,y) = 0 \Leftrightarrow (x,y) \in \theta$
(PM2) $\forall x,y \in U : \ p(x,y) = p(y,x)$
(PM3) $\forall x,y,z \in U : \ p(x,z) \leq p(x,y) + p(y,z)$

In other words p is pseudometric if and only if it satisfies generalized determinacy condition with some equivalence relation θ and it satisfies symmetry and triangle inequality conditions. If (U,d) is pseudometric space, then the relation satisfying with metric d the generalized determinacy condition is Δ_U, i.e. the identity relation on the set U.

Proposition 30.1 shows that the concept of a pseudometric space is a natural generalization of the concept of a metric space. Thus for the class of pseudometric space one can naturally generalize the concepts defined for metric spaces such as the concepts of *open ball*, *open set*, *closed set*, *continuous function*.

Definition 30.3. Let (U,p) be a pseudometric space and let $x_0 \in U$, $r \in \mathbb{R}^*_+$.

The set $K(x_0,r)$ of the form:

$$K(x_0,r) = \{x \in U : p(x_0,x) < r\}$$

is called the *open ball in the space (U,p) with a centre x_0 and radius r* or simply the *r-ball about x_0*.

A set $A \subseteq U$ is called *open set in the space (U,p)* iff for any $x \in A$ there is $r \in \mathbb{R}^*_+$, such that $K(x,r) \subseteq A$.

A set $A \subseteq U$ is called *closed set in the space (U,p)* iff it is a complement of an open set in the space (U,p), i.e. there is an open set $B \subseteq U$ such that $A = U \setminus B$. If

it will not lead to misunderstanding, we will say shortly open balls, open sets and closed sets.

An open set $A \subseteq U$ in the space (U,p) is called the *neighborhood of the point* $x_0 \in U$ *in the space* (U,p), shortly the *neighborhood of the* x_0 if and only if $x_0 \in A$. $top(U,p)$ denotes the family of all open sets in the space (U,p), i.e.

$$top(U,p) = \{A \subseteq U : A \text{ is open in the space } (U,p)\}.$$

Above definition directly entails the following proposition, which is a generalization of the well known fact from the theory of metric spaces for the class of pseudometric spaces:

Corollary 30.2. *Let* (U,p) *be a pseudometric space. Then*

(1) *A set* $A \subseteq U$ *is open in* (U,p) *if and only if A is a union of open balls contained in it.*
(2) *Every open ball in* (U,p) *is an open set in* (U,p).

Proposition 30.2. *Let* (U,p) *be a pseudometric space and let* $K(x,r)$ *be open ball in* (U,p). *Then:*

(1) $K(x,r)$ *is a union of atoms of the space* (U,p).
(2) *For any* $y \in U$ *the following implication holds:*

$$y_{/\sim_p} \cap K(x,r) \neq \emptyset \Rightarrow y_{/\sim_p} \subseteq K(x,r).$$

(3) *If an atom* $x_{/\sim_p}$ *is an open set in the space* (U,p), *then there is a number* $r \in \mathbb{R}^*_+$ *such that* $x_{/\sim_p} = K(x,r)$.

In other words every open ball in arbitrary pseudometric space is a union of its atoms.

Proof. (3.2.1) Let (U,p) be a pseudometric space while $K(a,r)$ an open ball in space (U,p). Note that for arbitrary $x,y \in U$ and $r \in \mathbb{R}^*_+$ the following holds $x_{/\sim_p} \subseteq K(x,r)$ and if $y \in K(x,r)$, then $y_{/\sim_p} \subseteq K(x,r)$, thus:

$$\bigcup_{x \in K(a,r)} x_{/\sim_p} = K(a,r).$$

Therefore every open ball in space (U,p) is a union of atoms of (U,p).

(3.2.2) Let $y \in U$. Assume that $y_{/\sim_p} \cap K(x,r) \neq \emptyset$, thus there is $z \in U$ such that $z \in y_{/\sim_p} \cap K(x,r)$, then $p(z,y) = 0$ and $p(x,z) < r$. From the triangle inequality condition we know that $p(x,y) \leqslant p(x,z) + p(z,y) = p(x,z) + < r$, thus $p(x,y) < r$, and so $y \in K(x,r)$, therefore $y_{/\sim_p} \subseteq K(x,r)$.

(3.2.3) Let atom $x_{/\sim_p}$ be an open set in space (U,p), then there is radius r, such that $K(x,r) \subseteq x_{/\sim_p}$. Point 2 of this proposition implies that for every ball of the form $K(x,l)$ it holds that $x_{/\sim_p} \subseteq K(x,l)$, therefore $x_{/\sim_p} = K(x,r)$.

Notice that the reverse proposition to proposition 30.2 does not hold, i.e. it is not true that every union of atoms in any pseudometric space is an open ball in that space. It is shown by the following counter examples:

Example 30.5. Let $(\mathbb{Z}, p_5)$ be a pseudometric space from the example 30.2. Let us note that reminders of division of numbers 6,7,8,9 by 5 are respectively $r(6,5) = 1$, $r(7,5) = 2$, $r(8,5) = 3$, $r(9,5) = 4$. Thus equivalence classes $6_{/\equiv_5}$, $7_{/\equiv_5}$, $8_{/\equiv_5}$, $9_{/\equiv_5}$ are mutually different. Let us note also that $p_5(7,8) = 1$ i $p_5(8,9) = 1$, so $p_5(7,8) = p_5(8,9)$. We will show that a set $7_{/\equiv_5} \cup 8_{/\equiv_5}$ can not be an open ball in the pseudometric space $(\mathbb{Z}, p_5)$. For non-direct proof assume contrary that $7_{/\equiv_5} \cup 8_{/\equiv_5} = K(a,r)$, where $K(a,r)$ is an open ball in the pseudometric space $(\mathbb{Z}, p_5)$. So $a \in 7_{/\equiv_5} \cup 8_{/\equiv_5}$.

Assume that $a \in 7_{/\equiv_5}$. Since $8 \in K(a,r)$, then $p_5(a,8) < r$. Because $p_5(a,7) = 0$, so $p_5(7,8) \leqslant p_5(7,a) + p_5(a,8) = 0 + p_5(a,8) = p_5(a,8) < r$, thus $p_5(7,8) < r$. Let us note that $p_5(7,6) = 1 = p_5(7,8)$, and that $p_5(a,6) \leqslant p_5(a,7) + p_5(7,6) = 0 + p_5(7,8) = p_5(7,8) < r$ so $p_5(a,6) < r$. Thus $6 \in K(a,r)$ and so $6_{/\equiv_5} \subseteq K(a,r)$. Since equivalence classes $6_{/\equiv_5}$, $7_{/\equiv_5}$, $8_{/\equiv_5}$ are mutually different, then $6_{/\equiv_5} \nsubseteq 7_{/\equiv_5} \cup 8_{/\equiv_5}$ and $6_{/\equiv_5} \cup 7_{/\equiv_5} \cup 8_{/\equiv_5} \neq 7_{/\equiv_5} \cup 8_{/\equiv_5}$. So, $K(a,r) = 6_{/\equiv_5} \cup 7_{/\equiv_5} \cup 8_{/\equiv_5}$, which contradicts assumption that $K(a,r) = 7_{/\equiv_5} \cup 8_{/\equiv_5}$.

Similarly one can show a contradiction assuming that $a \in 8_{/\equiv_5}$. Thus the set $7_{/\equiv_5} \cup 8_{/\equiv_5}$ can not be an open ball in the pseudometric space $(\mathbb{Z}, p_5)$.

Example 30.6. Let $(\mathbb{R}, p)$ be a pseudometric space form the example 30.1. Let us choose from $\mathbb{R}_+$ a decreasing sequence (x_n) which is convergent to 0. We have noted in the example 30.3 that

$$\sim_p = \{(x,y) \in \mathbb{R}^2 : \ |x| = |y|\},$$

so $0_{/\sim_p} = \{0\}$. Thus one can not find $r \in \mathbb{R}_+$ such that $K(0,r) = 0_{/\sim_p}$, since for each choice of $r \in \mathbb{R}_+$ the ball $K(0,r)$ contains infinitely many elements of the sequence (x_n) such that their equivalence classes modulo $\sim_p$ are mutually different.

Proposition 30.3. *Every open set in arbitrary pseudometric space is a union of atoms of that space.*

Proof. It is enough to note that:

$$A = \bigcup_{x \in A} x_{/\sim_p},$$

where (U,p) is a pseudometric space and $A \subseteq U$ is an open set in (U,p).

