Ladjel Bellatreche
Mukesh K. Mohania (Eds.)

LNCS 8057

Data Warehousing and Knowledge Discovery

15th International Conference, DaWaK 2013
Prague, Czech Republic, August 2013
Proceedings

Springer

Lecture Notes in Computer Science 8057

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Ladjel Bellatreche Mukesh K. Mohania (Eds.)

Data Warehousing and Knowledge Discovery

15th International Conference, DaWaK 2013
Prague, Czech Republic, August 26-29, 2013
Proceedings

Springer

Volume Editors

Ladjel Bellatreche
LIAS/ISAE-ENSMA, Téléport 2
1 avenue Clément Ader, BP 40109
86961 Futuroscope Chasseneuil Cedex, France
E-mail: ladjel.bellatreche@ensma.fr

Mukesh K. Mohania
IBM India Research Lab
Block No. 1, IIT
Hauz Khas, New Delhi 110016, India
E-mail: mkmukesh@in.ibm.com

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-40130-5 e-ISBN 978-3-642-40131-2
DOI 10.1007/978-3-642-40131-2
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2013944640

CR Subject Classification (1998): H.2, H.3, I.2.6, H.4, J.1

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

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

Preface

Data warehousing and knowledge discovery has been widely accepted as a key technology for enterprises and organizations to improve their abilities in data analysis, decision support, and the automatic extraction of knowledge from data. As a consequence of this spectacular development, new types of jobs related to the data warehouses have been created: data architect, data analysts, etc. With the big data and cloud dimensions, the management of relevant data to be considered in the warehouse becomes more and more complex in both structure and semantics. These dimensions also bring new issues such as scalability, a new computing infrastructure, and new types of data. Consequently, the process of retrieval and knowledge discovery from this huge amount of heterogeneous complex data represents the litmus test for the research in the area.

During the past few years, the International Conference on Data Warehousing and Knowledge Discovery (DaWaK) has become one of the most important international scientific events bringing together researchers, developers and practitioners to discuss the latest research issues and experiences in developing and deploying data warehousing and knowledge discovery systems, applications, and solutions. This year's conference (DaWaK 2013), built on this tradition of facilitating the cross-disciplinary exchange of ideas, experience, and potential research directions. DaWaK 2013 sought to introduce innovative principles, methods, algorithms and solutions, industrial products, and experiences to challenging problems faced in the development of data warehousing, knowledge discovery, data mining applications, and the emerging area of "cloud intelligence."

This year we received 89 papers and the Program Committee finally selected 24 full papers and eight short papers, making an acceptance rate of 36% of submitted papers. The accepted papers cover a number of broad research areas on both theoretical and practical aspects of data warehouse and knowledge discovery. In the area of data warehousing, the topic covered included the conceptual design, query optimization, map reduce paradigm, scalability, data compression, materialized views, data partitioning, distributed and parallel processing and data warehouses and data mining applications integration, recommendation and personalization, multidimensional analysis of text documents, and data warehousing for real-world applications such as health, spatial applications, energy, etc. In the areas of data mining and knowledge discovery, the topics included stream data analysis and mining, dimensionality reduction, traditional data mining techniques topics such as frequent item sets, clustering, association, classification ranking and application of data mining technologies to real-world problems. It is especially notable to see that some papers covered emerging real-world applications such as bioinformatics, social network, mobile data, energy power, email management, environment surveillance as well as

integration of multiple technologies such as conceptual modeling, evaluation metrics, and OLAP mining.

We would like to thank all the authors for submitting their research paper in DaWaK 2013. We express our gratitude to all the Program Committee members and the external reviews, who reviewed the papers very profoundly and in a timely manner. Finally, we would like to thank Mrs. Gabriela Wagner for her endless help and support.

May 2013

Ladjel Bellatreche
Mukesh K. Mohania

Organization

Conference Program Chairs

Ladjel Bellatreche	ENSMA, France
Mukesh Mohania	IBM India

Program Committee

Alberto Abelló	Universitat Politecnica de Catalunya, Spain
Rajeev Agrawal	North Carolina A&T State University, USA
Reda Alhajj	University of Calgary, Canada
Manish Anand	Salesforce.com, USA
Torben Bach Pedersen	Aalborg University, Denmark
Elena Baralis	Politecnico di Torino, Italy
Ladjel Bellatreche	ENSMA, France
Ladjel Bellatreche	LISI/ENSMA, France
Petr Berka	University of Economics, Prague, Czech Republic
Jorge Bernardino	ISEC - Polytechnic Institute of Coimbra, Portugal
Vasudha Bhatnagar	Delhi University, India
Manish Bhide	IBM Software Lab, India
Omar Boussaid	University of Lyon, France
Stephane Bressan	National University of Singapore, Singapore
Erik Buchmann	Karlsruhe Institute of Technology, Germany
Longbing Cao	University of Technology Sydney, Australia
Nitesh Chawla	University of Notre Dame, USA
Frans Coenen	The University of Liverpool, UK
Bruno Cremilleux	Université de Caen, France
Judith Cushing	The Evergreen State College, USA
Alfredo Cuzzocrea	ICAR-CNR and University of Calabria, Italy
Karen Davis	University of Cincinnati, USA
Frank Dehne	Carleton University, Canada
Antonios Deligiannakis	Technical University of Crete, Greece
Alin Dobra	University of Florida, USA
Dejing Dou	University of Oregon, USA
Curtis Dyreson	Utah State University, USA
Johann Eder	University of Klagenfurt, Austria
Floriana Esposito	University of Bari "Aldo Moro", Italy
Vladimir Estivill-Castro	Griffith University, Australia

Christie Ezeife	University of Windsor, Canada
Ling Feng	Tsinghua University, China
Eduardo Fernandez_Medina	University of Castilla-La Mancha, Spain
Pedro Furtado	Universidade de Coimbra, Portugal
Ahmad Ghazal	Teradata, USA
Sergio Greco	University of Calabria, Italy
Neelima Gupta	University of Delhi, India
Frank Hoppner	Ostfalia University of Applied Sciences, Germany
Jimmy Huang	York University, Canada
Hasan Jamil	University of Idaho, USA
Chris Jermanine	Rice University, USA
Domingo-Ferrer Josep	University Rovira i Virgili, Spain
Vineet Kansal	ITS, India
Murat Kantarcioglu	University of Texas at Dallas, USA
Panagiotis Karras	Rutgers University, USA
Taghi Khoshgoftaar	Florida Atlantic University, USA
Sang-Wook Kim	Hanyang University, Republic of Korea
Arno Knobbe	Universiteit Utrecht, The Netherlands
Jens Lechtenboerger	Westfalische Wilhelms - Universität Münster, Germany
Wolfgang Lehner	Dresden University of Technology, Germany
Carson K. Leung	The University of Manitoba, Canada
Jinyan Li	University of Technology, Sydney, Australia
Sofian Maabout	University of Bordeaux, France
Patrick Marcel	Université François Rabelais Tours, France
Patrick Martin	Queen's University, Ontario, Canada
Maryvonne Miquel	INSA de Lyon, France
Mukesh Mohania	IBM India, India
Anirban Mondal	Xerox Research, India
Carlos Ordonez	University of Houston, USA
Apostolos Papadopoulos	Aristotle University, Greece
Jeffrey Parsons	Memorial University of Newfoundland, Canada
Dhaval Patel	National University of Singapore, Singapore
Lu Qin	The Chinese University of Hong Kong, Hong Kong
Sriram Raghavan	IBM Research - India
Zbigniew W. Ras	University of North Carolina, USA
Stefano Rizzi	University of Bologna, Italy
Domenico Sacca'	University of Calabria, Italy
Maria Luisa Sapino	Università degli Studi di Torino, Italy
Kai-Uwe Sattler	Ilmenau University of Technology, Germany
Neeraj Sharma	India IBM Labs, India
Cheng Sheng	Google Switzerland

Alkis Simitsis	HP Labs, Greece
Domenico Talia	University of Calabria, Italy
David Taniar	Monash University, Australia
Olivier Teste	IRIT, University of Toulouse, France
Dimitri Theodoratos	New Jersey Institute of Technology, USA
A Min Tjoa	Vienna University of Technology, Austria
Juan-Carlos Trujillo Mondéjar	University of Alicante, Spain
Panos Vassiliadis	University of Ioannina, Greece
Millist Vincent	University of South Australia
Gabriela Wagner	DEXA, Austria
Fan Wang	Microsoft, USA
Wei Wang	University of New South Wales, Sydney, Australia
Robert Wrembel	Poznan University of Technology, Poland
Wolfram Wöß	Johannes Kepler University Linz, Austria
Jierui Xie	Oracle Corporation, USA
Bin Zhou	University of Maryland, Baltimore County, USA
Esteban Zimányi	Université Libre de Bruxelles, Belgium

External Reviewers

Cem Aksoy	New Jersey Institute of Technology, USA
Hassan Ali	IRIT, University of Toulouse, France
Fabrizio Angiulli	University of Calabria, Italy
Carlos Blanco	University of Cantabria, Spain
Ananya Dass	New Jersey Institute of Technology, USA
Claudia d'Amato	University of Bari, Italy
Nicola Di Mauro	University of Bari, Italy
Nicola Fanizzi	University of Bari, Italy
Christina Feilmayr	Johannes Kepler University Linz, Austria
Arnaud Giacometti	Université de Tours, France
David Gil	University of Alicante, Spain
Himanshu Gupta	IBM, India
Nicolas Hanusse	University of Bordeaux, France
Dominique Haoyuan Li	Université de Tours, France
Fan Jiang	University of Manitoba, Canada
Petar Jovanovic	Universitat Politecnica de Catalunya-BarcelonaTech, Spain
Sharanjit Kaur	University of Delhi, India
Julius Köpke	University of Klagenfurt, Austria
Christian Koncilia	University of Klagenfurt, Austria
Thomas Leitner	Johannes Kepler University Linz, Austria
Stefano Lodi	University of Bologna, Italy

Noël Novelli	University of Marseille, France
Andrea Pugliese	University of Calabria, Italy
Jianbin Qin	University of New South Wales, Australia
Julien Rabatel	Université de Caen, France
Sayan Ranu	IBM, India
Cristina Romero	University Rovira i Virgili, Spain
Guillem Rufian	University Rovira i Virgili, Spain
Kameshwaran Sampath	IBM, India
Rakhi Saxena	University of Delhi, India
Arnaud Soulet	Université de Tours, France
Syed K. Tanbeer	University of Manitoba, Canada
Jiao Tao	Oracle Corporation, USA
Jyotsna Thalreja	University of Delhi, India
Christian Thomsen	Aalborg University, Denmark
Ronan Tournier	IRIT, Paul Sabatier University, Toulouse, France
Pierangelo Veltri	University Magna Graecia of Catanzaro, Italy
José Luis Vivas	University Rovira i Virgili, Spain
Arnau Vives	University Rovira i Virgili, Spain
Xiaoying Wu	Wuhan University, China

Table of Contents

Modeling and ETL

Query Optimization and Parallelism

Spatial Data Warehouses and Applications

Text Mining and OLAP

Recommendation and Prediction

Data Mining Optimization and Machine Learning Techniques

Mining and Processing Data Stream

Clustering and Data Mining Applications

Social Network and Graph Mining

Event Sequence and Web Mining

An Analytics-Aware Conceptual Model for Evolving Graphs

Amine Ghrab[1,2], Sabri Skhiri[1], Salim Jouili[1], and Esteban Zimányi[2]

[1] Eura Nova R&D, Mont-Saint-Guibert, Belgium
firstname.lastname@euranova.eu
[2] Université Libre de Bruxelles, Belgium
ezimanyi@ulb.ac.be

Abstract. Graphs are ubiquitous data structures commonly used to represent highly connected data. Many real-world applications, such as social and biological networks, are modeled as graphs. To answer the surge for graph data management, many graph database solutions were developed. These databases are commonly classified as NoSQL graph databases, and they provide better support for graph data management than their relational counterparts. However, each of these databases implement their own operational graph data model, which differ among the products. Further, there is no commonly agreed conceptual model for graph databases.

In this paper, we introduce a novel conceptual model for graph databases. The aim of our model is to provide analysts with a set of simple, well-defined, and adaptable conceptual components to perform rich analysis tasks. These components take into account the evolving aspect of the graph. Our model is analytics-oriented, flexible and incremental, enabling analysis over evolving graph data. The proposed model provides a typing mechanism for the underlying graph, and formally defines the minimal set of data structures and operators needed to analyze the graph.

1 Introduction

The relational model was considered for several decades as the default choice for data modeling and management applications. However, with the rise of Big Data, relational databases fell short of complex applications expectations. Big Data refers to data generated at unpredictable speed, scale, and size from heterogeneous sources, such as web logs and social networks. The distribution and variety of data makes ensuring ACID properties, required by the relational model, a very challenging task. This situation has lead to the development of new data models and tools, known as the NoSQL movement. NoSQL models are based on trading consistency for performance according to the CAP theorem, in contrast to relational ACID properties.

NoSQL databases can be divided into four families, namely key/value stores, column stores, document databases, and graph databases. Of particular relevance to this paper is the analysis of graph databases. Graphs have the benefit

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 1–12, 2013.

of revealing valuable insights from both the network structure and the data embedded within the structure [1]. Complex real-world problems, such as intelligent transportation as well as social and biological network analysis, could be abstracted and solved using graphs structures and algorithms. In this paper we introduce a new graph modeling approach for effective analysis of evolving graphs. By evolving we mean the variation of the values of an attribute across a discrete domain. Evolution could be over time, quantity, region, etc. In the corresponding non-evolving graph, the information would be discarded when the attributes or the topology changes.

The model introduces a typing system that entails explicit labeling of the graph elements. This might introduce a redundancy in the graph, since part of the facts could be discovered while traversing the data. However, we tolerate this redundancy in favor of richer and smoother analysis. Trading redundancy for the sake of better performance is a frequent choice when it comes to designing analytics-oriented data stores such as data warehouses. Within the proposed model, we define a set of operators to manipulate analytics-oriented evolving graphs. The goal is to help analysts navigate and extract relevant portions of the graph. Here, we provide the minimal set of operators. Nevertheless, richer analysis scenarios could be achieved by combining these operators.

We consider as a running example of this paper the Epinion product rating network [2], shown on Figure 1. The network is composed of a set of users grouped by group, and products grouped by category. Each user has a profile, a list of products ratings, and a linked by trust relationships with other users.

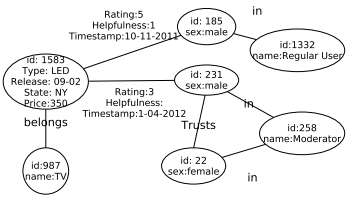


Fig. 1. Product rating network

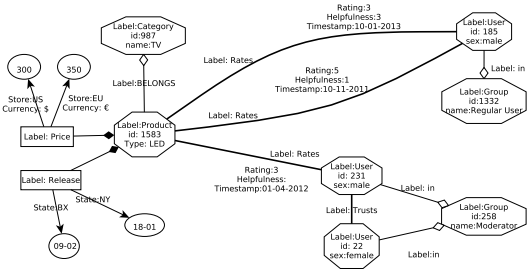


Fig. 2. Evolving product rating network

This network is sufficient to answer queries about the average rating of a product, or detection of communities of users. However, information about the evolution of the price by region or the history of rating of a given product by a user is impossible to obtain. This data is discarded and not versioned for further reuse. Hence, we enrich the original model with a typing system supporting network evolution. The evolving network keeps track of information such as the evolution of the price by region and the history of a product's rating by a user. Figure 2 depicts a part of the evolving network example. The evolving network could be used to answer rich queries like : (1) correlations between sales decrease

and product rating evolution, (2) detection of popular and trendy products, and (3) discovery of active and passive users. The previous queries could be then reused in richer scenarios such as (4) recommendation of new products, and (5) targeted advertising for influential people in the network.

The contributions of our work are summarized as follows:

- We define of a conceptual model for evolving graphs. The model is designed to handle analytics over large graphs by means of a novel typing mechanism.
- We propose a comprehensive set of querying operators to perform interactive graph querying through subgraph extraction functionalities.
- We describe a detailed use case of the model by reusing it as the ground for multidimensional analysis of evolving graphs.

The remainder of the paper is organized as follows. In Section 2, we develop a conceptual model to represent analytics-oriented evolving graphs. Section 3 defines the fundamental, general-purpose operators for querying the model. Section 4 demonstrates the usefulness of the proposed model for complex analytics by using it as the basis for multidimensional analysis. Section 5 discusses related work and compares it to our proposed model. Finally, Section 6 sketches future works and concludes the paper.

2 Evolving Graph Model

In this section, we present the evolving graph model serving as basis for the analysis framework. The **input** to our framework is a directed, attributed, heterogeneous multi-graph. Attributes are a map of key/value pairs attached to nodes and edges of the graph. Nodes (resp., edges) may have different attributes, and multiple edges may link the same pair of nodes.

We first define the typing mechanism that characterizes nodes and edges. We propose three types of nodes, defined next.

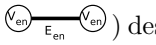
Definition 1. An **entity node** is defined by a tuple $\langle label, K_a, O_a \rangle$ where (1) *label* denotes the type of the entity node, such as user or product, (2) K_a is the map of key attributes that univocally identify the entity node, and (3) O_a is the map of optional attributes. The attributes of an entity node are immutable. The set of entity nodes is denoted as V_{en}. □

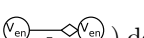
Definition 2. An **evolving node** keeps track of the discrete evolution of the attributes of entity nodes. Attributes of entity nodes that are subject to change are typed as evolving nodes. An evolving node contains only a label denoting its name and reflecting the original attribute it represents. Changes are treated as punctual events and reflect the discrete evolution of the attributes. The set of evolving nodes is denoted as V_{ev}. □

Definition 3. A **value node** has a unique attribute representing the value of its corresponding evolving node in a given context. The set of value nodes is denoted as V_v. □

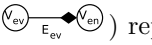
We adopt the UML notation for relationships to represent the edges of the graph. With regards to the nodes they link, we classify the edges as follows.

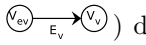
Edges linking entity nodes are of two types:

Definition 4. An **entity edge** (denoted by ) describes the *association* between two entity nodes. The set of entity edges is denoted as E_{en} ($E_{en} \subseteq V_{en} \times V_{en}$). □

Definition 5. A **hierarchical edge** (denoted by ) depicts an *aggregation* (i.e., part-of) relationship between two entity nodes. The set of hierarchical edges is denoted as E_h ($E_h \subseteq V_{en} \times V_{en}$). □

Both of the above edge types have attributes and labels. If an edge between two entity nodes evolves, it is replicated, and the new one is filled with the new value. We denote an entity (resp. hierarchical) edge as a tuple $\langle label, Atts \rangle$, where *label* is the type of the relationship and *Atts* is the set of its attributes.

Definition 6. An **evolving edge** (denoted by ) represents a *composition* relationship, i.e. a life-cycle dependency between nodes. It keeps track of the changing attributes extracted as new nodes. The set of evolving edges is denoted as E_{ev} ($E_{ev} \subseteq V_{en} \times V_{ev}$). □

Definition 7. A **versioning edge** (denoted by ) denotes a *directed association* between an evolving node and a value node. Evolving edges are attributed, where each attribute is a key/value pair describing the context for the value node. The set of versioning edges is denoted as E_v ($E_v \subseteq V_{ev} \times V_v$). □

We introduce now two new data entities oriented for analytics queries.

Definition 8. An **analytics hypernode** is an induced subgraph[1,2] grouping an entity node, all its evolving and value nodes, and all edges between them. An analytics hypernode whose entity node is v is denoted as $\Gamma_v = (V, E)$, where $V \subseteq (V_{en} \cup V_{ev} \cup V_v)$ and $E \subseteq (E_{ev} \cup E_v)$. Each node (resp., edge) is part of only one hypernode: $\forall u \in V$ (resp., $e \in E$), $\exists! \Gamma_v \mid u \in \Gamma_v$ (resp., $e \in \Gamma_v$). □

Definition 9. A **class** is a label-based grouping. A class denotes a set of analytics hypernodes whose underlying entity nodes share the same label. □

With the input graph clearly defined, we introduce the graph model as follows.

Definition 10. An **analytics-oriented evolving graph** is a single graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \alpha, \beta, \Lambda, \lambda)$, where:

- $\mathcal{V} = \{V_{en}, V_{ev}, V_v\}$ is the set of nodes.
- $\mathcal{E} = \{E_{en}, E_h, E_{ev}, E_v\}$ is the set of edges.
- $\alpha : (V_{en} \cup V_{ev}) \longrightarrow L_V$ is the function that returns the label for each entity or evolving node, where L_V is the set of labels of entity and evolving nodes.

[1] $G_2 = (V_2, E_2)$ is a subgraph of $G_1 = (V_1, E_1)$ if $V_2 \subseteq V_1$ and $E_2 \subseteq E_1$

[2] G_2 is an induced subgraph of G_1 if all edges between V_2 present in E_1 are in E_2

- $\beta : (E_{en} \cup E_h) \longrightarrow L_E$ is the function that returns the label for each entity or hierarchical edge. L_E is the set of labels of entity and hierarchical edges.
- $\Lambda_{key} : (V_{en} \cup V_v) \longrightarrow Dom(value)$ is the function that returns the value of an attribute given its *key*. Λ is applied only to entity and value nodes. $Dom(value)$ denotes the domain of *value*.
- $\lambda_{key} : (E_{en} \cup E_h) \longrightarrow Dom(value)$ is the function that returns the value of an attribute given its *key*. λ is applied only to entity and hierarchical edges. $Dom(value)$ denotes the domain of *value*. □

With regard to the Epinion network shown in Figure 2, *Product* and *User* are entity nodes, while *Price* is an evolving node attached to the products. Users are linked to each other by entity edges labeled *Trusts* and by hierarchical edges to their *Group*. The price keeps track of the evolution of the product price by region. Multiple ratings of the same product by the same user are recorded. The metamodel of an analytics-oriented evolving graph is shown on Figure 3.

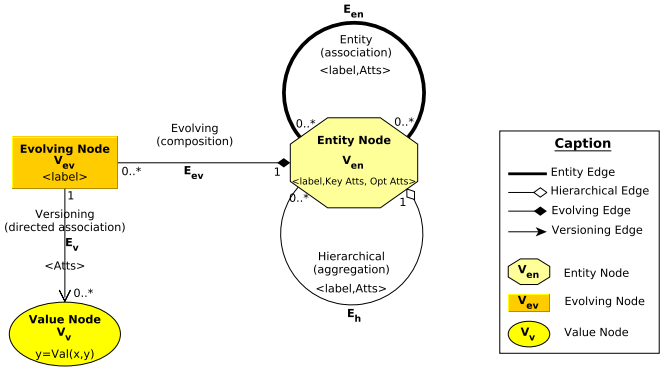


Fig. 3. Analytics-oriented evolving graph metamodel

3 Querying the Graph Model

Selection and projection are two fundamental operators in relational algebra, used to extract a subset of data according to predefined conditions on the data tuples. As their names imply, selection selects the set of tuples of a relation according to a condition on the values of their elements. Projection alters the structure of the relation by removing a subset of the elements of the tuples, and could be used to add elements by combining existing elements and constant values. In this paper, we redefine these two operators for evolving graph analysis. Then, we go a step further by introducing the traversal operation that is essential for graph analysis and provides a finer control of data extraction. However, we do not cover binary operations such as union and intersection of subgraphs, which we consider out of the scope of the model definition.

All the proposed operators perform subgraph extraction operations. Given an input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \alpha_1, \beta_1, \Lambda_1, \lambda_1)$, we denote the produced subgraph as $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', \alpha_2, \beta_2, \Lambda_2, \lambda_2)$ where:

- $\mathcal{V}' \subseteq \mathcal{V}$, and $\mathcal{E}' \subseteq \mathcal{E}$
- $\alpha_2(u) = \alpha_1(u), \forall u \in \mathcal{V}'$
- $\beta_2(e) = \beta_1(e), \forall e \in \mathcal{E}'$
- $\Lambda_2(u) = \Lambda_1(u)$, $\forall u \in V'_{en} \cup V'_v$
- $\lambda_2(u) = \lambda_1(u)$, $\forall e \in E'_{en} \cup E'_h$
- $\Gamma_v \subseteq \mathcal{G}'$, iff $v \in V'_{en}$.

These conditions are valid for the three following operators. For the remainder of the paper, asterisk $(*)$ denotes an optional parameter that could be supplied many times and $|S|$ denotes the cardinality of a set S. We start by examining the selection.

Definition 11. A **selection** $\sigma_{(\text{[NLabel,AttVals]*; [ELabel,AttVals]*})}(\mathcal{G})$ is a *partial*[3] subgraph extraction operation. It is applied on analytics hypernodes and the edges linking their entity nodes. It takes as input a list of the labels ($NLabel$) of the entity nodes V_{en}, underlying the targeted analytics hypernodes, (resp., a list of labels ($ELabel$) of their edges E_{en} and E_h) and the corresponding values of their targeted attributes $AttVals$. A selection returns a partial subgraph $\mathcal{G}'$ of $\mathcal{G}$ where :

- $\alpha_2(u) \in NLabel, \forall u \in V'_{en}$
- $\beta_2(e) \in ELabel, \forall e \in (E'_{en} \cup E'_h)$
- $u \in V'_{en}$ iff $\alpha_1(u) \in NLabel$ and $\exists (k_i, v_i) \in AttVals | \Lambda_{1_{key}}(u) = V_i, \forall key = k_i$
- $e \in E'_{en} \cup E'_h$ iff $\beta_1(e) \in ELabel$ and $\exists (k_i, v_i) \in AttVals \mid \lambda_{1_{key}}(e) = v_i$, $\forall key = k_i$
- $u \in \mathcal{V}'$, iff $\exists \Gamma_v \subseteq \mathcal{G}' \mid u \in \Gamma_v$. □

In the example of Figure 2, $\sigma_{(\text{User; Trusts})}(\mathcal{G}_{epinion})$ detects the communities of users trusting each other. This is accomplished by selecting all analytics hypernodes whose entity nodes are labeled as $User$ and linked by the entity edges labeled $Trusts$. The operation presented above is useful for models presenting *intra-class relationships*, i.e. relationships between analytics hypernodes with the same label. A further step is to perform *inter-class selections*. In this case, selection applies on an heterogeneous set of entity nodes and edges. $\sigma_{(\text{Product; User; Rates})}(\mathcal{G}_{epinion})$ is an inter-class selection. It selects the network comprised of $Rates$ relationships, $Product$ and $User$ analytics hypernodes.

Definition 12. A **projection** $\pi_{(\text{EvLabel, \{ValSet\}})}\{\mathcal{G}, NLabel\}$ is an *induced* subgraph extraction operation. It is applied on a single class of analytics hypernodes, selected through $NLabel$. Other analytics hypernodes remain untouched by this operation. Evolving nodes whose label is not in $EvLabel$ are removed from the targeted analytics hypernodes. It further narrows the range of values in the resulting subgraph by specifying for each versioning edge a key/value map of the requested values, $\{ValSet\}$. A projection returns an induced subgraph $\mathcal{G}'$ where:

[3] G_2 is a partial subgraph of G_1 if a subset of the edges between V_2 from E_1 is in E_2

- $\mathcal{E}' = \mathcal{E} \cap (\mathcal{V}' \times \mathcal{V}')$
- $u \in V'_{en}$ iff $\alpha_1(v) \in NLabel$
- $u \in V'_{ev}$ iff :$\alpha_1(u) \in EvLabel$ and $\exists \Gamma_v \subseteq \mathcal{G}' \mid u \in \Gamma_v$
- $u \in V'_v$ iff: $\exists \Gamma_v \subseteq \mathcal{G}' \mid u \in \Gamma_v$ and $\exists\ e = (u, u_e), e \in E'_v \mid u_e \in \Gamma_v$ and $\exists (k_i, v_i) \in ValSet | \lambda_{1_{key}}(e) = v_i, \forall key = k_i$. □

For the network of Figure 2, $\pi_{(\text{Price, (Store,\{EU,ME\})})}(\mathcal{G}_{epinion}, Product)$ acts only on Product hypernodes. It extracts the subgraph containing as evolving nodes only the Price. And for the Price Value nodes, only those representing EU and ME stores are kept, i.e, the US store is dropped from the resulting graph in all Product analytics hypernodes.

Definition 13. A **traversal** $\mathcal{T}_{(\text{Start,Pattern})}$ is a subgraph extraction operation. A traversal starts from an entity node and follows a guided navigation on the graph according to given rules. Traversal only navigates between entity nodes. However, the navigation rules could be applied at any node or edge to decide whether to include the current entity node, i.e., rules are applied to entity nodes attributes as well as any of their analytics hypernode internal edges and nodes. We refer to these navigation rules as patterns, and hence the subgraph extraction becomes a pattern matching operation. A pattern is a finite sequence of conditions dictating the navigation steps. At each node ($u \in V_{en}$) and edge ($e \in E_{en} \cup E_h$), the next step is defined by an expression applied on the labels or the attributes of the current element. For a step i, a pattern dictates the next elements to visit, and could be a combination of the following statements:

- $u \in V_{en} | \alpha(u) = label_i$, dictates the next nodes based on the supplied label
- $e \in (E_{en} \cup E_h) | \beta(e) = label_i$, dictates the next edges based on the label
- $u \in V_{en} | \Lambda_{key}(u) = val_i$, dictates the next nodes based on the supplied attribute value
- $e \in (E_{en} \cup E_h) | \lambda_{key}(e) = val_i$, dictates the next edges based on the supplied attribute value

$\mathcal{T}$ is applied as follows: $\mathcal{T}_{(Start,Pattern)}(\mathcal{G})$ and returns a partial subgraph of the input graph. Steps are separated by dashes ($-$). □

A typical traversal scenario is the following query, applied on the example shown in Figure 2: *for a user A, return all products she didn't rate, and whose trusted contacts have already rated above four*. Such a query is useful in a recommendation engine. This operation is expressed as follows:
$\mathcal{T}_{(User_A,Pattern)}(\mathcal{G}_{epinion})$, where $Pattern = [e \in E_{en} \mid \beta(e) = Trust] - [*] - [e \in E_{en} \mid \beta(e) = Rates\ \&\ \lambda_{Rating}(e) \geq 4] - [u \in V_{en} \mid (u, User_A) \notin \mathcal{E}]$.

In relational databases, the join operation introduces a heavy workload especially for highly connected tables [3]. In graphs, data is embedded within nodes connected through edges. The cost of running traversals within graphs is much lower than the equivalent joins in relational tables [4]. This makes graphs more suitable for highly connected data compared to relational tables. Moreover, as explained in Section 5, many of current graph databases provide partial or full support ACID properties.

The data structure and operations defined above yield the ground for defining an algebra for evolving graph data. However, this should be further investigated and enriched for the sake of completeness.

4 Multidimensional Graph Analysis

Data warehousing provides a particularly interesting use case for the implementation of our model. The subject-oriented and integrated view of data, provided by the data warehouse, makes it a suitable backbone for common analysis techniques such as reporting, complex querying, and data mining algorithms. We assume that the input graph is designed according to the metamodel defined above. The identification, versioning and insertion of the incoming nodes and edges in the studied graph is done through the ETL phase. The evolving aspect of the graph brings new challenges to the design of the ETL process. Such issues include, but are not limited to, the definition of new entity nodes and labels, the detection of new evolving attributes and the attachment of new values to evolving attributes. Due to space limitations, we have chosen to limit the application of our model to OLAP analysis. In this section, we briefly describe multidimensional concepts using the graph model proposed in the Section 2.

We limit the study to the structures and a subset of the operators defined in the reference algebra described in [5]. However, further research is needed to device new operators uniquely useful in evolving graphs. OLAP analysis enables us to discover hidden facts and relationships between users and products, such as user satisfaction, evolution of trendy categories, and influential groups.

Figure 4 illustrates the proposed multidimensional modeling stack. The physical level switches the focus from elementary nodes and edges to class and inter-class relationships. The logical level encapsulates classes into dimensions and aggregates their relationships. The logical level is similar to ROLAP star schema in that it organizes the studied domain into dimensions and prepares the cube construction. The conceptual level abstracts the graph data using cubes, and proposes user-friendly operations. Physical level has been described in detail in Section 2, and serves as the ground for various analysis techniques. We focus now on the logical and conceptual level, relevant to multidimensional analysis.

4.1 Data Structures

Dimensions. Within our model, a dimension is a tree of classes. A dimension D is defined by a tuple $\langle name, Tree \rangle$, where $name$ denotes the name of the dimension and $Tree$ is the tree of classes. The root of the tree is the highest class in the dimension hierarchy. A class could be involved in only one dimension. Each level is identified by the class $label$.

In our product rating network example, the $Item$ dimension D_{Item} is denoted as $\langle Item, ILevels \rangle$, where $ILevels = [Product \rightarrow Category]$ and $\rightarrow$ denotes the hierarchical relationship between classes. The shift from class to dimension is depicted on Figure 4, where the classes Product and Category are grouped in the same dimension, Item.

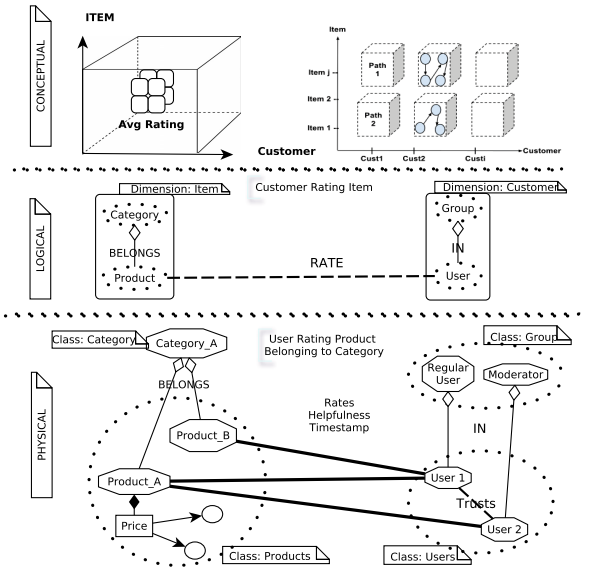


Fig. 4. Multidimensional analysis of an evolving network

Measures. We distinguish two types of measures, (1) informational measures, calculated from the internal attributes of the edges and nodes such as the average rating of a product, and (2) structural measures, result of algorithms performed on the structural properties of the graph, such as centrality metrics. For structural measures, a measure could be a subgraph such as the shortest path, or a numerical value such as the length of the path. Using the evolving nature of the graph, we can retrieve further insights such as the evolution of a product rating, or the evolution of shortest path between users and products. Measures are the metrics used to study a subject. A set of measures showing the data at the same granularity is called a fact. Informational measures are similar to the relational measures. Here we focus on the structural measures, specific for graphs.

Cube. A cube is a set of cells containing measures, and placed in the multidimensional space with regard to a base. A base is the minimal set of dimensions levels that univocally identify a cell within a multidimensional space. A cube $\mathcal{C}$ is defined by a tuple $\langle \mathcal{D}, \mathcal{M} \rangle$, where $\mathcal{D} = \{D_1, D_2, \ldots, D_m\}$ is the set of dimensions, and $\mathcal{M} = \{M_1, M_2, \ldots, M_n\}$ is the set of measures.

Figure 4 shows at the left a cube of informational measures (average rating of items by customers), and at the right a cube of structural measures (shortest path between customers and items).

4.2 Operations

The following representative, but non exhaustive, set of operations provides a high-level abstraction and are applied at the conceptual level to study OLAP cubes.

Slice. Removes a dimension from a cube. $Slice_{[D,Value]}(\mathcal{C})$ operates on the cube $\mathcal{C}$, and returns the subset of cells for which the value of dimension D is set to $Value$. In the cube of shortest path evolution of Figure 4, $Slice_{[Item,id=10]}(\mathcal{C})$ limits the set of studied items to one item whose id=10. This cube is computed after extracting the subgraph of the specific item from the graph of all items, through the selection operator of Section 3, $\sigma_{([\text{Product, (id,10)}]; \text{ User; Rates})}(\mathcal{G}_{epinion})$

Dice. Selects a subset of measures from the cube using a set of conditions on multiple dimensions. This operation is similar to slice, but operates on a range of values of a dimension, and on multiple dimensions at the same time. $Dice_{[User,id=10..30;Item,id=1..15]}(\mathcal{C})$, returns a subcube for which users and items identifiers are limited to the specified ranges.

Roll-Up. Aggregates classes sharing a common hierarchy using the hierarchical edges. This produces a summary graph with new nodes and edges that are not necessarily present in the original graph. Aggregations could be asynchronous. We could for example study relationships between Category and User rather than Category and Group. The roll-up operators performs structural changes to the graph. If the attributes of the elements involved in the aggregation are additive, an overlay is performed and the attributes values are simply incremented. Otherwise, graph summarization techniques such as those discussed on [6–8] could be used to implement the roll-up operation.

5 Related Work

Graph analytics are gaining a lot of momentum in the data management community in recent years. A survey of graph database models according to their data model, data manipulation operators, and integrity constraints is given in [1]. Current graph databases implement different general purpose data models, without a commonly agreed conceptual modeling approach. Neo4j[4] is a centralized graph database implementing the property graph[5] model and guaranteeing ACID constraints. Titan[6] is a distributed graph database implementing property graphs and supporting ACID and partial consistency. Graph querying is made either using traversal-oriented languages such as Gremlin[7], SQL-like languages such as Cypher, or through the database core API. Our model could be implemented using any graph database that supports the input graph described on Section 2. RDF is a widespread data model in the Web community, and could be an implementation candidate for our conceptual model. Pregel [9], and its open source implementation Giraph[8], are BSP graph processing frameworks designed to execute efficiently graph algorithms. Ren et al. [10] proposed an approach to compute

[4] http://neo4j.org/
[5] https://github.com/tinkerpop/blueprints/wikiproperty-graph-model
[6] http://thinkaurelius.github.com/titan/
[7] https://github.com/tinkerpop/gremlin/wiki
[8] http://giraph.apache.org/

graph-specific measures such as shortest path and centrality within a graph with gradually changing edge sets. In [11], the authors present a distributed graph database system for storing and retrieving the state of the graph at specific time points. Both of these two papers are based on the redundancy offered by historical graphs trace. The analysis tasks are limited to graphs specific measures and indices with no querying or multidimensional view of data. Moreover, we consider the historical variation as a specific case of graph evolution scenarios. Related research on versioning was done by the database community. In [12], the authors suggested a conceptual model for evolving object-oriented databases by studying the evolution of objects values, schema and relationships between the objects. Although some concepts are similar, modeling the versioning depends on the data structures specific for each data model. Multidimensional analysis of graphs data has been first proposed in [8]. The authors introduce informational and topological dimensions. Informational aggregations consist of edge-centric snapshot overlaying and topological aggregations consist of merging nodes and edges by navigating through the nodes hierarchy. However, the analysis is limited to homogeneous graphs. GraphCube [7] is applied in single large centralized weighted graph and do not address different edges attributes. Yin et al. [13] introduced a data warehousing model for heterogeneous graphs. They enriched the informational and topological dimensions with the Entity dimension and the Rotate and Stretch operations along with the notion of metapath to extract subgraphs based on edges traversals. However, HMGraph did not provide semantics of OLAP operations on the proposed graph data model. Distributed processing frameworks such as Hive [14] propose data warehousing on top of large volume of data. However, they are considering only the relational model.

6 Conclusions and Future Work

In this paper, we designed a conceptual model for evolving graphs. A plethora of graph database tools is currently developed with multiple management features. However, they do not address the management of evolving networks. Moreover, no common conceptual model for efficient analysis of large evolving networks is agreed. We have proposed our contribution to evolving graph analysis by introducing a well defined conceptual model. We illustrated the model with an application on the multidimensional analysis. However, large networks analysis requires more work to build a complete stack of analysis framework. As future work we plan to proceed in warehousing the evolving graphs. Further fundamental operations such as graph aggregations should be investigated for the evolving graphs. A framework for graph data warehousing should integrate an ETL module, which takes care of matching and merging tasks and provides a graph compliant to the proposed model. An exhaustive study of new OLAP operators in evolving graphs is needed. Current graph querying languages such as Cypher should be extended to support multidimensional queries in an MDX-like fashion. Moreover, distributed processing frameworks should be integrated for large graphs processing.

Acknowledgment. This work has been partially funded by the Wallonia Region in Belgium (Grant FIRST-ENTERPRISES N° 6850).

References

1. Angles, R., Gutierrez, C.: Survey of graph database models. ACM Comput. Surv. 40(1), 1:1–1:39 (2008)
2. Tang, J., Liu, H., Gao, H., Das Sarmas, A.: eTrust: understanding trust evolution in an online world. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 253–261. ACM (2012)
3. Vicknair, C., Macias, M., Zhao, Z., Nan, X., Chen, Y., Wilkins, D.: A comparison of a graph database and a relational database: a data provenance perspective. In: Proceedings of the 48th Annual Southeast Regional Conference, pp. 42:1–42:6. ACM (2010)
4. Sadalage, P.J., Fowler, M.: NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence. Addison-Wesley Professional (2012)
5. Romero, O., Abelló, A.: On the need of a reference algebra for OLAP. In: Song, I.-Y., Eder, J., Nguyen, T.M. (eds.) DaWaK 2007. LNCS, vol. 4654, pp. 99–110. Springer, Heidelberg (2007)
6. Tian, Y., Hankins, R.A., Patel, J.M.: Efficient aggregation for graph summarization. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 567–580. ACM (2008)
7. Zhao, P., Li, X., Xin, D., Han, J.: Graph cube: on warehousing and OLAP multidimensional networks. In: Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data, pp. 853–864. ACM (2011)
8. Chen, C., Yan, X., Zhu, F., Han, J., Yu, P.S.: Graph OLAP: a multi-dimensional framework for graph data analysis. Knowl. Inf. Syst. 21(1), 41–63 (2009)
9. Malewicz, G., Austern, M.H., Bik, A.J., Dehnert, J.C., Horn, I., Leiser, N., Czajkowski, G.: Pregel: a system for large-scale graph processing. In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data, pp. 135–146. ACM (2010)
10. Ren, C., Lo, E., Kao, B., Zhu, X., Cheng, R.: On querying historical evolving graph sequences. Proceedings of the VLDB Endowment 4(11), 726–737 (2011)
11. Khurana, U., Deshpande, A.: Efficient snapshot retrieval over historical graph data. arXiv preprint arXiv:1207.5777 (2012)
12. Andonoff, E., Hubert, G., Parc, A., Zurfluh, G.: Modelling inheritance, composition and relationship links between objects, object versions and class versions. In: Iivari, J., Rossi, M., Lyytinen, K. (eds.) CAiSE 1995. LNCS, vol. 932, pp. 96–111. Springer, Heidelberg (1995)
13. Yin, M., Wu, B., Zeng, Z.: HMGraph OLAP: a novel framework for multidimensional heterogeneous network analysis. In: Proceedings of the 15th International Workshop on Data Warehousing and OLAP, pp. 137–144. ACM (2012)
14. Thusoo, A., Sarma, J.S., Jain, N., Shao, Z., Chakka, P., Anthony, S., Liu, H., Wyckoff, P., Murthy, R.: Hive: a warehousing solution over a map-reduce framework. Proceedings of the VLDB Endowment 2(2), 1626–1629 (2009)

Declarative Expression and Optimization of Data-Intensive Flows

Georgia Kougka and Anastasios Gounaris

Department of Informatics
Aristotle University of Thessaloniki, Greece
{georkoug,gounaria}@csd.auth.gr

Abstract. Data-intensive analytic flows, such as populating a datawarehouse or analyzing a click stream at runtime, are very common in modern business intelligence scenarios. Current state-of-the-art data flow management techniques rely on the users to specify the flow structure without performing automated optimization of that structure. In this work, we introduce a declarative way to specify flows, which is based on annotated descriptions of the output schema of each flow activity. We show that our approach is adequate to capture both a wide-range of arbitrary data transformations, which cannot be supported by traditional relational operators, and the precedence constraints between the various stages in the flow. Moreover, we show that we can express the flows as annotated queries and thus apply precedence-aware query optimization algorithms. We propose an approach to optimizing linear conceptual data flows by producing a parallel execution plan and our evaluation results show that we can speedup the flow execution by up to an order of magnitude compared to existing techniques.

1 Introduction

Data-intensive analytic flows are typically encountered in business intelligence scenarios and are nowadays attracting renewed interest, since they go beyond traditional Extract - Transform - Load (ETL) flows [19,16]. ETLs are a special form of data flows used to populate a data warehouse with up-to-date, clean and appropriately transformed source records. They can be considered as a directed acyclic graph (DAG), similar to scientific and business workflows, capturing the flow of data from the sources to the data warehouse [18]. Next generation business intelligence (BI) involves more complex flows that encompass data/text analytics, machine learning operations, and so on [16]; in this work we target such BI flows.

Our motivation is twofold. Firstly, modern data flows may be particularly complex to be described manually in a procedural manner (e.g., [16]). Secondly, the vast amount of data that such flows need to process under pressing time constraints calls for effective, automated optimizers, which should be capable of devising execution plans with minimum time cost. In this work, we target two correlated goals, namely declarative statement and efficient optimization. Declarative statement of data flows implies that, instead of specifying the exact task ordering, flow designers may need to specify only higher-level aspects,

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 13–25, 2013.

such as the precedence constraints between flow stages, i.e., which task needs to precede other tasks. An example of an existing declarative approach is the Declare language that is based on linear temporal logic [12]. We follow a different approach that bears similarities with data integration mediation systems and allows the flow to be expressed in the form of annotated SQL-like queries.

Regardless of the exact declarative form a flow can be expressed, such declarative approaches are practical only under the condition that the system is capable of taking the responsibility for automatically devising a concrete execution plan in an efficient and dependable manner; this is exactly the role of query optimization in database systems, which also rely on declarative task specifications, and we envisage a similar role in data flow systems as well. Although traditional query optimization techniques cannot be applied in a straightforward manner, we propose an approach to optimizing linear conceptual data flows by producing a parallel execution plan, inspired by advanced query optimization techniques.

The contribution of this work is as follows. We demonstrate how we can express data flows in a declarative manner that is then amenable to optimization in a straight-forward manner. To this end, we use an annotated flavour of SQL, where flow steps are described by input and output virtual relations and annotations are inspired by the binding patterns in [5]. Our approach to declarative statement does not rely on the arguably limited expressiveness of relational algebra in order to describe arbitrary data manipulations, like those in ETLs, and is adequate to describe the precedence constraints between data flow tasks. In addition, we present optimization algorithms for logically linear flows that take into account the precedence constraints so that correctness is guaranteed. As shown in our evaluation, the approach that allows for parallel execution flows may lead to performance improvements up to an order of magnitude in the best case, and performance degradation up to 1.66 times in the worst case compared to the best current technique.

Structure: Sec. 2 presents our approach to declarative statement of data flows with a view to enabling automated optimization. The optimization algorithms for linear flows along with thorough evaluation of performance improvements are in Sec. 3. In Sec. 4 and 5 we discuss related work and conclusions, respectively.

2 Declarative Statement of Data Flows

We map each flow activity[1] to a virtual relation described by a non-changing schema. More specifically, each activity is mapped to a virtual annotated relation $R(A, a, p)$, where (i) R is the task's unique name that also serves as its identifier; (ii) $A = (A_1, A_2, \cdots, A_n)$ is the list of input and output attributes, which are also identified by their names; (iii) a is a vector of size equal to the size of A, such that the i-th element of a is "b" (resp. "f") if the i-th element of A must be *bound* (resp. *free*); and (iv) p is a list of sets, where the j-th set includes the names of the *bound* variables of other virtual relations that must precede $R.A_j$.

[1] The terms flow tasks and activities are used interchangeably.

The notation of the a vector is aligned to the notation of *binding patterns* in [5], and allows us to distinguish between the attributes that need to belong to the input (the *bound* ones) and the new attributes that are produced in the output (*free* attributes). In other words, a binding pattern for a relation R means that the attributes of R annotated with b must be given as inputs when accessing the tuples of R, whereas the attributes annotated by f denote the new attributes derived by the task invocation. For example, the relation $Task1(A : (X, Y, Z),\ a : (bbf),\ p : (\{Task2.X\}\{NULL\}\{NULL\}))$ corresponds to an activity called *Task1*, which needs to be given the values of the X and Y attributes as input and returns a new attribute Z. Attribute X must first be processed by $Task2$. For brevity, this relation can also be written as $Task1(X^{b,Task2}, Y^b, Z^f)$ Additionally, we treat data sources as specific data-producing activities, where all attributes are annotated with f. Linear conceptual flows comprise a single data source.

The following statements hold: (a) The output data items of each flow task are regarded as tuples, the schema of which conforms to the virtual relations introduced above. (b) Data sources are treated as specific data-producing activities, where all attributes are annotated with f. (c) The flow tasks, even when they can be described by standard relational operators (e.g., when they simply filter data), they are always described as virtual relations. (d) The relations can be combined with standard relation operators, such as joins and unions; concrete examples are given in the sequel. (e) For each attribute X that is *bound* in relation R, there exists a relation R', which contains attribute X with a *free* annotation. (f) A task outputing a *free* attribute must precede the tasks that employ the same attributes as *bound* attributes in their schema. (g) Simply relying on b/f annotations is inadequate for capturing all the precedence constraints in ETL workflows, where there may exist a *bound* attribute that is manipulated by a filtering task and also appears in the *bound* grouping values of an aggregate function: in that case, the semantics of the flow may change if we swap the two activities, as also shown in [6]. For that reason, it is always necessary to define the p list of each activity. (h) Although most ETL transformation can be described by static schemas, there may be data flow activities, such as some forms of pivots/unpivots [3] that cannot be mapped to the virtual relations as defined above, because the schema of their output cannot be always defined *a-priori*. (i) Tasks need not correspond to ETL transformation solely; they can also encompass intermediate result storage.

Precedence constraints of a flow form a directed acyclic graph (DAG) G in which there is a node corresponding to each flow task and directed edges from one task to another define the presence of precedence constraint between them. A main goal of the annotations is to fully capture the precedence constraints among tasks. This goal is attained because the edges in the precedence graph can be derived from the p list of each activity and the (f) item above.

2.1 Flow Examples

Linear Flows. Our first example is taken from [18] and is illustrated in Fig. 1. It is a linear flow that applies a set of filters, transformations, and aggregations to a

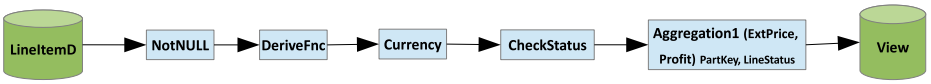


Fig. 1. A linear ETL flow

single table from the TPC-H decision support benchmark. In particular, the flow consists of 5 activities: (i) *NotNull*, which checks the fields *PartKey, OrderKey* and *SuppKey* for NULL values. Any NULL values are replaced by appropriate special values (or the tuple is dropped). (ii) *DeriveFnc*, which calculates a value for the new field *Profit* that is derived from other fields and more specifically by subtracting the values of fields *Tax* and *Discount* from the value in *ExtPrice*. (iii) *Currency*, which alters the fields *ExtPrice, Tax, Discount* and *Profit* to a different currency. (iv) *CheckStat*, which is a filter that keeps only records whose return status is `false`. (v) *Aggregation1*, which calculates the sum of *ExtPrice* and *Profit* fields grouped by values in *PartKey* and *LineStatus*.

All activities can be mapped to virtual relations, and the whole ETL can be modelled as a Select-Project-Join (SPJ) query in order to provide online updates to the view in Fig. 1. It is important to note that the relevant attributes of the source relation, *LineItem*, are annotated as *free* attributes. Also, the $PartKey^{b,CheckStat}$ attribute in the aggregation activity contains a task annotation, which allows us to define that the aggregation must only be performed after the *CheckStat* activity in order to ensure semantic equivalence with the flow in Fig. 1. Finally, in *Aggregation1*, the attributes *PartKey* and *LineStatus* have the same values with *PartKeyGroup* and *LineStatusGroup*, respectively, but the latter are annotated as *free* attributes in order to facilitate manipulation statements that build on the grouped values.

```
Select PartKeyGroup, LineStatusGroup, UpdatedSumProfit,
    UpdatedSumExtPrice
From   LineItem(PartKey^f, OrderKey^f, SuppKey^f, Discount^f, Tax^f, ExtPrice^f, ···) ⋈
       NotNull(PartKey^b, OrderKey^b, SuppKey^b) ⋈
       DeriveFnc(PartKey^b, Profit^f) ⋈
       Currency(PartKey^b, ExtPrice^b, Discount^b, Profit^b, Tax^b) ⋈
       CheckStat(PartKey^b, ReturnStatus^b ⋈
       Aggregation1(PartKey^{b,CheckStat}, LineStatus^b, Profit^b, ExtPrice^b,
                    ReturnStatus^b, LineStatusGroup^f, PartKeyGroup^f,
                    UpdatedSumProfit^f, UpdatedSumExtPrice^f)
Where   LineItem.PartKey = NotNull.PartKey and
        LineItem.PartKey = DeriveFnc.PartKey and
        LineItem.PartKey = Currency.PartKey and
        LineItem.PartKey = CheckStat.PartKey and
        CheckStat.PartKey = Aggregation1.PartKey
```

In the above example there are several precedence constraints that can automatically derived from the annotated query: *LineItem* must precede all other activities, *DeriveFnc* must precede *Currency* and *Aggregation1*, whereas *CheckStat* must precede *Agregation1* as well. Although those constraints seem restrictive, they do not preclude other flow structures, e.g., *CheckStat* to be applied earlier and *Currency* to be applied at the very end to decrease the number of total currency transformations.

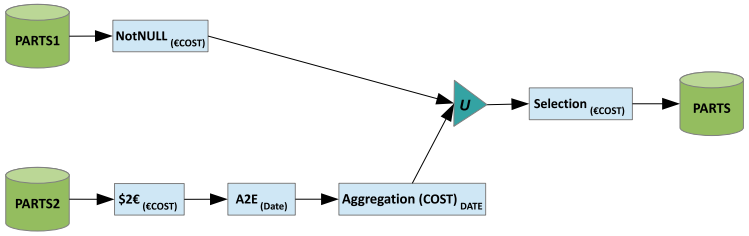


Fig. 2. A more complex ETL flow

More Complex Flows. Fig. 2 shows a more complex flow on top of two real data sources, also taken from [18]. The tasks employed are: (i) *NotNull*, which checks the field *Cost* for NULL values, so that such values are replaced or the tuple is dropped. (ii) *dollar2euro*, which changes the values in *Cost* from dollars to euros. (iii) *A2E*, which alters the format of the field *Date* from american to european. (iv) *Aggregation*, which calculates the sum of costs grouped by date, (v) *Selection*, which filters the (aggregated) cost field according to a user-defined threshold. Although we can describe this flow as a complex nested query, for clarity, we use two SPJ sub-queries. Note that, if the flow contains branches, these can be modeled as separate sub-queries in a similar manner. Also, although the selection task can easily be described by a simple select relational operator, we treat it as a separate relation.

Query I:
WITH Q ($PKEY$, $COST$, $DATE$) AS (
(Select *
From PARTS1($PKEY^f$, $COST^f$, $DATE^f$) $\bowtie$
NotNULL($PKEY^b$, $COST^b$)
Where PARTS1.PKEY = NN.PKEY)
UNION
(Select $PKEY$, $UpdatedAggCOST$, $DATEgroup$
From PARTS2($PKEY^f$,$COST^f$,$DATE^f$) $\bowtie$
dollar2euro($PKEY^b$,$COST^b$) $\bowtie$
A2E($PKEY^b$,$DATE^b$) $\bowtie$
Aggregation($PKEY^b$,$DATE^b$,$COST^b$,$DATEgroup^f$,$UpdatedAggCOST^f$)
Where PARTS2.PKEY = dollar2euro.PKEY and
PARTS2.PKEY = A2E.PKEY and
PARTS2.PKEY = Aggregation.PKEY)
)
Query II:
(Select *
From Q ($PKEY^f$, $COST^f$, $DATE^f$) $\bowtie$
Selection ($PKEY^b$, $COST^b$)
Where Q.PKEY = Selection.PKEY)

Real-World Analytic Flow. The data flow, which is depicted in Fig. 3, shows a real-world, analytic flow that combines streaming free-form text data with structured, historical data to populate a dynamic report on a dashboard [16]. The report combines sales data for a product marketing campaign with sentiments about that product gleaned from tweets crawled from the Web and lists total sales and average sentiment for each day of the campaign. There is a single streaming source that outputs tweets on products and the flow accesses four other static sources through lookup operations.

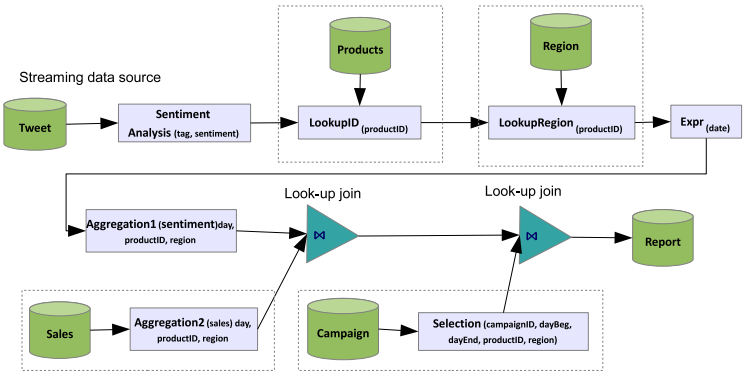


Fig. 3. A real-world analytic flow

The exact flow is as follows. When a tweet arrives as a timestamped string attribute (*tag*), the first task is to compute a single sentiment value in the range [-5 +5] for the product mentioned in the tweet. Then, two lookup operations are performed: the former maps product references in the tweet and the later maps geographic information (latitude and longitude) in the tweet to a geographical region (*region* attribute in the figure). The *Expr* task converts the tweet timestamp to a date. Then, the sentiment values are averaged over each region, product, and date. On a parallel path, the sales data have been rolled up to produce total sales of each product for each region and day. The rollups for sales and sentiment are joined in a pipelined fashion and finally the specific campaign of interest is selected and used to filter the result based on the information of the campaign data store [16]. In this final stage, we consider that the *Sales* and *Campaign* non-streaming sources are hidden behind the *Aggregation2* and *Selection* look-up tasks, respectively. The annotated query that describes this flow is shown below.

```
Select *
From   Tweet(tag^f, timestamp^f) ⋈
       Sentiment_Anal(tag^b, sentiment^f) ⋈
       LookupID(tag^b, productID^f) ⋈
       LookupRegion(tag^b, region^f) ⋈
       Expr(tag^b, timestamp^b, date^f) ⋈
       Aggregation1(tag^b, sentiment^b, productID^b, region^b, date^b,
                    productIdGroup^f, dateGroup^f, regionGroup^f, AvgAggSentiment^f) ⋈
       Aggregation2(productID^f, region^f, date^f, totalAggSales^f) ⋈
       Selection(productID^f, campaignID^f, dayBeg^f, dayEnd^f, region^b)
Where  Tweet.tag = Sentiment_Anal.tag and
       Tweet.tag = LookupID.tag and
       Tweet.tag = LookupRegion.tag and
       Tweet.tag = Expr.tag and
       Tweet.tag = Aggregation1.tag and
       Aggregation1.productID = Aggregation2.productID and
       Aggregation1.region = Aggregation2.region and
       Aggregation1.date = Aggregation2.date and
       Aggregation1.productID = Selection.productID and
       Aggregation1.region = Selection.region
```

2.2 Are Data Flows Queries?

The consensus up to now is that ETL and more generic data flows cannot be expressed as (multi-) queries, due to facts such as the presence of arbitrary manipulation functions that cannot be described by relational operators, and the presence of precedence constraints [4,13]. We agree that data flows cannot be described as standard SQL queries just by regarding manipulation functions as black box user-defined functions (UDFs). Nevertheless, as shown above, we can express data flows in an *SQL-like* manner, where the distinctive features are that (i) data manipulation steps are described through virtual relations instead of relational operators or UDFs on top of real relations; and (ii) the attributes are annotated so that precedence constraints can be derived. Our methodology thus does not suffer from the limitations when mapping a flow to a complex query with as many relations as the original data sources, which loses the information about precedence constraints.

3 Optimization of Linear Flows

Having transformed the flow specification to an annotated query, we can treat the flows as multi-source precedence-aware queries and benefit from any existing optimization algorithms tailored to such settings. We treat flow tasks as black-box operators. Note that we do not have to use multi-way joins regardless of the numerous joins appearing in the SQL-like statements, as in [17]. In this section, we firstly define the cost model, we then propose four optimization algorithms for minimizing the total execution cost in time units, and finally, we investigate the performance benefits.

Our optimization algorithms require that each flow activity is described by the following metadata:

- *Cost* (c_i): We use $c_i = 1/r_i$ to compute response time effectively, where r_i is the maximum rate at which results of invocations can be obtained from the i-th task.
- *Selectivity* (sel_i): it denotes the average number of returned tuples per input tuple for the i-th service. For filtering operators selectivity is always below 1, for data sources and operators that just manipulate the input, it is exactly one, whereas, for operators that may produce more output records for each input record, the selectivity is above 1.
- *Input* (I_i): The size of the input of the i-th task in number of tuples per input data tuple. It depends on the product of the selectivities of the preceding tasks in the execution plan.

Our aim is to minimize the sum of the execution time of each task. As such, the optimal plan minimizes the following formula: $(I_i c_i + I_2 c_2 + ... + I_n c_n)$.

In the following, we present our optimization approaches. Due to lack of space, we present only the main rationale.

PGreedy: The rationale of the *PGreedy* optimization algorithm is to order the flow tasks in such a way that the amount of data that is received by expensive

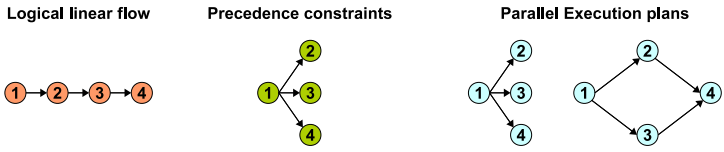


Fig. 4. A single linear conceptual data flow (left), along with its precedence constraints (middle) and two logically equivalent parallel execution plans (right)

tasks is reduced because of preceding filtering activities that prune the input dataset. Its main distinctive feature is that it allows for parallel execution plans, as shown in Fig. 4, where on the left part of the picture, a linear flow and its precedence constraints are depicted, while on the right two equivalent parallel execution plans of the same flow are presented (which both preserve all the precedence constraints). More specifically, depending on the selectivity values, the optimal execution plan may dispatch the output of an activity to multiple other activities in parallel, or place them in a sequence. To this end we adapt the algorithm in [17] with the difference that instead of considering the cost $I_i c_i$ in each step, we consider the $(1 - sel_i)(I_i c_i)$. The latter takes into account the selectivity of the next service to be appended in the execution plan and not only the selectivity of the preceding services. We refer the reader to [17] for the rest of the details. The complexity is $O(n^5)$ in the worst case.

Swap: The *Swap* algorithm compares the cost of the existing execution plan against the cost of the transformed plan, if we swap two adjacent tasks provided that the constraints are always satisfied. We perform this check for every pair of adjacent tasks. *Swap* is proposed in [15], where, to the best of our knowledge, the most advanced algorithm for optimizing the structure of data flows is proposed. The complexity of the *Swap* algorithm is $O(n^2)$.

Greedy: Greedy algorithm is based on a typical greedy approach by adding the activity with the maximum value of $(1 - sel_i)(I_i c_i)$, which meets the precedence constraints. The time complexity of *Greedy* algorithm is $O(n^2)$. It bears similarities with the *Chain* algorithm in [21]; latter appends the activity that minimizes $I_i c_i$. Similarly to *Swap* and contrary to *PGreedy*, it builds only linear execution plans.

Partition: The *Partition* optimization algorithm forms clusters with activities by taking into consideration their availability. Specifically, each cluster consists from activities that their prerequisites have been considered in previous clusters. After building the clusters, each cluster is optimized separately by checking each permutation of cluster tasks. Like *Greedy*, it was first proposed for data integration systems, and the details are given in [21]. *Partition* runs in $O(n!)$ time in the worst case, and is inapplicable if a local cluster contains more than a dozen of tasks.

3.1 Experiments

In our experiments, we compare the performance of the afore-mentioned algorithms. The data flows considered consist of $n = 5, 10, 25, 50, 100$ activities and

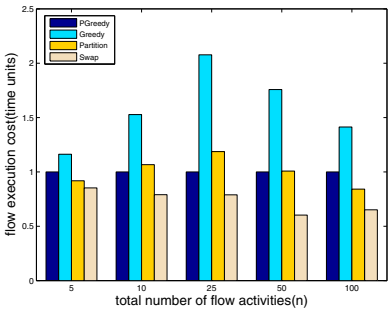


Fig. 5. Performance when $sel \in [0, 2]$, $cost \in [1, 10]$ and 25% prec. constraints

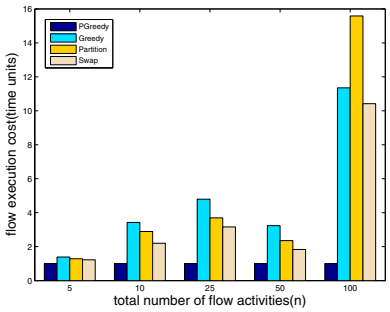


Fig. 6. Performance when $sel \in [0.5, 2.5]$, $cost \in [1, 10]$ and 25% prec. constraints

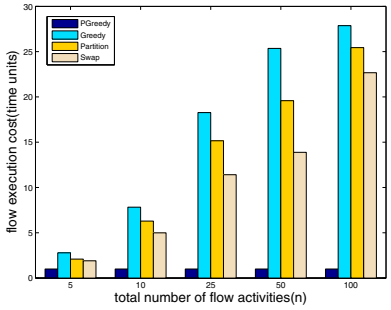


Fig. 7. Performance when $sel \in [1, 3]$, $cost \in [1, 10]$ and 25% prec. constraints

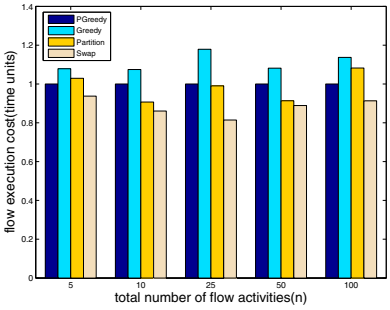


Fig. 8. Performance when $sel \in [0, 2]$, $cost \in [1, 10]$ and 50% prec. constraints

we experiment with 6 combinations of 3 selectivity value ranges and 2 sets of constraint probabilities. The cost value range is the same for all the sets of experiments: $c_i \in [1, 10]$. The results correspond to the average of the data flow response time in 20 runs after removing the lowest and highest values to reduce the standard deviation. In each run, the exact selectivity, cost values and the constraints for each task are randomly generated.

In the first experiment, the selectivity values in each run are randomly generated so that $sel \in [0, 2]$ (thus only half of the tasks are selective) and $cost \in [1, 10]$ with 25% probability of having precedence constraints between two activities. The normalized results are shown in Fig. 5. A general observation in all our experiments is that *Swap* consistently outperforms *Greedy* and *Partition*. From Fig. 5, we can observe that *Swap* outperforms *PGreedy* as well. For $n = 50$, Swap is 1.66 times faster. However, as less activities are selective, *PGreedy* yields significantly lower cost than *Swap*. As shown in Figs. 6 and 7, those performance improvement may be up to 22 times (one order of magnitude).

In the following experiment, we increase the probability of having a precedence constraint between two activities. The more the constraints, the narrower the space for optimizations. The results are presented in Figs. 8-10, which follow the

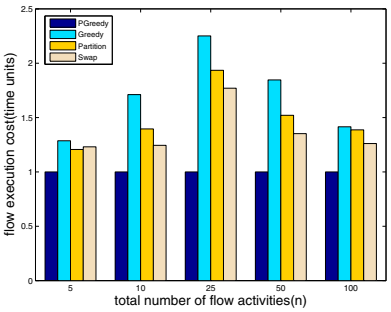


Fig. 9. Performance when $sel \in [0.5, 2.5]$, $cost \in [1, 10]$ and 50% prec. constraints

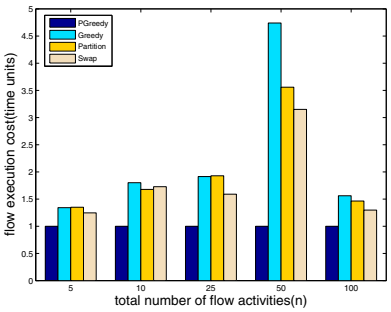


Fig. 10. Performance when $sel \in [1, 3]$, $cost \in [1, 10]$ and 50% prec. constraints

same pattern as above. In the worst case, *Swap* is 1.23 times faster than *PGreedy*, and, in the best case, *PGreedy* is 3.15 times faster. The general conclusions drawn is that *Greedy* and *Partition* are never the optimal choices, and *PGreedy* outperforms *Swap* if less than half of the tasks are selective.

Regarding the time needed for the optimizations, even when $n = 100$, the time for running *PGreedy* and *Partition* is approximately a couple of seconds using a machine with an Intel Core i5 660 CPU with 6GB of RAM. Thus it can be safely considered as negligible.

4 Related Work

Modern ETL and flow analysis tools, such as Pentaho's platform[2], do not support declarative statement of flows and automated optimizations of their structure. Declare is an example of a declarative flow language [12]; contrary to our proposal, it is based on linear temporal logic and can be used only through a graphical interface in the context of Yawl[3]. Declare can capture precedence constraints, and, as such, may stand to benefit from the optimizations proposed in this work, but does not perform any optimizations in its own right.

The potential of data management solutions to enhancing the state-of-the art in workflow management has been identified since mid 2000s. An example of strong advocates of the deeper integration and coupling of databases and workflow management systems has appeared in [14]. Earlier examples of developing data-centric techniques of manipulating workflows include the prototypes described in [7,11,9], which allow workflow tasks to be expressed on top of virtual data tables in a declarative manner but do not deal with optimization, although they can be deemed as enabling it. Other declarative approaches to specifying workflows, such as [1,22], are not coupled with approaches to capturing precedence constraints and optimizing the flow structure either.

[2] `http://www.pentaho.com/`
[3] `http://www.yawlfoundation.org/`

Data management techniques have been explored in the context of ETL workflows for data warehouses in several proposals, e.g., [4,13,15]. In [15], the authors consider ETL workflows as states and use transitions to generate new states in order to navigate through the state space. The main similarity with our work is the mapping of workflow activities to schemata, which, however, are not annotated and thus inadequate to describe precedence constraints on their own. Focusing on the physical implementation of ETL flows, the work in [18] exploits the logical-level description combined with appropriate cost models, and introduces sorters in the execution plans. In [16], a multi-objective optimizer that allows data flows spanning execution engines is discussed.

Another proposal for flow structure optimization has appeared in [20], which decreases the number of invocations to the underlying databases through task merging. In [10], a data oriented method for workflow optimization is proposed that is based on leveraging accesses to a shared database. In [6], the optimizations are based on the analysis of the properties of user-defined functions that implement the data processing logic. Several optimizations in workflows are also discussed in [2]. Our optimization approach shown in Section 3 is different from those proposals in that it is capable of performing arbitrary correct task reordering. In our previous work, we employ query optimization techniques to perform workflow structure reformations, such as reordering or introducing new services in scientific workflows [8].

5 Conclusions

As data flows become more complex and come with requirements to deliver results under pressing time constraints, there is an increasing need for more efficient management of such flows. In this work, we focused on data-intensive analytic flows that are typically encountered in business intelligence scenarios. To alleviate the burden to manually design complex flows, we introduced a declarative way to specify such flows at a conceptual level using annotated queries. A main benefit from this approach is that the flows become amenable to sophisticated optimization algorithms that can take over the responsibility for optimizing the structure of the data flow while taking into account any precedence constraints between flow activities. We discuss optimization of linear conceptual data flows, and our evaluation results show that we can speedup the flow execution by up to an order of magnitude if we consider parallel execution plans. Our future work includes the deeper investigation of optimization algorithms to non-linear conceptual flows and the coupling of optimization techniques that reorder tasks with resource scheduling and allocation in distributed settings.

Acknowledgements. This research has been co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: Thales. Investing in knowledge society through the European Social Fund.

References

1. Bhattacharya, K., Hull, R., Su, J.: A data-centric design methodology for business processes. In: Handbook of Research on Business Process Modeling. ch. 23, pp. 503–531 (2009)
2. Böhm, M.: Cost-based optimization of integration flows. PhD thesis (2011)
3. Cunningham, C., Graefe, G., Galindo-Legaria, C.A.: Pivot and unpivot: Optimization and execution strategies in an rdbms. In: VLDB, pp. 998–1009 (2004)
4. Dayal, U., Castellanos, M., Simitsis, A., Wilkinson, K.: Data integration flows for business intelligence. In: Proc. of the 12th Int. Conf. on Extending Database Technology: Advances in Database Technology, EDBT, pp. 1–11. ACM (2009)
5. Florescu, D., Levy, A., Manolescu, I., Suciu, D.: Query optimization in the presence of limited access patterns. In: Proc. of the 1999 ACM SIGMOD Int. Conf. on Management of Data, pp. 311–322 (1999)
6. Hueske, F., Peters, M., Sax, M., Rheinländer, A., Bergmann, R., Krettek, A., Tzoumas, K.: Opening the black boxes in data flow optimization. PVLDB 5(11), 1256–1267 (2012)
7. Ioannidis, Y.E., Livny, M., Gupta, S., Ponnekanti, N.: Zoo: A desktop experiment management environment. In: Proc. of 22th Int. Conf. on Very Large Data Bases VLDB, pp. 274–285 (1996)
8. Kougka, G., Gounaris, A.: On optimizing workflows using query processing techniques. In: Ailamaki, A., Bowers, S. (eds.) SSDBM 2012. LNCS, vol. 7338, pp. 601–606. Springer, Heidelberg (2012)
9. Liu, D.T., Franklin, M.J.: The design of GridDB: A data-centric overlay for the scientific grid. In: VLDB, pp. 600–611 (2004)
10. Minglun, R., Weidong, Z., Shanlin, Y.: Data oriented analysis of workflow optimization. In: Proc. of the 3rd World Congress on Intelligent Control and Automation 2000, vol. 4, pp. 2564–2566. IEEE Computer Society (2000)
11. Narayanan, S., Catalyrek, U.V., Kurc, T.M., Zhang, X., Saltz, J.H.: Applying database support for large scale data driven science in distributed environments. In: Proc. of the 4th Workshop on Grid Computing (2003)
12. Pešić, M., Bošnački, D., van der Aalst, W.M.P.: Enacting declarative languages using LTL: Avoiding errors and improving performance. In: van de Pol, J., Weber, M. (eds.) Model Checking Software. LNCS, vol. 6349, pp. 146–161. Springer, Heidelberg (2010)
13. Sellis, T.K., Simitsis, A.: Etl workflows: From formal specification to optimization. In: Ioannidis, Y., Novikov, B., Rachev, B. (eds.) ADBIS 2007. LNCS, vol. 4690, pp. 1–11. Springer, Heidelberg (2007)
14. Shankar, S., Kini, A., DeWitt, D.J., Naughton, J.: Integrating databases and workflow systems. SIGMOD Rec. 34, 5–11 (2005)
15. Simitsis, A., Vassiliadis, P., Sellis, T.K.: State-space optimization of etl workflows. IEEE Trans. Knowl. Data Eng. 17(10), 1404–1419 (2005)
16. Simitsis, A., Wilkinson, K., Castellanos, M., Dayal, U.: Optimizing analytic data flows for multiple execution engines. In: Proc. of the 2012 ACM SIGMOD Int. Conf. on Management of Data, pp. 829–840 (2012)
17. Srivastava, U., Munagala, K., Widom, J., Motwani, R.: Query optimization over web services. In: Proc. of the 32nd Int. Conference on Very Large Data Bases VLDB, pp. 355–366 (2006)
18. Tziovara, V., Vassiliadis, P., Simitsis, A.: Deciding the physical implementation of ETL workflows. In: Proc. of the ACM 10th Int. Workshop on Data Warehousing and OLAP DOLAP, pp. 49–56 (2007)

19. Vassiliadis, P., Simitsis, A.: Near real time ETL. In: New Trends in Data Warehousing and Data Analysis, pp. 1–31 (2009)
20. Vrhovnik, M., Schwarz, H., Suhre, O., Mitschang, B., Markl, V., Maier, A., Kraft, T.: An approach to optimize data processing in business processes. In: VLDB, pp. 615–626 (2007)
21. Yerneni, R., Li, C., Ullman, J.D., Garcia-Molina, H.: Optimizing large join queries in mediation systems. In: Beeri, C., Bruneman, P. (eds.) ICDT 1999. LNCS, vol. 1540, pp. 348–364. Springer, Heidelberg (1998)
22. Zhao, Y., Dobson, J., Foster, I., Moreau, L., Wilde, M.: A notation and system for expressing and executing cleanly typed workflows on messy scientific data. SIGMOD Rec. 34, 37–43 (2005)

Tabular Web Data: Schema Discovery and Integration

Prudhvi Janga and Karen C. Davis

University of Cincinnati, Cincinnati, Ohio, USA
jangapi@mail.uc.edu, karen.davis@uc.edu

Abstract. Web data such as web tables, lists, and data records from a wide variety of domains can be combined for different purposes such as querying for information and creating example data sets. Tabular web data location, extraction, and schema discovery and integration are important for effectively combining, querying, and presenting it in a uniform format. We focus on schema generation and integration for both a static and a dynamic framework. We contribute algorithms for generating individual schemas from extracted tabular web data and integrating the generated schemas. Our approach is novel because it contributes functionality not previously addressed; it accommodates both the static and dynamic frameworks, different kinds of web data types, schema discovery and unification, and table integration.

Keywords: Web tables, web lists, schema generation, schema discovery, schema integration, schema integration framework.

1 Introduction

With the continuing explosive growth of the web, a wealth of information has become available online. To access this information, individual users can either use search engines or navigate to a particular website following links. The former method returns links to vast amounts of data in seconds while the latter could be tedious and time consuming. The presentation of results using the former method is usually a web page with links to actual web data sources (or websites). The latter method takes the user to the actual web data source itself. Using the two most popular forms of web data presentation, web data cannot be easily queried, manipulated, and analyzed even though it is publicly and readily available. On the other hand, the web has become a primary source of important information for many companies that build web-based analytical and decision support systems, often referred to as web data warehouses. Traditional data warehousing technologies and architectures are ill-adapted to warehouse web data which is extremely diverse (multiformat data) and not just numeric or symbolic.

Web data is available in different formats such as structured, semi-structured, and unstructured. Heterogeneity of web data presents major challenges in the integration and representation of web data to end users. In this paper we focus on structured web data such as web tables, lists, and data records that can be loaded into relational databases after schema discovery and integration. We refer to this web data as tabular web data from here on. Uniform presentation of tabular web data involves steps such as location of tables or lists, extraction of located data, and schema discovery and

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 26–33, 2013.

integration followed by querying the extracted data. Efficiently extracted, analyzed, and queried tabular web data from a wide variety of domains could be used for answering factual queries, providing contextual search results (to provide search results that are contextually meaningful), and in other semantic services.

Consider a user search for keywords "U.S. Presidents" where the results are in the form of links to two or more related tables. Fig. 1 shows U.S presidents' information available on two different web sites[1] while Fig. 2 shows the same information from different web sources consolidated together into tables instead of a list of links to different web sites that contain the same information. However, creating comprehensive tables of information on U.S. presidents with attributes such as date of birth, term, name, and number of times elected is time-consuming and cumbersome with little support from existing database systems [CHK09].

Fig. 1. Snapshots of two websites showing information about U.S. Presidents

S. No.	President's Name	Birth Date	From-To	Terms Served	Vice President	Birth Place
1.	George Washington	02/22/1732	1789-1797	2	John Adams	Virginia
44.	Barack Obama	08/04/1961	2009-	2	Joseph Biden	Hawaii

Vice President	Born in	Birth Place
John Adams	30/10/1735	Massachusetts
Joseph Biden	11/20/1942	Pennsylvania

State	Capital	Population
Virginia	Richmond	8,096,604
Hawaii	Honolulu	1,374,810
Pennsylvania	Harrisburg	12,742,886
Massachusetts	Boston	6,587,536

Fig. 2. Tabular, Consolidated Presentation of Web Data for U.S Presidents

We observe that there has been extensive research focusing on the location and extraction of tables but not a lot of work has been done on schema discovery, schema unification, and integration of web tables. Most of the approaches stop at schema discovery or discuss table integration without schema discovery and schema unification in a web data context. We contend that schema discovery and schema unification are not only important for the integration of web tables but also useful in other applications such as schema auto-complete (schema auto-complete applications are designed to assist novice database users when designing a relational schema [CHW08]), context searching, example schema generation, schema mining, and synonym discovery.

[1] http://en.wikipedia.org/wiki/List_of_Presidents_of_the_United_States,_sortable_by_previous_experience; http://www.enchantedlearning.com/history/us/pres/list.shtml; date accessed 3/13/13

We concentrate on metadata recovery and on reverse engineering the schema from extracted web data. Section 2 describes the proposed framework for uniform presentation of consolidated web tables. In Section 3 we discuss our approach and the algorithm for schema discovery of tabular/list/template-based web data. Section 4 describes the schema unification algorithm proposed for merging individual schemas to obtain a final schema. The result of the unification technique is a merged final schema that can be used in the frameworks shown in Fig. 3 to achieve uniform presentation of tabular/list/template-based web data. We present an example illustrating the implementation of the algorithms in Section 5, and conclude with a discussion on the future work in Sections 6.

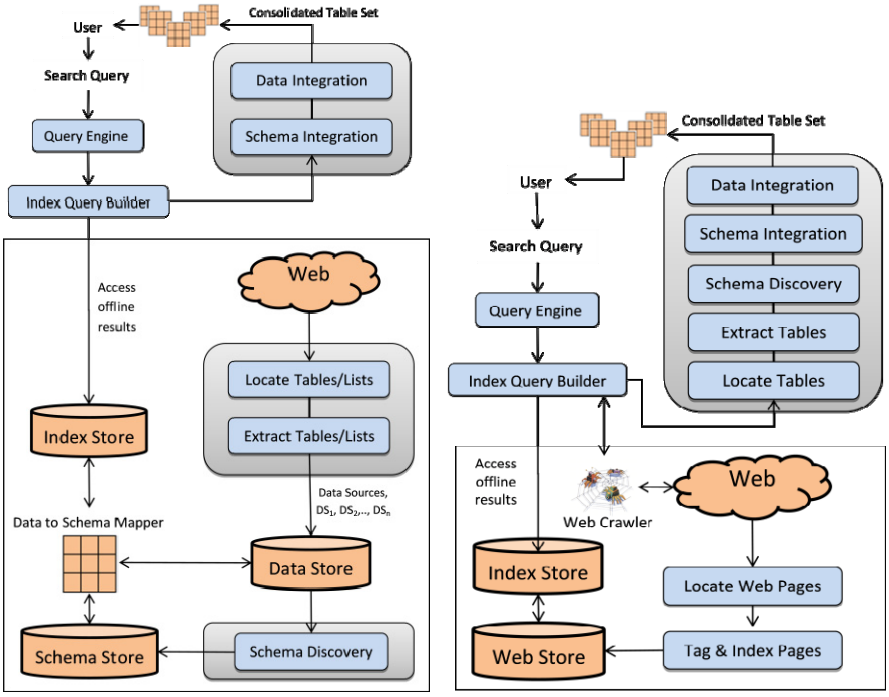


Fig. 3. Static and Dynamic Frameworks for Uniform Presentation of Tabular Web Data

2 Framework

We categorize schema discovery research efforts into two different frameworks in Table 1. In an on-demand (dynamic) framework, the query engine retrieves the indexed web page links that satisfy given search criteria. The retrieved web links are then fed into a conditional crawler that retrieves these web pages of interest. However, in a static framework, all the web pages are retrieved, indexed, and stored in web data stores ahead of time for future querying purposes. Previous approaches use one of the two approaches but not both. The modules common to both frameworks are discussed below.

2.1 Location of Tables/Lists

Detecting tables in web documents is algorithmically non-trivial; even when the system can tell that the web page is of interest for the given application, finding tables or lists of interest in that web page is often difficult [ENX01]. Other challenges include tables that span across multiple web pages, tables without table tags (also known as lists or data records), tables present in deep web databases, lists that are separated by visual decorators, excluding tables that are not relational in nature, and also identifying repetitive patterns to detect lists. Because many of the challenges have been addressed [H01, ETL05, CHZ08, CMH09, MTH+09, GS09], we use existing techniques to create a prototype system that locates tabular web data.

2.2 Extraction of Located Tables/Lists

Table extraction algorithms must be intelligent enough to understand the internal cell structure of the located tables because a located table can have split cells, errors, omissions, constraints, and even ill-formatted HTML [H01]. Extraction of list data has its own challenges such as header detection, delimiter detection, and record alignment [ETL05]. We use existing techniques to create a prototype data extraction system [ETL05, CHZ08, EMH11, MTH+09] in which we also address two additional issues: 1) the identification and extraction of linked tables or lists within a web page, and 2) data type detection of columns in tables/lists.

2.3 Schema Discovery and Integration Process

Schema discovery and integration is one of the most important steps to achieve a conceptual model of the extracted data which in turn makes the querying process systematic and much more efficient. Some on-demand approaches combine schema discovery with the querying process while others do it in two different steps. The schema discovery process usually generates a small schema for the extracted tables present within each web page, followed by a schema unification process that attempts to combine all the schemas from different web pages into one. There are several challenges in the process of schema discovery and unification as it involves identifying outliers, detecting related tables, recovering metadata, and possibly creating new tables. Other challenges include schema discovery of nested lists, gathering and understanding metadata information such as domain overlap, breadth of data, level of detail and context to better understand the tables.

Current techniques (summarized in Table 1) discuss the integration of web data either using techniques that do not include schema discovery and unification at all or using techniques that involve metadata recovery but are not complete. They do not address issues such as primary key detection, data type detection, context determination, table translation, and schema unification, as addressed in our work. To the best of our knowledge, no other research effort includes all of the features identified in the first column as our approach does.

Table 1. Comparison of Related Work

Features /Papers	*[ETL05]*	*[ELN06]*	*[CHW08] [CHZ08]*	*[MFH08]*	*[EMH11]*	*[MTH+09]*	*[GS09]*	*[WWW+12]*	*[SFG+12]*	*Our Research*
Framework type(s)	Static	N/A	Static	Dynamic	N/A	N/A	Static	Static	N/A	Static, Dynamic
Source(s) of web data discussed	HTML tables	Tables from web, PDF & doc	HTML tables	HTML tables	HTML lists	HTML data records	HTML lists	HTML tables	HTML tables	HTML tables, lists, & data records
Location of tables	●	●	●	prototype	○	●	●	●	○	Prototype
Extraction of tables	●	○	●	prototype	●	●	●	●	○	Prototype
Schema discovery/relation detection	●	○	incomplete	○	○	○	○	●	●	●
Schema unification	○	○	○	○	○	○	○	○	○	●
Table integration	○	○	○	○	○	○	●	○	○	●
Querying for data discussed	●	○	●	●	○	○	●	○	○	●

The symbol ● indicates "yes" while the symbol ○ indicates "no."

3 Schema Discovery Using Partial Schema Generation

Partial schema generation (PSG) creates a schema from all the tables/lists that have been located and extracted from the same web page. The tables from each of the web pages are extracted into objects called Data Source Objects (DSOs). For all the tables in each DSO, the operations can be grouped into three phases: (1) the data type detection phase, (2) the primary key detection phase, and (3) the relationship identification phase. Detecting the data type of each of the columns present in a given table involves determining all the possible data types for a given column and the best fit using frequency counts and making assignments based on the following precedence order: text, float & date/time, integer & time, and Boolean.

In the relationship identification phase we identify relationships among different tables present in a web page if there is more than one extracted table present. Since we already have the data type information as well as the primary key information from the previous phases, we can use that information and other techniques to identify relationships between two tables. Tables can be identical (based on similarity); we employ a synonym finding technique [CHW08], an extraction ontology technique [ETL05], and the column data type information detected in the previous phase. It generates a Boolean result stating whether the two columns can be treated the same or not. Another relationship is that one table can be composed from several other tables, and another is when composite keys overlap. Once this phase is complete we have all the metadata information and a partial schema for each DSO under consideration. The next section introduces our approach to schema unification.

4 Schema Unification

Our schema unification technique merges partial schemas generated from each of the web pages (or DSOs) into a global schema to achieve an integrated final schema. To merge two partial schemas, the relationships between the two schemas need to be

identified. Relationships between the partial schemas of any two DSOs are derived using an *Inter-schema Relationships Detection* algorithm. This algorithm uses an operator that uses a set of heuristics to determine relationships present between the two partial schemas.

Once the inter-schema relationships have been defined, we proceed to schema unification process. We first sort all the DSOs under consideration in ascending order of search result ranks so that most relevant results are given the highest priority. We then identify the outlier tables and drop them. Outliers are defined as the tables that do not have any derived relationships and are excluded from the unification process. After this step we use a schema unification operator to combine all the individual schemas into a merged final schema. This operator adds eligible tables to the final schema using the following heuristics:

- Merge tables that have the same number of columns and similar column names for all columns, reducing the number of tables in a merged schema.
- Identify and merge a table that is equivalent (similar columns) to a combination of multiple tables, helping to achieve a compact merged schema.
- Eliminate a table from being added into merged schema if a table or set of tables with similar columns already exists in the merged schema, helping to remove redundancies from a merged schema.
- Create a new table using columns missing from a merged schema table and add the table to the merged schema, making sure that important attributes/column data is not lost in schema unification.
- Create a new table corresponding to part of a table that appears repeatedly in several schemas (according to a user-defined threshhold.)

The last step is to clean the final schema. Since some new tables might have been added and some relationships might have changed, all the tables in the final schema are passed through the phases of primary key detection and relationship identification. Cleanup also includes finding and eliminating duplicate tables and subset tables, and ensuring that the schema is in the user's preferred normal form.

5 Implementation

We implement the algorithms proposed to generate and unify schemas using Microsoft Visual Basic and .Net framework 4. To explain the schema unification process we chose two data source objects whose partial schemas are shown in Figs. 4 and 5. The unification algorithm iterates through each table of the data source objects to check if a similar table is already included in the final schema. For example, *Web Table 1* in Fig. 4 is equivalent to *Web Table B* in Fig. 5 because they both have the same column count and also similar names and data types. Equivalent tables can replace each other. Hence, as *Web Table 1* is added to the final schema, *Web Table B* is replaced by *Web Table 1*. *Web Table C* is a part of *Web Table 2* as the column count of *Web Table C* is less than the column count in *Web Table 2* and all the column names are similar, so it to be eliminated from further consideration. The algorithm iterates through all the tables to generate a final unified schema shown in Fig. 6.

Web Table 1:	**S. No.**	**President's Name**	**Birth Date**	**Vice President**
Web Table 2:	**President's Name**	**From-To**	**Terms Served**	**Birth Place**

Fig. 4. Example Partial Schema 1

Web Table A:	**State**	**Capital**	**Population**	
Web Table B:	**#**	**President**	**Date of Birth**	**Vice President**
Web Table C:	**President's Name**	**From-To**		

Fig. 5. Example Partial Schema 2

Final Table 1:	**S. No.**	**President's Name**	**Birth Date**	**Vice President**
Final Table 2:	**President's Name**	**From-To**	**Terms Served**	**Birth Place**
Final Table 3:	**State**	**Capital**	**Population**	

Fig. 6. Example Merged Schema

6 Conclusions and Future Work

We present static and dynamic frameworks for relational web data discovery in different steps such as location, extraction, schema discovery, integration, and querying. We propose the Partial Schema Generation technique to discover a schema from tabular web data and a schema unification algorithm to integrate the discovered partial schemas. The main strengths of our approach are that (1) it is modular so that it supports either static or dynamic processing, and (2) it works well for heterogeneous tabular web data.

We have several research efforts in progress that space limitations do not allow us to include here. We have developed a heuristic algorithm for schema unification to improve performance compared to the exhaustive approach discussed here. We are currently researching schema quality metrics and schema matching/merging techniques. We have conducted some preliminary experiments using datasets derived from Google's web tables and other corporate web data sources. We have collected more than 30 manual schema integration efforts over our sample data in order to compare the quality of our algorithm's results to the efforts of human designers. We expect to obtain additional experimental results as evidence of the validity and scalability of our approach; we plan to release our datasets to support reproducibility of our results as well as further research when this is complete.

Future work can be done on improving existing tabular web data location and extraction techniques, conducting more experiments and also enhancing the schema unification algorithms using contextual information to extend schema knowledge from systems such as Probase [WLW+12] or those that use ontologies.

References

[CHK09] Cafarella, M.J., Halevy, A.Y., Khoussainova, N.: Data Integration for the Relational Web. In: Proceedings of the 35th International Conference on Very Large Data Bases (VLDB 2009), Lyon, France, August 24-28, pp. 1090–1101 (2009)

[CHM11] Cafarella, M.J., Halevy, A.Y., Madhavan, J.: Structured Data on the Web. Communications of the ACM (CACM) 54(2), 72–79 (2011)

[CHW08] Cafarella, M.J., Halevy, A.Y., Wang, D.Z., Wu, E., Zhang, Y.: WebTables: Exploring the Power of Tables on the Web. In: Proceedings of the 34th International Conference on Very Large Data Bases (VLDB 2008), Auckland, New Zealand, August 23-28, pp. 538–549 (2008)

[CHZ08] Cafarella, M.J., Halevy, A.Y., Zhang, Y., Wang, D.Z., Wu, E.: Uncovering the Relational Web. In: Proceedings of the 11th International Workshop on Web and Databases (WebDB 2008), Vancouver, BC, Canada (June 13, 2008)

[CMH09] Cafarella, M.J., Madhavan, J., Halevy, A.Y.: Web-scale Extraction of Structured Data. ACM SIGMOD Record 37(4), 55–61 (2009)

[ELN06] Embley, D.W., Lopresti, D.P., Nagy, G.: Notes on Contemporary Table Recognition. In: Bunke, H., Spitz, A.L. (eds.) DAS 2006. LNCS, vol. 3872, pp. 164–175. Springer, Heidelberg (2006)

[EMH11] Elmeleegy, H., Madhavan, J., Halevy, A.Y.: Harvesting Relational Tables from Lists on the Web. In: Proceedings of the VLDB Endowment, vol. 20(1), pp. 209–226 (2009)

[ENX01] Embley, D.W., Ng, Y.-K., Xu, L.: Recognizing Ontology-applicable Multiple-record Web Documents. In: Kunii, H.S., Jajodia, S., Sølvberg, A. (eds.) ER 2001. LNCS, vol. 2224, pp. 555–570. Springer, Heidelberg (2001)

[ETL05] Embley, D.W., Tao, C., Liddle, S.W.: Automating the Extraction of Data from HTML Tables with Unknown Structure. Data and Knowledge Engineering 54(1), 3–28 (2005)

[GS09] Gupta, R., Sarawagi, S.: Answering Table Augmentation Queries from Unstructured Lists on the Web. In: Proceedings of the VLDB Endowment, vol. 2(1), pp. 289–330 (2009)

[H01] Hurst, M.: Layout and Language: Challenges for Table Understanding on the Web. In: Proceedings of the International Workshop on Web Document Analysis (WDA 2001), Seattle, Washington, USA, pp. 27–30 (September 8, 2001)

[MFH08] Mergen, S., Freire, J., Heuser, C.: Mesa: A Search Engine for Querying Web Tables (2008), `http://www.scholr.ly/paper/1328437/mesa-a-search-engine-for-querying-web-tables`

[MTH+09] Miao, G., Tatemura, J., Hsiung, W.-P., Sawires, A., Moser, L.E.: Extracting Data Records from the Web Using Tag Path Clustering. In: Proceedings of the 18th International ACM Conference on World Wide Web (WWW 2009), Madrid, Spain, April 20-24, pp. 981–990 (2009)

[SFG+12] Sarma, A.D., Fang, L., Gupta, N., Halevy, A.Y., Lee, H., Wu, F., Xin, R., Yu, C.: Finding Related Tables. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2012), Scottsdale, Arizona, USA, May 20-24, pp. 817–828 (2012)

[WLW+12] Wu, W., Li, H., Wang, H., Zhu, K.Q.: Probase: A Probabilistic Taxonomy for Text Understanding. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2012), Scottsdale, Arizona, USA, May 20-24, pp. 481–492 (2012)

[WWW+12] Wang, J., Wang, H., Wang, Z., Zhu, K.Q.: Understanding tables on the web. In: Atzeni, P., Cheung, D., Ram, S. (eds.) ER 2012. LNCS, vol. 7532, pp. 141–155. Springer, Heidelberg (2012)

Uncoupled MapReduce: A Balanced and Efficient Data Transfer Model

Jie Zhang[1,2], Maosen Sun[1,2], Jian Lin[1,2], and Li Zha[1]

[1] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2] University of the Chinese Academy of Sciences, Beijing, China
{zhangjie,sunmaosen}@software.ict.ac.cn, {linjian,char}@ict.ac.cn

Abstract. In the MapReduce model, reduce tasks need to fetch output data of map tasks in the manner of "pull". However, reduce tasks which are occupying reduce slots cannot start to compute until all the corresponding map tasks are completed. It forms the dependence between map and reduce tasks, which is called the coupled relationship in this paper. The coupled relationship leads to two problems, reduce slot hoarding and underutilized network bandwidth. We propose an uncoupled intermediate data transfer model in order to address these problems. Three core techniques, including weighted mapping, data pushing, and partial data backup are introduced and applied in Apache Hadoop, the mainstream open-source implementation of MapReduce model. This work has been practised in Baidu, the biggest search engine company in China. A real-world application for web data processing shows that our model can improve the system throughput by 29.5%, reduce the total wall time by 22.8%, and provide a weighted wall time acceleration of 26.3%. What's more, the implementation of this model is transparent to user jobs and compatible with the original Hadoop.

Keywords: MapReduce, Data transfer, Uncoupled model.

1 Introduction

With the arrival of "big data" era, platforms which can process and store large data are receiving more and more attention, such as Dryad [7], Sector/Sphere [6], and MapReduce [5]. The MapReduce programming model proposed by Google has become the mainstream data-centric platform for large data processing because of its scalability and simplicity. The open-source implementation, Apache Hadoop [1], is used widely.

The MapReduce programming model is a software architecture for parallel computing on large data sets. In this architecture, one node works as the master where a JobTracker runs. The JobTracker is responsible for monitoring and managing map and reduce tasks. The other nodes work as slaves where TaskTrackers run. The TaskTrackers are responsible for executing map and reduce tasks. When a job is submitted, the related input data are divided into several splits. The JobTracker will pick up idle TaskTrackers to perform map tasks on the

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 34–45, 2013.

splits, and then perform reduce tasks on the intermediate output of map tasks. The final result will usually be a set of key-value pairs stored in the Hadoop Distributed File System (HDFS).

In the original MapReduce model, data are transferred in the way that can be described as "pull". When some of the map tasks are completed, the corresponding reduce tasks can fetch data in the shuffle phase. The reduce phrase will not start until all the map tasks finish. Thus, the reduce tasks will occupy the assigned slots all the time to wait for the completion of all map tasks, which is called the coupled relationship. Due to the coupled relationship between map and reduce tasks, it results in two problems, reduce slot hoarding [10] and underutilized network bandwidth.

1.1 Reduce Slot Hoarding Problem

In the MapReduce programming model, it usually begins to schedule reduce tasks when a certain amount of map tasks are completed. If a big job is submitted, reduce tasks of the job will occupy all the assigned slots until all the map tasks finish. Consequently, when another job is submitted at this moment, it will not get corresponding reduce slots even after all the map tasks finish. Therefore, the later one will starve until the big job is completed. This is called the reduce slot hoarding problem, which will seriously reduce the execution efficiency of jobs, especially for small ones.

One solution to this problem is to delay the reduce tasks. In [10], the authors put forward a solution that starts reduce tasks after the completion of map tasks. However, their tests show that it will decrease the whole throughput. We consider that if reduce tasks start early, partially overlapped with map tasks, they will get a part of the data from map tasks, which can save the time of data transfer and the total completion time.

1.2 Underutilized Network Bandwidth Problem

For a MapReduce job, the network load will mainly concentrate in reduce tasks. A reduce task includes three phases: shuffle, sort and reduce. The shuffle phase needs large network bandwidth, while map tasks do not. A map task can almost read data from local disks, because of its high data locality.

There are some capabilities to balance network load and optimize network bandwidth in the original version of MapReduce, such as scheduling the reduce tasks in advance. When the completion of map tasks reaches a certain ratio (default 5%), the reduce tasks will start so that they can run in parallel with maps. However, it can only alleviate this problem rather than resolve it. When some of the early-scheduled reduce tasks, whose desired intermediate data are ready, are getting partitions through shuffle, they cannot use much bandwidth in the map phrase. Conversely, they will occupy the reduce slots all the time. Besides, limited by the total slots, not all the reduce tasks can be scheduled. Therefore, these reduce tasks cannot work with map tasks simultaneously, and the network bandwidth will still be underutilized.

In this paper, we propose an uncoupled MapReduce model to address the above two problems, which can also improve the system throughput and overall resource utilization. The rest of this paper is organized as follows. In section 2, we present the uncoupled MapReduce model. Section 3 describes the architecture and implementation of this model. Section 4 offers the evaluation about our model and its application effects. At last, section 5 concludes the paper.

2 The Uncoupled MapReduce Model

An uncoupled MapReduce model with intermediate data transfer is designed to address the two problems, meanwhile improving the job execution efficiency and system throughput. Figure 1 shows the data flow in this model. The data transfer is completed in the map tasks in the uncoupled version of MapReduce, instead of in the reduce tasks as the original version does. It needs to meet two conditions as follows:

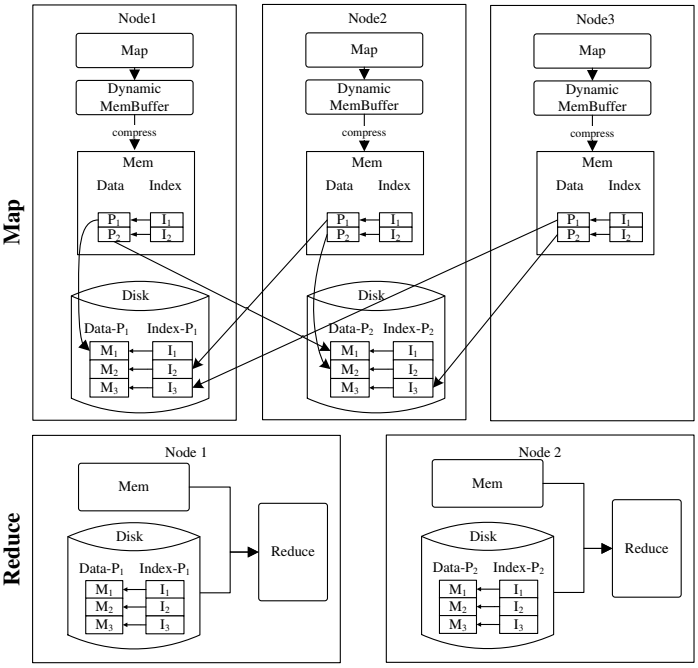


Fig. 1. The data flow in uncoupled MapReduce

– To address the reduce slot hoarding problem thoroughly, the reduce tasks should be scheduled after all the map tasks are completed. Reduce tasks will not occupy the slots, and they can read the data to process locally once launched, which improves job execution efficiency and system throughput.

– To address the underutilized network bandwidth problem, the data transfer process should be completed in map tasks. When reduce tasks start, they will read data directly from local disks. Therefore, the network load will not concentrate in the reduce tasks and the network bandwidth in the map tasks can be used fully.

However, there is a conflict between the two conditions. Map tasks need to transfer data to reduce tasks, but reduce tasks do not run until all the map tasks are completed. Therefore, some trade-offs are necessary to find out when and where the reduce tasks run. Three techniques are introduced to resolve the problem.

– **Weighted mapping.** This technique creates a mapping relationship between reduce tasks and nodes. Through the mapping relationship, the map tasks can find out the nodes where the corresponding reduce tasks will run, and they can transfer data to these nodes. In a heterogeneous cluster, the computing capability of each node is not identical. In consideration of this fact, each node has its own weight. The node with higher weight will get more tasks. This technique can guarantee balance and consistency of task assignment.
  - **Balance.** The balance means that the node with greater weight will be assigned more reduce tasks. A linear relationship exists between the number of assigned tasks and the node's weight. We assume that there are n nodes with weight $w_i (1 \leq i \leq n)$ in a cluster. Normalize the weight and get the normalization value w'_i from w_i, as shown in Equation 1.

$$w'_i = \frac{w_i}{\sum_{i=1}^{n} w_i} \tag{1}$$

    The variable M stands for the total number of reduce tasks, and M_i stands for the number of reduce tasks assigned to node i. Equation 2 shows the result of task assignment.

$$M_i = w'_i M = \frac{w_i}{\sum_{i=1}^{n} w_i} M \tag{2}$$

    The two equations above can ensure the balance of task assignment.
  - **Consistency.** The consistency means that the fixed mapping relationship between reduce tasks and nodes should be guaranteed and cannot be changed once decided. We assign reduce task $R_j (1 \leq j \leq M)$ with weight w_{R_j}. The relationship is expressed in Equation 3.

$$w_{R_j} = \frac{j}{M} \tag{3}$$

    Reduce task R_j with weight w_{R_j} will be mapped to node $k (1 \leq k \leq n)$ if they follow the relationship in Equation 4.

$$\sum_{i=1}^{k-1} w'_i < w_{R_j} \leq \sum_{i=1}^{k} w'_i \tag{4}$$

The mapping relationship is shown in Figure 2. The two kinds of weights will be normalized into the same range. Any module of a MapReduce application can inquire the relationship through Equation 4.

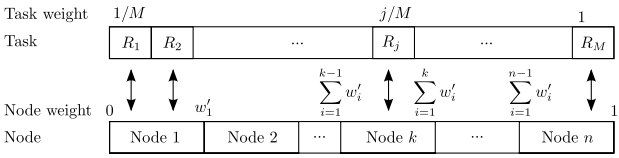


Fig. 2. The mapping relationship between reduce tasks and nodes

- **Data pushing.** In the uncoupled version of MapReduce, map tasks implement data transfer using data pushing. As shown in Figure 1, map tasks will put intermediate data into dynamic buffers and partition them. Then they push data to the reduce tasks in corresponding mapped node from partition 1 to n in order, which is infeasible in the original version because where reduce tasks are responsible for getting data from map tasks. This idea is partially inspired by the pipelined MapReduce [4]. In the uncoupled version, a server is setup in each node which is responsible for receiving data from map tasks as shown in Figure 3. When map tasks are generating intermediate output data, they will work as clients and push data to the servers. So there is no need to start reduce tasks before map tasks finish.
- **Partial data backup.** Each node has a server for receiving data from map tasks. If some servers go wrong, the data pushed by map tasks will be lost. It is costly to re-execute the completed map tasks. The partial data backup technique can resolve this problem. When the map tasks are pushing data, the data will also be backed up on local disks. When some servers go wrong, there is no need to re-execute the completed map tasks, because the data can be recovered by reduce tasks. A backup server is setup on each node which is responsible for managing the backup data. The reduce tasks will pull their own backup data by requesting each backup server. "Partial" here means if a partition is pushed from a map task to its own node, it will not be backed up. What's more, it is compatible with the original fault tolerance mechanism in MapReduce.

This model uncouples the dependency relationship of maps and reduces in MapReduce, and replaces the shuffle phase in the original version. Through the three techniques mentioned above, it can make sure that when the map tasks are completed, all the reduce tasks can read data from their local disks. Therefore, reduce tasks will not occupy the reduce slots to wait for the completion of map tasks. It can also satisfy the needs of slots from other jobs. This model can make full use of the network bandwidth in map and reduce tasks and balance the network load.

3 Architecture and Implementation

Considering the original architecture of MapReduce and the requirements of three techniques mentioned above, the architecture of uncoupled MapReduce is designed. It includes three kinds of modules: master control module, data transfer module, and fault tolerance module. We have implemented the architecture in Hadoop, integrating different modules in the master and slave nodes. The uncoupled MapReduce architecture and its implementation in Hadoop are presented in Figure 4.

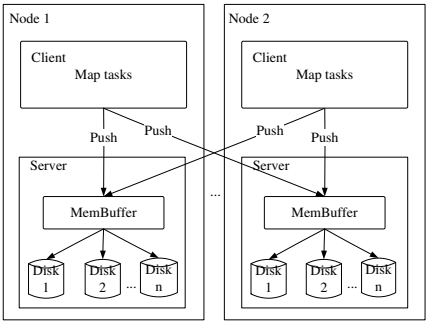


Fig. 3. The diagram of data pushing

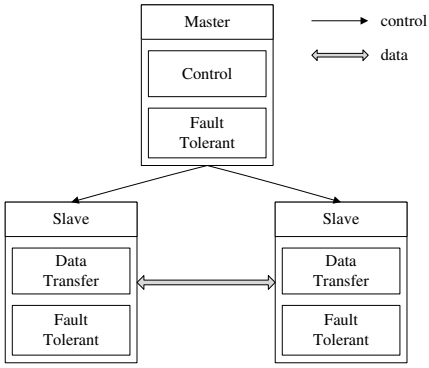


Fig. 4. The uncoupled MapReduce architecture and its implementation in Hadoop

3.1 Master Control Module

The master control module lies in the master node, responsible for monitoring and scheduling tasks, and coordinating other modules. Its functionalities are as follows.

- Create the mapping relationship between reduce tasks and nodes.
- Convey the mapping relationship information to data transfer modules and fault tolerance modules.
- Schedule tasks and make sure that reduce tasks will not start until all the map tasks are completed.
- Ensure balance and consistency of the mapping relationship, and make each node get proper tasks according to its computing capability.
- Coordinate the data transfer module and fault tolerance module. Normally, this module controls the data transfer module to complete the task of data pushing. If there is something wrong with data pushing, this module will notify the fault tolerance module to recover the missing data.

3.2 Data Transfer Module

The data transfer modules lie in all the slave nodes, responsible for processing, storing, and pushing data. Once a map task is completed, the module will push

data to the servers in the mapped nodes where the corresponding reduce tasks run. As mentioned above, the servers are responsible for receiving and managing the data. The functionalities of this module are as follows.

- Create a server in each slave node.
- Get the output data from the map tasks.
- Do some preprocessing on the data, such as compressing.
- Get the mapping relationship.
- Work as a client to push the data of each partition to their corresponding reduce tasks.
- The server in this module is responsible for receiving and managing data.

3.3 Fault Tolerance Module

The fault tolerance modules lie in all the nodes, responsible for processing exceptions. As our model changes the intermediate data transfer mode, we must make some supplements to the original fault tolerance mechanism. In this module, the partial data backup technique is introduced following specific rules. The functionalities of this module are as follows.

- Backup each partition that a map task will push to other nodes in the local disk.
- Assign the mapped reduce tasks to other nodes when a node is failed.
- Offer the backup data to reduce tasks through backup servers, if the backup data has been made successfully. A reduce task checks whether the node is the mapped one through weighted mapping mechanism. If it is not the mapped one, the reduce task will pull its own backup data by requesting to other backup servers.

4 Evaluations

We evaluate the model and its application effects using a micro-benchmark and a real-world example. In the micro-benchmark, we use a cluster to compare the job execution time in the uncoupled version of Hadoop with that in the original version. Our work has also been applied in a production environment of Baidu, which gives a comprehensive evaluation on the uncoupled MapReduce model.

- **Definition 1:** Wall time is the total time span from the moment a job is submitted to the moment it is completed.

- **Definition 2:** Throughput T is the number of jobs finished in a unit time interval. Suppose that N jobs are completed in a time interval t, we will get:

$$T = \frac{N}{t} \tag{5}$$

Our tests use the same workload in both the original version and the uncoupled version, so:

$$N_{original} = N_{uncoupled} \tag{6}$$

Suppose all the jobs are submitted simultaneously. t_{1i} is the wall time of job i in the original version, and t_{2i} is that in the uncoupled version. Use $t_{original} = \max(\{t_{1i}\})$ and $t_{uncoupled} = \max(\{t_{2i}\})$ $(1 \leq i \leq N)$ to represent the total wall time in the original version and the uncoupled version. Then we define the throughput increment rate as I:

$$I = \frac{T_{uncoupled} - T_{original}}{T_{original}} = \frac{\frac{N_{uncoupled}}{t_{uncoupled}} - \frac{N_{original}}{t_{original}}}{\frac{N_{original}}{t_{original}}} = \frac{t_{original}}{t_{uncoupled}} - 1 \tag{7}$$

Then we define the rate of total wall time reduction as r, and the rate of job i's wall time reduction as r_i:

$$r = \frac{t_{original} - t_{uncoupled}}{t_{original}} \tag{8}$$

$$r_i = \frac{t_{1i} - t_{2i}}{t_{1i}} \tag{9}$$

The impact factor of job i, λ_i, represents the proportion of job i's wall time in all the jobs. We can get λ_i from Equation 10:

$$\lambda_i = \frac{t_{1i}}{\sum_{i=1}^{N} t_{1i}} \tag{10}$$

The weighted wall time acceleration P represents the sum of wall time reduction rate with impact factors. It can be deduced from Equation 11:

$$P = \sum_{i=1}^{N} \lambda_i r_i = \sum_{i=1}^{N} \frac{t_{1i}}{\sum_{i=1}^{N} t_{1i}} \frac{t_{1i} - t_{2i}}{t_{1i}} = \frac{\sum_{i=1}^{N} (t_{1i} - t_{2i})}{\sum_{i=1}^{N} t_{1i}} = 1 - \frac{\sum_{i=1}^{N} t_{2i}}{\sum_{i=1}^{N} t_{1i}} \tag{11}$$

The throughput increment rate and total wall time reduction rate are the metrics reflecting the overall performance. The weighted wall time acceleration is the metric reflecting the cumulative performance of each job.

4.1 Micro-benchmark

The micro-benchmark is performed in a cluster with 6 nodes. The operating system is CentOS 6.1 x86_64, and the Hadoop version is 0.19. We use gridmix [2] applications for our test. Gridmix is a set of benchmark programs for Hadoop which contains several kinds of jobs. The micro-benchmark includes 3 jobs as shown in Table 1.

Figure 5 shows the execution time of each job and the total completion time of all the workloads. In our test, the total time of the original version is 3500s, and

that of the uncoupled version is 2620s. There is no reduce slot hoarding in the uncoupled version. The throughput increment is 34% (I) through Equation 7, The total wall time is reduced by 25% (r) through Equation 8, and the weighted wall time acceleration reaches 48% (P) through Equation 11. The test shows that our model can balance the network load properly and improve the system throughput.

4.2 Real-World Example

The uncoupled MapReduce implementation based on Hadoop has been deployed in a production environment of Baidu supporting some business applications. The real-world example provides strong evidence on the effects of this work.

Baidu is the biggest search engine company in China. It has tens of clusters performing Hadoop jobs for many web data processing applications, and generates more than 3 PB data volume per day [8]. Although the clusters can deal with hundreds of jobs everyday, they still meet with some problems. For example, the CPU and network bandwidth utilization rates are not high in spite of full workload. The uncoupled version of Hadoop has been deployed in a server cluster with 70 nodes, which is one of the shared Hadoop platforms for many departments. The resource scale reaches about 560 cores, 1,120 GB memory, and 770 TB storage. The operating system is Red Hat Enterprise Linux AS release 4, and the Hadoop version is 0.19. Many kinds of jobs run in this real environment, such as log analysis, inverted index, web ranking, etc. Four kinds of jobs are used in our test, including CPU-intensive ones and I/O-intensive ones. The job information is shown in Table 2.

Table 1. The workload of the micro-benchmark

Job name	Input size	Maps	Reduces
job1	50 GB	400	200
job2	12.5 GB	100	50
job3	6.25 GB	50	25

Table 2. The workload of the real-world example

Job name	Input size	Maps	Reduces
job1	3.4 TB	14700	1600
job2	3.2 TB	14400	1600
job3	353 GB	1700	800
job4	343 GB	1600	800

Figure 6 shows the execution time of each job and the total completion time of all the workload. In our test, the total time of the original version is 272 mins, and that of the uncoupled version is 210 mins. So we can get the rate of throughput increment:

$$I = \frac{t_{original}}{t_{uncoupled}} - 1 = 29.5\% \tag{12}$$

The rate of total wall time reduction:

$$r = \frac{t_{original} - t_{uncoupled}}{t_{original}} = 22.8\% \tag{13}$$

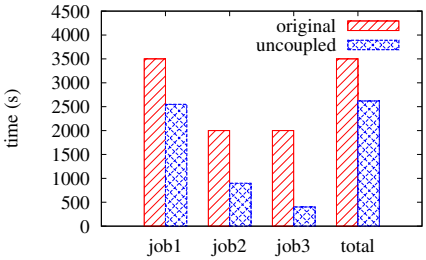


Fig. 5. Comparison of execution time in the micro-benchmark

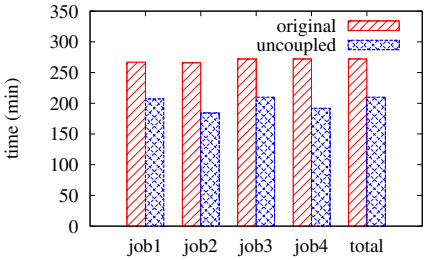


Fig. 6. Comparison of execution time in the real-world example

And the weighted wall time acceleration:

$$P = 1 - \frac{\sum_{i=1}^{4} t_{2i}}{\sum_{i=1}^{4} t_{1i}} = 26.3\% \tag{14}$$

In addition, job3 and job4 in the original version cannot get reduce slots when their map tasks are completed. They are delayed by 51 mins and 63 mins respectively because of reduce slot hoarding problem.

Figure 7, 8, 9, and 10 show the comparison of the original version and the uncoupled version in resources utilization. The workload is the above examples. In the uncoupled version, it took 3.5 hours to complete all the jobs, and the last map task finished in about 2.5 hours. While in the original version, it took 4.5 hours to complete all the jobs, and the last map task finished in about 3.3 hours.

Figure 7 illustrates the average CPU utilization rate of the cluster. The CPU utilization rate in the uncoupled version is higher than that in the original version. The reason can be explained from the map and reduce tasks respectively. In the map tasks, intermediate data are pushed and received by the corresponding servers, which results in more data preprocessing and transferring operations. In the reduce tasks, data is processed faster due to data locality.

Figure 8 shows the network load (send throughput) of the cluster. The uncoupled version uses more network bandwidth in the map tasks to transfer data. Each map will push all of its output data to its corresponding reduce tasks, instead of data being pulled partially by reduce tasks in the original version. There is little network bandwidth in reduce phase of the uncoupled version, because the reduce tasks will read data from their local disks. The peak in the end shows that HDFS uses more bandwidth to backup the output data of reduce tasks including the results and completion information of jobs.

Figure 9 and 10 show the disk I/O (write and read throughput) of the cluster. In the uncoupled version, more disk I/O workload is involved in the map tasks than that in the reduce tasks, because the map tasks will read data from disks and push them to the mapped nodes where the servers receive and write data.

While in the original version, the average throughput of disks is relatively low. Shuffle is an ineffective and complicated phase, which has a significant impact on the execution time and system throughput.

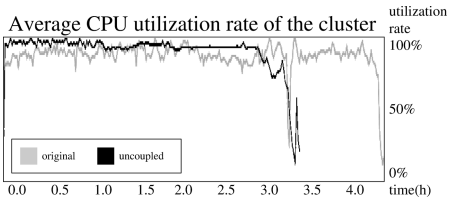


Fig. 7. Comparison of the average CPU utilization rate of the cluster

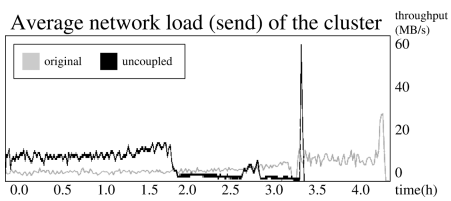


Fig. 8. Comparison of the average network load (send) of the cluster

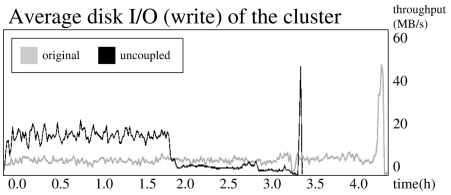


Fig. 9. Comparison of the average disk I/O (write) of the cluster

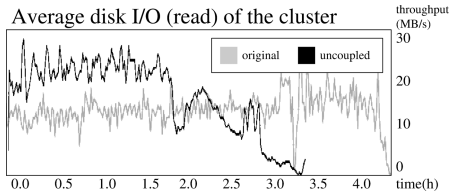


Fig. 10. Comparison of the average disk I/O (read) of the cluster

Overall, the uncoupled version of Hadoop increases the resource utilization rates, and avoids the waste of network and disk bandwidth.

5 Conclusion and Future Work

Although there are a variety of optimizations to improve the localization rate of data in the MapReduce model, data transfer is inevitable. Reduce tasks need to pull intermediate data from map tasks, which will decrease the execution efficiency of jobs. In [3], it shows that shuffle can take about 33% of the operating time in reduce tasks. Meanwhile, data transfer efficiency is very low, which is a system bottleneck [9]. Impacted by these facts, the coupled relationship between map and reduce tasks results in two problems, reduce slot hoarding and underutilized network bandwidth.

In this paper, we propose an uncoupled intermediate data transfer model to resolve the reduce slot hoarding and underutilized network bandwidth problem with the aim of improving the resource utilization and system throughput. Three techniques, including weighted mapping, data pushing, and partial data backup, are implemented. This work has been practised in Baidu, the biggest search engine company in China. In a real-world application, the test shows that the

throughput can be increased by 29.5%, the total wall time is reduced by 22.8%, and the weighted wall time acceleration reaches 26.3%, compared with the original version of Hadoop. The resource utilization rates of CPU, network and disk are also increased.

Two improvements are planned: 1) Design a kind of scheduler for this model which can use the cluster resources more reasonably. 2) Monitor workloads and resources dynamically, instead of setting constant slots and weights. Hopefully, our paper would assist in the study of heterogeneous resources utilization.

Acknowledgment. We would like to thank Ruijian Wang, Chen Feng, Fan Liang, and Dixin Tang from Institute of Computing Technology, Chinese Academy of Sciences for the valuable discussions. We also thank Chuan Xu, Linjiang Lian, and Meng Wang from Baidu for their assistance and support. This research is supported in part by the Hi-Tech Research and Development (863) Program of China (Grant No. 2011AA01A203, 2013AA01A213).

References

1. Apache hadoop, `http://hadoop.apache.org/`
2. Gridmix, `http://hadoop.apache.org/docs/stable/gridmix.html`
3. Chowdhury, M., Zaharia, M., Ma, J., Jordan, M.I., Stoica, I.: Managing data transfers in computer clusters with orchestra. In: Proceedings of the ACM SIGCOMM 2011 Conference, SIGCOMM 2011, pp. 98–109 (2011)
4. Condie, T., Conway, N., Alvaro, P., Hellerstein, J.M., Elmeleegy, K., Sears, R.: MapReduce online. In: Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation, NSDI 2010, pp. 1–15 (2010)
5. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. In: Proceedings of the 6th USENIX Symposium on Operating Systems Design & Implementation, OSDI 2004, pp. 137–150 (2004)
6. Gu, Y., Grossman, R.L.: Sector and sphere: Towards simplified storage and processing of large scale distributed data. arXiv:0809.1181 (2008)
7. Isard, M., Budiu, M., Yu, Y., Birrell, A., Fetterly, D.: Dryad: distributed data-parallel programs from sequential building blocks. In: Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems, EuroSys 2007, pp. 59–72 (2007)
8. Ma, R.: Introduction to part of the baidu's distributed systems, `http://www.slideshare.net/cydu/sacc2010-5102684`
9. Wang, Y., Que, X., Yu, W., Goldenberg, D., Sehgal, D.: Hadoop acceleration through network levitated merge. In: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2011, pp. 1–10 (2011)
10. Zaharia, M., Borthakur, D., Sen Sarma, J., Elmeleegy, K., Shenker, S., Stoica, I.: Job scheduling for multi-user MapReduce clusters. Tech. Rep. UCB/EECS-2009-55, EECS Department, University of California, Berkeley (2009)

Efficient Evaluation of Ad-Hoc Range Aggregates

Christian Ammendola[1], Michael H. Böhlen[2], and Johann Gamper[1]

[1] Free University of Bozen–Bolzano, 39100 Bozen–Bolzano, Italy
christian@ammendola.name, gamper@inf.unibz.it
[2] University of Zurich, 8050 Zurich, Switzerland
boehlen@ifi.uzh.ch

Abstract. θ-MDA is a flexible and efficient operator for complex ad-hoc multi-dimensional aggregation queries. It separates the specification of aggregation groups for which aggregate values are computed (base table $\mathbf{b}$) and the specification of aggregation tuples from which aggregate values are computed. Aggregation tuples are subsets of the detail table $\mathbf{r}$ and are defined by a general θ-condition. The θ-MDA requires one scan of $\mathbf{r}$, during which the aggregates are incrementally updated.

In this paper, we propose a two-step evaluation strategy for θ-MDA to optimize the computation of ad-hoc range aggregates by reducing them to point aggregates. The first step scans $\mathbf{r}$ and computes point aggregates as a partial intermediate result $\tilde{\mathbf{x}}$, which can be done efficiently. The second step combines the point aggregates to the final aggregates. This transformation significantly reduces the number of incremental updates to aggregates and reduces the runtime from $\mathcal{O}(|\mathbf{r}| \cdot |\mathbf{b}|)$ to $\mathcal{O}(|\mathbf{r}|)$, provided that $|\mathbf{b}| < \sqrt{|\mathbf{r}|}$ and $|\tilde{\mathbf{x}}| \approx |\mathbf{b}|$, which is common for OLAP. An empirical evaluation confirms the analytical results and shows the effectiveness of our optimization: range queries are evaluated with almost the same efficiency as point queries.

1 Introduction

Multi-dimensional aggregation queries, such as range aggregates that aggregate sets of tuples identified by a range condition, are an important class of queries in business intelligence and data warehousing applications. To efficiently process such queries, various techniques have been proposed, including extensions to SQL [7], generalized projections [8], pre-aggregation of data cubes [9], and (relative) prefix sums [6,10]. θ-MDA [1] is a flexible and efficient operator for *ad-hoc* multi-dimensional aggregation queries, where pre-computed aggregates are not available and a scan of the detail table is required.

Consider the relation $\mathbf{stays}$ in Fig. 1(a), which stores hospital stays of patients and has the following attributes: admission date (D), patient identifier (P), urgency category (U), and duration of the stay (S). To analyze the impact of urgent stays ($U{=}3$) on the average duration of stays, aggregates for stays with $U{\in}\{1,2\}$ are compared to aggregates for stays with $U{\in}\{1,2,3\}$. Consider the following query Q1: *Compute the cumulative average duration of stays per admission date ($C1$) and per admission date and urgency ($C2$).* The result of Q1 is shown in table $\mathbf{x}$. The first two columns represent the aggregation groups. The other two columns represent the aggregate results. The average is represented as a sum/count pair.

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 46–59, 2013.

stays

	D	P	U	S
r_1	31/01/13	P1	1	2
r_2	31/01/13	P2	1	4
r_3	31/01/13	P3	3	8
r_4	31/01/13	P4	2	4
r_5	31/01/13	P5	1	3
r_6	31/01/13	P6	2	3
r_7	01/02/13	P7	1	2
r_8	01/02/13	P8	1	3
r_9	01/02/13	P9	3	9
r_{10}	01/02/13	P10	3	6

(a) Detail Table

x

	D	U	$C1$	$C2$
x_1	31/01/13	2	24/6	16/5
x_2	31/01/13	3	24/6	24/6
x_3	01/02/13	2	44/10	21/7
x_4	01/02/13	3	44/10	44/10

(b) Result of Query Q1

Fig. 1. Running Example

$C1$ and $C2$ are *range aggregates*. Each aggregate is computed from the aggregation tuples: all detail tuples that satisfy a given range predicate (exemplified by hatched areas in Fig. 1). For instance, aggregate $x_3.C2$ is the average duration of a stay over all detail tuples in **stays** with $D \leq 01/02/13$ and $U \leq 2$, i.e., the tuples $\{r_1, r_2, r_4, r_5, r_6, r_7, r_8\}$. While one-dimensional range aggregates can be solved efficiently with SQL window functions, multi-dimensional range aggregates require expensive joins with range predicates [1].

The θ-MDA operator [1] offers a succinct formulation and efficient computation of ad-hoc multi-dimensional aggregation queries. The operator separates the specification of the aggregation groups in a base table $\mathbf{b}$ (first two columns in table $\mathbf{x}$ in Fig. 1(b)) from the specification of the aggregation tuples. For each aggregation group in $\mathbf{b}$ a θ-condition defines the aggregation tuples in the detail table $\mathbf{r}$ from which aggregate values are computed, e.g., $x_3.C2$ is computed from the aggregation tuples $\{r_1, r_2, r_4, r_5, r_6, r_7, r_8\}$. The evaluation strategy of θ-MDA works as follows: the result table $\mathbf{x}$ is initialized to $\mathbf{b}$ and one additional column for each aggregate, followed by a scan of $\mathbf{r}$ during which the aggregate values are incrementally updated. A major part of the evaluation cost for range aggregates is the incremental update of the aggregate values in $\mathbf{x}$ since each tuple in $\mathbf{r}$ affects many entries in $\mathbf{x}$. For instance, tuple r_1 affects both aggregates ($C1$ and $C2$) in all result tuples (x_1 to x_4). Moreover, for θ-conditions that reference a subset of the attributes in $\mathbf{b}$, redundant updates occur. Consider $C1$ with θ-condition $D \leq 31/01/13$. Entries x_1 and x_2 are updated for all input tuples with $D \leq 31/01/13$.

In this paper, we tackle these problems and propose an efficient two-step evaluation strategy for ad-hoc range aggregates by reducing them to point aggregates. The first step scans $\mathbf{r}$ and computes corresponding point aggregates as a partial intermediate result, $\tilde{\mathbf{x}}$. Point aggregates require much less incremental updates and can be computed very efficiently since only equality conditions are used to define aggregation tuples. The second step combines the result of the point aggregates with the help of the corresponding super aggregates [7] (e.g., SUM to add up COUNTs) and the θ-conditions to obtain the final result relation. To further reduce the number of updates for aggregates with θ-conditions that reference a subset of the attributes in $\mathbf{b}$, we split the intermediate result

table into a separate table for each aggregate function so that each detail tuple affects exactly one entry in each intermediate table. This two-step strategy reduces the number of incremental updates from $\mathcal{O}(|\mathbf{r}| \cdot |\mathbf{b}|)$ to $\mathcal{O}(|\mathbf{r}|)$ if $|\mathbf{b}| < \sqrt{|\mathbf{r}|}$ and $|\tilde{\mathbf{x}}| \approx |\mathbf{b}|$, which is common for OLAP queries, such as for the TPC-H benchmark. We integrate these optimizations into a new evaluation algorithm, termed $\texttt{TCMDA}^+$. An empirical evaluation confirms the analytical results and shows that range aggregates can be computed with almost the same efficiency as point aggregates.

The technical contributions can be summarized as follows:

- we show how the evaluation of range aggregates can be reduced to the evaluation of point aggregates by computing an intermediate table of corresponding point aggregates, which are then combined to the final aggregates;
- to further reduce the number of updates in the intermediate table, we maintain a separate intermediate table for each aggregate;
- we integrate these optimizations in a new evaluation algorithm, termed $\texttt{TCMDA}^+$, which reduces the runtime complexity from $\mathcal{O}(|\mathbf{r}| \cdot |\mathbf{b}|)$ to $\mathcal{O}(|\mathbf{r}|)$ if $|\mathbf{b}| < \sqrt{|\mathbf{r}|}$ and $|\tilde{\mathbf{x}}| \approx |\mathbf{b}|$;
- we report experimental results that confirm the analytical results and show that range aggregates are evaluated with almost the same efficiency as point aggregates.

The paper is organized as follows: Sec. 2 reports related work, followed by a summary of the θ-MDA operator in Sec. 3. In Sec. 4, we present the reduction of range to point aggregates. These optimizations are integrated in a new algorithm in Sec. 5. In Sec. 6, we report experimental results. Section 7 concludes the paper and points to future work.

2 Related Work

Various research work investigates multi-dimensional data aggregation techniques to gain more flexibility and/or performance. The *CUBE* operator [7] is part of the SQL standard and allows to express aggregation queries with equality constraints over several attributes in a concise way. For aggregation queries over a part of the data cube, grouping sets can be used. Additional support for aggregation queries over one dimension is provided by *window functions* in SQL:2003. The support is based on the ordering of tuples. Currently, SQL does not support the efficient evaluation of range aggregates over multiple dimensions [1].

An orthogonal approach to improve the query performance is to pre-compute aggregates. Harinarayan et al. [9] propose a strategy for the selection of a subset of all possible data cubes to be materialized. To avoid the complete re-computation of cubes when source relations change, incremental update strategies have been proposed [11,12,13,15]. Although the pre-computation of data cubes works well for point aggregates, the performance of range aggregates suffers since the cells in the data cube must be accessed repeatedly. To tackle this problem, Ho et al. [10] propose to maintain additionally a so-called *prefix sum cube*. Subsequent work has studied techniques to lower the comparably high update costs of the prefix sum cube [5,6,14].

The goal in this paper is the efficient evaluation of complex *ad-hoc* OLAP queries when pre-computed aggregates (cubes or prefix sum cubes) are *not* available. Such

an approach provides more flexibility for the exploration of large data sets when the requirements for the analysis and queries are not known a priori.

To efficiently answer ad-hoc OLAP queries with a single scan of the detail table, Akinde et al. [2] propose the *multi-dimensional join* (MDJ) and later the *generalized multi-dimensional join* (GMDJ) [3,4]. The operator has been used in complex OLAP settings to transform general sub-query expressions into expressions that use the GMDJ instead of joins, outer joins, or set difference. Sridhar et al. [16] use the GMDJ in combination with MapReduce to compute aggregation queries over RDF data.

The *θ-constrained multi-dimensional aggregation* (θ-MDA) operator [1] extends the MDJ and presents a detailed cost model together with algebraic transformation rules. θ-MDA outperforms SQL for complex multi-dimensional aggregation queries, such as range aggregates over multiple dimensions. In this paper, we propose an alternative evaluation strategy for θ-MDA, which significantly reduces the cost of the computation of range aggregates to almost the same cost as point aggregates.

3 Preliminaries

We assume two relations, $\mathbf{b}$ and $\mathbf{r}$, with schema $\mathbf{B} = (B_1, \ldots, B_t)$ and $\mathbf{R} = (A_1, \ldots, A_p)$, respectively. For a tuple x we write $x.\mathbf{B}$ as an abbreviation for $(x.B_1, \ldots, x.B_t)$. E/C denotes the renaming of E to C, $attr(\theta)$ denotes the set of attributes used in θ, and f_i denotes an aggregate function.

Definition 1. *(θ-MDA [1]) Let $\mathbf{b}$ and $\mathbf{r}$ be relations with schema $\mathbf{B}$ and $\mathbf{R}$, respectively, $F = (f_1/C_1, \ldots, f_m/C_m)$ be aggregate functions over attributes in $\mathbf{R}$, and $\Theta = (\theta_1, \ldots, \theta_m)$ be conditions with $attr(\theta_i) \subseteq \mathbf{B} \cup \mathbf{R}$. The θ-MDA operator is defined as*

$$\mathcal{G}^{\theta}(\mathbf{b}, \mathbf{r}, F, \Theta) = \{b \circ v \mid b \in \mathbf{b} \wedge v = (f_1(\mathbf{r}_{[b,\theta_1]}), \ldots, f_m(\mathbf{r}_{[b,\theta_m]}))\},$$

where $\mathbf{r}_{[b,\theta_i]} = \{r \in \mathbf{r} \mid \theta_i(r, b)\}$ are the aggregate tuples from which the aggregate values for aggregation group b are computed.

The *base table* $\mathbf{b}$ specifies the aggregation groups for which a result tuple is reported. The *detail table* $\mathbf{r}$ contains the data from which aggregate values are computed. F is a list of aggregate functions. Each f_i gets as argument a subset of $\mathbf{r}$, $\mathbf{r}_{[b,\theta_i]} \subseteq \mathbf{r}$, that is determined by a condition θ_i, and aggregates one of the attributes. Each entry in the *result table* $\mathbf{x}$ consists of a $\mathbf{b}$-tuple and the aggregation results stored in $C_1, \ldots, C_m$. Query Q1 can be formulated as $\mathcal{G}^{\theta}(\mathbf{b}, \mathbf{r}, F, \Theta)$ with $\mathbf{b} = \pi_{D,U}(\sigma_{U \in \{2,3\}}(\mathbf{stays}))$, $\mathbf{r} = \mathbf{stays}$, $F = ((\mathrm{SUM}(S)/\mathrm{COUNT}(S))/C1, (\mathrm{SUM}(S)/\mathrm{COUNT}(S))/C2)$, and $\Theta = (\theta_1, \theta_2)$ with $\theta_1 \equiv (r.D \leq b.D)$ and $\theta_2 \equiv (r.D \leq b.D \wedge r.U \leq b.U)$.

The evaluation of θ-MDA queries works as follows: (1) initialize the result table $\mathbf{x}$ to $\mathbf{b}$ and the neutral value for each aggregate function; (2) scan $\mathbf{r}$ and incrementally update the aggregates f_i in $\mathbf{x}$ that are affected by an $r \in \mathbf{r}$, i.e., satisfy condition θ_i. After processing all $\mathbf{r}$-tuples, $\mathbf{x}$ contains the result relation. The runtime complexity of this evaluation strategy is $\mathcal{O}(|\mathbf{r}| \cdot |\mathbf{b}|)$ with one scan of $\mathbf{r}$. Figure 2 shows the result table during the evaluation of the first three tuples. The first two detail tuples affect all eight aggregate values in $\mathbf{x}$, whereas r_3 requires six updates.

Observe that the number of aggregates that are updated for an $\mathbf{r}$-tuple depends on the θ_i-conditions. Range aggregates require many more updates than point aggregates.

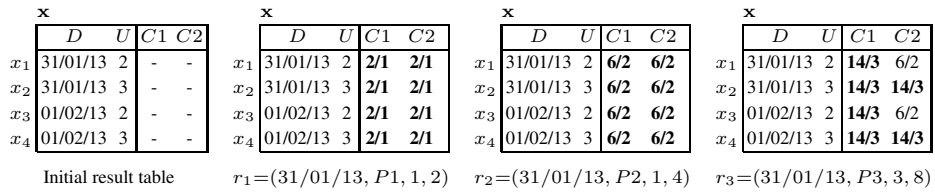


Fig. 2. Processing Tuples in θ-MDA Queries

4 A New Evaluation Strategy for θ-MDA Queries

4.1 Reducing Range to Point Queries

To tackle the problem of a large number of incremental updates for range aggregates and take advantage of the efficient computation of point aggregates, we propose an evaluation strategy that reduces range to point aggregates.

Proposition 1 (Reduction to Point Aggregates). *Let* $\mathbf{b}$, $\mathbf{r}$, $\mathbf{B}$, $\mathbf{R}$, F, Θ *be as in Def. 1,* $G = (g_1, \ldots, g_m)$ *be the super aggregates of the* $f_i \in F$, *and* $\mathbf{R}_i = \mathbf{R} \cap attr(\theta_i)$ *be the attributes in* $\mathbf{R}$ *that occur in* θ_i. *Then,* $\mathbf{x} = \mathcal{G}^\theta(\mathbf{b}, \mathbf{r}, F, \Theta)$ *can be computed as follows:*

1. *construct* $\tilde{\Theta} = (\tilde{\theta}_1, \ldots, \tilde{\theta}_m)$, *where* $\tilde{\theta}_i(r, b) = \bigwedge_{A \in \mathbf{R}_i} r.A = b.A$;
2. *compute an intermediate result table* $\tilde{\mathbf{x}} = \mathcal{G}^\theta(\pi_{\mathbf{R}_1 \cup \cdots \cup \mathbf{R}_m}(\mathbf{r}), \mathbf{r}, F, \tilde{\Theta})$;
3. *compute the result table* $\mathbf{x} = \{b \circ v \mid b \in \mathbf{b} \wedge v = (g_1(\tilde{\mathbf{x}}_{[b,\theta_1]}), \ldots, g_m(\tilde{\mathbf{x}}_{[b,\theta_m]}))\}$, *where* $\tilde{\mathbf{x}}_{[b,\theta_i]} = \pi_{\mathbf{R}_i, C_i}\{\tilde{x} \in \tilde{\mathbf{x}} \mid \theta_i(\tilde{x}, b)\}$.

First, m point aggregates are constructed by creating conditions $\tilde{\Theta} = \{\tilde{\theta}_1, \ldots, \tilde{\theta}_m\}$ such that each $\tilde{\theta}_i$ contains an equality constraint, $r.A = b.A$, for each attribute $A \in \mathbf{R}_i$ that is used in the corresponding θ_i. Second, a $\mathcal{G}^\theta$-call computes an intermediate result table, $\tilde{\mathbf{x}}$, with m point queries, where the base table is a projection of $\mathbf{r}$ to all $\mathbf{R}$-attributes that are used in Θ. This requires significantly less updates in $\tilde{\mathbf{x}}$ than the range queries would do. The final result table, $\mathbf{x}$, is derived from $\tilde{\mathbf{x}}$ using the aggregates g_i in combination with the original conditions θ_i. Following Gray et al. [7], we call the functions g_i that are needed to aggregate the intermediate values the super aggregates. For the standard aggregate functions we have the following pairs of aggregate/super aggregate: MAX/MAX, MIN/MIN, SUM/SUM, COUNT/SUM; average is replaced by sum divided by count. The super aggregates are computed over groups of entries, $\tilde{\mathbf{x}}_{[b,\theta_i]} \subseteq \tilde{\mathbf{x}}$, that are assigned to tuples $b \in \mathbf{b}$ using the original conditions θ_i. Note the projection to the aggregation group attributes $\mathbf{R}_i$ and the aggregate C_i. This is required to eliminate duplicates in situations when a condition $\tilde{\theta}_i$ references only a subset of the aggregation group attributes in $\tilde{\mathbf{x}}$, i.e., $\mathbf{R}_i \subset \mathbf{R}_1 \cup \cdots \cup \mathbf{R}_m$. Although in step 3 each tuple of $\tilde{\mathbf{x}}$ may affect multiple tuples in $\mathbf{x}$, the overall runtime is significantly reduced, provided that $\tilde{\mathbf{x}}$ is much smaller than $\mathbf{r}$, which is frequently the case in OLAP.

Figure 3 shows the evaluation of Query Q1. We have the attribute sets $\mathbf{R}_1 = \{D\}$ and $\mathbf{R}_2 = \{D, U\}$ and the conditions $\tilde{\Theta} = \{\tilde{\theta}_1, \tilde{\theta}_2\}$ with $\tilde{\theta}_1 \equiv (r.D{=}b.D)$ and

$\tilde{\theta}_2 \equiv (r.D{=}b.D \wedge r.U{=}b.U)$. These conditions together with the aggregate functions represent point aggregates, which are computed in the intermediate result table as $\tilde{\mathbf{x}} = \mathcal{G}^{\theta}(\pi_{D,U}(\mathbf{stays}), \mathbf{r}, F, \tilde{\Theta})$. Note the significant reduction of incremental updates in $\tilde{\mathbf{x}}$. For instance, tuple r_1 affects only four aggregates (both aggregates in $\tilde{x}_1$ and aggregate $C1$ in $\tilde{x}_2$ and $\tilde{x}_3$) instead of eight as in Fig. 2. To derive the final result table $\mathbf{x}$, the original conditions, θ_i, are used to determine the subsets $\tilde{\mathbf{x}}_{[b,\theta_i]}$. For result tuple x_1 with aggregation group $b = (31/01/13, 2)$ we have the following subsets:

$$\begin{aligned}
\tilde{\mathbf{x}}_{[(31/01/13,2),\theta_1]} &= \pi_{D,C1}\{\tilde{x} \in \tilde{\mathbf{x}} \mid \tilde{x}.D \leq 31/01/13\} \\
&= \pi_{D,C1}\{\tilde{x}_1, \tilde{x}_2, \tilde{x}_3\} = \{(31/01/13, 24/6)\}, \\
\tilde{\mathbf{x}}_{[(31/01/13,2),\theta_2]} &= \pi_{D,U,C2}\{\tilde{x} \in \tilde{\mathbf{x}} \mid \tilde{x}.D \leq 31/01/13 \wedge \tilde{x}.U \leq 2\} \\
&= \pi_{D,U,C2}\{\tilde{x}_1, \tilde{x}_2\} = \{(31/01/13, 1, 9/3), (31/01/13, 2, 7/2)\}.
\end{aligned}$$

The projection in $\tilde{\mathbf{x}}_{[(31/01/13,2),\theta_1]}$ removes duplicates that originate from $C1$ grouping only by D. Since the super aggregate of SUM and COUNT is SUM, the final aggregates are obtained by summing up the individual sums and counts, respectively. For instance, for the result tuple x_1 we get

$$\begin{aligned}
x_1.C1 &= \text{SUM/SUM}_{C1}(\tilde{\mathbf{x}}_{[(31/01/13,2),\theta_1]}) = \text{SUM/SUM}_{C1}(\{(31/01/13, 24/6)\}) \\
&= 24/6, \\
x_1.C2 &= \text{SUM/SUM}_{C2}(\tilde{\mathbf{x}}_{[(31/01/13,2),\theta_2]}) \\
&= \text{SUM/SUM}_{C2}(\{(31/01/13, 1, 9/3), (31/01/13, 2, 7/2)\} \\
&= (9{+}7)/(3{+}2) = 16/5.
\end{aligned}$$

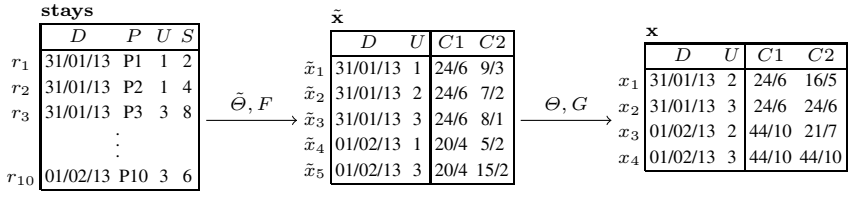


Fig. 3. θ-MDA Evaluation by Reduction to Point Queries

Theorem 1. *The evaluation strategy in Proposition 1 correctly computes the θ-MDA.*

Proof. The theorem applies the following reduction rule for distributive aggregates: an aggregate function, f, over a set of tuples, $\mathbf{r}$, can be reduced to the computation of partial aggregates over a partitioning $\mathbf{r}_1, \ldots, \mathbf{r}_k$ of $\mathbf{r}$, followed by an application of the corresponding super aggregate g, i.e., $f(\mathbf{r}) = g(f(\mathbf{r}_1), \ldots, f(\mathbf{r}_k))$. In step two, each $\tilde{\theta}_i$ induces a partitioning of $\mathbf{r}$ such that all tuples in the same partition have identical values for the attributes $\mathbf{R}_i = \mathbf{R} \cap attr(\theta_i)$. The call to $\mathcal{G}^{\theta}$ with base table $\pi_{\mathbf{R}_1 \cup \cdots \cup \mathbf{R}_m}(\mathbf{r})$ computes for each f_i the partial aggregation results over the individual partitions, i.e., $\tilde{x}.C_i = f_i(\{r \in \mathbf{r} \mid \tilde{\theta}(r, \tilde{x})\})$. In step three, the original $\mathbf{b}$ and θ_is are used to determine which entries $\tilde{x} \in \tilde{\mathbf{x}}$ to combine for each $b \in \mathbf{b}$. Each $\tilde{\mathbf{x}}_{[b,\theta_i]}$ collects all intermediate

tuples, $\tilde{x} \in \tilde{\mathbf{x}}$, that are assigned to b through θ_i. Since all $\mathbf{r}$-tuples in a partition that contributed to an intermediate aggregate, $\tilde{x}.C_i$, have identical values for the attributes $\mathbf{R}_i$, they would have been assigned by θ_i to the same b. The projection to $\mathbf{R}_i, C_i$ eliminates duplicates when $\mathbf{R}_i \subset \mathbf{R}_1 \cup \cdots \cup \mathbf{R}_m$. In such cases, several entries in $\tilde{\mathbf{x}}$ might store the same aggregation group for C_i, but only one can be considered for the final aggregation result. Thus, no spurious tuples are combined for the aggregates in the final result table, and the use of the original θ_i in step three guarantees that all input tuples are considered. □

The following lemma shows that the reduction of range aggregates to point aggregates can be expressed by a nested θ-MDA if all θ_is use the same set of aggregation group attributes $\mathbf{B}$.

Lemma 1. *Let* $\mathbf{b}$, $\mathbf{r}$, F, G, Θ, *and* $\tilde{\Theta}$ *be as in Proposition 1. Furthermore, let* $\mathbf{R}_i = \mathbf{R} \cap attr(\theta_i)$ *denote the attributes in* $\mathbf{R}$ *that occur in* θ_i *and* $\mathbf{R}_i = \mathbf{R}_j$ *for all* i, j, $1 \leq i, j \leq m$. *Then,* $\mathbf{x} = \mathcal{G}^\theta(\mathbf{b}, \mathbf{r}, F, \Theta)$ *can be computed as*

$$\mathbf{x} = \mathcal{G}^\theta(\mathbf{b}, \mathcal{G}^\theta(\pi_{\mathbf{R}_i}(\mathbf{r}), \mathbf{r}, F, \tilde{\Theta}), G, \Theta).$$

Proof. The inner $\mathcal{G}^\theta$-call computes the intermediate table $\tilde{\mathbf{x}}$, which is passed as detail table to the outer $\mathcal{G}^\theta$-call. In the outer call, the original θ_is assign to each $b \in \mathbf{b}$ the associated intermediate tuples, $\tilde{x} \in \tilde{\mathbf{x}}$. Since all aggregates use the same aggregation group attributes, $\mathbf{R}_i$, there are no duplicate aggregation groups in $\tilde{\mathbf{x}}$ for any aggregate, hence a projection as in Proposition 1 is not necessary. The super aggregates in G in combination with the original conditions Θ correctly combine the partial aggregates to the final result. □

Note that using the aggregation groups in $\mathbf{b}$ instead of $\pi_{\mathbf{R}_1 \cup \cdots \cup \mathbf{R}_m}(\mathbf{r})$ for the computation of the intermediate result table $\tilde{\mathbf{x}}$ would not be correct. This happens if $\mathbf{b}$ is sparse, i.e., $\mathbf{r}$ contains more combinations of the grouping attribute values than $\mathbf{b}$ does, i.e., $\mathbf{b} \subset \pi_{\mathbf{B}}(\mathbf{r})$. Since the $\tilde{\theta}_i$s use only equality constraints, some tuples in $\mathbf{r}$ might not be assigned to any entry in the intermediate table $\tilde{\mathbf{x}}$ although they contribute to the final aggregation result. Figure 4 shows table $\tilde{\mathbf{x}}$ for Query Q1 when using the original base table $\mathbf{b}$. In table $\tilde{\mathbf{x}}$, the pre-aggregates for the aggregation groups $(31/01/13, 1)$ and $(01/02/13, 1)$ are missing, but they are needed for the correct computation of the final result. A similar situation occurs if Θ uses a lower number of $\mathbf{B}$-attributes than $\mathbf{R}$-attributes. In this case, the intermediate table $\tilde{\mathbf{x}}$ misses attributes that are needed for the evaluation of the θ_is when producing the final result table.

4.2 Separate Intermediate Result Tables

Even if the $\tilde{\theta}_i$-conditions contain only equality constraints, a single $\mathbf{r}$-tuple might still affect several entries in $\tilde{\mathbf{x}}$. This is the case if a $\tilde{\theta}_i$ constrains only a subset of all grouping attributes, i.e., $\mathbf{R}_i \subset \mathbf{R}_1 \cup \cdots \cup \mathbf{R}_m$. For instance, condition $\tilde{\theta}_1 \equiv (r.D{=}b.D)$ of the first aggregate groups only by D. This produces duplicate aggregation groups in $\tilde{\mathbf{x}}$, such as $D = 31/01/13$ which is present in $\tilde{x}_1$, $\tilde{x}_2$, and $\tilde{x}_3$ (cf. Fig. 3). Each of the detail tuples $r_1, \ldots, r_6$ is assigned to each of these entries, yielding a total of 18 updates of

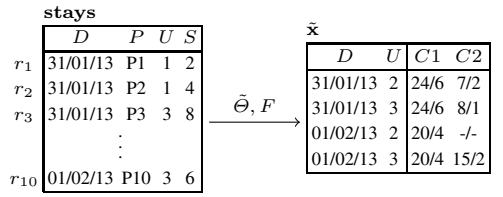


Fig. 4. Using Base Table $\mathbf{b}$ in $\tilde{\mathbf{x}}$

$C1$ instead of six. This type of *redundant* updates can be avoided by using a separate intermediate result table, $\tilde{\mathbf{x}}^i$, for each $\tilde{\theta}_i$ (f_i).

Proposition 2 (Separate Intermediate Result Tables). *Let* $\mathbf{b}$, $\mathbf{r}$, F, G, Θ, $\tilde{\Theta}$, *and* $\mathbf{R}_i$ *be as in Proposition 1. Then,* $\mathbf{x} = \mathcal{G}^\theta(\mathbf{b}, \mathbf{r}, F, \Theta)$ *can be computed as follows:*

1. *compute* m *intermediate result tables* $\tilde{\mathbf{x}}^i = \mathcal{G}^\theta(\pi_{\mathbf{R}_i}(\mathbf{r}), \mathbf{r}, f_i, \tilde{\theta}_i)$ *for* $i = 1, \ldots, m$*;*
2. *compute the result table* $\mathbf{x} = \{b \circ f \mid b \in \mathbf{b} \wedge f = (g_1(\tilde{\mathbf{x}}^1_{[b,\theta_1]}), \ldots, g_m(\tilde{\mathbf{x}}^m_{[b,\theta_m]}))\}$, *where* $\tilde{\mathbf{x}}^i_{[b,\theta_i]} = \{\tilde{x} \in \tilde{\mathbf{x}}^i \mid \theta_i(\tilde{x}, b)\}$.

Figure 5 shows the two intermediate result tables, $\tilde{\mathbf{x}}^1$ and $\tilde{\mathbf{x}}^2$, in our running example that replace table $\tilde{\mathbf{x}}$ from Fig. 3. Table $\tilde{\mathbf{x}}^1$ has one grouping attribute, whereas $\tilde{\mathbf{x}}^2$ has two. The detail tuples $r_1, \ldots, r_6$ require now a total of six updates of $C1$ in table $\tilde{\mathbf{x}}^1$ (one for each tuple), instead of 18 in Fig. 3.

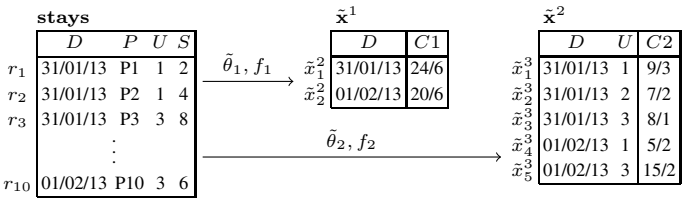


Fig. 5. Separate Intermediate Result Tables

Theorem 2. *The evaluation strategy in Proposition 2 correctly computes* θ*-MDA.*

Proof. The proof is similar as for Theorem 1 with two differences. In the first step, m intermediate result tables are constructed using the $\tilde{\theta}_i$s. Since the aggregation groups of each $\tilde{\mathbf{x}}^i$ are produced by a projection of $\mathbf{r}$ to the attributes $\mathbf{R}_i$ that are used in the corresponding θ_i-condition, duplicate aggregation groups are avoided. The second step merges the intermediate tables to produce the final result table in the same way as in Proposition 1, except the projection in $\tilde{\mathbf{x}}_{[b,\theta_i]}$, which is not needed since no duplicate aggregation groups exist. □

Corollary 1. *The evaluation of* θ*-MDA queries using separate intermediate result tables,* $\tilde{\mathbf{x}}^i$*, as in Proposition 2, requires for each* $r \in \mathbf{r}$ *exactly one update in each* $\tilde{\mathbf{x}}^i$*.*

Proof. None of the intermediate result tables, $\tilde{\mathbf{x}}^i$, contains duplicate aggregation groups and all aggregates are point queries. Thus, each condition, $\tilde{\theta}_i$, associates each detail tuple, $r \in \mathbf{r}$, to exactly one entry in $\tilde{\mathbf{x}}^i$. □

5 Algorithm TCMDA$^+$

Algorithm 1 shows a new algorithm, TCMDA$^+$, for the evaluation of θ-MDA queries that adopts the optimization techniques introduced before.

Algorithm 1. TCMDA$^+(\mathbf{b}, \mathbf{r}, F, \Theta)$

input : base table $\mathbf{b}$, detail table $\mathbf{r}$, aggregate functions $F = (f_1, \ldots, f_m)$, conditions $\Theta = (\theta_1, \ldots, \theta_m)$
output : result relation $\mathbf{x}$

```
// Initialize intermediate result tables
```
Let $\mathbf{R}_i \leftarrow \mathbf{R} \cap attr(\theta_i)$ for $i = 1, \ldots, m$;
Let $(\mathbf{R}_{j_1}, F_{j_1}), \ldots, (\mathbf{R}_{j_k}, F_{j_k})$, $k \leq m$, be a partitioning of F according to $\mathbf{R}_i$;
foreach *partition* $(\mathbf{R}_j, F_j)$ **do**
  $\tilde{\mathbf{x}}^j \leftarrow$ empty table with schema $(\mathbf{R}_j, C_{j_1}, \ldots, C_{j_{k_j}})$;
  Create an index on $\tilde{\mathbf{x}}^j$ over the attributes $\mathbf{R}_j$;
  $\tilde{\theta}_j(r, b) = \bigwedge_{A \in \mathbf{R}_j} r.A = b.A$;

```
// Scan detail table r and update intermediate result tables
```
foreach *tuple* $r \in \mathbf{r}$ **do**
  foreach *partition* $(\mathbf{R}_j, F_j)$ **do**
    if $\exists \tilde{x} \in \tilde{\mathbf{x}}^j$ *such that* $\tilde{\theta}_j(r, \tilde{x})$ **then**
      $\tilde{x}.C_{j_i} \leftarrow g_{j_i}(\tilde{x}.C_{j_i}, f_{j_i}(\{r\}))$ for $i = 1, \ldots, k_j$;
    else
      $\tilde{\mathbf{x}}^j \leftarrow \tilde{\mathbf{x}}^j \cup \{r.\mathbf{R}_j \circ (f_{j_1}(\{r\}), \ldots, f_{j_{k_j}}(\{r\}))\}$;

```
// Build final result table x
```
$\mathbf{x} = \mathbf{b} \times \{(v_1, \ldots, v_m)\}$;
Create index on $\mathbf{x}$ over attributes $\mathbf{B}$;
for $i = 1$ **to** m **do**
  foreach $\tilde{x} \in \tilde{\mathbf{x}}^i$ **do**
    foreach $x \in \mathbf{x}$ *such that* $\theta_i(\tilde{x}, x)$ **do**
      $x.C_i \leftarrow g_i(x.C_i, \tilde{x}.C_i)$;

return $\mathbf{x}$;

The algorithm starts with the initialization of empty intermediate result tables. According to Proposition 2, for each $\tilde{\theta}_i$ a separate table is created. This leads to tables with identical grouping attributes if different θ_is reference the same attributes in $\mathbf{R}$, i.e., $\mathbf{R}_i = \mathbf{R}_j$ for $i \neq j$. Therefore, in the algorithm we apply a further optimization and merge tables with identical grouping attributes to a single table with one column for each aggregate function. For each intermediate result table, $\tilde{\mathbf{x}}^j$, constructed in this way we create an index over the grouping attributes. The conditions $\tilde{\theta}_j$ are generated as described in Proposition 1. Next, the detail table is scanned, and for each $r \in \mathbf{r}$ the aggregates in the intermediate result tables, $\tilde{\mathbf{x}}^j$, are updated. If an entry in $\tilde{\mathbf{x}}^j$ exists that matches tuple r, the aggregates are incrementally updated. Otherwise, a new entry is created and the aggregate values are initialized to the functions evaluated over r. Finally, the result table $\mathbf{x}$ is initialized to $\mathbf{b}$ with the aggregates initialized to the neutral

values v_i. The final result table is computed by combining the partial aggregates from the intermediate result tables $\tilde{\mathbf{x}}^j$ using the super aggregates as described in Proposition 2. For that the intermediate result tables are scanned and the aggregate values in the final result table are incrementally updated.

Figure 6 illustrates a few steps of the computation of Query Q1. Empty intermediate result tables $\tilde{\mathbf{x}}^1$ and $\tilde{\mathbf{x}}^2$ are created for the partitions $(\{D\}, \{f_1\})$ and $(\{D, U\}, \{f_2\})$. The first tuple r_1 creates a new entry in both tables. Tuple r_2 creates no new entries in any of the intermediate tables, it only updates aggregates. Tuple r_3 updates $C1$ in $\tilde{\mathbf{x}}^1$ and creates a new entry in $\tilde{\mathbf{x}}^2$. After processing r_{10}, the intermediate tables contain the same partial aggregate values as in Fig. 5.

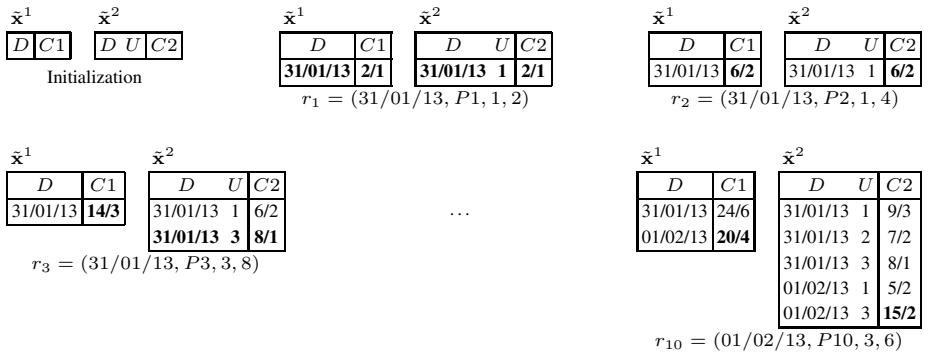


Fig. 6. Processing of Detail Tuples and Computation of Intermediate Result Tables in TCMDA$^+$

Complexity Analysis. We analyze the complexity of the TCMDA$^+$ algorithm in terms of incremental updates to the aggregate values and compare it to the original TCMDA algorithm [1]. As parameters we consider the two input relations, $\mathbf{b}$ and $\mathbf{r}$, while the number of aggregate functions and θ-conditions are considered to be constant.

The complexity of TCMDA is $C_{\texttt{TCMDA}} = |\mathbf{b}| + |\mathbf{r}| \cdot u$, where u is the average number of updates in the result table $\mathbf{x}$. The number of updates depends on the constraints in the θ_is and ranges between 0 and $|\mathbf{b}|$. For range aggregates, u is much higher than for point aggregates. The complexity of TCMDA$^+$ is $C_{\texttt{TCMDA}^+} = |\mathbf{b}| + |\mathbf{r}| + |\tilde{\mathbf{x}}| \cdot u$, where $|\tilde{\mathbf{x}}|$ is the size of the largest intermediate result table and u is the average number of updates in the result table $\mathbf{x}$. The number of updates for each $r \in \mathbf{r}$ in the intermediate tables $\tilde{\mathbf{x}}^i$ is always one due to the reduction to point aggregates and the use of separate intermediate tables. The computation of the final result table $\mathbf{x}$ requires on average u updates for each $\tilde{x} \in \tilde{\mathbf{x}}$, where $|\tilde{\mathbf{x}}| \leq |\mathbf{r}|$ and u ranges between 1 and $|\mathbf{b}|$.

The worst case complexity of TCMDA is $\mathcal{O}(|\mathbf{r}| \cdot |\mathbf{b}|)$. For the TCMDA$^+$ algorithm, we distinguish three cases. First, for $|\mathbf{b}| < \sqrt{|\mathbf{r}|}$ and $|\tilde{\mathbf{x}}| \approx |\mathbf{b}|$, which is common for OLAP, we get a worst case complexity of $\mathcal{O}(\mathbf{r})$, and thus a significant improvement over TCMDA. Second, for $|\mathbf{b}| > \sqrt{|\mathbf{r}|}$ and $|\tilde{\mathbf{x}}| \approx |\mathbf{b}|$ we have $\mathcal{O}(|\mathbf{b}|^2)$. Third, if $|\tilde{\mathbf{x}}| \gg |\mathbf{b}|$ (or almost as large as $|\mathbf{r}|$) the algorithm degrades to $\mathcal{O}(|\mathbf{r}| \cdot |\mathbf{b}|)$ and has the same complexity as TCMDA.

To summarize, for typical applications of θ-MDA, where $|\mathbf{b}| < \sqrt{|\mathbf{r}|}$ and $|\tilde{\mathbf{x}}| \approx |\mathbf{b}|$, the proposed optimization reduces the complexity of the θ-MDA evaluation from $\mathcal{O}(|\mathbf{r}| \cdot |\mathbf{b}|)$ to $\mathcal{O}(|\mathbf{r}|)$ for both range and point aggregates.

6 Experiments

Setup and Data. We implemented the algorithms TCMDA from [1] and TCMDA$^+$ described in this paper in C using Oracle 11g for storing the data. The experiments run on a machine with two AMD Opteron processors (1.8 GHz and 2.6 GHz), 16 GB of main memory, and Ubuntu 10.04. For the experiments we used the *Orders* table of the TPC-H benchmark[1]. We generated tables of different size and ran queries over them using the aggregate function COUNT.

Varying the Size of the Detail Table. Figure 7 presents the runtime by varying the size of the detail table between 2 and 10 million tuples. The θ-conditions use the operators $\leq$ and $\neq$ (range aggregates). In all experiments, TCMDA$^+$ clearly outperforms TCMDA, and the runtime of TCMDA grows faster than for TCMDA$^+$. This improvement of up to a factor of five can be attributed to the reduction to point aggregates, which reduces the number of updates for each detail tuple to one. As expected, the less selective the θ-constraints are, the bigger the performance improvement since for less selective conditions TCMDA needs to update comparably more aggregates for each $r \in \mathbf{r}$.

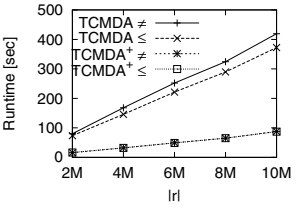
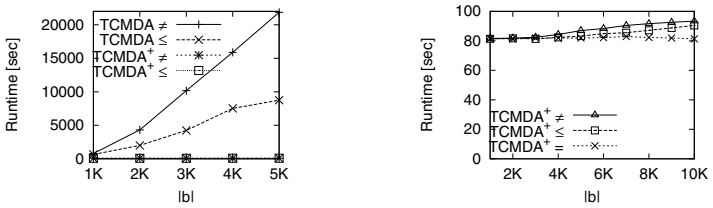


Fig. 7. Varying $|\mathbf{r}|$ ($|\mathbf{b}| = 500$, $|\Theta| = 1$, $|\theta| = 1$)

Varying the Size of the Base Table. The experimental results with a varying size of the base table are shown in Fig. 8 and exhibit an even better performance improvement than for varying $|\mathbf{r}|$. The growing size of $\mathbf{b}$ affects the runtime of TCMDA drastically, as shown in Fig. 8(a). For $|\mathbf{b}| = 5000$ the runtime is about 13 times larger for $\leq$ and about 27 larger for $\neq$ with respect to $|\mathbf{b}| = 1000$. In contrast, the runtime of TCMDA$^+$ is not affected by the growing base table. Figure 8(b) shows the runtime of TCMDA$^+$ with a larger base table varying between 2000 and 1000 tuples. The experiment confirms our analytical results that the runtime of TCMDA$^+$ is growing slowly for $\leq$ and $\neq$ queries when $|\mathbf{b}| > \sqrt{|\mathbf{r}|} \approx 3000$; for $=$ (point queries) the runtime remains constant.

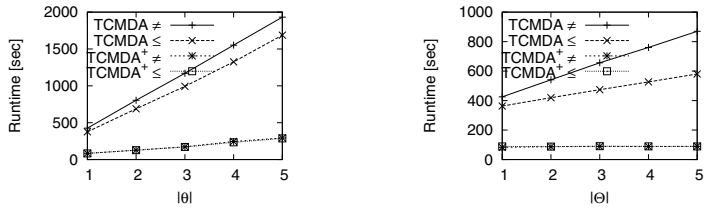
[1] TPC-H benchmark framework: http://www.tpc.org/tpch/



(a) Both algorithms (b) Only TCMDA$^+$

Fig. 8. Varying $|\mathbf{b}|$ ($|\mathbf{r}| = 10M$, $|\Theta| = 1$, $|\theta| = 1$)

Varying the Conditions. In Fig. 9, the number of constraints in the conditions θ_i and the number of θ-conditions in Θ are varied, respectively. Again, TCMDA$^+$ clearly outperforms TCMDA. The runtime for θ-conditions with a number of constraints that varies between 1 and 5 is shown in Fig. 9(a). The increase in the runtime is due to the more expensive evaluation of the θ-conditions containing more constraints. Fig. 9(b) illustrates the experimental results for a growing number of conditions in Θ. A higher number of conditions results in more aggregates to be computed, hence more incremental updates are required. For TCMDA, additional θ-conditions result in significantly more aggregates to be updated for each $\mathbf{r}$-tuple. TCMDA$^+$ is less affected because each additional θ-condition means only one more aggregate update for each $\mathbf{r}$-tuple.

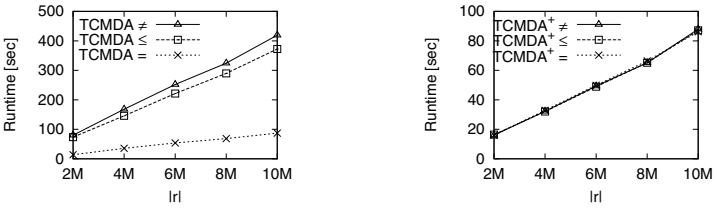


(a) Varying Size of θs (b) Varying Number of θs

Fig. 9. Varying Conditions ($|\mathbf{r}| = 10M$, $|\mathbf{b}| = 500$, $|\Theta| = 1$)

Comparing Different Constraint Types. In Fig. 10, we analyze how the constraint operators $=$, $\leq$, and $\neq$ affect the runtime (i.e., range vs. point aggregates). TCMDA is very sensitive to the type of constraint operators that are used, yielding runtimes that for $\leq$ and $\neq$ are much longer than for $=$ (see Fig. 10(a)). This is due to the higher number of aggregate updates for each $\mathbf{r}$-tuple in range aggregates (i.e., lower selectivity of the operator). In contrast, TCMDA$^+$ in Fig 10(b) is robust and independent of the type of constraint operators. Range aggregates are computed with the same efficiency as point aggregates.

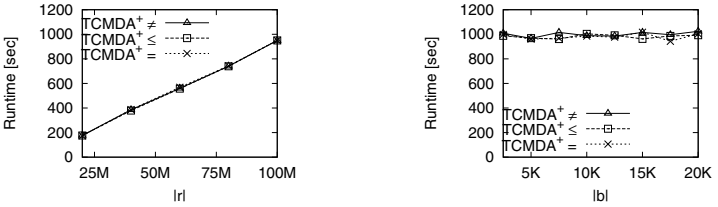
Figure 11 analyzes TCMDA$^+$ and different types of constraints for larger detail and base tables. Both graphs confirm the analytical results, i.e., the runtime of TCMDA$^+$ is linear in $|\mathbf{r}|$ when $|\mathbf{b}| \leq \sqrt{|\mathbf{r}|} = 10000$. In Fig. 11(a), the detail table varies between



(a) TCMDA (b) TCMDA$^+$

Fig. 10. Varying $|\mathbf{r}|$ ($|\mathbf{b}| = 500$, $|\Theta| = 1$, and $|\theta| = 1$)

25 and 100 million tuples, with a base table size of 1000 and one θ-condition with one constraint. The runtime shows a linear growth and is not affected by the type of the query. In Fig. 11(b), the size of the base table varies between 2500 and 20000 tuples, with a detail table of 100 million tuples and one θ-condition with one constraint. The runtime is constant for $|\mathbf{b}| \leq 10000$. For larger base tables $\mathbf{b}$, the runtime is slowly increasing, though the increase is less evident than in Fig. 8(b), where the base table is comparably larger.



(a) Varying $|\mathbf{r}|$ ($|\mathbf{b}|$=1000) (b) Varying $|\mathbf{b}|$ ($|\mathbf{r}|$=100M)

Fig. 11. TCMDA$^+$ Scalability Experiments

7 Conclusions

In this paper, we studied the efficient evaluation of complex ad-hoc multidimensional aggregation queries with the θ-MDA operator. We proposed a solution to reduce the evaluation of θ-MDA range aggregates to θ-MDA point aggregates. Point aggregates require significantly less incremental updates. This optimization has been integrated into a new evaluation algorithm, termed TCMDA$^+$, which reduces the runtime complexity from $\mathcal{O}(|\mathbf{r}| \cdot |\mathbf{b}|)$ to $\mathcal{O}(|\mathbf{r}|)$ if $|\mathbf{b}| < |\sqrt{\mathbf{r}}|$ and $\tilde{\mathbf{x}} \approx \mathbf{b}$, which is common in OLAP queries, such as for the TPC-H benchmark. Extensive experiments have shown performance improvements of more than an order of magnitude for TCMDA$^+$, and range aggregates can be computed with almost the same performance as point aggregates.

Future work points in several directions. First, we will investigate the integration of θ-MDA as an algebraic operator into the kernel of PostgreSQL. Second, we plan to adapt the evaluation strategy for θ-MDA queries and leverage MapReduce techniques for distributed query processing. Finally, it could be interesting to identify applications with very large base tables and develop optimization strategies for such settings.

Acknowledgments. We are indebted to Andreas Heinisch for providing us the C implementation of the algorithms and his support to run the experiments.

This work was funded in part by the Swiss National Science Foundation (SNSF) through the Tameus project (proposal no 200021_135361) and the Autonomous Province of Bozen-Bolzano through the AQUIST project.

References

1. Akinde, M., Böhlen, M.H., Chatziantoniou, D., Gamper, J.: θ-constrained multi-dimensional aggregation. Information Systems 36, 341–358 (2011)
2. Akinde, M., Chatziantoniou, D., Johnson, T., Kim, S.: The MD-join: An operator for complex OLAP. In: Proceedings of ICDE, Washington, DC, USA, pp. 524–533 (2001)
3. Akinde, M.O., Böhlen, M.H.: Generalized MD-joins: Evaluation and reduction to SQL. In: Jonker, W. (ed.) Databases in Telecommunications II. LNCS, vol. 2209, pp. 52–67. Springer, Heidelberg (2001)
4. Akinde, M.O., Böhlen, M.H., Johnson, T., Lakshmanan, L.V.S., Srivastava, D.: Efficient OLAP query processing in distributed data warehouses. Information Systems 28, 111–135 (2003)
5. Chun, S.-J., Chung, C.-W., Lee, J.-H., Lee, S.-L.: Dynamic update cube for range-sum queries. In: VLDB, pp. 521–530 (2001)
6. Geffner, S., Agrawal, D., Abbadi, A.E., Smith, T.R.: Relative prefix sums: An efficient approach for querying dynamic olap data cubes. In: ICDE, pp. 328–335 (1999)
7. Gray, J., Bosworth, A., Layman, A., Reichart, D., Pirahesh, H.: Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. In: Proceedings of ICDE, Washington, DC, USA, pp. 152–159 (1996)
8. Gupta, A., Harinarayan, V., Quass, D.: Aggregate-query processing in data warehousing environments. In: VLDB, pp. 358–369 (1995)
9. Harinarayan, V., Rajaraman, A., Ullman, J.D.: Implementing data cubes efficiently. In: Proceedings of SIGMOD, New York, NY, USA, pp. 205–216 (1996)
10. Ho, C.-T., Agrawal, R., Megiddo, N., Srikant, R.: Range queries in OLAP data cubes. In: Proceedings of SIGMOD, Tucson, Arizona, USA, May 13-15, pp. 73–88 (1997)
11. Hurtado, C.A., Mendelzon, A.O., Vaisman, A.A.: Maintaining data cubes under dimension updates. In: ICDE, pp. 346–355. IEEE Computer Society (1999)
12. Lee, K.Y., Kim, M.H.: Efficient incremental maintenance of data cubes. In: Proceedings of the VLDB Conference, pp. 823–833 (2006)
13. Lehner, W., Sidle, R., Pirahesh, H., Cochrane, R.: Maintenance of automatic summary tables. In: Proceedings of SIGMOD, pp. 512–513 (2000)
14. Liang, W., Wang, H., Orlowska, M.E.: Range queries in dynamic OLAP data cubes. Data Knowl. Eng. 34(1), 21–38 (2000)
15. Mumick, B.S., Quass, D., Mumick, B.S.: Maintenance of data cubes and summary tables in a warehouse. In: Proceedings of SIGMOD, New York, NY, USA, pp. 100–111 (1997)
16. Sridhar, R., Ravindra, P., Anyanwu, K.: RAPID: Enabling scalable ad-hoc analytics on the semantic web. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 715–730. Springer, Heidelberg (2009)

SPIN: Concurrent Workload Scaling over Data Warehouses

João Pedro Costa and Pedro Furtado

[1] ISEC-Institute Polytechnic of Coimbra
jcosta@isec.pt
[2] University of Coimbra
pnf@dei.uc.pt

Abstract. Increasingly, data warehouse (DW) analyses are being used not only for strategic business decisions but also as valuable tools in operational daily decisions. As a consequence, a large number of queries are concurrently submitted, stressing the database engine ability to handle such query workloads without severely degrading query response times. The query-at-time model of common database engines, where each query is independently executed and competes for the same resources, is inefficient for handling large DWs and does not provide the expected performance and scalability when processing large numbers of concurrent queries. However, the query workload, which is mainly composed of aggregation star queries, frequently has to process similar data and perform similar computations. While materialized views can help in this matter, their usefulness is limited to queries and query patterns that are known in advance. The reviewed proposals on data and processing sharing suffer from memory limitations, reduced scalability and unpredictable execution times when applied to large DWs, particularly those with large dimensions. We present SPIN, a data and processing sharing model that delivers predictable execution times to aggregated star-queries even in the presence of large concurrent query loads, without the memory and scalability limitations of existing approaches. We used the TPC-H benchmark to experimentally evaluate SPIN.

1 Introduction

Current database engines execute queries following a query-at-a-time model, with every query being independently processed without any data and processing sharing considerations, with each competing for resources (IO, CPU, …). While this may not raise performance issues for most operational systems, it is a performance killer when dealing with large Data Warehouses (DW). In this context, large fact and dimension relations are concurrently scanned by each query of the concurrently running workload, and tuples are independently filtered, joined and aggregated. This lack of data and processing sharing result in lack of scalability and the system is unable to provide predictable execution times. Therefore, predictable execution times under scalable data volumes and query workloads can only be attained through the use of high level of data and processing sharing among concurrently running queries.

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 60–71, 2013.

Recent proposals aimed to provide improved sharing. [1] focuses on sharing fact table reading and joining costs among running queries, by using a set of dimension filters to perform fact-to-dimension joins. However, the usefulness of the approach is limited to small dimensions that can fit entirely in memory and, as recognized in [2], large dimensions may severely impact performance.

We argue that there's a need for improved data and processing sharing among a large number of concurrently running star queries. Such approach should also deal with scalable data volumes and processing infrastructures.

In this paper we present SPIN, a data and processing sharing model that can deliver predictable execution times to a large set of concurrently running aggregation star queries. It has minimum memory requirements and can handle large data volumes and be deployed over scalable processing infrastructures with almost linear speedups. We discuss the SPIN characteristics and how they overcome the limitations of recent proposals on data and processing sharing, such as memory limitations, reduced scalability and unpredictable execution times when applied to large DWs, particularly those with large dimensions.

The paper is organized as follow: section 2 reviews related work on data and processing sharing, and their limitations on delivering scalable and predictable performance. Section 3 presents and discusses SPIN and how it can overcome such limitations. We evaluate SPIN in section 4, and finally we present conclusions in section 5.

2 Related Work

The usage pattern of DWs is changing from the traditional, limited set of simultaneous users and queries, mainly well-known reporting queries, to a more dynamic and concurrent environment, with more simultaneous users and ad-hoc queries. DW query patterns are mainly composed by star aggregation queries, which contain a set of query predicates (filters) and aggregations. Fig. 1 illustrates the query template.

```
SELECT dim attributes, aggregation functions
FROM   fact, set of dimension tables
WHERE  join conditions
AND    dim attribute conditions
GROUP BY dim attributes
```

Fig. 1. Template of an Aggregated Star Query

The query-at-a-time execution model of traditional RDBMS systems, where each query is executed following its own execution plan, does not provide a scalable environment to handle much larger, concurrent and unpredictable workloads. The use of large parallel deployments does not solve this problem because the additional computational and storage capabilities only lessen it, while introducing others problems such as load-balancing, optimal data distribution and network capacity. Queries submitted to a star schema model have common processing tasks, particularly those related to IO processing of the fact table (costly operations).

Analyzing the execution query plan, we observe that the low-level data access methods, such as sequential scan, represent a major weight in the total query execution time. One way to reduce such a burden is to store relations in memory. However, the amount of available memory is limited, insufficient to hold large DW, and is also required for performing join and sort operations.

Cooperative scans [3] enhances performance by improving data sharing between concurrent queries, by performing dynamic scheduling of queries and their data requests taking into account with the current executing actions. While this minimizes the overall IO costs, by mainly using sequential scans instead of a large number of costly random IO operations, and the number of scan operations (since scans are shared between queries), it introduces undesirable delays to query execution and does not deliver predictable query execution times.

QPipe [4] applies on-demand simultaneous pipelining of common intermediate results across queries, avoiding costly materializations and improving performance when compared to tuple-by-tuple evaluation.

CJOIN[1] [2] applies a continuous scan model to the fact table, reading and placing fact tuples in a pipeline, and sharing dimension join tasks among queries, by attaching a bitmap tag to each fact tuple, one bit for each query, and attaching a similar bitmap tab to each dimension tuple referenced by at least one of the running queries. Each fact tuple in the pipeline goes through a set of filters (one for each dimension) to determine if it is referenced by at least one of the running queries. It not, the tuple is discarded. Tuples that reach the en d of the pipeline (tuples not discarded in filters) are then distributed to dedicated query aggregations operators, one for each query.

CJoin overcomes the limitations of the query-at-a-time model, allowing high level of concurrency, with multiple concurrent queries being processed at a time, by scheduling processing tasks so that they can share IO, particularly scanning tasks. In this model, after creating the execution plan of multiple queries a pre-processor analysis the processing tasks and schedules them so that they can share processing tasks. This is applied not only to IO processing tasks but also to filtering and aggregation tasks, arranged in a pipelined fashion. The system is continuously scanning the fact table and tuples are put in a pipeline for processing. The pre-processor add a bit vector to each tuple that it receives, one bit for each query in the workload, before forwarding the tuples to the pipeline. A similar bit vector is also added to each dimension tuple, where each bit indicates if the dimension tuple satisfies the restrictions (of filtering conditions) of the corresponding query. This bit vector information is used to decide (filter) which fact tuples satisfy the conditions and should be forward into the pipeline.

While this approach reduces IO cost, it requires all dimension tables to reside in memory in order to be probed for performing hash joins, and to continuously update dimension bit vectors (with varying numbers of bits) when new queries are submitted or running queries have finished. However, in practice, dimension' sizes can be large. As a consequence, it may require external hash-joins and therefore resulting in slower performance and unpredictable query execution times.

We tackle the dimension size problem using a different approach, which has small memory requirements and can effectively be deployed into parallel shared nothing architectures composed of heterogeneous processing nodes. Our proposal, SPIN is conceptually related to CJoin, and QPipe in what concerns the continuous scanning of

fact data, but it uses a simpler approach with minimum memory requirements and does not have the limitations of such approaches. SPIN uses a de-normalized model, as proposed in [5] as a way to avoid join costs, at the expense of additional storage costs. Since it has fully data and processing scalability, ONE allows massive parallelization [6], which provides balanced data distribution, scalable performance and predictable query execution times. The CJOIN logic for small in-memory relations and the dynamic scheduling of cooperative scans can be integrated in SPIN.

3 The SPIN Processing Model

SPIN uses the de-normalized data model (ONE) proposed in [5], [6], this means that the star schema is physically organized as a single de-normalized relation (O_d). SPIN provides workload scale-out by combining (merging) data requests from all queries to be satisfied by a sequential continuous scan executed in a circular loop.

It views the ONE relation (O_d) logically as a circular relation, i.e. a relation that is constantly scanned in a circular fashion (when the end is reached, it continues scanning from the beginning). The relation is divided into a set of logical fragments (or chunks), with the chunk size adjusted to storage characteristics. The circular loop is continuously spinning, sequentially reading data chunks, while there are queries running. Data is read from storage and shared to all running concurrent queries, as illustrated in Fig. 2.

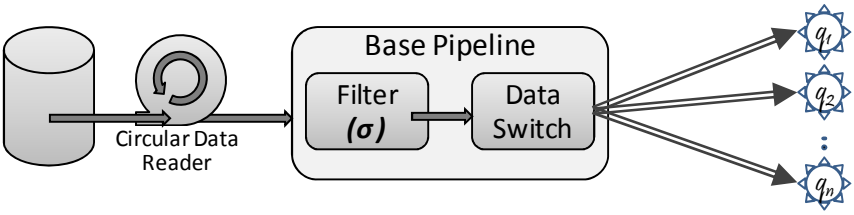


Fig. 2. – SPIN Data processing model

A Data Reader sequentially reads chunks of relation O_d, and continuously fills a data pipeline. Tuples when entering the pipeline, have to go through a fast and simple selection operator to early discard large subsets of tuples not requested by currently registered queries (early selection). Only tuples that are relevant for at least one query flows to a Data Switch (*DS*), which diverts tuples to each running query, building a dedicated logical branch for each query. For performance reasons, some fast early selection predicates are incorporated within the Data Reader.

Since O_d is de-normalized, no costly join tasks need to be processed, only query operations. Any query q, when submitted, starts consuming and processing the tuples that are currently in the data pipeline.

3.1 Query Registering and Processing

A Query Handler handles query (de)registering. Any query q registers as a consumer of the data in the pipeline, following a publish-subscribe model. Each query q, has to process every tuple from relation O_d, at most once, although data query processing does not need to start at record 0.

For each running query, a Query Handler maintains an indicator of the first logical tuple (position in the circular loop) consumed by the query. This logical first row position is fundamental to determine when the end of each query is reached. When that occurs, then the query q has considered all tuples for execution, therefore it terminates the query execution and sends the query results to the client. Fig. 3 illustrates the data reading circular process, depicting the logical position for queries q_1, q_2 and q_3.

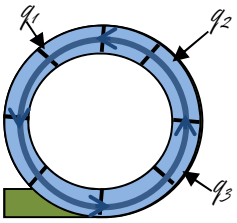


Fig. 3. SPIN sequential data reading loop

As each query starts consuming tuples from the current position (logical beginning) in the circular loop, without the need to start from a specific record, the reading cost (IO cost) is shared among all running queries Q_r without introducing additional IO overhead or random reads. Others costs related to query processing are at subsequent executing phases, such as selection, logical branching and pipeline processing.

When submitted, a query is analyzed and decomposed into a sequentially-organized set of predicates, computations and aggregations tasks. This decomposition into tasks allows SPIN to determine and update the set of early selection predicates that are placed at the base data pipeline. The remaining query tasks are mapped into SPIN operators for execution. To ensure a fast early selection phase, complex (costly) query predicates are placed at latter stages.

3.2 SPIN Operators and Data Processing Pipelines

SPIN follows a flow oriented processing model where each query is decomposed into tasks that are later mapped to operators placed along a query-specific processing pipeline. A processing pipeline is a collection of sequentially-organized operators that transform tuples as they flow along the pipeline (illustrated in Fig.4).

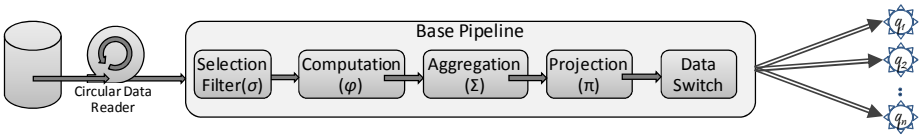


Fig. 4. SPIN Data pipeline processing model

SPIN include the following base operators:

- **Selection Operators (σ)** – apply predicate clauses to filter incoming tuples, trashing those that do not meet the predicate clauses. Each Selection Operation maps a query predicate (default) or a set of related query predicates. The selection operators are typically placed at the beginning of the processing pipeline, to filter tuples that pass-through the pipeline and go into subsequent processing operators. Selection Operators are placed in a sequential ordered fashion according to their selectivity and evaluation costs, with more restrictive and faster placed at early stages.
- **Projection Operators (π)** – restrict to a subset of tuple attributes that flows throw the pipeline. A projection and a selection operator can be combined into a single step operator.
- **Computation Operators (φ)** - perform tuple level data transformations, including arithmetic (e.g. a + b) and string manipulation (e.g. substring). A dedicated Computational Operator is built for each specific transformation. A Computation Operator maps a tuple-level arithmetic, function or operation expressed in any of the query clauses (e.g. the arithmetic expression *QUANTITY * PRICE*). The mapping into φ is particularly relevant for complex computations that appear in several query clauses. Query predicates that include computations (e.g. *QUANTITY * PRICE > 1000*) may be mapped into a σ, preceded by a φ that performs the computation. The goal is to build fast selection operation with simple and fast evaluation predicates.
- **Data Switches (DS)** –forward incoming data tuples into a set of data outputs, called **logical data branches (B)**. Tuples are forwarded to all branches b, $b \in B$, or forwarded according to each branch conditions. For each branch, a tuple is only forwarded if it matches the branch's conditions b_p (if exists). Tuples not matching any branch predicates are trashed.
- **Aggregation Operators (Σ)** – perform group by computations, by grouping tuples according to GROUP By clauses, and applying aggregation functions (e.g. *SUM*). Aggregation operators output results when all tuples for a given query as been considered for processing.

A **Data Processing Pipeline** (or simply referenced as pipeline) is a set of sequentially organized operators. A pipeline represents a set of common operations that need to be performed to incoming tuples. Each query is decomposed into tasks (e.g. filtering, aggregation) that are mapped into a set of sequential operators and placed a along a query-specific pipeline. Query processing only starts after the pipeline is built and a logical branch is registered as a consumer of the base data pipeline.

3.3 Workload Data Processing Tree and Logical Data Paths

The number of operators (e.g. selection and aggregation operators) and query-specific pipelines (one for each query) increases with the query load, and can rapidly exhaust memory and processing resources. With large concurrent query loads, queries may have common query predicates, computations, and aggregations, resulting in multiple similar operators (each doing its own computation) being placed in query-specific pipelines. The circular Data Reader shares the IO reading cost among queries, but SPIN exploits further data processing opportunities.

Groups of queries may share the same query predicates, computations or aggregations. For each query, SPIN splits the query-specific pipeline into an equivalent ordered set of sequentially connected partial pipelines. Each of these partial pipelines, composed with one or set of logically related operators, is connected as a data consumer of its predecessor. For the currently running query load, similar partial pipelines (with the same operators over the same tuples) from different queries are combined into a common pipeline and a data switch is appended to end to share its results. The subsequent connected data pipelines are then connected as logical branches of this common data pipeline, consuming its output. A set of parallel query-specific pipelines with common operators are rearranged in order to push–forward and orchestrate similar operators into a common processing pipeline. At the end of this pipeline, a *DS* diverts tuples to further processing in subsequent logical branches. Fig. 5 illustrates a SPIN processing layout with two logical branches composed by two query-specific processing pipelines connected to a common processing pipeline.

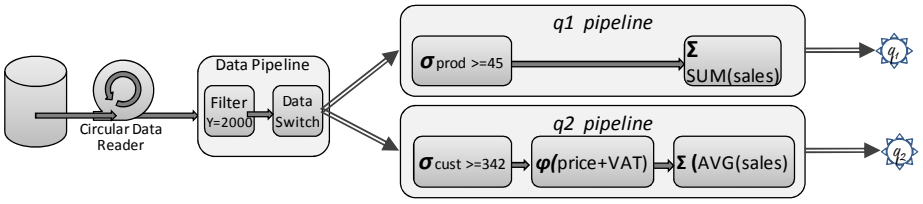


Fig. 5. SPIN Data pipeline processing model

3.4 Building the Workload Processing Tree

As a result, the initial query-specific processing pipelines of simultaneously running queries are split, merged and organized into a workload data processing tree. In the end, for each query there will be a logical data path traversing logical branches and pipelines that is equivalent to the initial query-specific data pipeline.

The merging of common data processing pipelines is enforced through all the query execution steps, to maximize data and processing sharing and reduce memory and processing usage. For instance, selection operator common to currently running queries are pushed closer to the Data Reader to reduce the data volume within the pipelines and thus increasing the level of data processing sharing. The base pipeline trashes tuples not required by any of its logical branches.

For example, consider that three similar aggregation queries are currently running, with different query predicates: $q_1\,(\sigma_{p=a})$, $q_2\,(\sigma_{y=2000})$ and $q_3\,(\sigma_{p=a\,\Lambda\,y=2000})$. An initial query-specific pipeline is built for each query which has to be connected to the base pipeline. Fig. 6 depicts the query-specific pipeline, one for each query.

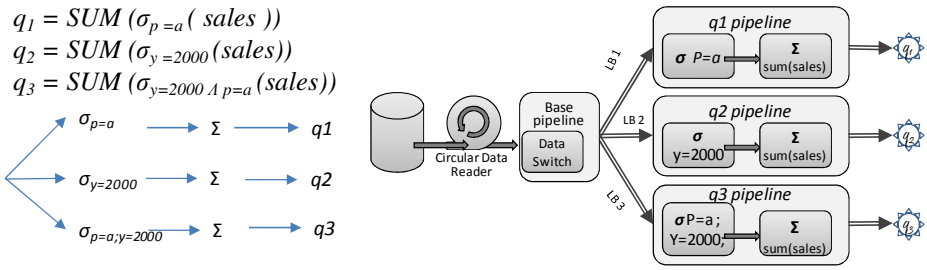


Fig. 6. SPIN deployment of query-specific pipelines

SPIN, when a new query is submitted, before connecting the corresponding query-specific pipeline to the base pipeline, tries to maximize data and processing sharing with the already running queries. It accomplishes this by maintaining and updating a workload processing tree, using the following steps:

1. The query-specific pipeline built for a query, is split into a set of sequentially connected partial-pipelines (the query logical data path - Q_{path})
2. If the current workload processing tree contains an equivalent logical data path (T_{path}), then the matching partial-pipelines are removed from Q_{path}, and the remaining are connected to the end of T_{path} as a new logical branch. Then the query starts processing tuples.
3. Otherwise, for each partial-pipeline
   3.1. the selection predicates are used to build a selection region
   3.2. Starting from the root of the workload processing tree, and while there's a logical branch with similar processing operators,
      3.2.1. If the pipeline selection region intercepts the selection region of partial-pipeline then the partial-pipeline is divided into two non-overlapping data pipelines, originating two distinct Q_{path}
      3.2.2. Otherwise, if the selection region of the current pipeline does not overlaps, but uses common predicate attributes, then it is registered as a logical branch of its predecessor. If required the selection predicates and switching conditions of this predecessor is updated accordingly.
   3.3. For each of the resulting Q_{path}, the algorithm goes to step 1.
4. if no matching point exists, the query Q_{path} is connected to the base pipeline

The number and placement of *DS*s, and logical branches, are orchestrated in order to minimize the switching cost DS_{cost}, the number of evaluated predicates, the predicate evaluation costs and the memory requirements for branch management. New logical branches are created and connected to *DS* when query predicates of processing pipelines queries do not match the predicates of the existing branches.

3.5 Merging and Reusing Intermediate Results

Afterwards, an optimization process transverses all logical branches to try to push-forward operators into preceding pipelines. The deployment of $\boldsymbol{\sigma}$ and *DS* is planned in order to reduce the data volume that traverse the pipelines, and maximize data and processing sharing. SPIN applies a merging process that analyses selection predicates and how processing and intermediate results can be shared among processing pipelines. For each logical data path, it follows the path backward to the source, and at each pipeline **P** of the data path, it determines if exists other logical paths that is processing, or has already processed, a subset of the tuples that this pipeline has to process. When a logical path **LP** exists, then this pipeline is divided in two sequential pipelines (**P1** and **P2**). The latter (**P2**) is connected to **P1** and **LP** and starts consuming their outputs and merging the results. The selection predicates of **P1** are updated to exclude the predicates of the logical path **LP**. This can result in multiple alternative branching deployments.

To evaluate these alternative deployments, and merging configurations, the merging process uses several data volume metrics: n_{tuples} as the number of relation tuples, n_{eval} as the total number of evaluated tuples by $\boldsymbol{\sigma}$s, and n_{ag} as the total number of aggregated tuples by $\boldsymbol{\Sigma}$s. For the initial deployment of the example, without considering considered merging, the number of evaluated and aggregated tuples are computed, respectively, as

$$n_{eval} = 3 \times n \text{ and } n_{ag} = n_{eval}\,(\sigma_{p=a}) + n_{eval}\,(\sigma_{y=2000}) + n_{eval}\,(\sigma_{p=a\,,\,y=2000})$$

After the merging process, the number of evaluated tuples n_{eval} is reduced from $n_Q \times n_{tuples}$ to $\Sigma\ n_{eval}\ (p_\sigma)$ with p_σ as each pipeline predicates. Fig.7 depicts the final deployment after the merging process.

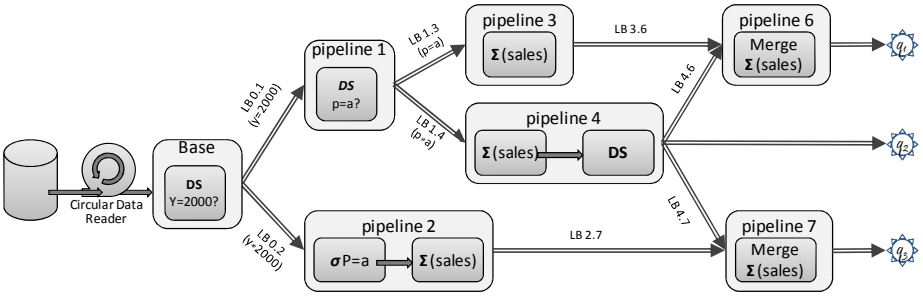


Fig. 7. Aggregation Branch processing

The number of aggregated tuples n_{ag} is also reduced from $\Sigma\ n_{eval}\ (\sigma_q)$ to $+\ n_{eval}\ (\sigma_{y=2000}) +\ n_{eval}\ (\sigma_{p=a\,,\,y=2000})$.The total number of evaluated and aggregated tuples are computed, respectively as

$$n_{eval} = n_{eval}\,(\sigma_{y=2000}) + n_{eval}\,(\sigma_{y=2000}) + n_{eval}\,(\sigma_{y\neq 2000}) = n_{tuples} + n_{eval}\,(\sigma_{y=2000})$$

$$n_{ag} = n_{eval}\,(\sigma_{p=a\,,\,y=2000}) + n_{eval}\,(\sigma_{p=a\,,\,y\neq 2000}) + n_{eval}\,(\sigma_{p\neq a\,,\,y=2000})$$

$$= n_{eval}\,(\sigma_{y=2000}) + n_{eval}\,(\sigma_{p=a\,,\,y\neq 2000})$$

3.6 Query Handling and Workload Processing Tree Reorganization

The data branching deployment is continuously reorganized as new queries are submitted. New queries that cannot be directly plugged into the workload processing tree are kept as distinct branches connected to the base data pipeline.

When a query finishes its execution, the Query Handler, removes from the workload processing tree the query-specific pipelines that are not used in other logical paths. Pipelines used by other logical paths are maintained, only the query-specific outputs are detached. Query-specific logical branches are detached and removed. Then a reorganization process is triggered to update the workload processing tree.

For instance, in the previous example (Fig. 7), when the query q3 finishes its execution, the query-specific pipeline 7 is detached from both pipeline 2 and 4 before being removed. As a consequence, pipeline 2 is also removed and the base pipeline switch is replaced with the switch and logical branches in pipeline 1. Then pipeline 1 is also removed. Operators in use by other currently running queries are updated to reflect the removal of query-specific clauses. A data branching reorganization process is triggered to determine if a better logical branching deployment can deliver improved performance.

For lack of space, several SPIN optimization mechanisms, such as: data columnar storage organization, compression, partial de-normalized relations with in-memory dimensions, of massive data sharding, data loading and snapshot isolation are left out of this paper and will be discussed in other works.

4 Evaluation

We developed in Java a SPIN implementation to evaluate its performance and scalability capabilities. The experimental results presented in this section were obtained using the release 1.5.1 (November 2012), which has about 30klocs and 140 java classes, and physically store tuples in row-wise format. For benchmarking purposes we used TPC-H benchmark, and built several variants of the query Q5 with different selectivity and aggregation groups, to evaluate their impact in performance and scalability. In this paper, the SPIN setup manages a full de-normalized relation as proposed in [5], with disabled optimization features (e.g. compression, materialized views, automatic in-memory bit-selections). We evaluate SPIN performance using an Intel i5 processor, with 8GB of RAM and 3 SATAIII disks with 2Terabytes each, running a default Linux Server distribution. We also evaluated the TPCH setup on two distinct DBMS engines: PostgresSQL 9.0 [7] and DbmsX (a well known commercial RDBMS). The TPCH setup was populated with the TPC-H data generator tool (DBGEN) available at [8] and the ONE relation with a modified version that generates the de-normalized data as single flat file. We used an additional server, connected through a gigabit-Ethernet switch, for submitting a varying concurrent query load. The query load consisted in a total of 1000 aggregation star queries generated by a varying number of simultaneous concurrent clients. The depicted results were obtained as average of 30 runs.

4.1 Influence of Number of Queries in Query Performance

Query performance and throughput of common RDBMS that follow a query-at-time execution model, which is highly influenced by the number of concurrently running queries that being executed. In this setup we used a scale factor 10 (SF=10) and varied the number of concurrent running queries. The average execution time (a) and throughput (b) are depicted in Fig. 8.

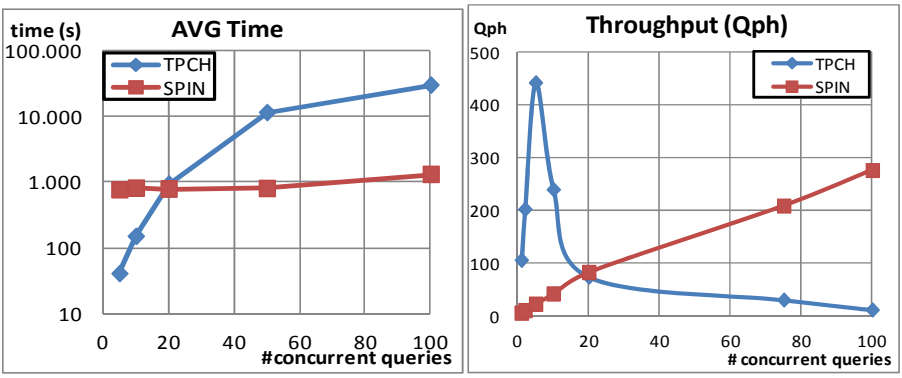


Fig. 8. Average execution time a) and throughput b) for varying query loads

We observe that with low concurrent query loads, the TPCH setup deliver better execution times and yields higher throughputs. However as we increase the number of running concurrent queries, the average execution time of TPCH decays significantly due to the increasing number of queries that are competing for resources. On the other hand, the average execution time with the SPIN setup shows a slightly increase with higher concurrent query loads. This is due to the pipeline management overheads and the cost of processing the query-specific pipelines that cannot be combined with other query pipelines. Regarding throughput, SPIN delivers an almost linear throughput.

4.2 Influence of Data Scale in Throughput

Throughput, besides the query load, is also influenced by the data volume. Fig. 9 depicts the throughput for two distinct data volumes: SF1 (a) and SF10 (b).

We were unable to timely obtain the results for TPCH for larger data volumes. In Fig. 9, we observe that throughput of SPIN increases almost linearly with the number of concurrent running queries, as more data and processing is shared among queries. With SF1, TPCH yields significantly higher throughput since all data and processing is done almost exclusively in memory. However as the number of concurrent queries increase we observe a significant drop in throughput as the running queries exhaust the available memory. SPIN, does not have this memory issues and can be massively partitioned among low-end commodity servers.

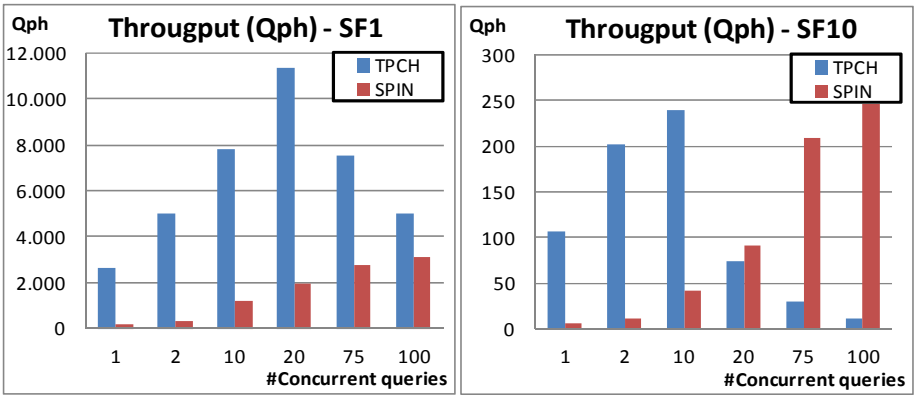


Fig. 9. Throughput for varying query loads with a) SF =1 and b) SF =10

5 Conclusions

We present SPIN, a data and processing sharing model that deliver predictable execution times to star-join queries even in the presence of large concurrent workloads, without the memory and scalability limitations of existing approaches. We used the TPC-H benchmark to evaluate SPIN and to show its ability to provide scalable performance and predictable execution times even in presence of large concurrent query loads.

Currently we are extending the SPIN processing model to deliver assured time guarantees with large parallel heterogeneous deployments, with massive data sharding.

References

[1] Candea, G., Polyzotis, N., Vingralek, R.: A scalable, predictable join operator for highly concurrent data warehouses. Proc. VLDB Endow. 2, 277–288 (2009)

[2] Candea, G., Polyzotis, N., Vingralek, R.: Predictable performance and high query concurrency for data analytics. The VLDB Journal 20(2), 227–248 (2011)

[3] Zukowski, M., Héman, S., Nes, N., Boncz, P.: Cooperative Scans: Dynamic Bandwidth Sharing in a DBMS. In: Proceedings of the 33rd International Conference on Very Large Data Bases, pp. 723–734 (2007)

[4] Harizopoulos, S., Shkapenyuk, V., Ailamaki, A.: QPipe: A Simultaneously Pipelined Relational Query Engine. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, pp. 383–394 (2005)

[5] Costa, J.P., Cecílio, J., Martins, P., Furtado, P.: ONE: a predictable and scalable DW model. In: Proceedings of the 13th International Conference on Data Warehousing and Knowledge Discovery, Toulouse, France, pp. 1–13 (2011)

[6] Costa, J.P., Martins, P., Cecílio, J., Furtado, P.: A Predictable Storage Model for Scalable Parallel DW. In: 15th International Database Engineering and Applications Symposium (IDEAS 2011), Lisbon, Portugal (2011)

[7] PostgreSQL, `http://www.postgresql.org/`

[8] TPC-H, TPC-H Benchmark (April 13, 2012), `http://www.tpc.org/tpch/`

Lily: A Geo-Enhanced Library for Location Intelligence

Matteo Golfarelli[1], Marco Mantovani[2], Federico Ravaldi[2], and Stefano Rizzi[1]

[1] DISI – University of Bologna,
V.le Risorgimento 2, 40136 Bologna, Italy
{matteo.golfarelli,stefano.rizzi}@unibo.it
[2] iConsulting Srl,
Via Bazzanese 32/7, 40033 Casalecchio di Reno (BO), Italy
{m.mantovani,f.ravaldi}@iconsulting.biz

Abstract. Location intelligence is a set of tools and techniques to integrate a geographical dimension into BI platforms, aimed at enhancing their capability of better monitoring and interpreting business events. Though most commercial data warehouse tools have implemented spatial extensions to support GIS integration, the user experience with spatial data is still mostly limited to the visualization of maps labeled with numerical indicators. To overcome this limit we developed *Lily*, a geo-enhanced library that adds true location intelligence capabilities to existing BI platforms. Lily provides end-users with a highly-interactive interface that seamlessly achieves a bidirectional integration between the BI and the geospatial worlds, so as to enable advanced analytical features that truly take into account the spatial dimension. In this paper we describe Lily from a functional and architectural point of view, and show an example where Lily is coupled with the Oracle Suite to be used for location intelligence in the field of telecommunications.

Keywords: Spatial data warehouse, Business intelligence, GIS.

1 Introduction

Over 80% worldwide companies take their business decisions based on data characterized by a spatial component [6]. This fact, together with the ongoing success of online services based on geo-localization such as Google Maps and Facebook Places, and with the widespread diffusion of smartphones and tablets (both capable of detecting the owner's exact location) has contributed to raise the interest of decision-makers in spatial analyses, that allow the relationship between events and their territorial distribution to be precisely established and effectively examined. In response to this need, *spatial data warehouses* have been emerging as an enabling technology for spatial analyses. A considerable research on spatial data warehouses has been done over the last years; the specific topics investigated range from conceptual modeling for spatial data [10,3] to spatial OLAP operators [11,4] and efficient processing of spatial data [17,12].

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 72–83, 2013.

In much the same way as business intelligence (BI) tools build on traditional data warehouses to overcome the limits of OLAP analyses and enable more advanced techniques such as data mining and what-if analysis, there is a need for building sophisticated applications on spatial data warehouses to benefit from spatial data in a broader sense. In this direction, we use the term *location intelligence* to denote a set of tools and techniques that integrate a geographical dimension into BI platforms, aimed at enhancing their capability of discovering patterns, opportunities, and risks, and at better monitoring and interpreting business events with specific reference to their territorial distribution. While geographical information systems (GISs) are used by expert users for operational tasks to achieve tactical benefits, location intelligence is meant to be used by business users for decision-making tasks to achieve strategical benefits.

The human mind processes visual patterns 60,000 times faster than tabular lists. So, location intelligence is really useful when huge amounts of data can be aggregated into numerical indicators whose graphical positioning on an interactive map allows spatial patterns to emerge more clearly than a simple tabular representation or a chart. On the other hand, using BI techniques to investigate complex phenomena on a map fed with consistent and integrated data leads to leave behind the purely spatial analyses made possible by GISs. In general, location intelligence can be applied whenever there is a need for discovering meaningful correlations between phenomena described by indicators that share nothing but spatial proximity.

Spatial extensions have been implemented on most commercial data warehouse tools in order to integrate them with GISs [8,9]. However, the user experience with spatial data is still mostly limited to the visualization of maps labeled with numerical indicators of business, which hardly enables decision-makers to take full advantage of the huge information power lying with geography. To overcome this limit we developed *Lily* (Location Intelligence Library), a geo-enhanced library that adds true location intelligence capabilities to existing BI platforms. Lily is written in Javascript and and leverages the AJAX technologies to provide end-users with a highly-interactive interface that seamlessly achieves a bidirectional integration between the BI and the geospatial worlds, so as to enable advanced analytical features (such as what-if analysis and data mining, etc.) taking truly into account the spatial dimension.

In this paper we describe Lily from a functional point of view (Section 4), show an architecture for coupling Lily with the Oracle Suite (Section 5), and discuss an example where Lily is used for location intelligence in the field of telecommunications (Section 6). The paper outline is completed by Section 2, that discusses the related works, Section 3, where the main industrial solutions for location intelligence are classified, and Section 7, that draws the conclusions.

2 Location Intelligence in the Literature

Although researches in spatial data warehouse have been pioneered since 1997, when the term *SOLAP* —Spatial OLAP— has been coined [1], only in recent

years the research community thoroughly investigated the integration of GIS and OLAP technologies. The main topics that have been investigated are the *architectures*, the *data models*, and the *operators and algorithms* necessary to express and optimize SOLAP queries.

As to the architectures, in [2] the authors classify the solutions according to the type of coupling between the GIS and data warehouse components. *Loosely-coupled* architectures [7] rely either on import-export-reformatting or mapping of data between GIS and OLAP, *semi-tightly coupled* ones are either GIS-dominant or OLAP dominant solutions, while *tightly-coupled* architectures rely on fully integrated spatial OLAP technology. While the last type of architecture provides for larger querying capabilities, the supporters of the first type assert that it favors autonomy, updating, and maintenance of the databases. According to [13] a tight-coupling architecture should be organized on three-levels, characterized by a multidimensional/spatial query engine that translates visual queries to a relational-multidimensional DBMS supporting spatial measures and attributes. In this direction the GeoMDQL architecture is based on a spatial data warehouse, a multidimensional and spatial engine, and a GUI [16].

Research on data models is aimed at enriching the multidimensional model to properly support SOLAP queries. Stefanovic et al. classify spatial dimension hierarchies according to their spatial references in non-geometric, geometric to non-geometric, and fully geometric [17]. Malinowski and Zimányi focus on the conceptual representation of a spatial cube, that is obtained by extending their MultiDimER model with pictograms that represents spatial dimensions, levels, measures, and relationships [10]. In [16], a UML-based formalization for a spatial data warehouse meta-model is proposed that includes a large set of stereotypes modeling the spatial concepts (e.g., points, polygons, and lines).

As to the topics related to querying, many efforts have been done to increase the expressiveness of traditional OLAP operators and languages. In [16] the authors propose GeoMDQL, a query language based on MDX [18] and OGC (Open Geospatial consortium) for querying spatial data cubes. Also GISOLAP-QL [7] extends MDX, but while GeoMDQL relies on a tightly-coupled architecture, GISOLAP-QL relies on a loosely-coupled one. The type of coupling is reflected in the query structure, that is composed of two parts divided by a pipe: the GIS part returns the objects that satisfy a given condition, while the OLAP part performs an aggregation over the resulting objects. In [3,4] the authors propose the GeoCube algebra that extends SOLAP with five new operators, i.e., classify, specialize, permute, OLAP-buffer, and OLAP-overlay. In addition to classical drill and slice OLAP operators, GeoCube provides two operators for navigating the measure hierarchy and two spatial analysis operators that dynamically modify the structure of the spatial cube.

Though several Spatial OLAP Visualization and Analysis prototypes have been developed, they have been mostly used for verifying the expressiveness of the models and languages discussed so far and do not provide advanced GUIs. The GeoMDQL GUI [18] extends JPivot and is capable of displaying results in charts, tables, and maps but it is not described in detail, making it hard to

understand how advanced the interaction with the system is. SOVAT [15] combines OLAP and GIS capabilities; its interface supports spatial drill-down and roll-up operations as well as *drill-out*, that allows users to submit queries based on both numerical and spatial aggregation. Finally, in [14] a two-level prototype (JMap) is proposed where the GUI and the query engine are collapsed in the same module. JMap supports both tabular and diagrammatic views; different types of maps can be shown, in particular, multimaps, complex thematic maps, and maps with superimposed diagrams. JMap also allows synchronization of operations and symbols from one display to another, so the same information can be visualized from different perspectives.

3 Location Intelligence in the Industry

In most BI implementations, location information is present in the form of a coarse-grained (and often static) dimension [5]. Of course, this is not sufficient for location intelligence. So, during the last few years, the main vendors of BI systems and GISs have been progressively moving towards location intelligence. The basic solutions consists of a set of static maps (usually based on pre-defined SVG images), graphically annotated with simple indicators and used together with traditional reports for a more effective visualization of information. They are specifically built for specific analyses, poorly reusable, and with limited or no interaction. Examples of this kind of solutions can be found in many reporting tools such as Hyperion Web Analysis, Microsoft Reporting Services, and Business Objects Xcelsius.

Some more advanced solutions provide a communication interface between legacy BI platforms and GISs by enabling them to share the same analysis context. These solutions have started emerging recently, as a result of joint collaborations between BI and GIS vendors. This is the case of APOS LIS (APOS Location Intelligence Solution), that integrates SAP Business Objects and ESRI ArcGIS; of Business Geografic GeoQlik and GeoBI, that integrate a custom GIS with QlikView and SAP Business Objects respectively; of the bridges by Galigeo, that make their GIS communicate with BI plaftorms such as SAP Business Objects, IBM Cognos, and Microsoft; of ESRI Maps for IBM Cognos. Though bridges offer advanced capabilities for interacting with maps (they use fully-featured, state-of-the-art GIS tools), they do not achieve a true integration between spatial and business data, because users still operate on two separate systems: the BI tool contains all the business data, while spatial information resides on the GIS side.

The most advanced solutions rely on fully-integrated spatial OLAP technology to provide larger querying capabilities and better scalability and efficiency. The basic idea is that spatial data are stored together with business data in a spatial data warehouse; the main solutions in this category are Oracle Database (with its Spatial option), Microsoft SQL Server (the recent 2012 release introduces many new spatial features), and PostgreSQL (with its PostGIS extension). A unified front-end may then be available to produce fully-integrated OLAP-spatial reporting, like done by Oracle Business Intelligence in the Oracle suite.

4 Lily: A Functional View

With reference to the design cycle of a location intelligence system, Lily comes into play during the phase where user applications are developed. The functionalities provided by Lily can be classified as follows:

1. **Geo-enhanced query formulation**.
   - *Spatial drill*, that provides guided geography-based navigation of a domain of interest from a high-level perspective to the deepest detailed information available. This enables an analysis of a given numerical indicator either by means of a standard territorial hierarchy —beginning with the country level to proceed with Nielsen areas, regions, provinces, municipalities, and so on— or by implementing a custom-based hierarchy (custom sales areas, districts, etc.).
   - *SOLAP queries*. This function supports the formulation of queries by actively involving the geographical dimension. In particular, proximity-based slice&dice enables proximity analyses of business phenomena, such as the selection of a set of points of sales, real estate properties, or crime locations. End-users may use prefixed shapes or draw custom areas directly onto a map and use this selection tool as a way to filter the subjects of interest or as a benchmark for comparison purposes. In this way the system can answer questions such as: *Show the locations of the crimes that were suffered from victims aged less than 18 within a 10-mile radius from a school (or a river, a car park, a sports centre, a tube station, etc.).*
   - *Geo-coding*. The geocoding process is necessary to handle points as spatial objects, and consists in translating an unstructured alphanumerical address into a couple of latitude-longitude coordinates to store them into a spatial point format, and vice versa, i.e., inferring the address of a place from its geographic coordinates. Lily can leverage different geocoding mechanisms: the most commonly used is through the public web-services exposed by vendors such as Nokia, Tomtom, and Google (the web service accepts alphanumerical addresses as input and returns latitude/longitude coordinates together with quality information, e.g., exact match, street-level match, city-level match); another option, more efficient for large amounts of data, is to exploit the native geocoding capabilities of spatial-enabled DBMSs (in this case, address-level information must be loaded into the database).
2. **Geo-enhanced processing**.
   - *Spatial triggering*. This feature enables end-users to actively interact with maps by triggering custom scripts, which allows developers to associate complex processing (such as data mining, forecasting, what-if analysis, etc.) to any map area. Custom complex algorithms can be implemented to leverage mathematical models or data mining features, so it is

possible to simulate scenarios and gain forecast insights for questions like: *What are the distinguishing features of the areas with highest crime rate? Which areas show these features to some extent? Are the schools in a specific district ready for plan enrollment and student services for next year? Where is the best location for a new store, based on the market territory potential?*. In general, it is possible to trigger any custom action (an algorithm or just a specific visualization) in response to the selection of any feature of interest on the map.

- *Spatial clustering*. This function enables the detection and graphical representation of clusters of features of interest to provide an holistic view of a phenomenon, such as the distribution of residents, pollution, politicians, or any custom function that combines elementary data onto a wide geographic area. The same concept can be applied to *in-doors analysis*: *How do customers move inside a shopping centre, an airport, a museum, or a casino? How do customers visit my stores —i.e., where do they go, how long do they remain in a specific section, which areas are rarely visited, etc.?*

3. **Geo-enhanced data visualization**.
   - *Integration with external maps*. External maps are typically used as background imagery, or to visually identity correlations between different data sets. They are integrated at run-time as raster images (tiles), in the form of layers that can be stacked below (or on top of) the internal vector (interactive) or raster (static) themes; to achieve good performances, all vector themes must be based on spatial data residing on the spatial data warehouse. Lily is compatible with several map providers such as Google Maps, Bing Maps, OpenStreetMap, Ordinance Survey and, in general, with all providers using the Web Map Service (WMS) and Tile Map Service (TMS) standards. Using the same mechanism, it is also possible to interact with custom base-maps hosted on in-house GIS systems.
   - *Real-time refresh*. Lily supports real-time visualization of spatial data, not only through periodical refresh of the whole map, but also by triggering updates of colors or by smoothly reproducing the movements of objects (e.g., for vehicle tracking purposes), using the asynchronous AJAX technology. This is useful for instance to monitor territory sensors in real-time, or when monitoring road traffic accidents and congestions.
   - *Multi-layer representation*. Several layers can be overlapped onto the same map to represent different but complementary phenomena such as the distribution of residents, customers, roads, stores, and competitors. These layers can be related to different dimensions or different hierarchical levels.
   - *Spatial KPI visualization*. KPIs (*key performance indicators*) can be represented by means of complex visualizations, such as bar charts or pie charts properly positioned in the areas of a map. Moreover, it is possible to view the distribution of specific phenomena exploting the visual power of choropleth maps or heat maps.

– *Temporal slider*. It is possible to animate a map, by showing the evolution of a phenomenon over time and seeing the trend based on the variation of a layer. For example, a dashboard can monitor crimes over time in a specific area (e.g., to check the effects of a preventive action), or the amount of investments made for each branch can be analysed over time.

In Table 1 we show which of these functionalities can be supported by each architecture for location intelligence. Of course, not all the commercial tools that adopt a given architecture actually support all the indicated functionalities; for instance, Oracle with Spatial option adopts a tightly-coupled architecture but does not support real-time refresh and multi-layer representation.

Table 1. Functional support on different architectures for location intelligence

	Spatial drill	SOLAP queries	Geo-coding	Spatial triggering	Spatial clustering	Integr. with ext. maps	Real-time refresh	Multi-layer repr.	Spatial KPI visual.	Temporal slider
Loosely-coupled	✓					✓		✓*	✓	
OLAP-dom. semi-tightly coupled						✓		✓*	✓	
GIS-dom. semi-tightly coupled	✓		✓			✓	✓	✓*	✓	
Tightly-coupled	✓	✓	✓			✓	✓	✓	✓	✓

* Only at a single aggregation level.

5 Lily: An Architectural View

Though Lily is inserted into a tightly-coupled architecture to achieve great performances and advanced functionalities (e.g., active spatial queries and spatial drill), it follows a mash-up approach, whose main advantages are a fast deployment and the possibility of reusing existing services by transparently integrating them within a single user interface. In particular, Figure 1 shows how, within the current Oracle architecture for location intelligence, the Lily component can be interposed between Oracle Business Intelligence (OBI, it includes a presentation front-end and a multidimensional engine called Oracle BI Server) and MapViewer (the Oracle component in charge of map rendering and of supporting the connection between spatial and business data). Lily is currently implemented to be specifically connected with MapViewer; however, any other presentation service (e.g., Business Objects or Microstrategy) can be used instead of OBI provided that it supports web-based reporting.

Figure 2 shows a functional view of the process though which queries are answered in a Lily-based architecture. Assuming that in all the reports shown to end-users we can separate *non-spatial items* (e.g., cross-tables and diagrams) from *spatial items* (e.g., a heat map), two relevant scenarios can be distinguished:

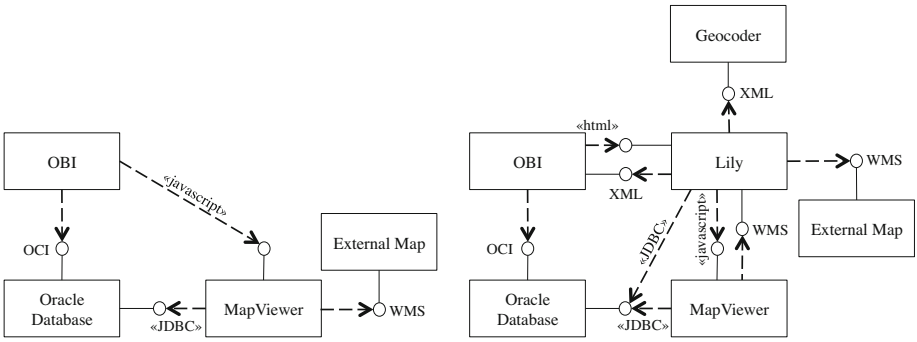


Fig. 1. Oracle-based architecture for location intelligence with —right— and without —left— Lily (OCI: Oracle Call Interface; WMS: Web Map Service)

- *OLAP queries.* These are queries that end-users formulate by interacting with a non-spatial item, so that spatial items are only involved in visualizing the results. An OLAP query q is sent to Oracle BI Server, that translates it into SQL. The resulting SQL query is sent to Oracle Database that returns the corresponding data to Oracle BI Server, that in turn sends these data to the front-end to feed non-spatial items and also routes them to MapViewer. At the same time, the rendering statements associated with q are sent to Lily that translates them into a SQL spatial query q_{spa}. Oracle Spatial & Graph answers q_{spa} and returns the corresponding spatial data to MapViewer that, based on the intensional mappings provided by Lily, integrates them with the non-spatial data received by Oracle Database to build an integrated rendering for spatial items.
- *SOLAP queries.* These queries are formulated by interacting with a spatial item. Lily translates SOLAP queries into SQL and sends them to MapViewer,

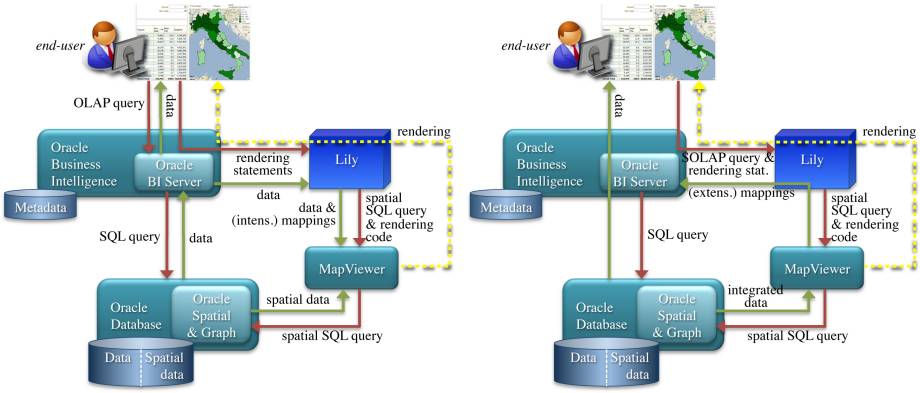


Fig. 2. Functional view of how OLAP (left) and SOLAP (right) queries are answered using Lily

that in turn routes them to Oracle Spatial & Graph for answering. The integrated (spatial and non-spatial) data returned are sent to MapViewer that builds the integrated rendering for spatial items and, based on the instructions provided by Lily, creates the extensional mappings necessary for synchronizing the non-spatial with the spatial component of the results. Oracle BI Server uses these mappings to send a non-spatial query to Oracle Database, that returns the data for updating the non-spatial items in the end-user report.