Notice that the converse proposition to the above proposition does not hold, i.e. not every union of atoms of arbitrary pseudometric space is an open set in that space. Consider as a counterexample arbitrary metric space which is not discrete. As counterexamples in the class of proper pseudometric spaces one can take into account pseudometric spaces from the earlier examples 30.5 i 30.6. Notice also that one can find pseudometric spaces such that every union of its atoms is an open set

in that spaces. As an example of such spaces can serve the pseudometric space from the example 30.2:

Example 30.7. Let $(\mathbb{Z}, p_n)$ be a pseudometric space from the example 3.3 for some $n \in \mathbb{N}$. Let us note that a function p_n takes its values in the set of natural numbers, so a distance between any two objects form considered space is either equal to 0 or is greater or equal to 1. Thus for arbitrary number $a \in \mathbb{Z}$ and for a radius $r < 1$ holds $K(a,r) = a_{/\sim_{p_n}}$. Since every atom is an open ball, then every union of atoms of the space $(\mathbb{Z}, p_n)$ is an open set in $(\mathbb{Z}, p_n)$.

Corollary 30.3. *Let (U,p) be pseudometric space such that every union of its atoms is an open set, then*

$$top(U,p) = Sg^c(U_{/\sim_p}).$$

In other words all open sets in the space (U,p) create a complete field of sets which is completely generated by the atoms of (U,p).

Proof. Given proposition 30.3 it is enough to show inclusion $(\supseteq)$. From theorem on generalized normal form we know that field of sets $Sg^c(U_{/\sim_p})$ consist of exactly unions of g-components over family $U_{/\sim_p}$, whereas for fakt 1.14 we know that $\prod^*_{U_{/\sim_p}} = U_{/\approx_{U_{/\sim_p}}}$, therefore field $Sg^c(U_{/\sim_p})$ consist of exactly unions of elements from family $U_{/\approx_{U_{/\sim_p}}}$, i.e. unions composed of blocks of the partition determined by indiscernibility relation with respect to partition $U_{/\sim_p}$. Since $\sim_p \in Eq(U)$, then since for every equivalence relation $R \in Eq(U)$ it holds that $\approx_{U_{/R}} = R$, we conclude that $\approx_{U_{/\sim_p}} = \sim_p$, thus from the Abstraction Principle we get $U_{/\approx_{U_{/\sim_p}}} = U_{/\sim_p}$. Therefore field of sets $Sg^c(U_{/\sim_p})$ consist of exactly sets being unions of elements of $U_{/\sim_p}$, and so unions of atoms of space (U,p). Since (U,p) is a pseudometric space in which every union of atoms is an open set, then $Sg^c(U_{/\sim_p}) \subseteq top(U,p)$.

In the theory of metric spaces, the class of bounded spaces is distinguished. In such spaces there is the least upper bound of distances between its points called a diameter of a given space (see [3, 7]). For example the real plane with Euclidean metric is not a bounded space, but arbitrary circle with metric induced by Euclidean metric is a bounded metric space. A diameter of this space is ordinary diameter of this circle and this justify used name. Thus one can note that concepts of diameter and bounded space are some generalizations of concepts form a classical geometry. These concepts can be also defined for pseudometric spaces. Here we also consider concepts of *a lower diameter lower bonded spaces*.

Definition 30.4. Let (U,p) be a pseudometric space.

A space (U,p) is *upper bounded* iff there is a number $a_0 \in \mathbb{R}_+$ such that

$$a_0 = \sup\{p(x,y) : x,y \in U\}.$$

The number a_0 is called *the upper diameter* of the space (U,p).

A space (U,p) is *lower bounded* iff there is a number $a_0 \in \mathbb{R}_+$ such that

$$a_0 = \inf\{p(x,y) : x,y \in U \text{ and } p(x,y) \neq 0\}.$$

The number a_0 is called *the lower diameter* of the space (U,p).

If a space (U,p) is upper and lover bounded, then it is called *double bounded*.

Example 30.8. Let $(\mathbb{Z},p_n)$ be the pseudometric space from Example 30.2 for $n \in \mathbb{N}$ and $n \neq 1$. Notice that the space $(\mathbb{Z},p_n)$ is double bounded. Since $n \geqslant 2$, then the lower bound of the space $(\mathbb{Z},p_n)$ is 1, i.e.

$$1 = \inf\{p_n(x,y) : x,y \in U \text{ and } p_n(x,y) \neq 0\}.$$

Notice also that in the space $(\mathbb{Z},p_n)$ the greatest reminder from division by n is equal to $n-1$, and the least reminder from division by n is equal to 0. Thus $|(n-1)-0| = |n-1|$. Since $n \geqslant 2$, then $n-1 \geqslant 0$ and $|n-1| = n-1$. Thus $n-1$ is the upper diameter of the space $(\mathbb{Z},p_n)$.

Let us note also that the pseudometric space from Example 30.1 neither is upper nor lower bounded. The following proposition holds:

Proposition 30.4. *If a pseudometric space (U,p) is lower bounded, then every union of atoms of (U,p) is an open set in the space (U,p).*

Proof. Assume that pseudometric space (U,p) is lower bounded. Let $r_0 \in \mathbb{R}$ be the lower diameter of space (U,p). Therefore for every $x \in U$ it holds that $K(x,r_0) = x_{/\sim_p}$, then every atom is an open ball with the center in its arbitrary element and a radius r_0.

Let us recall a definition of a clo-open topology.

Definition 30.5. Let $(U,\mathcal{O})$ be a topological space. A family of closed and open sets in the space $(U,\mathcal{O})$, clo-open sets for short, is denoted by $Clop(U,\mathcal{O})$. A topology $\mathcal{O}$ on the set U is called *closed - open*, shortly *clo-open* iff $\mathcal{O} = Clop(U,\mathcal{O})$. Then a topological space $(U,\mathcal{O})$ is called *closed - open*, *clo-open* for short.

Notice that clo-open sets create field of sets (see *e.g.* [6]): if $(U,\mathcal{O})$ is a topological space, then $(Clop(U,\mathcal{O});\cap,\cup,',\emptyset,U)$ is a field of sets over U. One can show that a topological space $(U,\mathcal{O})$ is clo-open iff $\mathcal{O}$ is a complete field of sets over U.

For metric spaces it holds that the family of open sets of any metric space (U,d) is a topology on U. This fact can be generalized for the class of pseudometric spaces.

Proposition 30.5. *If (U,p) is a pseudometric space, then $top(U,p)$ is a topology on U.*

Let us note that it does not hold for every pseudometric space that every atom is an open set in that space. Moreover, in each pseudometric space an atom is an open ball if and only if it is an open set in that space. In order to characterize pseudometric spaces in which every atom is an open set we shall introduce a new concept.

Definition 30.6. Let (U,p) be a pseudometric space. A space (U,p) is *lower bounded in the point* $x \in U$ if and only if there is a number $r \in \mathbb{R}^*_+$, such that r is a lower bound of the set

$$\{p(x,y) : y \in U \text{ and } p(x,y) \neq 0\}$$

with respect to the natural order in the set of real numbers. Such number $r \in \mathbb{R}^*_+$ is called *the lower bound of the space* (U,p) *in the point* $x \in U$.

Definition 30.4 and above definition directly imply the following corollary:

Corollary 30.4. *If a pseudometric space* (U,p) *is lower bounded, then it is lower bounded in each its point (in each point* $x \in U$*).*

Theorem 30.1. *Let* (U,p) *be a pseudometric space and let* $A \subseteq U$*. The following conditions are equivalent:*

(1) *A pseudometric space* (U,p) *is lower bounded in each its point.*
(2) $U_{/\sim_p} \subseteq top(U,p)$*, i.e. every atom of the space* (U,p) *is an open set in* (U,p).
(3) *Every union of atoms of the space* (U,p) *is an open set in* (U,p).
(4) *A is an open set in the space* (U,p) *if and only if A is a union of atoms of the space* (U,p).
(5) *i)* $top(U,p)$ *is a clo-open topology on U and*
ii) $U_{/\sim_p} \subseteq top(U,p)$.
(6) *i)* $top(U,p)$ *is a complete field of sets and*
ii) $At(top(U,p)) = U_{/\sim_p}$*, i.e. equivalence classes of the relation* $\sim_p$ *are exactly atoms of* $top(U,p)$ *as a complete field of sets.*
(7) $top(U,p) = Sg^c(U_{/\sim_p})$*, i.e a topology* $top(U,p)$ *as a complete field of sets is completely generated by the family* $U_{/\sim_p}$.
(8) *The family of atoms of the pseudometric space* (U,p) *is a base of topology* $top(U,p)$.

Proof. We first prove equivalence $(1) \Leftrightarrow (2)$, and then the following implications: $(2) \Rightarrow (3) \Rightarrow (4) \Rightarrow (5) \Rightarrow (6) \Rightarrow (7) \Rightarrow (2)$.

Let (U,p) be a fixed pseudometric space.

$(1) \Rightarrow (2)$ Let us assume that pseudometric space (U,p) is lower bounded in every point. Let $B = x_{/\sim_p}$ for $x \in U$. Let r be an arbitrary lower bound of space (U,p) in point x. Since for every $y \in x_{/\sim_p}$, $p(x,y) = 0$, then $x_{/\sim_p} \subseteq K(x,r)$.

Let $y \in K(x,r)$, and so $p(x,y) < r$. Therefore $p(x,y) = 0$, because if $p(x,y) \neq 0$, then $r \leqslant p(x,y)$. Thus $y \in x_{/\sim_p}$, so $K(x,r) \subseteq x_{/\sim_p}$. Thus $K(x,r) = B$. Since $B \in U_{/\sim_p}$ was chosen arbitrary, then every atom of space (U,p) is an open ball, so is an open set in space (U,p).

$(2) \Rightarrow (1)$ Assume that every atom of pseudometric space (U,p) in an open set in this space. For arbitrary $x \in U$ atom $x_{/\sim_p}$ is then an open set in space (U,p). Therefore from proposition 30.2.3 we get that there is a radius r, such that $x_{/\sim_p} = K(x,r)$. Let us note that radius r is a lower bound of the set

$$\{p(x,y) : y \in U \text{ and } p(x,y) \neq 0\}.$$

Otherwise, if there was $z \in U$, such that $p(x,z) \neq 0$ and $p(x,z) \leqslant r$, then $z \in K(x,r)$, so $z \in x_{/\sim_p}$, thus $p(x,z) = 0$, which would give the contradiction. Therefore radius r is a lower bound of pseudometric space (U,p) in a point x. Since $x \in U$ was chosen arbitrarily, then we claim that space (U,p) is lower bounded in every point.

(2) $\Rightarrow$ (3) Assume that every atom of space (U,p) is an open set in (U,p), i.e. $U_{/\sim_p} \subseteq top(U,p)$. From proposition 30.5 we know that $top(U,p)$ is a topology on U. Thus every union of atoms of (U,p) is an open set in pseudometric space (U,p).

(3) $\Rightarrow$ (4) Let $A \subseteq U$. The implication: if A is a union of atoms of space (U,p), then A is an open set in space (U,p) follows from the point 3 of the above proposition whereas the converse implication follows from proposition 30.3.

(4) $\Rightarrow$ (5) From Proposition 30.5 we know that $top(U,p)$ is a topology on U, thus

$$Clop(top(U,p)) \subseteq top(U,p).$$

We have to show that $top(U,p) \subseteq Clop(top(U,p))$. Let $B \in top(U,p)$, thus B is an open set in space (U,p). Therefore for point 4 of the above proposition we know that B is a union of atoms of space (U,p). Note that B' - the complement of set B - is also an open set in space (U,p) (from the point 4 and the fact that the complement of a union of atoms of space (U,p) is also a union of atoms of (U,p) - it holds since the family of atoms of a pseudometric space (U,p) is a partition of that space), and so B is also a closed set. Thus $B \in Clop(top(U,p))$, therefore $top(U,p) = Clop(top(U,p))$.

It is enough to note that every atom is a union of atoms of a pseudometric space, thus form point 4 of this proposition we get that it is an open set, therefore $U_{/\sim_p} \subseteq top(U,p)$.

(5) $\Rightarrow$ (6) Since $top(U,p)$ is a topology on U, then we know that $Clop(top(U,p))$ is a union of sets. Since $top(U,p) = Clop(top(U,p))$, then $Clop(top(U,p))$ is a complete field of sets, so also $top(U,p)$ is also a complete field of sets, thus $top(U,p)$ is an atomic filed of sets.

We will show that $At(top(U,p)) = U_{/\sim_p}$. Let $B \in U_{/\sim_p}$. Let us recall a more general fact that every element of a complete filed of sets is a union of atoms of that sets. From the assumption we have that $U_{/\sim_p} \subseteq top(U,p)$, so B is a union of atoms of field $top(U,p)$. Because $U_{/\sim_p}$ and $At(top(U,p))$ are partitions of set U, so we can find set $A \in At(top(U,p))$, such that $A \cap B \neq \emptyset$. Thus there is $x \in U$, such that $x \in A \cap B$. Since A is an atom of field $top(U,p)$, then $A \subseteq B$. Because A is an open set, then we can find $r \in \mathbb{R}_+^*$, such that $K(x,r) \subseteq A$. Therefore $K(x,r) \cap B \neq \emptyset$, from this anf from proposition 30.2.2 we get that $B \subseteq K(x,r)$, thus $B \subseteq A$. Then we have shown that $A = B$ thus B jest is an atom of field $top(U,p)$. Because $B \in U_{/\sim_p}$ was chosen arbitrary, then $U_{/\sim_p} \subseteq At(top(U,p))$.

Let $A \in At(top(U,p))$. Since $At(top(U,p))$ i $U_{/\sim_p}$ are partitions of set U, then we can find $B \in U_{/\sim_p}$, such that $A \cap B \neq \emptyset$. Analogically, as above, one can show that $A = B$ and $A \in U_{/\sim_p}$, therefore $At(top(U,p)) \subseteq U_{/\sim_p}$.

$(6) \Rightarrow (7)$ Suppose that $top(U,p)$ is a complete field of sets and that $At(top(U,p)) = U_{/\sim_p}$. Thus $U_{/\sim_p} \subseteq top(U,p)$, therefore $Sg^c(U_{/\sim_p}) \subseteq top(U,p)$. Because $top(U,p)$ is a complete field of sets, then every element of $top(U,p)$ is a union of atoms of $top(U,p)$. Since $At(top(U,p)) = U_{/\sim_p}$, then every element of $top(U,p)$ is a union of elements form $U_{/\sim_p}$, so $top(U,p) \subseteq Sg^c(U_{/\sim_p})$. Therefore we have shown that $top(U,p) = Sg^c(U_{/\sim_p})$.

$(7) \Rightarrow (2)$ From assumption $top(U,p) = Sg^c(U_{/\sim_p})$ it follows directly that $U_{/\sim_p} \subseteq top(U,p)$, i.e. every atom of pseudometric space (U,p) is an open set in (U,p).

Applications of Theorem 30.1 in rough set theory and rough set interpretations of construction given in Theorem 30.1 are presented in Sections 30.5 and 30.6.

30.4 Continuity

Here we will consider an issue of a continuity in the pseudometric spaces. It should be clear that the concept of a continuous functions in the pseudometric spaces is a natural generalization of respective concept from metric spaces

Definition 30.7. Let (X,p), (Y,q) be pseudometric spaces. A function $f : X \longrightarrow Y$ is *continuous in a point* $a \in X$ if for every $B \subseteq Y$ - a neighborhood of $f(a)$ there is $A \subseteq X$ - a neighborhood of a such that $f(A) \subseteq B$. A function $f : X \longrightarrow Y$ is *continuous* if it is continuous in every point $a \in X$.

The next two propositions are natural generalizations of facts about continuous functions between metric spaces.

Proposition 30.6. *Let (X,p), (Y,q) be pseudometric spaces and let $f : X \longrightarrow Y$ be a function. Then the following conditions are equivalent:*

(1) *f is continuous in a point $a \in X$.*
(2) *For arbitrary $\varepsilon > 0$ it can be found $\delta > 0$ such that $f(K(a,\delta)) \subseteq K(f(a),\varepsilon)$.*
(3) *For arbitrary $\varepsilon > 0$ it can be found $\delta > 0$ such that for every $x \in X$ $p(a,x) < \delta \Rightarrow q(f(a),f(x)) < \varepsilon$.*

Proposition 30.7. *Let (X,p), (Y,q) be pseudometric spaces and let $f : X \longrightarrow Y$ be a function. Then the following conditions are equivalent:*

(1) *f is continuous function.*
(2) *For every set $B \subseteq Y$ which is open in the space (Y,q), a set $f^{-1}(B)$ is open in the space (X,p).*

We have shown that relations atomizing pseudometric spaces are connected with open sets in all pseudometric spaces and that they are closely related to open sets in pseudometric spaces lower bounded in each point. Now we characterize continuity of functions between pseudometric spaces lower bounded in each point by means of atomizing relations.