6 An Example

In this section we show an example in which Lily is coupled with the Oracle Business Intelligence Foundation Suite to achieve location intelligence functionalities for a major Italian telecommunication company.

The main functions provided can be summarized as follows:

1. *Geo-dashboard*: Area-based comparison of the main KPIs (number of activations, profitability, etc.) with guided navigation of detailed area-based reports.
2. *Volumes*: Area-based analysis of selling volumes by customer type, business channel, manager, etc.
3. *Sustainability*: Area-based analysis of economical sustainability by business format, business channel, manager, etc., with visualization of costs and revenues, influence areas of points-of-sales, etc.
4. *Points-of-sales Analysis*: Proximity-based analysis of a point-of-sales with reference to a benchmark set and analysis of correlations between points-of-sales; geographical representation of competitive pressure in terms of distance from competitor points-of-sales and identification of uncovered areas.
5. *Analysis of Territory Potential*: Identification of areas with high selling-potential (in terms of population, wealth KPIs, competitor presence, etc.) aimed at identifying the best positions for new points-of-sales; what-if simulation of the opening of a new point-of-sales with real-time computation and representation of its catchment area.

As shown in Figure 3, the geo-dashboard provides a summarized view of the main KPIs in table/chart format as well as on a map using the integration with external maps, multi-layer representation, geo-coding, and spatial KPI visualization functionalities of Lily. In particular, three themes are graphically overlapped: a bucket theme showing the boundaries of areas, a point theme showing all selected points-of-sales, and a pie chart theme indicating the customer base KPI on each area. The reference functional scenario is the one shown in Figure 2, left. To give a flavor of how query and data flow are processed by the different components, in Figure 4 we report examples for the main flows involved in the drawing of the pie charts; the rendering code and the intensional mappings are not shown because they come in form of complex Javascript code generated by Lily starting from rendering statements.

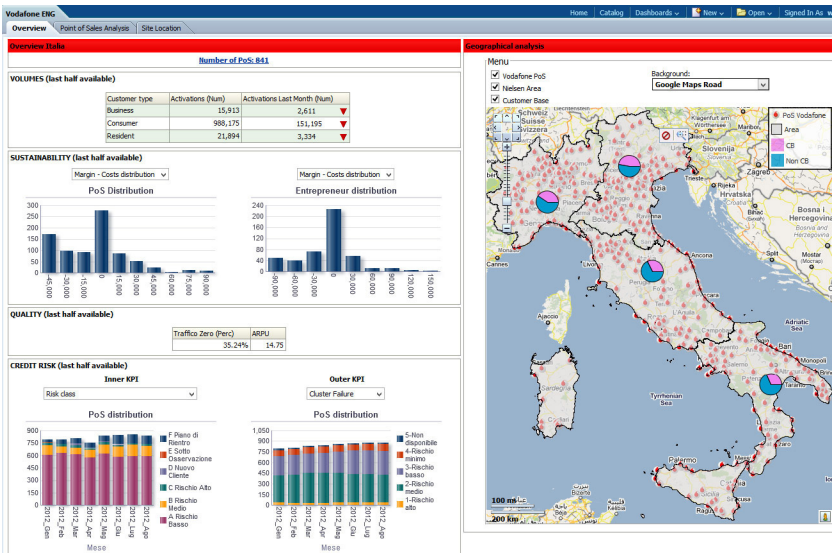


Fig. 3. Geo-dashboard

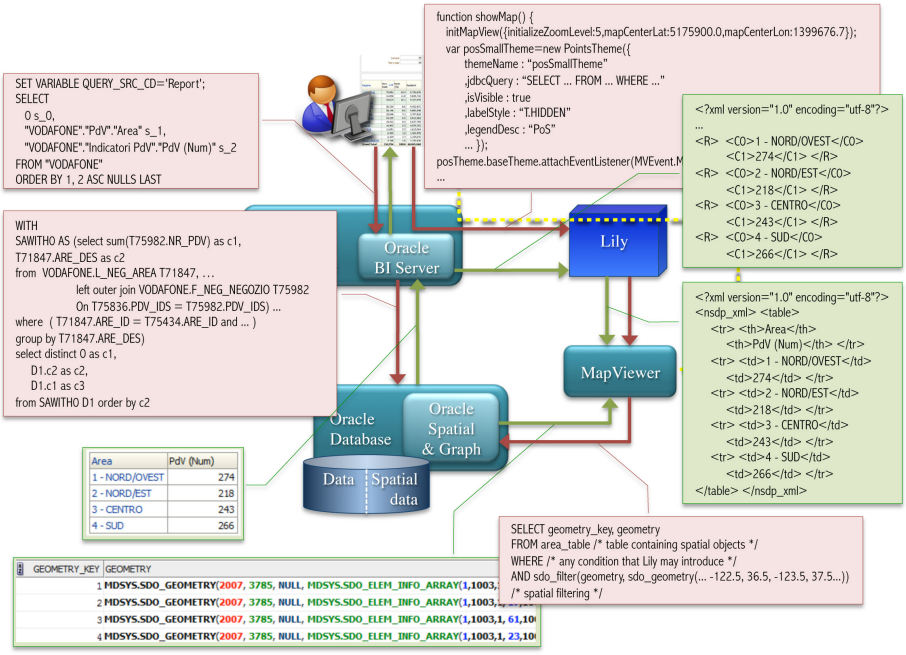


Fig. 4. Main query and data flows involved in the drawing of pie charts

Figure 5 shows an example of an analysis of territory potential. Besides all the Lily functionalities used for the geo-dashboard, also SOLAP queries are involved in this case for proximity analysis.

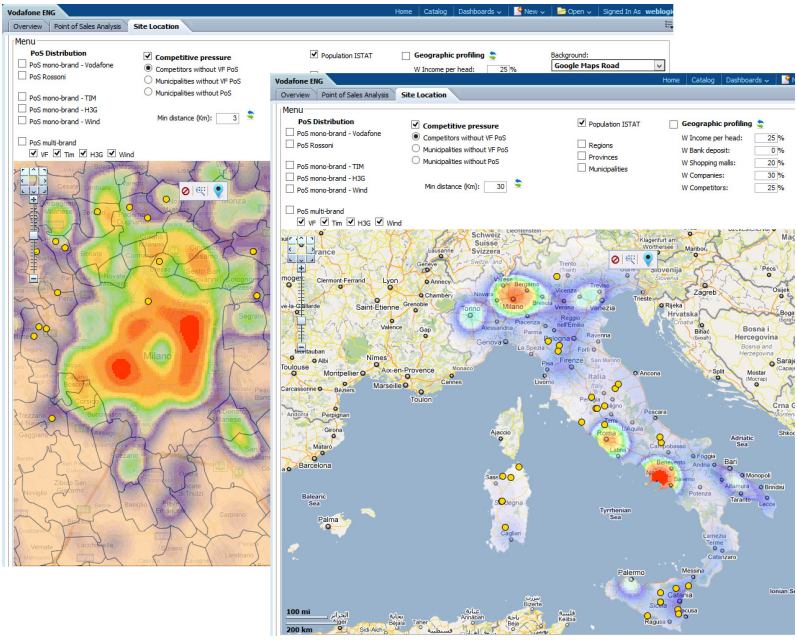


Fig. 5. Analysis of territory potential at national and local scale

7 Final Remarks

Integration of true location intelligence functionalities within BI platforms is still an ongoing process. Lily, the library we proposed in this paper, represents a step forward in this direction since it offers a set of geo-enhanced functionalities in the context of a tightly-coupled architecture, so as to ensure top performances coupled with transparent usage and programming. Although Lily has been designed to work with a generic BI platform, in its current implementation it only works with OBI. We are now working to remove this limitation by implementing the interfaces with other platforms that match the tightly-coupled paradigm. Achieving this goal is a bi-directional process that also requires the standardization of the Lily interfaces and the extension of the native functionalities provided by the BI platforms.

References

1. Bédard, Y., Larrivée, S., Proulx, M., Létourneau, F.: Étude de l'etat actuel et des besoins de R&D relativement aux architectures et technologies des data warehouses appliquées aux données spatiales. Tech. rep., Laval University (1997)
2. Bédard, Y., Han, J.: Fundamentals of spatial data warehousing for geographic knowledge discovery. In: Miller, H.J., Han, J. (eds.) Geographic Data Mining and Knowledge Discovery, pp. 45–66. Chapman & Hall (2009)
3. Bimonte, S., Bertolotto, M., Gensel, J., Boussaid, O.: Spatial OLAP and map generalization: Model and algebra. IJDWM 8(1), 24–51 (2012)
4. Bimonte, S., Miquel, M.: When spatial analysis meets OLAP: Multidimensional model and operators. IJDWM 6(4), 33–60 (2010)
5. Bitterer, A.: Location intelligence is expanding the scope of BI. Tech. Rep. G00239089, Gartner Research (2012)
6. BusinessWeek Research Services, Pitney Bowes Business Insight: Location intelligence: The new geography of business. White paper (2006)
7. Gómez, L.I., Haesevoets, S., Kuijpers, B., Vaisman, A.A.: Spatial aggregation: Data model and implementation. Inf. Syst. 34(6), 551–576 (2009)
8. Ihm, J., Lopez, X., Ravada, S.: Advanced spatial data management for enterprise applications. Oracle White Paper (2010)
9. Katibah, E., Stojic, M.: New spatial features in SQL Server Code. Microsoft White Paper (2011)
10. Malinowski, E., Zimányi, E.: Requirements specification and conceptual modeling for spatial data warehouses. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM Workshops 2006. LNCS, vol. 4278, pp. 1616–1625. Springer, Heidelberg (2006)
11. Papadias, D., Kalnis, P., Zhang, J., Tao, Y.: Efficient OLAP operations in spatial data warehouses. In: Proc. SSTD, Redondo Beach, CA, pp. 443–459 (2001)
12. Rao, F., Zhang, L., Yu, X., Li, Y., Chen, Y.: Spatial hierarchy and OLAP-favored search in spatial data warehouse. In: Proc. DOLAP, New Orleans, LA, pp. 48–55 (2003)
13. Rivest, S., Bédard, Y., Proulx, M.J., Nadeau, M.: SOLAP: a new type of user interface to support spatio-temporal multidimensional data exploration and analysis. In: Proc. ISPRS Workshop, Québec, Canada (2003)
14. Rivest, S., Bédard, Y., Proulx, M.J., Nadeau, M., Hubert, F., Pastor, J.: SOLAP technology: Merging business intelligence with geospatial technology for interactive spatio-temporal exploration and analysis of data. ISPRS Journal of Photogrammetry & Remote Sensing 60, 17–33 (2005)
15. Scotch, M., Parmanto, B.: SOVAT: Spatial OLAP visualization and analysis tool. In: Proc. HICSS, Big Island, HI (2005)
16. da Silva, J., de Oliveira, A.G., do Nascimento Fidalgo, R., Salgado, A.C., Times, V.C.: Modelling and querying geographical data warehouses. Inf. Syst. 35(5), 592–614 (2010)
17. Stefanovic, N., Han, J., Koperski, K.: Object-based selective materialization for efficient implementation of spatial data cubes. IEEE Trans. Knowl. Data Eng. 12(6), 938–958 (2000)
18. Whitehorn, M., Zare, R., Pasumansky, M.: Fast Track to MDX. Springer, Berlin (2005)

Discovering Co-location Patterns in Datasets with Extended Spatial Objects

Aibek Adilmagambetov, Osmar R. Zaiane, and Alvaro Osornio-Vargas

Department of Computing Science and Department of Paediatrics
University of Alberta, Edmonton, Canada
{adilmaga,zaiane,osornio}@ualberta.ca

Abstract. Co-location mining is one of the tasks of spatial data mining, which focuses on the detection of the sets of spatial features frequently located in close proximity of each other. Previous work is based on transaction-free apriori-like algorithms. The approach we propose is based on a grid transactionization of geographic space and designed to mine datasets with extended spatial objects. A statistical test is used instead of global thresholds to detect significant co-location patterns.

1 Introduction

Co-location mining aims to discover patterns of spatial features often located close to each other in geographic proximity. An example is a co-location of symbiotic species of plants and animals depending on ecological conditions. The main purpose of co-location mining is to come up with a set of hypotheses based on data features and statistics that can be useful for domain experts to reduce the range of possible patterns that are hidden and need to be checked. Even though this task seems to be similar to association rule mining (ARM), the adaptation of ARM techniques is not trivial due to the fact that features are embedded into a geographic space and there is no clear notion of transactions.

Most of the existing approaches to the co-location mining problem [1–4] deploy a framework which requires a user-defined minimum prevalence threshold. Without prior knowledge, it could be difficult to choose a proper threshold. Furthermore, spatial features often have various frequencies in datasets, and one global threshold might lead to omission of some co-location patterns and rules with rare events or detection of meaningless patterns. Another limitation of most algorithms is that they work with point spatial features and one neighborhood distance threshold, whereas in reality there are datasets which in addition to point instances also have lines and polygons, e.g., a road network map.

We propose a new framework which combines co-location mining, frequent pattern and association rule mining. A statistical test is used to determine the significance of co-location patterns and rules. A co-location is considered as significant if it has a surprisingly high level of prevalence in comparison with randomized datasets which are built under the null hypothesis that the features are independent from each other. We improve computation with filtering techniques.

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 84–96, 2013.

The motivating application of this paper is the detection of possible spatial associations of chemicals and cases of childhood cancer. Although some people are genetically predisposed to cancer, most of the cases of cancer are caused by environmental factors, such as air pollutants, radiation, infections, tobacco, and alcohol. However, the causes of childhood cancer are difficult to determine partially because of the fact that children's cancer cases are rare and the levels of exposure to environmental factors are hard to evaluate. A collaborative research effort with the Faculty of Medicine is trying to identify associations between cancer cases and known emissions by industry. Some chemicals are proven to be carcinogens while others are not known to cause cancer in isolation. It is unknown if certain combinations of chemicals can be associated with higher rates of cancer. Moreover, even if potentially problematic combinations are not emitted by the same industry, atmospheric conditions can contribute to the mixture. We deploy our model on the dataset containing information on chemical emission points and amounts of release in Alberta, Canada, and childhood cancer cases with their location when they were first diagnosed. Our data is obtained from the National Pollutant Release Inventory (NPRI), Canada's legislated, publicly accessible inventory of pollutant releases, as well as the health authorities in Alberta for 1254 cancer cases of children younger than 19 between 2002 and 2007. NPRI for the province of Alberta provided 1465 points releasing a variety of chemicals among 47 of interest, some carcinogenic and some not classifiable as to carcinogenicity. In this paper we explain a modeling framework which is used to handle the data as accurately as possible. While we are not intending to find causalities, the goal of the study is to identify potential interesting spatial associations in order to state hypotheses and investigate further the relationship between cancer and specific combinations of chemicals.

The remainder of the paper is organized as follows. The overview of the related work is given in Section 2. The proposed framework and its outline are described in Section 3. Section 4 describes the challenges and modeling framework used to mine the pollutants and childhood cancer cases. The experiments are presented in Section 5, followed by conclusions.

2 Related Work

2.1 Co-location Mining

Co-location mining algorithms can be divided into two classes of methods: spatial statistics approaches and spatial data mining approaches.

Spatial Statistics Approaches. use statistical techniques such as cross K-functions with Monte-Carlo simulations [5], mean nearest-neighbor distance, and spatial regression models [6]. The disadvantages of these approaches are the expensive computation time and the difficulty of application to patterns with more than two spatial features.

Spatial Data Mining Approaches. could be categorized into several types. Transaction-based approaches work by creating transactions over space and

using association rules [7–9]. One of the ways, a reference-centric model, creates transactions around a reference feature. However, this approach may consider the same instance set several times. Another approach, a window-centric model, divides the space into cells and considers instances in each cell as a transaction which causes a problem of some instance sets being divided by cell boundaries.

Spatial join-based approaches work with spatial data directly. They include cluster-and-overlay methods and instance-join methods. In the cluster-and-overlay approach a map layer is constructed for each spatial feature based on instance clusters or boundaries of clusters [10]. The authors propose two algorithms for cluster association rule mining, vertical-view and horizontal-view approaches. In the former, clusters for layers are formed and layers are segmented into a finite number of cells. Then, a relational table is constructed where the element is equal to one if the corresponding cell satisfies the event in a layer, and zero otherwise. The association rule mining is applied to the table. The second approach uses intersections of clustered layers. A clustered spatial association rule is of the form $X \rightarrow Y(CS\%, CC\%)$, where X and Y are the sets of layers, $CS\%$ is the clustered support and $CC\%$ is the clustered confidence. However, these approaches might be sensitive to the choice of clustering methods, and assume that features are explicitly clustered.

Another type of spatial join-based methods - instance-join algorithms - is similar to classical association rule mining. Shekhar and Huang [1] proposed a co-location pattern mining framework which is based on neighborhood relations and the participation index concept. The basic concepts of the co-location mining framework are analogous to concepts of association rule mining. As an input, the framework takes a set of spatial features and a set of instances, where each instance is a vector that contains information on the instance id, the feature type of the instance, and the location of the instance. As an output the method returns a set of co-location rules of the form $C_1 \rightarrow C_2(PI, cp)$, where C_1 and C_2 are co-location patterns, PI is the prevalence measure (the participation index), and cp is the conditional probability. A co-location pattern is considered prevalent, or interesting, if for each feature of the pattern at least $PI\%$ instances of that feature form a clique with the instances of all other features of the pattern according to the neighborhood relationship. Similarly to association rule mining, only frequent $(k-1)$-patterns are used for the k-candidate generation.

The approaches mentioned above use thresholds for measures of interestingness, which causes meaningless patterns to be considered as significant with a low threshold, and a high threshold may prune interesting rare patterns.

3 Algorithm

Various approaches to the co-location mining problem have been proposed during the past decade. However, most of them focused on improving the performance of existing frameworks which have several disadvantages. Several studies addressed these issues but only separately, and these issues remain major hurdles for some real-world applications such as our motivating problem of finding co-locations of cancer cases and pollutant emission points.

First, the usage of thresholds for the detection of interesting co-location patterns and rules is the main limitation factor of many co-location mining algorithms. In spatial datasets the features usually have a varying number of instances; they could be extremely rare or be present in abundance. Therefore, one threshold for participation index (or any other significance measure) cannot capture all meaningful patterns, while other patterns could be reported as significant even if their relation is caused by autocorrelation or other factors. In addition most current algorithms use a candidate generation process which forms $(k+1)$-size candidates only from significant k-size patterns. However, a set of features could be interesting even if some of its subsets are not significant (for example, two chemicals may not be correlated with disease separately, but cause it when they are combined). In this work we use the statistical test which replaces one global threshold. It is proposed for co-location mining by Barua and Sander [11]. The pattern is considered significant, if the probability of seeing the same or greater value of the prevalence measure in N artificial datasets is less than α (the significance level) under the null hypothesis that there is no spatial dependency among features of the pattern. Each candidate pattern is evaluated separately rather than applying one threshold for all of them.

Second, most co-location mining approaches are designed for spatial datasets with point features. However, other types of objects may exist in spatial data such as lines (roads) and polygons (polluted regions). Even though the framework for extended objects [4] deals with lines and polygons, it also uses one threshold for the prevalence measure. If the statistical test is applied to this model, computationally expensive GIS overlay methods should be used for each candidate pattern in order to calculate its prevalence measure in a real and randomized datasets. When the number of patterns and simulation runs in the statistical test are large, this method could become prohibitively expensive.

We propose a new framework that addresses the aforementioned limitations. It uses grid-based "transactionization" (creating transactions from a dataset). The statistical test is performed on the derived set of transactions to get significant co-location rules or patterns.

3.1 Algorithm Design

The objective is to detect significant patterns in a given spatial dataset that have the prevalence measure value higher than the expected one. The spatial dataset may contain points, lines or polygons. A buffer is built around each spatial object, and it defines the area affected by that object; for example, the buffer zone around an emission point shows the area polluted by a released chemical. The buffer size might be one for all objects or it might be different for each of the spatial instances depending on various factors which may vary for different applications. In addition, the likelihood of the presence of the corresponding feature in the region covered by the object and its buffer is not uniform and may depend on factors such as the distance from the object.

We propose a new transaction-based approach that is suitable for extended spatial objects. Previous transaction-based methods have some limitations. A

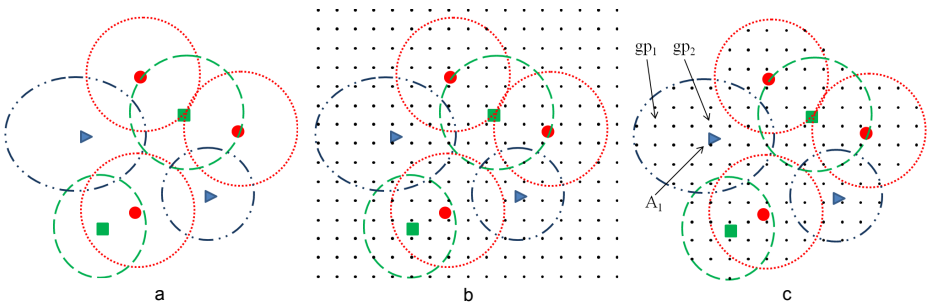


Fig. 1. Transactionization step: (a) an example spatial dataset with point feature instances and their buffers; (b) a grid imposed over the space; (c) grid points which intersect with buffers are used to create transactions

window-centric model cuts off neighborhood relations of instances located close to each other but in different partitions. A reference-centric model may get duplicate counts of spatial instances. In addition, it is nontrivial to generalize this approach to applications with no reference feature. Instead of these models we propose a new transactionization method. In order to transform spatial data into transactions, we use a grid which points are imposed over the given map. Fig. 1 (a) displays an example dataset with buffers around spatial point instances, and a grid is laid over it (Fig. 1 (b)). Similarly, buffers can also be created around linear and polygonal spatial objects. In a two-dimensional space, the grid points represent a square regular grid.

Each point of the grid can be seen as a representation of the respective part of the space. A grid point may intersect with one or several spatial objects and their buffers. A transaction is defined as a set of features corresponding to these objects. A probability of a feature being in a transaction is also stored and it may depend on the distance from the spatial object. For example, the grid point gp_2 in Fig. 1 (c) is located closer to the point A_1 than the point gp_1; therefore, $p(A, gp_2) > p(A, gp_1)$. The granularity of the grid should be carefully chosen for each application and it could depend on the average size of the region covered by a spatial object and its buffer. Choosing too great a distance between grid points may negatively affect the accuracy of the results because small feature regions and their overlaps might get a different number of intersecting grid points depending on the grid imposition. The short distance between grid points leads to a great number of derived transactions, and the following computation of pattern significance levels might become prohibitively expensive.

Given a set of transactions T, derived after the transactionization of the spatial dataset, and a set of spatial features F, the prevalence measure value is calculated for all candidate co-location patterns or rules. In some applications experts look for sets of features that are co-located with each other. The expected support $ExpSup(P)$ might be used to define the level of the interestingness of a pattern P. For other applications, researchers intend to analyze a predefined set of rules. For example, for a dataset of disease outbreaks and possible cause

factors a typical co-location rule is of the form $C \rightarrow D$, where C is a subset of cause features and D is a disease feature. For these projects, the expected confidence $ExpConf(X \rightarrow Y)$ can be used as a prevalence measure of a co-location rule $(X \rightarrow Y)$, where $X \subseteq F$, $Y \subseteq F$, and $X \cap Y = \emptyset$. Algorithm 1 shows the outline of our model in the case when co-location patterns are mined.

Definition 1. *The probability $p(P,t)$ of the pattern P occurring in a transaction t is the product of the corresponding feature instance probabilities, $p(P,t) = \prod_{f \in P} p(f,t)$.*

Definition 2. *The expected support $ExpSup(P)$ of a pattern P is defined as the sum of expected probabilities of presence of P in each of the transactions t in the database, $ExpSup(P) = \sum_{t \in T} p(P,t)$.*

Definition 3. *The expected confidence $ExpConf(X \rightarrow Y)$ of a rule $X \rightarrow Y$ is defined as $ExpConf(X \rightarrow Y) = ExpSup(X \cup Y)/ExpSup(X)$.*

The next step, the statistical test, helps to estimate the likelihood of seeing the same level of the prevalence measure or greater under a null hypothesis that features of a pattern or rule are spatially independent from each other.

Definition 4. *A pattern P is said to be significant at level α, if the probability p of seeing the observed expected support $ExpSup_{obs}$ or larger in a dataset, complying with a null hypothesis, is not greater than α. (The same for $ExpConf_{obs}$)*

Let us suppose that the expected confidence $ExpConf$ is used as a prevalence measure. Let $ExpConf_{obs}(X \rightarrow Y)$ denote the expected confidence of a co-location rule $X \rightarrow Y$ in a real dataset, and $ExpConf_{rand}(X \rightarrow Y)$ - the expected confidence of $X \rightarrow Y$ in a randomized dataset which is generated under the null hypothesis. In order to estimate the probability p, the expected confidence of the co-location rule in R randomized datasets is calculated. Having the number of simulations R, the value of p is computed as:

$$p = \frac{R_{\geq ExpConf_{obs}} + 1}{R + 1}, \tag{1}$$

where $R_{\geq ExpConf_{obs}}$ is the number of simulations in which $ExpConf_{rand}(X \rightarrow Y) \geq ExpConf_{obs}(X \rightarrow Y)$. The observed dataset is added to both numerator and denominator.

If the p-value is less or equal to the predefined level of significance α, the null hypothesis is rejected. Therefore, the co-location rule $X \rightarrow Y$ is significant at level α.

3.2 Candidate Filtering Techniques

The calculation of the p-value is repeated for all candidate co-location patterns or rules. The number of candidates grows exponentially with the number of spatial

features in the dataset. In addition, the accuracy of the p-value depends on the number of simulation runs; therefore, the more randomized datasets are checked, the more accurate are the results. These two factors may lead to an enormous amount of computation. However, the support of a co-location decreases as the size of a candidate pattern or rule increases, because less transactions contain all its features. Therefore, one might put a threshold on the support or the maximal size of a candidate in order to analyze only patterns and rules that are backed by a meaningful number of transactions. In addition, we use the following filtering techniques to exclude candidate patterns and rules that are de facto insignificant.

- First, after the calculation of the prevalence measure for candidate patterns in a real dataset, a subset of patterns may have a prevalence measure value equal to zero. Obviously, these patterns cannot be statistically significant and they can be excluded from the set of candidate patterns (lines 6-7 in Algorithm 1).
- Second, during the calculation of the p-value for the candidate patterns for which the observed prevalence is higher than zero, some of the candidate patterns might show a p-value that exceeds the level α. For example, let us assume that the number of simulation runs is 99 and $\alpha = 0.05$. If after ten simulation runs the prevalence measure of a pattern P is greater than the observed prevalence in 5 randomized datasets, pattern P already surpassed the threshold ($(5+1)/(99+1) > 0.05$) and, therefore, can be excluded from the following 89 checks (lines 15-17 in Algorithm 1). With this filter, after the last simulation run the set of candidates contains only significant patterns.

4 Modeling Framework

The modeling framework that is used to handle and analyze the data is an important part of practical research. In theoretical studies it could be simplified in order to generalize the task and define algorithms that could be applied for a wide range of applications. However, the usage of general approaches and algorithms may result in misleading or even wrong results. For example, the neighborhood distance threshold is an important measure of interaction and relationship between features. Obviously, one distance threshold cannot capture accurately all links among features. In biology, various animal species have different home ranges, areas where they search for food; rodents may require little space, while birds forage on wider regions. Another example is derived from urban studies. Two points of interest, e.g., a shopping mall and a grocery store, could be situated on a distance exceeding a threshold, but if they are connected by a high quality road, they are more likely to be co-located than other two points positioned seemingly close to each other but separated by some obstacle. Most domains of research, if not all, have their own nuances that must be taken into account by researchers in order to get most accurate and significant results.

The motivating task of this paper, detecting co-locations of pollutants and cancer cases, has unique difficulties and challenges. The distribution of a pollutant is not uniform and it could depend on several factors: types of pollutants,

Algorithm 1. Mining significant co-location patterns

Input: Spatial dataset S; Level of significance α; Number of simulation runs R.
Output: Set of significant co-location patterns P

Impose a grid over the real dataset
$T \leftarrow$ set of derived transactions
$CP \leftarrow$ set of candidate patterns
for each $cp \in CP$ **do**
  $cp.ExpSup_{obs} \leftarrow ComputeExpSup(T)$
  if $cp.ExpSup_{obs} = 0$ **then**
    $CP \leftarrow CP/cp$
  end if
end for
for $i = 1 \rightarrow R$ **do**
  Impose a grid over the i-th randomized dataset
  $T \leftarrow$ set of derived transactions
  for each $cp \in CP$ **do**
    $cp.ExpSup_{sim}[i] \leftarrow ComputeExpSup(T)$
    if $cp.ExpSup_{sim}[i] \geq cp.ExpSup_{obs}$ **then**
      $cp.R_{\geq ExpSup_{obs}} \leftarrow cp.R_{\geq ExpSup_{obs}} + 1$
      $cp.\alpha \leftarrow \frac{cp.R_{\geq ExpSup_{obs}}+1}{R+1}$
      if $cp.\alpha > \alpha$ **then**
        $CP \leftarrow CP/cp$
      end if
    end if
  end for
end for
$P \leftarrow CP$
return P

amounts of release, climatic conditions (wind, precipitation), topography, etc. Various chemicals have different levels of harmfulness. In addition, the pollutant concentration is directly proportional to the distance from an emitting point. These are only several examples. We show how we tackled some of these problems such as pollutant amounts, wind speed and direction, and the concentration of chemicals. Certainly, we do not aim to reproduce complicated air pollution distribution models. Instead, our model gives a simple framework that increases the accuracy of results while operating with available data.

4.1 Pollutant Amounts

The dataset on pollutants contains the data on yearly releases of chemicals. For our research we take an average amount of release for a year on given facilities and chemicals, which is further normalized by Toxic Equivalency Potentials (TEPs) when they are available. TEP shows the relative risk associated with one kilogram of a chemical in comparison with the risk caused by one kilogram of benzene. Chemicals with high TEPs are extremely toxic. The range of the

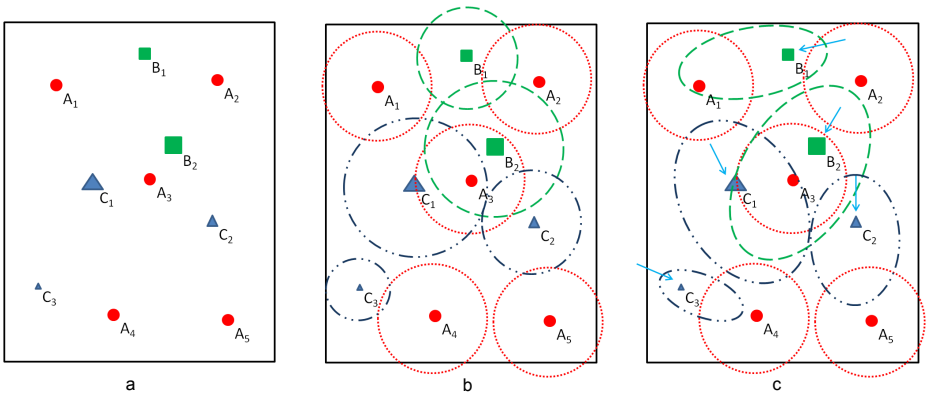


Fig. 2. Modeling framework usage examples: (a) an example spatial dataset (A - cancer, B and C - pollutants); (b) buffer sizes vary depending on the pollutant release amount; (c) buffer shapes change with the wind direction and speed (shown by arrows)

average amount values varies from several kilograms to tens of thousands tons; the maximum average yearly release in the dataset is 80,000 tons. Certainly, one distance threshold for all pollutant emissions is inaccurate, because the more a chemical is released, the farther it distributes from a source point. Fig. 2 (a) displays an example dataset containing cancer points (feature A) and chemical points (features B and C). On Fig. 2 (b) buffer zones around pollutant points are based on the amount released. For example, instance C_1 has a larger zone affected by this source point than instance C_3 which has smaller amount of emission. Buffer zones of cancer points denote average active living zones.

For simplicity, we decided to take the maximal distance as the natural logarithm function of the release amount. This function gives a smooth curve which does not grow as fast as linear or root functions that give large numbers for heavier releases. Even though this technique oversimplifies the real world conditions of pollutant dispersion, it helps to make the results more precise. Other functions can be used to calculate the maximal distribution distance and they can depend on a type of a pollutant (a heavier chemical settles faster and on a shorter distance from a chimney) or a height of a chimney. An additional point that could be considered in future work is that the area very close to a chimney does not get polluted, and the higher is the chimney, the bigger is that area.

4.2 Wind Speed and Direction

The climatic conditions and topographical features may affect the distribution of chemicals in the air. The examples of these factors are prevailing winds, precipitation, relative humidity, mountains, hills, etc. At the first step in this part of the modeling framework we include the wind speed and the prevailing wind direction on source points as variables of the model.

Regarding the wind speed and direction, two situations are possible. First, the region where a facility is located is windless throughout the year. In this case, the pollutant is assumed to disperse in a circular region around the source point with the radius of the circle derived from the released amount as discussed in the previous subsection. However, the second situation is more frequent - there is nonzero wind speed with a prevailing wind direction. In this case we presume that the original distribution circle is morphed into a more ellipse-like region. Our calculations of the characteristics of the ellipse are based on the works by Getis and Jackson [12], and Reggente and Lilienthal [13]. The major axis of the ellipse is in the direction of the prevailing wind. Furthermore, the coverage area of the ellipse is supposed to remain constant. The source point can be placed on the major axis of the ellipse between the center and upwind focus; in our model we locate it in the middle of the segment between these two points. Fig. 2 (c) illustrates elliptical buffer regions; their forms are dependent on the wind speed and its frequent direction.

The lengths of the major semi-axis a and minor semi-axis b are derived from the equations: $a = r + \gamma|\boldsymbol{v}|$, $b = \frac{r^2}{a}$, where r - the radius of the original circle, $\boldsymbol{v}$ - the wind speed, and γ - the stretching coefficient.

The larger the value of the stretching coefficient, the longer is the length of the ellipse's major axis. We fixed γ at 0.3, but it could have a different value for each of the pollutants. The calculation of b follows our assumption that the area of the ellipse is equal to the area of the original circle.

In order to get the values of the wind speed and prevailing wind direction, the interpolation of wind fields between weather stations is used. The data of monitoring stations comes from two sources. First, the data on 18 stations is obtained from Environment Canada, which provide climate normals that are based on climate stations with at least 15 years of data between 1971 and 2000. The most frequent wind direction is the direction (out of possible eight directions) with the highest average occurrence count. Second, the data on 156 stations is derived from Agro-Climatic Information Service (ACIS), a provincial government service. The data is daily, between 2005 and 2011. In order to make the data consistent, the average wind speed and the most frequent wind direction are calculated using the same methods as for the federal government website data. The climate normals from two sources are combined and used to make interpolations in ArcGIS tool [14]. However, ArcGIS is restricted to linear surface interpolations and the wind direction is a nonlinear attribute. In linear systems (e.g., the number of sunny days) the number goes only in one direction. On the other hand, nonlinear systems may have several paths. For example, there are clockwise and counter-clockwise directions to move from 90° to 270°: through 0° or 180°.

Interpolation of wind fields requires a technique that considers nonlinear nature of the wind direction attribute. The transformation is done according to the work by Williams [15]. The wind speed and wind direction from each monitoring station is represented as a vector with the magnitude S (wind speed) and direction θ (wind direction). The vector is divided into axial components $X = S \sin\theta$ (northern wind) and $Y = S \cos\theta$ (eastern wind).

Based on these two components, two ArcGIS surface interpolations are created. The type of interpolation used is spline. As a result we get two grids: for northern X' and eastern wind Y'. The magnitude of the vector, the wind speed S', is computed as:

$$S' = \sqrt{X'^2 + Y'^2}. \tag{2}$$

The calculation of wind direction angle θ' is more complicated. From geometry, the wind direction is calculated as $\theta' = \tan^{-1}(Y'/X')$. However, due to the fact that the inverse tangent is defined only for values between -90° and 90°, each quadrant (the section of the graph which depends on the signs of wind vector components; for example, Quadrant I is bounded by positive X' and Y', Quadrant II - by positive X' and negative Y') requires its own formula [15]. As a result we get interpolated values of wind speed and wind direction for each point of the studied space.

5 Experimental Evaluation

We conducted experiments on a real dataset, containing data on pollutant emissions and childhood cancer cases. The information on pollutants is for the 2002-2007 period and contains the type of chemical, location of release, and average amount of release per year. In order to get reliable results the chemicals that had been emitted from less than three facilities are excluded from the dataset. There are 47 different chemicals and 1,465 pollutant emission points; several chemicals might be released from the same location. The number of cancer points (addresses where a child lived when cancer was diagnosed) is 1,254. Claiming discovering causality is wrong and controversial and thus we attempt only to find "associations" rather than "causalities". The results are still subject to careful evaluation by domain experts in our multidisciplinary team and the publication of the found associations is not authorised at this point. It suffices to mention, however, that some surprising rules were discovered indicating significant association between groups of chemicals, that were not categorized individually as carcinogens, and childhood cancer, as well as rules with pairs of chemicals such that one was known as carcinogenic but did not associate with cancer in our data except in the presence of another that acted as a catalyzer.

We are interested in co-location rules of the form $Pol \rightarrow Cancer$, where Pol is a set of pollutant features and $Cancer$ is a cancer feature. The expected confidence is used as a prevalence measure. The distance between points in a grid is 1 km; its effect is also evaluated. The number of simulations for the statistical test is set to 99, so that with the observed data the denominator in Equation (1) is 100. The level of significance α is set to 0.05. The size of an antecedent of candidate rules is up to three. Larger candidates have low support values due to the fact that the average number of features in a transaction in the experiment is 1.95.

The randomized datasets that are used in the statistical test are generated as follows. Pollutant emitting facilities are not random and usually located close to regions with high population density, while they are not present in other

places (e.g., in protected areas). Due to this observation, we do not randomize pollutant points all over the region, but instead keep locations of facilities and randomize pollutants within these positions. Out of 1,254 cancer points, 1,134 are located within dense "urban" municipalities (cities, towns, villages, etc.) and the rest are diagnosed in "rural" areas. In order to have the randomized cancer occurrence rate close to the real-world rate, we keep the number of cancer feature instances positioned in "urban" ("rural") regions the same as in the real dataset. The number of random cancer cases placed within each "urban" municipality is directly proportional to the number of children counted in the 2006 census.

Effect of Filtering Techniques. The number of candidate co-location rules in the experiment is 17,343 (co-locations with the antecedent size up to three). With a naive approach all candidates would be checked in each simulation run. With our filter excluding rules with zero-level confidence, 10,125 candidates remain. The usage of the second filtering method (the exclusion of candidates which p-values passed α) considerably reduces the amount of computation. While in the first simulation run the confidence value is computed for 10,125 rules, in the last run only 482 candidates are evaluated.

Effect of the Grid Granularity. As already mentioned, the granularity of the grid (the distance between grid points which affects the number of points per unit of space) is crucial for the accuracy of the results. Having too long distance between grid points may lead to omission of some regions of the space especially when the average buffer distance is short. On the other hand, too short distance between points leads to the greater number of transactions. Decreasing the distance by two increases the transaction set size approximately by four. Therefore, more computation needs to be done. The grid resolution might be set up depending on the average buffer size.

In addition to the grid with 1 km granularity, we conducted two experiments with 2 and 0.5 km grids. As mentioned above, the algorithm finds 482 co-location rules with 1 km grid. With 2 km granularity 547 rules are detected from which 335 are present in both 1 and 2 km result sets, and 212 are unique for 2 km grid. The difference means that 2 km distance between grid points is too long for our dataset, where the average buffer size is 7.3 km, and its accuracy is comparatively low due to the less number of transactions. The 0.5 granularity grid reported 472 co-location rules as significant. From these, 426 are found with both 1 and 0.5 km grids, and 46 rules are identified only by 0.5 grid. As we can see, the difference between 0.5 and 1 km result sets is smaller than between 1 km and 2 km grids. As the distance between points in a grid decreases, the accuracy of the results improves.

6 Conclusion

In this paper, we proposed a new solution to the co-location mining problem. The transactionization step allows the conversion of spatial data into a set of

transactions. The usage of thresholds like in previous algorithms is replaced by the statistical test. In addition, our approach takes into account uncertainty of data. In order to decrease computation, the filtering techniques are presented. The experiments on real and synthetic datasets showed that our approach finds significant co-location patterns and rules.

References

1. Shekhar, S., Huang, Y.: Discovering spatial co-location patterns: A summary of results. In: Jensen, C.S., Schneider, M., Seeger, B., Tsotras, V.J. (eds.) SSTD 2001. LNCS, vol. 2121, pp. 236–256. Springer, Heidelberg (2001)
2. Huang, Y., Pei, J., Xiong, H.: Mining co-location patterns with rare events from spatial data sets. Geoinformatica 10(3), 239–260 (2006)
3. Yoo, J.S., Shekhar, S.: A joinless approach for mining spatial colocation patterns. IEEE Trans. on Knowl. and Data Eng. 18(10), 1323–1337 (2006)
4. Xiong, H., Shekhar, S., Huang, Y., Kumar, V., Ma, X., Yoo, J.S.: A framework for discovering co-location patterns in data sets with extended spatial objects. In: Proc. of the 2004 SIAM International Conference on Data Mining (2004)
5. Cressie, N.: Statistics for spatial data. Wiley series in probability and mathematical statistics: Applied probability and statistics. J. Wiley (1991)
6. Chou, Y.: Exploring spatial analysis in geographic information systems. OnWord Press (1997)
7. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proc. of the 20th International Conference on Very Large Data Bases, pp. 487–499 (1994)
8. Koperski, K., Han, J.: Discovery of spatial association rules in geographic information databases. In: Egenhofer, M., Herring, J.R. (eds.) SSD 1995. LNCS, vol. 951, pp. 47–66. Springer, Heidelberg (1995)
9. Morimoto, Y.: Mining frequent neighboring class sets in spatial databases. In: Proc. of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 353–358 (2001)
10. Estivill-Castro, V., Lee, I.: Data mining techniques for autonomous exploration of large volumes of geo-referenced crime data. In: Proc. of the 6th International Conference on Geocomputation (2001)
11. Barua, S., Sander, J.: SSCP: Mining statistically significant co-location patterns. In: Pfoser, D., Tao, Y., Mouratidis, K., Nascimento, M.A., Mokbel, M., Shekhar, S., Huang, Y. (eds.) SSTD 2011. LNCS, vol. 6849, pp. 2–20. Springer, Heidelberg (2011)
12. Getis, A., Jackson, P.H.: The expected proportion of a region polluted, by k sources. Geographical Analysis 3(3), 256–261 (1971)
13. Reggente, M., Lilienthal, A.J.: Using local wind information for gas distribution mapping in outdoor environments with a mobile robot. In: 2009 IEEE Sensors, pp. 1715–1720 (2009)
14. ArcGIS Desktop: Release 10, ESRI (2011)
15. Williams, R.G.: Nonlinear surface interpolations: Which way is the wind blowing? In: Proc. of 1999 Esri International User Conference (1999)

SOLAP4epidemiologist: A Spatial Data Warehousing Application in Epidemiology Domain

Assuntina Cembalo, Michele Ferrucci, Francesca Maria Pisano, and Gianpaolo Pigliasco

C.I.R.A., Italian Aerospace Research Center, via Maiorise snc, 81043 Capua (CE), Italy
{a.cembalo,m.ferrucci,f.pisano,g.pigliasco}@cira.it

Abstract. During last decades, the Provinces of Naples and Caserta, in the Campania Region, experienced a dreadful increase in the level of pollution as effect of documented practices of illegal waste dumping and burning. In the same period, an abnormal increase in deaths due to cancer diseases was registered. Up to now, no impact of waste treatment on human health has been scientifically proven, but it has not even excluded yet.

We believe that the availability of simple-to-use analytics tools may be of great help to epidemiologists in managing and querying the huge amount of heterogeneous data disposable for epidemiologic purposes.

This paper presents an innovative decision support application SOLAP4epidemiologist, based on Spatial Data Warehousing technologies (Spatial ETL, GIS, Spatial OLAP) able to integrate structured and geo-referenced data coming from different sources and to investigate them by means of user-friendly GUI, using statistical charts and maps representations.

Keywords: Spatial OnLineAnalyticalProcessing, Spatial Extraction-Transformation-Loading, Data Integration.

1 Introduction

Nowadays, Data Warehousing systems [1] represent a well-known and widely spread Decision Support System paradigm; they enable decision makers to easily navigate huge amount of heterogeneous data coming from different sources in various domains. During the last decade, Data Warehousing systems have represented the right answer to decision makers emerging analytics needs, following the explosive growth of available data, due to innovative technological solutions for data collection and storage. A more recent technological breakthrough is relative to the management of geographical data, which turns in the challenge of making the geographical component of information easily available for decision makers' analysis. Recently, Spatial Data Warehousing has been proposed as the evolution of Data Warehousing system [2-3] and a new type of spatio-temporal multidimensional data exploration approach was born called SOLAP [4-6].

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 97–109, 2013.

It was found that "about 80% of all data stored in corporate databases has a geographical component" [7] (which can be represented by addresses, cities, roads, rivers, states, regions, areas, etc.); this means that in order to allow a comprehensive analysis of data, enabling decision makers to make better and quicker decisions, therefore, it is necessary to include spatial information in the process of analysis. Nowadays, users have pressing needs of easy-to-use informative tools that allow them to intuitively analyze the spatial relationships that exist among data.

It is worth noting that people are used to thinking in a visual and spatial manner, so some spatial relationships are often easier to understand if they are presented visually. Some phenomena, such as space-time evolution or spatial distribution, are not easily viewable by means of traditional statistical charts or tables, which are typically used in Business Intelligence (BI) applications. From all these considerations the Spatial BI [8] comes. Spatial BI is intended as a combination of software, database, analytical tools and methodologies that, combining geographic data with other business data, enable organizations to acquire critical insights, make better decisions and to optimize important processes and applications [9].

The Spatial Data Warehousing approach can be a successful analytics solution in any domain featured by huge amount of heterogeneous data (included geo-referenced data), coming from different and spatially distributed data sources. The Data Scientist is in charge of rightly designing and developing the Decision Support system, compliant to the specific needs of end users and exploiting the available data. For our experimentation, we chose the Epidemiology domain and in detail the on-the-edge issue of potential impact of waste treatment on human wealth in the Provinces of Naples and Caserta [10], in the Campania region in southern Italy. Up to now no such impact has been scientifically proven, but it has not even excluded yet [11]. During last decades, the Provinces of Naples and Caserta experienced a dreadful increase in the level of pollution as effect of documented practices of illegal waste dumping and burning. In the same period, an abnormal increase in deaths due to cancer diseases was registered. In the Province of Naples, significant exceeding mortality percentages of male deaths in the 43% of the municipalities and of female deaths in the 47% of the municipalities were calculated by the World Health Organization [12]. Many researchers in the epidemiology domain dedicated their efforts to deeply analyze and crossing data including medical records, industrial sites distribution, landfills dislocation, air quality data. We believe that traditional data analysis techniques and systems are not sufficient to adequately support epidemiologists in carrying on their studies. In our opinion, Spatial Data Warehousing can represent an innovative and valuable analytics tool at disposal of epidemiologists in managing and querying the huge amount of heterogeneous data, disposable for epidemiologic purposes. Indeed taking advantage of data redundancy, that avoids performing too many join operations, a Spatial Data Warehousing system allows the user to obtain useful information in a very quick manner; moreover the data stored within the Spatial DW cover a time span much wider than that typically covered by data stored in an operational system, providing a historical overview of the phenomenon. Finally the display of query results on maps allows the epidemiologist to locate the distribution and frequency of diseases on a particular geographic area of interest faster.

This paper presents an innovative decision support application **SOLAP4*epidemiologist***, based on Spatial Data Warehousing technologies (Spatial

ETL, GIS, Spatial OLAP) able to integrate structured and geo-referenced data coming from different sources and to make them analyzable by means of user-friendly GUI, using statistical graphs and maps representations.

The developed software prototype is going to be integrated into the platform to be created as final product of the research project I.D.E.S. (Intelligent Data Extraction Systems). This project aims to build a technological facility for scientific survey and cooperation between C.I.R.A. (Italian Aerospace Research Center) and local government, as well as private operators and anybody else is interested in Campania in the fields of safety and prevention, environment, market intelligence etc.

We gathered users' needs and simulated data samples as described in Section 2, compliant to the statistical distributions presented in [12], in order to provide a realistic analytics environment to potential end-user. In Section 3 the architecture of the system and its main components are explained, while the technological overview of open-source tools for the Spatial DW is shown in Section 4. The development process of **SOLAP4*epidemiologist*** application, including the detail at component level, is discussed in Section 5. Finally, Section 6 presents our conclusions.

2 Users' Needs and Data Sources

Basing on an interview to an expert in epidemiology domain, we elaborated some functional and user interface requirements at system level, which guided the design and development of SOLAP4*epidemiologist*. In detail, we were requested to provide an informative tool able to allow easily the visualization and navigation of data on synchronized geographical maps, charts and tables, the export of querying results in excel format, the integration of heterogeneous data such as environmental, epidemiologic, municipalities' ones, the elaboration of many standard epidemiologic indicators. We turned those needs in systems requirements; but it wasn't enough.

A correct design of a Data Warehousing System [7] should be guided by a mixed user- and data-driven approach, taking into account synergistically available data sources and users' needs, in order to avoid developing a system which can potentially satisfy users' needs in end, but invalidated by the lack of adequate data. In order to avoid such danger and in lack of official data sources, we simulated likely datasets.

We reproduced statistical samples of domain data according to statistical distributions elaborated in the World Health Organization report [12]. We simulated epidemiological data collected from 1994 to 2002 in 196 Municipalities in the Provinces of Naples and Caserta. In more detail, the statistical distributions documented in [12] showed aggregations of data with respect to the various analysis dimensions (date of death, geographical residence, gender, age, disease causing death, etc.). From such starting point, we reproduced randomly more than 300000 records concerning death occurrences caused by 17 different types of diseases.

The simulated dataset was integrated with some other informative sources, such as: administrative and statistical data from the Italian Institute of Statistics (ISTAT), mortality data distribution over the 196 Municipalities coming from Local Health Organizations and Districts (ASL), and ICD-IX data (International Statistical Classification of Diseases and Related Health Problems) [13].

Finally, the data sources included a PostgreSQL database instance containing death occurrences (residence, age, gender, etc.), a shape file describing geographical and administrative information related to all Italian municipalities and a structured text file reporting information about local health organizations called ASL.

3 Spatial DW Architecture

The software application presented in this article is aimed at supporting the epidemiologist user in the analysis of large amounts of data, through an easy-to-use tool. So the application is designed as a Spatial Data Warehousing system, which is an advanced system of analysis that allows user to simply query large amounts of data, using Spatial OLAP (SOLAP) technology and through a user friendly interface, to obtain summary information which includes geo-referenced data. Therefore a Spatial Data Warehousing system realizes the integration of spatial data within the Data Warehousing at the level of each individual component (ETL, Data Warehouse, OLAP queries) [14]. The Data Warehousing system was extended to the case of the processing of geo-referenced data; so the Spatial Data Warehousing system realized is made up of the following main components:

1. *Epidemiologic Spatial DW*: it is the data base that stores both the structured and geo-referenced data, arranged to be queried in multidimensional manner; therefore it is conceived as a *Spatial Data Warehouse* (Spatial DW), that is a Data Warehouse with the ability of handling geometric data types, which derive from three basic geometrical types, used to represent the various spatial objects: points, lines and polygons. Moreover, in addition to the traditional thematic dimensions, a Spatial DW also supports three types of spatial dimensions [15]:
   - *Non-geometric spatial dimensions*, which use the names of places, such as, for example, "Naples", "Province of Caserta", etc.
   - *Geometric spatial dimensions*, which comprise geometric shapes (ex. polygons to represent country boundaries) that are spatially referenced, for all dimension members and at all levels of details.
   - *Mixed spatial dimensions*, which comprise geometric shapes for a subset of members or levels of details.

   Besides traditional measures, calculated using *sum, count, average, or percentage* operators, a Spatial DW allows the use of spatial measures, defined through the topological operators, such as *intersection, union, difference, distance, equality, tangency, inclusion*, etc.
2. *Spatial ETL Application*: it is the software application, based on Spatial ETL technology (Extract, Transform, Load), that extracts, transforms and loads geo-referenced data within the Epidemiologic Spatial DW repository. The application extracts information from external structured and heterogeneous sources, including also standard storage formats of geographic data (such as shape files); then the extracted data are integrated and geo-referenced, and finally they are loaded into the Spatial DW.
3. *Solap4epidemiologist*: the client-server software application for multidimensional querying, which includes SOLAP interface and is based on metadata hypercube semantics. It provides the user with a user-friendly web interface, which allows the

epidemiologist to simply query the geo-referenced data, stored in the Epidemiologic Spatial DW, through a SOLAP server. The latter enables the interpretation and execution of the MDX query [16], built by the user through the selection of the measures and the dimensions of analysis. Finally the SOLAP interface allows the display of query results through maps, tables, and other types of graphs.

The system layered architecture is shown in the following figure (Fig. 1)

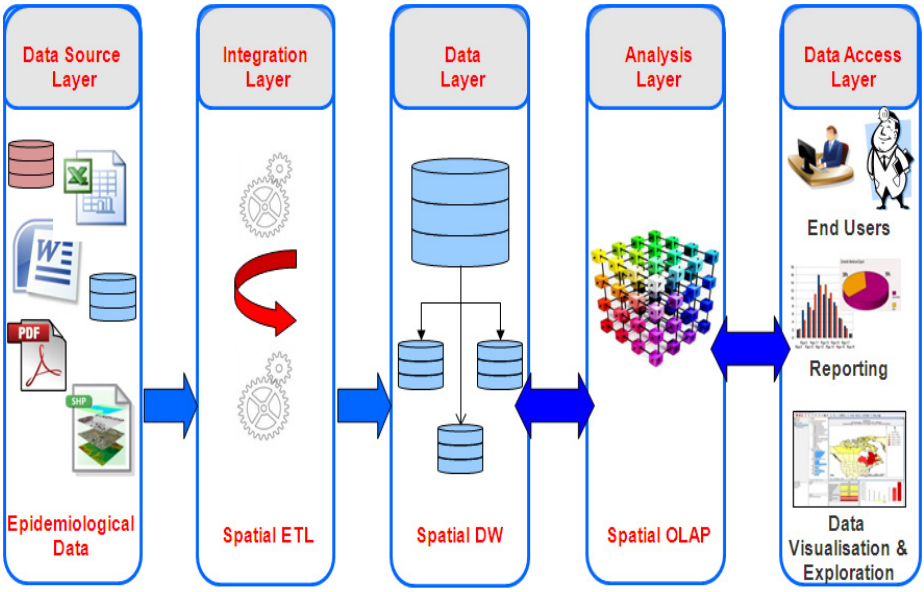


Fig. 1. Spatial Data Warehousing layered architecture

4 Open-Source Tools for Spatial DW Systems

One of the aims of the project was to make the entire Data Warehouse system using only open-source tools. To do this, a detailed technological recognition of available open-source applications was carried out. The following tools were identified:

1. GeoKettle 2.0 [17] is a release with GIS capabilities of the general-purpose ETL tool PDI (Pentaho Data Integrator), formerly known as Kettle, GPL-licensed and published by Spatialytics. Thanks to its easy-to-use GIS data managing, GeoKettle has been chosen to implement the Spatial ETL layer of the system.
2. PostGIS 1.5 [18] is the spatial extension of the open-source PostgreSQL 9.1 database, and allows to store the GIS database objects; PostGIS also implements and extends the standard functions defined in the Open GIS Consortium (OGC) [20]. It has been chosen to implement the data storage layer of the system.
3. GeoMondrian 1.0 [19] represents the spatial extension of the OLAP server Mondrian, and provides a consistent integration of spatial objects in the OLAP cubes. The tool implements also a native geometric data type and provides spatial

extensions to MDX language, so it has been chosen to be the Spatial OLAP server of the system.
4. GeoBI is a web application developed and distributed by Camptocamp [21]; it shows the results of multidimensional queries on three synchronized displays that contain a map, a table and other statistical graphs. It is the Spatial OLAP client of the system.
5. Tomcat 7 [22] is the well-known web container distributed by Apache Software Foundation, able to manage and support JSP pages, that is necessary to deploy both GeoBI and GeoMondrian, allowing users to connect to the system by the use of a web browser.

5 SOLAP4epidemiologist Development Process

This chapter describes the development process of the Spatial Data Warehousing application aiming at supporting the epidemiologist in analyzing the huge amount of heterogeneous data, related to the epidemiologic domain and located in the provinces of Naples and Caserta.

5.1 Spatial DW Design

According to the standard for the Data Warehouse, we extended the methodology proposed in [23] to manage geo-referenced sources, in order to design the Epidemiologic Spatial DW; so we used the graphical model of the Dimensional Fact Model to define the conceptual model of the Spatial DW.

The choice of facts, measures, dimensions and hierarchies of the fact schema was driven by the input datasets on one hand and the basic concepts of epidemiology on the other hand. Epidemiology is defined as the study of frequency, distribution and determinants of health/disease at the population level. The five keywords in the definition enclose the questions addressed by epidemiology:

- *Frequency:* HOW and WHEN diseases appear;
- *Distribution:* WHERE diseases are present;
- *Determinants:* WHAT is the cause of diseases;
- *Health/disease:* in addition to sick individuals, epidemiology even studies healthy ones;
- *Populations:* analyses are performed at the level of population, which is generally stratified on the basis of various standard factors, such as age and sex.

All these considerations have led to the identification of the main fact "mortality" and the creation of a fact schema which includes seven thematic dimensions, described below (Fig. 2):

- The dimension *Disease*, which contains all cancers and non-tumor diseases, which caused the death of considered patients. For each disease, it is given the corresponding ICD9-CM code [13]*;* the latter was also used to define a taxonomy consisting of 34 diseases. The hierarchy of the dimension *Disease* is characterized by three levels: *category, subcategory, disease.*

- The dimension *Sex*, which refers to the sex of the patients.
- The dimension *Age*, which contains all the ranges of the patients' age, consisting of 5-year intervals, from *0-5* to *66-70*, plus the *71+* range.
- The dimension *Profession*, related to the patients' job. The dimension hierarchy is characterized by the first three levels of the *ISTAT classification of professions* [24]: the first level consists of 9 *major occupational groups*, the second level includes 37 *professional groups*, and the third level contains 129 *professional classes*.
- The dimension *Education Level*, which refers to the degree of education of considered patients.
- The dimension *Civil State*, which allows to specify whether the dead patient, was married or not.
- The temporal dimension *Date of Death*, which contains the date of death of considered patients. It is hierarchically structured as follows: *date of death*, *month*, *year* and *quadriennium*.

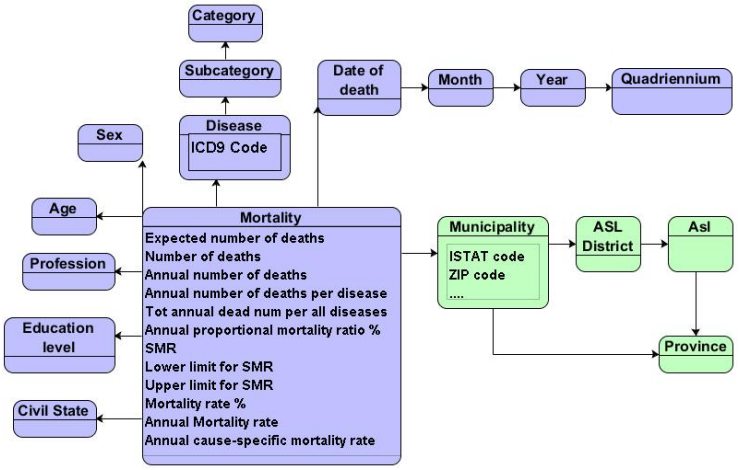


Fig. 2. Conceptual model of the Epidemiologic Spatial DW

In addition, we designed the spatial dimension *Geography*, which refers to the residence municipality of the patient. It is a spatial geometric dimension, since all members, at all levels of detail, are geometric shapes, in particular polygons or multi-polygons. For the spatial dimension we used two converging hierarchies:

- *Municipality / ASL District / ASL / Province*
- *Municipality / Province*

The levels used in the two converging hierarchies are described in detail below:

- *Municipality*: contains the 196 Municipalities in the Provinces of Naples and Caserta. Data relating to Municipalities were recovered from the ISTAT site [25] and the paper [12], and refer to the year 2001. For each city of the designed spatial dimension, it is specified a set of descriptive attributes such as *ISTAT code, ZIP code, city population, population residing in areas of impact, index of environmental pressure from waste disposal, index of socioeconomic deprivation*

(it represents, at the municipal level, the material and social hardship of the population).
- *ASL District*: contains the ASL Districts to which Municipalities in the Provinces of Naples and Caserta belong.
- *ASL*: contains the Local Health Authorities codes of Caserta and Naples.
- *Province*: contains information on the Provinces of Naples and Caserta.

The conceptual design of the Epidemiologic Spatial DW also included the definition of a set of standard indicators used in epidemiology; they can be calculated with respect to all dimensions' combinations. Typically the results of epidemiological measurements are expressed as "proportion", "ratio" or "rate", so for the conceptual model of the Spatial DW we identified the following measures, peculiar in epidemiological studies:

- *Expected number of deaths*
- *Number of deaths*
- *Annual number of deaths*
- *Annual number of deaths per disease*
- *Total annual dead number per all diseases*
- *Annual proportional mortality ratio as a percentage*
- *Standardized Mortality Ratio (SMR)*
- *Lower limit of the confidence interval for SMR at 95%*
- *Upper limit of the confidence interval for SMR at 95%*
- *Mortality rate as a percentage*
- *Annual Mortality rate*
- *Annual cause-specific mortality rate.*

Starting from the facts, the dimensions of analysis and the measures identified in the conceptual design phase, we built the logical model of the Epidemiologic Spatial DW, using the open-source database *PostGIS 1.5*. Therefore we designed a star schema, characterized by one fact table related to mortality (Fig. 3). For each dimension of analysis identified in the previous step we built a table, which includes the set of all the attributes that describe the dimension at different levels of aggregation. For the geography dimension we had to populate the PostGIS meta-table *geometry_columns*, specifying the spatial reference system, the geometric type and the number of spatial coordinates. For the fact table, then, we built a composite key, which consists of the set of foreign keys pointing to the dimension tables related to the fact table. Finally we defined the metadata hypercube, consisting of an XML file that defines the measures and the hierarchies of analysis dimensions, specifying all the different levels of aggregation, through the spatial extension of the *Multidimensional Expressions* (MDX) [16] language, which allows managing the Geometry data type for spatial dimensions. Since measures are defined and implemented in the hypercube of metadata, it is very easy to modify them or add new ones. The metadata hypercube was built using the open source tool *GeoMondrian Workbench 1.0*.

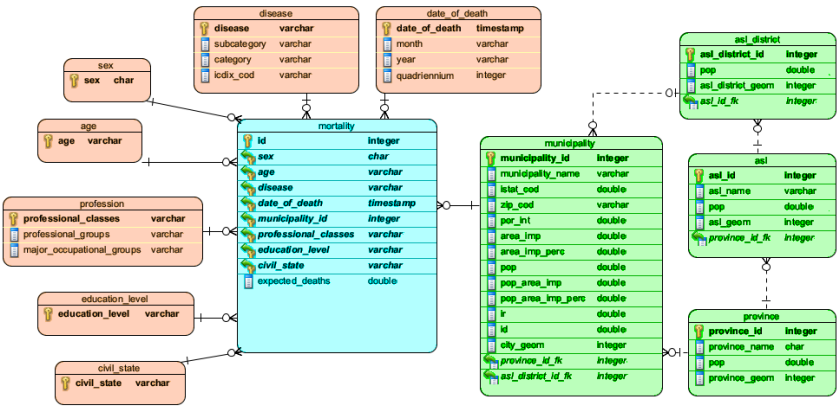


Fig. 3. Logical model of the Epidemiologic Spatial DW

5.2 Spatial ETL

The whole design process of Spatial ETL was guided by two different approaches: a model-driven approach and a data-driven approach. The model-driven operations were mainly focused on the logical model of the Epidemiologic Spatial DW (Fig. 3): a different loading-data transformation was designed for each of the 12 tables of the logical model, and the execution of each transformation should satisfy with the following architectural constraints:

- The fact table "mortality" has to be the last to be populated, as it contains all the foreign keys to the various dimension tables.
- The geographical dimension consists of two hierarchies, MUNICIPALITY -> PROVINCE and MUNICIPALITY -> ASL_DISTRICT -> ASL -> PROVINCE, so the loading order should be: Province, ASL, ASL District and finally Municipality.

These constraints were satisfied by the design of 12 different loading transformations, among which the transformation related the fact table is the last to be executed and the 4 transformations concerning geographical dimension are organized in a single ETL job, thus ensuring the correct temporal order of execution.

Fig. 4. ETL job to populate geographic dimension

The data-driven approach is useful to determine how to fill each table that is which operations we need to get the desired result, starting from the available data.

Input information was organized into 4 distinct data structure:

1. Municipalities (196 rows): this table stores administrative attributes of the Municipalities of the Provinces of Naples and Caserta, that are not necessary or useful to disaggregate for the purpose of the system, such as total number of residents, administrative area, ISTAT code, postal code and so on.
2. Patients (333,346 rows): it is the biggest table, which contains information regarding the patients detected in the period 1994-2001 classified according to the type of disease found (tumor diseases, non-tumor diseases).
3. Administrative limits: this is a shape file, retrieved from ISTAT website, containing the geographic information (i.e. administrative limits) about all the Italian municipalities, in the form of multi-polygons.
4. ASL data: this is a structured text file, containing for each Municipalities of the Provinces of Naples and Caserta the competent ASL and ASL district.

The aim of the Spatial ETL application is to extract and transform data from these sources, enriching the information by combining the data from the different sources, to finally load the transformed data in the appropriate table of the designed Spatial Data Warehouse. The transformations related to the geographical dimension enrich the data for the tables of this dimension, extracting and manipulating ISTAT multi-polygons, related to administrative perimeters of each Municipality, and grouping them to create the administrative perimeters of ASL districts, ASL and Provinces. The following (Fig. 5) is, by way of example, the transformation for the loading of the table "Municipality", which is essentially constituted by these steps:

1. Extraction and Selection of data from the Municipalities source.
2. Extraction and Selection of GIS data from the ISTAT data source.
3. Sorted Join of these two fluxes.
4. A sequence of Database Lookup Steps to retrieve the foreign key from the previously loaded tables. (ASL_DISTRICT, ASL and PROVINCE).
5. Loading Database Step to write out the rows into the table "Municipality".

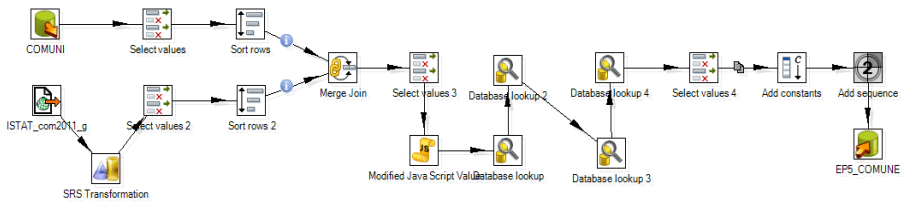


Fig. 5. ETL transformation to populate "Municipality" table

All data transformations and jobs have been implemented through the use of the open-source Spatial ETL tool GeoKettle 2.0 [17]. The main problem faced during the ETL activities concerns the geo-referencing of data: indeed, it is essential that all records belonging to the tables of the geographical dimension of the DW are geo-referenced; this means each geographical record must be associated in a unique way to a geometric multi-polygon describing its administrative limits. The only available source of geometric data (retrieved from ISTAT - Italian Institute of Statistics) provides only

administrative boundaries of the municipalities, while no geometric data is directly provided about other geographic entities. This means that the records belonging to the Municipality table can be directly geo-referenced, while it remains open the problem of geo-referenced records that belong to geographic entities of the upper level (ASL District, ASL and Province). This problem was solved by the use of "Group-by" blocks within the transformations, to group the municipalities on the basis of geographical entities of higher level and obtaining administrative boundaries of such entities by the use of the "Geometric UNION" function provided by GeoKettle.

5.3 Spatial OLAP Web Application

To perform multidimensional queries we designed the web application *Solap4epidemiologist*, based on Spatial OLAP query interface. The application uses the open-source tool GeoMondrian 1.0 as Spatial OLAP server. We used the open-source application GeoBI by Camptocamp as SOLAP client, which shows the results of queries on three synchronized displays that contain a map, a table and other graphics, such as pie charts and histograms (Fig. 6). We used Tomcat 7 as web container. Through the GeoBI GUI, the user can choose from a combo-box the dimensions and the measures of interest; he can also get different levels of analysis detail, through drill-down and roll-up operators, by clicking on both the map and the table. Moreover the user can select the colors of the map and the type of statistical graph to be overlapped on the map itself. The following figure (Fig. 6) shows the number of deaths, occurred throughout the time period studied, in the provinces of Naples and Caserta, by sex and type of illness (cancerous or non-cancerous disease).

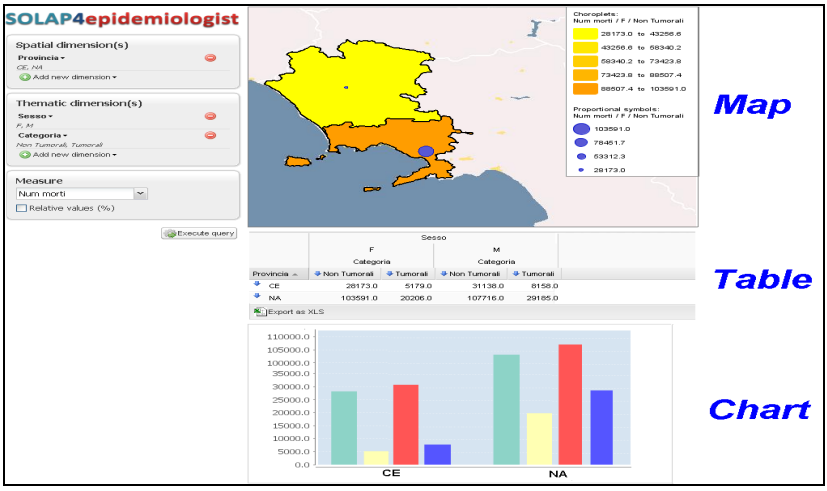


Fig. 6. Web application *SOLAP4epidemiologist* for multidimensional queries

To allow users to process query results in one of the most used format, we worked to integrate in the web application the function of exporting to Microsoft Excel® the table (datagrid) containing results from a spatial query. We integrated this functionality by simply adding a "Export as XLS" button on the grid panel. When the user presses the

button, a process is launched resulting in opening an Excel worksheet and filling it by taking data line by line from the effective grid in the web page. The only care we had to take for coding this function is that, dealing with a multidimensional query, we had to properly capture more lines for column headers, even by repeating group of headers, where there would be nested dimensions of analysis.

6 Conclusions

Recently, the extension to geographic data of the well-known multidimensional analysis paradigm of Data Warehousing uncovered new interesting opportunities of providing useful analytics tools able to take advantage of the geographic component of data, without sacrificing the successful multidimensional analysis approach. Scientists and industrial decision makers found in the OLAP technology the right answer to their needs emerged from the explosive growth of data to be analyzed: the extension to Spatial OLAP represents the technological key-enabler to perform integrated analysis of also geo-referenced data by making use of client tools, which don't require any informatics skill to end users. We applied the SOLAP approach to Epidemiology domain, featured by the availability of huge amount of heterogeneous data sources (structured and geo-referenced), geographically distributed. So far we realized a Spatial Data Warehousing application, **SOLAP4*epidemiologist***, able to integrate big bunches of data including medical records and administrative information and to make them available for advanced queries. We made use of open-source tools.

SOLAP4*epidemiologist* will let the epidemiologists to benefit of a simple-to-use querying tool which allow them to perform sophisticated queries over big bunch of data by just selecting from combo-boxes the criteria of analysis and the quantitative aspects they intend to investigate. Moreover, we intended to offer a client tool by means of which end users can take whole benefit of the geometric component of data. Epidemiologists will be able to get insight different levels of analysis detail by clicking on maps. SOLAP4*epidemiologist* completely fulfills the challenging tasks of performing advanced and quick analysis and offering querying results visualizations intuitively intelligible, without requiring any informatics skill to end users. The developed software prototype is going to be integrated into the platform to be created as final product of the research project I.D.E.S. (Intelligent Data Extraction Systems).

Acknowledgements. This work has been carried on within the research project I.D.E.S. – Intelligent Data Extraction Systems, funded by the Campania Region and EU within the framework of POR Campania FESR 2007 – 2013.

References

1. Turban, E., Aronson, J.E., Liang, T.-P., Sharda, R.: Decision Support and Business Intelligence Systems, 8th edn. Pearson International Edition (2007)
2. Bédard, Y., Merret, Han, J.: Fundamentals of Spatial Data Warehousing for Geographic Knowledge Discovery. In: Geographic Data Mining and Knowlwdge Discovery. Research Monographs in GIS, ch. 3, pp. 53–73. Taylor & Francis (2001)
3. Rivest, S., Bédard, Y., Marchand, P.: Toward better support for spatial decision making: defining the characteristics of Spatial On-line Analytical Processing (SOLAP). Geomatica 55(4), 539–555 (2001)

4. Rivest, S., Bédard, Y., Proulx, M.J., Nadeau, M.: SOLAP: a new type of user interface to support spatio-temporal multidimensional data exploration and analysis. In: Workshop International Society for Photogrammetry and Remote Sensing (ISPRS), Quebec, Canada (2003)
5. Rivest, S., Bédard, Y., Proulx, M.J., Nadeau, M., Hubert, F., Pastor, J.: SOLAP technology: Merging business intelligence with geospatial technology for interactive spatio-temporal exploration and analysis of data. ISPRS Journal of Photogrammery & Remote Sensing 60, 17–33 (2005)
6. Bimonte, S., Tchounikine, A., Miquel, M.: Spatial OLAP: Open issues and a Web based prototype. In: 10th AGILE Int. Conf. on Geographic Information Science (2007)
7. Golfarelli, M., Rizzi, S.: Data Warehouse Design – Modern Principles and methodologies. McGraw-Hill (2009)
8. Eldridge, A.: Spatial Business Intelligence. In: Coast, G. (ed.) Microsoft Tech, Australia, August 24-27 (2010), http://www.slideserve.com/jenski/spatial-business-intelligence
9. Malinowski, E., Zimanyi, E.: Spatial Data Warehouse: some solutions and unresolved problems. In: IEEE International Workshop on Databases for Next Generation Researchers (2007)
10. Impact of waste treatment on human health in Campania region (Italy), http://www.euro.who.int/en/what-we-do/health-topics/environment-and-health/health-impact-assessment/country-work/impact-of-waste-treatment-on-human-health-in-campania-region-italy
11. Relazione finale del gruppo di lavoro sulla situazione epidemiologica della regione Campania ed in particolare delle province di Caserta e Napoli (città esclusa), con riferimento all'incidenza della mortalità per malattie oncologiche (2013), http://www.salute.gov.it/imgs/C_17_pubblicazioni_1883_allegato.pdf
12. Study on the health impact of waste treatment in Campania region (Italy) (2007), http://www.protezionecivile.gov.it/cms/view.php?cms_pk=16909&dir_pk=395
13. International Classification of Diseases ICD-9-CM, http://www.icd9data.com/
14. Miller, H.J., Han, J.: Geographic Data Mining and Knowledge Discovery, 2nd edn. Chapman & Hall/CRC (2009)
15. Spatial OLAP Concepts, http://spatialolap.scg.ulaval.ca/concepts.asp
16. Tutorial: Introduction to Multidimensional Expressions (MDX), http://www.mdxtutorials.net/ (retrieved January 2011)
17. GeoKettle, http://www.spatialytics.org/projects/geokettle/
18. PostGIS, http://postgis.net/
19. GeoMondrian, http://www.spatialytics.org/projects/geomondrian/
20. Open Geospatial Consortium, http://www.opengeospatial.org/
21. CamptoCamp, http://www.camptocamp.org/
22. Tomcat, http://tomcat.apache.org/
23. Caranna, V.: Data Warehouse - III parte: Definiamo un modello progettuale per DWH. Mokabyte, 158 (2011)
24. Istat Classification of Professions, http://www.istat.it/it/archivio/18132
25. Istat Territorial Bases and Census Variables, http://www.ISTAT.it/it/archivio/44523

Document Difficulty Framework for Semi-automatic Text Classification

Miguel Martinez-Alvarez[1], Alejandro Bellogin[2], and Thomas Roelleke[1]

[1] Queen Mary, University of London
{miguel,thor}@eecs.qmul.ac.uk
[2] Centrum Wiskunde & Informatica (CWI)
alejandro.bellogin@cwi.nl

Abstract. Text Classification systems are able to deal with large datasets, spending less time and human cost compared with manual classification. This is achieved, however, in expense of loss in quality. Semi-Automatic Text Classification (SATC) aims to achieve high quality with minimum human effort by ranking the documents according to their estimated certainty of being correctly classified. This paper introduces the Document Difficulty Framework (DDF), a unification of different strategies to estimate the document certainty, and its application to SATC. DDF exploits the scores and thresholds computed by any given classifier. Different metrics are obtained by changing the parameters of the three levels the framework is lied upon: how to measure the confidence for each document-class (evidence), which classes to observe (class) and how to aggregate this knowledge (aggregation). Experiments show that DDF metrics consistently achieve high error reduction with large portions of the collection being automatically classified. Furthermore, DDF outperforms all the reported SATC methods in the literature.

1 Introduction and Motivation

Automatic Text Classification (TC) provide much faster and cheaper classification than human experts. However, even though there have been large improvements in the last decades, human experts achieve higher quality. Since the introduction of automatic classifiers, two alternative options can be applied. Firstly, a full-automatic classification system is applied, where every document is classified according to the decisions made by the classifier. Secondly, a completely manual classification is performed, where human experts classify each document. The main drawbacks of the latter option are its huge cost and potential unfeasibility for large collections. On the other hand, the quality achieved by the manual approach will be higher. Full automatic TC is preferred if large datasets are used (i.e. webpage classification) or when lower quality is not as important as the human effort required. On the other hand, manual classification is the best option for systems that require high-quality and have medium size such as law or medical data. In general, the time saved by using an automatic system is leveraged with the possible quality loss with respect to the manual classification.

This research focuses on an intermediate solution using Semi-Automatic Text Classification (SATC) [8,1]. The main goal is to achieve high quality with minimum human

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 110–121, 2013.

effort or, more specifically, to use human experts only for the documents that the automatic system is more likely to misclassify. Therefore, maximising the quality, while minimising the cost. Given a set of documents to be labelled and a specific classifier, a SATC algorithm needs to rank the documents according to the likelihood of their classification decisions to be correct. The ranking allows experts to inspect documents iteratively, starting with the most uncertain ones, until a specific point, where the rest of the documents are automatically assigned. In addition, this strategy can be applied with variable resources over time (i.e. less human experts could be available).

This paper introduces the **Document Difficulty Framework (DDF)**, a family of document certainty algorithms, and its application to SATC. DDF exploits the document-class confidence scores computed by a classifier and the class thresholds given by any class-based thresholding strategy to calculate the certainty of each document. This implies that the class scores for all documents have to be computed. The framework defines an array of different metrics, depending on three different dimensions: how the document-class evidence is computed (evidence), which classes will be considered (class), and how to aggregate a document-based certainty (aggregation).

The remainder of this paper is organised as follows: Section 2 presents the background and related research. Section 3 introduces DDF, and analyses its different variations. Experiments are explained and analysed in Sections 4. Finally, Section 5 concludes the paper and presents the future work.

2 Background and Related Research

2.1 Multi-Label Text Classification

In multi-label TC (ML-TC), each document can belong to multiple classes. As a result, a classification process assigns a boolean value to each pair $(d_j, c_i) \in D \times C$, where D is a set of documents and C is a set of predefined categories [9]. To achieve this goal, most classifiers use a two-step procedure. Firstly, the classifier produces a score for each pair (d_j, c_i), and then a thresholding strategy decides, for each of the scores, if that value implies that the document belongs to the class (T) or not (F). Given a classifier σ and a threshold function δ, both processes can be mathematically denoted as: $\sigma \in \Sigma : D \times C \rightarrow [0, \infty]$, $\delta : \Sigma \times D \times C \rightarrow \{T, F\}$.

There are different types of thresholding strategies, based on documents or topics. For instance, using RCut [12], the R classes with higher score for a specific document are selected. On the other hand, SCut computes the threshold per class which maximises its quality (i.e. measured using F_1) [12]. If the number of training examples for a category is small, SCut can compute a very high or very low threshold from a global point of view. To address these drawbacks, SCutFBR.1 and SCutFBR.0 were introduced [12]. These metrics modify the behaviour of SCut in the case that the quality obtained for a class is lower than a value (*fbr*). The former strategy uses the highest ranked document as threshold value, while SCutFBR.0 assigns an infinite value. As a result, SCutFBR.0 does not assign any document in any class with lower quality than fbr.

2.2 Semi Automatic Text Classification (SATC)

SATC assumes that neither manual, nor full automatic classification is the optimum solution. This situation appears when full automatic classification achieves lower than required quality, and a full manual classification is either too expensive or unfeasible due to lack of resources. The foundation of SATC is that if we are able to separate the documents with high probability to be correctly classified, and the ones that are probably wrong, the latter can be inspected by human experts while the former will be automatically classified. As a result, the resources (the human experts) are optimised, while the quality remains high. To solve this task, SATC methods rank the documents to be classified according to their uncertainty. SATC assumes that the documents with higher certainty are probably better classified, whereas the documents with higher uncertainty are incorrectly classified. Therefore, the quality is maximised if the human annotators inspect the documents starting from the ones with higher uncertainty. The possibility of combining human and automatic classification has been suggested before [13,9,6]. However, only two approaches have been proposed: Document Difficulty and Utility-Theoretic Ranking. Both of them have been proven to be well-suited for SATC. Nonetheless, their performance have never been compared in the literature.

Document Difficulty [8] uses the classification scores and thresholds as evidence to compute the document certainty, where the labels with greater certainty are those with larger relative difference with respect to their threshold. The aggregation of label confidences is performed by averaging the confidences for those classes the document will be labelled in. The reason for the name similarity is that DDF extends and generalises the principles we explained in [8], mainly exploiting the classification scores and the threshold values within a classifier-independent framework. However, while only one method was proposed in that work, DDF represents a family of certainty metrics, where the previous metric appears naturally as an special case. In addition, the evaluation is also different. Our previous work evaluated the quality of the subset of automatically classified documents, instead of the whole collection. This research analyses the quality of the full test set, including both manual and automatic classified documents subsets.

The Utility-Theoretic Ranking (UT) method [1] optimises the global quality of the system, exploiting the potential benefit of manually inspecting each document, using the confidence scores of a classifier, and the gain in terms of quality that could be achieved, if that label is actually correct. The main conceptual difference with our approach is that UT exploits the collection information, whereas DDF focuses on each document independently. Furthermore, DDF exploits threshold information, and class filtering for the aggregated document certainty.

Similar to SATC, Active Learning (AL) ranks documents according to their benefit in the learning process, selecting which unlabelled documents should be manually labelled and being included as training examples. However, SATC focuses on the classification step, while AL operates in the training phase, selecting the documents from which the classifier can learn the most. Extensive research has been done related to single-label AL [7,10]. However, very limited research has tried to address the same problem in a multi-label environment [4,11].

2.3 Semi Automatic Text Classification Evaluation

SATC is evaluated using traditional classification quality measures such as micro-averaged-F_1, once the human and the automatic decisions have been combined. This approach, introduced by Berardi *et al.* [1], provides quality values for different proportions of the collection being automatically classified, where the most uncertain documents are manually classified. In addition, the goal of SATC is not only to compute the quality but to analyse how it varies depending on the number or documents considered. Therefore, quality variations with respect to the full automatic quality with the same classifier are also computed. The main issue is that the relative quality increase is fully dependent on the base quality, when all the documents are automatically classified. For instance, in some cases, a 100% quality increase is impossible (i.e. full automatic classification achieving 95% quality), while more than 100% is possible for others, making a comparison over different classifiers impossible. Berardi *et al.* [1] addressed these challenges and introduced two alternatives based on the error reduction with respect to the full automatic system, instead of its quality increase. Error Reduction at rank (ER) measures the error reduction with a specific number of documents being automatically classified, where $E_p(n)$ models the error (defined as 1-quality) achieved by a classifier p with n documents being manually classified,

$$\mathrm{ER}_p(n) = \frac{\mathrm{E}_p(0) - \mathrm{E}_p(n)}{\mathrm{E}_p(0)} \tag{1}$$

Normalised Error Reduction at rank (NER) subtracts the error reduction at rank n achieved by a random ranker ($\frac{n}{|Te|}$, where $|Te|$ is the size of the documents to be classified) from ER in order to obtain more meaningful quality values,

$$\mathrm{NER}_p(n) = \mathrm{ER}_p(n) - \frac{n}{|Te|} \tag{2}$$

A third metric was also proposed by the same authors to include the specific position of each document into the evaluation: Expected Normalised Error Reduction (ENER) exploits the probability of a human expert inspecting n documents ($P_s(n)$),

$$\mathrm{ENER}_p = \sum_{n=1}^{|Te|} P_s(n) \cdot \mathrm{NER}_p(n) \tag{3}$$

$P_s(n)$ can follow different probability distributions, Berardi *et al.* [1] suggested the definition shown below, where p models the probability of the next document to be inspected,

$$P_s(n) = \begin{cases} p^{n-1} \cdot (1-p) & \text{if } n \in \{1, ..., |Te|-1\} \\ p^{n-1} & \text{if } n = |Te| \end{cases} \tag{4}$$

The value of p can be defined as a function of the expected ratio (ξ) of documents being manually classified. $p = \frac{1}{\xi \cdot |Te|}$. Therefore, p is computed for different expected ratios of manually classified documents. Extended information about this evaluation can be found in Berardi *et al.* [1].

Table 1. DDF Levels. c_i represents a class, d a document, and s the classifier's score for each document-class pair. $t(c_i)$ is the threshold for c_i, and $q(c_i)$ is the estimated quality for c_i.

Evidence ϵ; given $d, c_i, t(c_i)$		Class γ; given $d, c_i, t(c_i), \epsilon$		Aggregation α; given $\gamma(\epsilon, d, \cdot)$	
S	$s(d, c_i)$	**A**	$\epsilon(d, c_i)$	**M**	$\max_{c_i \in C} \gamma(\epsilon, d, c_i)$
A	$\ln(1 + \lvert s(d, c_i) - t(c_i) \rvert)$	**P**	$\begin{cases} \epsilon(d, c_i) & \text{if } s(d, c_i) \geq t(c_i) \\ 0 & \text{otherwise} \end{cases}$	**A**	$\text{avg}_{c_i \in C} \gamma(\epsilon, d, c_i)$
R	$\ln(1 + \frac{\lvert s(d, c_i) - t(c_i) \rvert}{t(c_i)})$			**W**	$\text{avg}_{c_i \in C} q(c_i) \cdot \gamma(\epsilon, d, c_i)$

3 Document Difficulty Framework for SATC

The Document Difficulty Framework (DDF) is a family of document certainty metrics within the context of TC. DDF extends and generalises the principles we explained in previous research [8], exploiting the classification scores and the threshold values within a classifier-independent framework to compute the document certainty. This computation is divided into three different levels, inspired by the comparison of multi-label AL metrics by Esuli *et al.* [4]: evidence, class and aggregation. The **evidence** level computes the confidence value for each document and category, using their classification score and the class threshold. The **class** level specifies which classes are to be considered in the final aggregation. The **aggregation** level combines the filtered confidence levels, producing a document-based certainty.

DDF is based on the composition of the three transformation functions, one for each level, where ϵ represents the evidence, γ the class, and α the aggregation level,

$$\text{certainty}(d) = \alpha\left(\{\gamma(\epsilon(d, \cdot))\}\right) \tag{5}$$

A method within the framework consists in a specific strategy for each level. Table 1 summarises the different candidates analysed herein for each DDF levels. Each method is represented as the concatenation of three letters, representing the strategy followed in each one of the levels. For instance, following Table 1, the difficulty measured by the APA variation is defined as follows,

$$\text{certainty}(d) = \text{avg}_{c_i \in C : s(d, c_i) \geq t(c_i)} \ln(1 + \lvert s(d, c_i) - t(c_i) \rvert) \tag{6}$$

This method considers absolute distance for evidence function (A), only for positive classes (P), and computing the final aggregation as the average topic confidence (A).

3.1 Evidence Level

The evidence level is responsible for computing a confidence value for a document-topic pair. It is modelled as a function $\epsilon \in \mathcal{E} : D \times C \rightarrow [0, \infty]$, where D denotes the set of documents, and C the classes. Three candidates are considered for this level: **score** (S), **absolute** difference (A) and **relative** difference (R).

The first strategy (S) follows the same principle as relevance sampling [7], exploiting the score obtained by the classifier. It assumes that the higher the value the more relevant the score. Therefore, classes with higher scores are the ones with more certainty.

The second method (A) exploits the score and the threshold, assuming that larger distances imply lower uncertainty and higher chance that the document is correctly classified. This assumption is similar to uncertainty sampling [7]. A logarithmic function is applied to limit the effect of very large differences.

The last method (R) applies the same principles as the difference approach. However, it uses a relative difference instead of the absolute value. The rationale is that the absolute distances can be misleading. For instance, a distance of 0.2 would be much more important if the threshold is 0.05 than if it is 0.6.

3.2 Class Level

The class level behaves as a filter, selecting whether to exploit the certainty of a specific label, and hence, if it will be available at the next aggregation step or not. It is defined as a function $\gamma \in \Gamma : \mathcal{E} \times D \times C \rightarrow [0, \infty]$, where a composition with an element $\epsilon \in \mathcal{E}$ would be applied. Two candidates are considered for this level: **all** (A) and **positive** (P).

The first strategy (A) consists on not applying any filtering, hence considering all the confidence scores for a specific document.

The second method (P) selects the classes for which the classification score is higher or equal than the threshold. These are the classes which will be assigned to the document if automatic classification is applied. This strategy aims to focus the difficulty computation on the positive labels. In TC, the positive labels are more representative that the negative ones due to the fact that the number of positive classes for a specific document is usually much smaller than the number of negative ones. For example, the average number of classes per document in `Reuters-21578` is 1.24, while the number of classes is 90. This approach assumes that if all classes are observed, the document certainties are somehow diluted because most of the documents will obtain a high confidence that do not belong to a large subset of the classes.

3.3 Aggregation Level

When multi-label data is used, a certainty value per document has to be provided, since the ranking of the labels can not be used to select nor to rank documents. For example, even if 90% of the most certain labels are selected, it is impossible to decide which documents should be automatically classified. For this reason, the filtered evidence per class should be combined into a single certainty metric for each document. This level is defined as a function $\alpha \in \mathcal{A} : \{\Gamma\} \times D \rightarrow [0, \infty]$. Typically, it will be applied to the set of possible functions $\gamma \in \Gamma$, one for each class $c_i \in C$. This is equivalent, taken a document d and an evidence level function ϵ as inputs, to the set $\Gamma(\gamma, \epsilon, d) = \{\gamma(\epsilon, d, c_i) : c_i \in C\}$. It should be noted that a one-to-one relation exists between the set of classes used in the definition of Γ and Γ itself, and thus, this notation could be simplified as in Table 1. Three candidates are considered for this level: **maximum** (M), **average** (A) and **weighted** (W).

The first method (M) selects the most certain class for each document. The goal is to rank higher documents with at least one class correctly classified. This is specially important for collections with a low number of classes per document.

The second method (A) averages the confidence values for the filtered classes, providing a general estimation of how certain the class labels are.

The third method (W) uses an averaged weighted linear combination (WLC), based on the quality estimation per class. Classes with low expected quality are weighted less, because even if their assignation seems certain, it is likely to be a misclassification. The estimated quality values are obtained in the cross-validation phase.

4 Results and Discussion

4.1 Experimental Set-Up

The quality of the certainty algorithms for SATC is evaluated using ER and ENER [1]. ER is plotted with different percentages of the collection being manually classified, while the ENER metric provides values to directly compare the quality achieved by DDF methods with other state of the art approaches. Two traditional TC collections (`Reuters-21578` and `20-newsgroups`) are used:

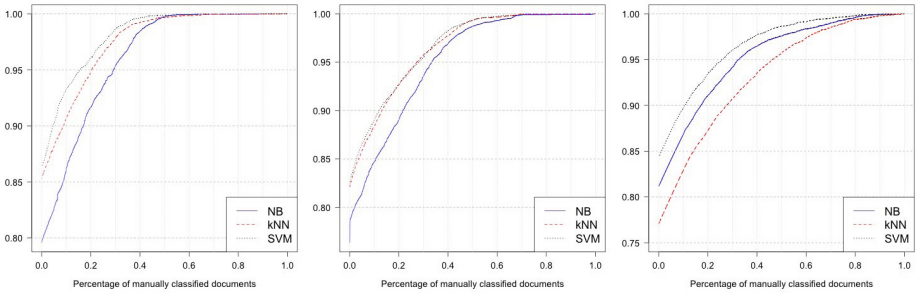
`20-newsgroups` is a collection of approximately $20,000$ newsgroup documents and 20 classes, with almost uniform distribution of documents over classes (Obtained from http://people.csail.mit.edu/jrennie/20Newsgroups/). The split for the collection is based on time as it is suggested. Cross-posting emails have not been considered. This collection has been selected to observe the behaviour of DDF with a single label collection.

`Reuters-21578` contains structured information about newswire articles that can be assigned to several classes. Two variations of the "ModApte" split are used. `Reuters-21578` uses only documents that belong to classes with at least one training and one test document. As a result, there are 7770/3019 documents for training and testing, observing 90 classes with a highly skewed distribution over classes (same as Yang *et al.* [12]). On the other hand, `Reuters-21578-115` uses documents belonging to classes with at least one training or testing document. This configuration has 9603/3299 documents for train and test respectively, and 115 classes. `Reuters-21578-115` allows a direct comparison with the results presented by Berardi *et al.* [1].

Three different families of classifiers have been used, namely Naive Bayes (Weka [5] implementation), SVM using LibSVM [3], and our own implementation of k-NN. Documents are represented using *ltc* [2] and χ^2 is used for feature selection: 5,000 features for NB and k-NN for `20-newsgroups`, and 3,000 for the others. SVM uses 10,000 features (based on full automatic classification experiments). Stemming and stop-words removal have been applied. SCutFBR.1 [12] thresholding strategy is applied with an *fbr* value of 0.3 and a 5-fold cross-validation process, optimising micro-average-F_1. The SVM scores are obtained by running LibSVM with the option (-b 1).

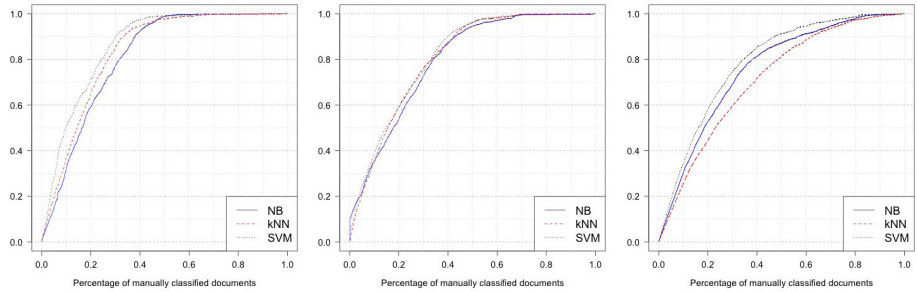
4.2 Results

Figure 1 shows the absolute quality (micro-averaged F_1) depending on the percentage of manually classified documents. Due to clarity, only the best DDF metric per collection is shown. It illustrates how the best DDF metrics achieve high quality levels, while manually assigning a small subset of the collections. For instance, for `Reuters-21578`,



(a) `Reuters-21578` SPW (b) `Reuters-21578-115` RAW (c) `20-newsgroups` SPW

Fig. 1. Micro-averaged F_1 evaluation for different ratios of manually classified documents



(a) `Reuters-21578` SPW (b) `Reuters-21578-115` RAW (c) `20-newsgroups` SPW

Fig. 2. Error Reduction (based on micro-averaged F_1) for different ratios of manually classified documents

micro-average F_1 of more than 95% can be achieved with as few as 20% of the documents manually classified. Furthermore, they also show that perfect quality is achieved with approximately 50 and 60% of documents manually classified for `Reuters-21578` and `Reuters-21578-115` respectively. `20-newsgroups` appears to be a more challenging collection for SATC. Perfect quality is only achieved only after 80% of documents are inspected by experts. The main reason for this seems to be the uniformed distribution of documents over classes and the high similarity between some of the classes. The best performing model in all cases is SVM, while k-NN is the second best algorithm, despite the fact that it performs poorly when applied to `20-newsgroups`.

ER evaluation is shown in Figure 2. Although the best model from this perspective is SVM, it illustrates almost overlapped curves for all different classifiers, specially for `Reuters-21578-115`. This result strongly supports the generalisation of DDF metrics. Tables 2-4 allow to directly compare different SATC methods. They show the ENER quality evaluation for all the DDF candidates, with different percentages of the documents expected to be manually classified (ξ). The best performing models presented in the literature are chosen as baselines: The baselines for `Reuters-21578-115` are the results reported by Berardi *et al.* [1] for their Utility-Theoretic method (UT). The method

Table 2. 20newsgroups ENER evaluation wrt. the ratio of manually classified docs. Best results per model in bold, best overall result also underlined. Increment (%) wrt. RPA between brackets.

	NB			kNN			SVM		
	0.05	0.1	0.2	0.05	0.1	0.2	0.05	0.1	0.2
RPA	.097 (0)	.165 (0)	.230 (0)	.076 (0)	.129 (0)	.185 (0)	.121 (0)	.194 (0)	.251 (0)
SAA	-.014 (-114)	-.020 (-112)	-.029 (-113)	.039 (-48)	.059 (-55)	.074 (-60)	.044 (-64)	.046 (-77)	.037 (-85)
SAM	.090 (-7)	.149 (-9)	.207 (-10)	.072 (-5)	.118 (-8)	.161 (-13)	.119 (-1)	.195 (1)	.265 (6)
SAW	.016 (-84)	.032 (-80)	.061 (-74)	.040 (-46)	.062 (-52)	.081 (-56)	.012 (-90)	.033 (-83)	.069 (-72)
SPA	.097 (-0)	.164 (-1)	.227 (-1)	.075 (-1)	.126 (-2)	.176 (-5)	.120 (-0)	.197 (2)	.267 (6)
SPM	.096 (-1)	.157 (-5)	.214 (-7)	.075 (-1)	.123 (-5)	.166 (-10)	.120 (-1)	.196 (1)	.266 (6)
SPW	.097 (0)	.165 (0)	.230 (-0)	.075 (-0)	.127 (-1)	.180 (-3)	**.121** (1)	**.198** (2)	**.268** (7)
AAA	.095 (-2)	.157 (-5)	.214 (-7)	.062 (-18)	.108 (-17)	.159 (-14)	.119 (-2)	.194 (0)	.260 (4)
AAM	.059 (-39)	.095 (-43)	.137 (-41)	.047 (-38)	.076 (-41)	.111 (-40)	.079 (-35)	.136 (-30)	.200 (-20)
AAW	.094 (-3)	.155 (-6)	.213 (-8)	.057 (-24)	.098 (-24)	.145 (-21)	.117 (-3)	.193 (-1)	.259 (3)
APA	.098 (0)	.166 (1)	.233 (1)	.076 (0)	.129 (0)	.185 (0)	.121 (0)	.197 (2)	.262 (4)
APM	.097 (-1)	.161 (-2)	.222 (-4)	.075 (-0)	.126 (-3)	.173 (-6)	.121 (-0)	.196 (1)	.261 (4)
APW	**.098** (1)	**.167** (1)	**.234** (2)	**.076** (0)	**.130** (1)	**.187** (1)	.121 (0)	.198 (2)	.265 (6)
RAA	.089 (-9)	.151 (-9)	.210 (-9)	.061 (-20)	.107 (-17)	.161 (-13)	.119 (-1)	.195 (1)	.260 (4)
RAM	.061 (-37)	.106 (-36)	.162 (-30)	.044 (-41)	.081 (-37)	.126 (-32)	.092 (-24)	.156 (-20)	.216 (-14)
RAW	.089 (-9)	.150 (-9)	.209 (-9)	.056 (-25)	.099 (-24)	.149 (-19)	.118 (-3)	.193 (-0)	.260 (3)
RPM	.096 (-1)	.159 (-4)	.217 (-6)	.075 (-0)	.126 (-2)	.174 (-6)	.120 (-0)	.193 (-0)	.250 (-0)
RPW	.098 (0)	.166 (1)	.233 (1)	.076 (0)	.129 (0)	.186 (1)	.121 (0)	.195 (1)	.256 (2)

we introduced in [8], modelled as RPA within the DDF framework, is used as baseline for the other two collections. UT quality for `20-newsgroups` and `Reuters-21578` was not reported by Berardi *et al.* [1]. In almost all cases, the performance of DDF is higher when SVM is used, instead of NB or k-NN. For `Reuters-21578-115`, several DDF metrics outperform both baselines (UT and RPA), being SAW (with NB), and RAW (with SVM) the best performers. The improvements are as high as 14 and 50% with respect to UT and RPA, respectively. In addition, RAW with NB also outperforms UT in some cases. All collections confirm the quality of DDF, with several candidates outperforming our previously proposed method (RPA) [8]. For `20-newsgroups`, there is almost not difference between the performance of candidates applying average aggregation and those applying weighted aggregation (e.g., APA vs APW). The main reason for this is that the classes quality are very similar. Furthermore, although no one of the best candidates includes the aggregation based on the maximum confidence (surprisingly being a single-label collection), this strategy achieves high quality (i.e. SPM is virtually as good as the best candidate for SVM).

`Reuters-21578` and `Reuters-21578-115` are analysed together as their main difference is the existence of documents without any correct class for the latter. The first observation, in terms of performance, is that models considering positive classes lose their competitiveness against selecting all classes, for `Reuters-21578-115`. The reason is that this strategy was conceived for collections where test documents have at least one correct class, as documents with no classes are assigned a large uncertainty. This also explains the poor performance of our previous method [8], with decreases of more than 50% ENER, both with respect to the best DDF metric and UT. Furthermore, the qualities achieved by the best model in `Reuters-21578` are significantly higher than those for `Reuters-21578-115` (with the exception of those based on NB). This means

Table 3. Reuters21578 ENER evaluation wrt. the ratio of manually classified docs. Best results per model in bold, best overall result also underlined. Increment (%) wrt. RPA between brackets.

	NB			kNN			SVM		
	0.05	0.1	0.2	0.05	0.1	0.2	0.05	0.1	0.2
RPA	.101 (0)	.178 (0)	.245 (0)	.138 (0)	.234 (0)	.320 (0)	.156 (0)	.250 (0)	.321 (0)
SAA	.027 (-73)	.032 (-82)	.026 (-89)	.093 (-33)	.159 (-32)	.232 (-27)	.050 (-68)	.052 (-79)	.042 (-87)
SAM	.089 (-12)	.162 (-9)	.244 (-0)	.103 (-26)	.176 (-25)	.251 (-22)	.171 (10)	.269 (8)	.349 (9)
SAW	**.140** (38)	**.210** (19)	.283 (16)	.099 (-28)	.168 (-28)	.242 (-24)	.157 (0)	.246 (-1)	.321 (0)
SPA	.103 (2)	.184 (4)	.267 (9)	.132 (-5)	.220 (-6)	.304 (-5)	.193 (24)	**.297** (19)	.376 (17)
SPM	.094 (-8)	.166 (-6)	.248 (1)	.117 (-15)	.187 (-20)	.258 (-19)	.171 (9)	.270 (8)	.351 (9)
SPW	.104 (2)	.185 (4)	.269 (10)	.135 (-2)	.226 (-3)	.311 (-3)	.192 (23)	.297 (19)	**.376** (17)
AAA	.105 (4)	.183 (3)	.265 (8)	.159 (15)	.233 (-0)	.302 (-6)	.175 (12)	.262 (5)	.337 (5)
AAM	.055 (-45)	.093 (-48)	.130 (-47)	.072 (-48)	.132 (-44)	.207 (-35)	.041 (-74)	.074 (-71)	.135 (-58)
AAW	.081 (-20)	.157 (-11)	.242 (-1)	**.169** (22)	.244 (4)	.314 (-2)	.157 (0)	.248 (-1)	.326 (2)
APA	.105 (4)	.192 (8)	.277 (13)	.139 (1)	.235 (1)	.323 (1)	.187 (19)	.293 (17)	.372 (16)
APM	.096 (-5)	.174 (-2)	.257 (5)	.122 (-11)	.195 (-16)	.267 (-16)	.164 (5)	.262 (5)	.342 (7)
APW	.105 (4)	.193 (8)	.280 (14)	.141 (2)	.240 (3)	**.329** (3)	.187 (20)	.294 (18)	.374 (16)
RAA	.115 (14)	.189 (6)	.267 (9)	.138 (-0)	.229 (-2)	.312 (-2)	.171 (9)	.260 (4)	.334 (4)
RAM	.082 (-20)	.135 (-24)	.178 (-27)	.082 (-41)	.156 (-33)	.235 (-27)	.119 (-24)	.192 (-23)	.250 (-22)
RAW	.126 (24)	.205 (16)	**.285** (17)	.163 (18)	**.251** (8)	.329 (3)	**.211** (35)	.295 (18)	.361 (13)
RPM	.093 (-8)	.150 (-16)	.193 (-21)	.124 (-10)	.197 (-16)	.265 (-17)	.136 (-13)	.203 (-19)	.254 (-21)
RPW	.106 (5)	.190 (7)	.267 (9)	.142 (3)	.241 (3)	.329 (3)	.170 (9)	.270 (8)	.344 (7)

Table 4. Reuters21578_115 ENER evaluation wrt. the ratio of manually classified docs. Best results per model in bold, best overall result also underlined. Increment (%) wrt. Utility Theoretic (UT) Ranking between brackets (Berardi *et al* [1]).

	NB			kNN			SVM		
	0.05	0.1	0.2	0.05	0.1	0.2	0.05	0.1	0.2
UT	.145	.221	.285	.145	.221	.285	.145	.221	.285
SAA	.014 (-90)	.017 (-92)	.012 (-96)	.064 (-56)	.116 (-48)	.181 (-36)	.049 (-66)	.052 (-77)	.046 (-84)
SAM	.131 (-9)	.181 (-18)	.240 (-16)	.062 (-57)	.118 (-46)	.190 (-33)	.121 (-17)	.201 (-9)	.279 (-2)
SAW	**.184** (27)	**.241** (9)	**.295** (4)	.071 (-51)	.125 (-43)	.191 (-33)	.128 (-12)	.203 (-8)	.272 (-4)
SPA	.055 (-62)	.125 (-43)	.209 (-27)	.065 (-55)	.138 (-37)	.224 (-21)	.096 (-34)	.190 (-14)	.280 (-2)
SPM	.047 (-68)	.107 (-52)	.187 (-34)	.055 (-62)	.113 (-49)	.187 (-34)	.086 (-41)	.173 (-22)	.261 (-9)
SPW	.055 (-62)	.125 (-43)	.209 (-27)	.066 (-55)	.141 (-36)	.228 (-20)	.094 (-35)	.189 (-14)	.279 (-2)
AAA	.119 (-18)	.168 (-24)	.226 (-21)	.135 (-7)	.198 (-10)	.261 (-8)	.101 (-31)	.177 (-20)	.259 (-9)
AAM	.009 (-94)	.019 (-91)	.030 (-90)	.054 (-63)	.105 (-53)	.169 (-41)	.040 (-72)	.082 (-63)	.151 (-47)
AAW	.028 (-81)	.083 (-63)	.160 (-44)	**.144** (-1)	.208 (-6)	.271 (-5)	.074 (-49)	.156 (-30)	.245 (-14)
APA	.055 (-62)	.127 (-43)	.211 (-26)	.067 (-54)	.148 (-33)	.239 (-16)	.094 (-35)	.191 (-14)	.283 (-1)
APM	.047 (-68)	.107 (-52)	.184 (-35)	.058 (-60)	.120 (-46)	.195 (-32)	.082 (-44)	.168 (-24)	.258 (-9)
APW	.055 (-62)	.127 (-42)	.213 (-25)	.068 (-53)	.149 (-32)	.241 (-15)	.093 (-36)	.190 (-14)	.283 (-1)
RAA	.160 (10)	.209 (-5)	.262 (-8)	.110 (-24)	.188 (-15)	.266 (-7)	.123 (-15)	.197 (-11)	.267 (-6)
RAM	.086 (-41)	.125 (-44)	.158 (-45)	.058 (-60)	.120 (-46)	.193 (-32)	.092 (-37)	.152 (-31)	.211 (-26)
RAW	.171 (18)	.225 (2)	.281 (-1)	.138 (-5)	**.214** (-3)	**.285** (-0)	**.165** (14)	**.238** (8)	**.302** (6)
RPA	.045 (-69)	.101 (-54)	.169 (-41)	.064 (-56)	.142 (-36)	.233 (-18)	.083 (-43)	.170 (-23)	.257 (-10)
RPM	.037 (-75)	.066 (-70)	.097 (-66)	.057 (-61)	.117 (-47)	.191 (-33)	.072 (-51)	.139 (-37)	.205 (-28)
RPW	.054 (-63)	.121 (-45)	.198 (-30)	.066 (-54)	.147 (-33)	.240 (-16)	.085 (-41)	.178 (-19)	.269 (-6)

Table 5. Average ENER evaluation for DDF patterns and xi=0.1

Collection	Model	S**	A**	R**	*A*	*P*	**A	**M	**W
20newsgroups	NB	.108	**.150**	.149	.108	**.163**	.130	.138	**.139**
	kNN	.102	.111	**.112**	.090	**.127**	**.110**	.108	.108
	SVM	.144	.186	**.188**	.149	**.196**	.170	**.179**	.168
Reuters21578_115	NB	.133	.105	**.141**	**.141**	.112	.125	.101	**.154**
	kNN	.125	.155	**.155**	**.155**	.135	.155	.116	**.164**
	SVM	.168	.161	**.179**	.162	**.176**	.163	.153	**.192**
Reuters21578	NB	.157	.165	**.174**	.152	**.179**	.160	.147	**.190**
	kNN	.189	.213	**.218**	.194	**.219**	.218	.174	**.228**
	SVM	.238	.239	**.245**	.211	**.271**	.236	.211	**.275**

that the addition of documents without correct classes makes the SATC problem more complex to solve, or at least that DDF metrics are less suited for this type of datasets.

Results also show that SAA (and SAW for `20-newsgroups` because of the similar qualities per class) is only suited for classifiers that do not normalise the scores per document. SAA performs as a random ranker for this type of classifiers which include the versions of NB and SVM presented on this paper. If the classification scores are normalised per document ($\sum_{c_i \in C} s(d, c_i) = 1$), SAA produces the same difficulty, independently of the document. Other very poor metric is AAM, because the highest confidence based on difference is usually based on a very low (or even zero) score. Therefore, the certainty computation will be uniquely based on this information.

Table 5 summarises the average quality for candidates sharing two strategies, with $\xi = 0.1$ (arbitrarily chosen). For instance, S** averages the error reduction for every variation using positive labels (this is, it encapsulates information about SAA, SAM, SAW, SPA, SPM, and SPW). This analysis provides information about which strategies are better for each level in different conditions, and it helps to understand some of the previously reported results from a general perspective. For the evidence level, the best strategy is the relative difference, independently of classifier and collection. This result confirms our previous assumptions made for the RPA method [8]. The class level illustrates that the selection of positive classes achieves good quality, as long as the assumption that all the documents have at least one correct class is correct. Otherwise, all classes should be considered. The aggregation level shows that the exploitation of quality estimation outperforms the other strategies for `Reuters-21578` and `Reuters-21578-115`. All strategies perform similarly for `20-newsgroups`.