Theorem 30.2. *Let (X,p), (Y,q) pseudometric spaces lower bounded in each point and let $f : X \longrightarrow Y$ be a function. Then the following conditions are equivalent:*

(1) *f is continuous function.*
(2) *$x \sim_p y \Rightarrow f(x) \sim_q f(y)$ for all $x,y \in X$.*

Proof. Let (X,p) and (Y,q) be fixed pseudometric spaces lower bounded in every point.

($\Rightarrow$) Assume that a function $f : X \longrightarrow Y$ is continuous. Chose $x,y \in X$, such that $x \sim_p y$. Since space (Y,q) is lower bounded in every point, then from proposition 30.1.2 we know that its atoms, particularly $f(x)_{/\sim_q}$ and $f(x)_{/\sim_q}$, are open sets in this space. Thus from assumption about continuity of function f we conclude that $f^{-1}(f(x)_{/\sim_q})$ and $f^{-1}(f(y)_{/\sim_q})$ are open sets in space (X,p), thus also they are unions of atoms (from proposition 30.1.4). Therefore since $x \in f^{-1}(f(x)_{/\sim_q})$ and $y \in f^{-1}(f(y)_{/\sim_q})$, then

$$x_{/\sim_p} \subseteq f^{-1}(f(x)_{/\sim_q}) \; i \; y_{/\sim_p} \subseteq f^{-1}(f(y)_{/\sim_q}),$$

because $f^{-1}(f(x)_{/\sim_q})$ and $f^{-1}(f(y)_{/\sim_q})$ are unions of atoms of space (X,p). Because $x \sim_p y$ to $x_{/\sim_p} = y_{/\sim_p}$, thus

$$x,y \in f^{-1}(f(x)_{/\sim_q}) \cap f^{-1}(f(y)_{/\sim_q}).$$

Thus, *e.g.* $x \in f^{-1}(f(y)_{/\sim_q})$, therefore $f(x) \in f(y)_{/\sim_q}$, thus $f(x) \sim_q f(y)$. Thus we have shown that for all $x,y \in X$, if $x \sim_p y$, then $f(x) \sim_q f(y)$.

($\Leftarrow$) Assume that $f : X \longrightarrow Y$ is a function and that for all $x,y \in X$ the following implication holds $x \sim_p y \Rightarrow f(x) \sim_q f(y)$. We will show that a function f is continuous. Let $a \in X$. Since space (Y,q) is lower bounded in every point, then by proposition 30.1.2 atom $f(a)_{/\sim_q}$ is an open set in space (Y,q). Note also that $a \in f^{-1}(f(a)_{/\sim_q})$. We will show that set $f^{-1}(f(a)_{/\sim_q})$ is open in space (X,p). Let $x \in f^{-1}(f(a)_{/\sim_q})$, then $f(x) \in f(a)_{/\sim_q}$, thus $f(x) \sim_q f(a)$. Since space (X,p) is lower bounded in every point, then atom $x_{/\sim_p}$ is an open ball in space (X,p). Thus there is $r \in (0,+\infty)$ such that $K(x,r) = x_{/\sim_p}$. Since by the assumption we have that

$$x \sim_p y \Rightarrow f(x) \sim_q f(y),$$

thus if $z \in x_{/\sim_p}$, then $f(z) \sim_q f(x)$, and because $f(x) \sim_q f(a)$, then $f(z) \sim_q f(a)$, so $f(z) \in f(a)_{/\sim_q}$, thus $f(x_{/\sim_p}) \subseteq f(a)_{/\sim_q}$. Therefore $f^{-1}(f(x_{/\sim_p})) \subseteq f^{-1}(f(a)_{/\sim_q})$. Because for every set $A \subseteq X$ the following inclusion holds $A \subseteq f^{-1}(f(A))$, so $x_{/\sim_p} \subseteq$

$f^{-1}(f(x_{/\sim_p}))$, thus $x_{/\sim_p} \subseteq f^{-1}(f(a)_{/\sim_q})$, therefore $K(x,r) \subseteq f^{-1}(f(a)_{/\sim_q})$. Since point $x \in f^{-1}(f(a)_{/\sim_q})$ was chosen arbitrarily, then function f is continuous in point a, what because arbitrary choosing $a \in X$ implies that function f is continuous.

30.5 Pseudometrics Determined by Families of Sets

We have pointed out that every pseudometric space determines some approximation space: namely an approximation space such that its equivalence relation atomizes initial pseudometric space. So the natural question arises: whether every approximation space determines a pseudometric space which is atomized be an equivalence relation taken from that approximation space. In order to answer this question we define here some distance functions determined by some families of sets,

Definition 30.8. Let U be arbitrary non-empty set and let $C \subseteq U$. A function $d_C : U^2 \to \mathbb{R}_+$ is defined as follows:

$$d_C(x,y) =_{def} \begin{cases} 0 : x \approx_C y \\ 1 : x \not\approx_C y, \end{cases}$$

i.e. d_C is a characteristic function of the relation $\not\approx_C$.

We say that a function d_C is *determined by the set C* or that *the set C determines a function* d_C.

Let $\mathcal{C} = \{C_i\}_{i \in I} \subseteq \mathcal{P}(U)$. A function $d_{\mathcal{C}} : U^2 \to \mathbb{R}_+$ is defined as follows:

$$d_{\mathcal{C}}(x,y) := \sum_{i \in I} d_{C_i}(x,y).$$

We say that a function $d_{\mathcal{C}}$ is *determined by the family* $\mathcal{C}$ or that *the family* $\mathcal{C}$ *determines a function* $d_{\mathcal{C}}$.

Proposition 30.8. *Let U be arbitrary non-empty set and let $C \subseteq U$. Then the following conditions hold:*

(1) *A function d_C is a pseudometric on U.*
(2) $\sim_{d_C} = \approx_C$*, i.e a relation $\approx_C$ atomizes a pseudometric space (U, d_C).*

Let us note that not every family of sets determines a pseudometrics, i.e. it can be found such family $\mathcal{C}$ such that a function $d_{\mathcal{C}}$ is not a pseudometric, as it is shown by the following example:

Example 30.9. Let us choose from the set $\mathbb{R}$ two positive mutually different numbers r_1 i r_2. Let $\Gamma(r_1,r_2) := \{[0,a) : a \in (r_1,r_2) \cap \mathbb{Q}\}$. It is well known that between any two real numbers there are infinitely many rational numbers. Since the set $\mathbb{Q}$ is denumerable, then $\Gamma(r_1,r_2)$ is an infinite and denumerable family of sets. Let us note that for every $A \in \Gamma(r_1,r_2)$, $r_1 \in A$ and $r_2 \notin A$, so $d_{[0,a)}(r_1,r_2) = 1$ for all $a \in (r_1,r_2)$. Thus a series $\sum_{a \in (r_1,r_2)} d_{[0,a)}(r_1,r_2)$ is divergent, so a function d_Γ is not a pseudometric.

Of course a series $\sum_{i\in I} d_{C_i}$ for any finite family of sets $\{C_i\}_{i\in I}$ is convergent, thus the following corollary holds:

Corollary 30.5. *Let U be any non-empty set, let $\mathcal{C} \subseteq \mathcal{P}(U)$ and $|\mathcal{C}| < \aleph_0$. Then a function $d_{\mathcal{C}}$ is a pseudometric on U.*

Let us note also that a reverse implication to the above corollary 30.5 does not hold, i.e it is not true that if a function $d_{\mathcal{C}}$ is a pseudometric on U, then a family $\mathcal{C}$ is finite. It is shown b the next example:

Example 30.10. Let us choose from $\mathbb{R}$ the following family of sets $\{[0,n]\}_{n\in\mathbb{N}}$. Notice that $\{[0,n]\}_{n\in\mathbb{N}}$ is a denumerable increasing family of sets. Notice also that all negative real numbers are mutually indiscernible with respect to the family $\{[0,n]\}_{n\in\mathbb{N}}$. Let us note that any two positive and different real numbers x,y are discernible with respect to family $\{[0,n]\}_n \in \mathbb{N}$ if and only if there is a number $n \in \mathbb{N}$ such that $x < n < y$ or $y < n < x$. Let us also note that between any two positive different real numbers there are only finitely many natural numbers. Thus for arbitrary mutually different real numbers x,y the series $\sum_{n\in\mathbb{N}} d_{[0,n]}(x,y)$ is convergent, so a function $d_{\{[0,n]\}_{n\in\mathbb{N}}}$ is a pseudometric on $\mathbb{R}$.