5 Conclusions and Future Work

SATC represents a largely unexplored task within TC which is critical in environments where high quality classification is needed, but resources are limited. Its main goal is to achieve high quality with minimum human effort, minimising the potential cost. This research introduces DDF, a document certainty framework based on classification scores and class thresholds, and its application to SATC.

DDF generalises several methods by abstracting three different levels, specifying how to manipulate the scores and thresholds to obtain a document certainty measure.

Results show that DDF metrics achieve virtually perfect classification with as low as 50% of documents being classified. SVM is the best classifier for DDF and RAW is its best overall variation, with the exception of `Reuters-21578-115`, where NB with the SAW strategy is the best alternative. DDF outperforms all the previously proposed methods in the literature for SATC. The strategy analysis shows that the best models should include a relative difference of scores, and the exploitation of estimated class quality. In addition, observing only the positive classes for a document achieves better quality, but only if all documents belong to at least one class.

Future work will provide deeper analysis of the combination between difference strategies, as well as their behaviour for different ratios of the collection. The combination of DDF and the Utility-Theoretic Ranking method, and the combination of DDF values as features for a meta-ranker are also interesting lines of research.

References

1. Berardi, G., Esuli, A., Sebastiani, F.: A utility-theoretic ranking method for semi-automated text classification. In: SIGIR (2012)
2. Buckley, C., Salton, G., Allan, J.: The effect of adding relevance information in a relevance feedback environment. In: SIGIR (1999)
3. Chang, C.-C., Lin, C.-J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology (2011)
4. Esuli, A., Sebastiani, F.: Active learning strategies for multi-label text classification. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 102–113. Springer, Heidelberg (2009)
5. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explor. Newsl (2009)
6. Larkey, L.S., Croft, W.B.: Combining classifiers in text categorization. In: SIGIR (1996)
7. Lewis, D.D., Gale, W.A.: A sequential algorithm for training text classifiers. In: SIGIR (1994)
8. Martinez-Alvarez, M., Yahyaei, S., Roelleke, T.: Semi-automatic document classification: Exploiting document difficulty. In: Baeza-Yates, R., de Vries, A.P., Zaragoza, H., Cambazoglu, B.B., Murdock, V., Lempel, R., Silvestri, F. (eds.) ECIR 2012. LNCS, vol. 7224, pp. 468–471. Springer, Heidelberg (2012)
9. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. (2002)
10. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. J. Mach. Learn. Res. (2002)
11. Yang, B., Sun, J.-T., Wang, T., Chen, Z.: Effective multi-label active learning for text classification. In: SIGKDD (2009)
12. Yang, Y.: A study on thresholding strategies for text categorization. In: SIGIR (2001)
13. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: SIGIR (1999)

Knowledge-Free Table Summarization

Dino Ienco[1,2], Yoann Pitarch[3], Pascal Poncelet[1,2], and Maguelonne Teisseire[1,2]

[1] IRSTEA, UMR TETIS, Montpellier, France
{dino.ienco,maguelonne.teisseire}@irstea.fr
[2] LIRMM CNRS Montpellier, France
pascal.poncelet@lirmm.fr
[3] LIRIS CNRS Lyon , France
yoann.pitarch@liris.cnrs.fr

Abstract. Considering relational tables as the object of analysis, methods to summarize them can help the analyst to have a starting point to explore the data. Typically, table summarization aims at producing an informative data summary through the use of metadata supplied by attribute taxonomies. Nevertheless, such a hierarchical knowledge is not always available or may even be inadequate when existing. To overcome these limitations, we propose a new framework, named **cTabSum**, to automatically generate attribute value taxonomies and directly perform table summarization based on its own content. Our innovative approach considers a relational table as input and proceeds in a two-step way. First, a taxonomy for each attribute is extracted. Second, a new table summarization algorithm exploits the automatic generated taxonomies. An information theory measure is used to guide the summarization process. Associated with the new algorithm we also develop a prototype. Interestingly, our prototype incorporates some additional features to help the user familiarizing with the data: (i) the resulting summarized table produced by **cTabSum** can be used as recommended starting point to browse the data; (ii) some very easy-to-understand charts allow to visualize how taxonomies have been so built; (iii) finally, standard OLAP operators, i.e. drill-down and roll-up, have been implemented to easily navigate within the data set. In addition we also supply an objective evaluation of our table summarization strategy over real data.

1 Introduction

Nowadays, modern technologies allow to collect huge amount of raw but potentially very knowledgable data which are available at a very low level of granularity, e.g, sensor data, web logs. When facing this very detailed information the domain expert might experience big trouble determining from where he/she can start exploring these data. Actually, this data exploration is typically the first step in any analysis process. Nevertheless, when the available quantity of data is too abundant or the user is not so expert in the domain, this first step is often problematic and automatic approaches can help to obtain some insights about the way to effectively explore the data [4]. For instance, Explorative Data

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 122–133, 2013.

Mining (EDM) tools help the user in preliminary analysis giving him/her the possibility to have a first meaningful view on the data. In this direction, it should be noted that two orthogonal research problems might coexist. Assuming a relational table setting, i.e., lines correspond to tuples whereas rows correspond to attributes/dimensions, the former problem is related to deal with the curse of dimensionality by the means of dimensionality reduction or aggregation [16], whereas the latter problem deals with reducing the number of tuples while providing the user with still a very representative view of his/her data. In this paper, we address the latter, i.e., the *tuple reduction* problem.

One way to manually deal with this *tuple reduction* problem is using data warehouses coupled with OLAP engines [1]. These interactive tools allow the analyst to aggregate and navigate through data by the mean of the drill-down and roll-up operators that exploit, when available, the attribute hierarchies/taxonomies[1]. OLAP engines demand human supervision while an automatic way for addressing this problem is supplied by the table summarization process (TSP) [12].

The table summarization process (TSP) provides an aggregated representation of a relation table and thus helps the user familiarizing with the data. Typically, TSP is particularly well adapted to the rapid development of mobile devices, e.g., smart phones or tablets, since it enables visualizing data respecting the screen size constraints imposed by the new devices generation [3]. To cope with this issue, TSP can be exploited from smartphone devices to supply synopsis (or summary) of relational data. In this work we focus our research on the second issue and thus propose a new innovative table summarization algorithm.

Our work is essentially motivated by the following observation: data often lacks of associated metadata information, e.g., attribute taxonomies. This lack drastically limits the panel of possible approaches (OLAP, TSP) to use to reduce the number of tuples. Indeed, both OLAP and existing TSP techniques assume attribute hierarchies to be available. This assumption is clearly a strong constraint because most of the times such a hierarchical knowledge is not available or may even be inadequate when existing. On the other hand, using inappropriate hierarchy can provide useless results since they can be too general and bias the OLAP and TSP results. Thus, proposing techniques that are able to meaningfully summarize huge quantities of data without leveraging any other knowledge than the one provided by the data might be very useful for analysis purpose.

Contrary to the previous techniques, our framework, called **cTabSum**, relaxes this constraint producing table summaries without needing any additional information. The main advantage of **cTabSum** lies in its ability to automatically generate contextual Attribute Values Taxonomies (cAVTs) and to use them to automatically summarize the data. This original feature also allows our approach to be used over relation tables where metadata information are too general or do not fit the domain of investigation. An example of application can be represented by the visualization, over mobile device, of query results. **cTabSum**

[1] In this paper we will use these two terms indistinguishly.

can automatically aggregate query results exploiting data dependency. After the performed aggregation, the summarized table can be easily visualized on a smartphone screen. The final result of **cTabSum** is a summarized table that can be considered as a recommended aggregation of the data. It can also be considered as an entry point from which the user can start browsing the data using drill-down and roll-up operators. The main contributions of our work are the following:

- A general table summarization approach that can be applied over data table where no metadata are available or metadata are too general for the considered domain;
- The final result can be exploited as a starting point to browse and navigate through the data;
- A prototype that integrates the **cTabSum** algorithm in an interactive and easy-to-use environment. Drill-Down and Roll-up operators are implemented to navigate data starting from the recommended summarization.

The rest of the paper is organized as follows. Section 2 presents the related work and positions our contributions w.r.t. the state of the art. In Section 3 we describe our proposal to produce TSP without any kind of metadata information. An experimental evaluation over real data is carried out in Section 4. In Section 5 we present the prototype built over the **cTabSum** algorithm and how the user can interact with the resulting summarized table. We also supply an illustrative example with the purpose to realize a qualitative evaluation of our approach. Section 6 concludes and describe some possible future works.

2 Related Work

A first approach proposed in the context of table summarization was presented in [12]. This approach creates and maintains table summaries through row and column reductions. To reduce the number of rows, the algorithm first partitions the original table into groups based on one or more attribute values of the table, and then collapses each group of rows into a single row relying on the available metadata, such as the concept hierarchy. In [14] the SaintEtiQ is proposed. This system computes and incrementally maintains a hierarchically arranged set of summaries of the input table. SaintEtiQ uses background knowledge (i.e., metadata) to support these summaries. [2] proposed to work at the metadata level to improve the final summarization and speed-up the whole process. The strategy is based on a pre-processing of the original concept taxonomy in order to obtain a more compact hierarchy. Once the reduced metadata is available it is employed to perform the table summarization task. As we can observe, all the previous algorithms exploit some metadata knowledge that compulsoliry need to be supplied by the user. On the contrary, our approach is able to perform and complete the process of table summarization without asking any additional information to the expert. This characteristic allows **cTabSum** being much more flexible and usable when no background knowledge is available.

3 cTabSum

In this section we first introduce preliminary notations and then describe in details the **cTabSum** framework. We define a table T as a set of tuples, i.e., $T = \{t_1, t_2, ..., t_n\}$. Each t_i is described over a set of attributes $A_1, \ldots, A_m$, i.e., $t_i \in (dom(A_1) \times dom(A_2) \times ... \times dom(A_m))$. Given $1 \leq i \leq n$ and $1 \leq j \leq m$, T_{ij} denotes the value of the attribute A_j in the i^{th} tuple of T. Given an attribute A_j, V is a partition over the values of that attribute such that $\forall_{v \in V}\ v \subset A_j$ and $\forall_{v,w \in V}\ v \cap w = \emptyset$. The partition V induces a generalization of the attribute A_j. More precisely, given a value $a_{jk} \in dom(A_j)$ (a_{jk} is the k-th value of the attribute A_j), it exists a value v s.t. $a_{jk} \in v$. For instance, given an attribute taxonomy over A_j, the ancestor of an attribute values can be seen as a generalization of it. The objective of the TSP is to find a summarized table $s(T) = \{g(t_1), g(t_2), ..., g(t_l)\}$ s.t. each $g(t_b)$ (generalized tuple) cover at least k tuples of the original table T and there are no overlap among the tuple set covered by generalized tuples.

3.1 Proposed Approach

Essentially the proposed strategy proceeds in two main steps. The first step consists in building the cAVTs (one for each attribute A_j over which T is defined). Leveraging data mining techniques, we can extract this meta-information starting from the data [9]. The second step performs the table summarization strategy. The algorithm uses the previously calculated cAVTs to generalize the data. To do so, a new approach that exploits a criterion coming from information theory to evaluate the quality of the possible solutions in the search space is proposed. Similarly to [2], **cTabSum** also requires a parameter k but it avoids any metadata information (taxonomies) as input.

The TSP extracts a summarized table with the constraint that each generalized tuple must cover at least k tuples of the original table, where k is a user-defined parameter. The table summarization task can be related to the k-anonymity problem[15]. While the algorithmic approaches could be similar, the final results are different in spirit. TSP supplies an informative compression of data to allow exploratory analysis in contrast to the k-anonymity purpose which aims at hiding some sensitive information. Another difference is that table summarization is performed for all the attribute of the data table while k-anonymization involves only the set of attribute composed by quasi-identifiers (sensible attributes).

3.2 Table-Summarization Quality Criterion

The TSP task can be seen as a search problem in which we want to find, if possible, the best compression of the original table taking into account the constraint k. This best compression also implies to have the minimum information loss w.r.t the original data. Thus, when facing with a set of possible intermediate solutions, it is needed to understand which one should be chosen. In the field of data compression, information theory measures are normally used to evaluate,

given a model, the gain in compression (and information loss) w.r.t. the uncompressed data [13]. As our final goal is to obtain a summarization of the original table, these kind of measures can be very suitable. Particularly, to evaluate how much information is lost during the summarization process, we employ the non-uniform entropy [7]. The non-entropy measure, that we call $InfoLoss(T, s(T))$, is defined as follows:

$$InfoLoss(T, s(T)) = \sum_{i=1}^{n} \sum_{j=1}^{m} -\log Pr(T_{ij} = a_{jk} | s(T)_{ij} = v) \quad (1)$$

where

$$Pr(T_{ij} = a_{jk} | s(T)_{ij} = v) = \frac{\#\{1 \leq i \leq n : T_{ij} = a_{jk}\}}{\#\{1 \leq i \leq n : T_{ij} \in v\}}$$

The conditional probability $Pr(T_{ij} = a_{jk} | s(T)_{ij} = v)$ indicates how the subset of original values of A_j are distributed w.r.t. their generalization induced by partition v. The non uniform entropy measure (w.r.t. the classical entropy measure) helps to better considering the distribution of the values belonging to the same generalization.

3.3 Building CAVTs

An important step of our strategy is the automatic construction of the cAVTs. To structure attribute values in taxonomies we exploit the approach proposed in [8]. This technique allows to extract distances between each pair of values of the same categorical attribute. Most in detail, given a categorical attribute A_j, over which the database D is defined, it selects a set of other attribute strictly related to it using correlation measure. This set of attribute is called context. After that, it uses the distribution of the values of attribute A_j w.r.t. the attributes in the context to infer a distance matrix that represents the distances between each pair of values $v_{jk}, v_{jl} \in A_j$, where v_{jk} is the k-th value of attribute A_j. Once the point-wise distance matrix for A_j is available, it is used as input for standard agglomerative hierarchical clustering[11] that produces the final cAVTs[9]. This approach is very useful when the original hierarchies are not available or to compare the induced taxonomies with the original ones understanding which one fit the actual analysis. As the TSP process requires hierarchies to summarize the original data, the cAVTs always supply this metadata knowledge. In particular, given an attribute, any cuts of the corresponding cAVT results in a partition of the domain of that attribute. Each of these partitions represents a possible generalization that can be used in the summarization process.

3.4 Constructing the Summary

At this point we are able to extract attribute taxonomies directly from data (cAVTs) and evaluate how far is a summary w.r.t. the original data (InfoLoss). Now we need to design a strategy to navigate the search space defined over all

the possible solutions. In this direction we design an algorithm for TSP following the general idea presented in [5]. There are three big differences between the proposed method and the one described in [5]:

1 **cTabSum** supplies a summary of the original data for exploratory purpose while in [5] the final goal is the anonymization of the data following the k-anonymity philosophy [15].
2 In [5] the goal is to preserve classification accuracy and for this reason the class variable is used to guide the privacy preservation process. **cTabSum** is designed for a totally unsupervised scenario where no class information is available and the final goal is to compress the original data. To implement this important difference, instead of the Information Gain used in the privacy preserving strategy we minimize the Information Loss criteria based on Non Uniform Entropy.
3 **cTabSum** does not require any metadata because it directly mines this knowledge from the data relaxing the strong assumption on the availability of such kind of information.

The general process is presented in Algorithm 1. It takes as input a value k and returns a set of generalized tuples that describes the original table. The parameter k indicates the minimum number of original tuples that need to be covered by each generalized tuple in the final result. The search strategy is performed by a top-down navigation over the space of possible solutions. The top-down search starts from the top of all the attribute value taxonomies (the coarsest granularity levels) and, at each step, tries to specialize and expand only one attribute taxonomy with the function $bestExpansion(T, H, x)$. This function chooses to expand the cAVT that produces the minimum loss of information evaluated by the Formula 1. Most in detail, for each attribute A_j the following steps are performed: (i) generate all the possible expansions starting from the actual frontier of the cAVT, (ii) choose the best expansion that minimize the measure in Formula 1.

As the process is performed over all the attributes simultaneosuly, $bestExpansion(T, H, x)$ can expand any of the taxonomies at any node. This means that in the final summarized table, for the same attribute, we can have a generalization (partition) at different levels of granularity. Once that the actual expansion y is chosen we evaluate both: the Information Loss w.r.t. the actual best solution and the constraint supplied by k. If both conditions are satisfied, y is marked as new best solution. At the end the algorithm returns the solution that minimize the Information Loss and, at the same time, the constraint k.

cTabSum is implemented in a tool that has the same name and it is public available at the url address: `http://www.lirmm.fr/~ienco/Dino_Ienco_Home_Page/cTabSum.html`.

4 Experiments

In this section we draw some experiments to analyze the performance of our approach using real world data table. The approach has been implemented in Java

Algorithm 1. $cTabSum(T, k)$

```
/* H is the set of cAVTs induced from T                                    */
H = extractCAVTs(T);
actualFrontiers = ∅;
Initialize x to the Summarization using the top value of each taxonomies in H;
actualFrontiers = actualFrontiers ∪ x;
InfLoss = Score(x);
result = x;
forall the x ∈ actualCuts do
    y = bestExpansion(T, H, x);
    actualFrontiers = actualFrontiers \ x;
    actualInfLoss = Score(y);
    if (actualInfLoss ≤ InfLoss AND y is valid w.r.t. k) then
        InfLoss = actualInfLoss;
        actualFrontiers = actualFrontiers ∪ y;
        result = y;
    end
end
return result;
```

and we carried out experiments on Mac OS X 10.6.8 with 4Gb of RAM and an i7 2.2GHz processor. Actually this task shares similarities to the k-anonymization problem, in which each tuple of the anonymized table needs to represent at least k tuples of the original table. For this reason we used the algorithm MinGen presented in [15] as competitor to evaluate the performance of **cTabSum**. This algorithm was originally developed for the k-anonymity problem. We couple MinGen with the original available taxonomies. The algorithm needs two parameters. The first parameter k is the minimum number of tuples indistinguishable over the set of sensitive attributes (quasi-identifiers) [15]. The quasi-identifiers are the attributes for which the generalization process is performed (and the attribute taxonomies is demanded). The second parameter is the suppression coefficient *sc* standing for the maximum number of tuples (in percentage w.r.t. the size of the table) that the algorithm suppresses to obtain a k-anonymized table. This parameter is not relevant for the table summarization task and it has thus been set to 0. To evaluate the performance we use the *Adult* dataset[2]. It is based on census data and has been widely used to evaluate classification and k-anonymization algorithms. Here, the same settings as in [10] are used. We obtain a dataset of 8 attributes. The attribute *age* is discretized in 8 equal-size bins. Starting from this dataset we sample 10% of the original dataset obtaining 4 522 tuples. As original hierarchies we use the taxonomies supplied in [10]. In our experiment we range the k parameter from 40 to 400 at step of 40. To evaluate the quality of the compressed table we adopt the non-uniform entropy measure (Formula 1). The idea is that a good k-anonymization (for this reason also a good summary of an original table) is the one that minimizes the Information Loss respecting the given constraints. As **cTabSum** generates attribute taxonomies directly from the data, a satisfactory result could be to obtain comparable performances compared to the ones involving the original hierarchies.

[2] `http://archive.ics.uci.edu/ml/`

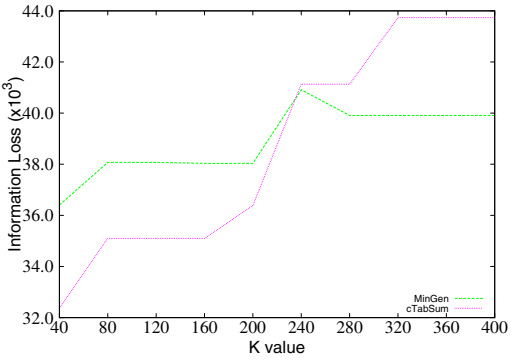


Fig. 1. Information Loss of **cTabSum** and *MinGen* over the *Adult* dataset

In Figure 1 we observe the behaviors of the different approaches. Generally, we can state that the two approaches have comparable performances. This is a good point for **cTabSum** that automatically generates attribute taxonomies from data. In particular for values of k lesser than 240 **cTabSum** obtains better results in terms of Information Loss. This value of 240 influences seriously the final summarization because for values above this threshold both algorithms return summaries of only 4 generalized tuples to represent the original data table. From an user point of view, this result can be useless due to the fact that it compresses to much the original information. With values lesser than 240, both approaches return more reasonable summarization composed at least of 8 tuples. These results can represent more reasonable starting points for an initial explorative analysis of the data.

4.1 Assessing Summarization via Swap Randomization

In this subsection we evaluate how far the results of **cTabSum** are from randomness. A table summarization could be seen as a partition of the original tuples in groups that share common characteristic w.r.t. the attribute taxonomies. In this way TSP produces a clustering of the tuples constrained by metadata. Unfortunately we do not have any background knowledge to evaluate the quality of this partition. For this reason following the idea proposed in [6], we compare our results w.r.t randomized partition. Practically, given a table summarization result, each generalized tuple represents a cluster of the original ones. Starting from this clustering result, we swap at random, tuples between clusters obtaining random partition. In the random partitions the marginal distribution given by the original clustering are maintained. To evaluate the clustering solution we employ the ZIP function always available over any OS. In particular we zipped each cluster separately and we sum the size of all of them for a clustering solution in order to obtain a value for each value of k. The value is expressed in Kb. Low values indicate high compression rate that means a kind of structure inside the partitions. We employ this kind of measure because because it naturally captures how good is

the compression of the original table. The ZIP procedure also constitutes an easy way to evaluate this quantity. Figure 2 shows the results. For each value of k we generate 100 random clustering following the distribution induced by the result of **cTabSum** and we report the average value. We avoid to visualize the standard deviation because it is always smaller than 1kb. We can observe that the results produced by **cTabSum** always give a lower size of the final ZIP files, this underline how the clustering induced by **cTabSum** contains a structure that can be easily recognize by a compression algorithm as the one implemented by the ZIP function.

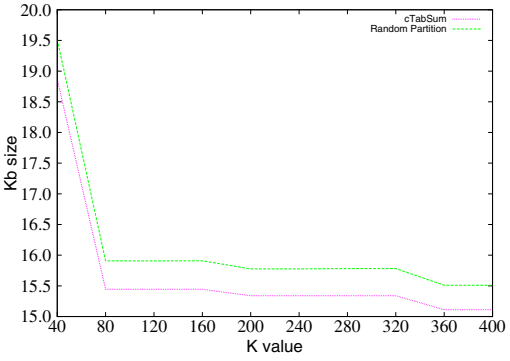


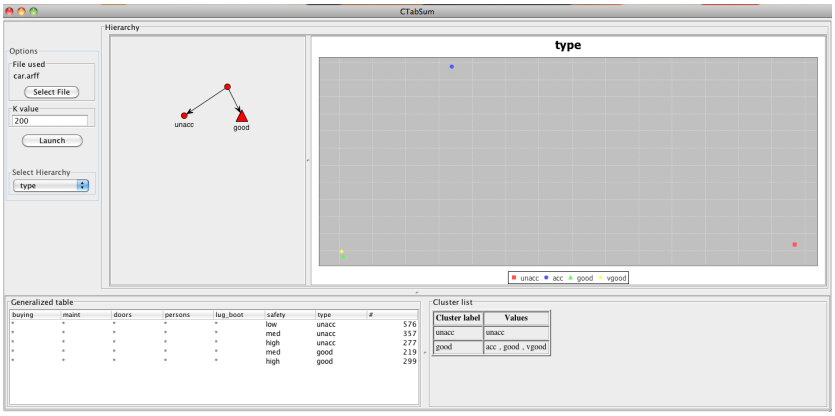
Fig. 2. Size in Kb of zip files comparing the group obtained by **cTabSum** and the randomization result

5 The cTabSum System

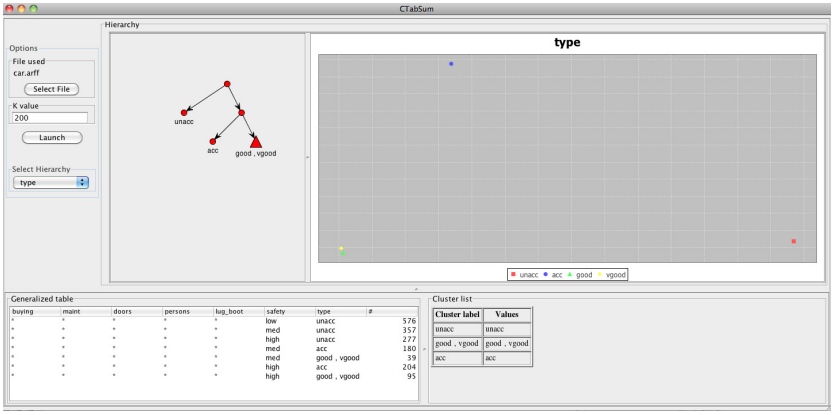
As we have obtained interesting results from an experimental point of view, our algorithm has been integrated in a prototype named **cTabSum**. **cTabSum** is a standalone software written in Java. It is based on several open source projects. First of all, it uses the *Weka Data Mining Library* to manage the data[3]. More precisely, our system requires a file in ARFF format (Attribute Relational File Format). In the GUI we integrate the *JFreeChart* Library[4] to produce and visualize 2D chart and the *JUNG* (Java Universal Network/Graph) library[5] to visualize attribute taxonomies.

Figures 3(a) and 3(b) show a screenshot of **cTabSum**. It illustrates an example of result obtained by a TSP. It allows to choose a source file (in ARFF format) and to set the value of the parameter k. In this example the *Car* dataset was used with a parameter value, k, equals to 200. In this dataset each instance is described over a set of discrete features (e.g., *buying*, *safety*, *type*, etc..). Once the TSP is launched, the final table summarization result is shown in the bottom part of the GUI. The result of the summarization process is displayed in

[3] `http://www.cs.waikato.ac.nz/ml/weka/`
[4] `http://www.jfree.org/jfreechart/`
[5] `http://jung.sourceforge.net/`



(a)



(b)

Fig. 3. The **cTabSum** system

table that contains all the original attributes plus an extra attribute, *Count*, to count how many original tuples are covered by each generalized tuple. The $*$ symbol indicates that the root level of the corresponding attribute taxonomy is used, i.e., all the values are grouped together. In the proposed example, the algorithm has found a solution that involves the specialization of only two attributes: *safety* and *type*. Starting from this result the user can interact with the data. The combo box in the top left part of the window allows to choose and investigate a particular attribute of the dataset. In the middle of the window, the cAVT can be visualized. In the tree visualization panel, two different types of nodes coexist: triangle nodes represent group of two or more original values while circle nodes represent group of only one attribute value.

In this example, we observe that the taxonomy associated to the attribute *Type* is cut in a way to produce two groups of attributes: the left node with only one value, *unacc* (unacceptable), and the right node that contains a group of

three values, *acc*, *good* and *vgood*. The actual taxonomy cut can be also visualized in the right bottom part of the interface in which each cluster is described with its representative plus the enumeration of the attribute values that it contains (clustering list). The user starts from this taxonomy cut to browse the hierarchy using drill-own and roll-up operators. These operations modify the displayed attribute taxonomy. The changes in the attribute taxonomy are propagated to all the other components in the GUI, i.e., *Generalized table* and *Cluster list*. In Figure3(b) the application displays the results after an user action over the attribute taxonomy. We can observe that the user has expanded the taxonomy *Type* thanks to a drill-down operation. It should be noted that this expansion impacts on both the panels *Generalized table* and *Cluster list* that now displays the current view of exploration. This means that the *Generalized table* panel visualizes the generalized tuples correspondingly to the actual cut of the different attribute taxonomies while the *Cluster list* shows the partition of leaves of the selected attribute with respect to the considered level of granularity.

Another feature implemented in **cTabSum** is shown in the right upper side. For each attribute, a 2D chart is plotted where the attribute values are positioned in a way that preserves the relationships existing in the original point-wise distance matrix. When a user selects a new attribute from the combo box, both the chart and the taxonomy are automatically updated to display information related to the newly selected attribute. As an additional feature supplied by our prototype, when there some attribute in the dataset are numerical, **cTabSum** exploits the pre-processing facility supplied by Weka and performs a discretization step deriving 10 equal-width bins for each numerical attribute. In this way our system is able to manage tables containing mixed attributes types. The **cTabSum** implementation is available at the URL: `http://www.lirmm.fr/~ienco/Dino_Ienco_Home_Page/cTabSum.html` .

6 Conclusion

The process of Table Summarization helps the analyst to have a first picture of the data. In our work a new algorithm to perform table summarization **cTabSum** is proposed. One of the main features of **cTabSum** consists in relaxing the strong assumption on the availability of attribute taxonomies. This is realized mining directly the attribute taxonomies from the data. Associated with the new algorithm we also developed a java prototype designed to fully exploit the knowledge available in the data. It uses **cTabSum** to aggregate the original data to supply (i) a more compact representation of them (ii) a starting point from which the user can navigate the table. More precisely, starting from the result of our system, the user can browse, interact and modify the data summary navigating through the attribute taxonomies performing both drill-down and roll-up operations. In the future we plan to integrate **cTabSum** in open source OLAP project (like Mondrian) in order to supply a sort of recommendation to the final user to start his/her analysis. From an algorithmic point of view we want eliminate the use of parameter k replacing it with a lower bound concerning

the minimum number of generalized tuples the final result may contain. Finally, the goal of our study is to summarize simple relational table, in which static information appears. In a next step, summarizing more complex data, e.g., data with temporal dimension, could be considered. In such a case, the generation of cAVTs will be different and much complicated and require further investigations.

References

1. Berson, A., Smith, S.J.: Data Warehousing, Data Mining, and Olap, 1st edn. McGraw-Hill, Inc., New York (1997)
2. Candan, K.S., Cataldi, M., Luisa Sapino, M.: Reducing metadata complexity for faster table summarization. In: EDBT, pp. 240–251 (2010)
3. Chittaro, L.: Visualizing information on mobile devices. IEEE Comp. 39(3), 40–45 (2006)
4. Dasu, T., Johnson, T.: Exploratory Data Mining and Data Cleaning. John Wiley & Sons, Inc. (2003)
5. Fung, B.B.C.M., Wang, K., Yu, P.S.: Top-down specialization for information and privacy preservation. In: ICDE, pp. 205–216 (2005)
6. Gionis, A., Mannila, H., Mielikäinen, T., Tsaparas, P.: Assessing data mining results via swap randomization. TKDD 1(3) (2007)
7. Gionis, A., Tassa, T.: k-anonymization with minimal loss of information. IEEE Trans. Knowl. Data Eng. 21(2), 206–219 (2009)
8. Ienco, D., Pensa, R.G., Meo, R.: From context to distance: Learning dissimilarity for categorical data clustering. TKDD 6(1), 1–27 (2012)
9. Ienco, D., Pitarch, Y., Poncelet, P., Teisseire, M.: Towards an automatic construction of contextual attribute-value taxonomies. In: SAC, pp. 432–437 (2012)
10. Iyengar, V.S.: Transforming data to satisfy privacy constraints. In: KDD, pp. 279–288 (2002)
11. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Comput. Surv. 31(3), 264–323 (1999)
12. Lo, M.-L., Wu, K.-L., Yu, P.S.: Tabsum: A flexible and dynamic table summarization approach. In: ICDCS, pp. 628–635 (2000)
13. MacKay, D.J.C.: Information Theory, Inference & Learning Algorithms. Cambridge University Press, New York (2002)
14. Saint-Paul, R., Raschia, G., Mouaddib, N.: General purpose database summarization. In: VLDB, pp. 733–744 (2005)
15. Samarati, P.: Protecting respondents' identities in microdata release. IEEE Trans. Knowl. Data Eng. 13(6), 1010–1027 (2001)
16. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Data Management Systems. Morgan Kaufmann (2005)

Predicting Your Next OLAP Query Based on Recent Analytical Sessions

Marie-Aude Aufaure[1], Nicolas Kuchmann-Beauger[1], Patrick Marcel[2], Stefano Rizzi[3], and Yves Vanrompay[1]

[1] MAS Laboratory, École Centrale Paris, France
[2] Université Francois Rabelais de Tours, France
[3] DISI, University of Bologna, Italy

Abstract. In Business Intelligence systems, users interact with data warehouses by formulating OLAP queries aimed at exploring multidimensional data cubes. Being able to predict the most likely next queries would provide a way to recommend interesting queries to users on the one hand, and could improve the efficiency of OLAP sessions on the other. In particular, query recommendation would proactively guide users in data exploration and improve the quality of their interactive experience. In this paper, we propose a framework to predict the most likely next query and recommend this to the user. Our framework relies on a probabilistic user behavior model built by analyzing previous OLAP sessions and exploiting a query similarity metric. To gain insight in the recommendation precision and on what parameters it depends, we evaluate our approach using different quality assessments.

Keywords: OLAP, Query recommendation, User modeling.

1 Introduction

Online Analytical Processing (OLAP) systems allow users to explore and analyze large volumes of data by formulating queries on multidimensional cubes. Despite the flexibility, usability, and efficiency of modern OLAP systems, the huge number of possible aggregations and selections that can be operated on multidimensional data may make the user experience disorientating and frustrating, so that users may need a long time to achieve their analysis goals.

The approach we propose in this paper to tackle this problem is based on a prediction of the most likely queries the user will submit next. We start by observing that OLAP workloads tend to show different patterns depending on the specific analytical tasks the user is performing. Our goal is to learn these patterns and represent them in a probabilistic model of the user's behavior; to this end, we analyze the query logs of a user, we cluster queries using a similarity metric, and we derive a Markov-based model of the user behavior. Using this model, we are able to predict the most likely queries the user will formulate next given the current query. Our approach takes a step towards improving the quality of the user experience with OLAP systems in different ways. First, given

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 134–145, 2013.

the current query in the OLAP session, a set of most probable next queries can be proactively recommended to the user to guide her in analyzing the data and prevent her from "getting lost" among multidimensional data. Secondly, query recommendation could allow the average length of OLAP sessions to be reduced because users are driven towards their analysis goal and can reach them in less steps. Indeed, if a user is given a choice between different future queries, she might be able to skip some steps and more quickly arrive at the expected results. Thirdly, the OLAP cache manager can exploit the predicted queries for estimating the benefits of a cached object for future queries. Using predictions, the cache manager can also prefetch objects that are likely to be of interest for future queries, which results in a reduction in latency time perceived by the user.

The rest of the paper is organized as follows. Section 2 gives an overview of related work in OLAP query prediction and recommendation. Section 3 describes formally the query model we incorporate in our approach. Then, Section 4 presents the different steps for recommendation, i.e., query clustering and query prediction based on a user behavior model, together with the query similarity metric we adopt. In Section 5 we experimentally evaluate our approach, while in Section 6 we conclude and give directions for future research.

2 Related Work

Recommending items of interest and queries has been intensively researched in the domains of Information Retrieval [1] and search engines [2]. Recently, in the database community, there has been an increasing interest in leveraging past queries or query answers to assist interactive relational database exploration [3–9]. The approaches proposed include past query browsing and/or searching [7], query completion [6] and query recommendation [5, 8, 9]. Noticeably, automatic query recommendation approaches either rely on the query answer and database instance, which may lead to efficiency problem, or treat sessions as sets of queries, overlooking the intrinsic sequential nature of the exploratory process. A framework for recommending OLAP queries has been presented in [10], whose authors group queries according to a finer similarity measure and then recommend queries by matching logged sessions to the current session. To the best of our knowledge, PROMISE is the first system applying predictive caching to multidimensional queries aimed at reducing the execution time of OLAP queries within a session [11]. PROMISE integrates a Markov model [12] where each state corresponds to a discrete point of the timescale (i.e., only one state is active at time t); a transition between two states corresponds to the probability to reach the next state given the previously visited states. While [10] bases its recommendations on a similarity metric and does not use a probabilistic model, PROMISE probabilistically represents next queries, mainly for query prefetching and not for recommendation. Besides, PROMISE use a coarse approach to group queries (grouping by similar group by set, measure set and slicer). Our system tries to combine both approaches by providing a probabilistic approach using Markov models where the states are clusters of queries, grouped according to a finer

similarity measure. Prediction models like the one used in PROMISE predict multidimensional queries based either on already visited queries (from query logs) or on resources provided by experts during a conceptual modeling process [11]. Sarawagi's work [13, 14] is a different, somewhat orthogonal approach, in the sense that the goal is not to model the user's querying behavior. Instead, users are led to the "most surprising" unvisited parts of the cube, whatever their past behavior was. Finally, Sarawagi leverages the query answers and the cube instance, while the present work requires to know only the query expression.

3 Query Model

We consider in the following a multidimensional schema $\mathcal{M} =< L, H, M >$ as defined in [15], where L is a finite set of levels, H a finite set of hierarchies (each including a subset of levels), and M a finite set of measures. We work with a basic form of OLAP query centered on a single multidimensional schema and characterized by an aggregation and a selection expressed through a conjunctive predicate. To be independent of the details related to logical design of multidimensional schemata and to specific query plans, we express queries using an abstract syntax. An OLAP query on schema $\mathcal{M}$ is thus defined as a triple $q =< g, P, Meas >$ where g is the query group-by set (including one level for each hierarchy in H), $P = \{c_1, \ldots, c_n\}$ is a set of Boolean clauses, one for each hierarchy, whose conjunction defines the selection predicate for the query, and $Meas \subset M$ is the measure set whose values are returned by the query. IPUMS is a public database storing census microdata for social and economic research (Minnesota Population Center, 2008). Its CENSUS multidimensional schema has five hierarchies as shown in figure 1, namely *RACE*, *TIME*, *SEX*, *OCCUPATION*, and *RESIDENCE*, and measures *AvgIncome*, *AvgCostGas*, *AvgCostWtr*, and *AvgCostElect*.

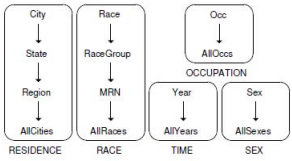


Fig. 1. Roll-up orders for the hierarchies in the CENSUS schema

Then, a query expressed as "Income per occupations and state in 2013" would have the following abstract representation:

$$q = \left[\begin{array}{c} < \mathsf{State}, \mathsf{AllRaces}, \mathsf{Year}, \mathsf{Occ}, \mathsf{AllSexes} > \\ \{\mathrm{TRUE}_{\mathsf{RESIDENCE}}, \mathrm{TRUE}_{\mathsf{RACE}}, (\mathsf{Year} = 2013), \mathrm{TRUE}_{\mathsf{OCCUPATION}}, \mathrm{TRUE}_{\mathsf{SEX}}\} \\ \{\mathsf{AvgIncome}\} \end{array} \right]$$

We give the MDX formulation of the query for illustration purposes below. MDX is a de-facto standard for querying multidimensional databases. Some of its distinguishing features compared to SQL are the possibility of returning query

results that contain tuples with different aggregation levels and the possibility of specifying how the results should be visually arranged into a multidimensional representation.

```
SELECT
    NON EMPTY {[Measures].[AvgIncome]} ON COLUMNS,
    NON EMPTY Hierarchize(Crossjoin({  [Residence].[State],
                                       [Race].[AllRaces],
                                       [Occupation].[Occ],
                                       [Sex].[AllSexes] }))
    DIMENSION PROPERTIES
    PARENT_UNIQUE_NAME ON ROWS
FROM ( SELECT {[Time].[Year].&[2013]}
        ON COLUMNS FROM [CENSUS])
```

4 Query Prediction for Recommendation

This section presents our approach for query prediction for recommendation. First we give an overview of the proposed architecture and we outline the query similarity metric we adopt, then we explain the clustering and prediction steps needed for recommendation. A sketch of the functional architecture we propose is shown in Figure 2. After the user has formulated a query, the query processing component is in charge of processing it and getting the results back from the data warehouse. Each query issued by the user in a session will also be stored in the query log. Based on the information available in the query log, the clustering & learning module is responsible for dynamically determining the user behaviour model (learning step) from the recognized clusters representing similar queries (clustering). The user's behaviour model, learned and updated by the clustering and learning module, will then be used by the prediction module. It should be noted that clustering and learning the user behavior model are done at regular times offline, and then the model is consulted at runtime to predict the next user query. Finally, the prediction module, guided by the user's current query, is based on the results from the discovery process previously stored in the past history of user queries. From this data, the prediction module is able to recommend the

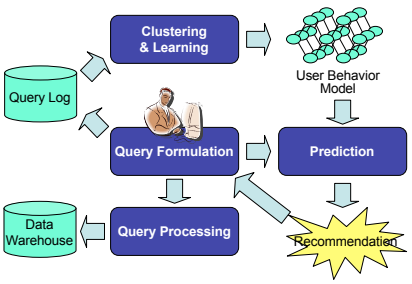


Fig. 2. Functional architecture for query recommendation

next query. From the user's behaviour model inferred by the learning module, the prediction module will determine the user's most likely future query. In order to recommend queries to users, we build a behaviour model that predicts the most likely next queries. The approach consists of 2 steps, the first being the clustering of similar queries based on a query similarity metric. In the second step we treat these clusters as states of a Markov chain model and compute the probability of the most likely next state (i.e. cluster of queries). In the clustering process, similar queries are grouped into clusters, as a way to reduce the size of the history log and to be able to recommend similar queries. These clusters are interpreted as states of a state machine, and the transition probabilities from one state to another are calculated based on the history. The interpretation of the current OLAP session as a trajectory of states allows to anticipate most probable future states. In our approach, this process consists of estimating the probabilities of moving from one state to other possible future states. The query finally recommended is then the one that is most similar to the current query and that is a member of the most likely next state, given the state the current query belongs to. The state the current query belongs to is the cluster whose members are on average most similar to the current query, again computed with the query similarity metric. So the recommended query is the most similar query in the most likely next state.

4.1 Query Similarity Metric

As shown in the previous section, our approach relies on a metric to compute the similarity between OLAP queries both for clustering and predicting. To be used in our context, a similarity metric must meet two requirements: First, efficiency is required because, during the prediction step, the current query q has to be matched to the closest state in the Markov model, so the similarity between the current query and all the states in the model must be computed. On the other hand, multidimensional databases store huge volumes of data, and as a consequence OLAP queries can return large result sets. Extensional computation of query similarity (i.e., made by comparing query results like in [16] and [17]) can thus pose serious efficiency problems. For this reason we use a metric that computes query similarity at the intensional level, i.e., by only looking at the query expressions. Second, the space of possible queries on a multidimensional schema is large, there is little probability that two queries are really identical. So, the metric we adopt should return a score and not a Boolean value like in [18].

We adopt as a metric for computing the similarity between two queries q and q' the one defined as σ_{que} in [15]:

$$\sigma_{\text{que}}(q, q') = \alpha \cdot \sigma_{\text{sel}}(q, q') + \beta \cdot \sigma_{\text{meas}}(q, q') + \gamma \cdot \sigma_{\text{gbs}}(q, q') \tag{1}$$

where σ_{sel}, σ_{meas}, and σ_{gbs} represent respectively the *selection*, *measure* and *group-by set* similarities as defined in [15], and $\alpha, \beta, \gamma \in [0, 1]$ are parameters to be experimentally determined. A priori, the three terms in the equation are not equally important. Based on a set of tests made with users, in [15] it is argued

that the selection predicate is the most important in determining similarity between OLAP queries, followed by the group-by set; the least significant term is the set of measures to be returned.

As an example we take two queries on the CENSUS schema specified as follows:

$$q_1 = \left[\begin{array}{c} < \mathsf{State}, \mathsf{RaceGroup}, \mathsf{Year}, \mathsf{Occ}, \mathsf{AllSexes} > \\ \{\mathrm{TRUE}_{\mathsf{RESIDENCE}}, \mathrm{TRUE}_{\mathsf{RACE}}, (\mathsf{Year} = 2005), \mathrm{TRUE}_{\mathsf{OCCUPATION}}, \mathrm{TRUE}_{\mathsf{SEX}}\} \\ \{\mathsf{AvgCostWtr},\ \mathsf{AvgCostElect}\} \end{array} \right]$$

$$q_2 = \left[\begin{array}{c} < \mathsf{State}, \mathsf{RaceGroup}, \mathsf{Year}, \mathsf{Occ}, \mathsf{AllSexes} > \\ \{\mathrm{TRUE}_{\mathsf{RESIDENCE}}, \mathrm{RaceGroup} = \mathrm{Chinese}, (\mathsf{Year} = 2005), \mathrm{TRUE}_{\mathsf{OCCUPATION}}, \mathrm{TRUE}_{\mathsf{SEX}}\} \\ \{\mathsf{AvgCostWtr},\ \mathsf{AvgCostElect}\} \end{array} \right]$$

For these queries, only differing in selection predicate, $\sigma_{\mathrm{que}}(q1, q2) = 0.95$, taking for simplicity equal weights for α, β and γ.

4.2 Clustering

The first step of our approach is the clustering of user's queries. During different analysis sessions, a user often expresses similar (but not identical) queries; for instance, she may formulate queries with the same group-by set but on different slices of data (which means, with different selection predicates). Indeed, as the query log contains a trace of all queries formulated by each user, it is very likely that some of them will be similar. The clustering algorithm is inspired by standard density-based clustering methods (k-means with dynamic number of clusters), but we define the clustering space as being build using the similarity metric introduced in the previous section. The input to this step are the abstract representations of the queries previously issued by the user, stored in the log. The goal is to determine query clusters in such a way that the queries in the same cluster are similar (in the sense of Section 4.2) to each other, and that queries in different clusters are not similar to each other. We start with randomly selecting a number of queries from the query logs, which will serve as the seeds for the clusters (Listing 1.1 line 3). This random selection of queries ensures that the clustering mechanism takes into account the density of queries over the query space introduced by the query similarity metric. Then, we assign each query q in the log (Listing 1.1 line 4) to the cluster whose queries on average have highest similarity with q (Listing 1.1 lines 6-8). To avoid that clusters grow unlimited in size, a cluster split rate is defined; when a cluster reaches a given number of member queries, the cluster is split in two (Listing 1.1 line 9).

```
1. int cluster_split_rate = t;
2. List<Query> queries = read_in(ipums_log.txt);
3. List<Cluster> clusters = select_random_seeds(nbOfSeeds);
4. for each Query q in queries{
5.   for each Cluster c in clusters{
6.      compute_average_similarity(q,c);}
7.      Cluster selected_c =
8.            assign_query_to_closest_cluster(clusters,q);
9.      if (selected_c.size()>t) selected_c.split();}
```

Listing 1.1. Clustering similar queries

4.3 Learning the User Behavior Model

We model the querying behavior of each user in the form of a Markov chain, where each query cluster (determined as explained in Section 4.3) is a state. The series of states satisfies the Markov property, i.e., the probability of reaching a state in the future, given the current and past states, is the same probability as that given only the current state. This means that past states give no information about future states. More precisely, if the system is in state x at time n, the probability that it moves to state y at time $n+1$ depends only on the current state x and not on past states. The transition probability distribution can then be represented as a matrix P, called a *transition matrix*, whose (i, j)-th element is defined as follows:

$$P_{ij} = Pr(X_{n+1} = j | X_n = i)$$

The initial probability $Pr(X_{n+1} = j | X_n = i)$ is $\frac{1}{m}$, where m is the number of states that can follow the current state. In our case, each state corresponds to a cluster of queries, giving as value for m the initial number of clusters that were identified in the clustering step.

The probability $Pr(X_{n+1} = j | X_n = i)$ could be updated by counting how often query q_j is preceded by query q_i and dividing this number by the total number of queries that were observed as following query q_i. This means however that the past is as important as the present. In the OLAP context, the series of queries a user performs will typically evolve over time. So, if the log includes for instance the queries performed by the user during the last six months, it is reasonable to have more recent queries having relatively more influence on the user behavior model than older ones. To this end, the transition probability function should be updated in such a way that recent transitions have more relevance than older ones, which we do using an exponential smoothing method:

$$P_{ij} = \rho \times x_j + (1 - \rho) P'_{ij}$$

where P'_{ij} represents the old probability and $x_j \in \{0, 1\}$ is the value for the choice taken at query q_i with respect to query q_j. If $x_j = 1$ then q_j was executed after q_i, if $x_j = 0$ it was not. Using this method, the sum of all outgoing probabilities remains 1, as required for a transition probability matrix. The *learning rate* $\rho \in [0, 1]$ is a real number that controls how important recent observations are compared to history. If ρ is high, the present is far more important than history; in this setting, the system will adapt quickly to the behavior of the user, which can be necessary in a rapidly changing environment or when the system is deployed and starts to learn. In a rather static environment, ρ can be set low. In conclusion, by incorporating the learning rate, we make sure the user behavior model is dynamic and evolves together with changing habits or preferences of users.

The algorithm for learning the user behavior model is shown below. Based on the clusters of queries that were obtained in the clustering step (Listing 1.2 line 1) and on the query log (Listing 1.2 line 2), a transition probability matrix (Listing 1.2 line 3) is constructed expressing the probability of going from each

cluster to each other cluster. For this, each query q_i and q_{i+1} in the log is considered (Listing 1.2 line 4). We check to which clusters q_i and q_{i+1} belong to (Listing 1.2 line 5-6) and update the probabilities in the transition probability matrix accordingly (Listing 1.2 line 7).

```
1. List<Cluster> states = get_query_clusters();
2. List<Query> queries = read_in(ipums_log.txt);
3. float[][] transition_matrix = new float[nb][nb];
4. for each Query q(i) and q(i+1) in queries
5.  Cluster p = assign_query_to_cluster(q(i), states);
6.  Cluster q = assign_query_to_cluster(q(i+1), states);
7.  update_probs (transition_matrix, p, q);
```

Listing 1.2. Learning the user behavior model

Once the user behavior model has been constructed, it can be exploited to recommend a next query to the user. Given the current user query, we identify to which state (i.e. cluster of queries) in the Markov model this query is closest to by computing the average similarity between the current query and each state. This similarity is equal to the average of the similarity between the current user query and the queries contained in the state. Since the similarity metric we use computes its value intensionally and not extensionally, this is computationally feasible. Once we have identified the appropriate state, the Markov model gives us the most probable next state by looking in the transition probability matrix for the highest probability on the row corresponding to the matching state. In this most probable next state, we retrieve the query that is most similar to the current query, and propose this one as the query predicted for recommendation. For example, given the current user query is query q_2 introduced in section 4.2, we identify the most probable next cluster from the transition probability matrix. This most likely next cluster turns out to have probability 0.36, which value gives us good confidence in the correctness of the prediction (as will be discussed in the next section). In this cluster, we select the query q_3 most similar to q_2 for recommendation, which is shown below. Query q_3 differs from q_2 by adding the *SumCostWtr* predicate.

$$q_3 = \left[\begin{array}{c} < \mathsf{State}, \mathsf{RaceGroup}, \mathsf{Year}, \mathsf{Occ}, \mathsf{AllSexes} > \\ \{\mathrm{TRUE}_{\mathsf{RESIDENCE}}, \mathrm{RaceGroup} = \mathrm{Chinese}, (\mathsf{Year} = 2005), \mathrm{TRUE}_{\mathsf{OCCUPATION}}, \mathrm{TRUE}_{\mathsf{SEX}}\} \\ \{\mathsf{AvgCostWtr},\ \mathsf{AvgCostElect},\ \mathsf{SumCostWtr}\} \end{array}\right]$$

5 Experiments and Evaluation

We evaluated our approach using a synthetic dataset of log traces of MDX queries, with the goal of answering the following questions: 1) How does the error rate evolve and is influenced by the cluster split rate (i.e. the average size of clusters). 2) How much does the error rate improve when using a threshold for the prediction probability P in considering to recommend.

To the best of our knowledge, available public dataset logs like the one used by [19] do not correspond to an OLAP query log in the sense that they do not have a multidimensional schema, and the SQL queries in these logs cannot be seen as OLAP queries. In particular, the vast majority of them does not have grouping and aggregation, and therefore do not fit our OLAP query model. Therefore, we opted for generating synthetic datasets for experimentation. The synthetic dataset consists of query session logs that were generated considering a specific policy, using the IPUMS CENSUS multidimensional schema. In a first step, random queries were generated and grouped together according to their similarity. This allows for a broad coverage of the space of possibe queries since there is a distance between the groups. Each group corresponds thus to one type of session. In a second step, we generate a number of sessions for each group, queries in the group acting as seeds. For this we select at random a query A and a query B in a the group, A being the start query of the session and B being the last query of the session. The shortest OLAP path (series of OLAP operations) between A and B is calculated and each OLAP operation is translated into one OLAP query, as such generating the session. While the specific queries contained in the path from A to B are fixed, variance was introduced in the order of the queries to obtain more realistic sessions. We used 75% of this dataset for training purposes to build the user behavior model, and 25% as a testing set to perform the evaluation.

In order to gain insight in the performance of the query prediction in terms of correctness, we propose in this section a set of metrics. The evaluation procedure for our approach consists in requesting a prediction (based on the current query) and comparing its correctness with the actual next query. By doing this for a series of queries, we get an overall view of prediction correctness. First, we compare the predicted and actual query incorporating the similarity measure $\sigma_{\text{que}}(q, q')$ defined in section 4, taking for q_1 the predicted query q_{pred}, and for q_2 the actual next query q_{act}. Then, we define the probability of correctness P, which is the probability of the most likely prediction, i.e. the query which has the highest probability of being executed next according to the prediction model. In statistics, this corresponds to the confidence one has in a classification.

The performance of the prediction approach can then be assessed by using the following metric:

$$S = \frac{1}{n} \sum_{t=1}^{n} \pi_t$$

where π_t equals $\sigma_{\text{que}}(q, q')$ at time t. Since the main goal of predicting the next query is to be able to proactively execute it, we consider that even if the predicted and actual next query are not exactly the same, largely similar operations will be performed and facts prefetched. Therefore, we do not choose the give π_t a value of 0 or 1 exclusively, but to allow for an inexact match between queries by using the value $\sigma_{\text{que}}(q, q')$. A variant of this metric is the thresholded S, being S_t, where only predictions are taken into account that have a probability P above a threshold (which we fixed at 0.3). Figure 3 shows how S and S_t evolve according to the cluster split rate.

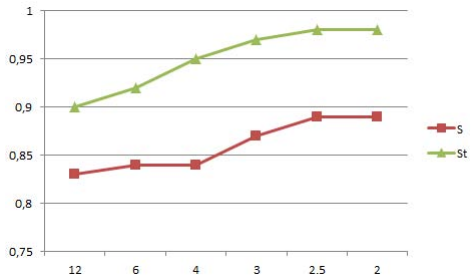


Fig. 3. S and St in function of cluster split rate

In addition we evaluate the recommendation mechanism by using the following metrics:

- *Standard error rate E_s:* This corresponds to an unweighted error rate being the proportion of incorrect predictions over all predictions performed.
- *Error rate with threshold E_t:* The same as the standard error rate, taking into account only the predictions which have a probability of correctness above a threshold.
- *Coverage C:* This metric is defined as the proportion of items (i.e. queries) for which it is possible to do a recommendation. In our case, we consider it possible to perform a recommendation in case the probability of correctness of the predicted query is above a threshold.

It should be noted that P, the probability of the prediction, can be used to take a decision on whether to effectively execute the predicted query proactively or not. By setting a threshold for P, only queries that are predicted with rather high confidence can be executed, minimizing the risk of executing a wrong query and thus wasting resources. The threshold for P is thus useful in deciding on the quality of an individual concrete prediction. On the other hand, the metric E_t tells something on the general performance in terms of correctness of the prediction approach.

Experiments (see figure 4) show that for a cluster split rate of 2.5 the error rate E_s is 0.44, which is too high for prefetching purposes but could be considered for recommendation purposes. However, if we introduce a threshold T of 0.3 on the probability P of the predictions, the error rate drops to 0.12, which is more acceptable. It should be noted that 26% of predictions were done with a probability of 0.3 or higher. The value of the treshold T is of great importance and influences the success of prediction. When setting it too low (e.g. T=0.2), the error rate increases significantly by 10 to 20%, depending on the number of clusters. When setting T too high (e.g. 0.4), the number of predictions satisfying a P above the treshold decreases dramatically. Moreover, the figure shows that at a cluster split rate of 2.5, E_t stabilizes for lower values of the split rate. The only difference is that when the split rate goes to 2, a more and more lower proportion of predictions satisfies the threshold T, which means the coverage

goes down, as can be seen in figure 4. For example, at split rate 2.5, 24% of predictions satisfies T, while at split rate 2.0, this is only 15%. This leads to a choice of balance between E_t and C because the lower E_t, the lower also the ability to do recommendations. Since minimizing E_t is most important to avoid giving erroneous recommendations to the user, and since E_t stabilizes at a split rate of 2.5 while having a coverage of 24%, we identify this point as optimal.

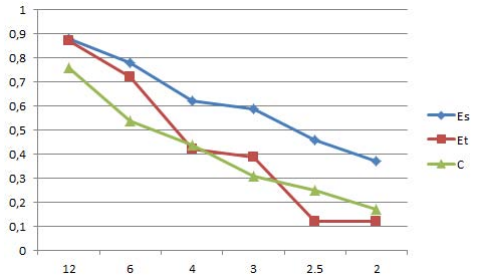


Fig. 4. Es, Et and C in function of cluster split rate

6 Conclusion and Future Work

In this paper, we proposed an approach to query recommendation that combines a probabilistic user behavior model with a query similarity metric. Instead of only relying on the similarity of queries to do a recommendation, the incorporation of the probability of a predicted query allows to define a threshold to decide on the trust one can have in the prediction. Introducing the threshold allows to avoid faulty predictions, improving the quality of experience for the user and avoiding waste of computational resources, while keeping the coverage at an acceptable level. Preliminary evaluation of the approach on a synthetic dataset makes us confident that the recommendation mechanism can provide added value to users in guiding them through their OLAP sessions. As for future work, our goal is to perform an evaluation involving real OLAP users, introducing subjective metrics to gain insight in how appropriate the recommendation is perceived by users. Also, the Markov-based user behavior model will be extended to include in the prediction process not only the current user query, but also characteristics of the whole current OLAP session. Finally, taking into account the n previous queries in doing prediction (higher-order Markov chains) will be investigated as an extension to our approach.

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Trans. on Knowl. and Data Eng. 17(6), 734–749 (2005)

2. Baeza-Yates, R., Hurtado, C.A., Mendoza, M.: Query recommendation using query logs in search engines. In: Lindner, W., Fischer, F., Türker, C., Tzitzikas, Y., Vakali, A.I. (eds.) EDBT 2004. LNCS, vol. 3268, pp. 588–596. Springer, Heidelberg (2004)
3. Khoussainova, N., Balazinska, M., Gatterbauer, W., Kwon, Y., Suciu, D.: A Case for A Collaborative Query Management System. In: CIDR 2009, Fourth Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7. Online Proceedings (2009)
4. Stefanidis, K., Drosou, M., Pitoura, E.: You May Also Like results in relational databases. In: Proceedings International Workshop on Personalized Access, Profile Management and Context Awareness: Databases, Lyon, France (2009)
5. Chatzopoulou, G., Eirinaki, M., Polyzotis, N.: Query Recommendations for Interactive Database Exploration. In: Winslett, M. (ed.) SSDBM 2009. LNCS, vol. 5566, pp. 3–18. Springer, Heidelberg (2009)
6. Khoussainova, N., Kwon, Y., Balazinska, M., Suciu, D.: SnipSuggest: Context-Aware Autocompletion for SQL. PVLDB 4(1), 22–33 (2010)
7. Khoussainova, N., Kwon, Y., Liao, W.-T., Balazinska, M., Gatterbauer, W., Suciu, D.: Session-based browsing for more effective query reuse. In: Bayard Cushing, J., French, J., Bowers, S. (eds.) SSDBM 2011. LNCS, vol. 6809, pp. 583–585. Springer, Heidelberg (2011)
8. Drosou, M., Pitoura, E.: Redrive: result-driven database exploration through recommendations. In: Macdonald, C., Ounis, I., Ruthven, I. (eds.) CIKM, pp. 1547–1552. ACM (2011)
9. Sellam, T., Kersten, M.: Meet Charles, big data query advisor. In: CIDR 2013, Sixth Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 6-9. Online Proceedings (2013)
10. Giacometti, A., Marcel, P., Negre, E.: A framework for recommending OLAP queries. In: Proc. DOLAP, Napa Valley, CA, pp. 73–80 (2008)
11. Sapia, C.: PROMISE: Predicting query behavior to enable predictive caching strategies for OLAP systems. In: Kambayashi, Y., Mohania, M., Tjoa, A.M. (eds.) DaWaK 2000. LNCS, vol. 1874, pp. 224–233. Springer, Heidelberg (2000)
12. Howard, R.: Dynamic programming and Markov processes. Technology Press of Massachusetts Institute of Technology (1960)
13. Sarawagi, S., Agrawal, R., Megiddo, N.: Discovery-driven exploration of OLAP data cubes. In: Schek, H.-J., Saltor, F., Ramos, I., Alonso, G. (eds.) EDBT 1998. LNCS, vol. 1377, pp. 168–182. Springer, Heidelberg (1998)
14. Sarawagi, S.: User-adaptive exploration of multidimensional data. In: Proc. VLDB, Cairo, Egypt, pp. 307–316 (2000)
15. Aligon, J., Golfarelli, M., Marcel, P., Rizzi, S.: Similarity measures for OLAP sessions. International Journal of Knowledge and Information Systems (to appear, 2013)
16. Giacometti, A., Marcel, P., Negre, E.: Recommending multidimensional queries. In: Pedersen, T.B., Mohania, M.K., Tjoa, A.M. (eds.) DaWaK 2009. LNCS, vol. 5691, pp. 453–466. Springer, Heidelberg (2009)
17. Drosou, M., Pitoura, E.: ReDRIVE: result-driven database exploration through recommendations. In: Proc. CIKM, Glasgow, UK, pp. 1547–1552 (2011)
18. Yang, X., Procopiuc, C., Srivastava, D.: Recommending join queries via query log analysis. In: Proc. ICDE, Shanghai, China, pp. 964–975 (2009)
19. Chatzopoulou, G., Eirinaki, M., Koshy, S., Mittal, S., Polyzotis, N., Varman, J.S.V.: The querie system for personalized query recommendations, 55–60 (2011)

Where Will You Go? Mobile Data Mining for Next Place Prediction

João Bártolo Gomes[1], Clifton Phua[2], and Shonali Krishnaswamy[1]

[1] Institute for Infocomm Research (I2R), A*STAR, Singapore
{bartologjp,spkrishna}@i2r.a-star.edu.sg
[2] SAS Institute Pte Ltd, Singapore
clifton.phua@sas.com

Abstract. The technological advances in smartphones and their widespread use has resulted in the big volume and varied types of mobile data which we have today. Location prediction through mobile data mining leverages such big data in applications such as traffic planning, location-based advertising, intelligent resource allocation; as well as in recommender services including the popular Apple Siri or Google Now. This paper, focuses on the challenging problem of predicting the next location of a mobile user given data on his or her current location. In this work, we propose *NextLocation* - a personalised mobile data mining framework - that not only uses spatial and temporal data but also other contextual data such as `accelerometer`, `bluetooth` and `call/sms` log. In addition, the proposed framework represents a new paradigm for privacy-preserving next place prediction as the mobile phone data is not shared without user permission. Experiments have been performed using data from the Nokia Mobile Data Challenge (MDC). The results on MDC data show large variability in predictive accuracy of about 17% across users. For example, irregular users are very difficult to predict while for more regular users it is possible to achieve more than 80% accuracy. To the best of our knowledge, our approach achieves the highest predictive accuracy when compared with existing results.

1 Introduction

Next place prediction is a particular problem of location prediction where the challenge consists of predicting the next location of a mobile user given his current location [13,10]. Most existing work models next place prediction as a classification problem, where spatial and temporal data is used for training.

However, issues such as the integration of other rich contextual data, available on smartphones nowadays such as `accelerometer`, `bluetooth` and `call/sms` logs have not been seriously investigated. In addition, most existing approaches focus mainly on the classification problem assuming the data is in a centralised server while other problem specific issues related to user behavioural changes, privacy, data management and scalability have not been explored in-depth.

To address these issues in this paper, we propose *NextLocation* - a novel integrated framework for the next place prediction problem - that predicts the next

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 146–158, 2013.

location using only current location and contextual data for each mobile phone user. *NextLocation* learns an "anytime" classification model which incorporates past data to predict the next place in an incremental manner. It enables greater personalisation and privacy while bringing the whole learning process on-board the mobile device. Moreover, in addition to spatial and temporal information, the proposed approach combines other context information available on the mobile device. The main advantages of the *NextLocation* for next place prediction are:

- Privacy-preserving, as it allows the user to be in control of their personal data. Using *NextLocation* for mobile data mining, personal mobile phone data is not shared with an external party without permission.
- Reduced communication overheads in terms of bandwidth as well as battery drain, since local processing is usually less expensive than wireless data transfer [15].
- Dynamic instead of static model building facilitates the model adaptation so that the current up-to-date user behaviour is reflected.
- Allows personalised online estimation of the next place predictive accuracy.
- Enables an alternative business model where advertisement providers can push content that is relevant to a certain location and the user only receives it when he is about to visit that location.

The rest of the paper is organised as follows. The following Section reviews the related work. Section 3 presents next place prediction as a classification problem, which is followed by a detailed description of the feature engineering from Nokia Mobile Data Challenge (MDC) in Section 4. The proposed approach for next place prediction (*NextLocation*) is presented in Section 5. The experimental setup and results are discussed in Section 6. Finally, in Section 7, conclusions of this work as well as ideas for future work are presented.

2 Related Work

Location prediction assumes that mobile sensor observations from wireless local network (Wi-Fi), Global System for Mobile Communications (GSM), Global Positioning System (GPS) are available. The prediction task consists of using such data to know and understand the user's current location. The research on mobile user visiting behaviour, can bring additional value to different domains, such as mobile advertising, resource allocation and disaster relief.

In this work, we are interested in a related but more challenging location prediction problem, which aims to predict the next location without knowing in advance the readings from future sensor data. In general, the mobile data used for the next location problem consists of the historical information about the `visit` sequences and associated context information (for example, timestamps, `accelerometer`, `bluetooth` and `call/sms` log) from these visits.

There is extensive research on the problem of predicting future locations. Most of such work creates a model based on frequent patterns and association rules from a history of user/collective trajectories as an ordered sequence of

locations that are timestamped [13]. Other sequential learning models such as Hidden Markov Models [12], Conditional Random Fields [14] and Particle Filters [2] have been also applied to this problem. However, the problem addressed in this paper is different because, for any user, the prediction of the next location assumes only knowledge about the current location (without data about previous locations). This limited/reduced history makes our problem more general as it is not unusual to have gaps in mobile sensor data. *Gaps* refer to significant time periods where the mobile phone is not collecting data (for example, when the mobile phone has run out of battery).

Recently the Nokia Mobile Data Challenge (MDC) released a large dataset for research and one of the dedicated tasks consisted of next place prediction [10]. From this MDC challenge, several approaches were able to predict the next place with high accuracy [1,18,5,11,17]. The proposed approaches focused on learning a model for each user which captures the spatio-temporal trajectory of user visits. Significant effort was dedicated to feature engineering for each approach.

Still, two main issues remain relatively unexplored in the literature of next place prediction. First, privacy issues arise from using such data, although there are efforts in the direction of anonymization [10]. Second, rich context information can be exploited for personalisation. In this paper, we try to address these issues by proposing a mobile data mining framework that does not require the raw data to be disclosed and that the model built is highly personalised.

3 Next Place Prediction: Definition

First let us assume we are interested in finding the next destination of a single user when (s)he is still at the current location. It is easy to generalise from this problem to multiple users. Consider $L = \{l_1, ..., l_n\}$ to be the set of values of visited (for a minimum time threshold) spatial locations, the $T = \{t_1, ..., t_n\}$ to be the set of timestamps and $C = \{c_1, ..., c_n\}$ to be the set of context information where c_i represents itself a set of attribute value pairs that are in available at t_i. This context information is usually the data available in the user's mobile phone and can be collected from the `accelerometer`, `bluetooth`, `call/sms` log, `wlan` (Wi-Fi) or `phone status` (consider that for some users charging the phone is only performed at certain locations).

Given a series of historical visits to different locations in the past, that constitutes the data available for training $H = \{(L, C, T)\} = \{(l_1, c_1, t_1), ..., (l_k, c_j, t_j)\}$ and the context $C = ctx(t_i)$ at $T = t_i$ of the latest location $L = loc(t_i)$, the next place prediction problem can be formulated as finding the most likely location $argmax_{l \in L}(p(L_{next} = l | T = t_i, C = ctx(t_i), L = loc(t_i)))$

Please note that the prior only considers the current location and not the past location or a sequence of previous locations as is usually modelled using Hidden Markov Models (HMM) [12] or Conditional Random Fields (CRF) [14]. The reason for this is simple, it is not always possible to have a sequence of visits without gaps, therefore, we prefer to define the more general problem where we are able to make a prediction if we know at least the current location.

In Section 6 we will describe in detail a particular instantiation of the problem, the associated feature engineering process, and report and discuss the results of our experiments with real data.

4 Mobile Data Challenges

In this section, we discuss the challenges that come from collecting mobile data for the next place prediction problem. Understanding the whole data process and its requirements allowed us to design and explore the alternative solution proposed in this paper. The following subsections describe what we consider some of the challenges that need to be addressed to transform the data available in the mobile phone so that it can be used to induce a model useful for next place prediction as previously defined.

Location Detection. The raw data of each user's location is usually estimated based on GPS and Wi-Fi that is then transformed into a semantic place (for example, workplace, home, or restaurant) which captures most of the mobility/location-based information without including the actual geographic coordinates/access points. Moreover, information from social networking services that support location, such as Four-Square, Facebook or Google Latitude already allows the user to 'check in'. These services already include automatic location detection which can be leveraged to create or enrich the temporal series of semantic locations, that we require for next place prediction. Therefore, this paper will not focus on semantic place (for example, place tagged as home, workplace, or transportation place) prediction but on next place prediction as we formally define in Section 3.

User Specificity. Next place prediction is a user specific problem as the set of locations visited is personal and even if this set might overlap among different users the trajectory of user visits to different locations is most likely unique. It is therefore, hard or impossible to accurately learn joint models over multiple users as can be performed in other classification tasks such as activity/speech recognition. The challenge of user specificity motivates the use of a personalised model.

Evolving Data. The user movement behaviour might change over time. For instance, changing house/city/country/workplace can have a profound impact on the most recent movement pattern. Therefore, we propose that modelling should be adaptive and the usage of an incremental anytime model, that incorporates new information and forgets old outdated information. Moreover, the model should incorporate novel locations seamlessly.

Sparse and Missing Data. It is possible to have missing data or gaps in the sequence of visits to particular locations. This is the main reason that led us to formulate the next place prediction problem considering only the current location and not a sequence of past locations that precede the current location in time. This challenge is related with model evaluation as the number of observations (evidence that from location l_i the user moved to location l_j) and how representative they are of user mobility patterns will have a high impact on the accuracy of the learnt model.

5 Next Place Prediction: *NextLocation*

The framework proposed in this paper, that we call *NextLocation*, models next place prediction as a classification problem. However, instead of executing the traditional learning process (i.e., data collection, data transfer, model building, model deployment), we create an integrated framework that is executed on the mobile device itself.

One key innovation of *NextLocation* is that it preserves user privacy as it allows the building of a model for next place prediction without disclosure of private user data. Such framework gives the user control over who can use the model's results (i.e., the next location predictions) without disclosing the real locations visited and associated context data.

Figure 1 illustrates the *NextLocation* learning process and its components. We can see that the pre-processing component, anytime model, and accuracy estimator play a central role in the proposed framework. Each of these components performs the following:

- Pre-Processing - the raw data must be pre-processed/transformed for next place prediction. Here, the location data from a visit is enriched with other context information. The pre-processing component only requires to keep a short term sliding window of data (i.e., current and previous visit). When updating the model, the data represents the previous visit location and its context information, and the target variable (to predict) is the current location.
- Anytime Model - must be able to integrate new information as it is available (such as new visits) and must also be able to predict the next location. Any classification algorithm that learns incrementally can in principle be used in this component to create/update the anytime model. Moreover, these algorithms are light-weight and can be executed using the computational resources usually available on current smartphones. However, it is beneficial if the algorithm can adapt the anytime model when there is evolution in the observed data.
- Accuracy Estimator - comparing the anytime model prediction with the actual destination allows us to keep an estimate of next place prediction accuracy.

Past data can be discarded once it is incorporated into the anytime model, consequently, the memory consumption of the *NextLocation* learning process is

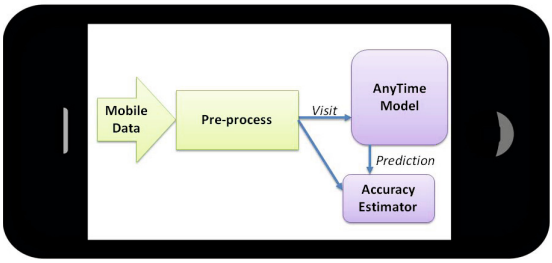


Fig. 1. *NextLocation*: framework overview

very low compared to approaches that require all data to be collected beforehand and processed in batch mode.

Adapting the Model. Given the issue of data evolution it is important to adapt the anytime model. For instance, adapting the model to a new living environment that causes a change in the user mobility patterns. In such situations it is likely that the most recent past represents the activities of interest and less importance should be given to older records that represent past behaviour.

On-line Model Evaluation. As part of the proposed framework we keep an estimate of the anytime model accuracy online. Here we briefly formalise the evaluation procedure. The prequential-error [4] is computed based on an accumulated sum of a loss function L between the anytime model prediction $\widehat{l_i}$ and the location that is visited next l_i.

6 Experimental Evaluation

This section describes the experiments that were performed to evaluate *Next-Location* approach feasibility and accuracy. The data used in the experiments has been released for the Nokia Mobile Data Challenge (MDC) [10], and was collected from the smartphones of almost 200 participants over the course of over one year in a real world environment.

There was a significant effort for data transformation/feature engineering. We used a total of 70 features including: 11 temporal features, 8 `accelerometer` features, 2 `bluetooth` features, 23 `calllog` (call/sms) features, 20 `visit` related features, 6 `system` features.

6.1 Nokia MDC Dataset

The MDC data was collected on a 24/7 basis over months. In the Dedicated Track of this competition, which included the task of next place prediction, the raw location data is transformed into the sequence of visits to symbolic places.

The users in MDC data are sampled into three separate sets. The training data set (called setA) consists of mobile phone data collected from 80 persons during a period of time varying from a few weeks to two years. The unseen data for each participant in setA is used to build the test data set (called setC), where the unseen data corresponds to the continuation of setA (in time). The setC ground truth was never released after the challenge. However, the validation set (called setB)was released and is used to evaluate the results. The validation set contains visits that were randomly chosen from the last part of setA (in time). The validation set was built by filtering data in setA with time intervals corresponding to the randomly chosen visits.

For the MDC challenge, participants were free to estimate the context from all the available data within a determined time interval (i.e., current location corresponding to a visit in a place). The visits were timestamped with the start/end point entering/leaving the location visited. *Trusted* visits (provided with raw

features in the form of trusted_start, trusted_end in the `visit_sequence` tables) are more reliable than untrusted visits. For this task only visits where the mobile user stays in that location for 20 or more minutes are considered. Moreover, information about whether the transition for that visit location is to be trusted or not is available (i.e., reliable sensor data).

The MDC database has 5 main types of data: environmental, personal, phone usage, phone status, and visits data. This data is represented across 18 tables, with more than 130 raw attributes, and is approximately 50 GB in size. A detailed description of the data collection campaign is available in [10].

A significant challenge that we observed when working with this data, was the fact that while some users had highly regular patterns of movement, for some users there was significant variability. Clearly, the former mobile users have a higher predictability of movement, than the latter. This is further compounded by the fact that some mobile users have significantly more data than the others (though it must be said that more data does not necessarily in this case imply higher predictability).

6.2 Data Transformation

In the MDC dataset the transformation of raw location data into a sequence visits (each visit is more than 20 minutes as provided in the challenge) to symbolic places was already processed. The timestamped visit sequence is the key data for next place prediction and is similar to what has been proposed in [16]. However, other context information that might be used to improve the predictive performance needs to be derived from the raw data associated to those visits. In our experiments we ended up with 70 features. In this section we describe our feature extraction process. We would like to note that these features were all calculated per user and using a sliding window approach, that is, the raw data is processed locally and then discarded without the need to keep all the information in main memory. A frequency table with statistics (e.g. number of visits on particular temporal periods) about the different locations is also kept to calculate more sophisticated features (such as from `bluetooth`).

Temporal Features. From the start and end timestamps of a particular visit several temporal features were generated. The duration of the visit, the day of the week, weekend or workday, period of the day in two different sets: (AM/PM) ; (morning, afternoon, evening, night); hour of the day (0h-24h). These features can be calculated from both the start and end timestamps.

Phone Status Features. Several types of data about the phone status and the phone operating system was recorded. From this data we derived features to capture the phone status that was characteristic of the visit to a particular location. The most frequent profile (general, silent), the most frequent ring tone used (normal, silent), minimum and maximum battery level, phone charging status, maximum inactive time.

Phone Usage Features. From the phone usage, we consider the information available in the call log, in particular, the most frequent number. We expect that

this might help us to capture situations where our next destination is highly correlated with receiving a certain call or text. Usually, before a mobile user leaves the current location for the next destination, the last call or SMS can be quite predictive of the next destination (for example, the mobile user calls the person who (s)he will meet later in the next destination). The features generated were the most frequent number: overall, in a call, in a text, in an incoming/outgoing overall, in an incoming/outgoing call, in an incoming/outgoing text, missed call, and the same features calculated but instead of the most frequent the last observation (e.g., last number called, last text sent). From the last call we calculate its duration and if it is an incoming or outgoing call. In addition, we calculate the number of: missed calls, incoming/outgoing calls, incoming/outgoing texts.

Environmental Features. For the environmental features we explored data from 4 different sensors, `accelerometer`, `bluetooth`, `wlan` and `gsm`. However, since the data is anonymized per user it was impossible to capture information across users. For instance, if two users are in contact with the same GSM tower or Wi-Fi access point the hashed values or the corresponding cell tower ID and access point mac address will appear different despite being the same physical object. Therefore, we used information that is personalised and for which the hash key matches to the same object that might capture some useful information to the mobile user destination. As environmental features we used:

- `bluetooth`: Similarly to the motivation behind the features we have generated from the call log we tried to understand if there is a certain bluetooth device nearby that influences the next place. We generated one feature that requires some statistics about the current location and observed bluetooth mac addresses for the location. The process tries to calculate the likelihood that a certain mac address in the current location is associated with a particular destination.
- `accelerometer`: Accelerometer features that might help to characterise the activity at a given location [6,7]. This captures a different type of activity compared to the phone status inactivity feature. For instance, there might be situations with no interaction with the phone but since the phone is being carried by the mobile user the accelerometer registers movement. In other situations, the accelerometer registers no movement at all. The features used are: the minimum, maximum, average and standard deviation of the 3 axis accelerometer vector norm captured during the whole visit period and during the last 10 minutes.

Frequency Tables. Finally for situations where the amount of data of a given user is scarce, we keep a simple frequency table of the most frequent destination and the most frequent destination given the possible temporal features.

6.3 Techniques

In this work, we evaluated different classification techniques to compare across the different models built. In our research, we have employed WEKA, a popular

suite of data mining software, to benchmark different classification techniques. We have also explored Massive Online Analysis (MOA) - an open-source framework for data stream mining written in Java. Related to the WEKA project, it includes a collection of machine learning algorithms and evaluation tools (e.g., prequential-error) particular to data stream learning problems.

Using Weka. We performed evaluation of the predictive accuracy using the validation set on the training data. We decided to explore several classification algorithms and our preliminary results indicated a slightly superior performance of the J48 algorithm for decision tree induction. However, our understanding while working on this problem is that the quality of the instances (i.e., observations) and features that describe them are the most important factors to achieve high predictive accuracy. Consequently, we studied feature selection and instance weighting.

- Feature Selection - As final step after feature engineering, we performed feature selection. This involved using well-known techniques for identifying/ranking which features have the best ability to predict the next location based on the subject's current location.
  We select dynamically for each user the best features (out of the 70 that we constructed/used) using two well-known feature selection techniques from the WEKA. First, information gain and second, cross-validated best feature subset evaluation (CfsSubsetEval). Therefore, the set of features that is selected for each user is different according to their productiveness for that given user/context.
- Instance Weighting - Another issue that we are faced in next place prediction is the quality of the observations, this is, the uncertainty associated with them (due to sensor reading uncertainty/unavailability) and also how relevantly they represent the user mobility patterns. Since the data has information about the uncertainty (a flag associated with a trusted visit, start and end time), we decided to explore this information and perform instance weighting in function of the confidence for their trusted start and end time.

Using MOA. In MOA we preliminarily explored different algorithms and obtained good results with Hoeffding Trees. Because of the data evolution issue described in this paper, we decided to experiment with a drift detection technique (SingleClassifierDrift). This algorithm implements a well known drift detection method proposed in [3]. Because of the good preliminary results with Hoeffding Trees it was used as the base learner parameter, using other Hoeffding tree classifier variations we obtained similar results.

6.4 Results and Discussion

The results presented in this section measure the accuracy on the validation set. This allowed us to compare our approach to existing published results. In Figure 2, *uid* stands for user ID and it does not run in sequence. We can observe that the accuracy for each user on task can have high variance. The results also showed

that some more irregular users are very difficult to predict while for some regular users is possible to achieve more than 80% accuracy. We should also note that the some results are biased negatively as these users have a very short history of visits.

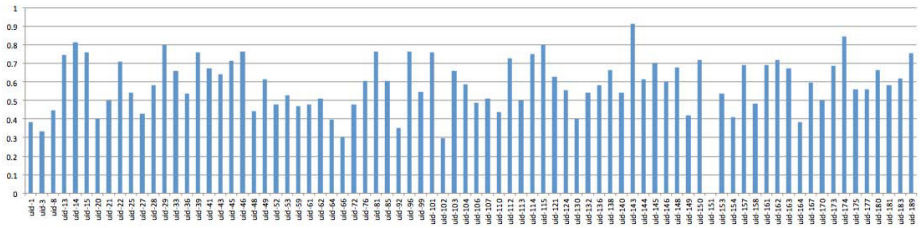


Fig. 2. Predictive accuracy across users

Feature Selection. In Table 1, we can see the results of our experiments with feature selection. We can observe that in general most features seem relevant to the next place prediction. Consequently, since the feature selection process was performed per user we make sure that the selection process was personalised. We can see that keeping almost all the 70 features (92%) seems to give the best results. If the set is reduced further a minor decrease in predictive performance is experienced. This finding is interesting as it shows that our effort to build sophisticated features can bring additional predictive accuracy to next place prediction. As feature selection is not so useful for our already predictive features, the results from the subsequent experiments use all the features (without any feature selection).

Table 1. Accuracy with Feature selection and Instance Weighting

NFeat%	35%	71%	85%	92%	100%
Accuracy	58.1%	58.8%	59.51%	**59.54%**	59.4%
$Weight_T$	0.0	0.25	0.50	0.75	1.0
Accuracy	57.7%	59.2%	59.3%	59.1%	**59.4%**
$Weights_S$	0.0	0.0	0.5	0.5	0.7
$Weight_E$	0.0	0.5	0.5	0.7	0.5
Accuracy	58.5%	59.4%	59.57%	59.48%	**59.6%**

Instance Weighting. In Table 1, we can see the results of our experiments with instance weighting based on trusted transition. The weight assigned to the instances in case they belong to a trusted transition is determined by $Weight_T$. We can observe that the results are similar among the experiments that still consider the trusted transitions. However, not including weights for trusted transition instances ($Weight_T = 0.0$) will have a significant impact on performance. This can be a consequence of the high number of untrusted transitions (42% of the data or 21356 visits) - in general, more data will be helpful.

In Table 1, different weights are assigned to the transactions based on the trusted start/end flag. Again not including untrusted transitions affects the performance as less instances are available for training. The combination of weights on trusted start time $Weight_S = 0.7$ and weights on trusted end time $Weight_E = 0.5$ gives the best overall results on the validation dataset.

Table 2. Comparision with Nokia MDC results

Method	Validation	Competition
ANN [1]	*60.83%**	56.22%
SVM [18]	55.69%	52.83%
HPHD [5]	50.53%	52.42%
Ensemble [11]	55.3%	-
DecisionTree [17]	*61.11%**	-
OurBest	**59.6%**	-

Comparing with Nokia MDC Best Results Here we compare our best results (OurBest uses all 70 features, has $Weight_T = 1$, $Weight_S = 0.7$, and $Weight_E = 0.5$) with the best results published. Table 2 summarises the best predictive accuracies of 5 other methods (the first three from winning teams). We should note that the ones with asterisk (*) indicate that the reported predictive accuracy was using a different evaluation strategy, and their results have likely over-fitted the training data. This happens in results with a Artificial Neural Network (ANN) proposed in [1] (60.83% in validation set with a significantly lower 56.22% in competition's test set) and in the J48 DecisionTree approach proposed in [17], where the authors use their own test set as opposed to the proposed validation set for the Nokia MDC.

From the results that are comparable (without the asterisk) we can see that our best results achieve the highest accuracy. This may be due to the large effort put in feature generation as not a big difference was observed among different techniques.

Online Learning. Here we report experiments with MOA using the SingleClassifierDrift algorithm. Evaluating for the same validation set we obtained an average accuracy of 42.22%. Again for some more predictable users it was possible to get close to 80% while for one user was not possible to predict anything. When we compare the results with our best batch results in Figure 2, the batch approach achieves better accuracy overall but for a small number of users the results are better with the incremental approach. The batch approach is on average (per user) 17% better than the online approach. Still, when comparing the accuracy with the published results for this dataset, the online approach is still very competitive.

In future work, based on lessons learnt here and our existing work in recurring concepts [9,8], we plan to develop our own online method for the next place prediction problem.

7 Conclusions and Future Work

In this paper we propose the *NextLocation* framework, a mobile data mining approach to the next place prediction problem. The main advantage of *NextLocation* is that it is a privacy-preserving solution that fully runs on the mobile device itself. Sensitive data about the user locations and context are not disclosed. Moreover, *NextLocation* uses an adaptive anytime model which enables adaptation to changes in the user mobility patterns. Finally, it keeps an estimate of the anytime model accuracy in real-time.

This paper also reports on our experiments analysing data from the Nokia Mobile Data Challenge (MDC). The results on MDC data show great variability in predictive accuracy across users, where irregular users are very difficult to predict while for more regular users it is possible to achieve more than 80% accuracy. To the best of our knowledge, our approach achieves the highest predictive accuracy when compared with existing results.

In future work, in line with the last experiments on online learning conducted in this work we plan to develop an online algorithm particularly designed for next place prediction.

References

1. Etter, V., Kafsi, M., Kazemi, E.: Been there, done that: What your mobility traces reveal about your behavior (2012)
2. Fox, V., Hightower, J., Liao, L., Schulz, D., Borriello, G.: Bayesian filtering for location estimation. IEEE Pervasive Computing 2(3), 24–33 (2003)
3. Gama, J., Medas, P., Castillo, G., Rodrigues, P.: Learning with drift detection. In: Bazzan, A.L.C., Labidi, S. (eds.) SBIA 2004. LNCS (LNAI), vol. 3171, pp. 286–295. Springer, Heidelberg (2004)
4. Gama, J., Sebastiao, R., Rodrigues, P.P.: Issues in evaluation of stream learning algorithms. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 329–338. ACM, New York (2009)
5. Gao, H., Tang, J., Liu, H.: Mobile location prediction in spatio-temporal context
6. Gomes, J., Krishnaswamy, S., Gaber, M.M., Sousa, P.A., Menasalvas, E.: Mars: a personalised mobile activity recognition system. In: 2012 IEEE 13th International Conference on Mobile Data Management (MDM), pp. 316–319. IEEE (2012)
7. Gomes, J.B., Krishnaswamy, S., Gaber, M.M., Sousa, P.A.C., Menasalvas, E.: Mobile activity recognition using ubiquitous data stream mining. In: Cuzzocrea, A., Dayal, U. (eds.) DaWaK 2012. LNCS, vol. 7448, pp. 130–141. Springer, Heidelberg (2012)
8. Gomes, J.B., Menasalvas, E., Sousa, P.A.: Learning recurring concepts from data streams with a context-aware ensemble. In: Proceedings of the 2011 ACM Symposium on Applied Computing, pp. 994–999. ACM (2011)
9. Gomes, J.B., Sousa, P.A., Menasalvas, E.: Tracking recurrent concepts using context. Intelligent Data Analysis 16(5), 803–825 (2012)
10. Laurila, J.K., Gatica-Perez, D., Aad, I., Blom, J., Bornet, O., Do, T.M.T., Dousse, O., Eberle, J., Miettinen, M.: The mobile data challenge: Big data for mobile computing research. In: Mobile Data Challenge by Nokia Workshop, in conjunction with Int. Conf. on Pervasive Computing, Newcastle, UK (2012)

11. Lu, Z., Zhu, Y., Zheng, V.W., Yang, Q.: Next place prediction by learning with multiple models
12. Mathew, W., Raposo, R., Martins, B.: Predicting future locations with hidden markov models (2012)
13. Monreale, A., Pinelli, F., Trasarti, R., Giannotti, F.: Wherenext: a location predictor on trajectory pattern mining. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 637–646. ACM (2009)
14. Pan, R., Zhao, J., Zheng, V.W., Pan, J.J., Shen, D., Pan, S.J., Yang, Q.: Domain-constrained semi-supervised mining of tracking models in sensor networks. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1023–1027. ACM (2007)
15. Sherchan, W., Jayaraman, P.P., Krishnaswamy, S., Zaslavsky, A., Loke, S., Sinha, A.: Using on-the-move mining for mobile crowdsensing. In: 2012 IEEE 13th International Conference on Mobile Data Management (MDM), pp. 115–124. IEEE (2012)
16. Spaccapietra, S., Parent, C., Damiani, M.L., De Macedo, J.A., Porto, F., Vangenot, C.: A conceptual view on trajectories. Data & Knowledge Engineering 65(1), 126–146 (2008)
17. Tran, L.H., Catasta, M., McDowell, L.K., Aberer, K.: Next place prediction using mobile data
18. Wang, J., Prabhala, B.: Periodicity based next place prediction

An Extended Local Hierarchical Classifier for Prediction of Protein and Gene Functions

Luiz Henrique de Campos Merschmann[1] and Alex Alves Freitas[2]

[1] Federal University of Ouro Preto, Computer Science Department, Brazil
luizhenrique@iceb.ufop.br
[2] University of Kent, School of Computing, UK
a.a.freitas@kent.ac.uk

Abstract. Gene function prediction and protein function prediction are complex classification problems where the functional classes are structured according to a predefined hierarchy. To solve these problems, we propose an extended local hierarchical Naive Bayes classifier, where a binary classifier is built for each class in the hierarchy. The extension to conventional local approaches is that each classifier considers both the parent and child classes of the current class. We have evaluated the proposed approach on eight protein function and ten gene function hierarchical classification datasets. The proposed approach achieved somewhat better predictive accuracies than a global hierarchical Naive Bayes classifier.

Keywords: Hierarchical Classification, naive Bayes, Bioiformatics, Protein Function Prediction.

1 Introduction

Classification is a well-known data mining task, where the algorithm builds, from the training set, a classifier that is used to predict the class labels of instances in the test set. A very active research area in bioinformatics consists of using classification methods to predict protein and gene functions. Although several sequencing genome projects have generated the full genome sequence of many organisms in the last decades, the functions of many proteins and genes still remain unknown. This is because, in general, determining the functions of genes and proteins is much more difficult and time-consuming than finding out their sequences.

A popular approach for biologists to infer new protein/gene functions is to use techniques that perform a similarity search in a protein/gene database containing proteins/genes with known functions. Basically, these techniques compute the similarity between the sequence of a protein/gene with unknown function and the sequences of the proteins/genes in a database. Thus, the new protein/gene is assigned to the class of its most similar protein(s)/gene(s) in the database [1].

Nevertheless, similarity-based protein/gene function prediction methods have some limitations. First, it is known that proteins/genes with similar sequences

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 159–171, 2013.

can have different functions [2]. Second, function prediction based only on sequence similarity does not consider many relevant biochemical properties of proteins/genes [3].

Aiming at solving these limitations, several works have proposed approaches that consist of inducing classification models from protein/gene data, where each protein/gene is represented by a set of attributes, and the new proteins/genes are classified by the induced model. This approach, which was adopted in this work, give us the opportunity to use a variety of classification methods for the protein/gene function prediction.

Most classification methods can deal only with flat classification problems, where there are no hierarchical relationships among the classes. However, in many problems the classes are naturally organized into hierarchies. Hierarchical classification problems are particularly common in the prediction of protein and gene functions, where the classes (functions) to be predicted are arranged in a tree or DAG (Direct Acyclic Graph) structure.

The prediction of protein and gene functions is challenging, mainly because there are usually hundreds or thousands of classes in the hierarchy and the class distribution is usually highly skewed, i.e., different class labels occur with very different frequencies. To simplify the problem, some works have just ignored the hierarchical class structure and addressed this problem as a traditional flat classification problem [4,5,6]. However, such works lose valuable information about parent-child class relationships, which is avoided by using a hierarchical classification method. One such method is described in [7], where the authors evaluated two hierarchical classification methods, based on a global and local version of a hierarchical Naive Bayes classifier - with the global version obtaining better predictive accuracy on eight protein function prediction datasets.

In this work, we propose an extended local hierarchical Naive Bayes classifier that (unlike a conventional local approach) exploits parent-child relationships between the classes in order to build a binary classifier for each class in the hierarchy. Then, the results of these binary classifiers are combined to produce the final classification for an instance. We evaluated our proposal on the same eight protein datasets used in [7] and on other ten gene function datasets, and compared it against the global-model approach proposed in [7].

The remaining of this paper is organized as follows. Section 2 presents an overview on hierarchical classification. In Section 3, we describe the hierarchical classifier proposed in this work. Section 4 presents the experimental setup and reports the results obtained in the comparative experiment. Finally, the conclusion and directions for future work are described in Section 5.

2 Hierarchical Classification

Hierarchical classification methods can be analyzed according to different aspects. The first one regards the type of hierarchical structure (tree or DAG) the method is able to deal with. This structure represents the relationships between the classes of the problem to be solved. Basically, in a tree each class is associated

with at most one parent class, while in a DAG a child class can have multiple parent classes.

The second aspect is how deep in the hierarchy the classification is done. A method can either perform mandatory leaf node predictions, where each instance must be assigned classes at leaf nodes in the class hierarchy; or non-mandatory (optional) leaf node predictions, where the most specific class assigned to an instance can be any (internal of leaf) class node in the hierarchy.

The third aspect refers to the number of different path labels in the hierarchy a method can assign an instance to. A method can be able to predict, to each instance, multiple paths of labels in the class hierarchy, or be restricted to predict just a single path of labels.

The fourth aspect concerns how the hierarchical structure is handled by the method. Three different approaches are presented in the literature: flat classification, which ignores the class hierarchy and performs predictions considering only the leaf node classes; local model approaches, when a set of local models are employed; and global model approaches, when a single classification model is built considering the class hierarchy as a whole during a single run of the classification algorithm. Since flat classification is out of the scope of this work, only local model approaches and global model approach are discussed next.

2.1 Local Classification Approach

In this approach, multiple classifiers are built, each one with a local view of the problem. Note that the class hierarchy is taken into account through a local perspective. Based on the different ways of using the local information, the classifiers can be grouped into three different categories [8]: local per node approach, local per parent node approach and local per level approach.

The local per node approach creates one binary classifier for each class node in the hierarchy (except the root node). Each classifier predicts whether or not an instance belongs to its corresponding class. The dashed rectangles in Figure 1(a) represent the classifiers. Note that this approach allows an instance to be assigned to classes in distinct branches in the hierarchy, which can lead to a class-membership inconsistency. To avoid that, several inconsistency removal methods are available [9,10,11].

In the local per parent node approach, a multi-class classifier is trained for each parent node in the hierarchy aiming at distinguishing between its child nodes. This approach is often used with a top-down prediction strategy when classifying new test instances. To illustrate this strategy, consider the hierarchy in Figure 1(b), where each dashed rectangle represents a classifier used to predict one of the child class nodes related to that classification node. Suppose a new test instance is assigned the class 1 by the root node classifier. Then, at the first hierarchy level, the classifier related to class node 1 will assign to this instance one of the child classes (1.1 or 1.2) of that node, and so on, until the instance is classified at the deepest appropriated level.

Finally, the local per level approach consists of training a multi-class classifier for each level of the class hierarchy. This is the hierarchical approach least used

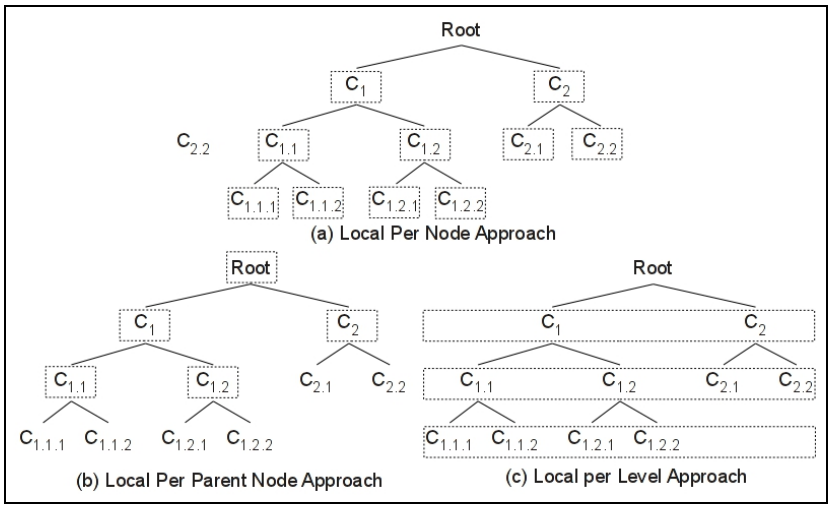


Fig. 1. Types of hierarchical classification approaches

in the literature [12]. Its major disadvantage is to be prone to class-membership inconsistency. For example, using the class hierarchy shown in Figure 1(c), three classifiers would be trained, one for each hierarchy level (represented by dashed rectangles). Then, given a instance to be classified, it is possible to have the following predictions: class 1 at level 1, class 2.1 at level 2 and class 1.1.2 at level 3. Clearly, the predicted class 2.1 is not consistent with the classes 1 and 1.1.2. Hence, this kind of approach requires a post-processing procedure to correct inconsistent predictions.

2.2 Global Classification Approach

Instead of creating a set of classifiers, the global approach involves the training of a single classifier taking into account the class hierarchy as a whole. Then, given a new instance to be classified, the induced classifier is able to assign it a class from any level of the hierarchy.

While the local approach with the top-down class prediction strategy has the disadvantage of propagating a classification mistake at a given level of the hierarchy through all its deeper levels, the global approach avoids that drawback by performing the classification in a single step using a single classifier.

It is worth noting that the global approach lacks the modular nature of the local approach, i.e., the characteristic of dividing the training phase in different processes, each of them considering part of the class hierarchy. Therefore, the single classifier built by the global approach tends to be more complex than each individual classifier produced by local approaches. However, this modular nature of the local approach does not imply that they will have better predictive accuracy than global approaches.

In this work we propose an extended local hierarchical Naive Bayes classifier based on the local per node approach (as described in the next section) and compare it against a global classification approach.

3 The Proposed Hierarchical Classifier

The proposed classifier deals with hierarchical classification problems where the classes to be predicted are disposed in a tree-based structure, in the scenarios of mandatory leaf node prediction and prediction of a single path in the class hierarchy. The main goal is to exploit parent-child relationships between the classes when building a binary classifier for each class in the hierarchy. Then, the predictions made by the set of binary classifiers are combined in order to produce a consistent prediction.

The proposed classifier, named Extended Local Hierarchical Naive Bayes (ELHNB), is based on the local per node approach, creating one binary classifier for each node of the class hierarchy. The training of each classifier considers not only the local information related to each classification node as usual, but also information about the relationships between each class node and its parent and child nodes – where the latter type of information is the proposed extension. Note that, since our method is based on Naive Bayes, we make the assumption of class conditional independence, that is, the attributes are conditionally independent of one another given the class attribute.

Let $D = \{\boldsymbol{d^1}, \ldots, \boldsymbol{d^t}\}$ be a set of training instances. Each instance $\mathbf{d^j}$, $j = 1, \ldots, t$, is represented by its attribute vector $\boldsymbol{X^j} = \{x_1^j, x_2^j, \ldots, x_n^j\}$ and is associated with a binary class vector $\boldsymbol{C^j} = \{c_1^j, c_2^j, \ldots, c_m^j\}$, where n is the number of predictor attributes and m is the number of classes in the hierarchy. Each c_i^j is assigned the value 1 if the instance d^j is associated with the class C_i, and 0 otherwise.

We train a Naive Bayes classifier to predict the class label vector $\boldsymbol{C} = \{c_1, c_2, \ldots, c_m\}$ for a new instance $\boldsymbol{X} = \{x_1, x_2, \ldots, x_n\}$, i.e., to learn a predictor $f : \boldsymbol{X} \mapsto [c_1, c_2, \ldots, c_m]$. The proposed approach has two phases. The first one is a local classification phase, where we train a binary classifier for each class. Each binary classifier performs a probabilistic classification for each class C_i, i.e., it computes $P(C_i = 1|\boldsymbol{X})$ and $P(C_i = 0|\boldsymbol{X})$. In the second phase we use the probabilities $P(C_i|\boldsymbol{X})$ calculated in the first phase to generate a consistent class vector prediction for the instance $\boldsymbol{X}$.

Recall that the classes are structured into a tree. Given a class C_i, the set of nodes formed by its parent and child nodes, termed neighbors of C_i, is represented by $\boldsymbol{N_i}$. The labels of the nodes (classes) contained in $\boldsymbol{N_i}$ are coded through a vector $\boldsymbol{Y_i} = \{y_1, y_2, \ldots, y_k\} \in \{0, 1\}^{k_i}$, where k_i is the number of neighbors of C_i. In the local classification phase, for each C_i, we compute $P(C_i = c_i)$, $c_i \in \{0, 1\}$, taking into account the relationships between the class C_i and its

parent and child nodes in the hierarchy. It can be determined by computing the following marginal probability:

$$P(C_i = c_i) = \sum_{\boldsymbol{Y_i} \in \{0,1\}^{k_i}} P(C_i = c_i | \boldsymbol{Y_i}) \times P(\boldsymbol{Y_i}) \tag{1}$$

Thus, $P(C_i = c_i|\boldsymbol{X})$ is obtained conditioning Equation 1 on instance $\boldsymbol{X}$ as follows:

$$P(C_i = c_i|\boldsymbol{X}) = \sum_{\boldsymbol{Y_i} \in \{0,1\}^{k_i}} P(C_i = c_i|\boldsymbol{X}, \boldsymbol{Y_i}) \times P(\boldsymbol{Y_i}|\boldsymbol{X}) \tag{2}$$

Applying Bayes' theorem on each term of the product in Equation 2:

$$P(C_i = c_i|\boldsymbol{X}, \boldsymbol{Y_i}) = \frac{P(\boldsymbol{X}|C_i = c_i, \boldsymbol{Y_i}) \times P(C_i = c_i|\boldsymbol{Y_i})}{\sum_{c_i \in \{0,1\}} P(\boldsymbol{X}|C_i = c_i, \boldsymbol{Y_i}) \times P(C_i = c_i|\boldsymbol{Y_i})} \tag{3}$$

and

$$P(\boldsymbol{Y_i}|\boldsymbol{X}) = \frac{P(\boldsymbol{X}|\boldsymbol{Y_i}) \times P(\boldsymbol{Y_i})}{\sum_{\boldsymbol{Y_i} \in \{0,1\}^{k_i}} P(\boldsymbol{X}|\boldsymbol{Y_i}) \times P(\boldsymbol{Y_i})} \tag{4}$$

Substituting Equations 3 and 4 into Equation 2, we have:

$$P(C_i = c_i|\boldsymbol{X}) = \sum_{\boldsymbol{Y_i} \in \{0,1\}^{k_i}} \left(\frac{P(\boldsymbol{X}|C_i = c_i, \boldsymbol{Y_i}) \times P(C_i = c_i|\boldsymbol{Y_i})}{\sum_{c_i \in \{0,1\}} P(\boldsymbol{X}|C_i = c_i, \boldsymbol{Y_i}) \times P(C_i = c_i|\boldsymbol{Y_i})} \times \frac{P(\boldsymbol{X}|\boldsymbol{Y_i}) \times P(\boldsymbol{Y_i})}{\sum_{\boldsymbol{Y_i} \in \{0,1\}^{k_i}} P(\boldsymbol{X}|\boldsymbol{Y_i}) \times P(\boldsymbol{Y_i})} \right) \tag{5}$$

As $\sum_{\boldsymbol{Y_i} \in \{0,1\}^{k_i}} P(\boldsymbol{X}|\boldsymbol{Y_i}) \times P(\boldsymbol{Y_i}) = P(\boldsymbol{X})$, we can rewrite Equation 5 as:

$$P(C_i = c_i|\boldsymbol{X}) = \sum_{\boldsymbol{Y_i} \in \{0,1\}^{k_i}} \left(\frac{P(\boldsymbol{X}|C_i = c_i, \boldsymbol{Y_i}) \times P(C_i = c_i|\boldsymbol{Y_i})}{\sum_{c_i \in \{0,1\}} P(\boldsymbol{X}|C_i = c_i, \boldsymbol{Y_i}) \times P(C_i = c_i|\boldsymbol{Y_i})} \times \frac{P(\boldsymbol{X}|\boldsymbol{Y_i}) \times P(\boldsymbol{Y_i})}{P(\boldsymbol{X})} \right) \tag{6}$$

Given that $P(\boldsymbol{X})$ is constant for all $\boldsymbol{Y_i}$ vector configurations, rearranging Equation 6 we get:

$$P(C_i = c_i|\boldsymbol{X}) = \frac{\sum_{\boldsymbol{Y_i} \in \{0,1\}^k} \left(\frac{P(\boldsymbol{X}|C_i = c_i, \boldsymbol{Y_i}) \times P(C_i = c_i|\boldsymbol{Y_i}) \times P(\boldsymbol{X}|\boldsymbol{Y_i}) \times P(\boldsymbol{Y_i})}{\sum_{c_i \in \{0,1\}} P(\boldsymbol{X}|C_i = c_i, \boldsymbol{Y_i}) \times P(C_i = c_i|\boldsymbol{Y_i})} \right)}{P(\boldsymbol{X})} \tag{7}$$

In Equation 7, aiming at reducing the number of parameters in evaluating $P(\boldsymbol{X}|C_i = c_i, \boldsymbol{Y_i})$ and $P(\boldsymbol{X}|\boldsymbol{Y_i})$, we use the Naive Bayes assumption that there are no dependence relationships among the attributes given the class. Then, these probabilities are computed as $P(\boldsymbol{X}|C_i = c_i, \boldsymbol{Y_i}) = \prod_{k=1}^{n} P(x_k|C_i = c_i, \boldsymbol{Y_i})$ and $P(\boldsymbol{X}|\boldsymbol{Y_i}) = \prod_{k=1}^{n} P(x_k|\boldsymbol{Y_i})$.

In the second phase, named global classification phase, we enforce hierarchical consistency of class labels using the probabilities $P(C_i = c_i|\boldsymbol{X})$ computed in the first phase. In order to obtain a consistent classification, for each possible path p in the hierarchy from *Root* node to node i, we compute the geometric average of probabilities $P(C_i = 1|\boldsymbol{X})$ along the path as follows:

$$GA_p = \sqrt[|L_p|]{\prod_{C_i \in L_p} P(C_i = 1|\boldsymbol{X})}, \tag{8}$$

where L_p is the set of classes in the path p.

As final solution, the instance $\boldsymbol{X}$ is assigned to the class vector $\boldsymbol{C}$ where the classes C_i contained in the path with the highest GA are set to 1 and the remaining to 0.

4 Computational Experiments

4.1 Baseline Method

Since we are proposing an extended local hierarchical Naive Bayes classifier, we use as a baseline method the global hierarchical Naive Bayes classifier proposed in [7], which achieved promising predictive performance when evaluated on eight protein datasets and outperformed a conventional local per parent node hierarchical classifier. In this conventional local hierarchical classifier, during the training phase, for each non-leaf node, a Naive Bayes multi-class classifier is trained to discriminate among the child class nodes of the classifier's corresponding node. Next, to implement the test phase, the top-down class prediction strategy is adopted.

The hierarchical classifier in [7] is an extension of the flat classification algorithm Naive Bayes to deal with hierarchical classification problems.

Given a new instance $X = \{x_1, x_2, \ldots, x_n\}$ to be classified, where each x_k refers to the value of attribute A_k, the flat Naive Bayes classifier simply assigns to it the class C_i associated with the maximum value of the posterior probability calculated as $P(C_i|X) \propto \prod_{k=1}^{n} P(x_k|C_i) \times P(C_i)$.

To explain how the hierarchical Naive Bayes [7] works, consider a tree-based hierarchy containing these class nodes: C_1, C_2, $C_{1.1}$, $C_{1.2}$, $C_{2.1}$ and $C_{2.2}$. To classify a new instance, the prior probabilities $P(C_1)$, $P(C_2)$, $P(C_{1.1})$, $P(C_{1.2})$, $P(C_{2.1})$ and $P(C_{2.2})$, and the likelihoods $P(x_k|C_1)$, $P(x_k|C_2)$, $P(x_k|C_{1.1})$, $P(x_k|C_{1.2})$, $P(x_k|C_{2.1})$ and $P(x_k|C_{2.2})$ are computed taking into account the class hierarchy, as follows. More precisely, to compute the prior probabilities and likelihoods during the training phase, the method takes into account that any instance which belongs to class C_i also belongs to all its ancestor classes. For example, if a training instance belongs to class $C_{1.2}$, that instance will be taken into account to compute the prior probabilities of both that class ($C_{1.2}$) and its ancestor classes (in this case, C_1). In addition, the attribute values of a training instance will be taken into account to compute the likelihoods associated with that instance's class and all its ancestor classes. These modification make the global hierarchical Naive Bayes able to predict classes at any level of the class hierarchy. For more details about this method, see [7].

4.2 Datasets

Experiments were conducted by running both the proposed and the baseline methods on 18 bioinformatics datasets, where eight are protein function and ten are gene function hierarchical classification datasets. As these datasets were obtained from different sources, we organized them into two groups.

Group A contains eight protein function datasets, referring to two different protein families: Enzymes and G-Protein-Coupled Receptors (GPCRs). Enzymes are proteins that catalyze chemical reactions [13] while GPCRs are transmembrane proteins that are the targets of many medical drugs [14]. We used four enzyme datasets (whose names start with EC – Enzyme Commission) and four GPCR datasets, where the predictor attributes correspond to protein properties and the classes to be predicted are hierarchical protein functions. Most predictor attributes are binary, indicating whether or not a protein signature (or motif) is present in a protein, but there are also two numeric attributes: the amino acid sequence length and molecular weight. The names of the datasets are also related to the type of motifs used: Interpro Entries, FingerPrints, Prosite Patterns and Pfam. These datasets[1] have also been used in previous hierarchical classification works [15], [16] and [7].

Group B contains ten gene function datasets, referring to the yeast genome. The predictor attributes include five types of bioinformatics data: secondary structure, phenotype, homology, sequence statistics, and expression. The classes to be predicted are taken from FunCat[2], a scheme for classifying the function of gene products developed by MIPS [17]. These datasets[3], initially presented in [18] and after updated and used in [19] were multi-label data, i.e., each instance was associated with one or more class paths in the hierarchy. Since in this work we are dealing with a single path label scenario, these datasets were converted into single label data by randomly choosing one class path for each instance.

Before running the classification algorithms, all datasets were preprocessed as follows: (a) All numeric attributes were converted into discrete ones by using an unsupervised discretization algorithm based on equal frequency binning (using 20 bins); (b) Every class with fewer than 10 instances was merged with its parent class. This process was repeated until every class in the hierarchy had at least 10 instances. If during this process the most specific class of an instance became the Root class, then that instance was removed; (c) Since in this work we are dealing with a mandatory leaf node prediction problem, after the previous step (b), we removed from the datasets all instances whose most specific class was not a leaf class node. Table 1 presents the main characteristics of the datasets after these pre-processing steps. This table shows, for each dataset, its number of attributes, number of instances and number of classes at each hierarchy level (1st/2nd/3rd/...). The pre-processed datasets used in our experiments are available at: http://www.decom.ufop.br/luiz/resources/.

4.3 Predictive Accuracy Evaluation Metrics

In order to evaluate the predictive accuracy of the hierarchical classifiers, we used the hierarchical F-measure, which is an adaptation of the flat F-measure tailored for hierarchical classification problems [20]. The hierarchical F-measure is computed as $hF = \frac{2 \times hP \times hR}{hP + hR}$, where hP and hR stand for hierarchical Precision

[1] https://sites.google.com/site/carlossillajr/resources
[2] http://mips.helmholtz-muenchen.de/proj/funcatDB/
[3] http://dtai.cs.kuleuven.be/clus/hmcdatasets/

Table 1. Characteristics of the Datasets

Group	Datasets	# Attributes	# Instances	# Classes/Level
A	GPCR-Pfam	75	6,524	12/52/79/49
	GPCR-Prosite	129	5,728	9/50/79/49
	GPCR-Prints	283	4,880	8/46/76/49
	GPCR-Interpro	450	6,935	12/54/82/50
	EC-Prints	382	11,048	6/45/92/208
	EC-Prosite	585	11,328	6/42/89/187
	EC-Pfam	708	11,057	6/41/96/190
	EC-Interpro	1,216	11,101	6/41/96/187
B	CellCycle	78	2,486	16/47/69/32/8
	Church	28	2,499	16/49/67/34/6
	Derisi	64	2,497	16/48/70/31/7
	Eisen	80	1,641	16/43/55/23/2
	Expr	552	2,554	16/49/68/28/5
	Gasch1	174	2,595	16/48/71/32/7
	Gash2	53	2,631	17/49/68/34/6
	Phenotype	70	1,023	15/43/40/15/1
	Sequence	479	2,689	17/48/65/29/5
	SPO	81	2,463	16/48/68/31/8

and hierarchical Recall, respectively. Considering that P_i is the set composed by the most specific class predicted for a test instance i and all its ancestor classes and T_i is the set composed by the true most specific class for a test instance i and all its ancestor classes, the hP and hR were defined in [20] as follows: $hP = \frac{\sum_i |P_i \bigcap T_i|}{\sum_i |P_i|}$ and $hR = \frac{\sum_i |P_i \bigcap T_i|}{\sum_i |T_i|}$.

Although these measures are recommended to evaluate hierarchical classification scenarios [8], there is a situation where their application faces a problem. Basically, hierarchical precision and hierarchical recall are measures related to the concepts of specialization and generalization errors, respectively. To illustrate these concepts, let us consider the following examples. Let C_1 be the most specific predicted class for an instance whose true most specific known class is $C_{1.3}$. In this case we have a generalization error, since the most specific class predicted is more generic than the true most specific known class for that instance. This generalization error is captured by the hierarchical recall measure, which for this example is $hR = 1/2$. Observe that in this case the hierarchical precision assumes the maximum value, i.e, $hP = 1$. On the other hand, if $C_{1.3}$ is the most specific predicted class for an instance whose true known class is C_1, we have a specialization error, as the predicted class is more specific than the true known class for that instance. Now, the hierarchical precision measure indicates this error ($hP = 1/2$), whilst the hierarchical recall measure assumes the maximum value $hR = 1$.

At first glance, hierarchical precision and hierarchical recall measures seem to penalize specialization and generalization errors appropriately. However, considering an over-specialization as an error (penalized by hP) can be unfair in some

kinds of applications, given that, if the true most specific known class for an instance is a more generic class like C_1, this does not mean that the prediction of the more specific class $C_{1.3}$ is an error. This may just mean that at present the more specific class of that instance is unknown, only its more generic class is currently known. Indeed, this kind of situation is relatively common in protein and gene function prediction, where more specific functional classes are often unknown and will be discovered later, with continuing advances in biological experiments that determine gene and protein functions.

Hence, we modified the definition of hP in order not to penalize over-specialized predictions. It is important to mention that even in mandatory leaf node prediction problems (the scenario considered here) over-specialized predictions can be made, since we can have leaf node classes at different levels in the hierarchy. Then, the adapted hP measure used in this work (which was used to measure the predictive accuracy of both hierarchical classification methods in our experiments) is defined as $hP = \frac{\sum_i |P_i \bigcap T_i|}{\sum_i min(|P_i|,|T_i|)}$, where $min(|P_i|, |T_i|)$ is the minimum value between $|P_i|$ and $|T_i|$.

4.4 Computational Results

As mentioned earlier, the Extended Local Hierarchical Naive Bayes classifier (ELHNB) was compared with the global-model Naive Bayes approach (GMNB) proposed in [7] on 18 bioinformatics datasets.

The performance of the hierarchical classifiers was measured by using the 10-fold cross validation [21] and the hierarchical F-measure (described in Section 4.3). The same ten folds in each dataset were used to evaluate the classifiers. In addition, for each dataset, in order to determine if there is a statistically significant difference between the hierarchical F-measure of the two hierarchical classifiers being compared, we used the Wilcoxon's Signed-Rank Test (two-sided test) as recommended by [22].

The results comparing the baseline GMNB with the proposed ELHNB are shown in Table 2. For each dataset, the third and fourth columns present the hierarchical F-measure values (hF) obtained by 10-fold cross validation (with the standard error in parentheses). The largest hierarchical F-measure value (hF) between those obtained by the two methods is in bold font. The last column presents the name of the method that achieved the best hF value when there is a statistically significant difference between the hF values of the two classifiers, or the symbol (–) to indicate that the difference between the hF values was not statistically significant.

In the eight datasets of Group A, the proposed ELHNB obtained significantly better results than GMNB for four datasets. In the remaining four datasets there was no statistically significant difference between the hF values of the two classifiers. In Group B, ELHNB outperformed GMNB in one dataset and GMNB outperformed ELHNB in another one. In the remaining datasets of this group, the difference of hF values between the classifiers was not statistically significant.

Overall, the proposed ELHNB reached results statistically equivalent or better than GMNB in 17 out of 18 datasets.

Table 2. Results of Comparative Experiment

Group	Data Sets	GMNB hF(std. error)	ELHNB hF(std. error)	Result of Statistical Test ($\alpha = 0.05$)
A	GPCR-Pfam	62.48 (0.31)	**62.99** (0.32)	–
	GPCR-Prosite	**61.58** (0.54)	61.44 (0.44)	–
	GPCR-Prints	79.66 (0.56)	**79.96** (0.51)	–
	GPCR-Interpro	80.44 (0.31)	**80.64** (0.36)	–
	EC-Prints	93.97 (0.24)	**94.99** (0.15)	**ELHNB**
	EC-Prosite	94.46 (0.08)	**96.84** (0.06)	**ELHNB**
	EC-Pfam	94.81 (0.14)	**95.42** (0.14)	**ELHNB**
	EC-Interpro	95.71 (0.12)	**96.33** (0.15)	**ELHNB**
B	CellCycle	12.45 (0.54)	**12.83** (0.57)	–
	Church	**10.65** (0.46)	10.61 (0.44)	–
	Derisi	10.42 (0.31)	**10.88** (0.58)	–
	Eisen	15.52 (0.87)	**16.52** (0.81)	**ELHNB**
	Expr	14.09 (0.73)	**14.63** (0.5)	–
	Gasch1	**15.31** (0.55)	14.20 (0.46)	**GMNB**
	Gash2	**15.70** (0.51)	15.42 (0.38)	–
	Phenotype	**8.08** (0.37)	7.22 (0.48)	–
	Sequence	**16.20** (0.79)	15.80 (0.76)	–
	SPO	**10.18** (0.52)	9.66 (0.59)	–

5 Conclusion

In this work, we proposed an extended local hierarchical Naive Bayes classifier based on a local per node approach, where a binary classifier is built for each class node in the hierarchy by exploiting the relationships between that class node and its parent and child nodes. The term "extended" is used to indicate that the proposed classifier extends the conventional local hierarchical approach by training each classifier with classes predicted for that classifier's neighbor (parent and child) nodes.

Different scenarios can be considered when dealing with hierarchical classification problems. In this paper we dealt with mandatory leaf node prediction problems, where the algorithm has to predict one of the leaf class nodes for each test instance. In addition, we focused on problems in which the classes to be predicted are disposed in a tree-based hierarchy and each data instance has a class label associated with a single path in this class hierarchy.

The evaluation of the proposed classifier was conducted on 18 bioinformatics datasets, where eight are protein function and ten are gene function hierarchical classification datasets.

Given the Bayesian nature of the proposed classifier, aiming at comparing it against a method of the same broad type, we used as a baseline method the global-model Naive Bayes approach proposed in [7]. In our experiments the proposed ELHNB classifier achieved predictive performance (measured by hierarchical F-measure) significantly better than the baseline GMNB method in 5

datasets, was significantly worse in only 1 dataset, and statistically equivalent in the remaining ones. Therefore, we conclude that the proposed extended local hierarchical Naive Bayes classifier has shown good predictive performance in the bioinformatics datasets used in this work, being somewhat more accurate than a global hierarchical classifier.

As future work, we intend to evaluate the ELHNB method in other hierarchical scenarios and compare it against other hierarchical classification approaches.

Acknowledgements. The first author is financially supported by CNPq - a Brazilian research-support agency (processes number 202120/2011-2 and 307711/2010-2). The authors also would like to thank the anonymous reviewers for their helpful comments.

References

1. Sleator, R.D., Walsh, P.: An overview of in silico protein function prediction. Archives of Microbiology 192(3), 151–155 (2010)
2. Gerlt, J.A., Babbitt, P.C.: Can sequence determine function? 1 (2000)
3. Syed, U., Yona, G.: Using a mixture of probabilistic decision trees for direct prediction of protein function. In: Proceedings of the Seventh Annual International Conference on Research in Computational Molecular Biology, RECOMB 2003, pp. 289–300. ACM, New York (2003)
4. Pavlidis, P., Cai, J., Weston, J., Noble, W.S.: Learning gene functional classifications from multiple data types. Journal of Computational Biology 9, 401–411 (2002)
5. Suhai, S., Glatting, K.H., Eils, R., Schubert, F., Moormann, J., König, R., Vinayagam, A.: Applying support vector machines for gene ontology based gene function prediction. BMC Bioinformatics 5 (2004)
6. Jung, J., Thon, M.R.: Automatic annotation of protein functional class from sparse and imbalanced data sets. In: Dalkilic, M.M., Kim, S., Yang, J. (eds.) VDMB 2006. LNCS (LNBI), vol. 4316, pp. 65–77. Springer, Heidelberg (2006)
7. Silla Jr., C.N., Freitas, A.A.: A global-model naive bayes approach to the hierarchical prediction of protein functions. In: Proc. of the 2009 Ninth IEEE International Conference on Data Mining, pp. 992–997. IEEE Computer Society (2009)
8. Silla Jr., C.N., Freitas, A.A.: A survey of hierarchical classification across different application domains. Data Mining and Knowledge Discovery 22(1-2), 31–72 (2011)
9. Wu, F., Zhang, J., Honavar, V.G.: Learning classifiers using hierarchically structured class taxonomies. In: Zucker, J.-D., Saitta, L. (eds.) SARA 2005. LNCS (LNAI), vol. 3607, pp. 313–320. Springer, Heidelberg (2005)
10. Barutcuoglu, Z., DeCoro, C.: Hierarchical shape classification using bayesian aggregation. In: Proc. of the IEEE International Conference on Shape Modeling and Applications, SMI 2006, p. 44 (2006)
11. Valentini, G.: True path rule hierarchical ensembles for genome-wide gene function prediction. IEEE/ACM Transactions on Computational Biology and Bioinformatics 8(3), 832–847 (2011)
12. Silla Jr., C.N.: Novel Approaches for Hierarchical Classification with Case Studies in Protein Function Prediction. PhD thesis, University of Kent (2011)
13. Grisham, C.M., Garrett, R.H.: Biochemistry. Saunders College Publishers, Philadelphia (1999)

14. Venkatakrishnan, A.J., Deupi, X., Lebon, G., Tate, C.G., Schertler, G.F., Babu, M.M.: Molecular signatures of g-protein-coupled receptors. Nature 494, 185–194 (2013)
15. Costa, E.P., Lorena, A.C., Carvalho, A.C.P.L.F., Freitas, A.A., Holden, N.: Comparing several approaches for hierarchical classification of proteins with decision trees. In: Sagot, M.-F., Walter, M.E.M.T. (eds.) BSB 2007. LNCS (LNBI), vol. 4643, pp. 126–137. Springer, Heidelberg (2007)
16. Holden, N., Freitas, A.A.: Improving the performance of hierarchical classification with swarm intelligence. In: Marchiori, E., Moore, J.H. (eds.) EvoBIO 2008. LNCS, vol. 4973, pp. 48–60. Springer, Heidelberg (2008)
17. Mewes, H.W., Heumann, K., Kaps, A., Mayer, K.F.X., Pfeiffer, F., Stocker, S., Frishman, D.: Mips: a database for genomes and protein sequences. Nucleic Acids Research 27(1), 44–48 (1999)
18. Clare, A., King, R.D.: Predicting gene function in saccharomyces cerevisiae. In: Proc. of the European Conference on Computational Biology, pp. 42–49 (2003)
19. Vens, C., Struyf, J., Schietgat, L., Džeroski, S., Blockeel, H.: Decision trees for hierarchical multi-label classification. Machine Learning 73(2), 185–214 (2008)
20. Kiritchenko, S., Matwin, S., Famili, A.F.: Functional annotation of genes using hierarchical text categorization. In: Proc. of the BioLINK SIG: Linking Literature, Information and Knowledge for Biology (2005)
21. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques, 3rd edn. Morgan Kaufmann Publishers, USA (2011)
22. Japkowicz, N., Shah, M.: Evaluating Learning Algorithms: A Classification Perspective. Cambridge University Press, New York (2011)

Supervised Dimensionality Reduction via Nonlinear Target Estimation

Josif Grabocka, Lucas Drumond, and Lars Schmidt-Thieme

Information Systems and Machine Learning Lab
Samelsonplatz 22, 31141 Hildesheim, Germany
{josif,ldrumond,schmidt-thieme}@ismll.uni-hildesheim.de

Abstract. Dimensionality reduction is a crucial ingredient of machine learning and data mining, boosting classification accuracy through the isolation of patterns via omission of noise. Nevertheless, recent studies have shown that dimensionality reduction can benefit from label information, via a joint estimation of predictors and target variables from a low-rank representation. In the light of such inspiration, we propose a novel dimensionality reduction which simultaneously reconstructs the predictors using matrix factorization and estimates the target variable via a dual-form maximum margin classifier from the latent space. The joint optimization function is learned through a coordinate descent algorithm via stochastic updates. Finally empirical results demonstrate the superiority of the proposed method compared to both classification in the original space (no reduction), or classification after unsupervised reduction.

Keywords: Machine Learning, Dimensionality Reduction, Feature Extraction, Matrix Factorization, Supervised Dimensionality Reduction.

1 Introduction

Dimensionality reduction is an important ingredient of machine learning and data mining. The benefits of projecting data to latent spaces constitute in (i) converting large dimensionality datasets into feasible dimensions, but also (ii) improving the classification accuracy of small and medium datasets [1]. Via carefully tuned dimensionality reduction (aka feature extraction) we are able to retrieve the necessary patterns from the datasets, by leaving out the noise. Traditional dimensionality reduction, (as described in Section 2.1), has been focused on extracting features prior to classification. Such a mentality has been recently found to perform non-optimal [2,3], since the features are not directly extracted/optimized for boosting classification. As a result there have been attempts to incorporate class supervision into feature extraction, (mentioned in Section 2.3), such that the latent features are guided to enforce the discernment/separation of instances belonging to opposite classes in the reduced space. Throughout this work we propose a principle, (details in Section 3.1), according to which dimensionality reduction should optimize the latent features through

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 172–183, 2013.

the same optimization function as the final classification method, thereby ensuring that the classification accuracy in the latent space is optimized. Inspired by the accuracy success of SVMs, that is significantly inherent to the kernel trick approach, we propose a novel supervised dimensionality reduction that incorporates kernel-based classification in the reduced dimension (Section 3). The novelty relies on defining a joint dimensionality reduction via matrix factorization, in parallel to a dual-form kernel-based maximum margin classification in the latent space. The reduced data is simultaneously updated in a coordinate descent fashion in order to optimize both loss terms. Experimental results, (Section 5), demonstrate the superiority of the proposed method compared to both unsupervised dimensionality reduction and classification in the original space. The main contribution of this work are:

1. Define a supervised dimensionality reduction with a kernel-based target variable estimation, in addition to the matrix reconstruction loss term
2. Derive a coordinate descent algorithm which simultaneously learns the latent factors for both loss terms
3. Provide empirical results to demonstrate the superiority of the method

2 Related Work

2.1 Dimensionality Reduction

Dimensionality reduction is a field of computer science that focuses on extracting lower dimensionality features from datasets [1]. Numerous techniques exist for extracting features. Principal Component Analysis (PCA) is a famous approach involving orthogonal transformations and selecting the topmost principal components, which preserve necessary variance [4]. Alternatively, Singular Value Decomposition decomposes a dataset into latent unitary, nonnegative diagonal and conjugate transpose unitary matrices [1].

Further elaborations of dimensionality reductions involve nonlinear decomposition of data [5]. For instance kernel PCA replaces the linear operations of PCA through nonlinear mapping in a reproducing kernel Hilbert space [6]. The whole subfield of manifold learning elaborates, as well, on nonlinear projections. Specifically, Sammon's mappings preserves the structure of instance distances in the reduced space [7], while principal curves embed manifolds using standard geometric projections [8]. More nonlinear dimensionality algorithms are described in [9]. In addition, temporal dimensionality reduction have been proposed in scenarios where the time difference of observations is not evenly spaced [10].

2.2 Matrix Factorization

Matrix factorization refers to a family of decompositions which approximates a dataset as a product of latent matrices of typically lower dimensions. A generalization and categorization of the various proposed factorization models as applications of Bregman divergences was elaborated in [11]. The learning of the

decomposition is typically conducted by defining a l2-norm and updating the latent matrices via a stochastic gradient descent [12]. Matrix factorization has been applied in a range of domains, ranging from recommender systems where decomposition focuses on collaborative filtering of sparse user-item ratings [13], up to time series dimensionality reduction [14].

2.3 Supervised Dimensionality Reduction

In addition to the standard dimensionality reduction and matrix factorization, there has been attempts to utilize the labels information, therefore dictating a supervised projection. Fisher's linear discriminant analysis is a popular supervised projection method [15]. The classification accuracy loss objective functions occurring in literature vary from label least square regression [16], to generalized linear models [17], linear logistic regression [2], up to hinge loss [3]. Another study aimed at describing the target variable as being conditionally dependent on the features [18]. Other families of supervisions aim at preserving the neighborhood structure of intra-class instances [19], or links in a semi supervised scenarios [20]. In comparison to the aforementioned, we propose a supervised dimensionality reduction with a kernel-based classifier, by directly optimizing the dual formulation in the projected space.

3 Proposed Method

3.1 Principle

The method proposed in this study relies on the principle that feature extraction, analogously referred also as dimensionality reduction, should not be conducted "ad-hoc" or via particular heuristics. Most of the classification tasks have a unifying objective, which is to improve classification accuracy. In that context we are referring as "ad-hoc" to the family of feature extraction techniques that don't directly optimize their loss functions for classification accuracy. Stated else-wise, we believe that instance labels should guide the feature extraction, such that the utilization of the extracted features improves accuracy. In that perspective, we propose a feature extraction method which operates by optimizing a joint objective function composed of the feature extraction term and also the classification accuracy term. Further details will be covered in the coming Section 3. In comparison with similar feature extraction ideas reviewed in Section 2.3, which use linear classifiers in the optimization, we propose a novel method which learns a nonlinear SVMs over the projected space via jointly optimizing a dual form together with dimensionality reduction.

3.2 Matrix Factorization as Dimensionality Reduction

Matrix factorization is a dimensionality reduction technique which decomposes a dataset $X \in \mathbb{R}^{(n+n')\times m}$ matrix of n training instances and n' testing instances, per m features, into two smaller matrices of dimensions $U \in \mathbb{R}^{(n+n')\times d}$ and

$V \in \mathbb{R}^{d \times m}$ [12]. The latent/reduced projection of the original data X is the latent matrix U, where d is the dimensionality of the projected space. Typically d is much smaller than m, meaning that the dimensionality is reduced. Such decomposition is expressed in form of a reconstruction loss, denoted $F_R(X, U, V)$ and depicted in Equation 1. The optimization aims at computing latent matrices U, V such that their dot product approximates the original matrix X via an Euclidean distance (l_2 norm) loss. In addition to the l_2 reconstruction norm, we also add l_2 regularization terms weighted by factors λ_U, λ_V in order to avoid over-fitting.

$$\underset{U,V}{\operatorname{argmin}} \; F_R(X, U, V) = ||X - UV||^2 + \lambda_U ||U||^2 + \lambda_V ||V||^2 \tag{1}$$

Bias terms, $B_U \in \mathbb{R}^{(n+n') \times 1}, B_V \in \mathbb{R}^{1 \times m}$ are added to the reconstruction loss [12], such that each element of B_U incorporates the prior belief value of the respective instance, while each element of B_V the prior belief value of the respective feature. More concretely the loss can be expanded as a reconstruction of each cell $X_{i,j}$ as depicted by Equation 2.

$$\begin{aligned} \underset{U,V,B_U,B_V}{\operatorname{argmin}} \; F_R(X, U, V) = & \sum_{i=1}^{n+n'} \sum_{j=1}^{m} \left(X_{i,j} - \left(\sum_{k=1}^{d} U_{i,k} V_{k,j} + B_{U_i} + B_{V_j} \right) \right)^2 \\ & + \lambda_U \sum_{i=1}^{n+n'} \sum_{k=1}^{d} U_{i,k}^2 + \lambda_V \sum_{k=1}^{d} \sum_{j=1}^{m} V_{k,j}^2 \end{aligned} \tag{2}$$

3.3 Kernel-Based Supervision of Dimensionality Reduction

Matrix factorization, as described in Section 3.2, is guided only by the reconstruction loss. Such approach doesn't take into consideration the classification accuracy impact of the extracted features, therefore the produced reduced dimensionality data is not optimized to improve accuracy. In order to overcome such a drawback, the so called supervised dimensionality reduction has been proposed by various authors [2]. The key commonalities of those supervised dimensionality methods rely on defining a joint optimization function, consisting of the reconstruction loss terms and the classification accuracy terms.

The typical classification accuracy loss term focuses on defining a classifier in the latent space, i.e. U, via a hyperplane defined by the weights vector W, such that the weights can correctly classify the training instances of U in order to match observed label Y. Equation 3 defines a cumulative joint optimization function using an introduced classification accuracy term, denoted $F_{CA}(Y, U, W)$. The trick of such a joint optimization constitutes on updating U simultaneously, in order to minimize both F_R and F_{CA} via gradient descent on both loss terms. The hyper parameter β is a switch which balances the impact of reconstruction vs classification accuracy.

$$F(X, Y, U, V, W) = \beta \, F_R(X, U, V) + (1 - \beta) \, F_{CA}(Y, U, W) \tag{3}$$

In comparison to previous approaches that propose linear models, in this study we propose a kernel-based **binary** classifier approach in the latent space U. Let us initially define the classification accuracy loss term, denoted $F_{CA}(Y, U, W)$, in Equation 4, in form of a maximum margin soft SVMs with hinge loss [21]. Such form of the SVMs is called the primal form. The parameter C scales the penalization of the instances violating the distances from the maximum margin. Please note that W_0 is the intercept bias term of the hyperplane weights vector W.

$$\begin{aligned} \underset{U,W}{\operatorname{argmin}} \;\; & F_{CA}(Y,U,W) = \frac{1}{2}||W||^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.t:} \;\; & Y_i(\langle W, U_i\rangle + W_0) \geq 1 - \xi_i, \;\; i = 1, ..., n \\ & \xi_i \geq 0, \;\; i = 1, ..., n \end{aligned} \tag{4}$$

Unfortunately the primal form doesn't support kernels, therefore we have to convert the optimization functions into the dual form equation 5. In order to get rid of of the inequality constraint we apply Lagrange multipliers to include the inequalities by introducing dual variables α_i per instance and adding $\alpha_i\left(y_i(\langle W, U_i\rangle + W_0)\right)$ to the optimization function for all instance i. Then we solve the objective function for W and W_0 by equating the first derivative to zero. Putting the derived expressions of W and W_0 to the objective function, we obtain the so-called dual representation optimization:

$$\begin{aligned} \underset{U,\alpha}{\operatorname{argmin}} \;\; & F_{CA}(Y,U,\alpha) = \frac{1}{2}\sum_{i=1}^{n}\sum_{l=1}^{n}\alpha_i\alpha_l Y_i Y_l \langle U_{i,*}, U_{l,*}\rangle - \sum_{i=1}^{n}\alpha_i \\ \text{s.t:} \;\; & 0 \leq \alpha_i \leq C, \;\; i = 1, ..., n \\ & \sum_{i=1}^{n}\alpha_i Y_i = 0 \end{aligned} \tag{5}$$

Once the optimization model is build any new test instance X_t can be classified in terms of learned α as shown in Equation 6.

$$Y_t = sgn\left(\sum_{i=1}^{n}\alpha_i Y_i \langle U_{i,*}, U_{t,*}\rangle + W_0\right) \tag{6}$$

The dot product, found in the dual formulation, between the instance vectors appears both in the optimization function 5 and the classification function 6. Such a dot product can be replaced by the so called kernel functions [21]. Various kernel representations exists, however in this study, for the sake of clarity and generality, we are going to prove the concept of the method using polynomial kernels, defined in Equation 7, which are known to be successful off-the-shelf kernels [21].

$$K(U_{i,*}, U_{l,*}) = \left(\sum_{k=1}^{d} U_{i,k}U_{l,k} + 1\right)^p \tag{7}$$

The ultimate objective function that defines nonlinear supervised dimensionality reduction is presented in Equation 8. *This model, in cooperation with the forthcoming learning algorithm, are the main contributions of our paper.*

$$\begin{aligned}\underset{U,V,\alpha,B_U,B_V}{\operatorname{argmin}} \; F(X,Y,U,V,\alpha) = \beta \sum_{i=1}^{n+n'} \sum_{j=1}^{m} \left(X_{i,j} - \left(\sum_{k=1}^{d} U_{i,k} V_{k,j} + B_{U_i} + B_{V_j} \right) \right)^2 \\ + (1-\beta) \left(\frac{1}{2} \sum_{i=1}^{n} \sum_{l=1}^{n} \alpha_i \alpha_l Y_i Y_l \, K(U_{i,*}, U_{l,*}) - \sum_{i=1}^{n} \alpha_i \right) \\ + \lambda_U \sum_{i=1}^{n+n'} \sum_{k=1}^{d} U_{i,k}^2 + \lambda_V \sum_{k=1}^{d} \sum_{j=1}^{m} V_{k,j}^2 \end{aligned} \tag{8}$$

$$\begin{aligned} \text{s.t:} \quad & 0 \le \alpha_i \le C, \;\; i = 1, ..., n \\ & \sum_{i=1}^{n} \alpha_i Y_i = 0 \end{aligned}$$

Meanwhile the classification of a test instance U_t using kernels and the learned U, α, resulting from the solution of the dual joint optimization is shown in Equation 9.

$$Y_t = sgn \left(\sum_{i=1}^{n} \alpha_i Y_i \, K(U_{i,*}, U_{t,*}) + W_0 \right) \tag{9}$$

4 Learning Algorithm via Coordinate Descent

The objective function of Equation 8 is a non-convex function in terms of U, V and W, which makes it challenging for optimization. However stochastic gradient descent is shown to perform efficiently in minimizing such non-convex functions [12]. The benefits of stochastic gradient descent relies on better convergence, because cells of X are randomly picked for optimization, thus updating different rows of U, instead of iterating through the all the features of the same instance, thus resulting in subsequent updates of the same latent row of U.

On the other side, the classification accuracy terms of Equation 5 can be solved, in terms of α, by any standard SVMs dual solver method in case we consider U to be fixed. Thus, in an alternating fashion we solve the α-s by keeping U fixed. Then in the next step we update U using the learned α-s and V matrix, by taking a step in the negative direction of the overall loss w.r.t U. The update of V is performed as last step. Those three steps can be repeated until convergence as shown in the Algorithm 2.

Before starting the description of the algorithm let us define the gradients to be used for updating our latent matrices. We can represent the reconstruction loss F_R as sum of smaller loss terms $F_{Ri,j}$, per each cell (i, j) of the original

dataset X. Such decomposition will later enable stochastic gradient descent to optimize for each small loss term stochastically/randomly.

$$F_R(X,U,V) = \sum_{i=1}^{n+n'} \sum_{j=1}^{m} F_R(X,U,V)_{i,j} \tag{10}$$

$$F_R(X,U,V)_{i,j} = \beta \left(X_{i,j} - \left(\sum_{k=1}^{d} U_{i,k} V_{k,j} + B_{U_i} + B_{V_j} \right) \right)^2 + \lambda_U \frac{1}{m} \sum_{k=1}^{d} U_{i,k}^2 + \lambda_V \frac{1}{n+n'} \sum_{k=1}^{d} V_{k,j}^2 \tag{11}$$

Gradients:

$$e_{i,j} = X_{i,j} - \sum_{k=1}^{d} U_{i,k} V_{k,j} - B_{U_i} - B_{V_j}$$

$$\frac{\partial F_R(X,U,V)_{i,j}}{\partial U_{i,k}} = -2\beta\, e_{i,j}\, V_{k,j} + 2\lambda_U \frac{1}{m} U_{i,k} \tag{12}$$

$$\frac{\partial F_R(X,U,V)_{i,j}}{\partial V_{k,j}} = -2\beta\, e_{i,j}\, U_{i,k} + 2\lambda_V \frac{1}{n+n'} V_{k,j} \tag{13}$$

$$\frac{\partial F_R(X,U,V)_{i,j}}{\partial B_{U_i}} = -2\beta\, e_{i,j} \tag{14}$$

$$\frac{\partial F_R(X,U,V)_{i,j}}{\partial B_{V_j}} = -2\beta\, e_{i,j} \tag{15}$$

Similarly, we can split up the classification accuracy loss term, F_{CA}, into smaller loss terms $F_{CAi,l}$, defined per each instance pair (i,l).

$$F_{CA}(Y,U,\alpha) = \sum_{i=1}^{n} \sum_{l=1}^{n} F_{CA}(Y,U,\alpha)_{i,l} \tag{16}$$

$$F_{CA}(Y,U,\alpha)_{i,l} = (1-\beta) \left(\frac{1}{2} \alpha_i \alpha_l Y_i Y_l\; K(U_{i,*}, U_{l,*}) - \frac{1}{n^2} \sum_{i=1}^{n} \alpha_i \right) \tag{17}$$

Gradients:

$$\frac{\partial F_{CA}(Y,U,\alpha)_{i,l}}{\partial U_{i,k}} = (1-\beta)\, \frac{1}{2} \alpha_i \alpha_l Y_i Y_l\; p \left(\sum_{k=1}^{d} U_{i,k} U_{l,k} + 1 \right)^{p-1} U_{l,k} \tag{18}$$

$$\frac{\partial F_{CA}(Y,U,\alpha)_{i,l}}{\partial U_{l,k}} = (1-\beta)\, \frac{1}{2} \alpha_i \alpha_l Y_i Y_l\; p \left(\sum_{k=1}^{d} U_{i,k} U_{l,k} + 1 \right)^{p-1} U_{i,k} \tag{19}$$

The updates of α-s is carried through an algorithm which is a reduced version of the Sequential Minimal Optimization (SMO) [22]. Since the dual form optimization function contains the constraint $\sum_{i=1}^{n} \alpha_i Y_i = 0$, then any update of an α_i will violate the constraint. Therefore SMO updates the α-s in pair, offering

Algorithm 1. UpdateAlphaPair

Input: First alpha index i, Second alpha index j
Output: Updated α and W_0

$(\alpha_i^{old}, \alpha_j^{old}) \leftarrow (\alpha_i, \alpha_j)$
Let $s \leftarrow Y_i Y_j$
$(L, H) \leftarrow \left(\max(0, \alpha_j^{old} + s\alpha_i^{old} - \frac{s+1}{2}C), \min(C, \alpha_j^{old} + s\alpha_i^{old} - \frac{s-1}{2}C)\right)$
$E_k \leftarrow \left(\sum_{l=0}^{n} Y_l \alpha_l K(U_{l,*}, U_{k,*}) + W_0\right) - Y_k, \forall k \in \{i, j\}$
$\alpha_j^{new} \leftarrow \alpha_j^{old} - \frac{Y_j(E_i - E_j)}{2K(U_{i,*}, U_{j,*}) - K(U_{i,*}, U_{i,*}) - K(U_{j,*}, U_{j,*})^1}$

$$\alpha_j^{new,clipped} = \begin{cases} L\,, & \text{if } \alpha_j^{new} < L \\ \alpha_j^{new}\,, & \text{if } L < \alpha_j^{new} < H \\ H, & \text{if } \alpha_j^{new} > H \end{cases}$$

$\alpha_i^{new} \leftarrow \alpha_i^{old} + s(\alpha_j^{new,clipped} - \alpha_j^{old})$
$b_i \leftarrow E_i + y_i(\alpha_i^{new} - \alpha_i^{old})K(U_{i,*}, U_{i,*}) + Y_2(\alpha_j^{new,clipped} - \alpha_j^{old})K(U_{i,*}, U_{j,*}) + W_0$
$b_j \leftarrow E_j + y_i(\alpha_i^{new} - \alpha_i^{old})K(U_{i,*}, U_{i,*}) + Y_2(\alpha_j^{new,clipped} - \alpha_j^{old})K(U_{i,*}, U_{j,*}) + W_0$
$\mathbf{W_0} \leftarrow \frac{\mathbf{b_i + b_j}}{\mathbf{2}}$, $(\alpha_\mathbf{j}, \alpha_\mathbf{i}) \leftarrow \left(\alpha_\mathbf{j}^{\mathbf{new,clipped}}, \alpha_\mathbf{i}^{\mathbf{new}}\right)$
return α, W_0

three heuristics which defines which subset of the pairs should be updates first, in order to speed up the algorithm.

In difference to the original algorithm, we have ignored the selection heuristic for the *alpha* pairs to update. The reason for omitting the heuristics is due to the fact that U instances are continuously updated/modified. For instance, let us consider an imaginary instance U_i far away from the decision boundary, which means $\alpha_i = 0$. However in the next iteration, the instance U_i might be updated and move close to the boundary, meaning that α_i becomes a candidate for being updated ($0 < \alpha_i \leq C$), opposite to the functioning of SMO heuristic that would have avoided updating the instance, alluding that α_i is still 0.

The alpha updates rely on solving the function analytically for a pair of α-s at a step, until no $\alpha_i, \forall i$, violates the KKT [22] conditions described in Equation 20.

$$\begin{aligned} \text{Let } \hat{Y_i} \;=\; sgn\left(\sum_{j=1}^{n} \alpha_j Y_j\, K(U_{j,*}, U_{i,*}) + W_0\right) \\ \alpha_i = 0 \rightarrow Y_i \hat{Y_i} \geq 1 \\ 0 < \alpha_i < C \rightarrow Y_i \hat{Y_i} = 1 \\ \alpha_i = C \rightarrow Y_i \hat{Y_i} \leq 1 \end{aligned} \tag{20}$$

Therefore the learning algorithm will update all the pairs of α-s in each iteration. The SMO-like update of each pair of alphas is shown in the Algorithm 4, with more details in [22]. Please note that the algorithm also updates the hyperplane intercept W_0, which is used for classification of latent instances.

Having defined the gradients for updating latent matrices U, V with respect to the optimization loss and also the update rules for α-s, we can derive a final

learning algorithm based on coordinate gradient descent. Algorithm 2 shows the learning algorithm in full terms. The updates of each cell of U, V, B_U, B_V, as response to the reconstruction loss F_R and the classification accuracy loss F_{CA}, are conducted in the negative direction of the gradients scaled by hyper-parameter learning rates η_R, η_{CA}. The convergence is guaranteed by selecting small values for the learning rates. The stopping criteria is when the final loss from Equation 8 reaches an optimum, meaning it doesn't get further minimized.

Algorithm 2. Learning Algorithm

Input: Dataset matrix $X \in \mathbb{R}^{(n+n')\times m}$, Labels vector $Y \in \mathbb{R}^n$, **Parameters:** { Box constraint C, Optimization switch β, Latent dimensions d, Learning rates η_R, η_{CA}, Regularizations λ_U, λ_V, Kernel degree p }
Output: $U, V, B_U, B_V, \alpha, W_0$
Initialize $U \in \mathbb{R}^{(n+n')\times d}$, $V \in \mathbb{R}^{d\times m}$, $B_U \in \mathbb{R}^{(n+n')\times 1}$, $B_V \in \mathbb{R}^{1\times m}$ randomly
Initialize $\alpha \leftarrow \{0\}^n$, $W_0 \leftarrow 0$
while F **not** reached an optimum **do**
  for $\forall(i,j,k) \in (\{1...(n+n')\}, \{1...m\}, \{1...d\})$ *in random order* **do**
    $U_{i,k} \leftarrow U_{i,k} - \eta_R \frac{\partial F_R(X,U,V)_{i,j}}{\partial U_{i,k}}$
    $V_{k,j} \leftarrow V_{k,j} - \eta_R \frac{\partial F_R(X,U,V)_{i,j}}{\partial V_{k,j}}$
    $B_{U_i} \leftarrow B_{U_i} - \eta_R \frac{\partial F_R(X,U,V)_{i,j}}{\partial B_{U_i}}$
    $B_{V_j} \leftarrow B_{V_j} - \eta_R \frac{\partial F_R(X,U,V)_{i,j}}{\partial B_{V_j}}$
  end for
  for $\forall(i,l,k) \in (\{1...n\}, \{1...n\}, \{1...d\})$ *in random order* **do**
    $U_{i,k} \leftarrow U_{i,k} - \eta_{CA} \frac{\partial F_{CA}(Y,U,\alpha)_{i,l}}{\partial U_{i,k}}$
    $U_{l,k} \leftarrow U_{l,k} - \eta_{CA} \frac{\partial F_{CA}(Y,U,\alpha)_{i,l}}{\partial U_{l,k}}$
  end for
  for $\forall i \in \{1 \dots n\}$ **do**
    if α_i violates KKT of Equation 20 **then**
      for $\forall j \in \{1 \dots n\}$ *in random order* **do**
        $(\alpha, W_0) \leftarrow$ UpdateAlphaPair(i,j), from Algorithm 4
      end for
    end if
  end for
end while
return $U, V, B_U, B_V, \alpha, W_0$

5 Experimental Results

In order to compare the classification accuracy of our method Nonlinearly Supervised Dimensionality Reduction (**NSDR**), we implemented and compared against two baselines:

- **PCA-SVMs:** Matching against the standard PCA dimensionality reduction and then SVMs classification will demonstrate the advantage of supervised decomposition against unsupervised decomposition (PCA).
- **SVMs:** Comparison against the default SVMs will provide insights on the advantages of dimensionality reduction.

The experiments were conducted using five folds cross validation, where the data was divided into five splits and each split was, in turn, the test and the other four the training data.

The hyper parameters of our method and the baselines was selected using a validation data split from the training data. The best grid-search combinations of hyper parameters that yielded the best accuracy was selected for being applied to the test split. The ranges of search for the NSDR method were $\lambda_U \in \{10^{-6}, 10^{-5}, \ldots, 10^{0}, 10^{1}\}, \lambda_V \in \{10^{-6}, 10^{-5}, \ldots, 10^{0}, 10^{1}\}, \eta_R \in \{10^{-4}, 10^{-3}\}, \eta_{CA} \in \{10^{-4}, 10^{-3}\}, d \in \{25\%, 50\%, 75\%, 100\%\}$ of $m, \beta \in \{0.1, 0.5, 0.9\}, C \in \{0.1, 1, 10\}, p \in \{1, 2, 3, 4\}$. For PCA-SVMs there is a variance parameter $var \in \{0.5, 0.7, 1.0\} \times 100\%$. The other SVMs parameters C, p for both PCA-SVMs and SVMs were searched in the same ranges as the ones reported for NSDR previously.

5.1 Results and Interpretation

For the sake of empirical verification we randomly selected five popular binary datasets from the UCI repository. The results of the hyper parameter search over the selected datasets are shown in Table 1.

Table 1. Hyper-parameter Search Results

DATASET	NSDR	PCA-SVMs	SVMs
breast_cancer_ wisconsin	$\lambda_U = 10^{-5}; \lambda_U = 10^{-1}; \eta_R = 10^{-3};$ $\eta_{CA} = 10^{-4}; d = 9; \beta = 0.9; C = 10; p = 2$	$var = 1;$ $C = 10; p = 2$	$C = 10$ $p = 2$
ionosphere	$\lambda_U = 10^{-6}; \lambda_U = 10^{-6}; \eta_R = 10^{-3};$ $\eta_{CA} = 10^{-3}; d = 17; \beta = 0.5; C = 1; p = 2$	$var = 1;$ $C = 1; p = 3$	$C = 0.1$ $p = 2$
pi-diabetes	$\lambda_U = 10^{-2}; \lambda_U = 10^{-6}; \eta_R = 10^{-3};$ $\eta_{CA} = 10^{-4}; d = 8; \beta = 0.5; C = 0.1; p = 3$	$var = 1$ $C = 1; p = 3$	$C = 10$ $p = 3$
sonar	$\lambda_U = 10^{-2}; \lambda_U = 10^{-4}; \eta_R = 10^{-3};$ $\eta_{CA} = 10^{-4}; d = 30; \beta = 0.1; C = 1; p = 3$	$var = 0.7$ $C = 10; p = 2$	$C = 0.1$ $p = 3$
spect	$\lambda_U = 10^{-2}; \lambda_U = 10^{0}; \eta_R = 10^{-3};$ $\eta_{CA} = 10^{-3}; d = 11; \beta = 0.1; C = 1; p = 3$	$var = 1$ $C = 1; p = 3$	$C = 0.1$ $p = 2$

There is a strong message we can derive from the hyper parameters results of Table 1. In no case the winning kernel degree was found to be $p = 1$, pointing to the conclusion that in all the listed datasets, non-linear dimensionality reduction (i.e. kernel degree $p > 1$) is superior.

The accuracy results in terms of error ratios is presented in Table 2. The winning method is shown in **bold**. As we can observe our proposed method outperforms the baselines in the majority of the datasets as shown in the wins row.

Table 2. Classification Accuracy - Error Ratios

DATASET	NSDR	PCA-SVMs	SVMs
breast_cancer_wisconsin	**0.070 ± 0.018**	0.082 ± 0.019	0.073 ± 0.021
ionosphere	***0.066 ± 0.008**	0.091 ± 0.010	0.140 ± 0.018
pi-diabetes	0.287 ± 0.023	0.280 ± 0.006	**0.274 ± 0.030**
sonar	**0.202 ± 0.053**	0.226 ± 0.129	0.226 ± 0.056
spect	**0.206 ± 0.002**	0.243 ± 0.103	**0.206 ± 0.002**
Wins (sig/n.sig)	**3.5 (1/2.5)**	0	1.5 (1/0.5)

NSDR improves the classification on the *ionosphere* dataset with a significant difference, denoted by $*$, while on the other datasets the gap to the second best is smaller. It is interesting to observe that in the cases of *breast_cancer_wisconsin, pi-diabetes* and *spect* the performance of SVMs is better than PCA-SVMs. This observation leads to a reasoning that those datasets are hardly compressible, therefore unsupervised dimensionality reduction PCA is outperformed. However, due to the added advantage of nonlinear supervision, NSDR recovers the disadvantage of PCA-SVMs and wins on *breast_cancer_wisconsin* and co-wins on *spect*, while loosing only in the *pi-diabetes* dataset.

6 Conclusions and Future Work

Throughout this study we presented a nonlinearly supervised dimensionality reduction technique, which jointly combined a joint optimization on reconstruction and classification accuracy. The reconstruction terms were expressed as matrix factorization decomposition of latent matrices, while the classification accuracy as a dual form kernel maximum margin classifier. The reduced dataset is learned via a coordinate descent algorithm which updates the reduced dimensionality dataset w.r.t to both loss terms simultaneously. Empirical results over binary datasets shows that the proposed method outperforms the selected baselines in the majority of the datasets. Having proven the concept on binary classification, we plan to extend the model for multi-class data as future work.

References

1. Samet, H.: Foundations of Multidimensional and Metric Data Structures. The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling. Morgan Kaufmann Publishers Inc., San Francisco (2005)
2. Grabocka, J., Nanopoulos, A., Schmidt-Thieme, L.: Classification of sparse time series via supervised matrix factorization. In: Hoffmann, J., Selman, B. (eds.) AAAI. AAAI Press (2012)
3. Das Gupta, M., Xiao, J.: Non-negative matrix factorization as a feature selection tool for maximum margin classifiers. In: Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, pp. 2841–2848. IEEE Computer Society, Washington, DC (2011)
4. Jolliffe, I.T.: Principal Component Analysis, 2nd edn. Springer (October 2002)

5. Wismüller, A., Verleysen, M., Aupetit, M., Lee, J.A.: Recent advances in nonlinear dimensionality reduction, manifold and topological learning. In: ESANN (2010)
6. Hoffmann, H.: Kernel pca for novelty detection. Pattern Recogn. 40(3), 863–874 (2007)
7. Sun, J., Crowe, M., Fyfe, C.: Extending metric multidimensional scaling with bregman divergences. Pattern Recognition 44(5), 1137–1154 (2011)
8. Gorban, A.N., Zinovyev, A.Y.: Principal manifolds and graphs in practice: from molecular biology to dynamical systems. Int. J. Neural Syst. 20(3), 219–232 (2010)
9. Lee, J.A., Verleysen, M.: Nonlinear dimensionality reduction. Springer, New York (2007)
10. Gashler, M.S., Martinez, T.: Temporal nonlinear dimensionality reduction. In: Proceedings of the IEEE International Joint Conference on Neural Networks, IJCNN 2011, pp. 1959–1966. IEEE Press (2011)
11. Singh, A.P., Gordon, G.J.: A unified view of matrix factorization models. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part II. LNCS (LNAI), vol. 5212, pp. 358–373. Springer, Heidelberg (2008)
12. Koren, Y., Bell, R.M., Volinsky, C.: Matrix factorization techniques for recommender systems. IEEE Computer 42(8), 30–37 (2009)
13. Rendle, S., Schmidt-Thieme, L.: Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In: Pu, P., Bridge, D.G., Mobasher, B., Ricci, F. (eds.) RecSys, pp. 251–258. ACM (2008)
14. Cai, D., He, X., Han, J., Huang, T.S.: Graph regularized nonnegative matrix factorization for data representation. IEEE Trans. Pattern Anal. Mach. Intell. 33(8), 1548–1560 (2011)
15. Giannakopoulos, T., Petridis, S.: Fisher linear semi-discriminant analysis for speaker diarization. IEEE Transactions on Audio, Speech & Language Processing 20(7), 1913–1922 (2012)
16. Menon, A.K., Elkan, C.: Predicting labels for dyadic data. Data Min. Knowl. Discov. 21(2), 327–343 (2010)
17. Rish, I., Grabarnik, G., Cecchi, G., Pereira, F., Gordon, G.J.: Closed-form supervised dimensionality reduction with generalized linear models. In: ICML 2008: Proceedings of the 25th International Conference on Machine Learning, pp. 832–839. ACM, New York (2008)
18. Fukumizu, K., Bach, F.R., Jordan, M.I.: Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. Journal of Machine Learning Research 5, 73–99 (2004)
19. Salakhutdinov, R., Hinton, G.: Learning a nonlinear embedding by preserving class neighbourhood structure. In: Proceedings of the International Conference on Artificial Intelligence and Statistics, vol. 11 (2007)
20. Zhang, D., hua Zhou Songcan Chen, Z.: Semi-supervised dimensionality reduction. In: Proceedings of the 7th SIAM International Conference on Data Mining, pp. 11–393 (2007)
21. Scholkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge (2001)
22. Platt, J.: Fast training of support vector machines using sequential minimal optimization. In: Schoelkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods - Support Vector Learning. MIT Press (1998)

Concurrent Execution of Data Mining Queries for Spatial Collocation Pattern Discovery*

Pawel Boinski and Maciej Zakrzewicz

Poznan University of Technology, Institute of Computing Science
{pawel.boinski,maciej.zakrzewicz}@cs.put.poznan.pl

Abstract. In spatial databases, Collocation Pattern Discovery is a very important data mining technique. It consists in searching for types of spatial objects that are frequently located together. Due to high requirements for CPU, memory or storage space, such data mining queries are often executed at times of low user activity. Multiple users or even the same user experimenting with different parameters can define many queries during the working hours that are executed, e.g., at off-peak night-time hours. Given a set of multiple spatial data mining queries, a data mining system may take advantage of potential overlapping of the queried datasets. In this paper we present a new method for concurrent processing of multiple spatial collocation pattern discovery queries. The aim of our new algorithm is to improve processing times by reducing the number of searches for neighboring objects, which is a crucial step for the identification of collocation patterns.

1 Introduction

Most of the spatial datasets consist of instances that are described by *spatial features* which can be interpreted as a characteristic of space in a particular location. Typical examples of spatial features include species, business types or points of interest (e.g., hospitals, airports). Shekhar and Huang introduced an important concept of *spatial collocation patterns* [11]. The definition of a spatial collocation pattern (or in short a *collocation*) assumes that it is a subset of spatial features whose instances (e.g., particular airport and custom office) are frequently located together in a spatial neighborhood. Such patterns are the product of the data mining which is one of the most important steps in the Knowledge Discovery in Databases - a non-trivial process of discovering valid, novel, potentially useful and ultimately understandable patterns in the data [7].

For end users *a data mining system* can be regarded as an advanced database with sophisticated querying methods. Users define data mining queries, i.e., a classes of interesting patterns, sets of criteria and input datasets. The task of the data mining system is to choose and execute an appropriate algorithm and finally return discovered patterns to the users. In regular databases the time required

* This paper was funded by the Polish National Science Center (NCN), grant No. 2011/01/B/ST6/05169.

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 184–195, 2013.

to execute user commands is usually very short (except some administrative tasks), while in data mining systems, the time to answer a single query can be expressed in minutes or even hours. Therefore, in a real-life scenario, data mining queries are collected during the user's working hours and executed at nights, when the system activity is low. It is very likely that some of the data mining queries are related in such a way, that they share some input data. This relation can be utilized to process them all at once in such a way that the total processing time will be reduced in comparison with the straightforward serial execution (the low activity time slot is limited to 6-8 hours). Another possibility is to execute multiple queries in such order that consecutive queries can take advantage of already computed and stored results of previous queries. Finally, the spatial data mining system could execute queries that occur at random times by incorporating them into currently ongoing data mining process.

In this paper we propose a new algorithm for processing batches of spatial data mining queries for collocation patterns discovery. We introduce a concurrent collocation candidate generation method and an extended iCPI-tree structure that stores materialized neighbor relationships for multiple queries. Conducted experiments have confirmed the high efficiency of the proposed algorithm.

2 Motivation and Related Work

2.1 Motivation

Consider a database of spatial objects describing various facilities in a particular city. One can be interested in collocation patterns involving cinemas, food stores, tram stops and schools whereas another user may want to find collocations of such features as theaters, cinemas, opera houses and tram stops. The first user wants to analyze districts d_1 and d_2 of the city and is interested in patterns with at least 40% prevalence, while the second user wants to analyze districts d_2 and d_3 with minimum prevalence of 25%. The neighbor relation in both cases is Euclidean distance less than 100 m. In general, we assume that the neighbor relation is consistent across the processed queries, however it can vary depending on the location (e.g., different for cities and rural areas) in the analyzed space.

The most trivial approach is to execute each query separately. We will refer to this strategy as a *sequential processing*. The sequential processing is easy to implement although it cannot benefit from the input data shared by multiple queries. In the considered example, instances of features 'cinema' and 'tram stop' located in the district d_2 are shared among two queries. By merging the execution of these queries we can reduce the number of searches across space required to identify instances of certain candidates.

2.2 Related Work

One can notice that collocation discovery problem is substantially similar to the frequent itemset discovery problem presented in [1], however, direct application

of well-known association mining algorithms, e.g., *Apriori* [2], is very challenging. Difficulties arise from significant differences in the characteristics between classical market-basket data and spatial data. For example, instances in the market-basket analysis are precisely nested in transactions, while instances of spatial features are embedded in a continuous space and share 'hidden' neighbor relationships. As a result, new methods for collocation mining have been developed. The most interesting ones are *Co-Location Miner* [11], *Joinless* [14] and *iCPI-tree* [12] accompanied with our work on efficient processing of spatial data mining queries in a limited memory environment [3,4,5].

The problem of efficient execution of multiple queries in classical databases has been extensively studied (e.g., [8,10]) and basically consists in a single execution of expressions shared by at least two queries. This general idea remains the same in the context of spatial data mining queries, however due to the more complex processing it cannot be directly transferred. The problem of batch processing of queries for association discovery has been introduced in [13]. The authors proposed two solutions, named *Mine Merge* and *Apriori Common Counting*, to reduce the total I/O and CPU cost of executing a set of data mining queries.

To the best of our knowledge there are no current works on batch processing of collocation pattern mining queries, although there are some works on indexing collocation patterns for future reuse [6] and on incremental maintenance of collocation patterns when a set of new spatial data arrives [9]. In the first approach, there is a computationally demanding step of pre-calculating and materializing all collocation instances. To compensate the time required to perform this task, a significant number of queries (counted in dozens or even hundreds) must be executed afterwards. In the second approach results of the query are materialized and updated in the response to changing input dataset. Contrary, we do not pre-calculate or materialize collocations. In the proposed method collocation queries are executed concurrently only for the required subset of the input data.

2.3 Basic Definitions

Definition 1. *Let f be a spatial feature. An object x is an* ***instance*** *of the feature f, if x is a type of f and is described by a location and unique identifier. Let F be a set of spatial features and S be a set of their instances. Given a neighbor relation R, we say that the* ***collocation*** *C is a subset of spatial features $C \subseteq F$ whose instances $I \subseteq S$ form a clique w.r.t. the relation R.*

Definition 2. *The* ***participation ratio*** *$Pr(C, f_i)$ of a feature f_i in the collocation $C = \{f_1, f_2, \ldots, f_k\}$ is a fraction of objects representing the feature f_i in the neighborhood of instances of collocation $C - \{f_i\}$. $Pr(C, f_i)$* is equal to the number of distinct objects of $f_i$in instances of C divided by the number of all objects of f_i. *The* ***participation index (prevalence measure)*** *$Pi(C)$ of a collocation $C = \{f_1, f_2, \ldots, f_k\}$ is defined as $Pi(C) = \min_{f_i \in C} \{Pr(C, f_i)\}$.*

Lemma 1. *The participation ratio and participation index are monotonically non-increasing with increases in the collocation size.*

Definition 3. *Given a subset of spatial instances* $I = \{o_l, \ldots, o_v\}$*, where* $l, v \in \{1, 2, \ldots, m\}$*, if* $o_i \leq o_j$ *holds for any* $l \leq i \leq j \leq v$*, the* I *is called as an* ***ordered instance set****. If the feature of* o_i *is not the same as the feature of* o_l *and* $R(o_l, o_i)$ *holds for any* $l < i \leq v$*, the* I *is called as* ***ordered neighbor relationship set of the instance*** o_l*. The set of ordered neighbor relationship sets of all instances of a spatial feature* x *is denoted as* δ_x*. Given a set of spatial features* $F = \{f_1, f_2, \ldots, f_n\}$ *and a set of ordered instance neighbor relationship of spatial these features* $\delta = \delta_{f_1} \cup \delta_{f_2} \cup \ldots \cup \delta_{f_n}$*, a tree designed as follows is called as an* ***improved Collocation Pattern Instances tree (iCPI-tree)****. The iCPI-tree consists of one root labeled as "null" and a set of the spatial features sub-trees as the children root. The spatial feature* f_i *sub-tree consists of the root* f_i *and each subset of* δ_{f_i} *as a branch of the root. Each branch records an ordered neighbor relationship set of corresponding instance and relevant feature.*

2.4 The iCPI-Tree Based Method

The general approach to collocation mining has been proposed in [11]. It consists of three major steps: (1) generating collocation candidates, (2) identifying instances for candidates and (3) filtering candidates w.r.t. to the minimum prevalence threshold. These steps are executed iteratively. In $k-$th iteration, size-k candidates are processed. The first step can be accomplished by applying well-known *Apriori* strategy [2] due to the anti-monotonicity property of the prevalence measure. The last step is very straightforward and basically consists in computing prevalence measure for each candidate. The most time consuming part of the algorithm is the second step. The first idea presumed that spatial join should be used to find co-located objects. In [14] a concept of materialized neighborhoods has been introduced. Wang et. al [12] extended this concept by defining a tree structure (called iCPI-tree) for fast identification of neighbors.

In the *iCPI-tree* each child of the root node is a subtree that contains neighbors for instances of a specific spatial feature. Sub-trees are composed of nodes representing spatial features of neighbors and leafs corresponding to neighbor instances. For example, in Fig. 2 (section 3.2) the tree $iCPI_1$ contains two subtrees for features A and B. Given the instance $A7$ we can easily find that it has one neighbor with B feature ($B6$) and two neighbors with C feature ($C5$ and $C8$). During the execution of the algorithm, new instances of candidates are constructed from instances of collocations from previous iteration. For example, to find instances of candidate ABC, an instance $A1, B2$ can be used. The procedure searches for neighbors with feature C of $A1$ and $B2$. If there are common neighbors for both elements, a new instance is constructed. Using the $iCPI_1$ tree, one can find that there is an instance $A1, B2, C3$. For details of the iCPI-tree based algorithm please consult the paper [12].

3 Batch Processing of Spatial Data Mining Queries

In this section we introduce preliminaries, motivations and our new algorithm for batch processing of spatial data mining queries.

3.1 Preliminaries

Definition 4. *A **spatial data mining query** SDMQ is a tuple* (S, F, L, R, mp)*, where* S *is a spatial framework,* F *is a set of spatial features,* L *is a subset of spatial framework* S*,* R *is a neighbor relation and* mp *is a minimum prevalence. The result of the* $SDMQ$ *is a set of collocation patterns discovered from instances of* F *w.r.t. to* R *located in* L *having the prevalence not less than* mp*.*

Definition 5. *A **set of spatial data mining queries*** $QS = \{(SDMQ_1, t_1), (SDMQ_2, t_2), \ldots, (SDMQ_n, t_n)\}$ *consists of pairs* $(SDMQ_i, t_i)$*,* $1 \leq i \leq n$ *where* $SDMQ_i$ *is a spatial data mining query and* t_i *is the time of the arrival of this query to the data mining system.*

In this work we focus on the execution of batches of data mining queries, i.e., sets of n spatial data mining queries where for each $1 \leq i, j \leq n$, $t_i = t_j$.

Definition 6. *A set* $A_S = \{a_1, a_2, \ldots, a_m\}$ *is a set of **distinct areas** of* S*, i.e., set of uniquely numbered subsets of spatial framework* S *such that for each* $1 \leq i, j \leq m$*, areas* a_i *and* a_j *do not overlap and all areas from* A_s *constitute a framework* S*.*

Definition 7. *A **shared collocation pattern** is a subset of spatial features with an additional list of SDMQs that it belongs to. A **shared collocation instance** is a set of instances of collocation features located together in a spatial neighborhood with assigned set of distinct areas* $SA = \{a_k, \ldots, a_l\}$ *such that for each* $a_i \in SA$ *at least one collocation feature instance is located in* a_i*.*

Definition 8. *Given a set of* n *data mining queries* QS *and a set* $F = \{F_1 \cup F_2 \cup \ldots \cup F_n\}$*, where* F_i *denotes a set of spatial features of* $SDMQ_i \in QS$*, a **Common iCPI-tree** is an enhanced iCPI-tree such that for each spatial feature* $f_i \in F$*, sub-tree consists of the root* f_i *and each subset of a set of ordered instance neighbor relationship sets of all instances of* f_i*. Each instance node of* f_i *is extended with the identifier of the distinct area that it belongs to.*

3.2 The Common iCPI-Tree Based Method

In this section we introduce our new algorithm called *Common iCPI-tree* for the concurrent execution of multiple queries in a batch. The pseudocode for this algorithm is shown in Alg. 1. Within the following paragraphs we will refer to this pseudocode by putting the corresponding line numbers in brackets. The general idea introduced in the iCPI-tree method remains the same although there are additional algorithm steps and extensions of structures required to perform effective execution of batched queries.

To explain how our method works, we will use an example dataset shown in Fig. 1. There are 4 features A, B, C and D with the total of 21 instances. The batch is composed of two queries: $SDMQ_1 = (Input, \{A, B, C\}, 1.5 < x \leq 13, d \leq 2, 0)$ and $SDMQ_2 = (Input, \{A, B, C, D\}, 6.5 < x \leq 19.5, d \leq 2, 0)$. For

simplicity and better explanation of the algorithm both minimum prevalence thresholds are set to 0 and we use only x axis to specify query area of interest. A line connecting two objects represents a neighbor relationship (distance not grater than 2 units).

Algorithm 1. Common iCPI-tree based collocation mining algorithm

Input: QS - a set of n spatial data mining queries with the same r neighbor relation
Output: a set of collocation patterns
Variables: F_i - a set of $SDMQ_i$ spatial features, A - a set of distinct areas for QS, $CiCPI_k$ - a Common iCPI-tree, SC_k - a set of size-k shared collocation candidates, SP_k − a set of size-k prevalent shared collocations, SPI_k − a set of size-k shared clique instances

1: **procedure** COMMON_ICPI(QS)
2: A = genDistinctAreas(QS)
3: $CiCPI$ = genCommonTree ($F_1 \cup F_2 \ldots \cup F_n, A, r$); $k = 1$
4: SP_k = genOneElementSharedCollocations($F_1 \cup F_2 \ldots \cup F_n, A$)
5: **while** ($SP_k \neq \emptyset$) **do**
6: SC_{k+1} = AprioriGenSharedCandidates (SP_k)
7: **for** $sc \in SC_{k+1}$ **do** /* for each shared candidate */
8: **for** $sp_{inst} \in SPI_k$ with features equal to sc prefix **do**
9: **if** sp_{inst} belongs only to areas in sc **then**
10: CN = searchCommonNeighboors ($sp_{inst}, sc, CiCPI$)
11: **for** $ne \in CN$ **do**
12: $sc_{newInst} = sp_{inst} \cup \{ne\}$, add ne area to $sc_{newInst}$ areas
13: $SPI_{k+1} = SPI_{k+1} \cup \{sc_{newInst}\}$
14: **end for**
15: **end if**
16: **end for**
17: **end for**
18: SP_{k+1} = getPrevalent (SC_{k+1}, SPI_{k+1}, QS)
19: $k = k + 1$
20: **end while**
21: return $\bigcup(SP_2, \ldots, SP_{k-1})$
22: **end procedure**

In the original iCPI-tree method the most computationally demanding part is the step of searching a tree structure to construct new collocation instances having the clique property. The sequential processing strategy requires to construct and process a separate iCPI-tree for each query from the batch set (two iCPI-trees for sample data are shown in Fig. 2). We propose to build only one tree that contains instances for all queries (line 3). We refer to this structure as *Common iCPI-tree* (CiCPI-tree). To distinguish instances among different queries an additional identifier has to be stored with each node representing an object instance in the tree. In our opinion, the best solution is to use properly constructed bitmaps for that purpose. A bitmap (also known as a bitset, bit array or bit vector) is a compact structure that stores an array of bits. It is

extremely fast due to the hardware, low-level parallelism in processing whole words or bytes. To determine appropriate bitmaps for tree elements, first of all, space has to be divided into a set of distinct areas (line 2).

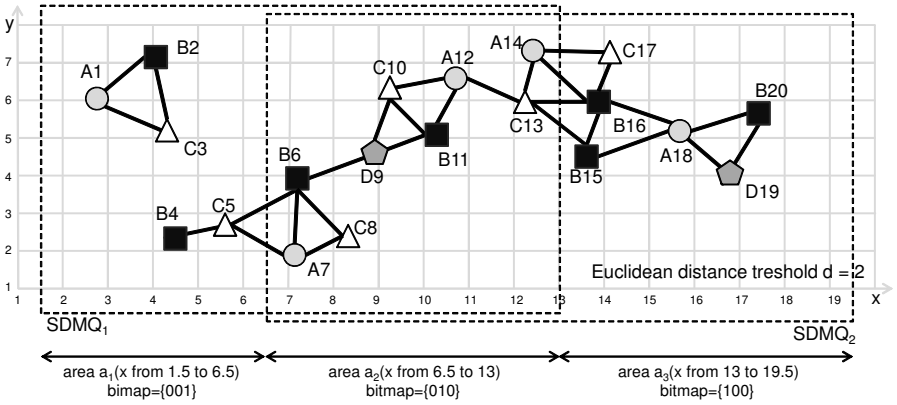


Fig. 1. Input datasets

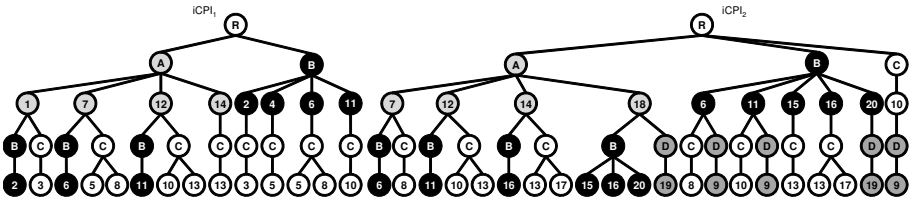


Fig. 2. Two iCPI-trees generated in sequential processing approach

In the sample dataset, three distinct areas can be distinguished: $a_1(1.5 < x \leq 6.5)$, $a_2(6.5 < x \leq 13)$ and $a_3(13 < x \leq 19.5)$. For each area, a unique bitmap is generated by setting the i-th element for the i-th area. Therefore we have identifiers $\{001\}$, $\{010\}$, $\{100\}$ for areas a_1, a_2 and a_3 respectively (notice: although three areas can be encoded on two bits, the mentioned solution is more efficient in the further processing). Given the set of distinct areas, the next step is to create the Common iCPI-tree. For each object o_i analyzed by at least one SDMQ all neighbors with features greater than o_i feature have to be found. To perform this task a plane sweep method or a spatial index can be utilized. Neighbors are ordered by their feature (e.g., using the lexical order) and their identifier. For each object a new bitmap is created. It must correspond to the bitmap of the area in which this object is located in. For example, given the object $A14\{010\}$ the final list of neighbors contains $B16\{100\}$, $C13\{010\}$ and $C17\{100\}$. The discovered neighborhoods (and their bitmaps) are inserted into the CiCPI-tree using the procedure described in [12]. Figure 3 presents final

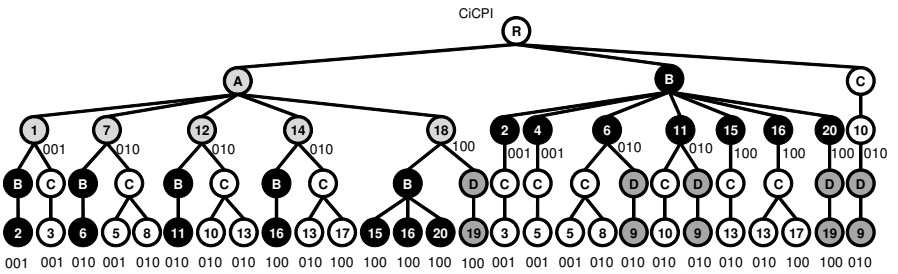


Fig. 3. Common iCPI-tree generated in concurrent processing approach

CiCPI-tree structure that will be used to scan for neighbor instances in the consecutive iterations.

In the k-th iteration (lines 5-20), the algorithm discovers prevalent size-k+1 collocations for each SDMQ by generating (shared) collocation candidates (line 6), identifying their instances (lines 8-16) and counting their prevalences (line 18). Due to the participation index definition, all size-1 collocations are prevalent (with prevalence = 100%). In the sequential processing approach, *AprioriGen* [2] method would be applied to each SDMQ separately. We propose to generate candidates simultaneously for all queries using *AprioriGenSharedCandidates* method (line 6). At the beginning, for each candidate collocation a bitmap is assigned. Each bitmap has length equal to the size of the batch and indicates queries which share this particular candidate. If such a candidate (or a collocation) is shared by the i-th SDMQ, the i-th bit is set. In our example there are 4 size-1 collocations: A, B, C and D. All except D are shared between $SDMQ_1$ and $SDMQ_2$ hence the following set of bitmaps is assigned: {11}, {11}, {11}, {10}. The general idea of AprioriGenSharedCandidates is similar to the original method. All pairs of size-k-1 collocations sharing at least one query are joined to get size-k candidates. Each generated candidate has a bitmap resulting from bitwise AND operation on all bitmaps from its size-k-1 subsets. Finally, a pruning step is applied to remove candidates that cannot be prevalent. In the introduced example, size-2 candidates are: $AB\{11\}$, $AC\{11\}$, $AD\{10\}$, $BC\{11\}$, $BD\{10\}$ and $CD\{10\}$, size-3: $ABC\{11\}$, $ABD\{10\}$, $ACD\{10\}$ and $BCD\{10\}$, size-4: $ABCD\{10\}$.

Starting with $k = 2$, instances for each size-k candidate are constructed by expanding instances of size-k-1 collocation discovered in the previous iteration (lines 8-16), however a sharing property of candidates and collocations must be taken into consideration. For $k = 2$ the process traverses the CiCPI-tree and for each instance of the first candidate feature, the neighbors with the second feature are retrieved from the tree. For example, given the candidate $AB\{11\}$, for the instance $A14$ there is one neighbor $B16$. Because the instance $A14, B16$ can be shared between two queries (the candidate $AB\{11\}$ is shared by $SDMQ_1$ and $SDMQ_2$), there is a necessity to store such information in the form of bitmap. It is a result of bitwise OR operation performed on bitmaps for individual objects.

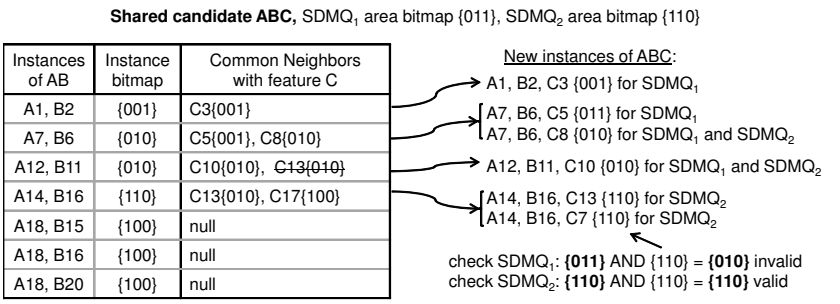


Fig. 4. Search procedure for instances of shared candidates

For the considered instance it is $\{010\} \cup \{100\} = \{110\}$. The same procedure is applied to the remaining instances of A ($A1$, $A7$, $A12$ and $A18$).

Let us now assume that there is a candidate $ABC\{11\}$. To generate its instances, we try to expand already known instances of $AB\{11\}$ by searching the tree for instances of C (line 10). For the aforementioned $A14, B16$ instance, neighbors $C13$ and $C17$ are retrieved from the CiCPI-tree. Both of them are common neighbors of $A14$ and $B16$, therefore instances $A14, B16, C13$ and $A14, B16, C17$ are created (lines 12-13). Once again bitmaps for such instances have to be computed using bitwise OR operation on the bitmap for $A14, B16$ and bitmaps for neighbors. The final bitmap is $\{110\}$ for both mentioned instances. Full example illustrating the search procedure for candidate $ABC\{11\}$ is shown in Fig. 4.

The prevalence for each query is computed by browsing through discovered instances. Because one instance can belong to multiple queries, it is necessary to increment prevalence counters only for applicable queries. To identify queries that a particular instance belongs to, bitmaps representing a sum of distinct query areas identifiers and a bitmap for candidate instance can be used. If for a given query the result of bitwise AND operation is equal to the instance bitmap, it means that such an instance belongs to this query. For example, the instance $A12, B11, C10\{010\}$ belongs to $SDMQ_1$ as well as to $SDMQ_2$ because its bitmap is contained in bitmaps representing areas for both queries (Fig. 4).

4 Experiments

In order to evaluate the performance of the Common iCPI-tree method we performed several experiments. For better control over the experiments, we used synthetic datasets that were generated using a method similar to the approach described in [14]. We have prepared 20 datasets with the following parameters. The number of spatial objects: 50.000-600.000, the number of spatial features: 20-60, the maximal collocation pattern length: 4-8, the number of noise instances: 20%-80%. To simulate dense and sparse datasets we used two spatial frameworks

with sizes 10000x10000 and 1000x1000 units. In all tests the neighbor distance threshold was set to 5 units. The experiments were conducted on a Linux PC with AMD Athlon64 4200+ processor and 4 GB of main memory. All programs were written in Java.

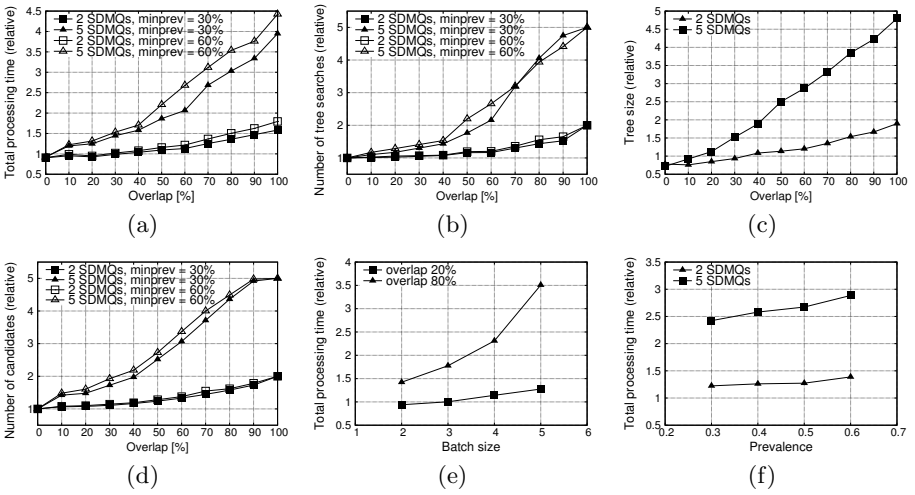


Fig. 5. Performance gain for CiCPI-tree in comparison with sequential processing

We have also prepared a set of 200 sample batches for experiments. We varied the number of queries in batch (2-5), the minimum prevalence threshold (0.3-0.6) and the level of overlapping between queries (0%-100%). The level of overlapping was equal to the average ratio of shared objects between each pair of batched queries. The CiCPI-tree method has been compared with sequential processing. For clarity, all presented results are *relative* to the results obtained from sequential execution. A particular value on the chart should be interpreted as the *acceleration* (or *improvement*) in comparison with sequential execution. For example, for charts presenting the time performance, values below 1, e.g. 0.8, mean that sequential processing took 80% of time required by CiCPI-tree solution, while value 4 means that CiCPI-tree method is 4 times faster than sequential processing.

Figure 5(a) presents how the average processing times change with the increasing overlap threshold. The series include batches of 2 and 5 queries with minimum prevalence set to 30% and 60%. As we expected, the performance gain increases with the increasing overlap of datasets. For bathes of 2 queries, the performance gain can be observed after exceeding 20% overlap, while for 5 queries even for 10% overlap the new algorithm results in faster execution times. When there is no overlap, the CiCPI method is about 10% slower than sequential processing. For 100% overlap batches of 2 and 5 queries are executed with

CiCPI-tree up to 1.75 and 4.5 times faster respectively. For example, in one of our tests the required processing time dropped from 26 minutes to less than 6 minutes.

Figure 5(b) presents the relative number of required tree searches for neighbors. Similarly to the previous experiment, the reduction of searches is increasing with the increasing overlap. With 100% overlap, our new algorithm performs only one tree search for all queries in batch, therefore there are 2 and 5 times less searches for batches of 2 and 5 queries.

Figure 5(c) presents the comparison of CiCPI-tree size and cumulative size of iCPI-trees generated in sequential processing. While the overlap is low, CiCPI-trees can reach bigger sizes than corresponding sets of iCPI-trees due to the overhead resulting from necessity to store additional bitmaps. With increasing overlap, such overhead is compensated by the elimination of redundant branches that can be found in iCPI-trees. In this chart there is no distinction between prevalence thresholds because the size of the CiCPI-tree (and corresponding iCPI-trees) does not depend on the prevalence value.

Figure 5(d) presents how the sharing property of collocations affects the total number of candidates. In comparison with sequential processing the number of candidates is greatly reduced even for low values of overlap threshold, especially for batches of 5 queries.

Finally, in the last series of experiments we analyzed how the number of queries in batch and minimum prevalence affect total processing time. As we expected the bigger the batch is, the bigger performance gain is achieved, notably for higher overlap thresholds (Fig. 5(e)). On the contrary, the prevalence measure does not have such essential impact on the performance gain (Fig. 5(f)). However, when the minimum prevalence threshold is low more multi-feature candidates are being generated. The possibility of sharing such candidates is limited and therefore the acceleration is reduced.

5 Summary and Future Work

In this paper we have defined the problem of efficient execution of batched spatial data mining queries for collocation patterns discovery. We have proposed a new algorithm, called CiCPI-tree, that significantly outperforms the straightforward serial execution of multiple queries. Processing times are reduced by eliminating redundant searches for neighbors and introducing shared representation of collocation instances with combined candidates generation.

In the future work we will focus on memory constraints that can be crucial when a batch of queries is being processed. In the ideal circumstances, the CiCPI-tree structure should fit in memory, however in real life applications this can be impossible. We believe that our previous researches on collocation pattern mining in limited memory environments can be adopted to concurrent processing of spatial data mining queries. Another interesting subject involve different strategies for processing spatial data mining queries, e.g., sets of queries with random times of arrival.

References

1. Agrawal, R., Imieliński, T., Swami, A.: Mining Association Rules Between Sets of Items in Large Databases. SIGMOD Rec. 22(2), 207–216 (1993)
2. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules in Large Databases. In: Proceedings of the 20th International Conference on Very Large Data Bases, pp. 487–499. Morgan Kaufmann Publishers Inc., San Francisco (1994)
3. Boinski, P., Zakrzewicz, M.: Hash Join Based Spatial Collocation Pattern Mining. Foundations of Computing and Decision Sciences 36(1), 3–15 (2011)
4. Boinski, P., Zakrzewicz, M.: Collocation Pattern Mining in a Limited Memory Environment Using Materialized iCPI-Tree. In: Cuzzocrea, A., Dayal, U. (eds.) DaWaK 2012. LNCS, vol. 7448, pp. 279–290. Springer, Heidelberg (2012)
5. Boinski, P., Zakrzewicz, M.: Partitioning Approach to Collocation Pattern Mining in Limited Memory Environment Using Materialized iCPI-Trees. In: Morzy, T., Härder, T., Wrembel, R. (eds.) Advances in Databases and Information Systems. AISC, vol. 186, pp. 19–30. Springer, Heidelberg (2013), http://dx.doi.org/10.1007/978-3-642-32741-4_3
6. Celik, M., Kang, J.M., Shekhar, S.: Zonal Co-location Pattern Discovery with Dynamic Parameters. In: ICDM, pp. 433–438. IEEE Computer Society (2007)
7. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P.: From Data Mining to Knowledge Discovery in Databases. AI Magazine 17, 37–54 (1996)
8. Giannikis, G., Alonso, G., Kossmann, D.: SharedDB: Killing One Thousand Queries With One Stone. Proc. VLDB Endow. 5(6), 526–537 (2012), http://dl.acm.org/citation.cfm?id=2168651.2168654
9. He, J., He, Q., Qian, F., Chen, Q.: Incremental Maintenance of Discovered Spatial Colocation Patterns. In: Proceedings of the 2008 IEEE International Conference on Data Mining Workshops, ICDMW 2008, pp. 399–407. IEEE Computer Society, Washington, DC (2008), http://dx.doi.org/10.1109/ICDMW.2008.60
10. Sellis, T.K.: Multiple-query optimization. ACM Trans. Database Syst. 13(1), 23–52 (1988), http://doi.acm.org/10.1145/42201.42203
11. Shekhar, S., Huang, Y.: Discovering Spatial Co-location Patterns: A Summary of Results. In: Jensen, C.S., Schneider, M., Seeger, B., Tsotras, V.J. (eds.) SSTD 2001. LNCS, vol. 2121, pp. 236–256. Springer, Heidelberg (2001)
12. Wang, L., Bao, Y., Lu, J.: Efficient Discovery of Spatial Co-Location Patterns Using the iCPI-tree. The Open Information Systems Journal 3(2), 69–80 (2009)
13. Wojciechowski, M., Zakrzewicz, M.: Methods for Batch Processing of Data Mining Queries. In: Haav, H.M., Kalja, A. (eds.) Proceedings of the Fifth International Baltic Conference on Databases and Information Systems (DB&IS 2002), pp. 225–236. Institute of Cybernetics at Tallin Technical University (June 2002)
14. Yoo, J.S., Shekhar, S., Celik, M.: A Join-Less Approach for Co-Location Pattern Mining: A Summary of Results. In: Proceedings of the IEEE International Conference on Data Mining, pp. 813–816. IEEE Computer Society, Washington (2005)

Depth-First Traversal over a Mirrored Space for Non-redundant Discriminative Itemsets

Yoshitaka Kameya and Hiroki Asaoka

Dept. of Information Engineering, Meijo University
1-501 Shiogama-guchi, Tenpaku-ku, Nagoya 468-8502, Japan
ykameya@meijo-u.ac.jp

Abstract. Discriminative pattern mining is known under the names of subgroup discovery, contrast set mining, emerging pattern mining, etc. and has been intensively studied for the last 15 years. Based on the sophisticated techniques developed so far (e.g. branch-and-bound search, minimum support raising, and redundancy elimination including the use of closed patterns), this paper proposes an efficient exact algorithm for finding top-k discriminative patterns that are not redundant and would be of value at a later step in prediction or knowledge discovery. The proposed algorithm is unique in that it conducts depth-first search over enumeration trees in a mirrored form of conventional ones, and by this design we can keep compact the list of candidate top-k patterns during the search and consequently high the minimum support threshold. Experimental results with the datasets from UCI Machine Learning Repository clearly show the efficiency of the proposed algorithm.

Keywords: discriminative pattern mining, top-k mining, minimum support raising, closed itemsets, dual-monotonicity, suffix enumeration trees.

1 Introduction

Discriminative pattern mining is known under the names of subgroup discovery [21], contrast set mining [1], emerging pattern mining [3], supervised descriptive rule discovery [11], cluster grouping [23], and so on and has been intensively studied for the last 15 years. The obtained discriminative patterns can be used to characterize a particular class c of interest, or to build more precise classifiers. One current issue in discriminative pattern mining is to deal with the redundancy among the obtained patterns. For example, suppose that we are performing top-k mining and a pattern $\{\mathsf{A}\}$ is significantly relevant to a class c of interest. Then, the patterns including A, such as $\{\mathsf{A},\mathsf{B}\}$, $\{\mathsf{A},\mathsf{C}\}$ and $\{\mathsf{A},\mathsf{B},\mathsf{C}\}$, also tend to be relevant to c, and would occupy the list of the final top-k patterns. So we hope to find a more informative collection of top-k patterns by eliminating such redundancy in a reasonable way. For instance, let us introduce a constraint called *productivity* [20]. Then, if a pattern $\{\mathsf{A},\mathsf{C},\mathsf{D}\}$ is more relevant to the class c of interest than another pattern $\{\mathsf{A},\mathsf{C}\}$, we consider that there is something meaningful in the combination of A, C and D, but if the former is less relevant than the latter, the former can be considered redundant and removed.

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 196–208, 2013.

In the literature of frequent/discriminative pattern mining, we have conventionally used *prefix enumeration trees* (e.g. [1,2,19]), illustrated in Fig. 1 (left) which include items A, B, C and D ordered as $\mathsf{A} \prec \mathsf{B} \prec \mathsf{C} \prec \mathsf{D}$. This paper, on the other hand, proposes to use *suffix enumeration trees* [10], illustrated in Fig. 1 (right), as a mirrored form of prefix enumeration trees. In a prefix (resp. suffix) enumeration tree, the parent of each node (pattern)[1] is its immediate prefix (resp. suffix). It is less known that when branching ascendingly w.r.t. $\prec$, FP-Growth [8] implicitly runs over suffix enumeration trees. The merit of using suffix enumeration trees comes from a property that *when a node $\boldsymbol{x}$ is visited in a depth-first (and left-to-right) search, all of $\boldsymbol{x}$'s sub-patterns have already been visited.* In the suffix enumeration tree in Fig. 1, when visiting $\{\mathsf{A}, \mathsf{C}, \mathsf{D}\}$, we have already visited $\{\mathsf{A}\}$, $\{\mathsf{C}\}$, $\{\mathsf{D}\}$, $\{\mathsf{A}, \mathsf{C}\}$, $\{\mathsf{A}, \mathsf{D}\}$ and $\{\mathsf{C}, \mathsf{D}\}$, but this is not the case in prefix enumeration trees. This property on the visiting order makes efficient the tests on the set-inclusion-based constraints among patterns.

Based on the observation above, we propose an efficient exact algorithm for finding top-k non-redundant discriminative patterns under two set-inclusion-based constraints among patterns. One is closedness, which has been studied in frequent pattern mining [15,19], and the other is productivity illustrated before. We can say that the proposed algorithm has two technical contributions. First, it adopts a new, relaxed condition called *dual-monotonicity*, which is desired to be satisfied by the scores on relevance. Indeed, dual-monotonicity is shown to play a crucial role in various aspects of the proposed algorithm. Second, the proposed algorithm introduces a mirrored version of prefix-preserving closure extension in LCM [19] in order to traverse over a search space like a suffix enumeration tree. We show formally and empirically that this mirrored operation successfully keeps compact the list of candidate top-k patterns during the search and consequently high the minimum support threshold.

The remainder of this paper is outlined as follows. Section 2 describes dual-monotonicity together with the concepts and the techniques that have been developed in the literature of discriminative pattern mining. Section 3 presents our proposed method. Some experimental results are reported in Section 4, and lastly Section 5 concludes the paper.

2 Dual-Monotonicity of Relevance Scores

2.1 Preliminaries

First, we introduce some notations. We consider a dataset $\mathcal{D} = \{t_1, t_2, \ldots, t_N\}$ of size N, where t_i is a transaction, a set of items. The set of all items appearing in $\mathcal{D}$ is denoted by $\mathcal{X}$. Also each transaction belongs to one of pre-defined classes $\mathcal{C}$, and let c_i be the class of transaction t_i. A pattern $\boldsymbol{x}$ is a subset of $\mathcal{X}$, and let $\mathcal{P}$ be the set of all possible patterns. We say that $\boldsymbol{x}$ matches a transaction t_i when $\boldsymbol{x} \subseteq t_i$. Depending on the context, we interchangeably denote a pattern as

[1] We only consider enumeration trees which have a one-to-one map between the nodes in a tree and the possible patterns. So we refer to a node by its corresponding pattern.

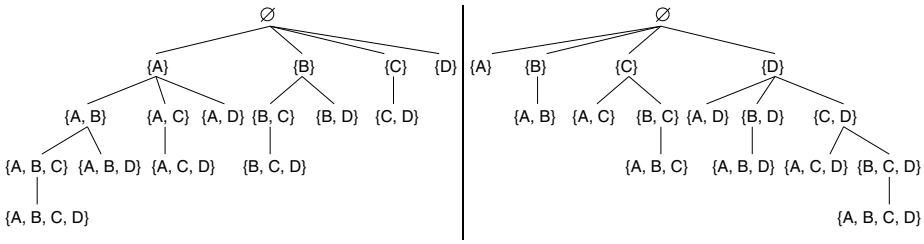


Fig. 1. A prefix enumeration tree (left) and a suffix enumeration tree (right)

a vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, as a set $\boldsymbol{x} = \{x_1, x_2, \ldots, x_n\}$, or as a conjunction $\boldsymbol{x} = (x_1 \wedge x_2 \wedge \ldots \wedge x_n)$. Besides, we introduce some total order $\prec$ among items, and by default, we place the items in a transaction/pattern following $\prec$. Transaction/patterns can be ordered in a lexicographical way w.r.t. $\prec$.

The probabilities treated in this paper are all empirical ones, i.e. they are computed from the statistics on the dataset $\mathcal{D}$. First, we define some subsets of $\mathcal{D}$: $\mathcal{D}_c = \{i \mid c_i = c, 1 \leq i \leq N\}$, $\mathcal{D}(\boldsymbol{x}) = \{i \mid \boldsymbol{x} \subseteq t_i, 1 \leq i \leq N\}$ and $\mathcal{D}_c(\boldsymbol{x}) = \{i \mid c_i = c, \boldsymbol{x} \subseteq t_i, 1 \leq i \leq N\}$, where $c \in \mathcal{C}$ is the class of interest. A joint probability $p(c, \boldsymbol{x})$ is then obtained as $|\mathcal{D}_c(\boldsymbol{x})|/N$. Also we use a symbol $\neg$ for negation, e.g. we have $\mathcal{D}_{\neg c} = \mathcal{D} \setminus \mathcal{D}_c$, $p(c, \neg\boldsymbol{x}) = |\mathcal{D}_c \setminus \mathcal{D}_c(\boldsymbol{x})|/N$, $p(\neg c, \boldsymbol{x}) = |\mathcal{D}(\boldsymbol{x}) \setminus \mathcal{D}_c(\boldsymbol{x})|/N$, and so on. Using joint probabilities, marginal probabilities and conditional probabilities are computed, e.g. we obtain $p(\boldsymbol{x}) = p(c, \boldsymbol{x}) + p(\neg c, \boldsymbol{x})$, $p(c) = p(c, \boldsymbol{x}) + p(c, \neg\boldsymbol{x})$ or $p(c \mid \boldsymbol{x}) = p(c, \boldsymbol{x})/p(\boldsymbol{x})$.

2.2 Relevance Scores

As stated before, we seek for k patterns that are relevant to a class c of interest. For that, we first adopt a relevance score R_c, a function from $\mathcal{P}$ to $\mathbb{R}$ (the set of real numbers). The relevance measured by $R_c(\boldsymbol{x})$ can also be regarded as interestingness of a class association rule $\boldsymbol{x} \Rightarrow c$, where $\boldsymbol{x}$ is a pattern. Among dozens of relevance scores proposed so far [4,10,11], we adopt *F-score* $\mathrm{F}_c(\boldsymbol{x}) = 2p(c \mid \boldsymbol{x})p(\boldsymbol{x} \mid c)/(p(c \mid \boldsymbol{x}) + p(\boldsymbol{x} \mid c)) = 2p(c, \boldsymbol{x})/(p(c) + p(\boldsymbol{x}))$, which is a popular measure in information retrieval or evaluation of classifiers.

Now consider a class association rule $\boldsymbol{x} \Rightarrow c$ applied to the original dataset $\mathcal{D}$. Then, true positive rate (TPR) is written as $p(\boldsymbol{x} \mid c) = p(c, \boldsymbol{x})/p(c)$ in our notation, and called the *positive support* of $\boldsymbol{x}$ for class c. Similarly, false positive rate (FPR) is written as $p(\boldsymbol{x} \mid \neg c)$ and called the *negative support*. The ROC space [4,14] is then formed by TPR (as y-axis) and FPR (as x-axis) and each pattern $\boldsymbol{x}$ is located as a point (v, u) in the ROC space, where u and v respectively indicate TPR and FPR of $\boldsymbol{x}$. For brevity, 'support' means positive support unless explicitly noted. Since we seek for the patterns relevant to a particular class c, we are only interested in the patterns $\boldsymbol{x}$ such that $p(\boldsymbol{x} \mid c) \geq p(\boldsymbol{x} \mid \neg c)$ or equivalently $p(c \mid \boldsymbol{x}) \geq p(c)$.

In the literature, the *convexity* of the relevance score has been exploited in branch-and-bound pruning [7,13,14,23]. Recently Nijssen et al. introduced

a property called *zero diagonal convexity* [14]: $R_c(\boldsymbol{x})$ is zero diagonal convex iff $R_c(\boldsymbol{x})$ is convex and reaches its minimum in all the points on the diagonal TPR = FPR in the ROC space. Several popular relevance scores such as the Fisher score, information gain, Gini index, χ^2, support difference are zero diagonal convex [14]. Furthermore, zero diagonal convexity can be relaxed as a new condition called *dual-monotonicity*, which is defined as follows:

Definition 1. *Let R_c be a relevance score for a class c of interest. Then, R_c is dual-monotonic iff $R_c(\boldsymbol{x})$ is monotonically increasing w.r.t. $p(\boldsymbol{x} \mid c)$ and monotonically decreasing w.r.t. $p(\boldsymbol{x} \mid \neg c)$ wherever $p(\boldsymbol{x} \mid c) \geq p(\boldsymbol{x} \mid \neg c)$.* □

Recall here that $R_c(\boldsymbol{x})$ is a function of TPR $u = p(\boldsymbol{x} \mid c)$ and FPR $v = p(\boldsymbol{x} \mid \neg c)$. Interestingly, dual-monotonicity includes two out of Piatetsky-Shapiro's three conditions desired for relevance scores [4,17].[2] Besides, dual-monotonicity obviously holds if $R_c(\boldsymbol{x})$ satisfies zero diagonal convexity. In contrast, for F-score, dual-monotonicity holds while convexity does not. As we will see, dual-monotonicity plays a crucial role in various aspects of the proposed algorithm.

2.3 Redundancy Elimination with Set-Inclusion-Based Constraints

As stated in the introduction, elimination of redundant patterns is necessary for having more informative results. In this paper, we focus on redundancy elimination based on set-inclusion-based constraints among patterns. One popular technique for this is to use the closedness constraint, and we additionally introduce a generalized version of set-inclusion-based constraint called *productivity*.

To explain the closedness constraint, we first introduce a closure operator Γ such that $\Gamma(\boldsymbol{x}, \mathcal{D}) = \bigcap_{t \in \mathcal{D}(\boldsymbol{x})} t$, where $\mathcal{D}$ is the transactions and $\boldsymbol{x}$ is some pattern. Here $\Gamma(\boldsymbol{x}, \mathcal{D})$ is called a *closure* of $\boldsymbol{x}$ w.r.t. $\mathcal{D}$. A closed pattern is then a pattern $\boldsymbol{x}$ such that $\boldsymbol{x} = \Gamma(\boldsymbol{x}, \mathcal{D})$. Each closed pattern $\boldsymbol{x}$ is the maximal pattern in an equivalence class $[\boldsymbol{x}] = \{\boldsymbol{x}' \mid \mathcal{D}(\boldsymbol{x}) = \mathcal{D}(\boldsymbol{x}')\} = \{\boldsymbol{x}' \mid \boldsymbol{x} = \Gamma(\boldsymbol{x}', \mathcal{D})\}$ and seen as a representative of $[\boldsymbol{x}]$. Since the size of $[\boldsymbol{x}]$ can be exponential, focusing only on closed patterns often leads to a significant reduction of the search space.

Next, let us consider a situation for discriminative pattern mining. Let c be a class of interest, $\mathcal{D}_c$ the transactions that belong to c, and $\boldsymbol{x}$ some pattern. Also let $\boldsymbol{x}^* = \Gamma(\boldsymbol{x}, \mathcal{D}_c)$. We further note that $\mathcal{D}_c(\boldsymbol{x}^*) = \mathcal{D}_c(\boldsymbol{x})$ since $\boldsymbol{x}^*$ and $\boldsymbol{x}$ are in the same equivalence class $[\boldsymbol{x}]$, and $\mathcal{D}'(\boldsymbol{x}^*) \subseteq \mathcal{D}'(\boldsymbol{x})$ for any transactions $\mathcal{D}'$ since $\boldsymbol{x}^*$ is the maximal pattern in $[\boldsymbol{x}]$. Then, under a dual-monotonic relevance score R_c, we have $R_c(\boldsymbol{x}^*) \geq R_c(\boldsymbol{x})$ since $p(\boldsymbol{x}^* \mid c) = p(\boldsymbol{x} \mid c)$ (from $\mathcal{D}_c(\boldsymbol{x}^*) = \mathcal{D}_c(\boldsymbol{x})$) and $p(\boldsymbol{x}^* \mid \neg c) \leq p(\boldsymbol{x} \mid \neg c)$ (from $\mathcal{D}_{\neg c}(\boldsymbol{x}^*) \subseteq \mathcal{D}_{\neg c}(\boldsymbol{x})$) [5,18]. Now, interestingly, it is also justified from the viewpoint of the relevance score R_c to focus only on the closed patterns obtained by the closure operator to $\mathcal{D}_c$, and such closed patterns are often called the *closed-on-the-positives*. Hereafter we abbreviate $\Gamma(\boldsymbol{x}, \mathcal{D}_c)$ as $\Gamma_c(\boldsymbol{x})$ and called it the *closure of $\boldsymbol{x}$ on the positives*.

[2] The remaining one is that $R_c(\boldsymbol{x}) = 0$ when $p(\boldsymbol{x} \mid c) = p(\boldsymbol{x} \mid \neg c)$, i.e. $R_c(\boldsymbol{x})$ reaches zero in all the points on the diagonal TPR = FPR [4,17].

In addition, we introduce another set-inclusion-based constraints called productivity, whose original version is defined with confidence (i.e. $R_c(\boldsymbol{x})$ is fixed as $p(c \mid \boldsymbol{x})$) [2,20]. Productivity is defined as follows:[3]

Definition 2. *Let c be a class of interest. Then, for a pair of patterns $\boldsymbol{x}$ and $\boldsymbol{x}'$ in $\mathcal{P}$, $\boldsymbol{x}$ is* weaker than *$\boldsymbol{x}'$ iff $\boldsymbol{x} \supset \boldsymbol{x}'$ and $R_c(\boldsymbol{x}) \leq R_c(\boldsymbol{x}')$. A pattern $\boldsymbol{x}$ is* productive *iff $\boldsymbol{x}$ is not weaker than any sub-pattern of $\boldsymbol{x}$.* □

In the literature [5,6,12], a pattern $\boldsymbol{x}$ is said to be *dominated* by another pattern $\boldsymbol{x}'$ iff $\mathcal{D}_c(\boldsymbol{x}) \subseteq \mathcal{D}_c(\boldsymbol{x}')$ and $\mathcal{D}_{\neg c}(\boldsymbol{x}) \supseteq \mathcal{D}_{\neg c}(\boldsymbol{x}')$, and a pattern $\boldsymbol{x}$ is *relevant* iff $\boldsymbol{x}$ is not dominated by any other pattern. Garriga et al. [5] derived a condition equivalent to this relevance: *a pattern $\boldsymbol{x}$ is relevant iff $\boldsymbol{x}$ is closed on the positives and there is no generalization $\boldsymbol{x}' \subset \boldsymbol{x}$ closed on the positives such that $\mathcal{D}_{\neg c}(\boldsymbol{x}') = \mathcal{D}_{\neg c}(\boldsymbol{x})$*. Here it is straightforward to show that under a dual-monotonic relevance score R_c, productivity implies relevance in the sense above (i.e. productivity is a tighter constraint) among the patterns closed on the positives. From this observation, in this paper, we aim to find top-k productive closed-on-the-positives.

2.4 Branch-and-Bound Pruning in Top-k Mining

Suppose that we conduct a branch-and-bound search for top-k patterns under a dual-monotonic relevance score R_c. Also consider an anti-monotonic upper bound $\overline{R}_c(\boldsymbol{x})$ of $R_c(\boldsymbol{x})$ of a pattern $\boldsymbol{x}$. Then, if it is found that $\overline{R}_c(\boldsymbol{x}) < R_c(\boldsymbol{z})$, where $\boldsymbol{z}$ is the pattern with the k-th greatest score, we can safely prune the subtree rooted by $\boldsymbol{x}$. This pruning exploits the anti-monotonicity of $\overline{R}_c$, which guarantees $R_c(\boldsymbol{x}') \leq \overline{R}_c(\boldsymbol{x}') \leq \overline{R}_c(\boldsymbol{x}) < R_c(\boldsymbol{z})$ for any super-pattern $\boldsymbol{x}'$ of $\boldsymbol{x}$.

Several previous methods obtain the upper bound by considering the most optimistic scenario. Since $R_c(\boldsymbol{x})$ is dual-monotonic, by definition $R_c(\boldsymbol{x})$ is monotonically increasing (resp. decreasing) w.r.t. $p(\boldsymbol{x} \mid c)$ (resp. $p(\boldsymbol{x} \mid \neg c)$), and both $p(\boldsymbol{x} \mid c)$ and $p(\boldsymbol{x} \mid \neg c)$ are anti-monotonic w.r.t. pattern-inclusion. Thus, the most optimistic scenario when extending $\boldsymbol{x}$ into $\boldsymbol{x}'$ is that $p(\boldsymbol{x}' \mid c)$ remains $p(\boldsymbol{x} \mid c)$ and $p(\boldsymbol{x}' \mid \neg c)$ turns to be zero. So a general heuristic for obtaining an upper bound $\overline{R}_c(\boldsymbol{x})$ is to substitute $p(\boldsymbol{x} \mid \neg c) := 0$ into the definition of $R_c(\boldsymbol{x})$.[4] After having the upper bound $\overline{R}_c(\boldsymbol{x})$ where $p(\boldsymbol{x} \mid \neg c)$ is constant at zero, $\overline{R}_c(\boldsymbol{x})$ is always anti-monotonic w.r.t. pattern-inclusion thanks to the dual-monotonicity of R_c. For example, F-score is defined as $\mathrm{F}_c(\boldsymbol{x}) = 2p(c \mid \boldsymbol{x})p(\boldsymbol{x} \mid c)/(p(c \mid \boldsymbol{x}) + p(\boldsymbol{x} \mid c))$, so we obtain its upper bound as $\overline{\mathrm{F}}_c(\boldsymbol{x}) = 2p(\boldsymbol{x} \mid c)/(1 + p(\boldsymbol{x} \mid c))$. The above heuristic is applicable to any dual-monotonic relevance scores.[5]

Then, we translate branch-and-bound pruning into minimum support raising [9]. In top-k mining, we usually use an ordered list of candidate patterns,

[3] Weakness in this definition is also called *strong dominance* in the context of relevant explanation in Bayesian networks [22].

[4] Equivalent substitutions are also possible: $p(c \mid \boldsymbol{x}) := 1$, $p(\neg\boldsymbol{x} \mid \neg c) := 1$, and so on.

[5] However, for some scores whose upper bounds are always high, there would be no chance of pruning. For instance, a relevance score called growth rate [3] $\mathrm{GR}_c(\boldsymbol{x}) = p(\boldsymbol{x} \mid c)/p(\boldsymbol{x} \mid \neg c)$ goes into infinity when substituting $p(\boldsymbol{x} \mid \neg c) := 0$.

called the *candidate list* in this paper, and insert the patterns found in the middle of the search into the list in the descending order of the relevance score. A pattern $\boldsymbol{x}$ cannot stay in the final top-k patterns if $\overline{\mathrm{F}}_c(\boldsymbol{x}) = 2p(\boldsymbol{x} \mid c)/(1 + p(\boldsymbol{x} \mid c)) < \mathrm{F}_c(\boldsymbol{z})$, where $\boldsymbol{z}$ is the k-th pattern. Here we rewrite this condition as: $p(\boldsymbol{x} \mid c) < \mathrm{F}_c(\boldsymbol{z})/(2 - \mathrm{F}_c(\boldsymbol{z})) = U_c(\boldsymbol{z})$. When the relevance score is dual-monotonic, this rewriting is always valid.[6] Here $U_c(\boldsymbol{z})$ works as a threshold for the support of $\boldsymbol{x}$ and leads to minimum support raising. That is, starting with a small value (e.g. $\sigma_{\min} := 1/|\mathcal{D}_c|$), the minimum support is repeatedly updated by $\sigma_{\min} := \max\{U_c(\boldsymbol{z}), \sigma_{\min}\}$ during the search. Thanks to the translation above, we can inherit fast frequent pattern mining algorithms like FP-Growth [8].

2.5 Search Strategies under Productivity

In this subsection, we describe search strategies for handling the productivity constraint in top-k branch-and-bound search. As stated before, top-k mining often uses a candidate list L that stores the current top-k candidate patterns. By the nature of minimum support raising, a new minimum support is set heavily depending on the k-th greatest score in L, and thus the effect of minimum support raising is rather limited if we relax the size limit of L and add the patterns that are not truly productive patterns. For example, consider the prefix enumeration tree in Fig. 1, where $\boldsymbol{x} = \{\mathsf{A}, \mathsf{C}, \mathsf{D}\}$ is visited before $\boldsymbol{x}' = \{\mathsf{A}, \mathsf{D}\}$ in a depth-first search. When visiting $\boldsymbol{x}$, it is inevitable to add $\boldsymbol{x}$ into L since it is uncertain whether $\boldsymbol{x}$ is weaker than $\boldsymbol{x}'$ at the moment. Contrastingly, we should keep the candidate list L as compact as possible, i.e. wherever possible we should filter out immediately the patterns that are not truly productive patterns.

Fortunately, if a pattern $\boldsymbol{x}$ is guaranteed to be visited after all sub-patterns of $\boldsymbol{x}$ have been visited, we can filter out $\boldsymbol{x}$ immediately if $\boldsymbol{x}$ is weaker than some existing pattern in the candidate list L. Typically breadth-first search, where shorter patterns are visited earlier, enables this filtering. Also for the same purpose, recent work by Grosskreutz et al. [6] proposes a memory-efficient method based on iterative deepening where 'depth' is the maximum size of patterns to be found. Furthermore, in this paper, following RP-Growth [10], we adopt memory-efficient depth-first traversal over a search space like suffix enumeration trees illustrated in the introduction. We take this strategy because the overhead of iterative deepening seems not ignorable in finding long patterns.

Furthermore, we also inherit aggressive pruning from RP-Growth. This pruning is based on an extended notion of weakness defined as follows:

Definition 3. *Let c be a class of interest, and $\boldsymbol{x}$, $\boldsymbol{x}'$ be a pair of patterns in $\mathcal{P}$. Then, $\boldsymbol{x}$ is prunably weaker than $\boldsymbol{x}'$ iff $\boldsymbol{x} \supset \boldsymbol{x}'$ and $\overline{R}_c(\boldsymbol{x}) \leq R_c(\boldsymbol{x}')$.* □

If a pattern $\boldsymbol{x}$ is prunably weaker than some pattern $\boldsymbol{x}'$ in the current candidate list, any super-pattern of $\boldsymbol{x}$ is also weaker than $\boldsymbol{x}'$, and thus we can safely prune the subtree rooted by $\boldsymbol{x}$ in finding top-k productive patterns.

[6] For many scores, it is possible to have $U_c(\boldsymbol{z})$ in closed form. A typical exception is the case with information gain, since it includes the entropy function in its definition. In such a case, the threshold in the right hand side should be numerically solved.

3 The Proposed Method

3.1 Suffix-Preserving Closure Extension

Based on the concepts and the techniques in Section 2, from now, we propose an efficient exact algorithm for top-k productive closed-on-the-positives. We have seen that, via minimum support raising, branch-and-bound pruning can be plugged into frequent closed pattern mining algorithms like LCM [19]. LCM is the first algorithm that introduces *prefix-preserving closure extension* (PPC extension, for short) for avoiding duplicate visits to a pattern. So we aim to run LCM over suffix enumeration trees, by introducing a mirrored version of PPC extension, called *suffix-preserving closure extension* (SPC extension).

Definition 4. *Let c be a class of interest, $\mathcal{D}_c$ the transactions in c, $\boldsymbol{x}$ a pattern in $\mathcal{P}$, x an item in $\mathcal{X}$ and $\Sigma_x(\boldsymbol{x}) = \boldsymbol{x} \cap \{x' \mid x \prec x'\}$. Then, the suffix-preserving closure extension of a pattern $\boldsymbol{x}$ by an item x is defined as $\boldsymbol{x}^* = \Gamma_c(\{x\} \cup \boldsymbol{x})$ such that (i) $x \notin \boldsymbol{x}$, (ii) $x \prec \mathrm{core}(\boldsymbol{x})$ and (iii) $\Sigma_x(\boldsymbol{x}^*) = \Sigma_x(\boldsymbol{x})$, where $\mathrm{core}(\boldsymbol{x})$ is the maximum item x w.r.t. $\prec$ such that $\mathcal{D}_c(\boldsymbol{x} \cap \{x' \mid x \preceq x'\}) = \mathcal{D}_c(\boldsymbol{x})$.* □

Here $\Sigma_x(\boldsymbol{x})$ is the suffix of $\boldsymbol{x}$ starting from the successor of x, and Condition (iii) says that such a suffix must be preserved between the original pattern and the extended one. To handle core$(\cdot)$ procedurally, a useful property is known [19]: *in an SPC extension $\boldsymbol{x}^* = \Gamma_c(\{x\} \cup \boldsymbol{x})$, $\mathrm{core}(\boldsymbol{x}^*)$ is exactly the added item x.* In this paper, for each pattern $\boldsymbol{x}$, we consider to record all of such added items in a chain of SPC extensions producing $\boldsymbol{x}$ and call them the *core items* in $\boldsymbol{x}$.

To illustrate, let us consider nine example transactions shown in Fig. 2 (top-left), each belongs to one of two classes $\{+, -\}$. Here we are interested in class $+$. Then, we have a total order $\mathsf{A} \prec \mathsf{B} \prec \mathsf{C} \prec \mathsf{E} \prec \mathsf{D}$ as a descending order of F-score in Fig. 2 (bottom-left). Fig. 2 (top-right) is an enumeration tree obtained by the exhaustive applications of SPC extension, where, at each branch, the core items to be added are chosen in the ascending order w.r.t. $\prec$. Such a tree is hereafter called an *SPC enumeration tree*. In Fig. 2, the core items are underlined, and among them, the doubly underlined ones are the core items which are added last. For example, given an empty pattern $\boldsymbol{x} = \emptyset$, we apply SPC extension by item C to the example transactions and obtain $\boldsymbol{x}^* = \Gamma_c(\{\mathsf{C}\} \cup \emptyset) = \{\mathsf{A}, \mathsf{C}\}$. In this case, C is a (the last) core item in $\{\mathsf{A}, \mathsf{C}\}$, while A is not.

3.2 Properties Related to SPC Extension

In this subsection, we show some key properties related to SPC extension in finding top-k productive closed-on-the-positives. Before that, we first obtain a tree like Fig. 2 (bottom-right) by extracting core items from an SPC enumeration tree, and call it a *core enumeration tree*. Let T and T_{core} be an SPC enumeration tree and its core enumeration tree, respectively. It is then easy to show that for a node (pattern) $\boldsymbol{x}$ in T_{core}, the corresponding node in T is its closure on the positives, i.e. $\Gamma_c(\boldsymbol{x})$. Also we see that T and T_{core} are isomorphic from the way of adding core items. For example, for the node $\{\mathsf{B}, \mathsf{D}\}$ in the core enumeration tree

Class	Transactions
+	{A, B, D, E}
+	{A, B, C, D, E}
+	{A, C, D, E}
+	{A, B, C}
+	{B}
−	{A, B, D, E}
−	{B, C, D}
−	{C, D, E}
−	{B, D, E}

Item x	$p(x \mid +)$	$p(x \mid -)$	$\mathrm{F}_+(x)$
A	0.8	0.25	0.780
B	0.8	0.75	0.627
C	0.6	0.50	0.571
D	0.6	1.00	0.462
E	0.6	0.75	0.511

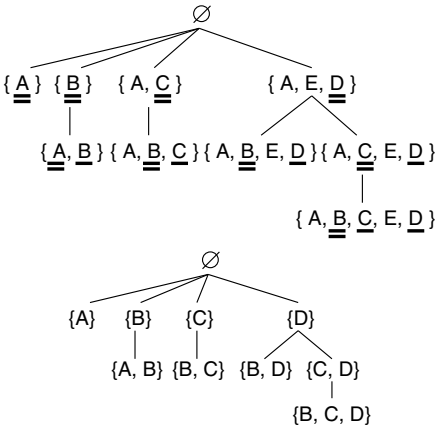


Fig. 2. Example transactions (top-left), F-scores of items (bottom-left), the enumeration tree with SPC extension (top-right) and its core enumeration tree (bottom-right)

in Fig. 2, we have {A, B, E, D} in the SPC enumeration tree in Fig. 2. Besides, a core enumeration tree is a suffix enumeration tree with some subtrees being removed. Indeed, Fig. 2 (bottom-right) is the suffix enumeration tree in Fig. 1 with several leaves ({A, C}, {A, D}, etc.) being removed. This property comes from Condition (ii) of SPC extension (we are only allowed to add, as a new core item, a predecessor of the last core item) together with the fact that the core items to be added at each branch are chosen ascendingly w.r.t. $\prec$.

From the observations above, we will prove a couple of propositions that justify depth-first search with SPC extension (i.e. depth-first search over an SPC enumeration tree) in finding top-k productive closed-on-the-positives. We write $\boldsymbol{x} \sqsubset \boldsymbol{x}'$ iff $\boldsymbol{x}$ is visited before $\boldsymbol{x}'$ is visited, and note that for two distinct patterns in an SPC enumeration tree T, the corresponding patterns in T's core enumeration tree T_{core} are distinct, and vice versa (since T and T_{core} are isomorphic).