Proposition 30.9. *Let U be arbitrary non-empty set. Then the following conditions hold:*

(1) *Every partition of a set U determines a pseudometric on U.*
(2) *If Π is a partition of a set U, then (U,d_Π) is a pseudometric spaces double bounded, so it is also lower bounded in each point.*
(3) *If Π is a partition of U, then a relation $\approx_\Pi$ atomizes a pseudometric space (U,d_Π).*

Proof. (1) Let Π be a partition of a set U and let $A \in \Pi$. Note that for all $x,y \in U$ $d_A(x,y) = 1$ if and only if $x \in A$ and $y \notin A$ or contrary $x \notin A$ and $y \in A$. In other cases $d_A(x,y) = 0$, thus independently on the cardinality of partition Π, it determines pseudometric d_Π on U.

(2) Let Π be the partition of set U. In point (1) we have shown that function d_Π is a pseudometric on U, consequently pair (U,d_Π) is a pseudometric space. Note that $d_\Pi(x,y) = 0$ if and only if there is a set $A \in \Pi$, such that $x,y \in A$, in all remain cases $d_\Pi(x,y) = 2$. This is because if x and y are chosen from different blocs of partition Π, say $x \in B$, $y \in C$, then $d_B(x,y) = 1$ and $d_C(x,y) = 1$, whereas for other $D \in \Pi$, $d_D(x,y) = 0$, thus $d_\Pi = \sum_{E\in\Pi} d_E(x,y) = 2$. Therefore both upper and lower diameters of space (U,d_Π) are equal to 2. Thus particularly space (U,d_Π) is lower bounded in every point.

(3) Let Π be a partition of set U. Since $d_\Pi(x,y) = 0$ if end only if there is set $A \in \Pi$, such that $x,y \in A$, then for every $x \in U$ and for every $B \in \Pi$ such that $x \in B$ the following holds $x_{/\sim_{d_\Pi}} = B$. Therefore $U_{/\sim_{d_\Pi}} = \Pi$. From the Abstraction Principle it follows that $\Pi = U_{/\approx_\Pi}$, consequently $U_{/\sim_{d_\Pi}} = U_{/\approx_\Pi}$ and therefore $\sim_{d_\Pi} = \approx_\Pi$. Thus relation $\approx_\Pi$ atomizes space (U,d_Π).

Proposition 30.10. *Let U be a non-empty set, let also a family $\mathcal{C} \subseteq \mathcal{P}(U)$ determine a pseudometric on U and let $\approx_{\mathcal{C}} \neq \nabla_U$. Then:*

(1) *Pseudometrics $d_{\mathcal{C}}$ takes its values in the set of natural numbers, i.e. $d_{\mathcal{C}}(U \times U) \subseteq \mathbb{N}$.*
(2) *A relation $\approx_{\mathcal{C}}$ atomizes a space $(U, d_{\mathcal{C}})$, i.e. $\sim_{d_{\mathcal{C}}} = \approx_{\mathcal{C}}$.*
(3) *A pseudometric space $(U, d_{\mathcal{C}})$ is lower bounded, so it is lower bounded in each point.*
(4) *$top(U, d_{\mathcal{C}}) = Sg^c(U_{/\approx_{\mathcal{C}}})$, i.e. open sets in the pseudometric space $(U, d_{\mathcal{C}})$ are exactly definable sets in an approximation space $(U, \approx_{\mathcal{C}})$.*

Proof. Point (1) follows directly from definition 30.8.

(2) From definition of function $d_{\mathcal{C}}$ we have that for arbitrary $x, y \in U$, $d_{\mathcal{C}}(x, y) = 0$ iff for every $C \in \mathcal{C}$, $x \approx_{\mathcal{C}} y$. Thus, by the definitions of the relations $\sim_{d_{\mathcal{C}}}$ and $\approx_{\mathcal{C}}$ we have that $(x, y) \in \sim_{d_{\mathcal{C}}} \Leftrightarrow (x, y) \in \approx_{\mathcal{C}}$. Thus $\sim_{d_{\mathcal{C}}} = \approx_{\mathcal{C}}$.

(3) Since $\approx_{\mathcal{C}} \neq \nabla_U$, then there are elements $x, y \in U$ such that $d_{\mathcal{C}}(x, y) > 0$. Note that number 1 is a lower diameter of space $(U, d_{\mathcal{C}})$,thus space $(U, d_{\mathcal{C}})$ is lower bounded, so by corollary 30.4 we get that space $(U, d_{\mathcal{C}})$ is lower bounded in every point.

(4) From point (3) of this proposition we know that pseudometric space $(U, d_{\mathcal{C}})$ is lower bounded in every point. By virtue of proposition 30.1 this is equivalent to $top(U, d_{\mathcal{C}}) = Sg^c(U_{/\sim_{d_{\mathcal{C}}}})$. From point (2) we know that $\sim_{d_{\mathcal{C}}} = \approx_{\mathcal{C}}$. Thus $top(U, d_{\mathcal{C}}) = Sg^c(U_{/\approx_{\mathcal{C}}})$.

Notice that not every pseudometric space determined by a family of sets is upper bounded. As an example of a pseudometric of such type can serve the space from Example 30.10:

Example 30.11. Let $\{[0, n]\}_{n \in \mathbb{N}}$ be a family of sets from Example 30.10. Thus $(\mathbb{R}, d_{\{[0,n]\}_{n \in \mathbb{N}}})$ is a pseudometric space. Let $a \in \mathbb{N}$ and $a \neq 0$. Let us choose arbitrary positive real number $r \in \mathbb{R}_+$. Notice that $d_{\{[0,n]\}_{n \in \mathbb{N}}}(r, r + a) = a$. Notice also that for arbitrary choice of $a \in \mathbb{N}^*$ a function $d_{\{[0,n]\}_{n \in \mathbb{N}}}$ can take arbitrary big value in the set $\mathbb{N}$. Thus a pseudometric space $(\mathbb{R}, d_{\{[0,n]\}_{n \in \mathbb{N}}})$ is not upper bounded.

Considerations of this section lead to the following theorem which gives an answer to the question posed at its beginning: from Proposition 30.10.2 it follows that

Theorem 30.3. *Every approximation space atomizes some pseudometric space.*

In other words, every equivalence relation can be represented by or derived from some pseudometric space. One of the main directions of developing rough set theory is based on generalizing approximation space to the case of tolerance approximation spaces and tolerance approximation operators defined for arbitrary tolerance relations, i.e. for not necessary transitive tolerances [21]. Thus another natural question arises: whether every tolerance relation can be represented by means of pseudometric spaces?

In order to answer this question let us consider some way od determining tolerance relations by pseudometric space. Let (U,p) be a pseudometric space and let $\varepsilon \in [0,+\infty)$. Elements $x,y \in U$ are in the relation p_ε if and only if $p(x,y) \leqslant \varepsilon$. It is easy to show that p_ε is a tolerance relation on U. It turns out that there is a positive answer to the last question: an arbitrary tolerance relation can be determined in this way as it is shown by the following "technical" example.

Example 30.12. Let (U,τ) be a tolerance space (i.e. τ is a tolerance relation on U). A function $\rho : U \times U \longrightarrow [0,+\infty)$ is defined as follows:

$$\rho(x,y) =_{def} \begin{cases} 0 : x = y \\ 1 : (x,y) \in \tau \ \& \ x \neq y \\ 2 : (x,y) \notin \tau. \end{cases}$$

>From this definition it follows that for arbitrary $x \in U$, $\rho(x,x) = 0$. The symmetry condition is also obvious and one can easy check that the triangle inequality conditions also holds. Thus a function ρ is a pseudometric on U. Let us note that $\rho(x,y) \leqslant 1 \Leftrightarrow (x,y) \in \tau$, for any $x,y \in U$. Thus $\rho_1 = \tau$, i.e. a tolerance relation determined by the pseudometric ρ and the number 1 is equal to a tolerance relation τ.

30.6 Pseudometrizability of Topological Spaces

Let us recall that a topological space $(X,\mathcal{O})$ is metrizable iff there is a metric space (X,d) such that $top(U,d) = \mathcal{O}$, i.e. a family $\mathcal{O}$ exactly consists of open sets in the metric space (X,d). It is well known that not all topological spaces are metrizable. The problem of characterization of metrizable topological spaces is one of the main in topology and it has been extensively studied (see [3]). For pseudometric spaces we can consider an analogical notion.

Definition 30.9. A topological space $(U,\mathcal{O})$ is *pseudometrizable* if there is a pseudometric space (U,p) such that

$$\mathcal{O} = top(U,p).$$

Then a topological space $(U,\mathcal{O})$ is said to be *metrizable jest by the space* (U,p).