Proposition 1. *Let $\boldsymbol{x}_1^*$ and $\boldsymbol{x}_2^*$ be two distinct patterns in an SPC enumeration tree T. Also let $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, respectively, be the corresponding patterns in T's core enumeration tree T_{core}. Then, if $\boldsymbol{x}_1 \subset \boldsymbol{x}_2$, we have $\boldsymbol{x}_1^* \subset \boldsymbol{x}_2^*$ and $\boldsymbol{x}_1^* \sqsubset \boldsymbol{x}_2^*$.* □

Proof. First, from $\boldsymbol{x}_1^* = \Gamma_c(\boldsymbol{x}_1)$ and $\boldsymbol{x}_2^* = \Gamma_c(\boldsymbol{x}_2)$, it is easy to see that $\boldsymbol{x}_1 \subset \boldsymbol{x}_2 \Rightarrow \boldsymbol{x}_1^* \subset \boldsymbol{x}_2^*$ since $\boldsymbol{x}_1 \subseteq \boldsymbol{x}_2 \Rightarrow \Gamma_c(\boldsymbol{x}_1) \subseteq \Gamma_c(\boldsymbol{x}_2)$ from the monotonicity of the closure operator [15]. We then have $\boldsymbol{x}_1 \subset \boldsymbol{x}_2 \Rightarrow \boldsymbol{x}_1 \sqsubset \boldsymbol{x}_2$ since any core enumeration tree is a part of a suffix enumeration tree. Also $\boldsymbol{x}_1^* \sqsubset \boldsymbol{x}_2^* \Leftrightarrow \boldsymbol{x}_1 \sqsubset \boldsymbol{x}_2$ holds since T and T_{core} are isomorphic. It immediately follows that $\boldsymbol{x}_1 \subset \boldsymbol{x}_2 \Rightarrow \boldsymbol{x}_1^* \sqsubset \boldsymbol{x}_2^*$. □

Proposition 2. *Let $\boldsymbol{x}_1^*$ and $\boldsymbol{x}_2^*$ be two distinct patterns in an SPC enumeration tree T. Also let $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, respectively, be the corresponding patterns in T's core enumeration tree T_{core}. Also suppose that $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are not subsets of each other. Then, $\boldsymbol{x}_1^* \subset \boldsymbol{x}_2^* \Rightarrow \boldsymbol{x}_1^* \sqsubset \boldsymbol{x}_2^*$.* □

Proof. We prove this by contraposition. First assume that $\boldsymbol{x}_1^* \sqsupset \boldsymbol{x}_2^*$ and equivalently $\boldsymbol{x}_1 \sqsupset \boldsymbol{x}_2$. Also consider the common suffix $\boldsymbol{y}$ of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. Furthermore, let x (resp. x') be the maximum item w.r.t. $\prec$ in $\boldsymbol{x}_1 \setminus \boldsymbol{y}$ (resp. $\boldsymbol{x}_2 \setminus \boldsymbol{y}$). Then, we necessarily have $x' \prec x$ since $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are not subsets of each other, and $\boldsymbol{x}_2 \sqsubset \boldsymbol{x}_1$. From Condition (iii) of SPC extension, $\Gamma_c(\{x'\} \cup \boldsymbol{y})$ never contains x, and accordingly neither does $\boldsymbol{x}_2^*$ $(= \Gamma_c(\boldsymbol{x}_2))$. On the contrary, $\boldsymbol{x}_1^*$ $(= \Gamma_c(\boldsymbol{x}_1))$ always contains x since x is its core item. Therefore, $\boldsymbol{x}_1^*$ can never be a subset of $\boldsymbol{x}_2^*$. Now we have $\boldsymbol{x}_1^* \subset \boldsymbol{x}_2^* \Rightarrow \boldsymbol{x}_1^* \sqsubset \boldsymbol{x}_2^*$. □

Proposition 3. *When a pattern $\boldsymbol{x}$ is visited in a depth-first search with SPC extension, all of $\boldsymbol{x}$'s sub-patterns have already been visited.* □

Proof. It is sufficient to show that $\boldsymbol{x}_1^* \subset \boldsymbol{x}_2^* \Rightarrow \boldsymbol{x}_1^* \sqsubset \boldsymbol{x}_2^*$, where $\boldsymbol{x}_1^*$ and $\boldsymbol{x}_2^*$ be two distinct patterns in an SPC enumeration tree, say T. Here we first assume that $\boldsymbol{x}_1^* \subset \boldsymbol{x}_2^*$. Also let $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, respectively, be the corresponding patterns in T's core enumeration tree T_{core}. Then, we consider three exhaustive cases: (1) $\boldsymbol{x}_1 \supset \boldsymbol{x}_2$, (2) $\boldsymbol{x}_1 \subset \boldsymbol{x}_2$, and (3) $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are not subsets of each other. The first case is incompatible with the assumption $\boldsymbol{x}_1^* \subset \boldsymbol{x}_2^*$ since $\boldsymbol{x}_2 \subset \boldsymbol{x}_1 \Rightarrow \boldsymbol{x}_2^* \subset \boldsymbol{x}_1^*$ from Proposition 1. In the second case, we have $\boldsymbol{x}_1^* \sqsubset \boldsymbol{x}_2^*$ again from Proposition 1. In the last case, we also have $\boldsymbol{x}_1^* \sqsubset \boldsymbol{x}_2^*$ from Proposition 2. □

Now, as described in Section 2.5, we can filter out a new pattern $\boldsymbol{x}$ immediately if $\boldsymbol{x}$ is weaker than some existing pattern in the candidate list. So the size of the list is kept being k except when there are ties at the bottom.[7]

3.3 Algorithm Description

In this subsection, we present the proposed algorithm for finding top-k productive closed-on-the-positives. First, we introduce four global variables c, k, $\sigma_{\min}$ and L, which stand for the class of interest, the number of patterns to be output, the minimum support and the candidate list, respectively. The values of c and k are given by the user. On the other hand, $\sigma_{\min}$ and L are respectively initialized as $1/|\mathcal{D}_c|$ and $\emptyset$. The total order $\prec$ among items is considered as a descending order of the relevance score R_c in use.[8] The central procedure is Grow, shown in Algorithm 1, and we call Grow$(\emptyset)$ to run the algorithm. After termination, the final top-k patterns are stored in the candidate list L.

Given the current pattern $\boldsymbol{x}$, Grow$(\boldsymbol{x})$ works as follows. First, we compute the set B of items that satisfy Conditions (i) and (ii) of SPC extension (Line 1). Note that B contains all possible items if $\boldsymbol{x} = \emptyset$. Then, we try to branch by each item x in B ascendingly w.r.t. $\prec$ (Line 2). We create a new pattern $\boldsymbol{x}^*$ (Line 4) with pruning by the minimum support (Line 3) and by Condition (iii) of SPC extension (Line 5). After computing the relevance score of $\boldsymbol{x}^*$ (Line 6), we add

[7] Of course, such ties do not affect the minimum support threshold to be raised.

[8] This total ordering has a preferable side-effect that, as is seen in Fig. 1 (right) and Fig. 2 (top-right), we try the combinations of promising items earlier in a suffix/SPC enumeration tree and hence the minimum support tends to be raised quickly.

Algorithm 1. GROW($\boldsymbol{x}$)

Require: $\boldsymbol{x}$: the current pattern
1: $B := \{x \mid x \notin \boldsymbol{x}$ and x is a predecessor of the last core item added into $\boldsymbol{x}\}$
2: **for each** $x \in B$ enumerated in the ascending order of $\prec$ **do**
3: **if** $p(\{x\} \cup \boldsymbol{x} \mid c) < \sigma_{\min}$ **then continue**
4: $\boldsymbol{x}^* := \Gamma_c(\{x\} \cup \boldsymbol{x})$
5: **if** $\Sigma_x(\boldsymbol{x}^*) \neq \Sigma_x(\boldsymbol{x})$ **then continue**
6: Compute $R_c(\boldsymbol{x}^*)$
7: **if** $\boldsymbol{x}^*$ is not weaker than any patterns in L **then**
8: Insert $\boldsymbol{x}^*$ into L following the descending order of R_c
9: **if** $|L| \geq k$ **then**
10: Remove the patterns with the score below the k-th pattern's score from L
11: $\sigma_{\min} := \max\{U_c(\boldsymbol{z}), \sigma_{\min}\}$, where $\boldsymbol{z}$ is the k-th pattern in L
12: **end if**
13: **end if**
14: Call GROW($\boldsymbol{x}^*$) **if** $\boldsymbol{x}$ is not prunably weaker than any patterns in L
15: **end for**

$\boldsymbol{x}^*$ into the candidate list L if $\boldsymbol{x}^*$ is not weaker than any existing pattern in L (Lines 7–13). If L is full, we replace with $\boldsymbol{x}^*$ the patterns that are less relevant than $\boldsymbol{x}^*$ (Line 10) and update the minimum support (Line 11). Here $U_c(\boldsymbol{z})$ is a new threshold based on $\boldsymbol{z}$'s score (Section 2.4). Finally, for $\boldsymbol{x}^*$ having passed the filter on prunable weakness, we call GROW recursively (line 14).

4 Experimental Results

We conducted an experiment to confirm the efficiency of the proposed algorithm. The datasets are originally collected in UCI Machine Learning Repository and we used a preprocessed version available from `http://dtai.cs.kuleuven.be/CP4IM/datasets/`. The statistics are summarized in Table 1.

In the experiment, we compare the search strategies described in Section 2.5, i.e. breadth-first search (BFS), depth-first search over a prefix/suffix enumeration tree (DFS(Prefix)/DFS(Suffix)) and iterative deepening (ID) recently proposed by Grosskreutz et al. [6].[9] We specified $k = 10$ as the number of productive closed-on-the-positives to be found. F-score is used as the relevance score. The result is shown in Fig. 3. The y-axis indicates the number of visited nodes, which equals the number of times the closure operator was applied. The measurements at the y-axis are presented in logarithmic scale. From Fig. 3, we first see that DFS(Suffix) (the proposed method) outperforms DFS(Prefix) for most of datasets.

[9] We also implemented an optimization described in Section 5.1 of [6], where we skip the depths for which no patterns exist by keeping track the length of the shortest pattern exceeding the current depth at each iteration. All implementations are written in Java, and for comparison, we did not use elaborate data structures like FP-trees but pseudo-projection-style databases [16].

Table 1. Statistics on the datasets. "#Trs." indicates the number of transactions.

Dataset	#Trs.	#Items	Dataset	#Trs.	#Items	Dataset	#Trs.	#Items
anneal	812	93	hypothyroid	3,247	88	splice-1	3,190	287
audiology	216	148	kr-vs-kp	3,196	73	tic-tac-toe	958	27
australian-credit	653	125	lymph	148	68	vote	435	48
german-credit	1,000	112	mushroom	8,124	119	zoo-1	101	36
heart-cleveland	296	95	primary-tumor	336	31			
hepatitis	137	68	soybean	630	50			

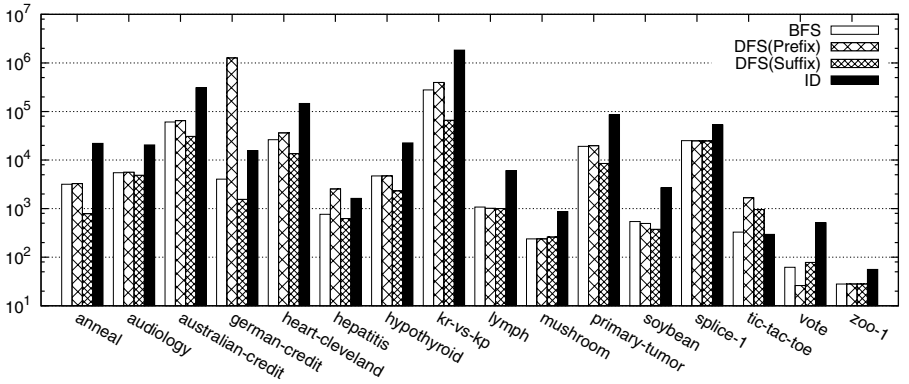


Fig. 3. The result of a comparison among search strategies

In particular, DFS(Prefix) works poorly for german-credit. In addition, as stated before, the overhead of iterative deepening is not ignorable in this experiment. On comparison between BFS and DFS(Suffix), we can say that for some small datasets like tic-tac-toe, BFS works better than DFS(Suffix), but for the datasets where the search is costly, DFS(Suffix) outperforms BFS (remind here that the y-axis is in logarithmic scale). Totally, the proposed method DFS(Suffix) stably runs fast in comparison with the other search strategies.

5 Conclusion

In this paper, we proposed an efficient exact algorithm for finding top-k discriminative patterns that are not redundant and would be of value at a later step in prediction or knowledge discovery. Redundancy among discriminative patterns are eliminated by two set-inclusion-based constraints, closedness and productivity. Such constraints are efficiently tested with suffix-preserving closure extension under dual-monotonic relevance scores. We showed formally and empirically that the proposed algorithm successfully keeps compact the list of candidate top-k patterns during the search and consequently high the minimum support threshold. In future, we would like to extend the proposed algorithm for more complex data such as sequences.

Acknowledgments. This work was supported in part by JSPS KAKENHI Grant Number 24700141.

References

1. Bay, S.D., Pazzani, M.J.: Detecting group differences: mining contrast sets. Data Mining and Knowledge Discovery 5, 213–246 (2001)
2. Bayardo, R., Agrawal, R., Gunopulos, D.: Constraint-based rule mining in large, dense databases. Data Mining and Knowledge Discovery 4, 217–240 (2000)
3. Dong, G., Li, J.: Efficient mining of emerging patterns: discovering trends and differences. In: Proc. of KDD 1999, pp. 43–52 (1999)
4. Fürnkranz, J., Gamberger, D., Lavrač, N.: Foundations of Rule Learning. Springer (2012)
5. Garriga, G.C., Kralj, P., Lavrač, N.: Closed sets for labeled data. J. of Machine Learning Research 9, 559–580 (2008)
6. Grosskreutz, H., Paurat, D.: Fast and memory-efficient discovery of the top-k relevant subgroups in a reduced candidate space. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (eds.) ECML PKDD 2011, Part I. LNCS, vol. 6911, pp. 533–548. Springer, Heidelberg (2011)
7. Grosskreutz, H., Rüping, S., Wrobel, S.: Tight optimistic estimates for fast subgroup discovery. In: Proc. of ECLM/PKDD 2008, pp. 440–456 (2008)
8. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: Proc. of SIGMOD 2000, pp. 1–12 (2000)
9. Han, J., Wang, J., Lu, Y., Tzvetkov, P.: Mining top-K frequent closed patterns without minimum support. In: Proc. of ICDM 2002, pp. 211–218 (2002)
10. Kameya, Y., Sato, T.: RP-growth: top-k mining of relevant patterns with minimum support raising. In: Proc. of SDM 2012, pp. 816–827 (2012)
11. Kralj Novak, P., Lavrač, N., Webb, G.I.: Supervised descriptive rule discovery: a unifying survey of contrast set, emerging pattern and subgroup mining. J. of Machine Learning Research 10, 377–403 (2009)
12. Lavrač, N., Gamberger, D., Jovanoski, V.: A study of relevance for learning in deductive databases. J. of Logic Programming 40, 215–249 (1999)
13. Morishita, S., Sese, J.: Traversing itemset lattices with statistical metric pruning. In: Proc. of PODS 2000, pp. 226–236 (2000)
14. Nijssen, S., Guns, T., De Raedt, L.: Correlated itemset mining in ROC space: a constraint programming approach. In: Proc. of KDD 2009, pp. 647–656 (2009)
15. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering frequent closed itemsets for association rules. In: Beeri, C., Bruneman, P. (eds.) ICDT 1999. LNCS, vol. 1540, pp. 398–416. Springer, Heidelberg (1998)
16. Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M.C.: Prefixspan: mining sequential patterns efficiently by prefix-projected pattern growth. In: Proc. of ICDE 2001, pp. 215–224 (2001)
17. Piatetsky-Shapiro, G.: Discovery, analysis, and presentation of strong rules. In: Knowledge Discovery in Databases, pp. 229–248. AAAI Press (1991)
18. Soulet, A., Crémilleux, B., Rioult, F.: Condensed representation of emerging patterns. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 127–132. Springer, Heidelberg (2004)

19. Uno, T., Asai, T., Uchida, Y., Arimura, H.: An efficient algorithm for enumerating closed patterns in transaction databases. In: Suzuki, E., Arikawa, S. (eds.) DS 2004. LNCS (LNAI), vol. 3245, pp. 16–31. Springer, Heidelberg (2004)
20. Webb, G.I.: Discovering significant patterns. Machine Learning 68, 1–33 (2007)
21. Wrobel, S.: An algorithm for multi-relational discovery of subgroups. In: Komorowski, J., Żytkow, J.M. (eds.) PKDD 1997. LNCS, vol. 1263, pp. 78–87. Springer, Heidelberg (1997)
22. Yuan, C., Lim, H., Lu, T.C.: Most relevant explanation in Bayesian networks. J. of Artificial Intelligence Research 42, 309–352 (2011)
23. Zimmermann, A., De Raedt, L.: Cluster grouping: from subgroup discovery to clustering. Machine Learning 77, 125–159 (2009)

Stream Mining of Frequent Patterns from Delayed Batches of Uncertain Data

Fan Jiang and Carson Kai-Sang Leung*

University of Manitoba, Winnipeg, MB, Canada
kleung@cs.umanitoba.ca

Abstract. Streams of data can be continuously generated by sensors in various real-life applications such as environment surveillance. Partially due to the inherited limitation of the sensors, data in these streams can be uncertain. To discover useful knowledge in the form of frequent patterns from streams of uncertain data, a few algorithms have been developed. They mostly use the sliding window model for processing and mining data streams. However, for some applications, other stream processing models such as the time-fading model are more appropriate. Moreover, batches of data in the stream may be delayed and not arrived in the intended order. In this paper, we propose mining algorithms that use the time-fading model to mine frequent patterns when these batches in the streams of uncertain data were delayed and arrived out of order.

1 Introduction and Related Works

Frequent pattern mining [2,6,14,21] helps to discover implicit, previously unknown, and potentially useful knowledge in the form of frequently occurring sets of items that are embedded in the data. Nowadays, the automation of measurements and data collection is producing tremendously huge volumes of data. For instance, the development and increasing use of a large number of sensors (e.g., acoustic, chemical, electromagnetic, mechanical, optical radiation and thermal sensors) for various real-life applications (e.g., environment surveillance, security, manufacturing systems) have led to *data streams*. To discover useful knowledge from these streaming data, several mining algorithms [9,10] have been proposed. In general, mining frequent patterns from dynamic data streams is more challenging than mining from traditional static transaction databases due to the following characteristics of data streams:

1. *Data streams are continuous and unbounded*. As such, we no longer have the luxury to scan the streams multiple times. Once the streams flow through, we lose them. We need some techniques to capture important contents of the streams. For instance, *sliding windows* capture the contents of a fixed number (w) of batches (e.g., w most recent batches) in the streams. Alternatively, *time-fading windows* capture contents of all the batches but weight recent data heavier than older data (i.e., monotonically decreasing weights from recent to older data).

* Corresponding author.

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 209–221, 2013.

2. *Data in the streams are not necessarily uniformly distributed.* As such, a currently infrequent pattern may become frequent in the future and vice versa. We have to be careful not to prune infrequent patterns too early; otherwise, we may not be able to get complete information such as frequencies of some patterns (as it is impossible to recall those pruned patterns).

Many existing mining algorithms discover frequent patterns from *precise* data (in either static databases [5,8,17,20] or dynamic data streams [7,22]), in which users definitely know whether an item is present in, or absent from, a transaction in the data. However, there are situations in which users are uncertain about the presence or absence of items. For example, due to dynamic errors (e.g., inherited measurement inaccuracies, sampling frequency of the sensors, deviation caused by a rapid change of the measured property over time such as drift or noise, wireless transmission errors, network latencies), streaming data collected by sensors may be uncertain. As such, users may highly suspect but cannot guarantee that an item x is present in a transaction t_i. The uncertainty of such suspicion can be expressed in terms of *existential probability* $P(x, t_i) \in (0, 1]$, which indicates the likelihood of x being present in t_i in probabilistic data. With this notion, every item in t_i in (static databases or dynamic streams of) precise data can be viewed as an item with a 100% likelihood of being present in t_i. A challenge of handling these uncertain data is the huge number of "possible worlds" (e.g., there are two "possible worlds" for an item x in t_i: (i) $x \in t_i$ and (ii) $x \notin t_i$). Given m independent items in all transactions, there are $O(2^m)$ "possible worlds" [11].

In past few years, several mining algorithms have been proposed to discover frequent patterns from uncertain data. However, most of them (e.g., UH-Mine [1], UV-Eclat [3], U-Eclat [4], UF-growth [16], PUF-growth [18], U-VIPER [19]) mine frequent patterns from *static databases*—but *not* dynamic streams—of uncertain data. For the algorithms that mine from data streams (e.g., UF-streaming [13]), they use the *sliding window model*. While the sliding window model is useful for situations where users are interested in discovering frequent patterns from a fixed-size time window (e.g., frequent patterns observed in the last 24 hours), there are also other situations where users are interested in a variable-size time window capturing all historical data with stronger preference on recent data than older one (i.e., *time-fading model*). To handle these situations, we designed in DaWaK 2011 an algorithm [12,15] that uses the time-fading model to mine frequent patterns from streaming uncertain data. Such an algorithm works well when batches of streaming data come according to an ordered sequence. However, due to various factors (e.g., network congestion), batches of streaming data may get delayed. Hence, a logical question is: How to mine frequent patterns from these out-of-order (out-of-sequence) batches? *Key contributions* of this paper are our two new algorithms for mining frequent patterns from these out-of-order batches of streaming uncertain data.

This paper is organized as follows. The next section describes how to mine frequent patterns from *ordered* batches of uncertain data streams. In Section 3,

we introduce our new tree-based mining algorithms that also use the time-fading model but to discover frequent patterns from *out-of-order* batches of streaming uncertain data. Evaluation results are shown in Section 4. Finally, Section 5 presents the conclusions.

2 Mining Ordered Batches of Uncertain Data Streams

In DaWaK 2011, we designed an algorithm [12,15]—which we refer as **TUF-streaming(Ordered)** in the remainder of the current paper—to mine *ordered* batches (cf. *out-of-order* batches) of uncertain data streams using a time-fading window.

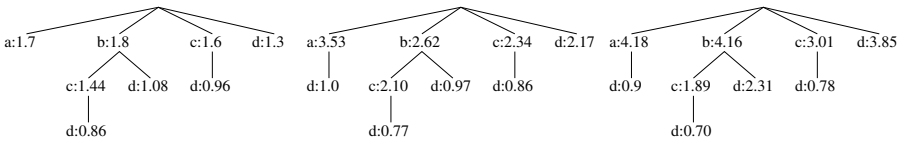
TUF-streaming(Ordered) first calls UF-growth [16] to find "frequent" patterns from the current batch of transactions in the streams using *preMinsup* as the threshold. A pattern is *"frequent"* (more precisely, *subfrequent*) if its expected support $\geq$ *preMinsup*. Note that, although users are interested in truly frequent patterns (i.e., patterns with expected support $\geq$ user-specified *minsup* threshold, where *minsup* > *preMinsup*), *preMinsup* is used in attempt to avoid pruning a pattern too early. This is important because data in the continuous streams are not necessarily uniformly distributed. To elaborate, UF-growth constructs a UF-tree to capture contents of uncertain data. Each tree node keeps an item x, its existential probability $P(x, t_i)$, and its occurrence count. The UF-tree is constructed in a similar fashion to that of the FP-tree [8] except that nodes in the UF-tree are merged and shared only if they represent the same x and $P(x, t_i)$. Once the UF-tree is constructed, UF-growth extracts appropriate tree paths to mine frequent patterns using the "possible world" interpretation [11]. When items within a pattern X are independent, the *expected support* of X in a batch B_i can be computed by summing (over all transactions $t_1, ..., t_{|B_i|}$) the product of existential probabilities of items within X [11]:

$$expSup(X, B_i) = \sum_{j=1}^{|B_i|} \left(\prod_{x \in X} P(x, t_j) \right). \tag{1}$$

TUF-streaming(Ordered) then stores the mined "frequent" patterns and their expected support values in a tree structure (called UF-stream), in which each tree node X keeps a single value. The expected support of X for all T batches is computed as follows:

$$expSup\left(X, \cup_{i=1}^{T} B_i\right) = \left[expSup\left(X, \cup_{i=1}^{T-1} B_i\right) \times \alpha\right] + expSup(X, B_T), \tag{2}$$

where α is the time-fading factor with value falls into the range of (0,1]. A low α factor expresses stronger preference on recent data than older one. When a new batch flows in, TUF-streaming(Ordered) assigns lighter weights to old batches than recent batches. This process is repeated for each batch in the stream.



(a) After mining B_1. (b) After mining B_2. (c) After mining B_3.

Fig. 1. The UF-stream structures for the TUF-streaming(Ordered) algorithm

Example 1. Consider the following stream of uncertain data:

Batches	Transactions	Contents
B_1	t_1	{a:0.7, d:0.1, e:0.4}
	t_2	{a:1.0, b:0.9, c:0.8, d:0.6}
	t_3	{b:0.9, c:0.8, d:0.6}
B_2	t_4	{a:1.0, c:0.1, d:0.7}
	t_5	{a:1.0, d:0.3, e:0.1}
	t_6	{b:1.0, c:0.8}
B_3	t_7	{a:1.0, c:0.9, d:0.3}
	t_8	{b:0.9, d:0.8}
	t_9	{b:0.9, d:0.8, e:0.7}

Here, each transaction contains items and their corresponding existential probabilities, e.g., $P(a, t_1)$=0.7. Let the user-specified *minsup* threshold be 1.0. When using the time-fading model, TUF-streaming(Ordered) uses Equation (2) to compute expected support values. For instance, Fig. 1(a) shows the expected support values stored in the UF-stream structure after mining "frequent" patterns from batch B_1.

Afterwards (say, after mining B_i for $i \geq 2$), instead of appending the expected support values of "frequent" patterns mined from B_i, the algorithm modified the stored value. For instance, after mining B_2, TUF-streaming(Ordered) stores their sum 1.44α+0.8 $\approx$ 2.10 as $expSup(\{b, c\}, B_1 \cup B_2)$ in Fig. 1(b). Similarly, after mining B_3, TUF-streaming(Ordered) stores their sum $(1.44\alpha + 0.8)\alpha + 0$ $\approx$ 1.89 as $expSup(\{b, c\}, B_1 \cup B_2 \cup B_3)$ in Fig. 1(c). □

3 Mining Out-of-Order Batches of Uncertain Data Streams

The TUF-streaming(Ordered) algorithm works well when batches arrive in an order that is generated. In other words, batches of streaming data come according to an ordered sequence (say, Batch B_i arrives before Batch B_j for any $i < j$). However, each batch of uncertain streaming data can be of a different size or contain different number of transactions. As the contents of each batch may vary, it is possible that a later batch may arrive before an earlier batch. Moreover, various other factors (e.g., network congestion) may also contribute to the delay of some batches so that a later batch may arrive before an earlier batch. If it happens, how could we mine frequent patterns from these out-of-order batches?

A natural approach to mine frequent patterns from the out-of-order batches is to keep a long list of expected support values. Specifically, when a batch B_i arrives and "frequent" patterns are found from B_i, the expected support of each of these patterns is then put in an appropriate position of the list of expected support values for the pattern (based on the available information about the batch B_i such as the batch sequence i or the timestamp). Then, when all out-of-order batches arrive (at a later time), the expected support of a pattern X from B_1 to B_T can be computed using the following equation:

$$expSup\left(X, \cup_{i=1}^{T} B_i\right) = \sum_{i=1}^{T} \left(expSup(X, B_i) \times \alpha^{T-i}\right), \quad (3)$$

where α is the time-fading factor with value falls into the range of (0,1]. This equation sums the weighted expected support values of each batch. If there is any delay in the arrival of some batches (i.e., on the dual side, there are some early arrivals), this natural approach stores the relevant information (e.g., each "frequent" pattern X and its expected support value $expSup(X, B_e)$ found in the early-arrival batch B_e) in the UF-stream structure. However, the approach delays the computation of expected support $exupSup(X, \cup_{i=1}^{T} B_i)$ of X from B_1 to B_T until all these T batches arrive. See the following example.

Example 2. Let us revisit Example 1. If batch B_3 arrives before B_1 and B_2, the above approach (i) processes B_3, (ii) computes the expected support values of all "frequent" patterns in B_3, and (iii) stores these values in the 3rd position of the list of expected support values in the tree node of the UF-stream structure. Then, when batch B_1 arrives, the approach (i) processes B_1, (ii) computes the expected support values of all "frequent" patterns in B_1, and stores (iii) these values in the 1st position of the list of expected support values in a similar fashion. The approach keeps track of which batches have been processed. When all of the batches have arrived, the natural approach computes expected support solely based on the expected supports stored in the list by using Equation (3). □

3.1 Handling Batches with a Fixed-Size List

There are several potential problems associated with the above approach. First, the list can be very long and potentially unbounded. Consider a situation in which there is severe delay for an early batch (say, B_2 arrives after B_{100}). In this case, we need to keep track of all incoming batches from B_1 to B_{100}, i.e., a long list of 100 expected support values of each node in the UF-stream structure. The situation can be worsened if B_2 arrives even later as it would result in a much longer list. Hence, it consumes a large memory space. Moreover, what if B_2 was lost and does not arrive? The size of the list can be unbounded. The second problem is that the above approach cannot compute the expected support of any pattern from B_1 to B_T until the late arrival of B_2. The situation can be worsened if B_2 arrives even later as it would result in a much longer wait. Hence, it requires a long runtime (including a long waiting time). Moreover, what if B_2

was lost and does not arrive? The computation of expected support may not be able to perform.

Can we do better? We propose an algorithm to solve the above problems. Specifically, instead of keeping a very long and potentially unbounded list, we fix the size of the list. In our fixed-size list (buffer array), we store a certain number (w) of expected support values (in a sorted order of time information). The size w of this list can be specified by the user or by the program designer. In this list, expected support values are sorted according to the batch (either by batch number i for batch B_i or by the timestamp/"time tag"). The key idea is that we keep a fixed-size list. Batches are sorted according to their original intended order. Hence, we have the same effect of waiting (a bit) for any delayed batch as we did in the above natural approach. However, we do not wait forever as in the natural approach (due to the very long and potentially unbounded list). As we keep a fixed-size list, we process the list as soon as it is full. Once the batches on that list have been processed, we empty the list and await for the next w batches which may be in order or out-of-order.

More specifically, for all delayed data batches, they have time to "catch up". In other words, users can determine how long they want to give to a delayed data stream by assigning a value to w. When the array is full, the algorithm applies TUF-streaming to the "oldest" value in the array, and stores it into the tree node. Now, the process of storing data becomes: When a new pattern X with a support value $expSup(X, B_i)$ needs to be stored into the UF-stream structure, our algorithm called **TUF-streaming(Delay)** performs the following five steps:

1. (Special case) If $expSup(X, B_i)$ and also the latest element in the buffer array also equals to 0, we only need to insert one 0 into the end of the array.
2. Check the "time tag" of the latest element in the buffer array, compare it with the "time tag" of X.
3. If the "time tag" of X is later than the latest element in the buffer array, insert $expSup(X, B_i)$ into the end of buffer array.
4. If the "time tag" of X is earlier than the latest element in the buffer array, move on to the 2nd latest element, compare the "time tag" with X. Repeat this process until reach the correct location in the buffer array for inserting $expSup(X, B_i)$.
5. After any of the above insertions, if the buffer array is full, shift the earliest element in the buffer array out. We process the array element.
6. (Worst case) If the "time tag" of X is earlier than all elements in the buffer array, we process pattern X immediately (this process could be already too late, which means, users need a larger size w for buffer array).

Example 3. Consider a sequence of data batches that flows in to the central server, shown in Table 1. For simplicity, let us assume that, in Table 1, every data stream only contains two transactions. Every transaction only contains one item. Let us set the size of the buffer array to 2. The process of how the algorithm works is shown in Fig. 2. The following two time points are special cases: (i) Batch B_3 (in Fig. 2(c)), the time tag of the data {a:1.5} (4:00) is earlier

Table 1. A running example for TUF-streaming(Delay)

Batches	TID	Itemsets	Time Tag
B_1	t_1	{a:0.9}	5:00
	t_2	{a:0.8}	
B_2	t_3	{b:0.75}	5:15
	t_4	{b:0.75}	
B_3	t_5	{a:0.8}	4:00
	t_6	{a:0.7}	
B_4	t_7	{c:0.5}	6:00
	t_8	{c:0.7}	
B_5	t_9	{a:0.5}	5:30
	t_{10}	{a:0.9}	
B_6	t_{11}	{b:0.6}	7:00
	t_{12}	{b:0.7}	
B_7	t_{13}	{a:0.7}	3:00
	t_{14}	{a:0.5}	
B_8	t_{15}	{a:0.35}	6:30
	t_{16}	{a:0.95}	
B_9	t_{17}	{c:0.65}	6:30
	t_{18}	{c:0.7}	

than the old data in the buffer array (5:00). Hence, the algorithm inserts the data {a:1.5} (4:00) before {a:1.7} (5:00). (ii) Batch B_7 (in Fig. 2(g)), the time tag of data {a:1.2} (3:00) is earlier than all old data in the buffer array. Hence, the algorithm processes {a:1.2} (3:00) immediately. □

3.2 Handling Batches with Sequential Number

A potential problem associated with TUF-streaming(Delay) is that the expected support of a pattern X may not be accurate. It happens when any batch (say, B_2) was very late and does not catch up with the current batch (i.e., does not arrive before the list is full). As such, the computation has been performed on an incomplete information (e.g., missing B_2). Once the computation (which is a sum of products), it cannot be undone.

Can we do better? Without loss of generality, let us assume that each batch can be identified by a sequential number i (say, consecutive batch number). Then, we propose an algorithm, called **TUF-streaming(Seq)**. Such an algorithm computes the expected support of X by Equation (2), which is a recursive function. On the surface, it may appear to be impossible to handle out-of-order batches. A careful analysis of the equation reveals that it is feasible to handle out-of-order batches. If there is any delay in the arrival of some batches, the algorithm updates the stored expected support value by multiplying appropriate weights. See the following example.

Example 4. Revisit Example 1. If batch B_3 arrives before B_1 and B_2, TUF-streaming(Seq) (i) processes B_3, (ii) computes the expected supports of all "frequent" patterns in B_3, and (iii) stores the expected support values. Then, when

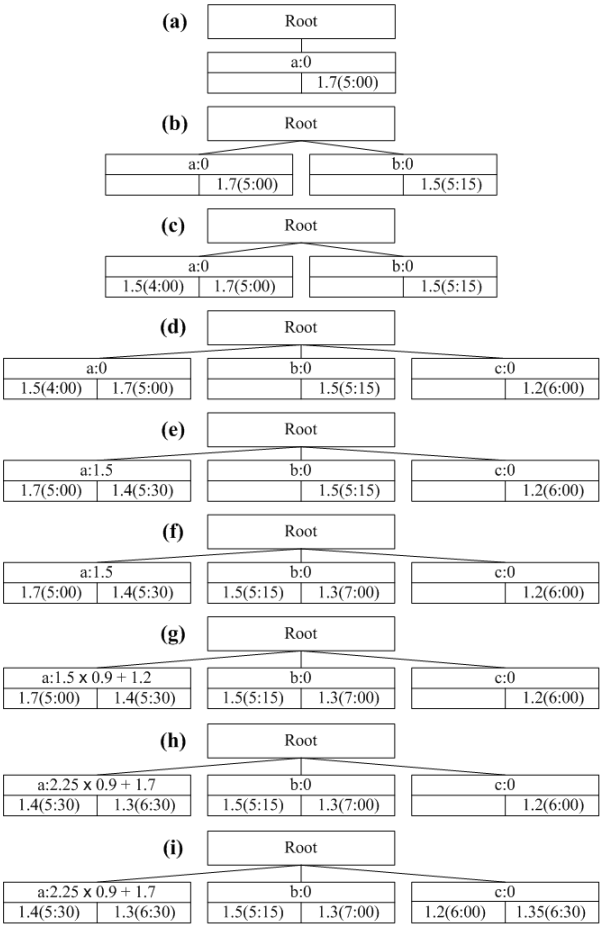


Fig. 2. TUF-streaming(Delay) Example

batch B_1 arrives, the algorithm (i) processes B_1, (ii) computes the expected supports of all "frequent" patterns in B_1, and (iii) "updates" the expected support values. The algorithm keeps track of which batches have been processed. This is possible because of the following:

$$
\begin{aligned}
& expSup\left(X, \cup_{i=1}^{3} B_i\right) \\
= & \; expSup\left(X, \cup_{i=1}^{2} B_i\right) \times \alpha + expSup(X, B_3) \\
= & \left[expSup\left(X, \cup_{i=1}^{1} B_i\right) \times \alpha + expSup(X, B_2)\right] \times \alpha + expSup(X, B_3) \\
= & \left[expSup\left(X, B_1\right) \times \alpha + expSup(X, B_2)\right] \times \alpha + expSup(X, B_3)
\end{aligned}
$$

$$
\begin{aligned}
= & \; expSup(X, B_3) + \\
& \left[expSup(X, B_1) \times \alpha^{3-1}\right] + \\
& \left[expSup(X, B_2) \times \alpha^{3-2}\right].
\end{aligned}
$$

When all batches have arrived (i.e., no gap), the expected support values stored in the node are accurate; prior to that time, the stored expected support values may be lower than the real/accurate one. □

Note that TUF-streaming(Seq) solves the two problems associated with the original natural approach. First, we no longer need to keep a very long and potentially unbounded list. We do not even need to keep a fixed-size bounded list (as in TUF-streaming(Delay)). All we need to keep is just a number/value in TUF-streaming(Seq). Second, computation can be performed as soon as the arrival of any (out-of-order) batch. We no longer need to wait for all out-of-order batches (as in the natural approach) or for a fixed number (i.e., w) out-of-order batches. We (partially) compute the expected support of all pattern in TUF-streaming(Seq), and these expected support values will be updated (if needed) when the next batch arrives.

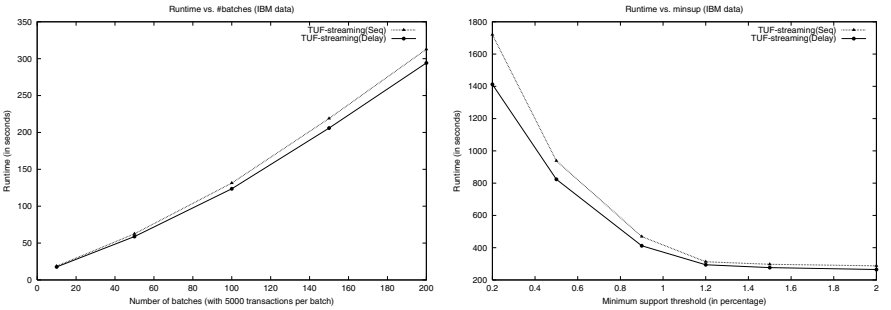
4 Evaluation

In this section, we analytically and experimentally examine (i) accuracy, (ii) runtime, and (iii) memory consumption of our algorithms. Experimentally, we conducted several tests to evaluate the algorithms using different datasets, which included IBM synthetic data and UCI real data. We used an IBM synthetic dataset with 1M records with an average transaction length of 10 items and a domain of 1,000 items. We assigned an existential probability from the numerical range (0,1] to every item in each transaction. We set each batch to contain 5,000 transactions (for a maximum of w=200 batches). The reported figures are based on the average of multiple runs in a time-sharing environment using an 800 MHz machine. Runtime included CPU and I/Os for mining of "frequent" patterns and maintenance of the UF-stream structure. We evaluated different aspects of our proposed algorithms, which were implemented in C.

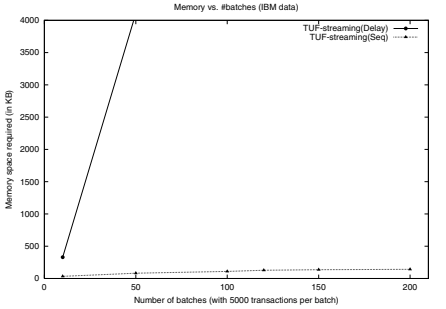
4.1 Accuracy

In terms of accuracy, when all batches arrive (even with some delays), the TUF-streaming(Seq) algorithm is as accurate as TUF-streaming(Ordered). Even when batches arrive out-of-order, as long as all batches arrive, the collection of frequent patterns discovered by TUF-streaming(Seq) is identical to those discovered by TUF-streaming(Ordered). Recall that TUF-streaming(Ordered) mines frequent patterns from ordered batches of uncertain data.

Similarly, when all delayed batches arrive (even with some delays or out-of-order) before TUF-streaming(Delay) starts processing any batches, TUF-streaming(Delay) is as accurate as TUF-streaming(Ordered). The set of frequent patterns discovered by TUF-streaming(Delay) is identical to those discovered by TUF-streaming(Ordered). However, when TUF-streaming(Delay) starts processing the batches before some very late-comers, the set of discovered frequent patterns may be different. Fortunately, this potential problem can be fixed by lengthening the list (i.e., increasing the number of batches w to be kept in the list).



(a) #batches vs. runtime (b) *minsup* vs. runtime



(c) #batches vs. memory

Fig. 3. Experimental results on IBM synthetic data

4.2 Runtime

As for the runtime, both TUF-streaming(Seq) and TUF-streaming(Delay) are analytically required to visit every node in the UF-stream structure regardless of whether or not the corresponding pattern is "frequent", i.e., visit $|\cup_i FP_i|$ nodes for each of T batches for a total of $T \times |\cup_i FP_i|$ visits, where T is the number of batches mined so far at time T (e.g., visited 26 nodes for Fig. 1).

Fig. 3(a) shows that, when the number of batches (w) increased, the runtime increased. See Fig. 3(a). Both TUF-streaming(Seq) and TUF-streaming(Delay) took almost the same amount of time. The latter inserted appropriately the expected support values of "frequent" patterns discovered from a new batch whenever the batch was processed and mined, whereas the former took slightly more time to update the expected support due to multiplication and addition. Both algorithms visited all nodes in the UF-stream structure.

We also varied *minsup* values. Fig. 3(b) shows that, when *minsup* increased, the number of expected support values stored in the UF-stream structure decreased for all algorithms because the number of "frequent" patterns mined from the stream decreased.

4.3 Memory Consumption

With respect to memory consumption, let $|FP_i|$ denote the number of "frequent" patterns mined from Batch B_i. The TUF-streaming(Delay) algorithm keeps a list

of w expected support values, where the length w of such a list can be specified by the user. In other words, TUF-streaming(Delay) requires $w \times |\cup_i FP_i|$ expected support values to be stored in the UF-stream structure.

In contrast, the TUF-streaming(Seq) algorithm requires a smaller amount of space than TUF-streaming(Delay) because each node stores a single value (i.e., a total of $|\cup_i FP_i|$ values to be stored in the UF-stream structure.

Fig. 3(c) shows that, when the number of batches increased, the number of expected support values stored in the UF-stream structure also increased. For TUF-streaming(Delay), the increase was almost linear as the list of expected support values in each node increased proportional to the number of batches. Moreover, as data are not necessarily uniformly distributed, different patterns can be discovered from different batches. These add a few patterns to the collection of patterns to be kept in the UF-stream structure. In contrast, as TUF-streaming(Seq) only kept a single value for each node, its memory consumption was independent of the number of batches.

In addition to conducting experiments with the IBM synthetic data, we also evaluated our algorithms using real data (e.g., mushroom, retail, kosarak) from the Frequent Itemset Mining Dataset Repository as well as the from the UC Irvine Machine Learning Repository. The experimental results are consistent with those experimented with the IBM synthetic data. For lack of space, we omit the graphs.

5 Conclusions

In this paper, we proposed two tree-based mining algorithms for mining frequent patterns from dynamic streams of uncertain data with the time-fading model. Both algorithms apply UF-growth with *preMinsup* to find "frequent" patterns. The mined patterns are then stored in the UF-stream structure together with their expected support values. Then, when the next batch of streaming transactions flows in, the algorithms update the UF-stream structure accordingly. As each batch of uncertain data can be of a different size or contain different numbers of transactions, batches of data streams may be delayed and not arrive in sequential order. Our two algorithms—namely, TUF-streaming(Delay) and TUF-streaming(Seq)—handle these out-of-order batches of uncertain streaming data, from which frequent patterns are mined. Evaluation results showed the correctness and effectiveness of our proposed algorithms when using the time-fading model for the stream mining of frequent patterns from delayed batches of uncertain data.

Acknowledgements. This project is partially supported by NSERC (Canada) and University of Manitoba.

References

1. Aggarwal, C.C., Li, Y., Wang, J., Wang, J.: Frequent pattern mining with uncertain data. In: ACM KDD 2009, pp. 29–37 (2009)
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: VLDB 1994, pp. 487–499 (1994)

3. Budhia, B.P., Cuzzocrea, A., Leung, C.K.-S.: Vertical frequent pattern mining from uncertain data. In: KES 2012, pp. 1273–1282 (2012)
4. Calders, T., Garboni, C., Goethals, B.: Efficient pattern mining of uncertain data with sampling. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010, Part I. LNCS (LNAI), vol. 6118, pp. 480–487. Springer, Heidelberg (2010)
5. Ezeife, C.I., Zhang, D.: TidFP: mining frequent patterns in different databases with transaction ID. In: Pedersen, T.B., Mohania, M.K., Tjoa, A.M. (eds.) DaWaK 2009. LNCS, vol. 5691, pp. 125–137. Springer, Heidelberg (2009)
6. Fariha, A., Ahmed, C.F., Leung, C.K.-S., Abdullah, S.M., Cao, L.: Mining frequent patterns from human interactions in meetings using directed acyclic graph. In: Pei, J., Tseng, V.S., Cao, L., Motoda, H., Xu, G. (eds.) PAKDD 2013, Part I. LNCS (LNAI), vol. 7818, pp. 38–49. Springer, Heidelberg (2013)
7. Giannella, C., Han, J., Pei, J., Yan, X., Yu, P.S.: Mining frequent patterns in data streams at multiple time granularities. In: Data Mining: Next Generation Challenges and Future Directions, pp. 105–124 (2004)
8. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: ACM SIGMOD 2000, pp. 1–12 (2000)
9. Huang, D., Koh, Y.S., Dobbie, G.: Rare pattern mining on data streams. In: Cuzzocrea, A., Dayal, U. (eds.) DaWaK 2012. LNCS, vol. 7448, pp. 303–314. Springer, Heidelberg (2012)
10. Jiang, N., Gruenwald, L.: Research issues in data stream association rule mining. ACM SIGMOD Record 35(1), 14–19 (2006)
11. Leung, C.K.-S.: Mining uncertain data. WIREs Data Mining and Knowledge Discover 1(4), 316–329 (2011)
12. Leung, C.K.-S., Cuzzocrea, A., Jiang, F.: Discovering frequent patterns from uncertain data streams with time-fading and landmark models. In: Hameurlain, A., Küng, J., Wagner, R., Cuzzocrea, A., Dayal, U. (eds.) TLDKS VIII. LNCS, vol. 7790, pp. 174–196. Springer, Heidelberg (2013)
13. Leung, C.K.-S., Hao, B.: Mining of frequent itemsets from streams of uncertain data. In: IEEE ICDE 2009, pp. 1663–1670 (2009)
14. Leung, C.K.-S., Hayduk, Y.: Mining frequent patterns from uncertain data with MapReduce for Big Data analytics. In: Meng, W., Feng, L., Bressan, S., Winiwarter, W., Song, W. (eds.) DASFAA 2013, Part I. LNCS, vol. 7825, pp. 440–455. Springer, Heidelberg (2013)
15. Leung, C.K.-S., Jiang, F.: Frequent pattern mining from time-fading streams of uncertain data. In: Cuzzocrea, A., Dayal, U. (eds.) DaWaK 2011. LNCS, vol. 6862, pp. 252–264. Springer, Heidelberg (2011)
16. Leung, C.K.-S., Mateo, M.A.F., Brajczuk, D.A.: A tree-based approach for frequent pattern mining from uncertain data. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS (LNAI), vol. 5012, pp. 653–661. Springer, Heidelberg (2008)
17. Leung, C.K.-S., Tanbeer, S.K.: Mining popular patterns from transactional databases. In: Cuzzocrea, A., Dayal, U. (eds.) DaWaK 2012. LNCS, vol. 7448, pp. 291–302. Springer, Heidelberg (2012)
18. Leung, C.K.-S., Tanbeer, S.K.: PUF-tree: A compact tree structure for frequent pattern mining of uncertain data. In: Pei, J., Tseng, V.S., Cao, L., Motoda, H., Xu, G. (eds.) PAKDD 2013, Part I. LNCS (LNAI), vol. 7818, pp. 13–25. Springer, Heidelberg (2013)

19. Leung, C.K.-S., Tanbeer, S.K., Budhia, B.P., Zacharias, L.C.: Mining probabilistic datasets vertically. In: IDEAS 2012, pp. 199–204 (2012)
20. Qu, J.-F., Liu, M.: A fast algorithm for frequent itemset mining using Patricia* structures. In: Cuzzocrea, A., Dayal, U. (eds.) DaWaK 2012. LNCS, vol. 7448, pp. 205–216. Springer, Heidelberg (2012)
21. Tanbeer, S.K., Leung, C.K.-S.: Finding diverse friends in social networks. In: Ishikawa, Y., Li, J., Wang, W., Zhang, R., Zhang, W. (eds.) APWeb 2013. LNCS, vol. 7808, pp. 301–309. Springer, Heidelberg (2013)
22. Yu, J.X., Chong, X., Lu, H., Zhou, A.: False positive or false negative: mining frequent itemsets from high speed transactional data streams. In: VLDB 2004, pp. 204–215 (2004)

Fast Causal Network Inference over Event Streams

Saurav Acharya and Byung Suk Lee

Department of Computer Science, University of Vermont, Burlington, VT 05405, USA
{sacharya,bslee}@uvm.edu

Abstract. This paper addresses causal inference and modeling over event streams where data have high throughput and are unbounded. The availability of large amount of data along with the high data throughput present several new challenges related to causal modeling, such as the need for fast causal inference operations while ensuring consistent and valid results. There is no existing work specifically for such a streaming environment. We meet the challenges by introducing a time-centric causal inference strategy that leverages temporal precedence information to decrease the number of conditional independence tests required to establish the dependencies between the variables in a causal network. Dependency and temporal precedence of cause over effect are the two properties of a causal relationship. We also present the *Temporal Network Inference* algorithm to model the temporal precedence relations into a temporal network. Then, we propose the *Fast Causal Network Inference* algorithm for faster learning of causal network using the temporal network. Experiments using synthetic and real datasets demonstrate the efficacy of the proposed algorithms.

Keywords: Causal inference; Event streams; Temporal data.

1 Introduction

In recent years, there has been a growing need for active systems that can perform causal inference in diverse applications such as health care, stock markets, user activity monitoring, smart electric grids, and network intrusion detection. These applications need to infer the cause of abnormal activities immediately such that informed and timely preventive measures are taken. As a case in point, consider a smart electric grid monitoring application. The failure of a component can cause cascading failures, effectively causing a massive blackout. Therefore, the identification of such cause and effect components in a timely manner enables preventive measures in the case of failure of a cause component, thereby preventing blackouts.

Causal network, a directed acyclic graph where the parent of each node is its direct cause, has been popularly used to model causality [1–7]. There are two distinct types of algorithms for learning a causal network: score-based [1–4] and constraint-based [5–8]. Both types of algorithms are slow and, therefore,

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 222–235, 2013.

not suitable for event streams where prompt causal inference is required. Score-based algorithms perform a greedy search (usually hill climbing) to select a causal network with the highest score from a large number of possible networks. With an increase in the number of variables in the dataset, the number of possible networks grows exponentially, resulting in slow causal network inference. On the other hand, constraint-based algorithms (e.g., PC algorithm [7]) discover the causal structure via a large number of tests on conditional independence(CI). There can be no edge between two conditionally independent variables in the causal network [9]. In a causal network of n variables, two variables X and Y are said to be conditionally independent given a condition set S if there is at least one variable in S such that X and Y are independent. The condition set S consists of all possible 2^{n-2} combinations of the remaining $n-2$ variables, and therefore the computational complexity grows exponentially as the number of variables increases. So, the current techniques for causal inference are slow and not suitable for event streams which have a high data throughput and where the number of variables (i.e., event types) is large.

With this concern, this paper describes a new time-centric causal modeling approach to speed up the causal network inference. Every causal relationship implies temporal precedence relationship [10]. So, the idea is to incorporate temporal precedence information as an important clue to reducing the number of required CI tests and thus maintaining feasible computational complexity. This idea achieves fewer computations of CI test due to two factors. First, since causality requires temporal precedence, we ignore the causality test for those nodes with no temporal precedence relationship between them. Second, in the CI test of an edge, we exclude those nodes from the condition set which do not have temporal precedence relationship with the nodes of the edge. Therefore, it reduces the size of the condition set which is a major cause of the exponential computational complexity. In addition, the temporal precedence relationship intuitively orients the causal edge unlike the constraint-based algorithms where a separate set of rules are needed to infer the causal direction (details in Section 3.3).

The contributions of this paper are summarized as follows. First, it presents a temporal network structure to represent temporal precedence relationships between event types and proposes an algorithm, *Temporal Network Inference(TNI)*, to construct a temporal network applicable in streaming environment. Second, it introduces a time-centric causal modeling strategy and proposes an algorithm, *Fast Causal Network Inference(FCNI)*, to speed up the learning of causal network. Finally, it empirically demonstrates the advantages of the proposed algorithm in terms of the running time and the total number of CI tests required for the learning of causal network by comparing it against the state-of-art algorithm for causal network inference, called the PC algorithm (details in Section 3.3).

The rest of this paper is organized as follows. Section 2 reviews the existing work on causal network inference. Section 3 presents the basic concepts used in the paper. Section 4 and Section 5 propose the learning of temporal network and faster causal network, respectively. Section 6 evaluates the proposed FNCI algorithm. Finally, Section 7 concludes the paper and mentions further research.

2 Related Work

As explained earlier, there are two main approaches for causal network inference.

The first approach, score-based [1–4], performs greedy search (usually hill climbing) over all possible network structures in order to find the network that best represents the data based on the highest score. This approach, however, has two problems. First, it is slow due to the exhaustive search for the best network structure. An increase in the number of variables in the dataset increases the computational complexity exponentially. Second, two or more network structures, called the equivalence classes [11], may represent the same probability distribution, and consequently the causal directions between nodes are quite random. There is no technique for alleviating these problems in a streaming environment. Thus, score-based algorithms are not viable for streams.

The second approach, constraint-based [5–8], does not have the problem of equivalence classes. However, it is slow as it starts with a completely connected undirected graph and thus performs a large number of CI tests to remove the edges between conditionally independent nodes. The number of CI tests increases exponentially with the increase in the number of variables in the dataset. To alleviate this problem, some constraint-based algorithms start with a minimum spanning tree to reduce the initial size of condition sets. However, this idea trades the speed with the accuracy of the causal inference. The constraint-based algorithms include IC* [5], SGS [6], PC [7], and FCI algorithm [7]. The FCI algorithm focuses on the causal network discovery from the dataset with latent variables and selection bias, which is quite different from the scope of this paper. The PC algorithm is computationally more efficient than IC* and SGS. This is why we evaluate the proposed *FCNI* algorithm by comparing it against the PC algorithm. Like the others, the PC algorithm starts with a completely connected undirected graph. To reduce the computational complexity, it performs CI tests in several steps. Each step produces a sparser graph than the earlier step, and consequently, the condition set decreases in the next step. However, the computational complexity is still $O(n^2 \cdot 2^{n-2})$. (The details are explained in Section 3.3.) Therefore, the current constraint-based algorithms are not suitable for fast causal inference over streams.

To the best of our knowledge, there exists no specific work in the causal network inference in a streaming environment. A new approach is needed for faster causal network inference.

3 Basic Concepts

This section presents some key concepts needed to understand the paper.

3.1 Event Streams, Type, and Instance

An event stream in our work is a sequence of continuous and unbounded timestamped events. An event refers to any action that has an effect and is created by one event owner. One event can trigger another event in chain reactions. Each

event instance belongs to one and only one event type which is a prototype for creating the instances. Two event instances are related to each other if they share common attributes such as event owner, location, and time. We call these attributes *common relational attributes(CRAs)*.

In this paper we denote an event type as E_j and an event instance as e_{ij}, where i indicates the *CRA* and j indicates the event type.

Example 1. Consider a diabetic patient monitoring system in a hospital. Each patient is uniquely identifiable, and each clinical test or measurement of each patient makes one event instance. For example, a patient is admitted to the hospital, has their blood pressure and glucose level measured, and takes medication over a period of time. This creates the instances of the above event types as a result. Typical event types from these actions include regular-insulin-dose-given, hypoglycemic-symptoms-exists, blood-glucose-measurement-decreased, increased, etc. Note that the patient ID is the CRA, as the events of the same patient are causally related.

3.2 Conditional Mutual Information

A popular approach for testing the conditional independence, with respect to the joint probability P, of two random variables X and Y given a subset of random variables S is conditional mutual information(CMI) (e.g., [8, 12]). CMI gives the strength of dependency between variables in a measurable quantity, which helps to identify strong and weak causal relationships in the final causal network.

To test whether X and Y are conditionally independent given S, we compute the conditional mutual information $I_{MI}(X, Y|S)$ as

$$I_{MI}(X, Y|S) = \sum_{x \in X} \sum_{y \in Y} \sum_{s \in S} p_{X,Y,S}(x, y, s) log_2 \frac{p_{X,Y|S}(x, y|s)}{p_{X|S}(x|s) p_{Y|S}(y|s)}$$

where p is the probability mass function calculated from the frequencies of variables.

We only keep the record of these frequencies, not the whole events, by updating them as a new batch of events arrives. Consequently, the independence test procedure is incremental in our case.

It is said that two variables X and Y are independent when $I_{MI}(X, Y|S) = 0$; otherwise, they are dependent. However, this presents us with the risk of spurious relationships due to weak dependencies (we cannot assume $I_{MI}(X, Y|S) = 10^{-5}$ and $I_{MI}(X, Y|S) = 10$ provide the same degree of confidence in the dependency). With an increase in the value of $I_{MI}(X, Y|S)$, the dependency between the variables X and Y grows stronger. Therefore, to prune out the weak dependencies, we need to set a threshold value of mutual information below which we ignore the evidence as weak. To do so, we relate CMI with G^2 test statistics [7, 13] as below where N_s is the number of samples.

$$G^2(X, Y|S) = 2 \cdot N_s \cdot log_e 2 \cdot I_{MI}(X, Y|S)$$

Under the independence assumption, G^2 follows the χ^2 distribution [14], with the degree of freedom *df* equal to $(r_x - 1)(r_y - 1)\prod_{s \in S} r_s$, where r_x, r_y, and r_s are the number of possible distinct values of X, Y, and S, respectively. So, we use χ^2 test, which provides a threshold based on *df* and significance level α, to validate the dependency result. We set α as the universally accepted value of 95%.

3.3 The PC Algorithm

The PC algorithm [7] (Algorithm 1) starts with a completely connected undirected graph on which the CI tests are performed to remove edges between independent nodes. The key idea is that a causal network has an edge between X and Y in the topology if and only if X and Y are not independent given all condition subsets of the remaining neighbor nodes [15]. In Algorithm 1, the topology of the causal network is learned in the steps 1 to 10. The network topology is then assigned causal direction in the steps 11 to 17.

Algorithm 1. PC algorithm

1: Construct the completely connected undirected graph G on the n nodes;
2: Initialize Neighbors(G,X) as the set of nodes adjacent to the node X in G, and SepSet(X,Y), a set of nodes that causes independence between X and Y nodes, as empty;
3: $k \leftarrow 0$;
4: **repeat**
5: **repeat**
6: Select any edge $X - Y$ such that $|Neighbors(G, X) \backslash Y| \geq k$;
7: **repeat**
8: Select any subset S of $Neighbors(G, X) \backslash Y$ such that $|Neighbors(G, X) \backslash Y| = k$;
9: If X and Y are independent given S, remove $X - Y$ from G, remove Y from Neighbors(G,X), remove X from Neighbors(G,Y), and add S to SepSet(X,Y) and SepSet(Y,X);
10: **until** every subset S of $Neighbors(G, X) \backslash Y$ such that $|Neighbors(G, X) \backslash Y| = k$ has been selected.
11: **until** every edge $X - Y$ such that $|Neighbors(G, X) \backslash Y| \geq k$ has been selected.
12: k = k + 1;
13: **until** every edge $X' - Y'$ satisfies $|Neighbors(G, X') \backslash Y'| < k$.
14: **for** each triplet of nodes X, Y, Z such that the edges $X - Y$ and $Y - X$ exist in G but not $X - Z$ **do**
15: Orient $X - Y - Z$ as X→Y←Z if and only if SepSet(X,Z) does not contain Y;
16: **end for**
17: **repeat**
18: **If** there exists $X \rightarrow Y$ and $Y - Z$, but not $X - Z$, **then** orient $Y - Z$ as $Y \rightarrow Z$;
19: **If** there exists $X - Y$ and a directed path from X to Y, **then** orient $X - Y$ as $X \rightarrow Y$;
20: **until** no edge can be oriented.

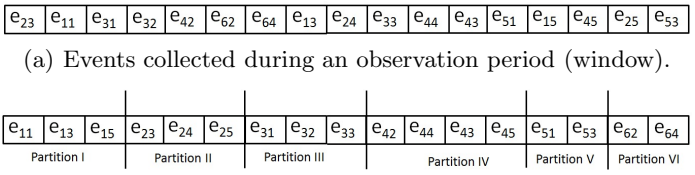
4 Learning Temporal Precedence Relationships

In this section, we describe an incremental approach to model temporal precedence relationships from time-stamped events into a *temporal network*.

4.1 Temporal Network Model

A temporal network is a directed acyclic graph of nodes representing event types where an edge between two nodes represents the temporal precedence relationship between them. To facilitate the handling of events in a streaming environment, we use a time-based window over the stream. Typically, the application offers a natural observation period (e.g., hour) that makes a window.

As mentioned earlier in Section 3.1, two events are related to each other if they share the same *common relational attribute(CRA)*). So, the events in a window are arranged by CRA and ordered by the timestamp as they arrive, producing a *partitioned window* as a result. Figure 1 illustrates it.



(a) Events collected during an observation period (window).

(b) Events in the window partitioned by CRA.

Fig. 1. Partitioned window of events

With the arrival of a new batch of event instances, we augment each partition in the new window by prefixing it with the last instance of the partition with the same CRA value in the previous window. This is necessary in order to identify the temporal precedence between instances that are separated into the two consecutive batches.

To determine when an edge, say $E_i \rightarrow E_j$, should be added in a temporal network, a measure providing an evidence of temporal precedence between the event types should be defined. The evidence we use is the frequency of the observation of an instance of E_j following an instance of E_i. The *temporal strength* of an edge identified is defined below.

Definition 1 (Temporal strength). *Consider an edge $E_i \rightarrow E_j$ $(i \neq j)$ in a temporal network of n event types. Let f_{ij} be the total number of observations in which an event of type E_i precedes an event of type E_j over all partitions in the* partitioned window*. Then, we define* temporal strength*, s_{ij}, of the edge $E_i \rightarrow E_j$ as*

$$s_{ij} \triangleq \frac{f_{ij}}{\sum_{k=0}^{(n-1)} f_{ik}}$$

4.2 Temporal Network Inference Algorithm

The idea behind the *TNI* algorithm is to collect events from an event stream in a *window* and then use temporal precedence information from the sequence of event pairs in the window to construct a temporal network at the event type level. The overall algorithm is centered on a *frequency matrix*, which is initially empty (i.e., all zero elements) and updated with each new batch of events. The algorithm has two steps for each window, outlined in Algorithm 2.

1. Update the frequency matrix *FM* by observing the precedence relationships of event pairs in the partitioned window (steps 3–13 in Algorithm 2). An element f_{ij} in *FM* reflects the total number of times events of type E_i precede events of type E_j ($i \neq j$). Each time an event pair (e_{oi}, e_{oj}) is observed in the event stream such that e_{oi} precedes e_{oj}, increase the value of f_{ij} by 1.
2. Determine the edges of the temporal network using the frequency matrix (steps 14–24 in Algorithm 2). For each pair of an edge and its reversed edge, select the edge with the greater frequency. Calculate the temporal strength of the selected edge, e.g., $E_i \rightarrow E_j$, and store it in the element s_{ij} of the *strength matrix SM*. Set the strength of the ignored edge with the lower frequency to zero. If a cycle is introduced, remove the edge with the lowest temporal strength in the cycle.

5 Learning Causal Network in Reduced Time

In this section, we describe a new approach to reduce the number of CI tests needed to infer the causal structure, thereby speeding up the learning process. The idea is to incorporate temporal precedence relationships to learn the causal network. The correctness of our approach is shown as follows. First, a temporal precedence relationship is a mandatory condition for inferring causality [10]. Therefore, causal relationship subsumes temporal precedence relationship, that is, the causal network is a subgraph of the temporal network. Second, a causal network should satisfy the *Causal Markov Condition (CMC)* [6, 9, 16] where for every node X in the set of nodes N, X is independent of its non-descendants excluding its parents (i.e., $N \backslash (Descendants(X) \cup Parents(X))$) conditional on its parents. In a temporal network of vertex (or node) set N, a node is temporally independent, and therefore causally independent, of all its non-descendants (except its parents) given its parents. In other words, the temporal network obeys CMC which is a necessary condition for the causal network. Therefore, our idea of considering a temporal network as an initial causal network is correct.

5.1 Fast Causal Network Inference Algorithm

The idea behind FCNI algorithm is to reduce the number of CI tests by incorporating temporal precedence information. The algorithm has two steps, as outlined in Algorithm 3.

Algorithm 2. Temporal Network Inference (TNI)

Require: an edgeless network structure TN, event stream(s) S

1: Initialize an empty *frequency matrix (FM)*, an empty strength matrix SM, two empty buffers B_p and B_c (used to store "parent" events and "child" events, respectively);
2: **for** each window W in S **do**
3: **for** each partition P (corresponding to CRA a) in W **do**
4: **for** i = 1 to $t_n - 1$ where t_n is the number of unique timestamp in P **do**
5: Clear B_p and B_c;
6: Insert all events with timestamp t_i and t_{i+1} into B_p and B_c, respectively;
7: **for** all event instances e_{ap} and e_{ac} in B_p and B_c, respectively, **do**
8: **if** type(e_{ac}) $\neq$ type(e_{ap}) {*//There cannot be causal relationships between events of the same type.*} **then**
9: Increase the frequency of element $f_{type(e_{ap}),type(e_{ac})}$ in FM by 1;
10: **end if**
11: **end for**
12: **end for**
13: **end for**
14: **for** each pair of elements f_{ij} and f_{ji} in FM **do**
15: $s_{ij} \leftarrow 0$, $s_{ji} \leftarrow 0$;
16: **if** $f_{ij} > f_{ji}$ **then**
17: Add an edge $E_i \rightarrow E_j$ in TN and set its strength to $s_{ij} = \frac{f_{ij}}{\sum_{k=0}^{n-1} f_{ik}}$;
18: **else if** $f_{ji} > f_{ij}$ **then**
19: Add an edge $E_j \rightarrow E_i$ in TN and set its strength to $s_{ji} = \frac{f_{ji}}{\sum_{k=0}^{n-1} f_{jk}}$;
20: **end if**
21: **end for**
22: **if** an edge is added and it introduces cycle in TN **then**
23: Remove the edge with the lowest temporal strength (in SM) in the cycle;
24: **end if**
25: **end for**

1. The first step is to construct a temporal network by running the *TNI* algorithm. The temporal network is set as the initial causal network. Note that since temporal precedence is a requirement for a causal relationship, all causal relationships are theoretically guaranteed be in the temporal network.
2. The second step is to adapt the ideas of constraint-based algorithms to learn the final causal network by pruning out the edges between independent nodes. We perform CI tests on every edge between nodes in the initial causal network to verify dependency between them. Conditionally independent nodes are considered to be spurious and hence the edge between them is removed. Steps 2 to 22 perform this step. The main difference from the PC algorithm is the manner in which CI tests are performed. In the PC algorithm, the condition set S for an edge $E_i - E_j$ considers the neighbors of both E_i and E_j whereas in the FCNI algorithm, as the edges are already directed, the condition set S for an edge $E_i \rightarrow E_j$ needs to consider only the parents of E_j (E_j is independent of the parents of E_i that do not have edge to E_j). Consequently, we need fewer CI tests.

Algorithm 3. Fast Causal Network Inference (FCNI)

Require: *Window W, Edgeless Causal Network G = (N, ξ).*{N and ξ are the set of nodes and the set of edges, respectively.}

```
1: Run the TNI algorithm and initialize G with the learned temporal network;
2: for each directed edge (E_i, E_j) ∈ ξ do
3:    independent = IsIndependent(E_i, E_j, φ), where φ is the empty set;
      {IsIndependent(E_i, E_j, S) calculates I_MI(E_i, E_j|S) for CI test.}
4:    if independent is true then
5:       Remove (E_i, E_j) from ξ;
6:    end if
7: end for
8: k ← 0;
9: repeat
10:   for each directed edge (E_i, E_j) ∈ ξ do
11:      Construct a set of condition sets, Z, each of cardinality k from the parents of
         E_j excluding E_i;
12:      repeat
13:         Select any subset S from Z;
14:         independent = IsIndependent(E_i, E_j, S);
15:         Remove S from Z;
16:      until Z is empty or independent is true
17:      if independent is true then
18:         Remove (E_i, E_j) from ξ;
19:      end if
20:   end for
21:   k = k + 1;
22: until number of parents of E'_j in every directed edge (E'_i, E'_j) ∈ ξ is less than k.
```

5.2 Complexity Analysis

The complexity of the FNCI algorithm for a causal network G is bounded by the largest degree of each node. Let n be the number of nodes (i.e., event types). Then in the worst case, since the causal network inference starts with a temporal network, the number of CI tests required by the algorithm is given as

$$CI_{max} = \sum_{i=1}^{n} d_i 2^{|Z_i|}$$

where $d_i \equiv (i-1)$ is the maximum degree of incoming edges to the node i (there are $\sum_{i=1}^{n} d_i = \frac{n \cdot (n-1)}{2}$ directed edges in the network G) and Z_i is the maximum condition set to each edge involving node i such that $|Z_i| = d_i - 1 = i - 2$. So the computational complexity of *FCNI* algorithm is $O(n \cdot 2^{n-2})$ in the worst case. In contrast, the PC algorithm (described in Section 3.3), whose condition set of each node is of cardinality $n-2$ nodes, has the worst case computational complexity of $O(n^2 \cdot 2^{n-2})$. Therefore, in the worst case, the *FCNI* algorithm is n times faster than the PC algorithm.

In the best case, the causal network G takes the form of a minimum spanning tree with $n-1$ edges. In this case, the FCNI algorithm and the PC algorithm require $n-1$ and $4n-6$ CI tests, respectively.

Note that the FCNI algorithm starts with a sparse network as it has only those edges that satisfy the temporal precedence relationships. So, in practice, it starts closer to the best case. In contrast, the PC algorithm starts with a completely connected dense network. So, it starts from the worst case.

6 Performance Evaluation

We conducted experiments to compare the proposed FCNI algorithm against the PC algorithm in terms of the accuracy, the running time, and the number of CI tests required on both the algorithms. Section 6.1 describes the experiment setup, including the evaluation metrics and the platform used. Section 6.2 explains the datasets used and Section 6.3 presents the experiment results.

6.1 Experiment Setup

Evaluation Metrics. Intuitively, the quality of causal network learning algorithms are best evaluated by examining how closely the constructed causal network structures resemble the target causal network. In this regard, we adopt the structural Hamming distance proposed by [17] as the quality metric of the output causal network. The nodes (i.e., event types) are fixed as given to the algorithms, and therefore the network structures are compared with respect to the edges between nodes. There are three kinds of possible errors in the causal network construction: reversed edges, missing edges, and spurious edges. We use the number of the erroneous edges of each kind as the evaluation metric.

Platform. The experiments are conducted on RedHat Enterprise Linux 5 operating system using Java(TM) 2 Runtime Environment–SE 1.5.0_07 in Vermont Advanced Computing Core (VACC) cluster computers.

6.2 Datasets

Experiments are conducted using both synthetic and real datasets.

Synthetic Datasets. A synthetic dataset is reverse-engineered from a target causal network. Given the control parameters – the number of event owners n_o and the number of event types n, the idea is to generate a random causal network, and then convert the causal network to an event stream which reflects the underlying probability distribution of the causal network. In the interest of space, the details of the event stream generation are not described here. We assume that the event owner is the *CRA*. The dataset is represented by a collection of files in which the events are shuffled according to the owner ID while preserving the temporal order. We create five datasets (see their profiles in Table 1), representing target causal networks of 4, 8, 12, 16 and 20 nodes each.

Table 1. Profiles of the five synthetic datasets

Dataset	n	$\mathbf{n}_{edges}$	$\mathbf{n}_o$	$\mathbf{n}_{instances}$
DS_1	4	4	5000	13108
DS_2	8	16	30000	108334
DS_3	12	32	500000	3163237
DS_4	16	46	6553600	49008538
DS_5	20	62	52428800	511972810

(n_{edges} is the number of *actual* edges in the network. $n_{instances}$ is the number of event instances in the dataset.)

Real Dataset. The real dataset D_R contains diabetes lab test results [18] of 70 different patients over a period ranging from a few weeks to a few months. The dataset has a total 28143 records, about 402 records for each patient. Each record has four fields – date, time, test code, test value. The clinical data of a patient is independent of other patients. Therefore, the patient ID is the *CRA* for this dataset. There are 20 different test codes appearing in the file from which we define 13 different event types of interest. The details of the event types are omitted due to the space limit.

6.3 Experiment Results

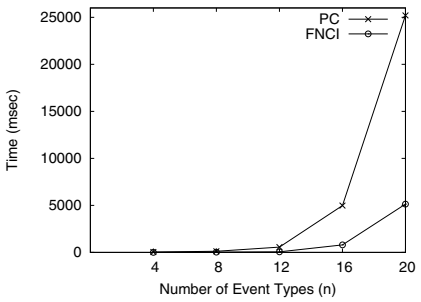
We ran the FCNI and PC algorithms over each of the five synthetic datasets and the real dataset. We present our evaluation results in three parts. First, we compare the quality of the generated networks against the target causal network and determine how closely they resemble the true causal network. (The details of the true causal networks are omitted due to the space limit.) Specifically, we count the number of spurious edges, the number of missing edges, and the number of reversed edges. Second, we compare the running time (CPU time) of the two algorithms, and finally, we evaluate the number of CI tests performed on both algorithms. We show that reducing the number of CI tests is the key to reducing the running time of causal network inference. The experiment is repeated ten times for each dataset (DS_1 through DS_5 and D_R) to calculate the average number of erroneous edges, the average running time, and the average number of CI tests. We assume the events are in temporal order.

Comparison of the Accuracy of the PC and FCNI Algorithms. Table 2 presents the number of erroneous edges in the causal network produced by the PC and FCNI algorithms. The results show that the quality of the causal network from the FCNI algorithm is similar to that of the PC algorithm. First, the number of missing and spurious edges are comparable. This is due to the reliance of both algorithms on the same test statistics (CMI in our case) to infer the independence of two event types. Second, the number of reversed edges is zero for the FCNI algorithm. Clearly the FCNI algorithm, through the temporal network, is much better at determining the correct causal edge direction. It is because the fact that the cause always precedes its effect is embodied in its

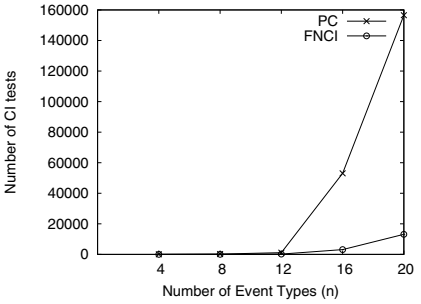
Table 2. Number of erroneous edges

	Algorithms	DS_1	DS_2	DS_3	DS_4	DS_5	D_R
Missing Edges	*PC*	0	0	0	0	1	1
	FCNI	0	1	0	0	1	1
Reversed Edges	*PC*	0	2	0	2	3	2
	FCNI	0	0	0	0	0	0
Spurious Edges	*PC*	0	3	0	4	3	1
	FCNI	0	3	0	4	3	1

temporal precedence relationship. Overall, we conclude that the FCNI algorithm produces almost the same topology as the PC algorithm, while the accuracy of the causal direction is improved.



(a) Running time for varying number of event types.



(b) Number of CI tests for varying number of event types.

Fig. 2. Comparison of the PC and FCNI algorithms

Comparison of the Running Time of the PC and FCNI Algorithms. Figure 2(a) plots the average running time of FCNI and PC algorithms against the number of event types (n) in the synthetic dataset. In all cases, the FCNI algorithm is much faster than the PC algorithm. Clearly, the temporal precedence information helps to reduce the size of condition set and the edges to test. As n increases, the running times of both PC and FCNI algorithms increase. However, the rate of increase of the running time of the PC algorithm is much higher than that of the FCNI algorithm. Therefore, with an increase in n, the ratio of running time between the two algorithms increases. The same observation is made in the real dataset where the running time of the PC and FCNI algorithms are 817 and 118 msecs, respectively. These results verify the important role of temporal precedence relationships to reduce the running time.

Comparison of the Number of CI Tests of the PC and FCNI Algorithms. Figure 2(b) shows that the FCNI algorithm performs fewer CI tests

than the PC algorithm in all synthetic datasets. The CI tests required are minimized, due to the temporal precedence information, by reducing the size of the condition set and the number of edges to test. With an increase in the number of event types (n), the rate of increase of the number of CI tests in the PC algorithm is much higher than that in the FCNI algorithm. A similar observation is made in the real dataset where the number of CI tests of the PC and FCNI algorithms are 1239 and 192, respectively. These results confirm the important role of temporal precedence relationships in reducing the number of CI tests. Note the result of CI tests (Figure 2(b)) looks almost the same as that of the running time (Figure 2(a)). This demonstrates that CI tests are the major performance bottleneck and validates the key idea of our work that reducing the number of CI tests reduces the run time.

7 Conclusion and Future Work

In this paper, we presented a novel strategy to incorporate temporal precedence relationships to learn the causal network over event streams. First, we introduced the *Temporal Network Inference* algorithm to model temporal precedence information. Then, we presented the *Fast Causal Network Inference* algorithm to reduce the running time complexity of learning causal network by eliminating unnecessary CI tests. We showed the experiment results to validate our approach by comparing against the state-of-the-art PC algorithm. For the future work, we plan to explore the temporal semantics further for causal network inference over out-of-order event streams.

References

1. Heckerman, D.: A Bayesian approach to learning causal networks. In: UAI, pp. 285–295 (1995)
2. Ellis, B., Wong, W.H.: Learning causal Bayesian network structures from experimental data. J. American Statistics Association 103(482), 778–789 (2008)
3. Li, G., Leong, T.-Y.: Active learning for causal Bayesian network structure with non-symmetrical entropy. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS, vol. 5476, pp. 290–301. Springer, Heidelberg (2009)
4. Meganck, S., Leray, P., Manderick, B.: Learning causal Bayesian networks from observations and experiments: a decision theoretic approach. In: Torra, V., Narukawa, Y., Valls, A., Domingo-Ferrer, J. (eds.) MDAI 2006. LNCS (LNAI), vol. 3885, pp. 58–69. Springer, Heidelberg (2006)
5. Pearl, J.: Causality: Models, Reasoning and Inference, 2nd edn. Cambridge University Press (2009)
6. Spirtes, P., Glymour, C.N., Scheines, R.: Causality from probability. In: ACSS (1990)
7. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search. MIT Press (2000)

8. Cheng, J., Greiner, R., Kelly, J., Bell, D., Liu, W.: Learning Bayesian networks from data: an information-theory based approach. Artificial Intelligence 137(1-2), 43–90 (2002)
9. Pearl, J.: Causal diagrams for empirical research. Biometrika 82, 669–688 (1995)
10. Popper, K.: The Logic of Scientific Discovery, Reprint edn. Routledge (October 1992)
11. Chickering, D.M.: Learning equivalence classes of Bayesian-network structures. J. Machine Learning Research 2, 445–498 (2002)
12. de Campos, L.M.: A scoring function for learning Bayesian networks based on mutual information and conditional independence tests. J. Machine Learning Research 7, 2149–2187 (2006)
13. Bishop, Y.M., Fienberg, S.E., Holland, P.W.: Discrete Multivariate Analysis: Theory and Practice. MIT Press (1975)
14. Kullback, S.: Information Theory and Statistics, 2nd edn. Dover Publication (1968)
15. Spirtes, P., Meek, C.: Learning Bayesian networks with discrete variables from data. In: KDD, pp. 294–299 (1995)
16. Pearl, J.: Graphs, causality, and structural equation models. Sociological Methods and Research 27, 226–284 (1998)
17. Tsamardinos, I., Brown, L.E., Aliferis, C.F.: The max-min hill-climbing Bayesian network structure learning algorithm. Mach. Learn. 65(1), 31–78 (2006)
18. Frank, A., Asuncion, A.: UCI machine learning repository (2010)

SSCJ: A Semi-Stream Cache Join Using a Front-Stage Cache Module

M. Asif Naeem[1], Gerald Weber[2], Gillian Dobbie[2], and Christof Lutteroth[2]

[1] School of Computing and Mathematical Sciences, Auckland University of Technology
[2] Department of Computer Science, The University of Auckland, Private Bag 92006, Auckland, New Zealand
mnaeem@aut.ac.nz, {gerald,gill,lutteroth}@cs.auckland.ac.nz

Abstract. Semi-stream processing has become an emerging area of research in the field of data stream management. One common operation in semi-stream processing is joining a stream with disk-based master data using a join operator. This join operator typically works under limited main memory and this memory is generally not large enough to hold the whole disk-based master data. Recently, a number of semi-stream join algorithms have been proposed in the literature to achieve an optimal performance but still there is room to improve the performance. In this paper we propose a novel Semi-Stream Cache Join (SSCJ) using a front-stage cache module. The algorithm takes advantage of skewed distributions, and we present results for Zipfian distributions of the type that appear in many applications. We analyze the performance of SSCJ with a well known related join algorithm, HYBRIDJOIN (Hybrid Join). We also provide the cost model for our approach and validate it with experiments.

Keywords: Semi-stream processing, Stream-based join, Data warehousing; Performance measurement.

1 Introduction

Stream-based joins are important operations in modern system architectures, where just-in-time delivery of data is expected. We consider a particular class of stream-based join, a semi-stream join that joins a single stream with a slowly changing table. Such a join can be applied in real-time data warehousing [6,4]. In this application, the slowly changing table is typically a master data table. Incoming real-time sales data may comprise the stream. The stream-based join can be used for example to enrich the stream data with master data. In this work we only consider one-to-many equijoins, as they appear between foreign keys and the referenced primary key in another table.

For executing stream-based operations, the large capacity of current main memories as well as the availability of powerful cloud computing platforms means, that considerable computing resources can be utilized. For master data of the right size for example, main-memory algorithms can be used.

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 236–247, 2013.

However, there are several scenarios, where stream joins that use a minimum of resources are needed. One particular scenario is an organization trying to reduce the carbon footprint of the IT infrastructure. A main memory approach as well as cloud-computing approaches can be power-hungry. Also in the area of mobile computing and embedded devices a low-resource consumption approach can be advantageous. Therefore, approaches that can work with limited main memory are of interest.

In the past, the algorithm HYBRIDJOIN (Hybrid Join) [7] was proposed for joining a stream with a slowly changing table with limited main memory requirements. This algorithm is an interesting candidate for a resource aware system setup. The key objective of this algorithm is to amortize the fast input stream with the slow disk access within limited memory budget and to deal with the bursty nature of the input data stream. Further details about HYBRIDJOIN are presented in Section 3.

Although the HYBRIDJOIN algorithm amortizes the fast input stream using an index-based approach to access the disk-based relation and can deal with bursty streams, the performance can still be improved if some characteristics of stream data are taken into consideration. We are looking for characteristics of data that are considered ubiquitous in real world scenarios. A Zipfian distribution of the foreign keys in the stream data matches distributions that are observed in a wide range of applications [1]. We therefore created a data generator that can produce such a Zipfian distribution. A Zipfian distribution is parameterized by the exponent of the underlying power law. In different scenarios, different exponents are observed, and determine whether the distribution is considered to have a short tail or a long tail. Distributions with a short tail would be more favourable for SSCJ from the outset, therefore we decided not to use a distribution with a short tail in order to not bias our experiment towards SSCJ. Instead we settled on a natural exponent that is observed in a variety of areas, including the original Zipf's Law in linguistics [5] that gave rise to the popular name of these distributions. The main result of our analysis is that SSCJ performs better on a skewed dataset that is synthetic, but following a Zipfian distribution as is found frequently in practice. For our analysis we do not consider joins on categorical attributes in master data, e.g. we do not consider equijoins solely on attributes such as gender.

The rest of the paper is structured as follows. Section 2 presents related work. In Section 3 we describe our observations about HYBRIDJOIN. Section 4 describes the proposed SSCJ with its execution architecture and cost model. Section 5 describes an experimental analysis of SSCJ. Finally, Section 6 concludes the paper.

2 Related Work

In this section, we present an overview of the previous work that has been done in this area, focusing on those which are closely related to our problem domain.

A seminal algorithm Mesh Join (MESHJOIN) [9,10] has been designed especially for joining a continuous stream with a disk-based relation, like the scenario

in active data warehouses. The MESHJOIN algorithm is a hash join, where the stream serves as the build input and the disk-based relation serves as the probe input. A characteristic of MESHJOIN is that it performs a staggered execution of the hash table build in order to load in stream tuples more steadily. The algorithm makes no assumptions about data distribution and the organization of the master data. The MESHJOIN authors report that the algorithm performs worse with skewed data.

R-MESHJOIN (reduced Mesh Join) [8] clarifies the dependencies among the components of MESHJOIN. As a result the performance has been improved slightly. However, R-MESHJOIN implements the same strategy as in the MESHJOIN algorithm for accessing the disk-based relation.

One approach to improve MESHJOIN has been a partition-based join algorithm [3], which can also deal with stream intermittence. It uses a two-level hash table in order to attempt to join stream tuples as soon as they arrive, and uses a partition-based waiting area for other stream tuples. For the algorithm in [3], however, the time that a tuple is waiting for execution is not bounded. We are interested in a join approach where there is a time guarantee for when a stream tuple will be joined.

Another recent approach, Semi-Streaming Index Join (SSIJ) [**?**] joins stream data with disk-based data. SSIJ uses page level cache i.e. stores the entire disk pages in cache. It is possible that not all the tuples in these pages are frequent in the stream and as a result the algorithm can perform suboptimally. Also the authors do not include the mathematical cost model for the algorithm.

3 Problem Definition

To clarify our observations, we present the HYBRIDJOIN algorithm in detail and at the end of this section we formulate an argument that we focus on in this paper.

A semi-stream join algorithm, HYBRIDJOIN, was based on two objectives. The first objective was to amortize the disk I/O cost over the fast input data stream more effectively by introducing an index-based approach to access the disk-based relation R. The second objective was to deal with the bursty nature of a data stream effectively. An abstract level working overview of HYBRIDJOIN is presented in Figure 1 where we consider m partitions in the queue to store stream tuples and n pages in disk-based relation R. In order to keep it simple, currently, we assume that the stream tuples are stored in a queue rather than in a hash table and the join is directly performed with the queue. The disk buffer is used to load one disk page into memory.

The key to HYBRIDJOIN is that, for each iteration the algorithm reads the oldest tuple from the queue and using that tuple as an index it loads the relevant disk page into the disk buffer. After loading the disk page into memory, the algorithm matches each tuple on the disk page with the stream tuples in the queue. When a match is found, the algorithm generates a tuple as an output after deleting it from the queue. In the next iteration, the algorithm again reads

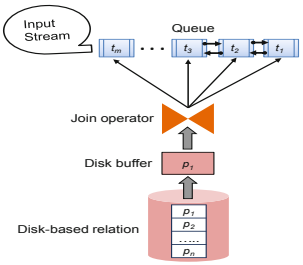


Fig. 1. HYBRIDJOIN working overview

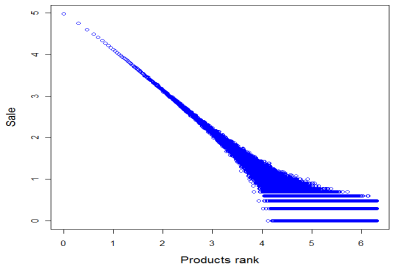


Fig. 2. Current market sales (on log-log scale)

stream input, extracts the oldest element from the queue, loads the relevant disk page into the disk buffer and repeats the entire procedure.

Although HYBRIDJOIN accesses R using an index reducing the I/O cost as compared to the other approaches, described in Section 2, if we analyse the current market sales then we observe that I/O cost can also be minimized further, ultimately improving the performance. To elaborate, we consider a benchmark which is based on current market sales based on the 80/20 rule [1]. According to this rule, 20 percent of products account for 80 percent of revenues and even in that 20 percent only a small number of products are sold very frequently. This rule can be implemented using Zipf's law with an exponent value equal to 1. The graphical representation of the benchmark is shown in Figure 2. From the figure it can be observed that the frequency of selling a small number of products is significantly higher compared to the rest of the products. Therefore, in the stream that propagates toward the warehouse, most of the tuples need to join with a small number of records on disk again and again. Currently the HYBRIDJOIN algorithm does not consider this feature and loads the pages from the disk frequently. Consider the reduction in I/O costs, if these pages can be held permanently in memory.

4 Semi-Stream Cache Join (SSCJ)

In this paper, we propose a new algorithm, SSCJ, which overcomes the issues stated above. This section presents a detailed description of SSCJ and its cost model.

4.1 Execution Architecture

The SSCJ algorithm possesses two complementary hash join phases, somewhat similar to Symmetric Hash Join. One phase uses R as the probe input; the largest part of R will be stored in tertiary memory. We call it the disk-probing phase. The other join phase uses the stream as the probe input, but will deal only with a small part of relation R. We call it the stream-probing phase. For

each incoming stream tuple, SSCJ first uses the stream-probing phase to find a match for frequent requests quickly, and if no match is found, the stream tuple is forwarded to the disk-probing phase.

The execution architecture for SSCJ is shown in Figure 3. The largest components of SSCJ with respect to memory size are two hash tables, one storing stream tuples denoted by H_S and the other storing tuples from the disk-based relation denoted by H_R. The other main components of SSCJ are a disk buffer, a queue, a frequency recorder, and a stream buffer. Relation R and stream S are the external input sources. Hash table H_R contains the most frequently accessed tuples of R and is stored permanently in memory. SSCJ alternates between stream-probing and disk-probing phases. According to the procedure described above, the hash table H_S is used to store only that part of the stream which does not match tuples in H_R. A stream-probing phase ends if H_S is completely filled or if the stream buffer is empty. Then the disk-probing phase becomes active. The length of the disk-probing phase is determined by the fact that a few disk partitions (or disk blocks) of R have to be loaded at a time in order to amortize the costly disk access. In the disk-probing phase of SSCJ, the oldest tuple in the queue is used to determine the partition of R that is loaded for a single probe step. In this way, in SSCJ it is guaranteed that every stream tuple loaded in memory will be processed in a certain time period. This is the step where SSCJ needs an index on table R in order to find the partition in R that matches the oldest stream tuple. However, a non-clustered index is sufficient, if we consider equijoins on a foreign key element that is stored in the stream. After one disk-probing phase, a number of stream tuples are deleted from H_S, so the algorithm switches back to the stream-probing phase. One phase of stream-probing with a subsequent phase of disk-probing constitutes one outer iteration of SSCJ.

The stream-probing phase is used to boost the performance of the algorithm by quickly matching the most frequent master data. For determining very frequent tuples in R and loading them into H_R, the frequency detection process is required. This process tests whether the matching frequency of the current tuple is larger than a pre-set threshold. If it is, then this tuple is entered into H_R. If there are no empty slots in H_R the algorithm overwrites an existing least frequent tuple in H_R. This least frequent tuple is determined by the component frequency recorder. The question of where to set the threshold arises, i.e. how frequently must a stream tuple be used in order to get into this phase, so that the memory sacrificed for this phase really delivers a performance advantage. The threshold is a flexible barrier. Initially, an

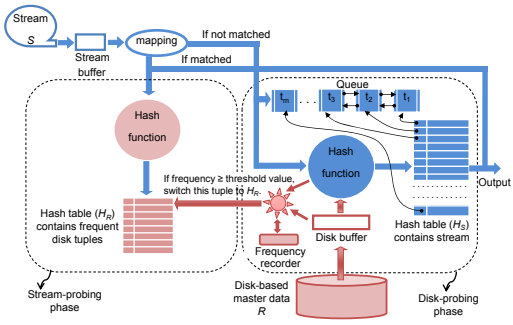


Fig. 3. Execution architecture of SSCJ

Table 1. Notations used in cost estimation of SSCJ

Parameter name	Symbol
Number of stream tuples processed in each iteration through H_R	w_N
Number of stream tuples processed in each iteration through H_S	w_S
Disk tuple size *(bytes)*	v_R
Disk buffer size *(tuples)*	d
Size of H_R *(tuples)*	h_R
Size of H_S *(tuples)*	h_S
Memory weight for the hash table	α
Memory weight for the queue	$1 - \alpha$
Cost to look-up one tuple in the hash table *(nano secs)*	c_H
Cost to generate the output for one tuple *(nano secs)*	c_O
Cost to remove one tuple from the hash table and the queue *(nano secs)*	c_E
Cost to read one stream tuple into the stream buffer *(nano secs)*	c_S
Cost to append one tuple in the hash table and the queue *(nano secs)*	c_A
Cost to compare the frequency of one disk tuple with the specified threshold value *(nano secs)*	c_F
Total cost for one loop iteration *(secs)*	c_{loop}

appropriate value is assigned to it while later on this value can be varied up and down depending on available size of H_R and the rate of matching the disk tuples in the disk buffer. The disk buffer stores the swappable part of R and for each iteration it loads a particular partition of R into memory. The other component queue is used to store the values for the join attribute. The main purpose of the queue is to keep the record of each stream tuple in memory with respect to time. The stream buffer is included in the diagram for completeness, but is in reality always a tiny component and it will not be considered in the cost model.

4.2 Cost Model

In this section we develop the cost model for our proposed SSCJ. The cost model presented here follows the style used for HYBRIDJOIN and MESHJOIN. Equation 1 represents the total memory used by the algorithm (except the stream buffer), and Equation 2 describes the processing cost for each iteration of the algorithm. The notations we used in our cost model are given in Table 1.

Memory Cost: The major portion of the total memory is assigned to the hash table H_S together with the queue while a comparatively much smaller portion is assigned to H_R, the frequency detector, and the disk buffer. The memory for each component can be calculated as follows:
Memory for the disk buffer (bytes)= $d \cdot v_R$
Memory for H_R (bytes)=$h_R \cdot v_R$
Memory for frequency recorder (bytes)=$8h_R$
Memory for H_S (bytes)=$\alpha(M - d \cdot v_R - h_R \cdot v_R - 8h_R)$
Memory for the queue (bytes) = $(1 - \alpha)(M - d \cdot v_R - h_R \cdot v_R - 8h_R)$

By aggregating the above, the total memory M for SSCJ can be calculated as shown in Equation 1.

$$M = d{\cdot}v_R{+}h_R{\cdot}v_R{+}8h_R{+}\alpha(M{-}d{\cdot}v_R{-}h_R{\cdot}v_R{-}8h_R){+}(1{-}\alpha)(M{-}d{\cdot}v_R{-}h_R{\cdot}v_R{-}8h_R) \quad (1)$$

Currently, the memory for the stream buffer in not included because it is small (0.05 MB is sufficient in our experiments).

Processing Cost: In this section we calculate the processing cost for the algorithm. To make it simple we first calculate the processing cost for individual components and then sum these costs to calculate the total processing cost for one iteration.
Cost to load d tuples from disk to the disk buffer (nanosecs)=$c_{I/O}(d)$
Cost to look-up w_N tuples in H_R (nanosecs)=$w_N \cdot c_H$
Cost to look-up disk buffer tuples in H_S (nanosecs)=$d \cdot c_H$
Cost to compare the frequency of all the tuples in disk buffer with the threshold value (nanosecs)=$d \cdot c_F$
Cost to generate the output for w_N tuples (nanosecs)=$w_N \cdot c_O$
Cost to generate the output for w_S tuples (nanosecs)=$w_S \cdot c_O$
Cost to read the w_N tuples from the stream buffer (nanosecs)=$w_N \cdot c_S$
Cost to read the w_S tuples from the stream buffer (nanosecs)=$w_S \cdot c_S$
Cost to append w_S tuples into H_S and the queue (nanosecs)=$w_S \cdot c_A$
Cost to delete w_S tuples from H_S and the queue (nanosecs)=$w_S \cdot c_E$
By aggregating the above costs the total cost of the algorithm for one iteration can be calculated using Equation 2.

$$c_{loop}(secs) = 10^{-9}[c_{I/O}(d){+}d(c_H{+}c_F){+}w_S(c_O{+}c_E{+}c_S{+}c_A){+}w_N(c_H{+}c_O{+}c_S)] \quad (2)$$

Since in c_{loop} seconds the algorithm processes w_N and w_S tuples of the stream S, the service rate μ can be calculated using Equation 3.

$$\mu = \frac{w_N + w_S}{c_{loop}} \quad (3)$$

In fact, based on the cost model we tuned SSCJ to a provably optimal distribution of memory between the two phases, and the components within the phases[1].

5 Performance Experiments

5.1 Experimental Setup

Hardware Specification: We performed our experiments on a *Pentium-core-i5* with 8GB main memory and 500GB hard drive as secondary storage. We implemented our experiments in `Java` using the `Eclipse IDE`. The relation R is stored on disk using a `MySQL` database.

[1] Due to the page limit we are unable to include the tuning of SSCJ in the paper.

Table 2. Data specification

Parameter	value
Size of disk-based relation R	100 million tuples (≈11.18GB)
Total allocated memory M	1% of R (≈0.11GB) to 10% of R (≈1.12GB)
Size of each disk tuple	120 *bytes* (similar to HYBRIDJOIN)
Size of each stream tuple	20 *bytes* (similar to HYBRIDJOIN)
Size of each node in the queue	12 *bytes* (similar to HYBRIDJOIN)

Measurement Strategy: The performance or service rate of the join is measured by calculating the number of tuples processed in a unit second. In each experiment both algorithms first complete their warm-up phase before starting the actual measurements. These kinds of algorithms normally need a warm-up phase to tune their components with respect to the available memory resources so that each component can deliver maximum performance. In our experiments, for each measurement we calculate the confidence interval by considering 95% accuracy, but sometimes the variation is very small.

Synthetic Data: The stream dataset we used is based on the Zipfian distribution. We test the performance of all the algorithms by varying the skew value from 0 (fully uniform) to 1 (highly skewed). The detailed specifications of our synthetic dataset are shown in Table 2.

TPC-H: We also analyze the performance of all the algorithms using the TPC-H dataset which is a well-known decision support benchmark. We create the datasets using a scale factor of 100. More precisely, we use table `Customer` as our master data table and table `Order` as our stream data table. In table `Order` there is one foreign key attribute `custkey` which is a primary key in `Customer` table. So the two tables are joined using attribute `custkey`. Our `Customer` table contains 20 million tuples while the size of each tuple is 223 bytes. On the other hand `Order` table also contains the same number of tuples with each tuple of 138 bytes.

Real-Life Data: Finally, we also compare the performance of all the algorithms using a real-life dataset[2]. This dataset basically contains cloud information stored in summarized weather reports format. The same dataset was also used with the original MESHJOIN. The master data table contains 20 million tuples, while the streaming data table contains 6 million tuples. The size of each tuple in both the master data table and the streaming data table is 128 bytes. Both the tables are joined using a common attribute, `longitude (LON)`, and the domain for the join attribute is the interval [0,36000].

5.2 Performance Evaluation

In this section we present a series of experimental comparisons between SSCJ and HYBRIDJOIN using synthetic, TPC-H, and real-life data. In order to un-

[2] This dataset is available at: `http://cdiac.ornl.gov/ftp/ndp026b/`

derstand the difference between the algorithms better, we include two other algorithms. First we include MESHJOIN, which is a seminal algorithm in the field that serves as a benchmark for semi-stream joins. Then we include R-MESHJOIN, which is a slight modification of MESHJOIN. It introduces an additional degree of freedom for the optimization of MESHJOIN.

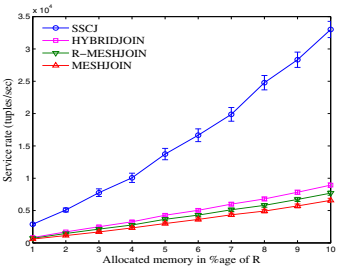
In our experiments we perform three different analyses. In the first analysis, we compare service rate, produced by each algorithm, with respect to the externally given parameters. In the second analysis, we present time comparisons, both processing and waiting time, for all four approaches. Finally, in our last analysis we validate our cost models for each of the algorithm.

External Parameters: We identify three parameters, for which we want to understand the behavior of the algorithms. The three parameters are: the total memory available M, the size of the master data table R, and the skew in the stream data. For the sake of brevity, we restrict the discussion for each parameter to a one dimensional variation, i.e. we vary one parameter at a time.

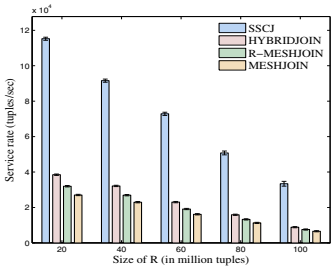
Analysis by varying size of memory M: In our first experiment we compare the service rate produced by all four algorithms by varying the memory size M from 1% to 10% of R while the size of R is 100 million tuples (≈11.18GB). We also fix the skew value equal to 1 for all settings of M. The results of our experiment are presented in Figure 4(a). From the figure it can be noted that SSCJ performs up to 4.5 times faster than HYBRIDJOIN in the case of a 10% memory setting. While in the case of a limited memory environment (1% of R) SSCJ still performs up to 3 times better than HYBRIDJOIN making it an adaptive solution for memory constraint applications. SSCJ also performs significantly better than both R-MESHJOIN and MESHJOIN.

Analysis by varying size of R: In this experiment we compare the service rate of SSCJ with the other three algorithms at different sizes of R under fixed memory size, ≈1.12GB. We also fix the skew value equal to 1 for all settings of R. The results of our experiment are shown in Figure 4(b). From the figure it can be seen that SSCJ performs up to 3 times better than HYBRIDJOIN under all settings of R. On the other hand if we compare the performance of SSCJ with MESHJOIN and R-MESHJOIN, it also performs significantly better than both of the algorithms under all settings of R.

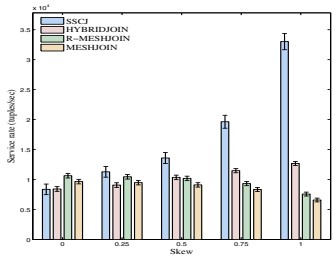
Analysis by varying skew value: In this experiment we compare the service rate of all the algorithms by varying the skew value in the streaming data. To vary the skew, we vary the value of the Zipfian exponent. In our experiments we allow it to range from 0 to 1. At 0 the input stream S is completely uniform while at 1 the stream has a larger skew. We consider the sizes of two other parameters, memory and R, to be fixed. The size of R is 100 million tuples (≈11.18GB) while the available memory is set to 10% of R (≈1.12GB). The results presented in Figure 4(c) show that SSCJ again performs significantly better among all approaches even for only moderately skewed data. Also this improvement becomes more pronounced for increasing skew values in the streaming data. At skew value equal to 1, SSCJ performs about 3 times better than HYBRIDJOIN. Contrarily, as MESHJOIN and R-MESHJOIN do not exploit the data skew in



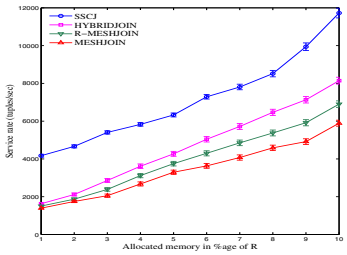
(a) Size of allocated memory varies



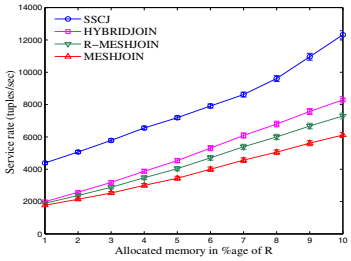
(b) Size of relation on disk varies



(c) Skew in data stream varies



(d) TPC-H dataset

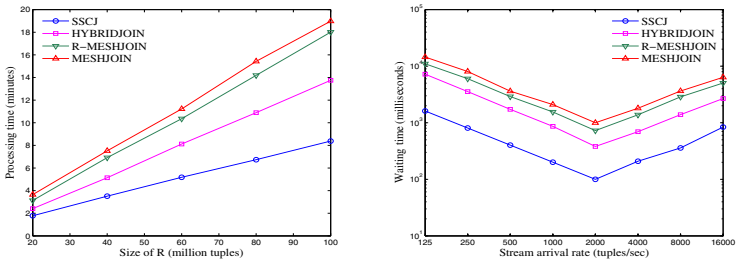


(e) Real-life dataset

Fig. 4. Performance analysis

their algorithms, their service rates actually decrease slightly for more skewed data, which is consistent to the original algorithms findings. We do not present data for skew value larger than 1, which would imply short tails. However, we predict that for such short tails the trend continues. SSCJ performs slightly worse than MESHJOIN and R-MESHJOIN only in a case when the stream data is completely uniform. In this particular case the stream-probing phase does not contribute considerably while on the other hand random access of R influences the seek time.

TPC-H and real-life datasets: We also compare the service rate of all the algorithms using TPC-H and real-life datasets. The details of both datasets have already been described in Section 5.1. In both experiments we measure



(a) Processing time (b) Waiting time

Fig. 5. Time analysis

the service rate produced by all four algorithms at different memory settings. The results of our experiments using TPC-H and real-life datasets are shown in Figures 4(d) and 4(e) respectively. From both figures it can be noted that the service rate in case of SSCJ is remarkably better among all three approaches.

Time Analysis: A second kind of performance parameter besides service rate refers to the time an algorithm takes to process a tuple. In this section, we analyze both waiting time and processing time. *Processing time* is an average time that every stream tuple spends in the join window from loading to matching without including any delay due to a low arrival rate of the stream. *Waiting time* is the time that every stream tuple spends in the stream buffer before entering into the join module. The waiting times were measured at different stream arrival rates. The experiment, shown in Figure 5(a), presents the comparisons with respect to the processing time. From the figure it is clear that the processing time in case of SSCJ is significantly smaller than the other three algorithms. This difference becomes even more pronounced as we increase the size of R. The plausible reason for this is that in SSCJ a big part of stream data is directly processed through the stream-probing phase without joining it with the whole relation R in memory.

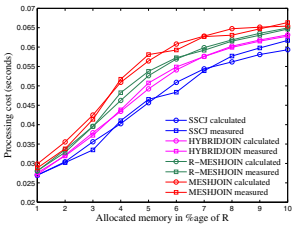


Fig. 6. Cost validation

In the experiment shown in Figure 5(b) we compare the waiting time for each of the algorithms. It is obvious from the figure that the waiting time in the case of SSCJ is significantly smaller than the other three algorithms. The reason behind this is that in SSCJ since there is no constraint to match each stream tuple with the whole of R, each disk invocation is not synchronized with the stream input.

Cost Analysis: The cost models for all the algorithms have been validated by comparing the calculated cost with the measured cost. Figure 6 presents the comparisons of both costs for each algorithm. The results presented in the figure show that for each algorithm the calculated cost closely resembles the measured cost, which proves the correctness of our cost models.

6 Conclusions

In this paper we propose a new semi-stream-based join called SSCJ and we compare it with HYBRIDJOIN and other earlier well-known semi-stream join algorithms. SSCJ is designed to make use of skewed, non-uniformly distributed data as found in real-world applications. In particular we consider a Zipfian distribution of foreign keys in the stream data. Contrary to HYBRIDJOIN, SSCJ stores these most frequently accessed tuples of R permanently in memory saving a significant disk I/O cost and accelerating the performance of the algorithm. We have derived a cost model for the new algorithm and validated it with experiments. We have provided an extensive experimental study showing an improvement of SSCJ over the earlier HYBRIDJOIN and other related algorithms.

References

1. Anderson, C.: The Long Tail: Why the Future of Business Is Selling Less of More. Hyperion (2006)
2. Bornea, M.A., Deligiannakis, A., Kotidis, Y., Vassalos, V.: Semi-streamed index join for near-real time execution of ETL transformations. In: IEEE 27th International Conference on Data Engineering (ICDE 2011), pp. 159–170 (April 2011)
3. Chakraborty, A., Singh, A.: A partition-based approach to support streaming updates over persistent data in an active datawarehouse. In: IPDPS 2009: Proceedings of the 2009 IEEE International Symposium on Parallel & Distributed Processing, pp. 1–11. IEEE Computer Society, Washington, DC (2009)
4. Karakasidis, A., Vassiliadis, P., Pitoura, E.: ETL queues for active data warehousing. In: IQIS 2005: Proceedings of the 2nd International Workshop on Information Quality in Information Systems, pp. 28–39. ACM, New York (2005)
5. Knuth, D.E.: The art of computer programming, 2nd edn. Sorting and searching, vol. 3. Addison Wesley Longman Publishing Co., Inc., Redwood City (1998)
6. Asif Naeem, M., Dobbie, G., Weber, G.: An event-based near real-time data integration architecture. In: EDOCW 2008: Proceedings of the 2008 12th Enterprise Distributed Object Computing Conference Workshops, pp. 401–404. IEEE Computer Society, Washington, DC (2008)
7. Asif Naeem, M., Dobbie, G., Weber, G.: HYBRIDJOIN for near-real-time data warehousing. International Journal of Data Warehousing and Mining (IJDWM) 7(4), 21–42 (2011)
8. Asif Naeem, M., Dobbie, G., Weber, G., Alam, S.: R-MESHJOIN for near-real-time data warehousing. In: DOLAP 2010: Proceedings of the ACM 13th International Workshop on Data Warehousing and OLAP, Toronto, Canada. ACM (2010)
9. Polyzotis, N., Skiadopoulos, S., Vassiliadis, P., Simitsis, A., Frantzell, N.E.: Supporting streaming updates in an active data warehouse. In: ICDE 2007: Proceedings of the 23rd International Conference on Data Engineering, Istanbul, Turkey, pp. 476–485 (2007)
10. Polyzotis, N., Skiadopoulos, S., Vassiliadis, P., Simitsis, A., Frantzell, N.: Meshing streaming updates with persistent data in an active data warehouse. IEEE Trans. on Knowl. and Data Eng. 20(7), 976–991 (2008)

Metrics to Support the Evaluation of Association Rule Clustering

Veronica Oliveira de Carvalho[1], Fabiano Fernandes dos Santos[2], and Solange Oliveira Rezende[2]

[1] Instituto de Geociências e Ciências Exatas,
UNESP - Univ Estadual Paulista, Rio Claro, Brazil
veronica@rc.unesp.br
[2] Instituto de Ciências Matemáticas e de Computação,
USP - Universidade de São Paulo, São Carlos, Brazil
{fabianof,solange}@icmc.usp.br

Abstract. Many topics related to association mining have received attention in the research community, especially the ones focused on the discovery of interesting knowledge. A promising approach, related to this topic, is the application of clustering in the pre-processing step to aid the user to find the relevant associative patterns of the domain. In this paper, we propose nine metrics to support the evaluation of this kind of approach. The metrics are important since they provide criteria to: (a) analyze the methodologies, (b) identify their positive and negative aspects, (c) carry out comparisons among them and, therefore, (d) help the users to select the most suitable solution for their problems. Some experiments were done in order to present how the metrics can be used and their usefulness.

Keywords: Association Rules, Clustering, Pre-processing.

1 Introduction

In the last years, researches have adopted some strategies to aid the user to find the relevant associative patterns of the domain. One of these strategies is to pre-process the data before obtaining the rules. For that, many approaches have been proposed, being clustering a promising one. In this case, the data are initially grouped into n groups. Association rules are extracted within each group and, in the end, n groups of rules are obtained. All these rules compose the rule set. According to [1], each group expresses its own associations without the interference of the other groups that contain different association patterns. The aim is to obtain potentially interesting rules that would not be extracted from unpartitioned data sets. The user must set the minimum support to a low value to discover these same patterns from unpartitioned data sets, causing a rapidly increase in the number of rules.

Distinct methodologies have been proposed to enable the described process. Each methodology uses a different combination of similarity measures with clustering algorithms to obtain the groups of rules. However, little has been done to

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 248–259, 2013.

analyze the performance of the methodologies or even to compare the results. So, there are some issues that have to be investigated:

Issue 1. Is there overlap between a rule set obtained through partitioned data, i.e., extracted from clustered data, in relation to a rule set obtained through unpartitioned data, i.e., extracted from traditional process? A ***r***ule ***s***et obtained through a ***p***artitioned data is named here as RsP and a ***r***ule ***s***et obtained through a ***t***raditional process is named here as RsT.

Issue 2. Is there overlap between the rules in RsT and RsP regarding the interesting knowledge? In other words, has RsP, in fact, more interesting patterns than RsT?

Issue 3. What is the process behavior regarding the number of rules that are obtained in RsP?

Based on the exposed arguments and on the three presented issues, nine metrics are proposed in this paper to support the evaluation of the methodologies that use clustering in the pre-processing step. Thereby, this paper will contribute with future researches since the metrics will provide criteria to: (a) analyze the methodologies, (b) identify their positive and negative aspects, (c) carry out comparisons among them and, therefore, (d) help the users to select the most suitable solution for their problems. It is important to say that the aim here is not to discover interesting rules, but to provide a standardized assessment procedure to support the evaluation of the methodologies that use clustering in the pre-processing step in order to discover the interesting rules.