Theorem 30.4. *A topological space $(U,\mathcal{O})$ is clo-open if and only if it is pseudometrizable by a pseudometric space which is lower bounded in each point.*

Proof. ($\Rightarrow$) Let space $(U,\mathcal{O})$ be clo-open so that $Clop(U,\mathcal{O}) = \mathcal{O}$. Therefore $\mathcal{O}$ is a complete field of sets thus every element of family $\mathcal{O}$, taken as a complete field of sets, is a union of atoms of this field. Other words $At(\mathcal{O})$ is a basis of topology $\mathcal{O}$. As we mention at the end of Section 30.2, it can be shown that know that for arbitrary $\mathcal{C} \subseteq \mathcal{P}(U)$ it holds that $At(Sg^c(\mathcal{C})) = U_{/\approx_\mathcal{C}}$ [30, 32], and with this,

substituting for $\mathcal{C}$ topology $\mathcal{O}$, we get $At(Sg^c(At(\mathcal{O}))) = U_{/\approx_{At(\mathcal{O})}}$. Since $\mathcal{O}$ is a complete field of sets, since every element of a complete field of sets is a union of its atoms, then $\mathcal{O} = Sg^c(At(\mathcal{O}))$ [30, 32]. From both these equalities we get equality $At(\mathcal{O}) = U_{/\approx_{At(\mathcal{O})}}$. From proposition 30.10.1 we know that $\sim_{d_{At(\mathcal{O})}} = \approx_{At(\mathcal{O})}$, so that $U_{/\sim_{d_{At(\mathcal{O})}}} = U_{/\approx_{At(\mathcal{O})}}$ and thus $At(\mathcal{O}) = U_{/\sim_{d_{At(\mathcal{O})}}}$. Therefore we conclude that topology $\mathcal{O}$ is exactly composed of sets being unions of atoms of pseudometric space $(U, d_{At(\mathcal{O})})$. From proposition 30.10.3 we know that space $(U, d_{At(\mathcal{O})})$ is lower bounded in every point. Thus from proposition 30.1 we conclude that open sets in space $(U, d_{At(\mathcal{O})})$ are exactly the sets being unions of atoms of space $(U, d_{At(\mathcal{O})})$. We have shown that $top(U, d_{At(\mathcal{O})}) = \mathcal{O}$. Thus topological space $(U, \mathcal{O})$ is pseudometrizable via space $(U, d_{At(\mathcal{O})})$, which is lower bounded in every point.

($\Leftarrow$) Let topological space $(U, \mathcal{O})$ be pseudometrizable via pseudometric space (U, p), which is lower bounded in every point. Thus $\mathcal{O} = top(U, p)$. Lower boundness of space (U, p) in every point by proposition 30.1 is equivalent to $top(U, p) = Sg^c(U/\sim_p)$. Thus $\mathcal{O} = Sg^c(U/\sim_p)$, so that $\mathcal{O}$ is a complete field of sets. Hence $\mathcal{O}$ is a clo-open topology.

30.7 Equivalence of Pseudometric Spaces

The next concept which can be naturally generalized from the class of metric space onto the class of pseudometric spaces is that of equivalence of metric spaces.

Definition 30.10. Let X be any non-empty set and p i q be pseudometrics on X. Spaces (X, p) i (X, q) are said to be *equivalent* if and only if

$$top(X, p) = top(X, q).$$

In other words pseudometric spaces are equivalent iff they determined the same open sets.

Example 30.13. Let $n \in \mathbb{N}\backslash\{0\}$ and $(\mathbb{Z}, p_n)$ be a pseudometric space from Example 30.2. >From this example we know that a congruence $\equiv_n$ atomizes a space $(\mathbb{Z}, p_n)$ and from Example 30.8 we know that a space $(\mathbb{Z}, p_n)$ is double bounded and so by the proposition 30.4 $(\mathbb{Z}, p_n)$ is also lower bonded in each point. The proposition 30.1 it is equivalent to the fact that $top(\mathbb{Z}, p_n) = Sg^c(\mathbb{Z}_{/\equiv_n})$.

Since $\equiv_n \in Eq(\mathbb{Z})$, then $\mathbb{Z}_{/\equiv_n}$ is a partition of the set $\mathbb{Z}$, thus from the proposition 30.9 we get that a function $d_{\mathbb{Z}_{/\equiv_n}}$ is a pseudometrics on $\mathbb{Z}$. A pseudometric space $(\mathbb{Z}, d_{\mathbb{Z}_{/\equiv_n}})$ is lower bounded in each point and $\approx_{\mathbb{Z}_{/\equiv_n}}$ is a relation atomizing this space The proposition 30.1 this is equivalent to $top(\mathbb{Z}, d_{\mathbb{Z}_{/\equiv_n}}) = Sg^c(\mathbb{Z}_{/\equiv_n})$. >From the principle of abstraction it follows that $\equiv_n = \approx_{U_{/\equiv_n}}$, so $Sg^c(\mathbb{Z}_{/\equiv_n}) = Sg^c(\mathbb{Z}_{/\approx_{\mathbb{Z}_{/\equiv_n}}})$. Then $top(\mathbb{Z}, p_n) = top(\mathbb{Z}, d_{\mathbb{Z}_{/\equiv_n}})$. Thus we have shown that pseudometric spaces $(\mathbb{Z}, p_n)$ and $(\mathbb{Z}, d_{\mathbb{Z}_{/\equiv_n}})$ are equivalent. Let us note also that

pseudometrics p_n and $d_{\mathbb{Z}_{/\equiv_n}}$ are different functions, so spaces $(\mathbb{Z}, p_n)$ and $(\mathbb{Z}, d_{\mathbb{Z}_{/\equiv_n}})$ are also different.

Now we can present next characterization of pseudometric spaces which are lower bounded in each point.

Theorem 30.5. *Every pseudometric space lower bounded in each point is equivalent to a double bounded pseudometric space.*

Proof. Let (U,p) be a pseudometric space lower bounded in every point. Thus from proposition 30.1 we know that $top(U,p) = Sg^c(U_{/\sim_p})$. From proposition 30.9.1 we know that function $d_{U_{/\sim_p}}$ is a pseudometric on U, whereas from proposition 30.9.3 we know that $\approx_{U_{/\sim_p}}$ atomizes space $(U, d_{U_{/\sim_p}})$, then $\sim_{d_{U_{/\sim_p}}} = \approx_{U_{/\sim_p}}$. From proposition 30.9.2 we get that space $(U, d_{U_{/\sim_p}})$ is lower bounded in every point, what by proposition 30.1 is equivalent to $top(U, d_{U_{/\sim_p}}) = Sg^c(U_{/\approx_{U_{/\sim_p}}})$. From the Abstraction Principle it follows that $U_{/\sim_p} = U_{\approx_{U_{/\sim_p}}}$, therefore $Sg^c(U_{/\sim_p}) = Sg^c(U_{/\approx_{U_{/\sim_p}}})$, so $top(U,p) = top(U, d_{U_{/\sim_p}})$. Thus pseudometric spaces (U,p) and $(U, d_{U_{/\sim_p}})$ are equivalent. From proposition 30.9.2 it follows that space $(U, d_{U_{/\sim_p}})$ is bounded. We have thus proved that space (U,p) is equivalent to a double bounded space determined by a partition of set U.

30.8 Topological Characterization of Attribute Dependency

Results presented in this chapter can be applied to provide a topological characterization of dependencies in information systems and to generalization of this concept to attribute dependencies in information systems with different object domains. In Section 30.2 we mention the fact that every information system can be represented by some set space. Now we use this fact for the sake of presentation simplicity and assume that for information system $S = \langle U, At, Val_a \rangle$ and attribute families $B, D \subseteq At$ we have families of sets $\mathcal{B}, \mathcal{D} \subseteq \mathcal{P}(Ob)$ such that $ind(B) = \approx_{\mathcal{B}}$ and $ind(D) = \approx_{\mathcal{D}}$ (another way would be redefining discernibility functions d_C and $d_{\mathcal{C}}$ for attributes instead of sets). Let us remind that D *depends on* B in S, symbolically $B \Rightarrow_S D$, iff $ind(B) \subseteq ind(D)$. Let us mention also that we denote identity function on domain X by id_X, i.e. $id : X \longrightarrow X$ and $id(x) = x$ for every $x \in X$. In order to prove a topological characterization of attribute dependency let us recall also the well know fact from topology:

Lemma 30.1. *Let $(X, \mathcal{O})$ and $(X, \mathcal{V})$ be topological spaces, then the following conditions are equivalent:*

(1) *function $id_X : (X, \mathcal{O}) \longrightarrow (X, \mathcal{V})$ is continuous,*
(2) $\mathcal{V} \subseteq \mathcal{O}$.

Theorem 30.6. *Let $S = \langle U, At, Val_a \rangle$ be an information system and let $B, D \subseteq At$. Thus $(U, d_{\mathcal{B}})$ and $(U, d_{\mathcal{D}})$ are pseudometric spaces and the following conditions are equivalent:*

(1) $B \Rightarrow_S D$,
(2) *function* $id_U : (U, d_{\mathcal{B}}) \longrightarrow (U, d_{\mathcal{D}})$ *is continuous.*

Proof. Assume that $S = \langle U, At, Val_a \rangle$ is an information system and let $B, D \subseteq At$. Thus from proposition 30.10 it follows that $(U, d_{\mathcal{B}})$ and $(U, d_{\mathcal{D}})$ are pseudometric spaces lower bounded in each point.

(1) $\Rightarrow$ (2) If D depends on B, then $ind(B) \subseteq ind(D)$ an so $\approx_{\mathcal{B}} \subseteq \approx_{\mathcal{D}}$. One can note that in such case every set from $U_{/\approx_{\mathcal{D}}}$ is a union of some sets from $U_{/\approx_{\mathcal{B}}}$ i.e. for every $A \in U_{/\approx_{\mathcal{D}}}$ there is $\mathcal{B}_1 \subseteq \mathcal{B}$ such that $A = \bigcup \mathcal{B}_1$. From the facts mentioned at the end of section 30.2 we know that $U_{/\approx_{\mathcal{B}}} = At(Sg^c(\mathcal{B}))$ and that every set from $Sg^c(\mathcal{B})$ is a union of classes from $U_{/\approx_{\mathcal{B}}}$ [30, 32]. Therefore $U_{/\approx_{\mathcal{D}}} \subseteq Sg^c(\mathcal{B})$. We know also that every set in a complete field of sets is a union of atoms of this field. Thus $Sg^c(U_{/\approx_{\mathcal{D}}}) \subseteq Sg^c(\mathcal{B})$. For the same reason it holds that $Sg^c(\mathcal{B}) = Sg^c(U_{/\approx_{\mathcal{B}}})$. Therefore $Sg^c(U_{/\approx_{\mathcal{D}}}) \subseteq Sg^c(U_{/\approx_{\mathcal{B}}})$.

From proposition 30.10 we know that $\sim_{d_{\mathcal{B}}} = \approx_{\mathcal{B}}$ and $\sim_{d_{\mathcal{D}}} = \approx_{\mathcal{D}}$ i.e. relations $\approx_{\mathcal{B}}$ and $\approx_{\mathcal{D}}$ atomize pseudometric spaces $(U, d_{\mathcal{B}})$ and $(U, d_{\mathcal{D}})$ respectively and from theorem 30.1 we know that $top(U, p) = Sg^c(U_{/\sim_p})$ for pseudometric space $U(U, p)$ which is lower bounded in each point. Since pseudometric spaces $(U, d_{\mathcal{B}})$ and $(U, d_{\mathcal{D}})$ are lower bounded in each points, then $top(U, d_{\mathcal{B}}) = Sg^c(U_{/\approx_{\mathcal{B}}})$ and $top(U, d_{\mathcal{D}}) = Sg^c(U_{/\approx_{\mathcal{D}}})$. Therefore $top(U, d_{\mathcal{D}}) \subseteq top(U, d_{\mathcal{B}})$. Thus from lemma 30.1 we get that function $id_U : (U, top(U, d_{\mathcal{B}})) \longrightarrow (U, top(U, d_{\mathcal{D}}))$ is continuous for topologies $top(U, d_{\mathcal{B}})$ and $top(U, d_{\mathcal{B}})$. Therefore by proposition 30.7 it follows for pseudometric spaces $(U, d_{\mathcal{B}})$ and $(U, d_{\mathcal{D}})$ that function $id_U : (U, d_{\mathcal{B}}) \longrightarrow (U, d_{\mathcal{D}})$ is continuous as well.

(2) $\Rightarrow$ (1) Assume that function $id_U : (U, d_{\mathcal{B}}) \longrightarrow (U, d_{\mathcal{D}})$ is continuous. Then by proposition 30.7 we get that for every set $A \subseteq U$ which is open in pseudometric space $(U, d_{\mathcal{D}})$, the set $id^{-1}(A)$ is open in space $(U, d_{\mathcal{B}})$. Since $id^{-1}(A) = A$ for every set $A \subseteq U$ therefore $top(U, d_{\mathcal{D}}) \subseteq top(U, d_{\mathcal{B}})$. As it was shown above $top(U, d_{\mathcal{B}}) = Sg^c(U_{/\approx_{\mathcal{B}}})$ and $top(U, d_{\mathcal{D}}) = Sg^c(U_{/\approx_{\mathcal{D}}})$ and thus $Sg^c(U_{/\approx_{\mathcal{D}}}) \subseteq Sg^c(U_{/\approx_{\mathcal{B}}})$. Since $U_{/\approx_{\mathcal{D}}} \subseteq Sg^c(U_{/\approx_{\mathcal{D}}})$ and so $U_{/\approx_{\mathcal{D}}} \subseteq Sg^c(U_{/\approx_{\mathcal{B}}})$, then by the fact mentioned at the end of section 30.2 (see also [30, 32]) in particular every equivalence class $A \in U_{/\approx_{\mathcal{D}}}$ is a union of equivalence classes from $U_{/\approx_{\mathcal{D}}}$. Since since both $U_{/\approx_{\mathcal{B}}}$ and $U_{/\approx_{\mathcal{D}}}$ are partitions of space U, then one can show that every equivalence class $B \in U_{/\approx_{\mathcal{B}}}$ is contained is some equivalence class $D \in U_{/\approx_{\mathcal{D}}}$ (see for example [30, 32]). Thus it is easy to note that $\approx \mathcal{B} \subseteq \approx \mathcal{D}$ and so $ind(B) \subseteq ind(D)$. Therefore family of attributes D depends on family of attributes B in information system S. *Q.E.D*

Theorem 30.6 shows a direction for generalization of concept of attribute dependency for information systems with different object domains.

Definition 30.11. Let $S = \langle X, At_1, Val_a \rangle$ and $V = \langle Y, At_2, Val_a \rangle$ be information systems and let $B \subseteq At_1$ and $D \subseteq At_2$. We say that D *depends on* B with respect

to function $f : X \longrightarrow Y$, symbolically $B \Rightarrow_f D$ if for all $x, y \in X$ the following implication holds

$$x \; ind(B) \; y \; \Rightarrow f(x) \; ind(D) \; f(y).$$

Theorem 30.7. *Let $S = \langle X, At_1, Val_a \rangle$ and $V = \langle Y, At_2, Val_a \rangle$ be information systems and let $B \subseteq At_1$ and $D \subseteq At_2$. Thus $(X, d_{\mathcal{B}})$ and $(Y, d_{\mathcal{D}})$ are pseudometric spaces lower bounded in each point and the following conditions are equivalent:*

(1) $B \Rightarrow_f D$,
(2) *function* $f : (X, d_{\mathcal{B}}) \longrightarrow (Y, d_{\mathcal{D}})$ *is continuous.*

Proof. Assume that $S = \langle X, At_1, Val_a \rangle$ and $V = \langle Y, At_2, Val_a \rangle$ are information systems and let $B \subseteq At_1$ and $D \subseteq At_2$. From proposition 30.10.3 we know that pseudometric spaces $(X, d_{\mathcal{B}})$ and $(Y, d_{\mathcal{D}})$ are lower bounded in each point. Then from theorem 30.2 immediately we get equivalence $(1) \Leftrightarrow (2)$.

One of the domains when this generalized concept of dependency can be applied is hierarchical information processing and perception based computing, for example in hierarchical approximation of complex vague concepts where where information systems from higher levels are are constructed on the basis of information systems from lover levels (see e.g. [23, 25–27]).

30.9 Conclusions

We discussed pseudometric spaces from the perspective of rough set theory. This discussion is based on a fundamental connection between pseudometric spaces and approximation spaces: namely that every pseudometric space determines an approximation space which atomizes it. We investigated open sets in pseudometric spaces from that perspective. We specially focused on pseudometric spaces which are lower bounded in each point since their open sets coincide with definable sets in approximation spaces which atomize them. We have shown also that every equivalence relation atomizes some pseudometric space.