This paper is organized as follows. Section 2 presents the proposed metrics. Section 3 describes some experiments that were carried out to show how the metrics can be used. Section 4 discusses the results obtained in the experiments. Section 5 surveys the related researches. Finally, conclusion is given in Section 6.

2 Proposed Evaluation Metrics

Nine metrics are proposed to support the evaluation of the methodologies that use clustering in the pre-processing step, as the methodologies described in Section 5. Each metric is related to an issue mentioned in Section 1. For each issue there are one or more metrics. To propose the metrics, we assume that RsP is better than RsT when it generates new knowledge in a few groups.

All metrics, with exception to M_{NR-RsP}, range from 0 to 1. Since RsP contains all the rules extracted within each group, repeated rules may exist in the set. In RsT the same doesn't occur since the rules are unique. Thus, it is important to notice, in the equations presented below, that although RsP is a set, it may have repeated elements, different from the traditional set theory. Thereby, in the following operations the resulting sets may contain some repeated rules.

Issue 1. Regarding the existing overlap among the rules in RsP and RsT, four metrics are proposed, which are described as follows:

M_{O-RsP} Measures the ratio of "old" rules in RsP, i.e., the ratio of rules in RsT found in RsP (Equation 1). A rule is considered "old" if it is in RsT, i.e., in the rule set obtained through the traditional process. Therefore, the higher

the value the better the metric, since the value indicates that there was no loss of knowledge during the process.

$$M_{O-RsP} = \frac{|RsT \cap RsP|}{|RsT|} . \tag{1}$$

M_{N-RsP} Measures the ratio of "new" rules in RsP, i.e., the ratio of rules in RsP not found in RsT (Equation 2). A rule is "new" if it isn't in RsT, i.e., in the rule set obtained through the traditional process. Although it is important that any knowledge be lost (metric M_{O-RsP}), it is expected that the ratio of "new" rules in RsP be greater than the ratio of "old" rules. Therefore, the higher the value the better the metric, since the value indicates the amount of knowledge, previously unknown, obtained during the process.

$$M_{N-RsP} = \frac{|RsP - RsT|}{|RsP|} . \tag{2}$$

$M_{R-O-RsP}$ Measures the ratio of "old" rules that repeat in RsP (Equation 3). It is considered that a rule should exist in only one of the clustering groups, since it has to be in a subdomain that expresses its own associations. Therefore, the lower the value the better the metric, since the value indicates that the knowledge, already known, is in subsets that express its own associations.

$$M_{R-O-RsP} = \frac{FindRepetitionRsP(RsT \cap RsP)}{|RsT \cap RsP|} , \tag{3}$$

FindRepetitionRsP: function that receives by parameter a set of non repeated rules and returns the number of rules in the set that repeat in RsP.

$M_{R-N-RsP}$ Measures the ratio of "new" rules that repeat in RsP (Equation 4). Idem to $M_{R-O-RsP}$. Therefore, as in $M_{R-O-RsP}$, the lower the value the better the metric, since the value indicates that the knowledge, previously unknown, is in subsets that express its own associations.

$$M_{R-N-RsP} = \frac{FindRepetition(RsP - RsT)}{|RsP - RsT|} , \tag{4}$$

FindRepetition: function that receives by parameter a set that may contain repeated rules and returns the number of rules in the set that repeat.

Issue 2. Regarding the existing overlap among the rules in RsP and RsT considering the interesting aspect of the knowledge, four metrics are proposed, which are described as follows:

$M_{N-I-RsP}$ Measures the ratio of "new" rules among the h-top interesting rules in RsP (Equation 5). Given a subset of h-top interesting rules, selected from RsP, it is expected that the ratio of "new" rules in this subset be as large as possible. The h-top rules are the h rules that contain the highest values regarding an objective measure, where h is a number to be chosen. Therefore, the higher the value the better the metric, since the value indicates that the cost of the process is minimized by the discovery of interesting knowledge, previously unknown, in RsP.

$$M_{N-I-RsP} = \frac{CountTopRules(h_{top\ of\ RsP}, RsP - RsT)}{|h_{top\ of\ RsP}|}\ , \tag{5}$$

CountTopRules: function that receives by parameter a set of h-top interesting rules and a set of rules and returns the number of rules that appears among the h-top.

$M_{O-I-N-RsP}$ Measures the ratio of "old" rules not in RsP among the h-top interesting rules in RsT (Equation 6). Given a subset of h-top interesting rules, selected from RsT, it is expected that all these rules are present in RsP. It is not desirable that the interesting patterns in RsT disappear in RsP, which would imply in the loss of relevant knowledge. Thus, this metric measures the ratio of "old" interesting rules not in RsP. The h-top rules are as described in $M_{N-I-RsP}$. Therefore, the lower the value the better the metric, since the value indicates that the interesting knowledge in RsT was not lost during the process.

$$M_{O-I-N-RsP} = \frac{CountTopRules(h_{top\ of\ RsT}, RsT - RsP)}{|h_{top\ of\ RsT}|}\ , \tag{6}$$

CountTopRules: idem Equation 5.

M_{C-I} Measures the ratio of common rules among the h-top interesting rules in RsP and the h-top interesting rules in RsT (Equation 7). Consider two subsets, S_1 and S_2, containing, respectively, the h-top interesting rules in RsP and the h-top interesting rules in RsT. This metric measures the existing intersection between these two subsets, which is expected to be as small as possible. Therefore, the lower the value the better the metric. The higher the intersection, the less relevant will be the process, since all the knowledge already known as interesting in RsT is also identified as interesting in RsP, not providing to the process any additional relevant information.

$$M_{C-I} = \frac{|h_{top\ of\ RsP} \cap h_{top\ of\ RsT}|}{h}\ , \tag{7}$$

h is the number to be chosen to realize the selection of the rules in both sets, i.e., RsP and RsT.

$M_{NC-I-RsP}$ Measures the ratio of groups in the clustering related to RsP that contains the h-top interesting rules in RsP (Equation 8). Therefore, the lower the value the better the metric. This means that just some of the groups would have to be explored by the user, which will contain the "new" relevant knowledge extracted during the process.

$$M_{NC-I-RsP} = \frac{FindGroups(h_{top\ of\ RsP})}{N}\ , \tag{8}$$

N: number of groups in the clustering; FindGroups: function that receives by parameter a set of h-top interesting rules, finds their groups and returns the number of distinct selected groups.

Issue 3. Regarding the process behavior related to the number of rules that are obtained in RsP, a unique metric is proposed, which is described as follows:

M_{NR-RsP} Measures the ratio of rules in RsP in relation to RsT (Equation 9). It is important to analyze the process in relation to the number of rules in RsP. It is not desirable to have a large increase in the volume of rules, because even if new patterns are discovered, it will be harder for the user to identify them. Therefore, the lower the value the better the metric, since the value indicates that although new patterns have been extracted, the number of extracted rules is not big enough to overload the user.

$$M_{NR-RsP} = \frac{|RsP|}{|RsT|} \,. \tag{9}$$

Relating the proposed metrics with the researchers found in the literature (Section 5), it can be observed that: (a) [1] is the only work that provides a similar analysis related to the metrics M_{O-RsP} and M_{N-RsP} in *Issue 1*; (b) none of them provide an analysis related to the aspects covered by *Issue 2*; [2,3,1] provide a similar analysis related to the metric M_{NR-RsP} in *Issue 3*. Thus, the necessity of a standardized assessment procedure becomes evident (more details in Section 5). Finally, it is important to say that we believe that these nine metrics cover, adequately, the three presented issues. However, as other issues arise, new metrics can be added to this assessment procedure.

3 Experiments

Some experiments were carried out in order to present how the metrics can be used. For that, two contexts were defined. Suppose a user decides to apply clustering in the pre-processing step. First of all, he has to find out the most suitable methodology to be used in his problem. After that, he has to check if the selected methodology was good enough for the problem, considering that different interests may be important for his decision. Thus, two different situations were regarded: (i) identify among some organizations the most suitable; (ii) analyze the process itself. An organization is obtained by the application of a clustering algorithm combined with a similarity measure. Therefore, the metrics provide the support to evaluate each situation under the discussed issues: while in (i) the data is initially clustered through some organizations in order to identify the organization that obtains a good association set, in (ii) the usefulness of the process itself is analyzed. Four data sets and four organizations were selected to be used in the experiments.

The four data sets were Adult (48842;115), Income (6876;50), Groceries (9835;169) and Sup (1716;1939). The numbers in parenthesis indicate, respectively, the number of transactions and the number of distinct items in each data set. The first three are available in the R Project for Statistical Computing through the package "arules"[1]. The last one was donated by a supermarket located in São Carlos city, Brazil. All the transactions in Adult and Income contain the same number of items (named here as standardized data sets (S-DS)), different from Groceries and Sup (named here as non-standardized data sets

[1] `http://cran.r-project.org/web/packages/arules/index.html`.

(NS-DS)), where each transaction contains a distinct number of items. Thus, the experiments considered different data types.

The four organizations were obtained by the combination of the algorithms and similarity measures presented in Table 1. Each combination gives an organization, i.e., a different way to analyze the process. Although it is necessary to set k, the number of groups to be generated, in order to obtain an organization, this value was used to analyze the organizations on different views. Despite the existence of algorithms designed for transactions, such as ROCK, the choices of the algorithms were made based on works that cluster the rules in the post-processing phase aiming a *posteriori* comparison. The similarity measures were chosen considering the works described in Section 5 – only the similarities among transactions were selected.

As described before, the rules are extracted within each group after clustering the data. The values of the minimum support (min-sup) and minimum confidence (min-conf) have to be set in order to extract a set of association rules. To automate the specification of the min-sup in each group, the following procedure was adopted: (i) find the 1-itemsets of the group with their supports, (ii) compute the average of these supports, (iii) use this average support as the min-sup of the group. Regarding min-conf, the following values were used for each data set: Adult 50%; Income 50%; Groceries 10%; Sup 100%. Thus, the same min-conf value was applied in all the groups of a given data set. These values were chosen experimentally. Although it is known that min-sup and min-conf impacts on the set of rules that are obtained, it was assumed that the focus was on the use of the metrics and, so, that the values were adequate to the proposed problem. Finally, the rules were extracted with an Apriori implementation developed by Christian Borgelt[2] with a minimum of two items and a maximum of five items per rule.

Considering the four organizations, the RsP sets were obtained. However, once almost all the metrics are based on the rules obtained through the traditional process, the four data sets were also processed to obtain the RsT sets. For that, the min-sup was set automatically, as described before. Regarding min-conf, the same values used in RsP were considered, i.e., Adult 50%, Income 50%, Groceries 10% and Sup 100%. Furthermore, as some of the metrics are based on the h-top interesting rules of a given rule set, an objective measure should be selected. Instead of choosing a specific measure, the average rating obtained through 18 objective measures (see Table 1) was considered as follows: (i) the value of 18 measures was computed for each rule; (ii) each rule received 18 ID's, each ID corresponding to the rule position in one of the ranks related to a measure; (iii) the average was then calculated based on the rank positions (ID's). Thus, the h-top rules were selected considering the best average ratings. h, also a number to be set, was defined, in all the sets (RsT and RsP), to assume 0.5% of the total of rules in RsT – as seen in Section 4, always the smallest set. Therefore, each rule set contains its own values that are proportional in all of them. Table 1 summarizes the configurations used to apply the proposed metrics.

[2] `http://www.borgelt.net/apriori.html`.

Table 1. Configurations used to apply the proposed metrics

Data sets	Adult; Income; Groceries; Sup
Algorithms	PAM; Ward [algorithms details in [5]]
Similarity measures	Agg; Denza
k	5 to 25, steps of 5
h	0.5% of the total of rules in RsT
Objective measures [measures details in [6]]	Added Value, Certainty Factor, Collective Strength, Confidence, Conviction, IS, ϕ-coefficient, Gini Index, J-Measure, Kappa, Klosgen, λ, Laplace, Lift, Mutual Information (asymmetric), Novelty, Support, Odds Ratio

4 Results and Discussion

Considering the configurations presented in Table 1 and the RsT sets above described, the experiments were carried out and the values of each metric obtained. Regarding the first proposed situation, i.e., identify among some organizations the most suitable (Section 3), an analysis based on the average of each metric, considering the different data types, apart from the data set, was carried out. Table 2 presents the results. Thus, in this case, the metrics will help the users to find out a suitable methodology for their problems. In order to aid the comparison of the results, all the metrics that present better results when their values are the smallest ($M_{R-O-RsP}$, for example) were processed to store the complement of the information. Therefore, all the metric, with exception to M_{NR-RsP}, have the same interpretation: the higher the value the better the performance. Furthermore, all the metrics can be seen in terms of percentage if multiplied by 100 (ex.: 0.858*100 = 85.8%).

Table 2. Average of the proposed metrics in the considered organizations

Data type	Algorithm	Measure	M_{O-RsP}	M_{N-RsP}	$M_{R-O-RsP}$	$M_{R-N-RsP}$	$M_{N-I-RsP}$	$M_{O-I-N-RsP}$	M_{C-I}	$M_{NC-I-RsP}$	M_{NR-RsP}
S-DS	PAM	Agg	0.858▲	0.906	0.239△	0.881△	0.716	0.891▲	0.838	0.596△	125.370▲
		Denza	0.730	0.936△	0.235	0.878	0.874▲	0.583	0.912△	0.563	160.534
	Ward	Agg	0.718	0.923	0.213△	0.871	0.920△	0.466	0.967△	0.503	129.681▲
		Denza	0.722△	0.928△	0.209	0.877△	0.870	0.509△	0.919	0.533△	133.994
NS-DS	PAM	Agg	0.880	0.901△	0.709▲	0.974△	0.750△	1.000△	1.000△	0.909▲	269.830▲
		Denza	0.924△	0.885	0.510	0.940	0.745	0.986	0.946	0.785	431.619
	Ward	Agg	0.976△	0.245	0.947▲	0.999△	0.211	0.997△	0.297	0.828	1.652▲
		Denza	0.973	0.693▲	0.755	0.974	0.604▲	0.997△	0.684▲	0.867△	221.917

Each average in Table 2 was obtained from the results of the experiments related to the presented configuration. The value 0.858 in M_{O-RsP} at S-DS:-PAM:Agg, for example, was obtained by the average of the values in M_{O-RsP} at Adult:PAM:Agg and Income:PAM:Agg over the values of k. The highest averages are marked with △ in each algorithm regarding each metric. The only exception is M_{NR-RsP}, where the lowest averages are highlighted. For the S-DS:PAM configuration, for example, the best average for M_{O-RsP} is the one related to Agg (0.858). However, since the averages are, in general, near, a marking based on the difference among the averages was also considered. The values marked with ▲ indicate that the difference between the averages of a given metric are

above 0.1 ($dif. \geq 0.1$). For the S-DS:PAM configuration, for example, the best average for M_{O-RsP} is the one related to Agg (0.858), presenting a difference of 0.128 in relation to Denza (0.858-0.730). Thereby, it is possible to visualize, for each configuration, the most suitable similarity measure. It is important to mention that the results are deterministic and, therefore, no statistical test was done to check if there is a significant difference among the averages. It can be noticed that:

S-DS:PAM. The most suitable measure for this configuration is Agg, since it presents better results in 6 of the 9 metrics in relation to Denza. Furthermore, in 3 of the 6 metrics Agg exhibits a difference above 0.1 in relation to Denza. In these cases, it can be noticed that the values in Agg are more representative than the values in Denza – observe, for example, that while Agg in $M_{O-I-N-RsP}$ presents a performance above 89%, Denza presents a performance below 59%.

S-DS:Ward. Although Denza presents a better performance in relation to Agg in 5 of the 9 metrics, Agg seems to be the most suitable measure for this configuration even presenting a better performance in 4 of the 9 metrics. This occurs because while Agg exhibits a difference above 0.1 in relation to Denza in 1 of the 4 metrics, Denza doesn't present any difference in any of the metrics. Although the difference occurs in only one of the metrics, the metric is important, since it measures how much the exploration space increases in relation to RsT.

NS-DS:PAM. The most suitable measure for this configuration is Agg, since it presents better results in 8 of the 9 metrics in relation to Denza. Furthermore, in 3 of the 8 metrics Agg exhibits a difference above 0.1 in relation to Denza. In these cases, it can be noticed that the values in Agg are more representative than the values in Denza – observe, for example, that while Agg in $M_{R-O-RsP}$ presents a performance above 70%, Denza presents a performance below 52%.

NS-DS:Ward. Both measures present a good performance in 4 of the 9 metrics, excluding the tie occurred in $M_{O-I-N-RsP}$. However, in 3 of the 4 metrics Denza exhibits a difference above 0.1 in relation to Agg. In these cases, it can be noticed that the values in Denza are more representative than the values in Agg – observe, for example, that while Denza in M_{N-RsP} presents a performance above 69%, Agg presents a performance below 25%. Therefore, the most suitable measure for this configuration is Denza.

Considering the exposed arguments, it can be noticed that:

S-DS. Comparing the results of PAM:Agg and Ward:Agg, PAM presents better results in 6 of the 9 metrics in relation to Ward (M_{O-RsP}, $M_{R-O-RsP}$, $M_{R-N-RsP}$, $M_{O-I-N-RsP}$, $M_{NC-I-RsP}$, M_{NR-RsP}) and a difference above 0.1 in 3 of the 6 metrics (M_{O-RsP}, $M_{O-I-N-RsP}$, M_{NR-RsP}). Therefore, the most suitable organization, to this type of data, according to the metrics, is PAM:Agg.

NS-DS. Comparing the results of PAM:Agg and Ward:Denza, PAM presents better results in 5 of the 9 metrics in relation to Ward (M_{N-RsP}, $M_{N-I-RsP}$, $M_{O-I-N-RsP}$, M_{C-I}, $M_{NC-I-RsP}$), excluding the tie occurred in $M_{R-N-RsP}$, and a difference above 0.1 in 3 of the 5 metrics (M_{N-RsP}, $M_{N-I-RsP}$, M_{C-I}).

Therefore, the most suitable organization, to this type of data, according to the metrics, is also PAM:Agg.

As observed, the most suitable organization according to the metrics, regarding the presented configurations (Table 1), apart from the data type used, is PAM:Agg. In other words, the user will obtain better results, i.e., reasonable rule set, if he initially clusters the data through PAM:Agg. However, in other domains, different aspects can be of interesting, providing the user a flexible way to solve his issues. Thus, in this first situation, the metrics provide criteria to carry out comparisons, helping the user to select the most suitable methodology for his problem.

From that point, supposing that PAM:Agg is a suitable solution for the user's problem, it is possible to analyze the process itself, i.e., to check if good results are really obtained. Observe that different interests may be important for his decision. Thus, the metrics provide criteria not only to analyze the process, but also to identify its positive and negative aspects, helping the user to reach a conclusion. To discuss this second situation, Table 3 presents the values of the metrics, in the selected organization, in each one of the data types. These values are the ones presented at S-DS:PAM:Agg and NS-DS:PAM:Agg in Table 2, but in their original scales, since the smaller scales (↓) were previously converted – the larger scales (↑) remain the same. The scale, in each metric, is found between "[]". It can be noticed that:

M_{O-RsP} Little knowledge is lost during the process, around 15%, since more than 85% of the rules in RsT are found in RsP. Thus, a positive aspect of the process is identified.

M_{N-RsP} Almost all the rules in RsP are "new", around 90%, indicating the discovery of a great amount of knowledge previously unknown. Thus, a positive aspect of the process is identified.

$M_{R-O-RsP}$ The repetition of "old" rules in RsP is high in both data types, around 30% at NS-DS and 77% at S-DS. Thus, a negative aspect of the process is identified.

$M_{R-N-RsP}$ The repetition of "new" rules in RsP is low, around 12%, indicating that the knowledge, previously unknown, is in subdomains that express their own associations. Thus, a positive aspect of the process is identified.

$M_{N-I-RsP}$ A great amount of the h-top interesting rules in RsP are "new", around 71%, indicating that the cost of the process is minimized by the discovery of interesting knowledge, previously unknown, in RsP. Thus, a positive aspect of the process is identified.

$M_{O-I-N-RsP}$ The loss of "old" and interesting knowledge is low, around 11%, since a great amount of the h-top interesting rules in RsT are found in RsP, around 89% (100%-11%). Thus, a positive aspect of the process is identified.

M_{C-I} The intersection between the h-top interesting rules in RsP and the h-top interesting rules in RsT is low, around 17%, indicating that few of the knowledge already known as interesting in RsT is found in RsP. Thus, a positive aspect of the process is identified.

$M_{NC-I-RsP}$ The number of groups that contain the h-top interesting rules in RsP at NS-DS is low, around 10%, which doesn't occur at S-DS, that is around 41%. Thus, a negative aspect of the process is identified.

M_{NR-RsP} The number of rules in RsP is far greater in relation to RsT, overloading the user with an excessive number of rules. Thus, a negative aspect of the process is identified.

Table 3. Average of the proposed metrics in the PAM:Agg organization

Data type	M_{O-RsP} [↑]	M_{N-RsP} [↑]	$M_{R-O-RsP}$ [↓]	$M_{R-N-RsP}$ [↓]	$M_{N-I-RsP}$ [↑]	$M_{O-I-N-RsP}$ [↓]	M_{C-I} [↓]	$M_{NC-I-RsP}$ [↓]	M_{NR-RsP} [↓]
S-DS	0.858	0.906	0.761	0.119	0.716	0.109	0.162	0.404	125.370
NS-DS	0.880	0.901	0.291	0.026	0.750	0.000	0.000	0.091	269.830

Summarizing, it can be observed that: (a) a great amount of interesting knowledge, previously unknown, is discovered, which are in subdomains that express their own associations; (b) little interesting knowledge is lost, which are not in subdomains that express their own associations; (c) since the number of rules is high and the interesting knowledge, previously unknown, is spread over the clustering groups, the user exploration can be hampered. Therefore, considering the positive and negative aspects of the process, the user can analyze the results, according to his interests, and conclude if good results were reached. It is relevant to say that the importance of each percentage depends on the user's needs, on the data sets, etc., and, therefore, has to be, in fact, validate by them. Regarding the presented context, it can be said that the process obtains reasonable results, since 6 of the 9 aspects were considered positives. However, if the weight of the 3 negative metrics is more important to the user, he can discard the results. Moreover, he can try to improve the process to obtain better results in these metrics. Thus, in this second situation, the metrics provide criteria to analyze the process based on different interests, identifying its positive and negative aspects, helping the user to reach a conclusion.

5 Related Works

There are many researches that initially cluster data aiming to discover and facilitate the search for the interesting pattern of the domain. Some of these works are described below.

[2] propose to split the data set items into groups in order to extract the rules. The authors evaluate many hierarchical algorithms combined with many similarity measures. Nevertheless, it is not understandable how the rules are obtained within the groups, since it is necessary to have a set of transactions and not a set of items. This means that it is not clear how the transactions are distributed over the groups. Among the similarity measures used by them, we emphasize Jaccard due to its use by the measure described below (Agg). The Jaccard between two items i_1 and i_2, expressed by P-J(i_1,i_2)=$\frac{|\{t \text{ covered by } i_1\} \cap \{t \text{ covered by } i_2\}|}{|\{t \text{ covered by } i_1\} \cup \{t \text{ covered by } i_2\}|}$, is the

ratio between the transactions t the items cover simultaneously and the total of transactions the items cover. An item covers a transaction t if the item is in t.

[3] propose an algorithm, named CLASD, to split the data set transactions aiming to discover associations on small segments (subsets) of the data. To cluster the transactions, a similarity measure proposed by them, named Agg, expressed by $\text{Agg}(t_1,t_2)=\frac{\sum_{p=1}^{m}\sum_{q=1}^{n} Af(i_p,j_q)}{m*n}$, is used. Thus, the similarity between two transactions t_1 and t_2 is computed by the affinity (Af) average among the transaction items, being Af equivalent to the measure P-J. Therefore, after computing P-J among the m items in t_1 and the n items in t_2, the average among them is obtained.

[1] propose an algorithm, named *Apriori Inverse*, to cluster the transactions and then extract a rare association rule set. To cluster the transactions, their algorithm initially finds k seeds (centroids), where k indicates the number of frequent itemsets that match some conditions. Each seed forms a group. After the seed generation, each transaction t is allocated to one of the groups based on the number of common items that occur between the transaction (t) and the group centroid.

There are other researches concerned with the clustering of transactions that, although not related to the extraction of association rules, could be used. In [4], for example, the authors propose an approach to identify, a priori, the potentially interesting items to appear in the antecedents and in the consequents of the association rules without extracting them. The approach is divided in two steps: the clustering of the transactions and the selection of the interesting items. To do the clustering the authors propose the use of incremental K-means with a similarity measure obtained through a Jaccard between transactions, expressed by $\text{Denza}(t_1,t_2)=\frac{|\{items\ in\ t_1\}\cap\{items\ in\ t_2\}|}{|\{items\ in\ t_1\}\cup\{items\ in\ t_2\}|}$. Therefore, the similarity between two transactions t_1 and t_2 is computed considering the items the transactions share.

Among the papers above described, little has been done to analyze the performance of the methodologies, allowing to identify their positive and negative aspects, or even to compare the results among them. In general, the researchers compare the number of rules and/or itemsets that are obtained from unpartitioned data and clustered data to expose the usefulness of the methodologies. This strategy can be found in [2,3,1] and is related to "Issue 3" of Section 1. However, [2] also analyze the process considering the complexity of the rules that are obtained – the greater the number of items that compose a rule the higher its complexity. [3,1] discuss over some rules found through clustering to show that the process provides the discovery of interesting patterns, but the analysis of the process is subjective. [3] also consider the execution time. Finally, [1] is the only work that allows a better analysis considering the existing overlap between the rules obtained from unpartitioned data and clustered data. This strategy is related to "Issue 1" of Section 1. Based on the presented arguments, as mentioned before, the necessity of a standardized assessment procedure becomes evident.

6 Conclusion

In this paper, nine metrics were proposed to support the evaluation of methodologies that use clustering in the pre-processing step to aid the discovery of associative interesting patterns. The metrics were developed to answer three main issues related to the presented context. Some experiments were carried out in order to present how the metrics can be used. For that, two different situations were regarded: (i) identify among some organizations the most suitable; (ii) analyze the process itself. Through the experiments, it could be noticed that the metrics provide criteria to: (a) analyze the methodologies, (b) identify their positive and negative aspects, (c) carry out comparisons among them and, therefore, (d) help the users to select the most suitable solution for their problems. Thus, based on the discussions, the usefulness and the importance of the metrics were demonstrated.

As a future work, to complement the results, an empirical study with human subjects will be done to verify if configurations with high metric values indeed produces rules that are more useful to end users. With this new analysis, we believe that a better understanding of the presented context will be reached and its importance highlighted. A new methodology that tries to optimize all these criteria, through an optimization technique, can be an interesting proposal.

Acknowledgments. We wish to thank Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) (processes numbers: 2010/07879-0 and 2011/19850-9) for the financial support.

References

1. Koh, Y.S., Pears, R.: Rare association rule mining via transaction clustering. In: 7th Australasian Data Mining Conference. CRPIT, vol. 87, pp. 87–94 (2008)
2. Plasse, M., Niang, N., Saporta, G., Villeminot, A., Leblond, L.: Combined use of association rules mining and clustering methods to find relevant links between binary rare attributes in a large data set. Computational Statistics & Data Analysis 52(1), 596–613 (2007)
3. Aggarwal, C.C., Procopiuc, C., Yu, P.S.: Finding localized associations in market basket data. IEEE Transactions on Knowledge and Data Engineering 14(1), 51–62 (2002)
4. D'Enza, A.I., Palumbo, F., Greenacre, M.: Exploratory data analysis leading towards the most interesting binary association rules. In: 11th Symposium on Applied Stochastic Models and Data Analysis, pp. 256–265 (2005)
5. Xu, R., Wunsch, D.: Clustering. IEEE Press Series on Computational Intelligence. Wiley (2008)
6. Tan, P.-N., Kumar, V., Srivastava, J.: Selecting the right objective measure for association analysis. Information Systems 29(4), 293–313 (2004)

An Automatic Email Management Approach Using Data Mining Techniques

Gunjan Soni and C.I. Ezeife*

School of Computer Science, University of Windsor, Windsor, ON, Canada
{sonig,cezeife}@uwindsor.ca

Abstract. Email mining provides solution to email overload problem by automatically placing emails into some meaningful and similar groups based on email subject and contents. Existing email mining systems such as BuzzTrack, do not consider the semantic similarity between email contents, and when large number of email messages are clustered to a single folder it retains the problem of email overload. The goal of this paper is to solve the problem of email overload through semantically structuring the user's email by automatically organizing email in folders and sub-folders using data mining clustering technique and extracting important terms from created folders using Apriori-based method for folder identification. This paper proposes a system named AEMS for automatic folder and sub-folder creation and later indexing the created folders. For AEMS module, a novel approach named Semantic non-parametric K-Means++ clustering is proposed for folder creation. Experiments show the effectiveness and efficiency of the proposed techniques using large volumes of email datasets.

Keywords: Email Mining, Email Overload, Email Management, Data Mining, Clustering, Feature Selection.

1 Introduction

Email is a widely used way of written communication over the internet. According to an estimate [1], the number of email messages sent daily has reached around *3.4 billion* in 2012, resulting in the evolution of the problem of email overload. Email overload [2] is a state of being completely overwhelmed of email inboxes by the amount of email one has received.

Email overload can be handled by managing email messages by summarization and automatically categorizing emails into folders. Automatic folder creation is, given a set of email messages and we need to semantically assign each message to similar groups according to the email content. Some automatic email folder creating techniques are given in [2], [3], [4] and [5]. Another solution to email overload is given by the email summarization ([6], [7]). The goal of email summarization is to provide

* This research was supported by the Natural Science and Engineering Research Council (NSERC) of Canada under an Operating grant (OGP-0194134) and a University of Windsor grant.

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 260–267, 2013.

concise, informative summary of email which in turn is helps to decide if the message demands immediate attention.

Both folder categorization and email summarization do not reduce email overload when email sender, subject and topic are not known. Therefore, our proposed approach overcomes this problem by folder creation based on email subject and content, sub-folder based on sender of the email and then index or view will be created in a separate web page, which contains link to the respective folder and sub-folder, and contains the summary of each folder.

1.1 Contributions

This paper proposes an algorithm AEMS (Automatic Email Management System) which manages emails by organizing similar emails in the folders (module 1 named AEG), then again organizes emails of each folder into subfolders (module 2 named APEG) where subfolder will contain emails sent by only a particular person and lastly creating the index, which contains name and link to the folders and sub-folders and also contains a summary annotation about the content of the respective folders.

For model AEG (Automatic Email Grouping), we have introduced document frequency based feature selection method named Associative term frequency. We also proposed novel Semantic Non-parametric K-Means++ Clustering method for folder creation, which avoids, (1) random seed selection by selecting the seed according to email weights, and (2) pre-defined number of clusters using the similarity between the email contents. Lastly, we have applied an Apriori-based folder summarization which extracts frequent patterns from the emails of respective folders useful for identification of content of folders.

2 Related Work

Similar to our AEG model, the work is shown in BuzzTrack [4], which used vector space model for email representation and cluster emails based on three measures: text similarity, subject similarity and people-based similarity. Next, kernel-selected email clustering [5] was introduced for email clustering. They consider the global weighting of each email subject and body for the creation of email VSM (vector space model) and then create email matrix and used an improved K-means clustering algorithm based on the lowest similarity. However, the work in [2], [3], [4] and [5] techniques are limited because if a created folder contains 2000 emails, it is hard to find an email of a specific individual whose name is not specifically known by the user.

On the other hand email summarization techniques such as: NLP and Machine Learning techniques based email summarization [6] extract the important candidate noun phrases (NPs) from the email messages and manually classify the selected NPs into those that should be or not included in the summary. These NPs are used to build training set which is then used to summarize incoming messages. Next, CWS [7] is email conversation summarizer which uses clue words to measure the importance of sentences in conversation summarization based on the clue words and sentence score

of a summary is produced. The work in [6] and [7] do not provide help to find a particular email when millions of emails are present.

3 The Proposed AEMS Model

In this section, AEMS module is presented which automatically and semantically arranges email in similar groups by summarizing the content of each group and creating a view called index. The AEMS module is divided into three sub-modules which include: Automatic Email Grouping (AEG), Automatic People based Email grouping (APEG) and Indexing.

3.1 AEG Model – Automatic Folder Creation and Topic Detection

The input as a set of email messages directly goes to AEG where, AEG is a process of creating main folder based on similar email messages and semantic similarity measures and includes the 4 stages of (1) pre-processing, (2) feature selection, (3) clustering algorithm and (4) topic detection with details presented next.

Step 1 and 2: Pre-processing and Feature Selection
For pre-processing, subject line and content of the email messages are extracted from each email and stop words are removed which can reduce the size of the email to be processed. Next, we review the features by taking each term from the processed data to calculate the associative term frequency (R_{tf}) of a particular term x, which is the number of emails that contain the term, x. Features will be selected according to the R_{tf}, where R_{tf} should be greater or equal to the user specified threshold, T_s and T_b depending on whether the term appears in subject or content of the email respectively.

$$R_{tf}(x) = (df_x * 100)/N \quad (1)$$

In equation 1, N is the total number of email messages in the dataset; df_x is the total number of emails in which the term *x* appeared. Once the feature terms are selected from the email, the email vector is created by combining the feature terms and removing the duplicate terms from the vector.

Step 3: Semantic Non-parametric K-Means++ Clustering Algorithm
Thirdly, step of the AEG process is to apply semantic_nonparametric_Kmeans++ algorithm of Fig. 2, where the emails are clustered together according to proposed the Semantic Non-parametric K-Means++ clustering algorithm. First, select the initial cluster center by calculating the email weight as shown in semantic_nonparametric_kmeans++ algorithm of Fig. 1, step 1. The initial cluster center is the email with the maximum weight, where email weight is the total number of feature terms in the email. After this, chose all other clusters center by calculating the similarity between all emails with the initial cluster center as shown in semantic_nonparametric_Kmeans++ algorithm of Fig. 1, step 2. Chose other clusters center

using a weighted probability distribution where an email x is chosen with probability proportional to $\mathrm{D}(X_{i,j})^2$ and $\mathrm{D}(X_{i,j})$ should be less or equal to β (the optimized value of β can overcome the problem of pre-defined cardinality of other clustering algorithms because once the β is set it will work for all type of data and user need not to give any input such as K in K-Means++), as shown in semantic_nonparametric_Kmeans++ algorithm of Fig. 1, step 3. The similarity $\mathrm{D}(X_{i,j})$(calculated using the semantic text similarity (STS) algorithm [8], which semantically finds similarity between two emails). Once all centers are created, then form the clusters by assigning email (X_i) to the cluster center C_k where, similarity $(\mathrm{D}(X_{k,i}))$ is minimum in comparison to other center, as in semantic_nonparametric_Kmeans++ algorithm (Fig. 1, step 4).

Algorithm: semantic_nonparametric_Kmeans++(X) – {Clustering algorithm}
Input: Email vector set (*X)*
Other Variables:
$\mathrm{D}(\mathrm{C}_{\mathrm{init}},x_i)$: Decimal value indicating similarity between initial cluster center and the email x_i
C_k: Email vector of text represented as cluster centers
β: Minimum threshold value, where $0.0 \leq \beta \leq 1.0$
|T|: Total number of features terms
E_w: Integer value as email weight (Total number of features)
$\mathrm{X_n}$: Particular email in email vector of text
C_{init}: Email vector of text represented as initial cluster center
Output: Set of clustered email represented as grouped text
Begin
1. FOR each email (X_i) in email vector set (X) DO
   1.1. Email weight of email X_i, $\mathrm{E_{wi}} = |\mathrm{T}|$ // Calculate each email weight.
   1.2 Initial cluster center, $X_j = \max(\mathrm{E_w})$ // Email having maximum weight is assigned as initial cluster center (x_j).
2. FOR each email (X_i) in email vector set (X) DO
   2.1 Calculate similarity $\mathrm{D}(X_{i,j})$// Calculate the similarity between initial cluster center and the each email using STS [8].
3. Choose all cluster centers X_k, select $X_k$with probability
$$\left(\frac{\mathrm{D}(X_{i,j})^2}{\sum_{x\in X}\mathrm{D}(X_{i,j})^2}\right) \text{ and } \mathrm{D}(X_{i,j}) \leq \beta$$
4. FOR each email (X_i) in email vector set (X) DO
   4.1 Assign email (X_i) to the cluster X_k where, similarity $(\mathrm{D}(X_{k,i}))$ is minimum.// Cluster formation
End

Fig. 1. Algorithm for Semantic Non-parametric K-Means++ Clustering

Step 4: Topic Detection and Folder Creation
Next, the folders are created by topic detection. The subject term with the highest R_{tf} in the whole cluster, C_k; is considered as a folder name.

3.2 APEG Model – An Automatic Sub-folder Creation

APEG is a process for creating the sub-folders. Sub-folder creation is based on email sender ID and contains the emails from that specific person in the respective folder. The whole process of APEG model is divided into two steps:

Step 1: Once the folders are created from the AEG model they serve as input for APEG model. So, firstly email ID of the sender is extracted from email message.

Step 2: Then some comparisons are made as follows:

a. If a sub-folder exists with the sender information, then that respective email message is moved to that sub-folder.

b. Else a new sub-folder is created with the name of the sender and email is then moved into that folder.

3.3 Indexing

Lastly, create index, which is a view of the hierarchical folder with links to emails and contains summary annotation of each folder. The output of APEG serves as input for indexing. Here, Apriori algorithm [10] is applied to the folder data to extract important terms which helps identify the content of folders. The whole process of indexing is divided in two steps (repeat following steps for all folders created):

Step 1: Feature terms of subject and content of each email from the folder are extracted using associated term frequency explained in section 3.1.

Step 2: Apply Apriori algorithm to the feature terms to extract the terms that are important for summary.

Step 3: Index/View is created as HTML web page which is a hierarchical representation of folders from AEG model, sub-folders from APEG model and containing link to each individual email. Additionally, summary of folder is contained.

4 Application Example for AEMS Module on Sample Data

Example 1: Given a user, *u* email inbox with say 5 emails from 2 senders, sender-1 and sender-2. Create topic folders F, sub-folders of sender (SF) and index *i* containing links to those F and SF (small size of file chosen only for illustration purposes). Consider thresholds $T_s = 30\%$, $T_b = 50\%$ and $\beta = 0.2$.

Solution 1: Five emails are input to the AEG model of AEMS system.

Step1: The subject and content of the email are extracted and all special characters, punctuation and stop words are removed, according to the section 3.1(Pre-processing) and these simple emails with 2-term subjects and up to 3-term contents are as follows:

Email X1 (sender1) – Subject: {T1, T2} and Content: {T1, T3, T5}
Email X2 (sender2) – Subject: {T1, T3} and Content: {T1, T4, T3}
Email X3 (sender1) – Subject: {T5, T2} and Content: {T5, T4}
Email X4 (sender2) – Subject: {T6, T1} and Content: {T6, T7, T2}
Email X5 (sender1) – Subject: {T4, T3} and Content: {T7, T3, T4}

Step 2: we need to find R_{tf} (document frequency of term) for each term and select only those terms with R_{tf} greater or equal to threshold of $T_s = 30\%$, $T_b = 50\%$, according to the section 3.1. The R_{tf} for subject and content are given below respectively.

R_{tf}(s) = {(T1, 3) (T2, 2) (T3, 2) (T4, 1) (T5, 1) (T6, 1) (T7, 0)}.
R_{tf}(b) = {(T1, 2) (T2, 1) (T3, 4) (T4, 3) (T5, 2) (T6, 1) (T7, 1)}.

Term {T1, T2, T3, and T4} is taken as feature term because there R_{tf} are greater or equal to the pre-defined threshold. Therefore the email vector by the selected feature terms will be: Email X1 with {T1, T2, T3}; Email X2 with {T1, T3, T4}; Email X3 with {T2, T4}; Email X4 with {T1, T2} and Email X5 with {T3, T4}

Step 3: Now we will cluster the email with Semantic Non-parametric K-Means++ clustering algorithm according to section 3.1. For this we need to follow steps below:

1. Find email containing the maximum weight ($\max(X_w)$). Here, $\max(X_w) = 3$ which is the email weight for X1, X2 and X3, obtained by calculating the total number of feature terms in email vector. Thus, first initial cluster center will be X1.
2. Calculate the similarity $D(X_{1,j})$ between pairs of emails X_1 and X_j to choose the next cluster centers (we chose STS [8] to find the similarity between emails). To compute the similarity between two emails X_1 and X_j, we need to find the common terms in the two emails and place in the set C and delete theses common terms from both emails as in $X_1 = X_1$ – set C and $X_j = X_j$ – set C. Then, calculate the string similarity between pairs of terms in X_1 and X_j using the average of 3 common similarity measures. Next, compute semantic similarity of pairs of terms of X_1 and X_j using SOC-PMI [12] (uses point-wise mutual information to sort lists of important neighbor words of the two target words. Then, consider the words which are common in both lists and aggregate their PMI values (from the opposite list) to calculate the relative semantic similarity) before computing the joint similarity matrix (M) of the two matrices for string and semantic similarities of the emails. Now, extract the maximum value from M and delete corresponding row and column (repeat till M become empty) and store the summation of extracted values in variable (say, S). Lastly, similarity score $D(X_{1,j})$ is calculated using equation 8

$$D(X_{1,j}) = ((S + \delta) * (m + n))/2mn \quad (8)$$

Here, m and n are the total number of terms in email X_1 and X_j respectively and δ is the total number of terms in set C.

The next center is chosen if its $D(X_{1,j}) \leq \beta$ and similarity proportional to $D(X_{1,j})^2$. Here, β is the pre-defined threshold value.

3. Again calculate the similarity of each email with each cluster center and assign that to the cluster where, the similarity is maximum. Suppose, the clusters formed are: C1 – {X1, X2, X4} and C2 – {X3, X5}.
4. Using these clusters, two folders are created and choose subject term as folder name, which have maximum R_{tf}. Therefore two folders created are: Folder T3 containing email X1, X2 and X4, and Folder T4 containing email X3 and X5.

Step 4: These two folders will be the input of the APEG model and APEG will create sub-folders according to the senders' email ID. Here, folder T3 will contain two sub-folders from sender-1 and sender-2, where sub-folder from sender-1 will contain email X1 and sub-folder from sender-2 will contain email X2 and X4. Similarity, for folder T4 will contain one sub-folder from sender-1 containing email X3 and X5.

Step 5: Finally, one index file will be created with links to the folders and sub-folders according to section 3.3 and also, contain email summary is created by applying Apriori algorithm.

5 Experimental Results

The AEMS module is implemented in Java and Eclipse is used as a development IDE. The hardware configuration to run the experiments is 3GB RAM, intel core i3 CPU, 2.34 GHz and 32-bit windows-7 operating system. To test our approach, we used publically available 20-Newsgroup collections [10]. It is a collection of 20,000 email messages, divided into 20 different newsgroups. The 20-Newsgroup data comes in, one file per email message containing email logs.

5.1 Evaluation Criteria

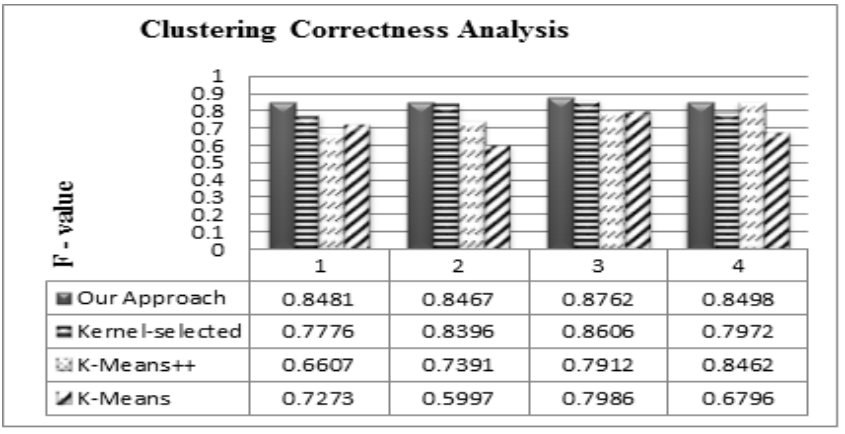
We used the F-value measure to evaluate the clustering quality and its formula is defined in equation 8:

$$F - Value, F(i,j) = 2 * P(i,j) * R(i,j)/P(i,j) + R(i,j) \quad (9)$$

where, $Precision, P(i,j) = n_{ij}/n_j$, and $Recall, R(i,j) = n_{ij}/n_i$
n_i is the number of clusters which human has labeled, n_j is the number of emails with clustering algorithms, and n_{ij} is the number of emails clustered correctly.

5.2 Study on Cluster Performance

We compared our clustering approach with standard K-means [9], K-Means++ [11] and Kernel-selected clustering [5], to show its efficiency of cluster correctness.



	1	2	3	4
Our Approach	0.8481	0.8467	0.8762	0.8498
Kernel-selected	0.7776	0.8396	0.8606	0.7972
K-Means++	0.6607	0.7391	0.7912	0.8462
K-Means	0.7273	0.5997	0.7986	0.6796

Fig. 2. F-value comparison of clustering Algorithm

We choose four folders from the 20-NewsGroup data set. These four folders consist of 1000 email messages each and results are evaluated terms of F-Value. Now, when experimenting data with Kernel-selected email clustering method and our proposed clustering algorithm, we have taken β = 0.5. We can observe from Fig. 2 that our approach performs better than the standard K-Means, K-Means++ and Kernel-selected clustering approach. Since the average of the F-Value when threshold is

taken as zero comes out to be 0.8552 for our clustering approaches whereas for Kernel-selected clustering method comes to be 0.8187.

6 Conclusions and Future Work

This paper proposed an Automatic Email Management System (AEMS) which clusters emails into meaningful groups and extract important feature words for identification of each folder. For AEMS, we proposed a novel feature selection based clustering approach. Future work could be that AEMS module, do not handle the processing of incoming messages; therefore, a method can be developed to immediately process incoming messages using classification methods. Additionally, some recommendation system can be built based on the emails logs for deletion of unused email.

References

1. Radicati, S., Hoang, Q.: Email statistics report, 2012-2016. The Radicati Group, Inc., London (2012)
2. Xiang, Y.: Managing Email Overload with an Automatic Nonparametric Clustering Approach. The Journal of Supercomputing 48(3), 227–242 (2009)
3. Schuff, D., Turetken, O., D'Arcy, J.: A multi-attribute, multi-weight clustering approach to managing "e-mail overload". Decision Support Systems, 1350–1365 (2006)
4. Cselle, G., Albrecht, K., Wattenhofer, R.: BuzzTrack: topic detection and tracking in email. In: Proceedings of the 12th International Conference on Intelligent User Interfaces (IUI 2007), New York, NY, USA (2007)
5. Yang, H., Luo, J., Yin, M., Liu, Y.: Automatically Detecting Personal Topics by Clustering Emails. In: 2010 Second International Workshop on IEEE Education Technology and Computer Science (ETCS), pp. 91–94 (2010)
6. Muresan, S., Tzoukermann, E., Klavans, J.L.: Combining linguistic and machine learning techniques for email summarization. In: Proceedings of the 2001 Workshop on Computational Natural Language Learning (2001)
7. Carenini, G., Ng, R.T., Zhou, X.: Summarizing email conversations with clue words. In: Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada (2007)
8. Islam, A., Inkpen, D.: Semantic text similarity using corpus-based word similarity and string similarity. ACM Transactions on Knowledge Discovery from Data (TKDD) 2, 10 (2008)
9. Han, J., Kamber, M.: Data mining: concepts and techniques. Morgan Kaufmann (2006)
10. Lang, K.: Newsweeder: Learning to filter netnews. In: Proceedings of the Twelfth International Conference on Machine Learning (1995)
11. Arthur, D., Vassilvitskii, S.: k-means++: the advantages of careful seeding. In: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1027–1035. ACM (2007)
12. Islam, A., Inkpen, D.: Second order co-occurrence PMI for determining the semantic similarity of words. In: Proceedings of the International Conference on Language Resources and Evaluation, Genoa, Italy, pp. 1033–1038 (2006)

A Data Mining-Based Wind Power Forecasting Method: Results for Wind Power Plants in Turkey

Mehmet Barış Özkan[1], Dilek Küçük[1], Erman Terciyanlı[1], Serkan Buhan[1], Turan Demirci[1], and Pinar Karagoz[2]

[1] TUBİTAK MAM Energy Institute, Ankara, Turkey
mehmet.ozkan@tubitak.gov.tr
[2] Middle East Technical University, 06531 Ankara, Turkey
karagoz@ceng.metu.edu.tr

Abstract. With the huge technological and industrial developments in recent years, the electricity demand of all countries has been increasing day by day. In order to supply the electricity needs, countries have been looking for ways of benefitting from their renewable energy sources efficiently and wind energy is an important and ubiquitous renewable energy source. However, due to wind's discontinuity and unstable characteristics, a reliable wind forecasting system is crucial not only for transmission system operators but also wind power plant (WPP) owners. This paper presents a reliable forecasting method based on data mining approaches. The method uses numerical weather predictions and past power measurements of the WPPs as input and it produces hourly short-term wind power forecasts for the WPPs for a time period of 48 hours. The method has been tested in the Wind Power Monitoring and Forecast Center (RİTM) project of Turkey for a duration of six months for 14 WPPs. The proposed model achieves better accuracy performance rates than those of the other well-known forecasting models for seven of WPPs selected for the testing procedure by the General Directorate of Renewable Energy in Turkey.

Keywords: wind power, clustering, numerical weather predictions, wind power forecasting, wind power monitoring.

1 Introduction

Excessive usage of non-renewable energy sources has become one of the main reasons of global warming problem in recent years. In addition to their harmful effects to the environment, these sources are envisioned to fall short for meeting the increasing energy demand all over the world as they are not sustainable. Due to these reasons, renewable energy sources have gained considerable attention in recent years. Wind is the one of most significant and ubiquitous energy sources

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 268–276, 2013.

among all renewable energy types [1]. Since it is a highly reliable, ubiquitous, and clean energy source, effective utilization of wind energy can help meet the energy demand in the world and can reduce the negative environmental effects of fossil-fueled power plants. However, unlike other renewable energy sources, wind has a volatile and variable characteristics and in order to control this unstable energy, a reliable forecast system is highly required for transmission system operators (TSOs) to be used for power grid administration and operation. Additionally, wind power plant (WPP) owners have to declare their day ahead production forecasts to electricity market operator and they profit according to the accuracy of their day ahead forecasts [3]. Hence, a reliable forecasting model is also crucial for WPP owners.

Due to importance of an accurate wind power forecast system, several forecast methods have been developed in the last years. All of these methods have different advantages and disadvantages according to wind characteristic of the WPP region. These methods can be classified in three groups as physical, statistical, and hybrid (combination of physical and statistical) approaches [4]. All of these methods use numerical weather predictions (NWPs) as initial input in order to generate short term forecasts (hourly forecasts for up to 48 hours) [3,4].

Physical models are based on the physical conditions in the WPP area. They use topological information of the WPP area, physical characteristics such as roughness and existing obstacles, and the turbine characteristics, especially the hub height of the turbines as inputs in addition to NWPs [5,6,7]. The WPP area with turbines are modelled in 3D space by using some modelling software such as WindSim and WAsP [7]. Basically, the aim of the model is to estimate the wind speed at turbine's hub height by using initial input data, more accurately. Mainly, methodologies like Computational Fluid Dynamics (CFD) are used to estimate local wind speed in turbine area. As final step, model output statistics (MOS) methods are applied to local wind speed to reduce the error rate [6].

The statistical models mainly aim to construct a mathematical model using historical power generation and NWPs. Linear regression, artificial neural networks (ANN), and support vector machines (SVM) are the three main methods in this category [8]. All of these methods have a training stage in order to learn the relationship between NWPs and WPPs past power generation values. [8,9]. These models update themselves dynamically with new incoming data. In forecast systems, statistical methods are employed more frequently, but, hybrid approaches are optimal since they also take local physical characteristics into account.

In this paper, a new statistical method for wind power forecasting is presented, which is based on data mining techniques. The proposed method is based on clustering the NWPs and then applying linear regression to the resulting clusters independently. It is being used on 14 WPPs that have been monitored within the scope of the Wind Power Monitoring and Forecast Center (RİTM) project [2] of Turkey and it is tested on 7 WPPs that are selected by the General Directorate of Renewable Energy, being the client of the (RİTM) project. The accuracy of the proposed method is compared to a well-known physical model and two other

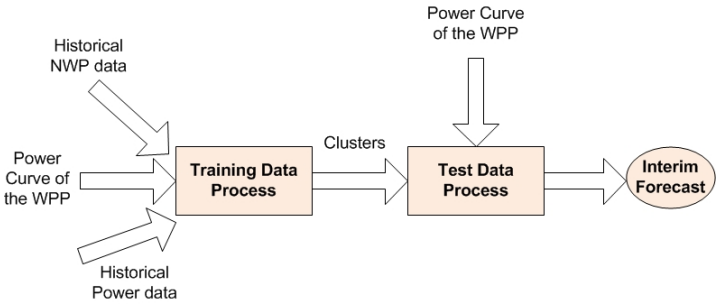


Fig. 1. The Overall Forecast Process

data mining techniques. The total installed capacities of these WPPs is 704.9 MW while the total installed capacity of all WPPs in operation in the country is 2202.4 MW, hence these 14 WPPs correspond to about 32% of all WPPs in the country in terms of installed capacity [12].

The rest of the paper is organized as follows: In Section 2, the details of the proposed wind power forecasting model are provided, Section 3 presents the evaluation and comparison results of the proposed model and finally Section 4 concludes the paper with further research directions.

2 The Proposed Wind Power Forecasting Model

The proposed method is based on combining dynamic clustering with linear regression. As the first step, clustering is applied on NWPs in a dynamic way. Within the scope of the RİTM project, NWPs are taken from three sources, namely, the outputs of executing the Weather Research and Forecasting (WRF) [13] meteorological model with two distinct input types: forecasts from Global Forecast System (GFS) [14] and European Centre for Medium-Range Weather Forecasts (ECMWF) [15], and the remaining NWP source is the output of the ALADIN [16,17] meteorological model obtained from Turkish State Meteorological Service . Dynamic clustering operation is applied to these sources [18,19]. In the next step, determined clusters are used in test phase in order to produce short term forecasts of WPPs. This forecast process executed on each of the aforementioned data sources is shown in Figure 1 where a final forecast for each distinct data source is obtained. As the final step, these forecasts are combined into a final forecast in combination phase. In the rest of this section, each of these steps are explained in detail.

2.1 Training Phase

The main aim of the training phase is to determine clusters that will be used during the testing phase. The inputs of the training phase are hourly historical NWPs corresponding to the WPP, hourly historical power production data of the

WPP, and the power curve of the WPP. The power curve of a WPP is a 360x251 matrix that associates wind direction and wind speed to an estimated wind power. Within the scope of the (RİTM) project, these curves are constructed using the physical modelling tools for each WPP. During the training phase, firstly, the 10x10 grid area of each WPP area is scanned where the WPP reference coordinate constitutes the center of the area.

The NWPs provide u and v components of wind to wind speed (s) and wind direction (d) according to Formula (1) and (2). For each WPP, for all of the 100 grid points considered, the best correlation coefficient for wind speed values are determined. These coefficient values are obtained by using linear regression method on the power estimations obtained by passing the wind speed and direction values through the power curve according to the average normalized mean absolute error (NMAE) rates calculated according to the Formula (3). In this formula, x_i is the real power, y_i is the estimation at i^{th} hour, C is the installed capacity of the WPP and N is the total number of hours processed in the training phase. Next, the grid point that has the minimum NMAE rate is determined as the best grid point among these 100 grid points.

$$s = \sqrt{u^2 + v^2} \tag{1}$$

$$d = (\arctan(u/v) \times 180)/\Pi + 180 \tag{2}$$

$$NMAE = \frac{\sum_{i=1}^{N} \frac{|x_i - y_i|}{C} \times 100}{N} \tag{3}$$

$$M = \begin{pmatrix} u_1 & v_1 & p_1 \\ u_2 & v_2 & p_2 \\ u_3 & v_3 & p_3 \\ . & . & . \\ . & . & . \\ u_N & v_N & p_N \end{pmatrix} \tag{4}$$

$$C = M \times M^{\top} \tag{5}$$

$$X = M \times E \tag{6}$$

After determining the best grid point, an Nx3 matrix is constructed including the u, v components of the wind and pressure (p) parameter available in the NWPs corresponding to this grid point. The structure of this matrix, M, is given in Formula (4), where N is the total number of hours processed in the training phase as specified before and u, $v$and p values are standardized and normalized versions of the corresponding values in the NWPs. Next, a covariance matrix, C, is calculated from M using Formula (5) and the most significant eigenvector of this matrix is extracted to calculate another Nx1 dimensional matrix, X, is calculated through Formula (6) where E is the most significant eigenvector

matrix with a dimension of 3x1. Hence, at the end of this process, the effects of the principal components *u, v* and *p* for each hour are obtained independently.

K-means clustering algorithm is applied on the X matrix to determine the clusters [11]. While deciding the number of the clusters, we consider the validity ratio described in [11].Firstly, for a determined historical test data, NMAE rates of the interim forecast for each cluster number between 2 to 15 were calculated separately for each WPP. In this test data, for each NWP source, the NMAE rates were observed to close to each other on average for the cluster numbers 3 to 7 and these error rates were lower in comparison to other number of clusters. However, average NMAE rates were too high when the number of clusters was 2 or when it is above 7 in the same test period. Then, for every training day, for each number of clusters from 3 to 7, their validity ratios [1] described in [11] are calculated and among all that has the highest validity ratio is selected as the number of clusters that the data will be partitioned in the test period. Therefore, the number of clusters for each WPP is dynamically determined with the constraint of being between 3 and 7, and their centroids are saved. After determining the clusters, each value in the X matrix is associated with a cluster according to the proximity of the values in the matrix to the clusters centroid points in terms of Euclidean distance. Next, for each cluster, the best representative grid point among the initial 100 grid points together with their correlation coefficients are determined using linear regression and considering only the hours corresponding to each cluster. The coordinates and coefficients of these representative grid points for each cluster are saved for later use during the test phase.

2.2 Test Phase

During the test phase, the upcoming 48-hour NWP data corresponding to the best grid point determined in the training phase are used to form an 48x3 M matrix. The same procedures utilized during the training phase are applied (with the previously calculated eigenvector) on this matrix to construct an X matrix with dimension 48x1. Using the previously saved centroids of the clusters, each hour in this matrix is assigned to an appropriate cluster label using the Euclidean distance. As the final stage of this phase, upcoming 48-hour NWPs for the representative grid points of each cluster are retrieved and the wind speed and direction estimates for each hour are obtained by using these NWPs and correlation coefficients of the corresponding cluster's representative grid points. Finally, these values are passed through WPP power curve and estimated power values for 48 hours are obtained.

2.3 Combination Phase

The process described in Figure 1 is conducted for each of NWP sources (WRF execution outputs on GFS and ECMWF global predictions and ALADIN outputs as obtained from Turkish State Meteorological Service) and in the end,

[1] validity=intra/inter where *intra* is the total distances of the each point to its cluster centroid point and *inter* is the minimum distance between cluster centers

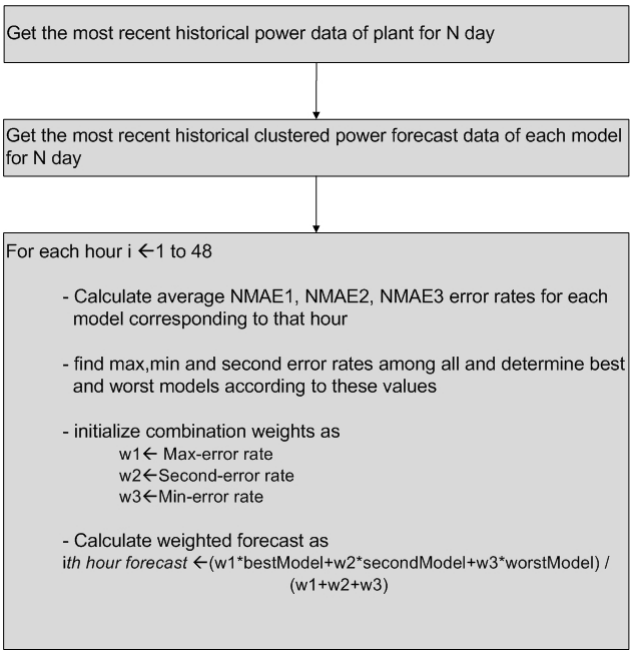


Fig. 2. Forecast Combination Process

three 48-hour forecast sets are obtained. These forecasts are combined into one ultimate forecast with a combination process as described in Figure 2. Corresponding hourly forecasts are combined by assigning different weights on the basis of NMAE rates. Using these recent error rates, for each hour, the best performing forecast is multiplied with the error rate of the worst forecast, the worst performing one is multiplied with the error rate of the best performing forecast, and the remaining forecast is multiplied by its own error rate. Hence, the individual forecasts are scaled according to their accuracies and the ultimate combined forecast is obtained as the weighted average of individual forecasts. This combination approach reduces the overall error rate of the WPPs in final prediction. For example, assume that for a WPP, each NWP's interim forecast NMAE error rates are 13.2 %, 12.8 % and 11.4 %, respectively on the average of six months test period. However, the NMAE rate of the combined forecast is 9.14 % for the same WPP in the same test period and the most of the improvements on the model come from this stage. Although, combining the forecasts with these determined weight reduces the error rate on the overall of this test period, it forms a more smoothed graphics if the interim forecasts are too different from each other and this situation causes to fail in determination of the sudden ups and downs in the power forecast.

Table 1. Evaluation Results (In terms of NMAE %)

WPP	ANN	SVM	Physical Model	New Model
WPP A	12.63 %	12.85 %	13.80 %	11.18 %
WPP B	11.13 %	11.71 %	12.37 %	9.14 %
WPP C	13.10 %	13.29 %	15.30 %	11.75 %
WPP D	13.78 %	14.83 %	16.15 %	12.81 %
WPP E	10.59 %	11.24 %	14.49 %	8.9 %
WPP F	18.53 %	20.23 %	14.26 %	12.04 %
WPP G	10.25 %	10.44 %	13.65 %	8.73 %

3 Experimental Results

This section presents the performance comparison between well-known statistical methods (ANN and SVM), a physical model [3], and the proposed statistical model. The ANN model is used as regression type model for power generation prediction of each WPP. The past wind speed and wind direction data of NWPs are used as inputs to train the network while the past wind power generation data is used as output of the ANN model. Then this trained model is used for online wind power generation forecasts by using NWP wind speed and wind direction values in the test data. A well known and fast library for ANN, which is Fast Artificial Neural Network Library (FANN), is used for this part of forecasts [20]. Similarly, the second statistical model SVM is used as regression type model which can be called Support Vector Regression (SVR) for power generation forecasts. As in ANN, SVM model uses past wind speed and wind direction data of NWPs as inputs and generates past wind power model for the training part. Next, trained SVM model produces the wind power forecast of the WPP by using the wind speed and wind direction test data. In order to implement the SVM model, Library for Support Vector Machines (LIBSVM) is used [21]. Each model has different characteristics, therefore, in some of the WPPs, the differences between the error rates increase due to WPPs physical characteristics. Evaluation results are calculated in terms of NMAE rates for seven WPPs. The average NMAE rates are calculated for a six month period for each model independently.

The evaluation results presented in Table 1 demonstrate that the proposed model achieves the best performance rates in all of the seven WPPs during this six month test period. The physical model has the poorest performance in all of the WPPs except WPP F. In this WPP, ANN and SVM achieve considerably lower performance rates in comparison to other WPPs. This WPP has a different production characteristic than the other WPPs, which leads to the poor performance of learning for ANN and SVM during this period. However, clustering the NWPs seems to alleviate this problem and hence the proposed model dramatically improves the estimations for this WPP. The proposed model has the lowest error rates for WPP E and WPP G and these WPPs are the two largest

WPPs among all 14 WPPs in terms of their installed capacities. The proposed model results in its highest error rate for WPP D, which has the lowest installed capacity among all other WPPs.

The total NMAE of the proposed model for all 14 WPPs monitored (with a total installed capacity of 704.9 MW) is %5.09. The actual production-estimation values for Turkey are publicly available at RİTM project Web site in a time-series graphic format (http://www.ritm.gov.tr/) .

The evaluation results demonstrate that the performance of the proposed hybrid statistical model is promising and performs better than the other approaches used for comparison. The results also show that in the other models, there are serious differences between error rates of the WPPs due to local weather characteristics while the performance rates of the proposed model are consistent for all WPPs and hence there is strong evidence that the model is applicable to all WPPs in different regions of Turkey.

4 Conclusion

Wind energy is an important type of renewable energy and producing accurate forecasts for WPPs is crucial to several involved parties including TSOs and WPP owners. In this paper, a new data-mining based statistical wind power forecast model is proposed and its evaluation results against other well-known models are presented. The proposed model has been applied on 14 WPPs monitored within the scope of RİTM project for six months. The performance results obtained from the proposed model and the other forecasting models on seven WPPs validate that it has several advantages over the other models for WPPs in Turkey in terms of accuracy and consistency.

For the future work, the effects of other clustering methodologies other than k-means may be investigated. Although at the end of six months, the proposed model produces lower error rates on average, in some days of the test period, the other models achieve better performance rates compared to new model. Therefore, a good combination process for all physical, ANN, SVM and the proposed statistical forecast model may also improve the overall performance of the forecasts for WPPs in Turkey. In addition, if the three interim forecasts are too different from each other for a specific day, then the combined forecast become a more smoothed series and it does not determine the ramps in the forecast properly. Therfore, in order to specify the ramps in the WPP another approach also must be included to current combination algorithm.

References

1. Butterfield, S., Sheng, S., Oyague, F.: Wind Energys New Role in Supplying the Worlds Energy: What Role will Structural Health Monitoring Play? In: Proceedings of the 7th International Workshop on Structural Health (2009)
2. RİTM Project Web Site, `http://www.ritm.gov.tr/root/index_eng.php`

3. Monteiro, C., Bessa, R., Miranda, V., Botterud, A., Wang, J., Conzelmann, G.: Wind Power Forecasting: State-of-the-Art 2009. Argonne National Laboratory Technical Report (2009)
4. Botterud, A., Wang, J., Miranda, V., Bessa, R.J.: WindPower Forecasting in U.S. Electricity Markets. ElectricityJournal 23(3), 71–82 (2010)
5. Al-Deen, S., Yamaguchi, A., Ishihara, T., Bessa, R.J.: A Physical Approach to Wind Speed Prediction for Wind Energy Forecasting. Journal of Wind Engineering, 349–352 (2006)
6. Russell, A.: Computational Fluid Dynamics Modeling of Atmospheric Flow Applied to Wind Energy Research, Masters Thesis, Boise State University (2009)
7. Berge, E., et al.: Wind in complex terrain.A comparison of WAsP and two CFD-models. In: Procedings of th 2nd European Wind Energy Conference (2006)
8. Milligan, M., Schwartz, M., Wan, Y.: Statistical Wind Power Forecasting for U.S Wind Farms. In: Proceedings of the 17th Conference on Probability and Statistics in the Atmospheric Sciences (2004)
9. Gomes, P., Castro, R.: Wind Speed and Wind Power Forecasting using Statistical Models: AutoRegressive Moving Average (ARMA) and Artificial Neural Networks (ANN). International Journal of Sustainable Energy Development 1(1/2) (2012)
10. Kusiak, A., Zheng, H., Song, Z.: Wind Farm Power Prediction: A Data Mining Approach. Wind Energy 2(3), 275–293 (2009)
11. Siddheswar, R., Rose, H.T.: Determination of Number of Clusters in K-Means Clustering and Application in Colour Image Segmentation. In: Proceedings of the 4th International Conference on Advances in Pattern Recognition and Digital Techniques (1999)
12. Terciyanlı, E., Demirci, T., Küçük, D., Buhan, S., Özkan, M.B., et al.: The Architecture of a Large-Scale Wind Power Monitoring and Forecast System. In: Proceedings of the International Conference on Power Engineering, Energy and Electrical Drives (2013)
13. Weather Research and Forecasting (WRF) Model Web Site, `http://www.wrf-model.org/index.php`
14. Global Forecast System (GFS) Web Site, `http://www.emc.ncep.noaa.gov/index.php?branch=GFS`
15. European Centre for Medium-Range Weather Forecasts (ECMWF) Web Site, `http://www.ecmwf.int`
16. ALADIN Model Web Site, `http://www.cnrm.meteo.fr/aladin`
17. Huth, R., Mldek, R., Metelka, L., Sedlk, P., Huthov, Z., Kliegrov, S., Kysel, J., Pokorn, L., Janouek, M., Halenka, T.: On the integrability of limited-area numerical weather prediction model ALADIN over extended time periods. Studia Geoph. Geod. 47, 863–873 (2003)
18. Hamill, T.M., Whitaker, J.S., Fiorino, M., Benjamin, S.G.: Global Ensemble Predictions of 2009s Tropical Cyclones Initialized with an Ensemble Kalman Filter. Monthly Weather Review 139(2), 668–688 (2011)
19. Lange, M.: Analysis of the uncertainty of wind power predictions, PhD Dissertation, Carl von Ossietzky Oldenburg University (2003)
20. Fast Artificial Neural Network Library (FANN) Web Site, `http://leenissen.dk/fann/wp/`
21. Library for Support Vector Machines (LIBSVM) Web Site cjlin/libsvm/, `http://www.csie.ntu.edu.tw/`

Disease Occurrence Prediction Based on Spatio-temporal Characterization – A Mesoscale Study for Knowledge and Pattern Discovery

Vipul Raheja and K.S. Rajan

Lab for Spatial Informatics
International Institute of Information Technology Hyderabad,
Hyderabad, India
vipul.raheja@research.iiit.ac.in, rajan@iiit.ac.in

Abstract. Many statistical modeling and simulation-based methods have been proposed to detect, simulate and predict disease in space and time. However, these models either make use of unprecedented amount of domain-related parameters, or suffer from issues of global-generality and local over fitting. In this work, a methodology is proposed for detection, simulation and prediction of disease in a region based on an aggregated idea of spatio-temporal zoning. Specifically, a novel method is developed to model meso-scale processes by capturing spatio-temporal correlation leveraging both potential dependencies among local coefficients, and the potential incongruities within global models for prediction of trends in the context of disease management. The method is able to infer causality, and simulate the spread given the initial sources derived primarily from the spatio-temporal history of the disease. Illustrative case study of Salmonellosis disease in USA is presented to demonstrate utility of the method. The prediction trends mimic the observed event data better than the standard methodology even though magnitudinal predictions need to be improved. It is evident that such a methodology will help prioritize decision-making process for better risk assessment and management including disease outbreak.

Keywords: Spatio-Temporal Prediction, Spatio-Temporal Characteristics, Zonal Aggregation, Disease Occurrence, Decision Support.

1 Introduction

Advancements in modern data acquisition techniques have facilitated the collection of large datasets of geo-tagged information. Therefore, the need for effective and efficient methods focusing on the development of theory, methodology, and practice to extract interesting and previously unknown, but potentially useful patterns from spatial and temporal data sets has recently emerged as a research priority [1].

A primary inferential objective in the analysis of disease incidence data is summarization and explanation of spatial and spatio-temporal patterns of disease (disease mapping); and also spatial smoothing and temporal prediction (forecasting) of disease

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 277–284, 2013.

risk. The field of spatial epidemiology has grown rapidly in the past two decades with the introduction of spatial and spatio-temporal hierarchical models [2]. Various statistical techniques for disease risk mapping [3] have become very popular in public health analysis. As discussed in [4], these methods enable the smoothing of ecological health indicators accounting for the geographical structure of the units under study. However, the implementation of these methods is not always easy or adequate for a quick response and in many instances need to adapted suitably to the problem at hand. This is due to both the subjective nature of the model/method choice as well as the parametric values which lead to locally fine-tuned models, limiting the widespread adoption of these techniques. Health experts are interested in understanding the consequent risks and build forecast models, either in the short or midterm as part of the disease management strategies. Currently, the latter are based on spatial, temporal and spatio-temporal co-occurrence of disease incidences, but in most of these cases the approaches consist of either a characteristic decoupling of spatial and temporal interdependencies [5,6], or an inherent dependency on various etiological and geographical factors.

Spatial autocorrelation statistics (Moran's *I*, Geary's *C* etc.) are global in the sense that they estimate the overall degree of spatial autocorrelation for a dataset, while ignoring spatial heterogeneity exhibited across geographic space including occurrence of natural break lines within these regions. On the other hand, Local spatial autocorrelation statistics provide estimates disaggregated to the level of the spatial analysis units, allowing assessment of the dependency relationships across space like Getis's *G* statistics, and localized versions of Moran's *I* and Geary's *C* statistics. Geographic analyses of natural phenomena may be separated into those that attempt generalizations to achieve 'global' insights, and those that attempt to explore and document local variations. In addition, the appropriateness of the scale of analysis [7,8] can further limit the understanding of the phenomena and limit the model design and applicability. Hence, in order to overcome the inherent assumptions of uniformity and to discern the intrinsic diversity in patterns, regional or meso-scale boundaries need to be identified in phenomena, which are not bounded by local restrictions like topography, but such natural breaks in their areas of prevalence are evident from the data itself. Attention in this paper is drawn towards capturing spatio-temporal correlation centered on the potential dependencies among the local coefficients and the potential incongruities within the global models. Hence, a new approach of meso-scale correlation analysis is proposed which leverages on natural artifacts which validate meaningful interpretation and search for spatial heterogeneities.

Recently, geographical, hydrological and climatic barriers have been recognized as being just as effective in delineating and controlling the spread of disease from an affected area - effectively isolating it within a zone within a given country [9]. Moreover, detection of the originating areas for variations in disease occurrence is a first step in the concept of zoning for health management. An approach that can tessellate the space and provide for a way to handle its temporal variations is needed, which the concept of 'zone' yields to.

1.1 Disease Prediction

Health experts are interested in early identification of an outbreak, understanding the risks and build forecast models, either in the short or mid-term as part of the disease management strategies. However, outbreaks are often well under way before public health authorities become aware of them, which is why spatio-temporal prediction becomes an essential application in health management. Numerous approaches to predict diseases in space and time have been proposed in literature. While temporal algorithms range from Scan Statistics [10,11], Regression, Generalized Linear Mixture Models, ARMA, Change detection and Wavelet based methods; spatial algorithms focus on Spatial Clustering, Geographic Analysis Machine and Spatial Scan Statistics [12] based methods. Many methods incorporating Spatio-Temporal interdependencies have also been developed focused on cluster and event detection in multivariate health data [13-16]. These have used the spatio-temporal characteristics of the data either singularly or combined with the prior domain knowledge to discover the underlying disease prevalence process to varying degrees.

2 Objectives

As part of this work, the spatio-temporal goals in disease analysis, prediction and management that will be addressed are – (i) Given spatially-explicit historical (time-series) data, can unique patterns representative of the underlying phenomena be extracted? (ii) Given that a disease has a reoccurrence or relapse at the same location, as a health emergency problem can one find the pattern of projected outcomes given the disease's early stage onset?

3 Methodology

The first part of the method utilizes the MiSTIC algorithm [17] to capture the inherent spatio-temporal interdependencies in the data and delineates the complete region into zones based on the temporal prevalence of observed values in their respective spatial locations; in its neighborhood; and spatial interactions. These derived zones are sub-regions of the study area, and are significant as they represent homogeneous entity in a collectively aggregated form, and also are representative of inherent correlations at a local scale. In this work, disease modeling and prediction has been explored for communicable (infectious) diseases from a broad perspective - due to external exposure, quantified by spatial proximity; or due to the inherent risk at a location, quantified by susceptibility. Non-communicable diseases – occupational or lifestyle diseases are not of interest for the proposed model.

3.1 MiSTIC: Focal Area Detection and Zone Delineation

Detection of Focal Areas. Focal Areas (polygons) are detected based on [17], for every time step over the entire region for the complete time period of study.

Spatio-temporal Analysis of Focal-Polygons and Respective Cores. A Core location (or Core) is defined as a set of areal units (polygons) which occur frequently (> minimum support) as the foci of a defined region [17]. This aggregation repeated over the entire study time period gives the final Cores. Raheja and Rajan [18] demonstrated that in case of disease spread, Cores in Contiguous Neighborhood (CC) behave better than Core locations in a defined Radius (CR) due to the localized nature of disease.

Delineation of Zones. In this step, the entire region is delineated into different zones corresponding to its unique representative focal areas. Zones with no areal units other than focal area itself are marked as outliers and not considered for analysis.

Data Post-processing – Spatio-Temporal Characterized Zone Delineation. Post-processing is done on the detected core locations for better understanding about the spatial distribution obtained. This largely depends upon the phenomenon being analyzed for the study – Physical or Non-Physical. The areas constituting a zone show a high degree of cohesion as compared to the inter-zone counterparts. On the other hand, the boundaries between zones are the visible representatives of natural break lines, derived from the data itself.

3.2 Spatio-temporal Prediction at Zonal Aggregation

From an epidemiological point of view, in order to quantify the spread and model the same for predictions, we classify the different risk factors involved in the spread of an epidemic into two broad types – (i) Factors dealing with local transmission within the zone largely characterized by its variability within the zone – physical and non-physical contact; (ii) Factors dealing with long-distance transmission within the zone –decay factor. These are reasonable assumptions because during a disease outbreak the real sources of infection are rarely identified when investigating them [13].

Time is introduced in the algorithm for two reasons – one, every areal unit is affected by the disease in different magnitudes at different time samples; and two, spatial spread of every outbreak is different at different points in time. Hence the parameters have to be time-dependent in order for the system to learn from the past patterns to mimic the evolution of spread and also to predict the future patterns based on local as well as aggregated patterns of disease rates.

(A) Spatial Influences on Disease Spread.

(a) **Contact Factors**.

(i) ***Common Boundary Shared (Physical contact).***

This transmission factor B_{ij} provides for vector movements between neighboring areal units *i* and *j*, both of which lie within the same zone. It is dependent on the spatial proximity between the two areal units in question,

$$B_{ij} = \frac{l_{ij}}{P_i}$$

Here, unit *i is* source of simulated outbreak, P_i is its perimeter, and l_{ij} is the shared boundary between itself and neighboring unit *j*.

(ii) Connectivity (Non-Physical contact).

The connectivity factor C_{ij} accounts for mobility flows that couple different subpopulations as a network of connections.

$$C_{ij} = \frac{r_{ij}}{R}$$

Here, r_{ij} represents the road network connections, and R represents total road network connections within the zone in consideration.

(b) **Decay Factor**.

The distance decay D_{ij} has been formulated to weigh the strength of correlation based on geographical proximity.

$$D_{ij} = \frac{M_i}{(1 + d_{ij})^2}$$

Here, M_i is the projected magnitude of disease rate in an area i (source areal unit/polygon or county), calculated using the ARIMA Model based on the temporal history of the county. Whereas d_{ij} is the Euclidean distance between the source and target areas i and j. The inverse squared relation $(1 + d_{ij})^2$ represents the spatial decay of disease effect from one county to another and is dependent on the distance between the two areal units.

(B) Disease Reoccurrence Patterns – Temporal Factors.

High disease occurrence implies that the areal unit in consideration is a disease focal area and has been quantified by its frequency at a particular location.

$$T_i = \frac{n_i}{N}$$

Here, n_i is the number of years when the county i is a disease foci and N is the total number of years of study.

(C) Spatio-temporal Prediction of Disease Occurrence

Once the disease is made to spread from seed locations into the neighborhood, it is propagated further into the zone based on the spatio-temporal parameters in its new local environment. This propagation keeps occurring till any significant change ceases to occur within the zone. The resulting magnitudes are the predicted outcomes.

4 Case Study: Salmonellosis in USA

4.1 Data Used

The Salmonellosis disease surveillance data with number of cases and rates per 100,000 of population, reported annually at county level is used in this study. The data exhibits heterogeneity in terms of economic, social and demographic settings.

The data for Florida State in USA was collected from FloridaCHARTS.com (Florida Department of Health) spanning 50 years (1961-2010).

4.2 Zoning: Boundaries Leading to Disease Area Delineation

The data was separated into training set and testing set, as stated in Figs 2 & 3. The set of valid focal areal units (polygons) were detected and corresponding zones were created and delineated for the training set. Fig. 1 depicts the corresponding zones, colored uniquely, mapped for Salmonellosis in Florida.

The inter-zone heterogeneities are confirmed by the presence of rivers along zonal boundaries, acting as natural barriers (Fig.1). Also, as can be seen from the figure, at inter-zone level, both road connectivity and road density is observed to be significantly less than the intra-zone counterparts. These observations reinforce the expectancy stated earlier in Section 3.1 about the intra-zone cohesion and inter-zone incongruence due to heterogeneous global variations and homogeneous local consistency in the underlying spatio-temporal process, expressed from the data itself, and further verified by domain-related input.

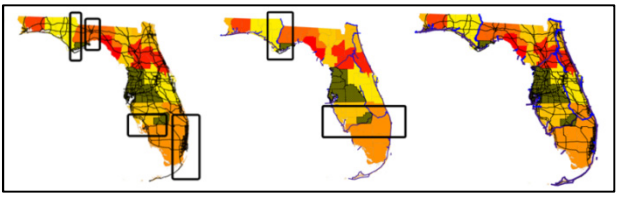


Fig. 1. Map showing [L-R] overlay of delineated Disease Zones in Florida, with Road Network; Major Inland Waterways; and Combined Overlays. Variations are highlighted by boxes.

4.3 Prediction

The proposed method aims to predict the locations most likely to trigger off an outbreak, as well as the consequences of that potential outbreak on rest of the areal units within the neighborhood zone. Disease spread in each county in the desired zone to be modeled was forecasted by first finding the best combination of outbreak sources, and then using them as seed points, simulating the transfer over space and time in the zone. The initial seed value based on ARIMA Time Series model was assigned to the source areas. This was done in multiple steps involving stationary characteristic checking, differentiation and correlogram/partial correlogram analysis. Most time series were modeled by the ARIMA (0,1,1) model, which represents the MA (Moving Average) version of the model. Once the best combination of source locations was found, disease was made to spread into its neighborhood based on its local spatio-temporal parameters as detailed in Section 3.2 until any significant change ceased to occur in the zone.

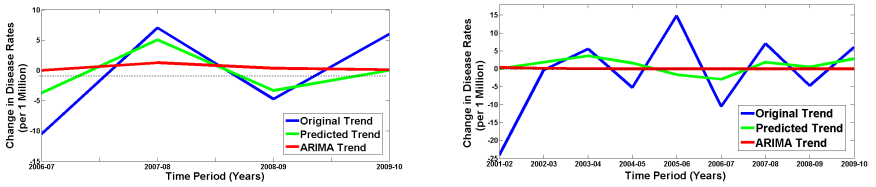


Fig. 2. Change in Disease Rates in Bay County Zone, FL: Observed, Predicted and ARIMA Trends 45/5 years Training/Testing and (b) 40/10 years Training/Testing

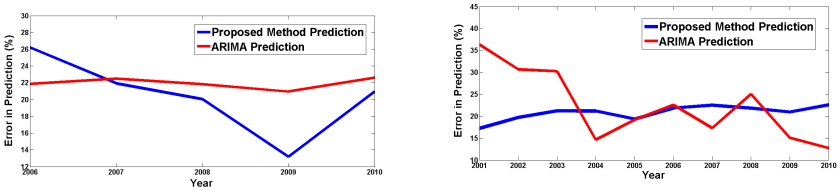


Fig. 3. Prediction Error comparison between ARIMA and Proposed Method, computed for Bay County Zone, FL (a) 45/5 years Training/Testing and (b) 40/10 years Training/Testing

5 Discussions and Conclusions

In this paper, we present a method that extracts the spatio-temporal characteristics intrinsically present in the data to describe a spatio-temporal dynamic process, in order to quantify and capture the otherwise indiscernible patterns in it. This allows for discovery of spatio-temporal process underlying such huge amounts of data. The meso-scale approach of using the zone has been demonstrated through the domain-related observations, which reinforces the understanding of this essential division. In this way, local correlations and global heterogeneities are both taken care of in an efficient manner. The trends predicted by the model, though Fig. 2 shows only one representative county, for all counties in Florida State are in concurrence with the observed event data in comparison to ARIMA's results. Though the magnitudinal errors (Fig. 3) are highly coupled to the duration of training and testing period, the proposed method results in better predictions than ARIMA when the testing period is short, implying that the seed values can bias the output due to a bias towards variability smoothening. Though the quality of the seed is an issue, overall the approach demonstrates that the method is not dependent on the data, and is robust in nature. It is independent and not reliant on domain related or process-related parameters in order to illustrate disease patterns in space and time. While, predicting highly accurate disease spread, especially magnitudes, still remains largely a complex problem in the public health research, integration of trend prediction into routine monitoring of health events can clearly emerge as an effective public health decision making tool, and the spatial heterogeneity captured here helps prioritize the response in case of outbreak situations.

Future work can range from improving prediction magnitudes to better choice of seeds by adopting other spatially, temporally or spatio-temporally elaborate methods. Also, by considering additional factors of disease spread dynamics, typically obtained

from physical, environmental forces and social networks etc., the modeling of population dynamics would be challenging in the spatially more finely-scaled settings and the emphasis on risk analysis alone would be less significant.

References

1. Mennis, J., Guo, D.: Spatial data mining and geographic knowledge discovery—An introduction. Computers, Environment and Urban Systems 33(6), 403–408 (2009)
2. Kottas, A., Duan, J.A., Gelfand, A.E.: Modeling Disease Incidence Data with Spatial and Spatio-Temporal Dirichlet Process Mixtures. Biometrical Journal 49(5), 1–14 (2007)
3. Martinez-Beneito, M.A., Lopez-Quilez, A., Botella-Rocamora, P.: Autoregressive spatio-temporal disease mapping. Stat Med. 27(15), 2874–2889 (2008)
4. Sahu, S.K., Mardia, K.V.: Recent Trends in Modeling Spatio-Temporal Data. Meeting of the Italian Statistical Society on Statistics and the Environment, 69–83 (2005)
5. De Cola, L.: Spatial Forecasting of Disease Risk and Uncertainty. Cartography and Geographic Information Science 29(4), 363–380 (2002)
6. Kelsall, J., Wakefield, J.: Modeling Spatial Variation in Disease Risk: A Geostatistical Approach. J. Am. Statist. Assoc. 97(459), 692–701 (2002)
7. Jetz, W., Rahbek, C., Lichstein, J.W.: Global Ecology and Biogeography Local and global approaches to spatial data analysis in ecology. Global Ecol. Biogeogr. 14, 97–98 (2005)
8. Paul Elhorst, J.: Specification and Estimation of Spatial Panel Data Models. International Regional Science Review 26(3), 244–268 (2003)
9. Surveillance and zoning for aquatic animal diseases. FAO Fisheries Technical Paper. Food and Agriculture Organization of the United Nations, Rome (2004)
10. Glaz, J., Balakrishnan, N.: Scan statistics and applications. Birkhauser, Boston (1999)
11. Hutwagner, L.C., Maloney, E.K., Bean, N.H., Slutsker, L., Martin, S.M.: Using laboratory-based surveillance data for prevention: an algorithm for detecting Salmonella outbreaks. Emerg. Infect. Dis. 3(3), 395–400 (1997)
12. Kuldorff, M.: A spatial scan statistic. Communications in Statistics: Theory and Methods 26(6), 1481–1496 (1997)
13. Gerbier, G., et al.: A point pattern model of the spread of foot-and-mouth disease. Preventive Veterinary Medicine 56(1), 33–49 (2002)
14. Maciejewski, R., et al.: Forecasting Hotspots—A Predictive Analytics Approach. IEEE Transactions on Visualization and Computer Graphics 17(4), 440–453 (2011)
15. Chainey, S., Tompson, L., Uhlig, S.: The utility of hotspot mapping for predicting spatial patterns of crime. Security Journal 21(1), 4–28 (2008)
16. Neill, D.B., et al.: Detection of emerging space-time clusters. In: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining. ACM (2005)
17. Sravanthi, K., Rajan, K.S.: Spatio-Temporal Mining of Core Regions: Study of Rainfall Patterns in Monsoonal India. In: 2010 IEEE International Conference on Data Mining Workshops, pp. 30–37 (2011)
18. Raheja, V., Rajan, K.S.: Extraction of Disease Occurrence Patterns using MiSTIC: Study of Salmonellosis in Florida. In: nternational Society for Disease Surveillance (ISDS) Annual Conference (2012)

Population Estimation Mining Using Satellite Imagery

Kwankamon Dittakan[1], Frans Coenen[1], Rob Christley[2], and Maya Wardeh[2]

[1] Department of Computer Science,
University of Liverpool, Liverpool, L69 3BX, United Kingdom
[2] Department of Epidemiology and Population Health,
University of Liverpool, Leahurst Campus,
Chester High Road, CH64 7TE Neston, Cheshire,United Kingdom
{dittakan,coenen,robc,maya.wardeh}@liverpool.ac.uk

Abstract. Many countries around the world regularly collect census data. This census data provides statistical information regarding populations to in turn support decision making processes. However, traditional approaches to the collation of censes data are both expensive and time consuming. The analysis of high resolution satellite imagery provides a useful alternative to collecting census data which is significantly cheaper than traditional methods, although less accurate. This paper describes a technique for mining satellite imagery, to extract census information, founded on the use of classification techniques coupled with a graph based representation of the relevant imagery. The fundamental idea is to build a classifier that can label households according to "family size" which can then be used to collect census data. To act as a focus for the work training data obtained from villages lying some 300km to the northwest of Addis Ababa in Ethiopia was used. The nature of each household in the segmented training data was captured using a tree-based representation. Each tree represented household had a "family size" class label associated with it. This data was then used to build a classifier that can be used to predict household sizes according to the nature of the tree-based structure.

Keywords: Satellite Image Analysis and Mining, Data Mining Applications, Population Estimation Mining.

1 Introduction

The work described in this paper is directed at the automated estimation of census information using data mining techniques applied to satellite imagery. The motivation for the work is that the collection of census data, when conducted in the traditional manner (using postal, email or interview approaches) is very resource intensive. This is especially the case in rural areas that lack sophisticated communication and transport infrastructure. The solution proposed in this paper is founded on the concept of using satellite imagery for population estimation. The idea is to use a small sample of satellite images of households, where the "family size" is known, to build a classifier that can then be used to predict household family sizes over a much wider area. The main issue to be addressed is how best to represent the household image data so that classification techniques can be applied. The solution presented in this paper is to first

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 285–296, 2013.

segment relevant satellite imagery so as to isolate individual households and represent individual household data using a quadtree based technique. The advantages offered by the proposed approach, in the context of census collection, are: (i) low cost, (ii) speed of collection and (iii) automated processing. The disadvantage is that it will not be as accurate as more traditional "on ground" census collection, however it is suggested that the advantages outweigh the disadvantages.

The proposed approach is more applicable with respect to rural areas than suburban and inner city areas. In this paper, the study area used as an exemplar application area is in the Ethiopia hinterland. More specifically training data obtained from two villages lying some 300 km to the northwest of Addis Ababa in Ethiopia was used, as shown in Figure 1 (the letters 'A' and 'B' indicate the village locations)[1].

The rest of this paper is organised as follows. In Section 2 some related work is briefly presented. Section 3 then provides a description of the proposed census mining framework. A brief overview of the proposed image segmentation process is presented in Section 4. In Section 5 details of the graph-based representation are presented, including a review of the proposed: quadtree decomposition, frequent subgraph mining and feature vector representation. The performance of the proposed census mining framework, using the Ethiopian test data, is then considered in Section 6. Finally, Section 7 provides a summary and some conclusions.

Fig. 1. Test site location

2 Previous Work

From the literature there have been a number of reports concerning automatic census collection founded on a variety of technologies. For example, in [3] voice recognition was used to automatically collect census data using telephone links. Another example can be found in [10] where census data was automatically recorded using PDAs. These two reported methods both demonstrated that time savings can be gained by at least partially automating the census gathering process.

[1] `http://maps.google.com`

Many methods for population estimation have been reported in the literature. These can be categorised as being founded on either: (i) areal interpolation or (ii) statistical modelling. The areal interpolation approach is typically used to identify areas of differing population densities, in other words to produce corse population density studies. The statistical modelling approach is typically used for identifying relationships between populations and other information sources such as Geographic Information System sources [12].

The approach advocated in this paper is founded on the use of satellite imagery. Satellite imagery has been used with respect to population estimation. For example, Google Earth satellite images have been used to estimate corse population densities at the city and village levels [7]. By identifying features such as dwelling units and residential areas satellite images have also been applied for the purpose of population estimation [1]. Further examples can be found in [2,8], where "night satellite" imagery was used to estimate population sizes according to the local densities of light sources.

In the context of the proposed population estimation mining the mechanism whereby satellite images are represented is important. There are many representation techniques available founded on image features such as: (i) colour, (ii) texture and (iii) structure. Colour histograms are wildly used to represent image content in terms of colour distribution. There are two major methods for histogram generation: the "binning" histogram and the clustering methods. The binning histogram method is used to generate a histogram by dividing the entire colour space into a number of bins and then, for each bin, recording the number of pixels that "fall into" that bin. Using the clustering method the colour space is first divided into a large number of bins and then a clustering algorithm is used to group them together [11]. The colour histogram representation has the advantages that it is easy to process and is invariant to translation and rotation of the image content [6].

Texture features can be used to describe a variety of surface characteristics of images [14]. There are three principle mechanisms that may be adopted to describe texture: statistical, structural and spectral. Statistical methods are related to the capture of image texture content using quantitative measures such as "smooth", "coarse" and "grainy". Structural methods are concerned with image texture, in terms of a set of structural primitives or elements (texels) and layout, that occur as repeating patterns. Finally spectral methods involved use the Fourier spectrum domain so that "high-energy narrow peaks" in the spectrum can be identified [5].

Structure features are used to describe the "geometry" of an image according to the relative position of elements that may be contained in an image, for example morphological and graph techniques. A well know structural image feature representation is the quadtree representation. This is then the fundamental representation used with respect to the work described in this paper.

3 Census Mining Framework

An overview of the proposed process for census mining is presented in this section. A schematic of the framework is shown in Figure 2. The framework comprised two phases (as represented by the rectangular boxes): (i) Preprocessing and (ii) Classification.

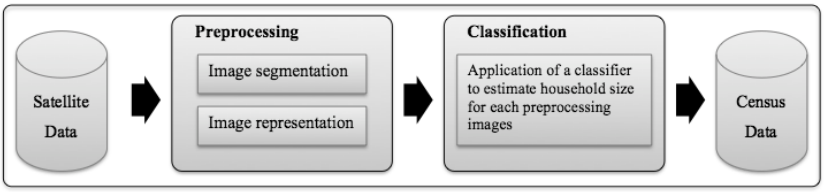


Fig. 2. Proposed census mining from satellite imagery framework

The first phase of the census mining framework is the preprocessing phase (the left rectangle in Figure 2) where the input data is prepared. The required preprocessing consists of two steps: image segmentation and image representation. The input is a satellite image of a given area covering (say) a number of villages. This image is then segmented in order to identify a set of individual households. This segmentation process was described in detail in [4]; however, for completeness, a brief overview of the process is presented in Section 4. Next the identified household pixel data is translated into a representation that allows for the application of a classifier. In this paper a novel graph-based representation technique is proposed, detail of which is presented in Section 5.

After the households have been segmented and appropriately represented the classification phase may be commenced. This is relatively straight forward once an appropriate classifier has been generated. There are many classifier generation techniques that may be adopted and some of these are considered with respect to the evaluation presented in Section 6.

4 Segmentation

This section presents a brief overview of the image segmentation process as applied to the input data. The image segmentation comprises three individual stages: (i) coarse segmentation, (ii) image enhancement and (iii) fine segmentation. The first stage is thus coarse segmentation whereby the input satellite imagery is roughly separated into a set of sub-images covering (typically) between one and four households each. Once the coarse segmentation process is completed, the next stage is image enhancement where a range of image enhancement processes are applied to the coarse segmented sub-images so as to facilitate the following fine segmentation of individual households. During the fine segmentation stage, the enhanced coarse segmented sub-images are segment further so as to isolate individual households so that we end up with one image per household. The image segmentation process was implement using the MATLAB (matrix laboratory) workbench.[2] Figure 3(a) and (b) show two fine segmented household images taken from test Site A and B respectively (see Figure 1).

The segmentation process is completed by translating the resulting RGB image data into a grayscale format ready for further processing followed by the application of a histogram equalisation process. Histogram equalisation is concerned with contrast adjustment using image histograms. Figure 4(a) shows the result when histogram equalisation is applied to the household image presented in Figure 3(a). For the purpose of

[2] http://www.mathworks.com

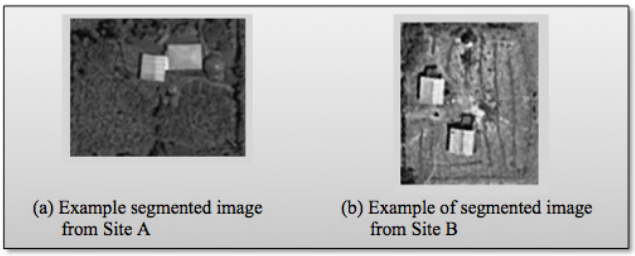


Fig. 3. Example of segmented household from Site A and Site B

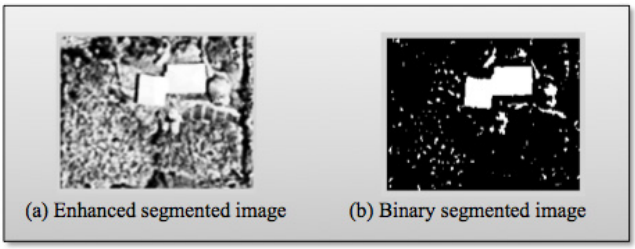


Fig. 4. Histogram equalisation and binary image transformation

the hierarchical quadtree decomposition (see below) a binary image transformation is then applied as shown in Figure 4(b).

5 Graph-Based Representation

Once a set of households has been fine segmented the next stage of the data preparation phase is to translate the segmented pixel data into a form suitable for the application of a classifier. The translation needs to be conducted in such a way that all salient information is retained while at the same time ensuring that the representation is concise enough to allow for effective further processing. The fundamental idea here is to adopt a graph based representation, more specifically a quadtree based representation (one per household). Quadtrees have been used extensively in the context of image processing (see for example [9]). However, the quadtree representation does not lend itself to ready incorporation with respect to classification algorithms. To do this we propose applying sub-graph mining to the quadtree data to identify frequently occurring patterns across the data that can be used as features in the context of a feature vector representation. The patterns of interest are thus frequently occurring sub graphs. An overview of the process is presented in Figure 5. The graph-based representation consists of four steps: (i) quadtree decomposition, (ii) tree construction, (iii) frequent subgraph mining and (iv) feature vector transformation.

The first step, the quadtree decomposition, commences by "cropping" each household image so that it is turned into a 128×128 pixel square image surrounding the main building comprising the household (this is automatically identifiable because it is

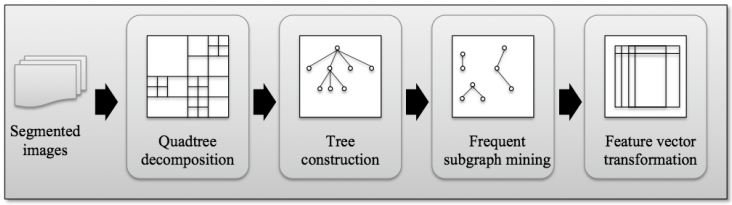


Fig. 5. Schematic illustration of the graph-based image representation processes

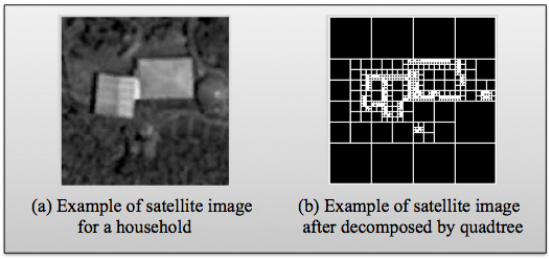


Fig. 6. The example of quadtree decomposition

the largest contiguous "white" region). The image was then recursively quartered into "tiles", as shown in Figure 6, until either: (i) uniform tiles (quadrants) were arrived at or (ii) a maximum level of decomposition was reached. Figure 6(a) shows an example preprocessed household image and the associated quadtree decomposition in Figure 6(b). The generated decomposition was then stored in a quadtree format. The nodes in this tree were labelled with a grayscale encoding generated using a mean histogram of grayscale colours for each block, in this manner eight labels were derived, each describing a range of 32 consecutive intensity values. Figure 7 presents an example of a quadtree where the top level node (the root) represents the entire (cropped) image, the next level (Level 1) its immediate child nodes, and so on. In the figure the nodes are labelled numerically from 1 to 8 to indicate the grayscale ranges.The edges are labelled using a set of identifiers $\{1,2,3,4\}$ representing the NW, NE, SW and SE child tiles associated with the decomposition of a particular parent tile. In Figure 7 the number in square brackets alongside each node is a unique node identifier derived according to the decomposition.

The quadtree (graph) based representation served to capture the content of individual fine segmented household images, although a disadvantage of the representation is the "boundary problem" where objects of interested may be located at the intersection of a decomposition. A second disadvantage is that the quadtree representation is not well suited to the purpose of classifier generation and subsequent usage of the generated classifier. The idea was therefore to identify frequently occurring patterns (subgraphs or subtrees) and treat these patterns as features within a feature vector representation. The motivation was the conjecture that such patterns would be indicative of commonly occurring features that might exist across the image set which in turn might be indicative

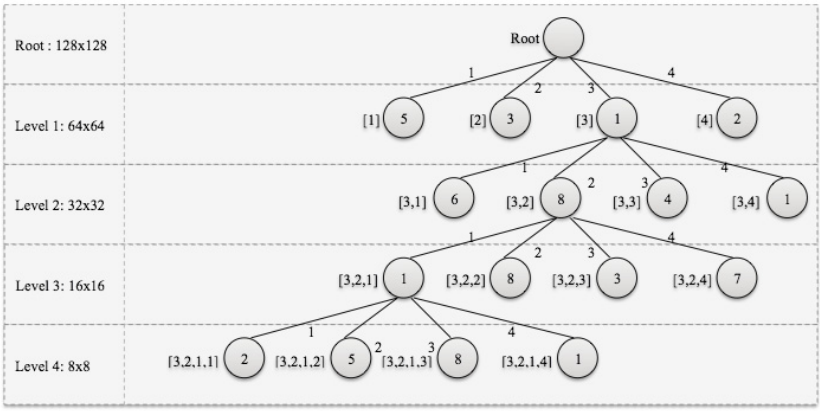


Fig. 7. The example of tree construction

of individual class labels. A number of different frequent subgraph miners could have been used; however, for the experiments described later in this paper, the well known gSpan frequent subgraph mining algorithm [13] was adopted. This uses the idea of a support threshold σ to define the concept of a frequent subgraph, the lower the value of σ the greater the number of frequent subgraphs that will be discovered. The selected value for σ will therefore influence the effectiveness of the final classifier.

Once a set of frequently occurring subgraphs has been identified these can be arranged into a feature vector representation such that each vector element indicates the presence or absence of a particular subgraph with respect to each household (record). Table 1 shows the format of the result. The rows in the table represent individual household (records) numbered from 1 to m, and the columns individual frequent subgraphs represented by the set $\{S_1, S_2, \ldots, S_n\}$. The values 0 or 1 indicate the absence or presence of the associated subgraph for the record in that row. This feature vector representation is ideally suited to both the application of classifier generation algorithms and the future usage of the generated classifiers.

Table 1. The example of Feature Vector

Vector	S_1	S_2	S_3	S_4	S_5	...	S_n
1	1	0	1	1	1	...	1
2	1	1	0	1	1	...	0
3	1	0	1	0	1	...	1
...	...	...	...	...	...	...	...
m	0	1	1	0	1	...	1

6 Evaluation

The evaluation of the proposed population estimation mining process is presented in this section. Extensive evaluation has been conducted with respect to the proposed

techniques. This section reports on only the most significant results obtained (there is insufficient space to allow for the presentation of all the results obtained). The evaluation was conducted by considering a specific case study directed at a rural area of Ethiopia. Sub-section 6.1 provides further detail of this study area. The overall aim of the evaluation was to provide evidence that census data can be effectively collected using the proposed approach. To this end three sets of experiments were conducted as follows:

1. A set of experiments to identify the most appropriate support threshold for use with respect to the frequent subgraph mining (Sub-section 6.2).
2. A set of experiments to analyse the most appropriate number (k) of features to retain during feature selection (Sub-section 6.3).
3. A set of experiments to determine the most appropriate classifier generation paradigm. To this end a selection of different classifier generators, taken from the Waikato Environment for Knowledge Analysis (WEKA) machine learning workbench[3], were considered (Sub-section 6.4).

Each is discussed in further detail in Sub-sections 6.2 to 6.4 below. Ten fold Cross-Validation (TCV) was applied throughout and performance recorded in terms of: (i) accuracy (AC), (ii) area under the ROC curve (AUC), (iii) sensitivity (SN), (iv) specificity (SP) and (v) precision (PR).

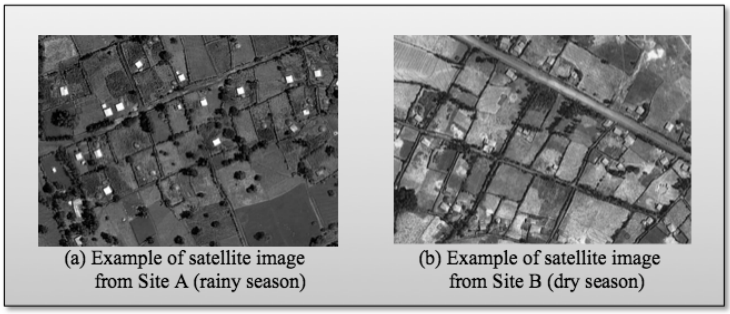


Fig. 8. Examples of satellite images from test Sites A and B

6.1 Data Set

For the evaluation reported in this section a training dataset comprising 120 records was used: 70 records from Site A and 50 from Site B (see Figure 1). High resolution satellite images were used, obtained from GeoEye at a 50cm ground resolution, made publicly available by Google Earth[4]. The images for site A were dated 22 August 2009, while those for Site B were dated 11 February 2012. The significance is that the Site A images were captured during June to August which is the "rainy season" in Ethiopia, and thus

[3] http://www.cs.waikato.ac.nz/ml/weka/
[4] http://www.google.com/earth/index.html

the households tend to have a green background; while the Site B images were captured during September to February which is the "dry season", hence the images tended to have a light-brown background. The contrast with respect to images obtained during the rainy season was much greater than those obtained during the dry season, hence it was conjectured that the dry season images (Site B) would provide more of a challenge.

The corresponding ground household information, required for the training data, included family size and household coordinates (latitude and longitude). This was collected by University of Liverpool ground staff in May 2011 and July 2012. With respect to all 120 records the following was noted with respect to family size: (i) the minimum was 2, (ii) the maximum was 12, (iii) the average was 6.31, (iv) the medium was 6 and (v) the standard deviation was 2.56. Therefore, for evaluation proposes, the labeled households were separated into three classes: (i) *small family* (less than 6 people), (ii) *medium family* (between 6 and 8 people), and (iii) *large family* (more than 8 people). Some statistics concerning the class distributions for the Sites A and B data sets are presented in Table 2.

Table 2. Class label distribution for Site A and B data sets

Location	Small family	Medium family	Large family	Total
Site A	28	32	10	70
Site B	19	21	10	50
Total	47	53	20	120

Table 3. Number of identified features produced using a range of σ values with respect to the Site A and B data

minSup	Site A	Site B
10	757	420
20	149	119
30	49	60
40	24	39
50	12	19

6.2 Subgraph Mining

In order to investigate the effect the value of the subgraph mining support threshold σ had on classification performance a sequence of different σ values were considered ranging from 10 to 50 incrementing in steps of 10. The number of features (subgraphs) generated in each case are presented in Table 3. From the table it can be seen that, as would be expected, the number of identified subgraphs decreases as the value for σ increases (and vice-versa). Note that attempts to conduct the sub-graph mining using σ values of less than 10 proved unsuccessful due to the computational resource required (subgraph mining is computationally expensive).

To reduce the overall size of the feature space Information Gain feature selection was applied to select the top k features ($k = 25$ was used because the experiments reported

in Sub-section 6.3, had revealed that this was the most appropriate value for k). Naive Bayes classification was applied with respect to each of the resulting datasets (because the experiments reported in Sub-section 6.4 had indicated that this produced the best overall result). The results are presented in Table 4 (best values are highlighted in bold). From the table it can be observed that best results were obtained using $\sigma = 10$ for both Site A (rainy season) and Site B (dry season); giving sensitivity values of 0.671 and 0.780, and AUC values of 0.769 and 0.829 respectively.

Table 4. Classification outcomes using a range of σ values with respect to Site A and B data ($k = 25$)

minSup	Site A					Site B				
	AC	AUC	PR	SN	SP	AC	AUC	PR	SN	SP
10	**0.671**	**0.769**	**0.686**	**0.671**	**0.765**	**0.780**	**0.829**	**757**	**0.780**	**0.813**
20	0.571	0.660	0.582	0.571	0.741	0.580	0.771	0.579	0.580	0.768
30	0.443	0.565	0.459	0.443	0.670	0.540	0.698	0.544	0.540	0.749
40	0.343	0.389	0.340	0.343	0.555	0.440	0.615	0.440	0.440	0.688
50	0.357	0.426	0.320	0.357	0.538	0.340	0.459	0.385	0.340	0.615

6.3 Feature Selection

To identify the effect on classification performance of the value of k with respect to the adopted Information Gain feature selection method, a sequence of experiments was conducted using a range of values fore k from 10 to 35 incrementing in steps of 5 and without feature selection. For the experiments $\sigma = 10$ was used because previous experiments, reported in Sub-section 6.2, had indicated that a value of $\sigma = 10$ produced the best performance. The Naive Bayes classifier was again adopted. The results produced are presented in Table 5. From the table it can be seen that: (i) for Site A the best result tended to be obtained using $k = 25$ (sensitivity = 0.671 and AUC = 0.769), and (ii) for Site B the best result tended to be obtained using either $k = 20$ (sensitivity = 0.780 and AUC = 0.838) or $k = 25$ (sensitivity = 0.780 and AUC= 0.829). Hence we conclude $k = 25$ to be the most appropriate value for k (this is why $k = 25$ was used with respect to the experiments reported in Sub-section 6.2).

6.4 Classification Learning Methods

To determine the most appropriate classification method eight different algorithms were considered: (i) Decision Tree generators (C4.5), (ii) Naive Bayes, (iii) Averaged One Dependence Estimators (AODE), (iv) Bayesian Network, (v) Radial Basis Function Networks (RBF Networks), (vi) Sequential Minimal Optimisation (SMO), (vii) Logistic Regression and (viii) Neural Networks. For the experiments $\sigma = 10$ was used because this produced the best result with respect to the experiments reported in Sub-section 6.2, together with $k = 25$ for feature selection because this produced the best result with respect to the experiments reported in Sub-section 6.3. The obtained results are presented in Table 6. From the Table it can be observed that:

Table 5. Comparison of different values of k with respect to Information Gain feature selection in terms of classification performance

Number of k	Site A					Site B				
	AC	AUC	PR	SN	SP	AC	AUC	PR	SN	SP
10	0.557	0.710	0.559	0.557	0.708	0.760	0.821	0.775	0.760	0.850
15	0.571	0.753	0.576	0.710	0.740	0.740	0.838	0.751	0.740	0.837
20	0.657	0.769	0.678	0.657	0.743	**0.780**	**0.838**	**0.786**	**0.780**	**0.859**
25	**0.671**	**0.769**	**0.686**	**0.671**	**0.765**	**0.780**	**0.829**	**0.805**	**0.780**	**0.852**
30	0.629	0.761	0.631	0.629	0.746	0.720	0.836	0.757	0.720	0.813
35	0.614	0.759	0.615	0.614	0.746	0.720	0.843	0.764	0.720	0.815
-	0.257	0.452	0.318	0.286	0.596	0.300	0.451	0.317	0.300	0.631

- With respect to the Site A data, the best results (Sensitivity = 0.700 and AUC= 0.718) were obtained using the AODE classifier, and with respect to the Site B data, the best results (Sensitivity = 0.780 and AUC= 0.829) were obtained using the Naive Bayes classifier .
- The C4.5 and Logistic regression classifiers did not perform well for Site A.
- The Logistic Regression and Neural Network classifiers did not perform well with respect to Site B.

Thus, in conclusion a number of different classifiers produced a good performance, but overall the Naive Bayes classifier proved to be the most effective.