Results presented within this paper can be used for defining approximation operators based on pseudometric spaces: namely operators which depend both on atomizing relations and distances with respect to appropriate pseudometrics. They are essential in tolerance rough sets methods. They can be also used in a study on the notion of *nearness* and its connections with rough sets. An interesting thing is to investigate connections of pseudometric spaces with *near sets* [17, 18] as well as with tolerance relations and proximity spaces which are closely related to *nearness* [34, 35].

Obtained results, by their connection with tolerance spaces, can be used for construction new, generalized attributes which are essential for interactive information systems [24, 26, 27]. Tolerance and pseudometric spaces are promising in modelling complex vague concepts [20] both for hierarchical approximation of complex vague

concepts from lower-level data (*e.g.* sensory data) [1, 22, 23] and for decision making using Wisdom technology [4, 5], where intelligent autonomous agents make adaptively of correct judgments to a satisfactory degree in the face of real-life constraints (*e.g.* time constraints). Therefore rough set methods based on pseudometric as well as tolerance spaces are important part or rough-granular approach to interactive computing [26, 27] whereas they are indispensable in perception based computing [25].

Acknowledgements. Some results contained in this paper are taken from a PhD thesis [30] prepared by the author in the Department of Logic, Jagiellonian University. Research reported in this work has been supported by the individual research project realized within Homing Plus programme, edition 3/2011, of Foundation for Polish Science, co-financed from European Union, Regional Development Fund, and by the grant 2011/01/D/ST6/06981 from the Polish National Science Centre.

References

1. Bazan, J.: Hierarchical Classifiers for Complex Spatio-temporal Concepts. In: Peters, J.F., Skowron, A., Rybiński, H. (eds.) Transactions on Rough Sets IX. LNCS, vol. 5390, pp. 474–750. Springer, Heidelberg (2008)
2. Demri, S., Orłowska, E.: Incomplete Information: Structures, Inference, Complexity. Springer, Heidelberg (2002)
3. Engelking, R.: General Topology. PWN (1977)
4. Jankowski, J., Skowron, A.: Wisdom technology: A rough-granular approach. In: Marciniak, M., Mykowiecka, A. (eds.) Bolc Festschrift. LNCS, vol. 5070, pp. 3–41. Springer, Heidelberg (2009)
5. Jankowski, J., Skowron, A.: Rough granural computing in human-centric information processing. In: Cyran, K.A., Kozielski, S., Peters, J.F., Stańczyk, U., Wakulicz-Deja, A. (eds.) Man-machine Interactions, pp. 23–42. Springer, Heidelberg (2009)
6. Kopelberg, S.: General theory of Boolean algebras. In: Monk, J.D., Bonnet, R. (eds.) Handbook of Boolean Algebras. North-Holland (1989)
7. Kuratowski, K.: Wstêp do teorii mnogoœci i topologii. PWN, Warszawa (1980)
8. Orłowska, E., Pawlak, Z.: Representation of nondeterministic information. Theoretical Computer Science 29, 27–39 (1984)
9. Orłowska, E.: Reasoning with incomplete information: Rough set based information logics. In: Algar, V., Bergler, S., Dong, F.Q. (eds.) Incompleteness and Uncertainty in Information Systems Workshop, pp. 16–33. Springer, Heidelberg (1993)
10. Pawlak, Z.: Information systems – Theoretical foundation. Information Systems 6, 205–218 (1981)
11. Pawlak, Z.: Rough sets. International Journal of Computing and Information Sciences 18, 341–356 (1982)
12. Pawlak, Z.: Rough sets. Theoretical Aspects of Reasoning About Data. Kluwer Academic Publishers, Dordrecht (1991)
13. Pawlak, Z.: Some Issues on Rough Sets. In: Peters, J.F., Skowron, A., Grzymała-Busse, J.W., Kostek, B.z., Świniarski, R.W., Szczuka, M.S. (eds.) Transactions on Rough Sets I. LNCS, vol. 3100, pp. 1–58. Springer, Heidelberg (2004)
14. Pawlak, Z., Skowron, A.: Rudiments of rough sets. Information Science 177, 3–27 (2007)
15. Pawlak, Z., Skowron, A.: Rough sets: Some extensions. Information Science 177, 28–40 (2007)

16. Pawlak, Z., Skowron, A.: Rough sets and boolean reasoning. Information Science 177, 41–73 (2007)
17. Peters, J.F.: Near sets. Special theory about nearness of objects. Fundamenta Informaticae 73, 1–27 (2006)
18. Peters, J.F., Wasilewski, P.: Foundations of near sets. Information Sciences 179(18), 3091–3109 (2009)
19. Polkowski, L.: Rough Sets: Mathematical Foundations. Physica-Verlag (2002)
20. Skowron, A.: Rough sets and vague concepts. Fundamenta Informaticae 64(1-4), 417–431 (2005)
21. Skowron, A., Stepaniuk, J.: Tolerance approximation spaces. Fundamenta Informaticae 27, 245–253 (1996)
22. Skowron, A., Szczuka, M.: Toward Interactive Computations: A Rough-Granular Approach. In: Koronacki, J., Raś, Z.W., Wierzchoń, S.T., Kacprzyk, J. (eds.) Advances in Machine Learning II. SCI, vol. 263, pp. 23–42. Springer, Heidelberg (2010)
23. Skowron, A., Wang, H., Wojna, A., Bazan, J.G.: A Hierarchical Approach to Multimodal Classification. In: Ślęzak, D., Yao, J., Peters, J.F., Ziarko, W.P., Hu, X. (eds.) RSFDGrC 2005. LNCS (LNAI), vol. 3642, pp. 119–127. Springer, Heidelberg (2005)
24. Skowron, A., Wasilewski, P.: Information Systems in Modeling Interactive Computations on Granules. In: Szczuka, M., Kryszkiewicz, M., Ramanna, S., Jensen, R., Hu, Q. (eds.) RSCTC 2010. LNCS, vol. 6086, pp. 730–739. Springer, Heidelberg (2010)
25. Skowron, A., Wasilewski, P.: An Introduction to Perception Based Computing. In: Kim, T.-h., Lee, Y.-h., Kang, B.-H., Ślęzak, D. (eds.) FGIT 2010. LNCS, vol. 6485, pp. 12–25. Springer, Heidelberg (2010)
26. Skowron, A., Wasilewski, P.: Information systems in modeling interactive computations on granules. Theoretical Computer Science 412(42), 5939–5959 (2011)
27. Skowron, A., Wasilewski, P.: Toward interactive rough–granular computing. Control & Cybernetics 40(2), 1–23 (2011)
28. Ślęzak, D., Wasilewski, P.: Granular Sets – Foundations and Case Study of Tolerance Spaces. In: An, A., Stefanowski, J., Ramanna, S., Butz, C.J., Pedrycz, W., Wang, G. (eds.) RSFDGrC 2007. LNCS (LNAI), vol. 4482, pp. 435–442. Springer, Heidelberg (2007)
29. Wasilewski, P.: Dependency and supervenience. In: Proc. of the Concurrency, Specification and Programming (CS& P 2003). Warsaw Univerity (2003)
30. Wasilewski, P.: On selected similarity relations and their applications into cognitive science. Unpublished doctoral dissertation, Jagiellonian University: Department of Logic, Cracow, Poland (2004) (in Polish)
31. Wasilewski, P.: Concept Lattices vs. Approximation Spaces. In: Ślęzak, D., Wang, G., Szczuka, M.S., Düntsch, I., Yao, Y. (eds.) RSFDGrC 2005. LNCS (LNAI), vol. 3641, pp. 114–123. Springer, Heidelberg (2005)
32. Wasilewski, P.: Indiscernibility relations (in preparation)
33. Wasilewski, P., Ślęzak, D.: Foundations of rough sets from vagueness perspective. In: Hassanien, A.E., Suraj, Z., Ślęzak, D., Lingras, P. (eds.) Rough Computing. Theories, Technologies and Applications, pp. 1–37. IGI Global, Hershey (2008)
34. Wasilewski, P., Peters, J., Ramanna, S.: Perceptual Tolerance Intersection. In: Szczuka, M., Kryszkiewicz, M., Ramanna, S., Jensen, R., Hu, Q. (eds.) RSCTC 2010. LNCS, vol. 6086, pp. 277–286. Springer, Heidelberg (2010)
35. Wasilewski, P., Peters, J., Ramanna, S.: Perceptual Tolerance Intersection. In: Peters, J.F., Skowron, A., Chan, C.-C., Grzymala-Busse, J.W., Ziarko, W.P. (eds.) Transactions on Rough Sets XIII. LNCS, vol. 6499, pp. 159–174. Springer, Heidelberg (2011)

Index