Table 6. Comparison of different classifier generators in terms of classification performance

Learning method	Site A					Site B				
	AC	AUC	PR	SN	SP	AC	AUC	PR	SN	SP
C4.5	0.529	0.620	0.526	0.529	0.691	0.600	0.711	0.605	0.600	0.769
Naive Bayes	0.671	0.769	0.686	0.671	0.765	**0.780**	**0.829**	**0.805**	**0.780**	**0.852**
AODE	**0.700**	**0.809**	**0.713**	**0.700**	**0.779**	0.740	0.820	0.738	0.740	0.842
Bayes Network	0.657	0.775	0.672	0.657	0.762	0.760	0.823	0.767	0.760	0.847
RBF Network	0.571	0.709	0.579	0.571	0.707	0.680	0.750	0.668	0.680	0.820
SMO	0.557	0.663	0.557	0.557	0.703	0.620	0.737	0.618	0.620	0.778
Logistic Regression	0.500	0.659	0.502	0.500	0.674	0.540	0.635	0.543	0.540	0.739
Neural Network	0.686	0.774	0.693	0.686	0.979	0.580	0.691	0.584	0.580	0.771

7 Conclusion

In this paper a framework for population estimation mining (census mining) was proposed founded on the concept of applying classification techniques to satellite imagery. Of particular note is the subgraph feature vector representation that was used to encode household imagery. The proposed framework was evaluated using test data collected from two villages in the Ethiopian hinterland. The conducted evaluation indicated that when using a minimum support threshold of $\sigma = 10$ for the subgraph mining, a value

of $k = 25$ for feature selection and a Naive Bayes, good results could be obtained. For future work the research team intend to conduct a large scale census collection exercise using the proposed framework.

References

1. Alsalman, A.S., Ali, A.E.: Population Estimation From High Resolution Satellite Imagery: A Case Study From Khartoum. Emirates Journal for Engineering Research 16(1), 63–69 (2011)
2. Cheng, L., Zhou, Y., Wang, L., Wang, S., Du, C.: An estimate of the city population in china using dmsp night-time satellite imagery. In: IEEE International on Geoscience and Remote Sensing Symposium, IGARSS 2007, pp. 691–694 (2007)
3. Cole, R.A., NovickNovick, D.G., Burnett, D., Hansen, B., Sutton, S., Fant, M.: Towards automatic collection of the us census. In: 1994 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 1994, vol. 1, pp. I/93–I/96 (1994)
4. Dittakan, K., Coenen, F., Christley, R.: Towards the collection of census data from satellite imagery using data mining: A study with respect to the ethiopian hinterland. In: Bramer, M., Petridis, M. (eds.) Proc. Research and Development in Intelligent Systems XXIX, pp. 405–418. Springer, London (2012)
5. Gonzalez, R.C., Woods, R.E.: Digital Image Processing, 3rd edn. Pearson Prentice Hall (2007)
6. Han, J., Ma, K.: Fuzzy color histogram and its use in color image retrieval. IEEE Transactions on Image Processing 11(8), 944–952 (2002)
7. Javed, Y., Khan, M.M., Chanussot, J.: Population density estimation using textons. In: 2012 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 2206–2209 (2012)
8. Jeong, S., Won, C.S., Gray, R.M.: Image retrieval using color histograms generated by gauss mixture vector quantization. Computer Vision and Image Understanding 94(13), 44–66 (2004) ¡ce:title¿ Special Issue: Colour for Image Indexing and Retrieval ¡/ce:title¿
9. Schalkoff, R.J.: Digital Image Processing and Computer Vision. Wiley (1989)
10. Vijayaraj, A., DineshKumar, P.: Article:design and implementation of census data collection system using pda. International Journal of Computer Applications 9(9), 28–32 (2010) Published By Foundation of Computer Science
11. Wang, S.L., Liew, A.: Information-based color feature representation for image classification. In: IEEE International Conference on Image Processing, ICIP 2007, vol. 6, pp. VI – 353–VI – 356 (2007)
12. Wu, S., Qiu, X., Wang, L.: Population Estimation Methods in GIS and Remote Sensing: A Review. GIScience and Remote Sensing 42(1), 58–74 (2005)
13. Yan, X., Han, J.: gspan: graph-based substructure pattern mining. In: Proceedings of 2002 IEEE International Conference on Data Mining, ICDM 2003, pp. 721–724 (2002)
14. Zhang, Y., He, R., Jian, M.: Comparison of two methods for texture image classification. In: Second International Workshop on Computer Science and Engineering, WCSE 2009., vol. 1, pp. 65–68 (2009)

GMAC: A Seed-Insensitive Approach to Local Community Detection

Lianhang Ma[1], Hao Huang[2], Qinming He[1], Kevin Chiew[3], Jianan Wu[1], and Yanzhe Che[1]

[1] College of Computer Science, Zhejiang University, P.R. China
[2] School of Computing, National University of Singapore, Singapore
[3] School of Engineering, Tan Tao University, Vietnam

Abstract. Local community detection aims at finding a community structure starting from a seed (i.e., a given vertex) in a network without global information, such as online social networks that are too large and dynamic to ever be known fully. Nonetheless, the existing approaches to local community detection are usually sensitive to seeds, i.e., some seeds may lead to missing of some true communities. In this paper, we present a seed-insensitive method called GMAC for local community detection. It estimates the similarity between vertices via the investigation on vertices' neighborhoods, and reveals a local community by maximizing its internal similarity and minimizing its external similarity simultaneously. Extensive experimental results on both synthetic and real-world data sets verify the effectiveness of our GMAC algorithm.

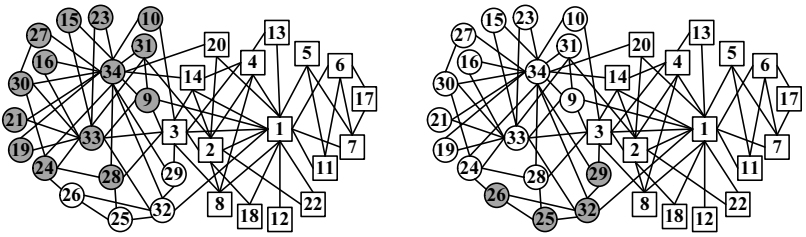
Keywords: Local community detection, similarity, seed-insensitive.

1 Introduction

Community detection in networks has received considerable attention in the fields of data mining and knowledge discovery. The majority of community detection algorithms assume that the global structure of a given network is known before the detection process [8]. Nevertheless, in reality, it is more often that the global structure is unavailable. For example, WWW and the online social networks such as Facebook and Twitter are always too large and dynamic to ever be known fully. Therefore, there are some other methods proposed to find local communities in a network without global information but starting from several specified vertices known as the seeds [5,12,14,16]. In literature, this issue is referred to as local community detection (in short as LCD henceforth).

In general, local community detection starts from one seed, and expands outward by absorbing external vertices into it until the local community quality stops improving. Therefore, how to evaluate the quality of a local community plays the key role in LCD. So far, there are two types of approaches proposed for this purpose, i.e., the degree-based and the similarity-based methods. Nonetheless, a stability problem [21] arises from these methods when they use different seeds for LCD. For example, Fig. 1 shows the LCD results of a state-of-the-art

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 297–308, 2013.



(a) Seed: Vertex 30 (b) Seed: Vertex 25

Fig. 1. LCD results of LTE on Karate network

algorithm LTE [12] using different seeds, in which the shape of each vertex indicates its true community and the colored vertices are the member of the detected local community. From the figure, we can observe that different seeds significantly affect the performance of LTE. Similar to this observation, the stability problem caused by different seeds commonly exists in other LCD algorithms.

Aiming at a seed-insensitive approach to LCD, we propose GMAC (**G**reedy **MA**ximization of **C**ompactness-Isolation) algorithm by introducing a novel similarity measure known as d-neighbors similarity (d-NS for short). Different from existing similarity measures that only focus on the adjacent vertices and their common neighbors, d-NS also takes into account non-adjacent vertices within a distance away so as to better reflect the real similarity between vertices especially the smilarity between non-adjacent vertices. Moreover, to quantitatively evaluate the quality of a local community, the d-NS across its internal vertices and the d-NS between its internal and external vertices are combined as a new metric for local community quality called Compactness-Isolation (CI for short). Our GMAC algorithm keeps absorbing selected external vertices to a local community until there is no improvement on this community's CI value.

Our key contribution is two-fold, i.e., (1) the d-NS measurment, a novel and more effective way to evaluate vertices' similarity, and (2) the GMAC algorithm, a seed-insensitive approach to LCD. Besides, another contribution of this paper is that we first propose two seed-insensitivity criteria for LCD algorithms.

The remaining sections are organized as follows. After literature review in Section 2, we give the problem statement and definitions of d-NS and CI in Section 3, followed by proposing our GMAC algorithm in Section 4 together with two seed-insensitivity criteria for LCD algorithms in Section 5. We present our experimental results in Section 6 before concluding the paper in Section 7.

2 Related Work

According to the ways of how to evaluate the quality of a local community, the existing approaches to LCD can be classified into two main categories, namely (1) the degree-based methods, and (2) the similarity-based methods.

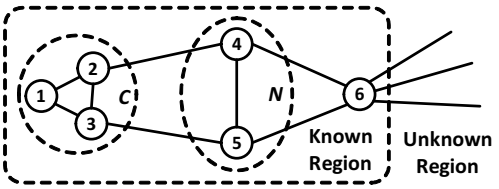


Fig. 2. Example of known and unknown regions during LCD

Degree-based methods evaluate the local community quality via the investigation on the vertices' degrees. Some naive solutions, such as the ℓ-shell search algorithm [2], discovery-then-examination approach [4], and outwardness-based method [1], only consider the number of edges inside and outside a local community. Some other work [5, 16] pay more attention on the boundary vertices, and can achieve high recall with a price of low precision. In addition, by randomizing the seed vertex and varying a resolution parameter which controls the size of the community, overlapping and hierarchical local communities can also be discovered via the investigation of vertex degree [14].

Similarity-based methods utilize similarities between vertices to help evaluate the local community quality. The LTE algorithm [12] is a representative of similarity-based methods, using a well-designed metric for local community quality known as Tightness. There are a few alternative similarity-based metrics such as VSP [15] and RSS [3] that can also help to evaluate the local community quality, although they are not originally designed for LCD.

Besides the above two types of LCD algorithms, another related work to LCD is identifying global communities by first detecting local communities and then merging them [7, 9, 11, 18].

3 Problem Statement and Preliminaries

3.1 Problem Statement

In this paper, we focus on detecting a local community in a undirected graph $G = (V, E)$ starting from a seed $s \in V$, where V is the set of vertices and E is the set of edges in graph G.

Note that in LCD, the entire network structure is unknown at the beginning. Besides the detected local community, only partial information, i.e., the local community's neighbors and their linkage information, is available after each detection process. There is an example in Fig. 2, in which the community C, its neighbors in set N, and any other vertices (i.e., vertex 6 here) linked to its neighbors are known, while the rest part of the network is temporarily unknown.

Given the above settings, our LCD problem can be defined as follows.

Input: A seed vertex s.

Output: The local community $C \subseteq V$ $(s \in C)$ with an optimized local community quality.

Table 1. d-neighbors similarity between vertices 17 and 3 in Karate network (see Fig. 1)

d	$N_{17}(d)$	$N_3(d)$	$\phi(17,3,d)$
1	6, 7	1, 2, 4, 8–10, 14, 28, 29, 33	0
2	1, 5, 11, and $N_{17}(1)$	5–7, 11–13, 15, 16, 18, 19, 20–25, 30–32, 34, and $N_3(1)$	$\frac{5}{30}$
3	2–4, 8, 9, 12–14, 18, 20, 22, 32, and $N_{17}(2)$	17, 26, 27, and $N_3(2)$	$\frac{12}{34}$

In what follows, we introduce a d-neighbors similarity based metric for local community quality known as Compactness-Isolation.

3.2 d-Neighbors Similarity

The linkage information between vertices plays a critical role to evaluate vertices' similarities. Nonetheless, most of the existing similarity measures only focus on adjacent vertices and their common neighbors, which may miss the information between the non-adjacent vertices. To avoid this limitation, we investigate the linkage information on a relatively larger level via the following concept known as d-level neighbors.

Definition 1 (d-level Neighbors). *The d-level neighbors of a vertex s, denoted by $N_s(d)$, is a set of vertices whose shortest path length to s is within d, i.e.,*

$$N_s(d) = \{v | v \in V, dist(v, s) \leqslant d\}$$

where $dist(v, s)$ is the length of shortest path between vertices v and s.

When $d = 1$, the d-level neighbors of a vertex is still its adjacent vertices. Thus, we usually set $d > 1$ to include the non-adjacent neighbors into the investigation. Furthermore, as pointed out by Huang *et al.* [12], two vertices bear higher similarity if they share more common neighbors. Hence, for each two vertices, we utilize their ratio of the common neighbors in their d-level neighbors to evaluate the similarity between them, and present the following similarity metric called d-neighbors similarity (d-NS for short).

Definition 2 (d-Neighbors Similarity). *The d-neighbors similarity between two vertices u and v, denoted by $\phi(u, v, d)$, is defined as*

$$\phi(u, v, d) = \frac{|N_u(d) \bigcap N_v(d)|}{|N_u(d) \bigcup N_v(d)|}.$$

Table 1 gives an example of d-NS, from which we can observe that d-NS can help quantitatively evaluate two vertices' similarity even if they are non-adjacent. Moreover, we can also observe that the d-NS between two vertices grows with d value. In fact, if d is extremely large, in the current known region of a network, all vertices will become the d-level neighbors of each vertex, and also the common neighbors of any two vertices, resulting in indistinguishable d-NS values (all equal or close to 1) for the vertex pairs. Therefore, it is more practicable to set $d = 2$ or $d = 3$.

3.3 Compactness-Isolation Metric

In community detection, two important characteristics of a community are widely used in the existing work [5, 8, 12, 14, 16], namely (1) the compactness characteristic, i.e., a community is a sub-graph within which the vertices are very similar to each other, and (2) the isolation characteristic, i.e., the internal vertices of a community differ from those from the outside. Motivated by these two characteristics, we propose the following local community quality metric called Compactness-Isolation (CI for short) based on our d-NS measure.

Definition 3 (Compactness-Isolation Metric). *For a local community C with a neighbor set N, the Compactness-Isolation value of C, denoted by $CI(C, d)$, is defined as*

$$CI(C,d) = \frac{\sum_{u,v \in C,(u,v) \in E} \phi(u,v,d)}{1 + \sum_{u \in C, v \in N,(u,v) \in E} \phi(u,v,d)}.$$

In CI, its numerator refers to the sum of d-NS values between any two internal vertices of the local community C, while its denominator is one plus the sum of d-NS values between C's internal vertices and their adjacent external vertices. Such a denominator is to avoid dividing by zero. According to the compactness and isolation characteristics, the vertices of a good local community should have high internal similarities and low external similarities, leading to a high CI value. Consequently, we can quantitatively evaluate the quality of local community via the computation of its CI value.

Since our d-NS measure evaluates vertices' similarity on a relatively larger level rather than only focusing on one-step linkages, it helps better reflect the true vertices' similarities in a local area. Hence, when a local community is going to absorb an external vertex, the d-NS based CI metric will better reflect the real change on the ratio of the internal similarities to the external similarities for the local community, and thus help it make a more correct decision.

In addition, based on the definition of CI, we have the following theorem.

Theorem 1. *Given a local community C in a network $G = (V, E)$ and the neighbor set $N = \{u \in V | u \notin C, v \in C, (u, v) \in E\}$ of C, when adding a node $a \in N$ into C, the CI value of the new local community $C \cup \{a\}$ will increase, i.e., $CI(C \cup \{a\}, d) > CI(C, d)$, iff $\frac{IS(a)}{ES(a)-IS(a)} > CI(C, d)$, where $IS(a) = \sum_{(v,a) \in E, v \in C} \phi(v, a, d)$, $ES(a) = \sum_{(u,a) \in E, u \notin C} \phi(u, a, d)$.*

Proof. By adding $a \in N$ into C, the gain $\triangle CI$ of CI value can be calculated as

$$\triangle CI = CI\left(C \cup \{a\}, d\right) - CI(C,d)$$

$$= \frac{\sum\limits_{u,v \in C,(u,v) \in E} \phi(u,v,d) + IS(a)}{\sum\limits_{u \in C, v \in N,(u,v) \in E} \phi(u,v,d) - IS(a) + ES(a) + 1} - \frac{\sum\limits_{u,v \in C,(u,v) \in E} \phi(u,v,d)}{\sum\limits_{u \in C, v \in N,(u,v) \in E} \phi(u,v,d) + 1}$$

As $a \in N$, $\sum_{u \in C, v \in N,(u,v) \in E} \phi(u, v, d) - IS(a) \geqslant 0$. Thus, the denominators in the above equation are all positive for certain. By combining the two fractions

above to one item and ignoring its positive denominator, inequality $\triangle CI > 0$ can be transformed as

$$\frac{1 + \sum_{u \in C, v \in N, (u,v) \in E} \phi(u, v, d) + \sum_{u,v \in C, (u,v) \in E} \phi(u, v, d)}{\sum_{u,v \in C, (u,v) \in E} \phi(u, v, d)} > \frac{ES(a)}{IS(a)}$$

which is equivalent to $\frac{IS(a)}{ES(a) - IS(a)} > CI(C, d)$, and the proof completes. ■

4 GMAC Algorithm

With the d-NS based local community quality metric, namely CI, we propose our GMAC algorithm for LCD by adopting a greedy strategy to optimize the quality of the detected local community in terms of its CI value.

4.1 Algorithm

The pseudo-code of GMAC is presented in Table 2. GMAC starts its LCD process from a seed s together with its neighbor set N (line 2). To absorb the vertices with the highest potential of being members of the local community, in the while-loop (lines 3–10), GMAC keeps searching vertex $a \in N$ which has the current maximum sum of d-NS values with vertices within C (line 4). If the inequality in line 5 holds, then a will be added to C (line 6) because the CI value of the new local community (i.e., $CI(C \cup \{a\}, d)$) will increase according to Theorem 1, and at the same time, the neighbor set N will be updated (line 6). Otherwise, a will be removed from current N (line 8). This while-loop will be discontinued when N becomes empty, and then the current C will be returned as the final local community (line 11).

Table 2. GMAC algorithm for local community detection

Input: A seed s, parameter d.	
Output: The local community C that s belongs to.	
1:	Initialize $C = \varnothing$, $N = \varnothing$;
2:	Add s to C, and add neighbors of s to N;
3:	**while** $N \neq \varnothing$ **do**
4:	$a = \arg\max_{a \in N} \sum_{u \in C} \phi(u, a, d)$;
5:	**if** $\frac{IS(a)}{ES(a) - IS(a)} > CI(C, d)$ **then**
6:	Add a to C and update N;
7:	**else**
8:	Remove a from N;
9:	**end if**
10:	**end while**
11:	Return C.

4.2 Complexity Analysis

In our GMAC algorithm, the most computationally expensive step is to find a vertex $a \in N$ having the maximal sum of d-NS values with vertices within C (line 4). By using a dynamic priority queue implemented by Fibonacci heap [12], the time complexity of this step can be reduced to $O(t^d \cdot (m + n \log n))$, where m is the number of edges in current known region of the given network, n is the number of known vertices, and t is the mean degree of known vertices. Since d is a small positive integer, for spare networks (i.e., t is small), time complexity of our GMAC algorithm is $O(m+n \log n)$, which is lower than most of the existing algorithms and comparable to the fastest algorithms like LMR [5] and LTE [12].

5 Seed-Insensitivity Criteria for LCD Algorithms

To quantitatively analyze the seed insensitivity of GMAC as well as other LCD algorithms, we present two seed-insensitivity criteria, i.e., Consistency of Detection Results, and Ratio of Consistency.

5.1 Consistency of Detection Results

The local community detected by an LCD algorithm $\mathcal{A}$ starting from a seed s can be denoted as a set $\mathcal{A}(s) = \{s\} \bigcup \{v_i | v_i \in V, 1 \leqslant i \leqslant p\}$. We define that algorithm $\mathcal{A}$ is strictly seed-insensitive on set $\mathcal{A}(s)$, iff

$$\mathcal{A}(s) = \mathcal{A}(v_i),\ \forall i \in \{1, 2, \ldots, p\}.$$

By extending the above strict definition to a more loose and quantitative measure, we propose the following criterion to evaluate an LCD algorithm's seed-insensitivity around a specific seed.

Definition 4 (Consistency of Detection Results). *For an LCD algorithm $\mathcal{A}$ using a seed s and with a detection result $\mathcal{A}(s) = \{s\} \bigcup \{v_i | v_i \in V, 1 \leqslant i \leqslant p\}$, let $f(v_i)$ denote the F-score of the detection result $\mathcal{A}(v_i)$ with $\mathcal{A}(s)$ as the ground truth. Then, the consistency of detection results of algorithm $\mathcal{A}$ on set $\mathcal{A}(s)$, denoted by $CDR(\mathcal{A}, s)$, is defined as*

$$CDR(\mathcal{A}, s) = 1 - \sqrt{\frac{1}{p+1} \sum_{v \in \mathcal{A}(s)} \Big(f(v) - \mu\Big)^2}$$

where μ is the mean value of $f(v)$ $(v \in \mathcal{A}(s))$.

In $CDR(\mathcal{A}, s)$, $\sqrt{\frac{1}{p+1} \sum_{v \in \mathcal{A}(s)} \big(f(v) - \mu\big)^2}$ refers to the standard deviation of the F-scores of algorithm $\mathcal{A}$'s detection results using different seeds $v \in \mathcal{A}(s)$. Thus, if detection results are more consistent, then the standard deviation is lower,

leading to a greater $CDR(\mathcal{A}, s)$ value. In other words, a greater $CDR(\mathcal{A}, s)$ value indicates that algorithm $\mathcal{A}$ has a higher seed-insensitivity on set $\mathcal{A}(s)$.

5.2 Ratio of Consistency

$CDR(\mathcal{A}, s)$ provides a measure for algorithms' seed-insensitivity on a sub-network, in what follows, we present another criterion, namely Ratio of Consistency, to evaluate LCD algorithms' seed-insensitivity on a entire network.

Definition 5 (Ratio of Consistency). *For an LCD algorithm $\mathcal{A}$ running on a given network $G = (V, E)$, the ratio of consistency of $\mathcal{A}$, denoted by $RC(\mathcal{A}, \tau)$, is defined as*

$$RC(\mathcal{A}, \tau) = \frac{1}{|V|}\Big|\big\{v|v \in V, CDR(\mathcal{A}, v) \geqslant \tau\big\}\Big|.$$

$RC(\mathcal{A}, \tau)$ refers to the ratio of vertices $\{v\}$ with which as the seeds algorithm $\mathcal{A}$'s $CDR(\mathcal{A}, v)$ is not less than a threshold $\tau \in [0, 1]$. If this ratio is high, then the algorithm is assumed to have a good seed-insensitivity on the given network.

6 Experiments

In this section, we evaluate our algorithm on (1) the LCD accuracy and (2) the seed insensitivity, and conduct some case studies to confirm its effectiveness. All algorithms involved in these experiments are implemented by Java, running on a desktop PC with Intel Core 2 CPU at 2.2GHz and 2GB of RAM.

6.1 Accuracy Comparison

Accuracy Comparison on Real-World Networks. We evaluate the accuracy performance of our GMAC algorithm on the following real-world networks, i.e., (1) Zachary's karate club network (Karate for short) [22], in which $|V| = 34$ and $|E| = 78$, (2) NCAA football network (NCAA for short) [10], in which $|V| = 115$ and $|E| = 616$, (3) a co-authorship network of scientists working on network theory and experiment (NetSci for short) [17], in which $|V| = 378$ and $|E| = 914$, and (4) Western States Power Grid of the United States (Power for short) [20], in which $|V| = 4941$ and $|E| = 6594$. Since the last two networks have no ground truth, we apply a state-of-the-art graph partition algorithm known as Normalized Cut [19] to identify communities with the global structure information of these two networks, and utilize its detection results as the ground truth for the LCD algorithms. In addition, since the global community quality metrics such as the well-known Modularity metric [6] is not suitable to evaluate the quality of a detected local community, we use each vertex in a community (based on the community ground truth) as a seed and report algorithms' average precision and recall on this community.

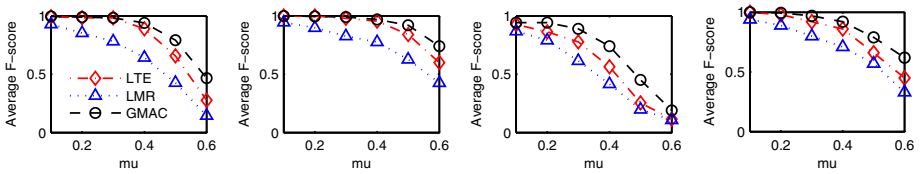
We compare our GMAC algorithm with a classical degree-based LCD algorithm LMR [5] and a high-performance similarity-based algorithm LTE [12]. The

Table 3. Average precision and recall of each algorithm on real-world networks

Karate		**LMR**			**LTE**			**GMAC**		
Comm.	**size**	**precision**	**recall**	**F-score**	**precision**	**recall**	**F-score**	**precision**	**recall**	**F-score**
Karate-A	16	0.89	0.60	**0.71**	0.94	0.52	0.67	1.00	0.55	**0.71**
Karate-B	18	0.89	0.53	0.66	0.95	0.34	0.50	1.00	0.62	**0.77**
NCAA		**LMR**			**LTE**			**GMAC**		
Comm.	**size**	**precision**	**recall**	**F-score**	**precision**	**recall**	**F-score**	**precision**	**recall**	**F-score**
AC	9	0.74	0.80	0.77	1.00	1.00	**1.00**	1.00	1.00	**1.00**
BE	8	0.59	0.67	0.63	1.00	1.00	**1.00**	1.00	1.00	**1.00**
Ten	11	0.27	0.33	0.30	1.00	1.00	**1.00**	1.00	1.00	**1.00**
MW	8	0.79	0.89	0.84	1.00	1.00	**1.00**	1.00	1.00	**1.00**
PT	10	0.64	0.82	0.72	1.00	1.00	**1.00**	1.00	1.00	**1.00**
USA	10	0.72	0.82	0.77	0.91	0.82	0.86	0.92	0.82	**0.87**
SE	12	0.85	1.00	0.92	1.00	0.58	0.74	1.00	1.00	**1.00**
WA	10	0.40	0.40	0.40	0.75	0.66	**0.70**	0.75	0.66	**0.70**
MA	13	0.86	1.00	**0.92**	1.00	0.50	0.67	0.94	0.51	0.66
Twelve	12	0.82	0.92	0.87	1.00	0.50	0.67	1.00	1.00	**1.00**
SB	7	0.48	0.53	0.50	0.63	0.51	0.56	0.64	0.52	**0.58**
NetSci		**LMR**			**LTE**			**GMAC**		
Comm.	**size**	**precision**	**recall**	**F-score**	**precision**	**recall**	**F-score**	**precision**	**recall**	**F-score**
NS-1	7	1.00	0.75	0.84	1.00	1.00	**1.00**	1.00	1.00	**1.00**
NS-2	9	0.90	0.80	0.85	0.90	1.00	**0.94**	0.91	1.00	**0.94**
NS-3	9	1.00	0.66	0.75	1.00	0.66	0.75	1.00	0.69	**0.81**
NS-4	9	1.00	0.59	0.71	1.00	0.64	0.75	1.00	0.66	**0.77**
NS-5	8	1.00	0.50	0.66	1.00	0.55	0.70	1.00	0.57	**0.73**
NS-6	9	1.00	0.49	0.65	0.90	0.55	0.68	1.00	0.56	**0.72**
NS-7	11	0.85	0.37	0.51	0.90	0.42	0.57	0.92	0.44	**0.60**
NS-8	18	0.91	0.38	0.52	0.90	0.42	0.57	0.98	0.48	**0.64**
NS-9	16	0.90	0.46	**0.56**	0.87	0.47	0.53	0.95	0.33	0.46
NS-10	17	1.00	0.32	0.46	1.00	0.39	0.52	1.00	0.41	**0.58**
NS-11	15	0.86	0.38	0.51	0.94	0.36	0.51	0.96	0.38	**0.54**
NS-12	27	0.93	0.28	0.42	0.98	0.33	**0.49**	0.97	0.29	0.44
NS-13	24	1.00	0.33	0.48	1.00	0.33	0.48	1.00	0.36	**0.53**
NS-14	22	0.97	0.24	0.38	1.00	0.33	0.48	1.00	0.36	**0.53**
NS-15	15	0.97	0.34	0.49	1.00	0.31	0.47	1.00	0.35	**0.51**
Power		**LMR**			**LTE**			**GMAC**		
Comm.	**size**	**precision**	**recall**	**F-score**	**precision**	**recall**	**F-score**	**precision**	**recall**	**F-score**
PG-1	9	1.00	1.00	**1.00**	1.00	1.00	**1.00**	1.00	1.00	**1.00**
PG-2	6	1.00	0.83	0.89	1.00	1.00	**1.00**	1.00	1.00	**1.00**
PG-3	7	0.97	0.85	0.91	1.00	0.87	**0.93**	1.00	0.87	**0.93**
PG-4	13	0.97	0.79	0.87	0.99	0.79	**0.88**	0.97	0.80	0.87
PG-5	6	1.00	0.55	0.70	1.00	0.75	0.83	1.00	1.00	**1.00**
PG-6	8	0.89	0.43	0.55	1.00	0.76	0.82	1.00	1.00	**1.00**
PG-7	11	1.00	0.33	0.47	1.00	0.71	0.78	1.00	0.70	**0.81**
PG-8	15	0.91	0.48	0.61	0.85	0.67	**0.75**	0.85	0.67	**0.75**
PG-9	10	0.48	0.55	0.51	0.97	0.63	0.73	0.93	0.65	**0.76**
PG-10	7	0.93	0.48	0.62	1.00	0.55	0.69	0.83	0.80	**0.79**
PG-11	13	1.00	0.66	**0.79**	0.92	0.54	0.68	0.93	0.55	0.69
PG-12	16	1.00	0.31	0.46	1.00	0.57	0.67	1.00	0.59	**0.70**
PG-13	8	1.00	0.39	0.55	1.00	0.53	0.67	1.00	0.82	**0.90**
PG-14	8	1.00	0.37	0.54	1.00	0.51	0.66	1.00	0.77	**0.83**
PG-15	10	0.91	0.33	0.49	1.00	0.48	**0.64**	0.86	0.45	0.57
PG-16	8	0.96	0.54	**0.68**	0.83	0.51	0.63	0.69	0.55	0.57
PG-17	16	0.95	0.33	0.48	0.94	0.50	0.63	0.94	0.54	**0.67**
PG-18	7	0.82	0.44	0.57	0.82	0.51	**0.62**	0.82	0.51	**0.62**
PG-19	13	0.94	0.45	0.58	0.88	0.49	0.62	0.89	0.54	**0.67**
PG-20	7	1.00	0.67	**0.77**	1.00	0.44	0.61	1.00	0.58	0.73
PG-21	6	0.85	0.44	0.53	1.00	0.44	0.60	1.00	0.56	**0.71**
PG-22	10	0.92	0.33	0.48	0.97	0.45	0.59	0.98	0.80	**0.87**
PG-23	11	1.00	0.32	0.48	1.00	0.42	0.57	0.94	0.61	**0.74**
PG-24	10	0.88	0.38	0.52	0.96	0.41	0.57	0.88	0.5	**0.61**
PG-25	12	0.90	0.44	0.56	0.89	0.50	0.57	0.84	0.69	**0.70**
PG-26	14	1.00	0.38	0.53	0.89	0.42	0.56	1.00	0.72	**0.81**
PG-27	12	0.88	0.27	0.40	0.86	0.43	0.56	0.77	0.46	**0.57**
PG-28	18	0.94	0.27	0.38	0.92	0.42	0.55	0.94	0.60	**0.71**
PG-29	9	0.55	0.30	0.37	0.48	0.66	0.53	0.57	0.66	**0.59**
PG-30	17	0.90	0.32	0.46	0.88	0.39	0.53	0.91	0.46	**0.58**

Table 4. Parameters of the LFR benchmark graphs

Network	n	k	$maxk$	$minc$	on	mu
5000S	5000	10	20	40	0	$0.1, 0.2, \ldots, 0.6$
5000D	5000	20	40	40	0	$0.1, 0.2, \ldots, 0.6$
10000S	10000	10	20	40	0	$0.1, 0.2, \ldots, 0.6$
10000D	10000	20	40	40	0	$0.1, 0.2, \ldots, 0.6$



(a) 5000S (b) 5000D (c) 10000S (d) 10000D

Fig. 3. Accuracy comparison on LFR benchmark graphs

comparison results are reported in Table 3 (arranged in decreasing order of LTE's F-score), from which we can observe that in terms of LCD accuracy, our GMAC algorithm usually outperforms LMR algorithm, and has a slight advantage over LTE algorithm, even though extensive experiments have proved that LTE is one of the most accurate algorithms among the state-of-the-art LCD approaches [12].

Accuracy Comparison on LFR Benchmark Graphs. The accuracy comparison is also conducted on a set of LFR benchmark graphs [13] with the parameters summarized in Table 4, where n is the number of vertices, k the average degree, $maxk$ the maximum degree, $minc$ the minimal community size, on the percent of overlapping vertices, and mu the mixing parameter. A greater mu means that communities are more indistinct in the corresponding graph.

Fig. 3 shows the accuracy comparison results, from which we can observe that our GMAC algorithm usually has a higher accuracy, and is more competent in handling networks with indistinct communities, compared against the other tested algorithms.

6.2 Seed-Insensitivity Comparison

In order to verify the advantage of our GMAC algorithm on seed-insensitivity, we evaluate each algorithm's performance on the two proposed seed-insensitivity criteria, namely Consistency of Detection Results (see Section 5.1) and Ratio of Consistency (see Section 5.2), and report the results in Table 5, where $\mathcal{A}$ represents each tested algorithm, $RC(\mathcal{A}, 0.9)$ the Ratio of Consistency of $\mathcal{A}$ with a threshold 0.9, and Aver. $CDR(\mathcal{A}, v)$ the average value of $\mathcal{A}$'s Consistency of Detection Results on all vertices of a given network.

As shown in the table, our GMAC algorithm outperforms the other tested methods on both seed-insensitivity criteria.

Table 5. Seed-insensitivity performance of each algorithm on real-world networks

Network	Size	$\mathcal{A}$	$RC(\mathcal{A}, 0.85)$	$RC(\mathcal{A}, 0.9)$	$RC(\mathcal{A}, 0.95)$	$RC(\mathcal{A}, 1)$	Aver. $CDR(\mathcal{A}, v)$
Karate	34	LMR	0.26	0.26	0.26	0.26	0.78
		LTE	0.73	0.73	0.73	0.73	0.91
		GMAC	**0.89**	**0.89**	**0.89**	**0.89**	**0.98**
NCAA	115	LMR	0.13	0.08	0.00	0.00	0.73
		LTE	0.90	0.83	0.83	0.83	0.96
		GMAC	**0.91**	**0.91**	**0.91**	**0.89**	**0.98**
NetSci	378	LMR	0.65	0.61	0.58	0.51	0.88
		LTE	0.69	0.67	0.67	0.67	0.90
		GMAC	**0.77**	**0.76**	**0.75**	**0.69**	**0.93**
Power	4941	LMR	0.48	0.45	0.43	0.43	0.84
		LTE	0.44	0.42	0.41	0.39	0.82
		GMAC	**0.54**	**0.49**	**0.46**	**0.44**	**0.86**

Table 6. Local communities discovered by GMAC on Amazon co-purchase network

Seed Used	Type	Degree	Size of Community	Member of Community
Cirque Reinvente by Cirque du Soleil	DVD	15	18	100% DVDs composed by Cirque du Soleil.
Harry Potter and the Sorcerer's Stone (Book 1)	Book	45	24	Other Harry Potter Series' books and DVDs (7 books and DVDs); the Lord of the Rings' DVDs (5 DVDs); the Matrix' DVDs (4 DVDs); other related DVDs (8 DVDs).

6.3 Case Study on Amazon Co-purchase Network

To confirm the effectiveness of our GMAC algorithm in practical application contexts, we perform it on an Amazon co-purchase network ($|V|= 585285$, $|E|= 4566749$) collected in Jan 2006. Table 6 shows some detection results, from which we see that (1) when DVD "Cirque Reinvente" composed by Cirque du Soleil is used as a seed, GMAC returns all Cirque du Soleil's DVDs bought by customers; and (2) if book "Harry Potter and the Sorcerer's Stone (Book 1)" is the seed, GMAC also has a reasonable detection results, i.e., other Harry Potter Series' books and DVDs, the DVD series of "The Lord of the Rings" , "The Matrix", and so on, indicating that the corresponding customers are very likely fiction fans. Other seeds are also used, and the results are similar to the above observation.

7 Conclusions

In this paper, we have proposed a seed-insensitive approach known as GMAC for local community detection by adopting a novel similarity measure called d-neighbors similarity, and introduced two seed-insensitivity criteria for local community detection algorithms. Extensive experimental results have verified the effectiveness of GMAC and its advantage in terms of seed-insensitivity.

Acknowledgements. This work was partly supported by the National Key Technologies R&D Program under Grants No. 2011BAD21B02 and 20128AH94F01.

References

1. Bagrow, J.P.: Evaluating local community methods in networks. Journal of Statistical Mechanics-Theory and Experiment 5, P05001 (2008)
2. Bagrow, J.P., Bollt, E.M.: Local method for detecting communities. Physical Review E 72(4), 046108 (2005)
3. Chen, H.H., Gou, L., Zhang, X.L., Giles, C.L.: Discovering missing links in networks using vertex similarity measures. In: SAC, pp. 138–143 (2012)
4. Chen, J., Zaïane, O.R., Goebel, R.: Local community identification in social networks. In: ASNAM, pp. 237–242 (2009)
5. Clauset, A.: Finding local community structure in networks. Physical Review E 72(2), 26132 (2005)
6. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Physical Review E 70(6), 066111 (2004)
7. Coscia, M., Rossetti, G., Giannotti, F., Pedreschi, D.: Demon: a local-first discovery method for overlapping communities. In: KDD, pp. 615–623 (2012)
8. Fortunato, S.: Community detection in graphs. Physics Reports 486(3-5), 75–174 (2010)
9. Gargi, U., Lu, W., Mirrokni, V.S., Yoon, S.: Large-scale community detection on youtube for topic discovery and exploration. In: ICWSM (2011)
10. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. Proceedings of the National Academy of Sciences 99(12), 7821–7826 (2002)
11. Gleich, D.F., Seshadhri, C.: Vertex neighborhoods, low conductance cuts, and good seeds for local community methods. In: KDD, pp. 597–605 (2012)
12. Huang, J.B., Sun, H.L., Liu, Y.G., Song, Q.B., Weninger, T.: Towards online multiresolution community detection in large-scale networks. PLOS One 6(8), e23829 (2011)
13. Lancichinetti, A., Fortunato, S.: Community detection algorithms: A comparative analysis. Physical Review E 80(5), 056117 (2009)
14. Lancichinetti, A., Fortunato, S., Kertesz, J.: Detecting the overlapping and hierarchical community structure in complex networks. New Journal of Physics 11, 033015 (2009)
15. Li, K., Pang, Y.: A vertex similarity probability model for finding network community structure. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) PAKDD 2012, Part I. LNCS, vol. 7301, pp. 456–467. Springer, Heidelberg (2012)
16. Luo, F., Wang, J.Z., Promislow, E.: Exploring local community structures in large networks. In: WI, pp. 233–239 (2006)
17. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. Physical Review E 74(3), 036104 (2006)
18. Rees, B.S., Gallagher, K.B.: Overlapping community detection by collective friendship group inference. In: ASNAM, pp. 375–379 (2010)
19. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(8), 888–905 (2000)
20. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393(6684), 440–442 (1998)
21. Xie, J., Kelley, S., Szymanski, B.K.: Overlapping community detection in networks: the state of the art and comparative study. ACM Computing Surveys 45(4) (2013)
22. Zachary, W.W.: An information flow model for conflict and fission in small groups. Journal of Anthropological Research 33, 452–473 (1977)

Finding Maximal Overlapping Communities

Eileen H.-C. Wei, Yun Sing Koh, and Gillian Dobbie

Dept of Computer Science, The University of Auckland
hwei019@aucklanduni.ac.nz, {ykoh,gill}@cs.auckland.ac.nz

Abstract. Social networks play an important role in everyday life. Nowadays there is various research that concentrates on detecting communities within these networks. Traditionally most of the community detection algorithms focus on detecting disjoint networks. However there is a need for overlapping community detection. In recent years there have been some attempts at detecting overlapping communities. Most of these techniques concentrate on just detecting these communities, none of this research tries to detect the maximal set of these communities which gives more stability. In this paper we propose a new method called Maximal-DSHRINK that allows us to detect the maximal set of overlapping communities within a social network. We show that the maximal set provides us with better quality in terms of modularity gain.

Keywords: Maximal Cluster, Modularity Gain, Social Network Analysis.

1 Introduction

As social networks have become an everyday norm, there is a growth in research related to this area. One particular strand of research concentrates on finding communities within these networks. The number of community detection algorithms has increased in recent years [1,2]. People in a social network can be naturally characterized into multiple community memberships. A person usually has disjoint connections to different social groups such as family or friends, however, they may also be associated with more than one group, for example a researcher may be active in several research areas. As such in an online social network, the number of communities a person can belong to is essentially unlimited because an individual can be associated with various groups simultaneously. This scenario is also common in other networks such as biological networks, where a node might have multiple functions. Thus increasingly there is a growing interest in overlapping community detection algorithms that allows us to identify a set of clusters, where nodes can belong to more than one cluster.

In this paper, we propose an algorithm to find the maximal set of overlapping communities. Most of the work around finding overlapping communities concentrates on finding overlapping communities that match a particular benchmark dataset, however none of this research concentrates on finding the maximal set of the communities. In a maximal set, a community is considered to be at a

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 309–316, 2013.

maximal size when all possible nodes that can be spanned into a particular community has been spanned without decreasing the quality of the community. This means we may find fewer overlaps in communities but on average the quality of the community based on the modularity gain may increase as compared to a non-maximal set.

In Section 2 we outline several existing algorithms that have been proposed related to overlapping community detection. Section 3 describes the preliminaries and definitions of the concepts related to our algorithm. In Section 4 we provide our algorithm, Maximal-DSHRINK, for community detection. In Section 5 we present thorough experimental evaluations of our results and conclude the paper in the last Section 6.

2 Related Work

Community detection in complex networks has attracted huge attention since its introduction. Overall, one can classify detection methods in two main categories, namely non-overlapping and overlapping communities. Overlapping community detection research itself can be categorized into five classes [3]. The clique percolation method [4] is based on the assumption that a community consists of overlapping sets of fully connected sub-graphs and communities are detected by searching for adjacent cliques. Algorithms using local expansion and optimization are based on growing a natural community or a partial community. Most of them depend on a local benefit function that characterizes the quality of a densely connected group of nodes [5,6]. Partitioning links [7] can be used for overlapping community detection. A node in the original graph is called overlapping if links connected to it are put in more than one cluster. Fuzzy community detection algorithms [8,9] calculate the strength of association between all pairs of nodes and communities using a soft membership vector or belonging factor. An agent based community detection [10] algorithm models each node in a social network as an autonomous agent. None of this research considers detecting a maximal overlapping community set. In our research we extend the DSHRINK algorithm which falls into the local expansion category to generate maximal sets.

3 Preliminaries

In this section, we briefly define several key concepts, terms and notation that are related to our algorithm.

Definition 1 (Structure). A network is defined as a graph $G = (V, E)$ which consists of a set of nodes denoted as a vertices set V, and a set of edges between any pair of nodes denoted as E. A set of communities in the network is denoted as C, whereby $C = \{C_1, \ldots C_k\}$ here C_k refers to a single community. A vertices's set consists of a number of nodes whereby $V = \{v_1, \ldots, v_j\}$ here v_j refers to a single node. We will be using $E(v_i, v_j)$ for the edge between v_i and v_j.

Clustering is a process of grouping objects that have similar characteristics. We therefore adopted a similarity distance-based approach to measure the distance between nodes.

Definition 2 (Similarity). The similarity between any pair of nodes v_i, $v_j \in V$ is defined as in [2]:

$$S(v_i, v_j) = \frac{\mid \Gamma(v_i) \cap \Gamma(v_j) \mid}{\sqrt{\mid \Gamma(v_i) \mid\mid \Gamma(v_j) \mid}} \tag{1}$$

where $\Gamma(v)$ is the set of v's neighbour nodes and itself.

We then calculate the distances between nodes [2].

Definition 3 (Distance). The distance between v_i, v_j is defined in [2] where:

$$d_M(v_i, v_j) = \mid\mid v_i - v_j \mid\mid M = \sqrt{(v_i - v_j)^T M (v_i - v_j)} \tag{2}$$

The matrix $M \in R^{(m \times m)}$ represents the distance matrix. The distance $d_M(v_i, v_j)$ is calculated by the square root of multiplication of the difference in similarity vectors of v_i and v_j.

Definition 4 (Local Community). Given a network $G = (V, E)$, a subset G' of G is defined as a local community iff: (1) $\forall v$ of $G' \in V$; (2) Distance of any pair of nodes in $G' < r$ where r, $r \in R^+$, is the radius of the local community G'.

Based on this definition, any node that is spanned into a local community has distance from any other node in that local community of less than the radius. In other words, nodes that have been clustered in the same cluster should have distance of at most r.

Definition 5 (Distance-based Modularity Gain). The quality or stability of a cluster can be evaluated using the distance-based modularity. The distance-based modularity of combining j communities $C_1, C_2, ..., C_j$ into a new cluster [2] is:

$$\Delta Qd = \frac{\sum_{s,t \in \{1,...,j\}, s \neq t} 2D^U_{st}}{D^T} - \frac{\sum_{s,t \in \{1,...,j\}, s \neq t} 2D^C_s D^C_t}{(D^T)^2} \tag{3}$$

where $D^U_{st} = \sum_{u \in C_s, v \in C_t} d_M(u, v)$ and is the sum of distance between any two nodes in communities C_s and C_t. D^C_i is the sum of distance between any node in cluster C_i and any node in the network G. D^T is the sum of distance between any two nodes in the network G.

A smaller value of ΔQd means a better quality of clustering, typically $\Delta Qd < 0$ to form a community. A reduced negative valued ΔQd means that there is an improvement in the quality of the community. We shrink the set of nodes when its combination can decrease the total distance-based modularity ΔQd, in other words, if the combination of nodes result in negative distance-based modularity, we consider that particular combination of nodes as a valid community.

Definition 6 (Maximal Cluster Set). A maximal cluster set, C_M, is denoted as a cluster set with the maximum number of nodes that can be spanned into the community without increasing the total distance-based modularity ΔQd of the community.

We aim to reduce the number of clusters and find the maximal cluster set; therefore we decompose the communities that do not form super nodes and try to span the nodes into super nodes to form a maximal cluster set. In the next section, we present our proposed technique.

4 Maximal Set for Overlapping Communities

In this section we describe our algorithm, Maximal-DSHRINK for community detection which is extended from the DSHRINK [2] approach. Our aim is to find the maximal cluster set which has a larger community size and better quality. By doing this, we span the nodes into possible local communities and shrink the ones which have negative ΔQd as super nodes while individual super nodes represent an individual cluster. We then extend the cluster sets to form maximal cluster sets. For example, assume there is a set of nodes in the network V=$\{v_1, v_2, v_3\}$. Initially all of the nodes are tagged as "unvisited". Here we view each node as a local community; therefore we have a community set C=$\{C_1, C_2, C_3\}$, whereby, $C_1 = \{v_1\}$, $C_2 = \{v_2\}$ and $C_3 = \{v_3\}$. In the next step we attempt to span every "unvisited" node to form stable clusters and therefore we begin with the first "unvisited" node v_1 and span it into a community according to Definition 4. We start from the first cluster C_1, and since v_1 is already in C_1, we then tag all the nodes in C_1, as visited. We then continue with the second "unvisited" node v_2. We start from the first cluster $C_1 = \{v_1\}$ again and examine the distance of all the nodes from C_1 with v_2. If one of the nodes from C_1 has a distance from v_2 of less than the radius r, we span v_2 into C_1 and tag all the nodes including v_2 in C_1 as visited. If there is no such node in C_1 that satisfies the condition, we then try to span v_2 into other possible communities $C_{2,3}$. We repeat the above step until there are no more "unvisited" nodes and then try to shrink the communities. After the above steps, we obtain C=$\{C_1, C_2, C_3\}$, whereby, $C_1 = \{v_1, v_2\}$with $\Delta Qd > 0$, $C_2 = \{v_2\}$ with $\Delta Qd < 0$, and $C_3 = \{v_1, v_3\}$ with $\Delta Qd > 0$. At this stage, we aim to shrink the communities which have negative ΔQd values as super nodes. As a result, only C_2 has $\Delta Qd < 0$ and therefore we label it as a super node. And since both C_1 and C_3 have $\Delta Qd > 0$, we will not shrink the communities. Now we have a super node C_2 and two non-super nodes C_1 and C_3. The next step is to tag all the nodes in the non-super nodes communities, C_1 and C_3, as "unvisited" and repeat the above spanning and shrinking procedures until untagging any nodes no longer decreases the number of super nodes. Finally, each super node represents an individual community.

Our algorithm consists of three different parts. The first two parts are adopted from the original DSHRINK algorithm and the last part, which is new, maximizes the cluster result. Firstly, we initialize each node in the network as an individual

community and build up a distance matrix by computing $d(v_i, v_j)$ where $v_i, v_j \in V$ and $v_i \neq v_j$. During this process, we also compute the sum of distances between each node v_i to every other node in the network v_j where $v \in V$ and $v_i \neq v_j$ which is stored in a similarity matrix S_i^T and the total distance of any pair of nodes in the network which is saved as D^T. Secondly, we begin to span every "unvisited" node to a possible local community. After spanning a node, we tag all the nodes in that particular local community as "visited" and repeat the above process repeatedly until there are no more unvisited nodes. The next step is to treat the nodes in each local community as an individual module and compute the modularity gain ΔQd of the combination of these modules according to Equation 3. Negative modularity gain indicates that this combination can decrease the total distance-based modularity ΔQd and for a stable community, it is tagged as a super node. Then, we tag all the nodes in the community as "unvisited" and repeat the first and second parts until there are no more communities that can be shrunk, that is, shrinking any community will not decrease the total distance-based modularity ΔQd.

Algorithm 1. Maximal-DSHRINK(G=(V, E))

Input: $G = (V, E)$: Information Network.
Output: $C^m = \{C_1^m, C_2^m, ..., C_k^m\}$: Maximal community set.

```
Initialize each node in V as an individual community and store it in C
for each node v_i ∈ V do
    for each node v_j ∈ V and v_i ≠ v_j do
        Compute S(vi, vj) using Eq 1;
        Compute d(vi, vj) using Eq 2 and store the distance into the distance matrix, D;
        S_i^T += S(v_i, v_j);
        D^T += d(v_i, v_j);
    end for
end for
C^d = DSHRINK(D, C)
num1 = number of super nodes at this stage;
numberOfSuperNode.decrease = true;
while (numberOfSuperNode.decrease) do
    for each C_j ∈ C do
        if !(Cj.isSuperNode) then
            for each node v_k in C_j do
                v_k.visited = false;
            end for
        end if
    end for
    C^m = DSHRINK(D, C^d);
    num2 = number of supernodes at this stage;
    if (num2<num1) then
        numberOfSuperNode.decrease = true;
    else numberOfSuperNode.decrease = false;
    end if
end while
return C^m;
```

Finally, we tag all the nodes in the communities that are not super nodes as "unvisited" and perform the first and second processes repeatedly until the number of super nodes cannot be decreased anymore. As a result, each super node is considered as an individual cluster. Any node that has not been assigned to a cluster is viewed as an outlier. Algorithm 1 shows our further implementation of Maximal-DSHRINK.

5 Experimental Evaluation

In this section we look at the datasets, evaluation measures and the results produced by our Maximal-DSHRINK method and compare that to the original DSHRINK method. We have used 40 data sets to examine our algorithm. The datasets are benchmarks[1], which have been used in previous overlapping community detection research. The datasets contain undirected and unweighted communities. Each dataset contains 1000 nodes and a varying number of edges in the range of 18954 to 20004.

To evaluate the quality of our cluster results we apply three different evaluation approaches to measure the effectiveness and the quality of our clusters: Silhouette value, Dunn's index and the difference in modularity gain. Both Silhouette value and Dunn's index are well known clustering evaluation metrics. We compute the difference in modularity gain of the clusters between the original DSHRINK algorithm and our maximal approach. Negative gain in the difference ΔQd indicates that there is a gain in stability level.

$$Diff\Delta Qd = \frac{\sum_{C \in i} \Delta Qd(C_i^d)}{\mid C^d \mid} - \frac{\sum_{C \in j} \Delta Qd(C_j^m)}{\mid C^m \mid} \tag{4}$$

where C_i^d is the cluster set resulting from the original DSHRINK algorithm and C_j^m is the maximal cluster set resulting from our approach. $Diff\Delta Qd$ is the difference between the average of the modularity gain of cluster sets generated by DSHRINK and our approach. A negative value in $Diff\Delta Qd$ means that our approach is obtaining a better modularity gain as compared to DSHRINK.

Table 1 summarises the result of the experiments for the two algorithms for each of the datasets. We notice that there is no significant difference between the Silhouette value and Dunn's index produced by our maximal set algorithm compared to the original DSHRINK when tested on all the datasets. This means that both methods produce similar cluster quality based on the cluster boundaries. However our technique has a reduced modularity gain, which means that more stable clusters are formed. We also present $\Delta Qd(C_{\oplus})$ to indicate the modularity gain of the exclusive cluster set $C_{\oplus}$, which is the resulting clusters that are not in C^d (the cluster set generated by DSHRINK) but in C^m (the cluster set generated by Maximal-DSHRINK) in Table 1. We notice that, those exclusive cluster sets have contributed negative gain to the total modularity gain and they are the crucial clusters that differentiate between the maximal cluster set and the original cluster set. The community size is shown in Figure 1 and the overlapping rate is shown in Figure 2. The x-axis on both these figures represent the number of communities that exist in the datasets. The community size is calculated by the average number of nodes contained in the community while the community per node is calculated by the average number of communities that each node is spanned into. Overall our algorithm produces communities with a larger community size, however the average number of communities that each

[1] http://www.netcom-analyzer.org/benchmarks

Table 1. Evaluation Result

	DSHIRNK			Maximal Set			Difference	
Dataset	$\Delta Qd(C^d)$	*Silhouette*	*DI*	$\Delta Qd(C^m)$	*Silhouette*	*DI*	$DiffQd$	$\Delta Qd(C_{\oplus})$
Network1	-0.00025274	0.9952544	0.3428223	**-0.0005361**	0.9895931	**0.3499128**	**-0.0002834**	-0.0001004
Network 2	-0.00088512	0.9823991	0.6459606	**-0.0023591**	0.9289951	**0.6565592**	**-0.0014740**	-8.9685E-05
Network 3	-9.61068E-05	0.9942691	0.3814203	**-0.0001463**	0.9916779	**0.6313003**	**-5.023E-05**	-7.8144E-05
Network 4	-0.000195427	0.9935839	0.3920497	**-0.0007063**	0.9720570	**0.7135083**	**-0.0005109**	-9.7318E-05
Network 5	-0.001017231	0.9848924	0.6333257	**-0.0015937**	0.9741634	**0.7440231**	**-0.0005765**	-9.3054E-05
Network 6	-0.00024663	0.9957481	0.5085037	**-0.0002748**	0.9950922	0.5085037	**-2.8195E-05**	-8.8572E-05
Network 7	-0.00069357	0.9922492	0.5206500	**-0.0007582**	0.9913158	0.5206500	**-6.4657E-05**	-0.0010912
Network 8	-0.00079240	0.9877459	0.5873393	**-0.0028774**	0.9345465	**0.8279644**	**-0.0020850**	-6.3598E-05
Network 9	-0.00185169	0.9788562	0.4403046	**-0.0023025**	0.9704539	0.4403046	**-0.0004509**	-4.3790E-05
Network 10	-0.00032196	0.9932012	0.5747560	**-0.0015877**	0.9470303	**0.7934666**	**-0.0012657**	-8.9015E-05
Network 11	-0.00034168	0.9900284	0.4780579	**-0.0017244**	0.9176829	**0.8640281**	**-0.0013827**	-0.0001080
Network 12	-1.03636E-07	0.9986603	0.5001223	-1.036E-07	0.9986603	0.5001223	4.37E-12	-9.5971E-05
Network 13	-0.00012889	0.9956307	0.4962049	**-0.0003384**	0.9885505	**0.7355317**	**-0.0002095**	-9.9339E-05
Network 14	-0.00022570	0.9953271	0.2192628	**-0.0005626**	0.9881697	**0.6420907**	**-0.0003369**	-0.0001115
Network 15	-0.00021742	0.9957051	0.6995695	**-0.0002202**	0.9957051	0.6995695	**-2.7428E-06**	-7.6453E-05
Network 16	-0.00031631	0.9944882	0.3048106	-0.0003068	0.9944882	0.3048106	9.4835E-06	-9.1092E-05
Network 17	-0.00018604	0.9964284	0.5832914	**-0.0004269**	0.9918057	**0.6347603**	**-0.0002408**	-0.0001379
Network 18	-0.00024374	0.9939674	0.4720211	**-0.0002466**	0.9939674	0.4720211	**-2.8231E-06**	-7.2634E-05
Network 19	-0.00016325	0.9964978	0.6383548	**-0.0004131**	0.9909608	**0.6439687**	**-0.0002498**	-0.0001230
Network 20	-0.00012037	0.9962761	0.6631964	**-0.0001577**	0.9950058	0.6785410	**-3.7313E-05**	-5.6020E-05
Network 21	-0.00019617	0.9951347	0.6204610	**-0.0006747**	0.9765417	**0.6670616**	**-0.0004786**	-9.5912E-05
Network 22	-0.00025507	0.9947728	0.6690906	**-0.0002625**	0.9947728	0.6690906	**-7.4133E-06**	-0.0001986
Network 23	-0.00017524	0.9959594	0.6704829	**-0.0004455**	0.9891571	**0.7232956**	**-0.0002703**	-8.287E-05
Network 24	-0.00017753	0.9954218	0.7013491	**-0.0001825**	0.9954218	0.7014391	**-4.9296E-06**	-8.8264E-05
Network 25	-9.0243E-05	0.9962080	0.6597236	**-9.8350E-05**	0.9962080	0.6597236	**-8.1138E-06**	-9.2735E-05
Network 26	-0.00010565	0.9963419	0.5567477	**-0.0001057**	0.9963419	0.5567477	**-8.6620E-08**	-0.0001988
Network 27	-0.00013752	0.9963304	0.5739437	**-0.0006362**	0.9812211	0.5515613	**-0.0004987**	-9.2304E-05
Network 28	-0.00014031	0.9965859	0.4848469	**-0.0001425**	0.9965859	0.4848469	**-2.2192E-06**	-8.2974E-05
Network 29	-1.4728E-05	0.9954775	0.6281135	**-3.8916E-05**	0.9914814	**0.6913431**	**-2.4187E-05**	-6.0662E-05
Network 30	-1.6862E-05	0.9959182	0.7100300	-1.5761E-05	0.9959182	0.7100300	1.1014E-06	-5.2846E-05
Network 31	-7.7637E-05	0.9958404	0.7261447	**-8.2723E-05**	0.9958404	0.7261447	**-5.0856E-06**	-0.0008364
Network 32	-0.00012625	0.9970751	0.7077546	**-0.0003368**	0.9914971	0.5301142	**-0.0002106**	-8.2491E-05
Network 33	-0.00020005	0.9952908	0.7011493	**-0.0002067**	0.9952908	0.7011493	**-6.6560E-06**	-4.8477E-05
Network 34	-8.6289E-05	0.9948652	0.6650557	**-8.9856E-05**	0.9948652	0.6650557	**-3.5674E-06**	-0.0001297
Network 35	-6.7601E-05	0.9958138	0.6994768	**-7.8575E-05**	0.9958138	0.6994768	**-1.0975E-05**	-9.1828E-05
Network 36	-0.00015799	0.9957643	0.6369897	**-0.0005155**	0.9853156	**0.6439727**	**-0.0003575**	-8.8020E-05
Network 37	-0.00017561	0.9971423	0.6682034	**-0.0009563**	0.9821507	**0.6971489**	**-0.0007807**	-9.1885E-05
Network 38	-2.3151E-05	0.9959931	0.6664640	**-0.0001179**	0.9721217	**0.8268187**	**-9.4789E-05**	-7.0347E-05
Network 39	-0.00022063	0.9945447	0.3771074	**-0.0002220**	0.9945447	0.3771074	**-1.3660E-06**	-4.2762E-05
Network 40	-8.7308E-05	0.9961813	0.6691524	**-9.192E-05**	0.9961813	0.6691524	**-4.6139E-06**	-5.1682E-05

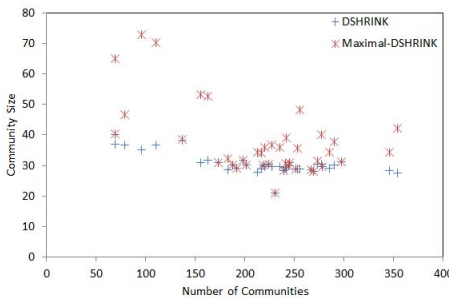


Fig. 1. Community Size

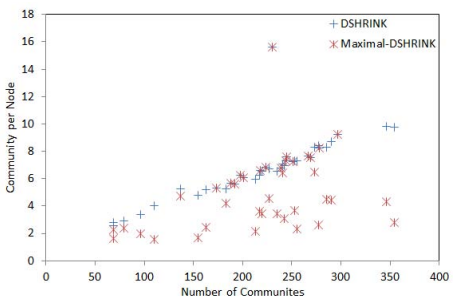


Fig. 2. Overlapping Rate

node is spanned into is lower. This means that each node participates in fewer overlapping communities using our algorithm as compared to DSHRINK.

6 Conclusions

In this paper, we presented a community detection algorithm originating from DSHRINK which detects overlapping communities in networks and expanded

it further to find maximal cliques. Our maximal approach has successfully improved the results generated from DSHRINK. Our algorithm gives results with consistently better modularity gain, which shows that the community set produced by Maximal-DSHRINK has better stability as compared to DSHRINK. Moreover, the community evaluation methods we applied have illustrated the effectiveness of our approach. One area of our future work is to optimize the cliques by decomposing the less stable cliques and ranking the communities in a particular order. Although the execution time needs not be maximized, further improvement in our algorithm's speed would be beneficial for community detection in massive networks.

References

1. Cuzzocrea, A., Folino, F.: Community evolution detection in time-evolving information networks. In: Guerrini, G. (ed.) EDBT/ICDT Workshops, pp. 93–96. ACM (2013)
2. Lin, W., Kong, X., Yu, P.S., Wu, Q., Jia, Y., Li, C.: Community detection in incomplete information networks. In: Proceedings of the 21st International Conference on World Wide Web, WWW 2012, pp. 341–350. ACM, New York (2012)
3. Xie, J., Kelley, S., Szymanski, B.K.: Overlapping community detection in networks: the state of the art and comparative study. ACM Computing Surveys 45(4) (2013)
4. Fortunato, S., Lancichinetti, A.: Community detection algorithms: a comparative analysis: invited presentation, extended abstract. In: Proceedings of the Fourth International ICST Conference on Performance Evaluation Methodologies and Tools, VALUETOOLS 2009, pp. 27:1–27:2 (2009)
5. Havemann, F., Heinz, M., Struck, A., Gläser, J.: Identification of overlapping communities and their hierarchy by locally calculating community-changing resolution levels. Journal of Statistical Mechanics: Theory and Experiment 2011(01) (2011)
6. Jin, D., Yang, B., Baquero, C., Liu, D., He, D., Liu, J.: A markov random walk under constraint for discovering overlapping communities in complex networks. Journal of Statistical Mechanics: Theory and Experiment 2011(05), P05031 (2011)
7. Ahn, Y.Y., Bagrow, J.P., Lehmann, S.: Link communities reveal multiscale complexity in networks. Nature 466(7307), 761–764 (2010)
8. Ding, F., Luo, Z., Shi, J., Fang, X.: Overlapping community detection by kernel-based fuzzy affinity propagation. In: 2010 2nd International Workshop on Intelligent Systems and Applications (ISA), pp. 1–4 (2010)
9. Gregory, S.: Finding overlapping communities in networks by label propagation. New Journal of Physics 12(10), 103018 (2010)
10. Xie, J., Szymanski, B.K.: Community detection using a neighborhood strength driven label propagation algorithm. In: Proceedings of the 2011 IEEE Network Science Workshop, NSW 2011, pp. 188–195. IEEE Computer Society, Washington, DC (2011)

Minimal Vertex Unique Labelled Subgraph Mining

Wen Yu, Frans Coenen, Michele Zito, and Subhieh El Salhi

Department of Computer Science, The University of Liverpool
Ashton Building, Ashton Street, Liverpool, L69 3BX, UK
{yuwen,coenen,michele,hsselsal}@liverpool.ac.uk

Abstract. This paper introduces the concept of Vertex Unique Labelled Subgraph Mining (VULSM), a specialised form of subgraph mining. A VULS is a subgraph defined by a set of edge labels that has a unique vertex labelling associated with it. A minimal VULS is then a VULS which is not a supergraph of any other VULS. The application considered in this paper, for evaluation purposes, is error prediction with respect to sheet metal forming. The minimum BFS Right-most Extension Unique Subgraph Mining (Min-BFS-REUSM) algorithm is introduced for identifying minimal VULS using a Breadth First Search(BFS) strategy.

Keywords: Data mining, Graph mining.

1 Introduction

This paper introduces the concept of Vertex Unique Labelled Subgraph Mining (VULSM), a form of graph mining. Given a subgraph g in some input graph G, if we consider only the structure and edge labelling there may be a number of different compatible vertex labelings with respect to G. A Vertex Unique Labelled Subgraph (VULS) is then a subgraph with a specific structure and edge labelling that has a unique vertex labelling associated with it. A minimal VULS is a VULS that does not contain any subgraphs that are also VULSs. This paper is directed at finding all minimal VULS in a single input graph. To this end the Minimal Breadth First Search Right-most Extension Unique Subgraph Mining (Min-BFS-REUSM) Algorithm is proposed. The distinction between VULSM and more traditional forms of subgraph mining [4,10,5,12] is that we are not interested in frequently occurring subgraphs but VULS. Broadly the proposed algorithm operates using a level-by-level approach. On each iteration a set of k-edge subgraphs that exist in G are identified (where k is also the iteration number). Any VULS identified in this set of subgraphs are stored and "removed" from G (thus G gets smaller and smaller). The process continues until G is empty or some user defined maximum size of VULS has been reached. subgraph generation is conducted using Right Most Extension [1] as popularised in the context of the gSpan transaction graph mining algorithm [11]. VULSM may be applied to various types of graph; in this paper we focus on undirected graphs.

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 317–326, 2013.

The application domain used to illustrate the work is error prediction in sheet metal forming. More specifically error prediction in Asymmetric Incremental Sheet Forming (AISF) [2,3,6,7,8]. An issue with sheet metal forming processes, such as AISF, is that distortions are introduced as a result of the application of the process. These distortions are non-uniform across the "shape" but tend to be related to local geometries. The idea is that the geometry of the piece to be manufactured can be represented as a grid, each grid centre point being defined using a X, Y, Z coordinate scheme. The entire grid can then be conceptualised as a graph such that each vertex represents a grid point and each vertex (except at the edges and corners) is connected to its four neighbours by a sequence of edges, which in turn can be labelled with "slope" values. Given an appropriate training set, each vertex can then be labelled with an error (distortion) value.

An example grid and corresponding graph are given in Figure 1. The grid comprises six grid squares. Each grid centre is defined by a X, Y, Z coordinate tuple (the number in each grid is Z value). Each grid centre point is associated with a vertex within the graph. The difference in Z value between each two neighbours in the grid is the "slope". The edges are labelled with these "slope" values. Each vertex will be labelled with an error values (e_1 to e_3 in the figure) describing the distortion experienced at that vertex as obtained from a "training set" (derived from "before and after" grid data). Identified VULS will describe local geometries each with a particular associated error pattern. This knowledge can then be used to predict errors in "unseen" grids so that some form of mitigating error correction can be applied.

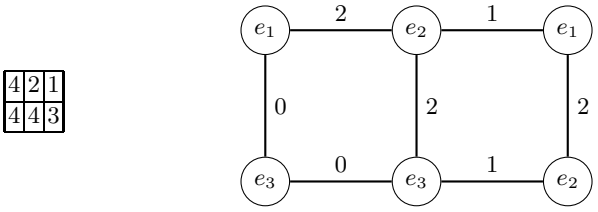


Fig. 1. Grid representation (left) and corresponding graph/lattice (right)

2 Formalism

This section presents a formal definition of the concept of a minimal VULS. A labelled graph G comprises a set of n vertices V, such that $V = \{v_1, v_2, \ldots, v_n\}$; and a set of m edges E, such that $E = \{e_1, e_2, \ldots, e_m\}$. The vertices are labelled according to a set of p vertex labels $L_V = \{l_{v_1}, l_{v_2}, \ldots, l_{v_p}\}$. The edges are labelled according to a set of q edge labels $L_E = \{l_{e_1}, l_{e_2}, \ldots, l_{e_q}\}$.

Alternatively we can think of a graph G as consisting of a set of k one-edge subgraphs: $G = \{P_1, P_2, \ldots, P_k\}$, where P_i is pair of vertices linked by an edge, thus $P_i = \langle v_a, v_b \rangle$ where $v_a, v_b \in V$). The size of a graph G ($|G|$) can thus be defined in terms of its one edge subgraphs, we refer to 1-edge, 2-edge and k-edge

subgraphs. For undirected graphs, the edge $\langle v_a, v_b \rangle$ is equivalent to $\langle v_b, v_a \rangle$. We use the notation $P_i.v_a$ and $P_i.v_b$ to indicate the vertices v_a and v_b associated with a particular vertex pair P_i. We indicate the sets of labels which might be associated with $P_i.v_a$ and $P_i.v_b$ using the notation $P_i.v_a.label$ and $P_i.v_b.label$ ($P_i.v_a.label$, $P_i.v_b.label \in L_V$). We indicate the edge label associated with P_i using the notaion $P_i.label$ ($P_i.label \in L_E$). We also assume that G is connected and labelled.

We use the same notation with respect to any subgraph G_{sub} of G ($G_{sub} \subseteq G$). Given some edge only labelled subgraph ($G_{subedgelab}$) of some fully labelled graph G ($G_{subedgelab} \subseteq G$) comprised of k edges, there may be many different vertex labelings that can be associated with this subgraph. We thus define a function, $getVertexLabels$, that returns the potential list of labels S that can be assigned to the vertices in $G_{subedgelab}$ according to G:

$$getVertexLabels(G_{subedgelab}) \rightarrow S$$

where $S = [[L_{v_{a1}}, L_{v_{b1}}], [L_{v_{a2}}, L_{v_{b2}}], \ldots, [L_{v_{ak}}, L_{v_{bk}}]]$ (where L_{v_i} is the set of labels associate with vertex v_i and $L_{v_i} \subseteq L_v$). Note that each element in S comprises two sub-sets of labels associated respectively with the start and end vertex for each edge in $G_{subedgelab}$, and that there is a one to one correspondence between each element (pair of label sets) in S with each element in $G_{subedgelab}$, hence they are both of the same size k (recall that k is the number of edges).

According to the above, the formal definition of the concept of a VULS is as follows. Given: (i) a k-edge edge labelled subgraph $G_{subedgelab} = \{P_1, P_2, \ldots, P_k\}$ ($G_{subedgelab} \subseteq G$), (ii) a list of labels that may be associated with the vertices in $G_{subedgelab}$, $S = [[L_{v_{a1}}, L_{v_{b1}}], [L_{v_{a2}}, L_{v_{b2}}], \ldots, [L_{v_{ak}}, L_{v_{bk}}]]$. If $\forall [L_i, L_j] \in S, |L_i| = 1, |L_j| = 1$ then $G_{subedgelab}$ is a k-edge VULS with respect to G. A VULS ϕ_i is minimal if there is no subgraph of ϕ_i that is also a VULS.

3 The Min-BFS-REUSM Algorithm

Min-BFS-REUSM algorithum is presented in this section. Recall that REUSM stands for Right-most Extension Unique Subgraph Mining. Right-most extension is the adopted iterative subgraph generation strategy. We use the prefix "Min" to indicate that this variation is for identifying all minimal VULS and "BFS" to indicate that it features a Breadth-First Search strategy.

The pseudo code for the Min-BFS-REUSM algorithum is presented in Algorithms 1 and 2. Algorithm 1 presents the high level control structure while Algorithm 2 the detail of determining whether a specific subgraph is a VULS or not. Considering Algorithm 1 first, the algorithm comprises one main procedure ($main$) and a sub-procedure ($genMinVULS$). The algorithm commences with an input graph G_{input} and a parameter max that defines the maximum size for a desired minimal VULS. If we do not limit the size of the searched-for VULSs the entire input graph may ultimately be identified as a minimal VULS which in the context of the target application will not be very useful. The output is a set of minimal VULS R. Note that all graphs are encoded using Minimal Depth

Algorithm 1. Min-BFS-REUSM

```
1: Input:
2: G_input = Input graph
3: max = Max subgraph size
4: Output:
5: R = Set of minimal VULS
6: Global variables:
7: G = G_input (Part of input graph not covered by minimal VULS)
8: coverage = 0
9: T_k = the set of k-edge subgraphs which are not VULS

10: procedure main(G_input, max)
11:     k = 1
12:     G_k = the set of k-edge subgraphs in G
13:     R = ∅
14:     while (k < max) do
15:         R = R ∪ genMinVULS(k, G_k)
16:         G_{k+1} = Set of (k + 1)-edge subgraphs in G (found by applying right most
    extension to each subgraph in T_k)
17:         k = k + 1
18:     end while
19: end procedure

20: procedure genMinVULS(k, G_k)
21:     T_k = ∅
22:     for all g ∈ G_k do
23:         if isaVULS(g, G_k) == true (Algorithm 2) then
24:             R = R ∪ g
25:             coverage = compute coverage using Equ. 1
26:             if coverage == 100% then
27:                 exit
28:             end if
29:             G = G − g
30:         else
31:             T_k = T_k ∪ g
32:         end if
33:     end for
34:     if T_k == ∅ then
35:         exit
36:     end if
37:     return R
38: end procedure
```

First Search (DFS) lexicographical ordering (as used in gSpan [11]). The global variable G (line 7 in Algorithm 1) is the part of G_{input} not covered by any of the identified minimal VULS so far, meanwhile, the global variable coverage (line 8) is employed to determine whether G_{input} is covered completely by the minimal VULS identified so far (if so the algorithm stops). The coverage is the percentage

of the number of vertices covered by the detected minimal VULS so far compared to the total number of vertices in the input graph G_{input} (Equation 1) (with respect to the sheet steel forming example application used as a focus for the work described in this paper, see Section 4, high coverage is desirable). The global variable T_k (line 9) is the set of k-edge non-VULS which will be extended further during the procedure to form $(k+1)$-edge candidate VULS.

$$coverage = \frac{num.\ vertices\ covered\ by\ VULS}{num.\ vertices\ in\ G_{input}} \times 100 \quad (1)$$

At the start of the procedure, G will be equal to G_{input} and coverage will be 0. We proceed in a breadth first manner starting with one-edge subgraphs ($k = 1$), then two edge subgraphs ($k = 2$), and so on. We continue in this manner until either: (i) $k = max$ or (ii) the coverage is equal to 100%. On each iteration the $genMinVULS$ procedure is called (line 15).

The $genMinVULS$ procedure takes as input the current graph size k (where k is the number of edges) and the set of k-edge subgraphs contained in the set G as pruned so far. The procedure returns the set of k-edge VULS. On each call the procedure $genMinVULS$ loops through the input set of k-edge subgraphs and (line 23) for each subgraph g determines whether it is a VULS or not by calling Algorithm 2 which is described in detail below. If g is a VULS it is added to the set R (line 24). We then (line 25) calculate the coverage so far, if this has reached 100% we have found the complete set of minimal VULS and we exit (line 27). Note that if coverage is equal to 100% the input set G, as pruned so far, will be empty. Otherwise, if the coverage is not 100%, we continue processing and (line 29) remove g from the global set G. If g is not a VULS we add it to T_k (line 31), T_k is the set of k-edge subgraphs which we will eventually be extended to form G_{k+1}, the set of $(k+1)$-edge subgraphs, ready for the next level of processing. Eventually all g in G_k will have been processed. If, at this stage T_k is empty there will be no more subgraphs that can be generated and the process will exit (line 35). Otherwise control will return to the *main* procedure and the set of $(k+1)$-edge subgraphs will be generated from T_k (the set of k-edge subgraphs that have not been found to be VULS) using a right most extension technique coupled with isomorphism checking to establish which $(k+1)$-edge subgraphs are contained in G as processed so far (line 16). This part of the algorithm is not presented here because it is similar to that found in traditional subgraph mining algorithms, for example gSpan [11]. The generated minimal VULS set R is then returned (line 37) back to the main process ready for the next iteration (unless the maximum value for k has been reached).

Algorithm 2 presents the pseudo code for identifying whether a given subgraph g is a VULS or not with respect to the current set of k-edge subgraphs G_k from which g has been removed. The algorithm returns $true$ if g is a VULS and $false$ otherwise. The process commences (line 8) by generating the potential list of vertex labels S that can be matched to g according to the content of G_k (see previous section for detail). The list S is then processed and tested. If there exists a vertex pair whose possible labelling is not unique (has more than one possible

labelling that can be associated with it) g is not a VULS and the procedure returns $false$, otherwise g is a VULS and the procedure returns $true$.

Algorithm 2. Identify VULS

```
1: Input:
2: g = a single k-edge subgraph (potential VULS)
3: G_k = a set of k-edge subgraphs to be compared with g
4: Output:
5: true if g is a VULS, false otherwise

6: procedure isaVULS(g, G_k)
7:     isVULS = true
8:     S = the list of potential vertex labels that may be assigned to g
9:     for all [L_i, L_j] ∈ S do
10:        if either |L_i| ≠ 1 or |L_j| ≠ 1 then
11:            isVULS = false
12:            break
13:        end if
14:    end for
15:    return isVULS
16: end procedure
```

Thus, as the process proceeds, the input graph G will be continuously pruned with respect to identified VULS. As a result G can become disconnected, any disconnected sub graph of size less than the current value of k cannot therefore contain any k-edge VULS. Although not shown in Algorithm 1 any disconnected subgraphs of size less than k can be discounted therefore speeding up the overall process.

4 Experiments and Performance Study

This section describes the evaluation of the proposed Min-BFS-REUSM algorithm. For experimental purposes the algorithm was implemented using the JAVA programming language; experiments were using a 2.7 GHz Intel Core i5 with 4 GB 1333 MHz DDR3 memory, running OS X 10.8.1 (12B19). The reported experiments were all conducted using real data taken from an AISF sheet metal forming application described in the introduction to this paper, more specifically the fabrication of flat topped pyramid shapes made out of sheet steel. This shape was chosen as it is frequently used as a benchmark shape for conducting experiments in the context of AISF [9].

4.1 Experimental Performance Measurement

Three performance measures were used to analyse the effectiveness of the proposed Min-BFS-REUSM algorithm : (i) run time (seconds), (ii) total number of VULS identified, (iii) coverage (%). With respect to the sheet steel forming example application high coverage values were desirable.

4.2 Data Sets

The data sets used for the evaluation consisted of before and after "coordinate clouds"; the first generated by a CAD system and the second obtained using optical measuring techniques after application of an AISF process.As noted in the introduction, each vertex was labelled with an error value while the edges were labelled according to the absolute difference in z of the two end vertices (the "slope"). Furthermore, the vertex and edge labels were discretised so that they were represented by nominal values (otherwise every edge pair was likely to be unique). In total ten data sets (graphs) were generated, numbered AISF1 to AISF10, using three different grid sizes (6×6, 10×10 and 21×21 which correspond to 36, 100, 441 as number of vertices translated to resulting graph respectively), and different numbers of edge and vertex labels (from 2 to 4 and 2 to 3 respectively). Some statistics concerning the data sets are presented in Table 1. Thus the graph edge labels describe the geometry of the shape to be manufactured while each vertex label describes the error occurring at that location between the desired shape and the actual shape produced. Any discovered VULS will then describe a particular geometry with a particular error pattern associated with it.

Table 1. Summary of AISF graph sets

graph set	# vertices	# edge labels	# vertex labels	graph set	# vertices	# edge labels	# vertex labels
AISF1	36	3	2	AISF6	100	2	3
AISF2	36	2	2	AISF7	441	3	2
AISF3	36	2	3	AISF8	441	2	2
AISF4	100	3	2	AISF9	441	2	3
AISF5	100	2	2	AISF10	441	4	2

4.3 Run Time Analysis

The results obtained for the run-time experiments are presented in Table 2 with respect to a range of *max* values. From the table the following can be noted. Firstly, as might be expected, the run time increases as the value of *max* increases, although the run time does not increase dramatically. Similarly, again as might be expected, it takes longer to process the larger graph sets than the smaller graph sets. As can be confirmed by inspection of Table 4, with respect to AISF1, AISF2, AISF3 and AISF5, Min-BFS-REUSM stops at $k = 4$, 6, 5 and 6 respectively. This is because at these values of k the coverage reaches 100% and the process is complete. Thus, in these cases the timings are almost the same for values of *max* greater than the k value when maximum coverage was reached. The reason for OME (Out of Memory Error) in the case of AISF8 and AISF9 is discussed in sub-section 4.5 below.

Table 2. Run time (seconds) comparison for a range of *max* values

graph set	*max* Value						graph set	*max* Value					
	3	4	5	6	7	8		3	4	5	6	7	8
AISF1	0.07	0.12	0.13	0.13	0.13	0.13	AISF6	0.17	0.29	0.51	0.68	0.77	0.97
AISF2	0.08	0.10	0.16	0.17	0.16	0.21	AISF7	0.33	0.42	0.61	0.65	0.73	0.87
AISF3	0.09	0.26	0.34	0.37	0.39	0.41	AISF8	0.31	0.41	0.70	0.81	1.69	OME
AISF4	0.16	0.19	0.27	0.29	0.32	0.47	AISF9	0.33	0.56	0.92	1.34	2.16	OME
AISF5	0.14	0.23	0.23	0.28	0.35	0.35	AISF10	0.32	0.45	0.60	0.74	0.88	1.43

4.4 Total Number of Discovered Minimal VULS

Table 3 presents the total number of discovered minimal VULS using Min-BFS-RESUM. From Table 3 it can be observed, as might be anticipated, that as the *max* value increases the total number of discovered minimal VULS increases. As already noted above, for AISF1, AISF2, AISF3 and AISF5, the coverage reaches 100% when $k = 4$, 6, 5 and 6 respectively at which point the algorithm stops. This is why in these cases the total number of identified VULS remains static at 24, 37, 69 and 79 for *max* values in excess of k.

Table 3. Total number of minimal VULS discovered for a range of *max* values

graph set	*max* Value						graph set	*max* Value					
	3	4	5	6	7	8		3	4	5	6	7	8
AISF1	9	24	24	24	24	24	AISF6	6	22	94	334	356	395
AISF2	7	13	29	37	37	37	AISF7	11	27	100	226	318	369
AISF3	1	11	69	69	69	69	AISF8	4	22	103	267	717	OME
AISF4	5	16	48	75	93	104	AISF9	4	26	125	272	401	OME
AISF5	13	32	51	79	79	79	AISF10	26	70	96	136	193	279

4.5 Coverage

A comparison of coverage is presented in Table 4. As was to be expected, as *max* increases the coverage rate increases. It should be noted that in some cases the coverage will not reach 100% because no more minimal VULS can be discovered. It is interesting to note, with respect to Table 4 that for AISF1 to AISF7 100% coverage is reached (more or less). It is anticipated that if some efficiency gains can be realised 100% coverage would also be achieved for AISF8 to AISF10. This is an excellent result with respect to AISF sheet metal forming application used as a focus with respect to the evaluation presented in this section.

In the context of AISF8 and AISF9 the OME occurs because: (i) these graphs are much larger than the graphs for AISF1 to AISF6, and (ii) fewer VULS were discovered during early stages of the Min-BFS-REUSM process. AISF7, AISF8, AISF9 and AISF10 were generated from the same 21×21 grid data set as described in subsection 4.2 above, however discretised in different manners using different numbers of edge and vertex labels (as shown in table 1). As k

Table 4. Coverage (%) for a range of *max* values

graph set	*max* Value						graph set	*max* Value					
	3	4	5	6	7	8		3	4	5	6	7	8
AISF1	52.8	100.0	100.0	100.0	100.0	100.0	AISF6	19.0	40.0	67.0	85.0	99.0	99.0
AISF2	86.1	91.7	97.2	100.0	100.0	100.0	AISF7	64.2	92.1	93.0	99.6	99.8	99.8
AISF3	13.9	47.2	100.0	100.0	100.0	100.0	AISF8	24.9	70.5	71.7	71.7	71.7	OME
AISF4	59.0	87.0	98.0	99.0	99.0	99.0	AISF9	7.0	27.7	58.7	69.8	76.0	OME
AISF5	80.0	81.0	81.0	100.0	100.0	100.0	AISF10	76.4	86.9	87.3	87.3	87.3	88.2

increases the rate at which the copy of the input graph G is pruned is slower for AISF8 and AISF9 than for AISF7 and AISF10 (as can be observed from Table 4). For example for $max = 3$ many fewer $k = 1$, 2 and 3-edge minimal VULS are discovered with respect to AISF8 and AISF9 than for AISF7 and AISF10. Thus for these two data sets the size of G is not significantly reduced during the early stages of Min-BFS-REUSM and hence the memory error occurs. This observation can be validated with reference to Table 5 which shows the number of vertices and edges contained in the AISF7, AISF8, AISF9 and AISF10 data sets before, during and after the application of the process of Min-BFS-REUSM when $max = 8$. From the table it can clearly be seen that many more edges and vertices are removed from G with respect to AISF7 and AISF10 than with respect to AISF8 and AISF9.

Table 5. Summary of AISF7 to AISF10 graph sets before, during and after application of the Min-BFS-REUSM process when $max = 8$

graph set	Prior to start		On completion		Removed		graph set	Prior to start		On completion		Removed	
	# verts.	# edges	# verts.	# edges	# verts.	# edges		# verts.	# edges	# verts.	# edges	# verts.	# edges
AISF7	441	840	291	214	150	626	AISF9	441	840	355	392	86	448
AISF8	441	840	372	460	69	380	AISF10	441	840	335	341	106	499

5 Conclusions and Further Study

In this paper we have introduced the concept of VULSM which, as illustrated, has application with respect to error prediction in sheet metal forming. We have also introduced the Min-BFS-RESUM algorithm an algorithm for identifying all minimal VULS in a given input graph. The significance of finding minimal VULS is that they can be used to build larger VULS, it is thus computationally efficient to find only minimal VULS than finding all VULS. The reported experimental results also indicate that our algorithm can successfully identify all minimal VULS in reasonable time and with (in some cases) excellent coverage (an important requirement in the context of the AISF sheet metal forming application used as a focus for the work). Having established a "proof of concept"

there are many interesting research problems related to VULSM that can now be pursued. For instance Min-BFS-REUSM can be modified to identify VULS in directed graphs or trees. The concept of Frequent VULSM (FVULSM) may be realised by combining the techniques of FSM (Frequent Subgraph Mining) and VULSM.

Acknowledgements. The research leading to the results presented in this paper has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement number 266208.

References

1. Asai, T., Abe, K., Kawasoe, S., Sakamoto, H., Arikawa, S.: Efficient substructure discovery from large semi-structured data. In: Proc. 2002 SIAM Int.Conf. Data Mining (2002)
2. Cafuta, G., Mole, N., Tok, B.: An enhanced displacement adjustment method: Springback and thinning compensation. Materials and Design 40, 476–487 (2012)
3. Firat, M., Kaftanoglu, B., Eser, O.: Sheet metal forming analyses with an emphasis on the springback deformation. Journal of Materials Processing Technology 196(1-3), 135–148 (2008)
4. Han, J., Cheng, H., Xin, D., Yan, X.: Frequent pattern mining: Current status and future directions. Data Mining and Knowledge Discovery 15(1), 55–86 (2007)
5. Huan, J., Wang, W., Prins, J., Yang, J.: SPIN: mining maximal frequent subgraphs from graph databases. In: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 581–586 (2004)
6. Jeswiet, J., Micari, F., Hirt, G., Bramley, A., Allwood, J.D.: Asymmetric single point incremental forming of sheet metal. CIRP Annals Manufacturing Technology 54(2), 88–114 (2005)
7. Liu, W., Liang, Z., Huang, T., Chen, Y., Lian, J.: Process optimal ccontrol of sheet metal forming springback based on evolutionary strategy. In: 7th World Congress on Intelligent Control and Automation, WCICA 2008, pp. 7940–7945 (June 2008)
8. Nasrollahi, V., Arezoo, B.: Prediction of springback in sheet metal components with holes on the bending area, using experiments, finite element and neural networks. Materials and Design 36, 331–336 (2012)
9. Salhi, S., Coenen, F., Dixon, C., Khan, M.: Identification of correlations between 3d surfaces using data mining techniques: Predicting springback in sheet metal forming. In: AI 2012, pp. 391–404. Springer, Cambridge (2012)
10. Yan, X., Han, J.: Close Graph: mining closed frequent graph patterns. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 286–295 (2003)
11. Yan, X., Han, J.: gSpan: Graph-based substructure pattern mining. In: Proceedings of the 2002 International Conference on Data Mining, p. 721 (2002)
12. Zhu, F., Yan, X., Han, J., Yu, P.S.: gPrune: A constraint pushing framework for graph pattern mining. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 388–400. Springer, Heidelberg (2007)

A Hybrid Graph-Based Method for Concept Rule Discovery

Alev Mutlu and Pinar Karagoz

Department of Computer Engineering,
Middle East Technical University, Ankara, Turkey
{mutlu,karagoz}@ceng.metu.edu.tr
http://www.ceng.metu.edu.tr

Abstract. In this paper we present a hybrid graph-based concept discovery method. Concept rule discovery aims at finding the definition of a specific concept in terms of relations involving background knowledge. The proposed method is a hybrid approach that combines path finding-based and substructure-based approaches. It distinguishes from other state of the art graph-based approaches in such a way that the data is initially stored in a relational database, and the concept descriptors are induced while the graph is constructed. In this approach, in addition to being the representation framework, the graph structure is also utilized to guide the concept induction process. A set of experiments is conducted on data sets that belong to different learning problems. The results show that the proposed approach has promising results in comparison to state of the art methods in terms of running time and accuracy.

Keywords: Concept Discovery, Graph, Hybrid, Path, Substructure.

1 Introduction

Concept is defined as a set of frequent patterns that are embedded in the features of the concept instances in the form of relations among objects [1]. Concept discovery is the problem of learning definitions of a specific relation, called *target relation*, in terms of other relations provided as *background knowledge* [2].

The concept discovery problem has extensively been studied by the ILP [3] community with successful applications in several domains such as bioinformatics [4], engineering [5], and environmental sciences [6]. Among several problems in concept discovery [7], a common problem faced by ILP-based concept discovery systems is the so called *local plateau problem* [8]. In such cases classical operators of ILP that refine concept descriptors by one literal at a time are insufficient to improve the quality of the concept descriptors and the systems perform a blind search. To the best of our knowledge, graph-based approaches were first introduced to the concept discovery problem to solve this issue by Richards and Mooney [9]. In their approach the refinement operators of ILP are upgraded to refine the concept descriptors by a set of literals such that arguments of the literals form a path.

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 327–338, 2013.

Graph-based concept discovery methods can be classified into two main categories: substructure-based approaches and path finding-based approaches. Substructure-based approaches [10] rely on the idea that if there exists a concept in a graph then it should appear as frequent substructures. Such systems employ expensive algorithms such as graph isomorphism to find similar substructures. The motivation behind the path finding-based approaches [11] is the assumption that a concept should appear as frequent, finite length paths that connect some arguments of positive target instances. Such approaches need to employ advanced indexing mechanisms [12] to keep track of the paths.

In this work we propose a hybrid framework for graph-based concept discovery. We employ directed, labeled graph where nodes represent arguments of the relations, and edges connect those nodes that form some background facts. The proposed approach inputs the data in relational format, generates a graph, extracts concept definitions, and outputs concept descriptors in the form of conjunctions of relations. Similar to substructure-based approaches, it groups similar arguments and represents them as a single node. Similar to path finding-based approaches, it infers the concept descriptors by finding paths that connect arguments of the target instances. Different than substructure-based approaches, the proposed approach does not employ graph isomorphism algorithms, which are known to be NP, to group similar nodes but rather constructs the graph in a compressed form by executing SQL queries on the input data. Different than path finding-based approaches, the proposed method does not search for paths within the graph, but infers such paths while constructing the graph. The proposed method does not need to employ advanced indexing mechanism to store paths either, but keeps such information within the nodes. The current implementation of the proposed method is limited to binary Head Output Connected (HOC) class of learning problems [13]. HOC is the class of predicates that have at least one output variable in the target concept.

The rest of the paper is organized as follows: In Section 2 we provide a brief background on graph-based concept discovery and HOC class of problems. In Section 3 we present the proposed method and list its distinguishing properties. In Section 4, we present the experimental results. Section 5 concludes the paper.

2 Background

Graph-based concept discovery systems can be classified as substructure-based approaches and path finding-based approaches. Systems that fall into the first group repeatedly look for similar substructures that appear within the graph, and replace these substructures with a single node. Such systems assume that the merged structures represent the concepts. Path fining-based approaches seek for paths that connect arguments of positive target instances and output such paths as concept descriptors.

SubdueCL [14] is a substructure-based concept discovery system. In SubdueCL data is represented as a directed, labeled graph where nodes store arguments of the facts, and labeled edges are the relation names connecting the arguments of the facts. In SubdueCL, substructures are evaluated according to the number of positive target instances and negative target instances they explain.

Graph Based Induction (GBI) [15] is another concept learning system based on substructure discovery. It employs colored digraph as the representation framework where colors attached to the nodes represent the attributes of the facts. GBI examines each connected pair of nodes, and merges the frequent typical ones. The final merged substructures are labeled as concepts. GBI is further extended by Matsuda et al. [16] to handle self loops.

Relational Pathfinding [9] is one of the earliest path finding-based approaches which is proposed to overcome the local plateau problem of ILP-based concept discovery systems. In Relational Pathfinding, similar to SubdueCL, nodes are fact arguments. Edges are labeled after the relation names and connect such pairs of nodes that they form a fact. It employs bidirectional breadth first search to discover the concept descriptors.

Mode Directed Path Finding [17] extends Relational Pathfinding algorithm for saturated bottom clauses. In Mode Directed Path Finding, the bottom clauses are transformed into hypergraphs and concept descriptors are induced by traversing the hypergraph. The approach also employs mode declarations to guide the search and to avoid generation of illegal concept descriptors.

Relational Paths Based Learning (RPBL) [11] is yet an other concept discovery system based on path finding. In RPBL, nodes represent binary facts, and edges connect nodes that share some arguments in common. Extensions to RPBL are proposed to deal with recursive concept definition learning, and to incorporate domain theories in [18]. To learn recursive concept descriptors, extended version of RPBL treats the target instances also as background knowledge. To apply domain theories into the learning process, they extend the graph in accordance with domain theories, i.e. by connecting nodes that hold with the domain theories.

Head Output Connected (HOC) is a class of predicates that have at least one output variable in the target concept. Such learning problems include list manipulation and software verification problems. Although HOC is, seemingly, a special class of learning problem, it has intensively been studied within the path finding-based concept learning community [9,13,17].

The proposed approach is similar to substructure-based approaches as it works on a compressed graph. Different than such studies, the graph is not compressed to find concept descriptors but to provide a compact representation of the data. Similar to path finding-based approaches it represents the concept descriptors as a path that connects arguments of some target instances. Different than such studies it does not look for paths on a already built graph, but discovers such paths while constructing the graph.

3 The Proposed Approach

In this section we firstly introduce our data representation model. Next, we present the hybrid graph-based concept discovery process, and lastly we list the distinguishing properties of the proposed method from state of the art methods.

We employ the *elti* data set given in Table 1 as a running example in this section. In the data set, predicate e stands for the *elti* relation, h stands for the *husband* relation, w stands for the *wife* relation, and b stands for the *brother* relation. All arguments are of type *person*. The *elti* relation is the concept to be learned, and *husband*, *brother*, and *wife* are the background relations. *elti* is a kinship relation in Turkish. Two people are called *elti* if they are wifes of two brothers.

Table 1. The *elti* data set

Target Instances	**Backgorund data**		
t_1 :e(cemile, ayse)	b_1 : b(mehmet, ismail)	b_2 : b(mehmet, ali)	b_3 : b(sadullah, yildirim)
t_2 :e(cemile, ayten)	b_4 : h(sadullah, nalan)	b_5 : h(ali, ayse)	b_6 : h(yildirim, bedriye)
t_3 :e(nalan, bedriye)	b_7 : h(mehmet, cemile)	b_8 : h(ismail, ayten)	b_9 : w(bedriye, yildirim)
	b_{10} : w(ayten, ismail)	b_{11} : w(ayse, ali)	b_{12} : w(nalan, sadullah)
	b_{13} : w(cemile, mehmet)		

3.1 Data Representation

We employ directed, labeled, acyclic graph as the representation framework. Different from conventional graph structures, in our data representation, a node holds a set of constants from the background knowledge and an edge connects a pair of such nodes. Edges are labeled with the background relation names.

In our model we define three types of nodes: the source node, s, the target node, t, and intermediate nodes, v. Note that in our study target instances are in binary form, node s stores the the first argument of the target instances and node t stores the second argument of the target instances.

3.2 The Proposed Method

The proposed approach inputs a set of target instances, a set of background data; minimum support, minimum confidence, and maximum rule length parameters. The target instances and the background data are initially stored in a database.

In the proposed approach concept descriptors are evaluated and pruned based on their support and confidence values. Support of a concept descriptor is the ratio of the number of target instances captured by the concept descriptor over total number of the target instances. Confidence refers to the number of target instances that correctly hold for the rule divided by the number of instances that hold for the body of the rule. Maximum rule length parameter limits the length of the concept descriptors. Concept descriptors that do not qualify the minimum support value are pruned. On the other hand, concept descriptors that

qualify the minimum support but fail with the minimum confidence threshold are further refined in the next iteration. If they qualify the both thresholds, then they are added to the solution set.

The method considers all of the uncovered target instances together so that it avoids the target instance ordering problem [19]. The grouping of the facts also allows embodying the generalization step of concept learning process into the graph-based concept induction process - on the contrary to many path-finding based systems where the generalization step is performed as a post-processing step. The proposed approach utilizes absorption operator of inverse resolution for generalization of concept instances.

Figure 1 illustrates the execution of the proposed method for the *elti* data set. The proposed method is composed of seven main components:

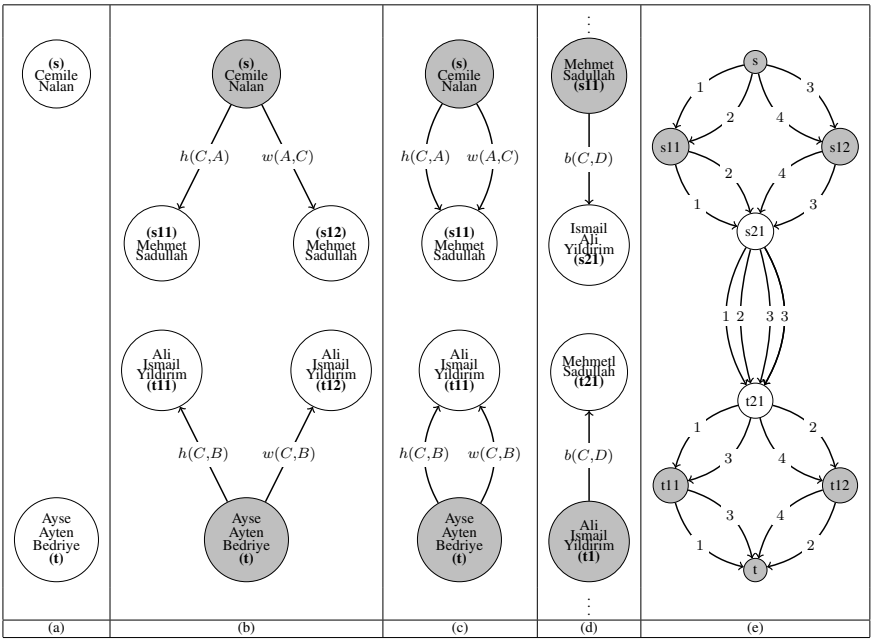


Fig. 1. Execution of the Algorithm

1. **Initialization:** In this step the source and the target nodes namely s and t, respectively, are created. This very initial graph is a disconnected graph with two nodes where vertex s stores the values of the first argument of the target instances, and the vertex t stores the values of the second argument of the target relations, Figure 1-a.
2. **Expansion:** In this step graph is expanded by adding nodes that contain facts related to the constants stored in tail nodes. To find such facts, constants in the tail nodes are sought in relations that do not appear as labels on their incoming edges as such expansions will create loops. As an example

in Figure 1-c we do not extend vertex *s11* with the *husband* and *wife* relations but only with the *brother* relation. Each newly node stores the path from the root node to itself. To find the content of *s21* the following query is executed:

SELECT h.arg2 from husband h WHERE h.arg1 = "Mehmet" or h.arg1 = "Sadullah"

3. **Merge:** In this step nodes that store the same content are represented as a single node and edges are rearranged accordingly. To realize this, content of nodes are stored in hash tables, and a hash based search is employed to find nodes that have the same content. The graph in Figure 1-b is changed into 1-c once this step is executed.
4. **Evaluation and pruning:** In this step support and confidence values of the current concept descriptors are calculated. To calculate these values, current concept descriptors are translated into SQL queries and these queries are run against the database. Please note that these concept descriptors are indeed the paths that connect the tail nodes to the source node, and this information is stored within each tail node. In Table 2 we provide support and confidence queries for *elti(A, B):- husband(B, A)*.

Table 2. Example Support and Confidence Queries

Support = $\frac{C1}{C2}$	C1:SELECT COUNT DISTINCT (e.arg1, e.arg2) FROM elti e, husband h WHERE e.arg1 = h.arg2 AND e.arg2 = h.arg1
	C2:SELECT COUNT DISTINCT(e.arg1, e.arg2) FROM elti e;
Confidence = $\frac{C3}{C4}$	C3:SELECT COUNT DISTINCT(h.arg1, h.arg2) FROM elti e, husband h WHERE e.arg1 = h.arg2 AND e.arg2 = h.arg1
	C4:SELECT COUNT DISTINCT(h.arg1, h.arg2) FROM husband h

5. **Check for intersection:** To find intersections, tail nodes and their heads of the subgraph with the root node *s* are compared to the tail nodes and their heads of the subgraph with the root node *t*. An intersection in a comparison means that such pairs connect the two subgraphs, hence form a path from one argument of the target relation to the other. To find the paths that connect nodes *s* and *t*, the path values of the intersecting nodes store are merged. In Figure 1-d such an intersection exists and the formed paths are:

   p1: e(A,B), w(A,C), b(C,D), b(C,D), h(C,B), e(A,B)
   p2: e(A,B), h(C,A), b(C,D), b(C,D), h(C, B), e(A,B)
   p3: e(A,B), w(A,C), b(C,D), b(C,D), w(C,B), e(A,B)
   p4: e(A,B), h(C,A), b(C,D), b(C,D), w(C,B), e(A,B)

   In the intersection step we allow partial matches, i. e. two tail nodes need not necessarily contain the same constants. By this way we can discover concept descriptors in noisy data. Similar mechanisms are also employed in substructure-based approaches, i. e. matches with partial distortions are allowed [20]. In this step the concept discovery process terminates if the length of the current paths is larger than $\lceil maximum\_rule\_length/2 \rceil$ and no intersection is found.
6. **Update path variables:** In this step the variables of concept descriptors are updated to be consistent with the constant values they hold for.

For this purpose we build a new substitution map based on the constants. As an example in partial path *e(A,B), w(A,C), b(C,D)* $D = \{$*Ismail, Ali, Yildirim*$\}$, while in the same constant set is represented with C in the partial path *b(C,D), h(C,B), e(A,B)*. To build the final concept descriptors repeating literals are removed from the paths, and variable names are updated to be consistent with each other. The candidate solution set will be:

c1: e(A,B):-h(C,A), b(C,D), h(D,B)
c2: e(A,B):-h(C,A), b(C,D), w(B,D)
c3: e(A,B):-w(A,C), b(C,D), h(D,B)
c4: e(A,B):-w(A,C), b(C,D), w(B,D)

In Figure 1-e we show the traversals of the solution clauses. Once final candidate solution set is constructed, the candidate solution clauses are pruned based on their support and confidence values. The proposed method can discover multiple concept descriptors at a time, to choose the best concept descriptor f-metric [21] is employed. In this step the concept discovery process terminates if the newly discovered concept descriptors have already been discovered in the previous iterations.

7. **Covering:** In this step target instances explained by the solution clauses are marked as *covered*. If the number of the remaining uncovered target instances is below $minimum\_support \times \#target\_instances$ the concept induction process terminates, else restarts with the initialization step.

3.3 Distinguishing Features of the Proposed Approach

The proposed method distinguishes from such studies in the following ways:

(a) While inducing the concept descriptors RPBL and Relational Pathfinding approaches select a target instance from the uncovered target instance set and build concept descriptors accordingly. The selection order of the target instances may change the resulting hypothesis set. The proposed method considers all of the target instances, hence avoids such problems and improves rule quality.
(b) Path finding-based methods proposed in [13,17] require mode declarations. Although mode declarations provide strong pruning mechanisms, they are generally hard to define. The proposed method does not require any mode declarations, but instead utilizes the graph structure to avoid creation of candidate concept descriptors with unbounded variables.
(c) The proposed method also differs from the state of the art path finding-based methods in terms of graph structure. It groups similar arguments and represents them as a single node instead of representing each argument or fact as a single node. Such a representation allows easier user interpretability.
(d) The proposed method differs from the path finding-based approaches as it does not need a post generalization step to form the final concept descriptors as it induces the concept descriptors in their most generalized form.

4 Experiments

In this section, we firstly describe the data sets employed in the experiments, and then discuss the performance of the proposed approach. The experimental results include discussion on the learning capability, running time of the proposed approach, and the quality of the induced concept descriptors. The results are presented in comparison with state of the art ILP-based concept discovery system CRIS [19] and path finding-based concept discovery system RPBL. CRIS uses a non-graph based approach. However, its mechanism is similar to the proposed approach in the sense that they both systems use support and confidence based pruning, coverage based iteration mechanism and similar way of rule generalization. CRIS and the proposed approach are run on the same environment, while results of RPBL are retrieved from the literature [11,18].

4.1 Data Sets

We conducted experiments on four different data sets. Although these data sets are similar in nature, they are attractive in the concept learning problem as each belongs to a different learning problem. *Elti* is a real world kinship data set which contains transitive relations in the target concept. *Dunur* is a real world kinship data set where facts indirectly related to the target instances exist. The *Same-Gen* data set is a real world kinship data set that contains recursive relations. *Family* [22] data set is highly relational data set that contains different kinship relations.

In Table 3 we list properties of the data sets and the experimental settings. We set the maximum rule length, minimum confidence, and minimum support parameters according to the reference papers [18,19] that we compare our method against.

Table 3. Data set properties and the experimental settings

Data Set	# Relations	# Facts	Min. Sup.	Min. Conf.	Max. Length
Elti	9	224	0.3	0.7	3
Dunur	9	224	0.3	0.7	3
Same-Gen	2	408	0.3	0.6	3
Family	12	744	0.1	0.7	9

4.2 The Experimental Results

To analyze the behavior of the proposed method on data sets that belong to different types of learning problems we conducted experiments on the *Elti*, *Dunur*, and *Same-Gen* data sets. The proposed method was successful at learning the target relations for those data sets. The proposed method found exactly the same set of the solution clauses for the *Dunur* and *Same-Gen* as CRIS did. For the *Elti* data set, the proposed method induced a subset of the concept descriptors found by CRIS. When we analyzed the missing concept descriptors, we realized

that the missing concept descriptors are, indeed, semantically identical to some of those induced but have different literal ordering. This result shows that the proposed approach is successful at avoiding induction of semantically similar concept descriptors that are different in presentation.

In order to analyze the quality of the induced concept descriptors, and the running time of the proposed approach we conducted further experiments. Below we discuss these results and compare them to CRIS and RPBL.

As the proposed approach discovered concept descriptors that cover the same set of target instances with CRIS for the *Dunur*, *Elti*, and *Same-Gen*, their accuracy and coverage values are identical. In Table 4 we list the coverage and accuracy values for those data sets. In order to find the number of false positive and false negative instances, data sets are extended with their duals under the Close World Assumption. These results are listed in Table 4.

Table 4. Coverage and Accuracy Results

Data Set	Coverage	Accuracy
Elti	1.0	1.0
Dunur	1.0	1.0
Same-Gen	0.84	1.0

In Table 5 we compare the proposed approach to RPBL. In this experiment we set the minimum support to 0.1, minimum confidence to 0.7 and maximum rule length according to the length of the longest path found by RPBL. The third column of Table 5 lists the number of the solution clauses induced, the fourth column lists the average length of concept descriptors. The last two columns list the precision and recall values.

Table 5. Family data set results

T. R.	**Alg.**	**#Cl.**	**Len.**	**Pre.**	**Recall**	**T. R.**	**Alg.**	**#Cl.**	**Len.**	**Pre.**	**Recall**
Brother	Proposed	2	2.5	100	100	Uncle	Proposed	9	2	100	100
	RPBL	2	6	95	96		RPBL	2	6	100	100
Niece	Proposed	7	2	100	98	Aunt	Proposed	10	2	100	100
	RPBL	2	6	95	96		RPBL	2	6	100	100
Nephew	Proposed	7	2	100	98	Son	Proposed	10	2	100	100
	RPBL	2	6	95	96		RPBL	2	6	100	100
Mother	Proposed	7	2	100	98	Father	Proposed	10	2	100	100
	RPBL	2	6	95	96		RPBL	2	6	100	100
Daughter	Proposed	7	2	100	98	Sister	Proposed	10	2	100	100
	RPBL	2	6	95	96		RPBL	2	6	100	100
Wife	Proposed	7	2	100	98	Husband	Proposed	10	2	100	100
	RPBL	2	6	95	96		RPBL	2	6	100	100

As the experimental results show the proposed approach finds larger number of concept descriptors with less number of literals. Concept descriptors with

many literals may be hard to interpret and such concept descriptors are subject to the overfitting problem. The proposed approach is capable of inducing much simpler concept descriptors with almost the same precision and recall values. The proposed approach missed to define 1 target instance for the husband, nephew, and wife relations; 2 target instances for the niece and daughter relations; and 4 target instances for the son relation. Indeed these misses are due to the minimum support and confidence values. For example, for the wife relation, there is 1 uncovered target instance and as 1 / 25 is less than the minimum support value, the proposed approach did not attempt to find a concept descriptor to explain the uncovered target instance.

Another performance dimension for concept discovery systems is their running time. Such systems usually build large search spaces and suffer from scalability and efficiency issues. In Table 6 we report the number of the evaluation queries executed by the proposed approach and CRIS during the concept induction process. In the same table we also list their running times. The experimental results show that the proposed approach executes less number of queries. Although CRIS has powerful pruning strategies, it generalizes the concept descriptors in all possible ways which results in a large search space. On the other hand, the proposed approach limits its search space by possible graph expansions only and builds a relatively small search space compared to CRIS.

Table 6. Comparison on running times and the number of queries executed

Data Set	Running Time(in seconds)		Number of Queries	
	The Proposed Approach	CRIS	The Proposed Approach	CRIS
Elti	0.51 (Speedup = 68)	35	2118 (Drop = 43%)	3571
Dunur	0.11 (Speedup = 236)	26	3192 (Drop = 15%)	3777
Same Generation	0.76 (Speedup = 15)	12	222 (Drop = 77%)	996

In Table 6 we also report the speedup and the drop in the number of executed queries. Although the proposed method achieves speedup in magnitudes and a significant drop in the number of evaluation queries, we could not find any correlation between these values. Both systems involve execution of SQL queries and execution time of each query may be affected by several factors such as the number of concurrently executing queries at that time.

Average concept descriptor learning time for RPBL is given around 0.02 seconds [18], and it is around 1 second for the proposed approach. Although we believe that comparison of the running times is not correct as the programs are run on different platforms, we provide this information to give a clue on the execution time of the systems.

To sum up all those comparisons we can say that the proposed method is compatible with ILP-based systems in terms of accuracy and coverage, and can generate concept descriptor considerably faster. The experimental results also show that the proposed method induces more number of but shorter concept descriptors compared to state of the art path finding-based approaches in comparable

running time. The induced concept descriptors retain the recall and precision values of the concept descriptors induced by state of the art path finding-based concept discovery systems.

5 Conclusion

In this paper we propose a hybrid graph-based concept discovery system. It is a hybrid approach in a way that similar to path finding-based approaches it looks for paths in a compressed graph, similar to substructure-based approaches. It differs from the path finding-based methods as it does not traverse the graph to find the paths, but instead builds the paths while constructing the graph. It differs from substructure-based approaches as it does not employ graph isomorphism algorithms to compress the graph but does so by executing SQL queries on the data set. Experimental results show that the proposed method is compatible with ILP-based systems in terms of accuracy and coverage, and can generate concept descriptor considerably faster. The experimental results also show that the proposed method induces more number of and shorter, i.e., more human interpretable, concept descriptors concept descriptors compared to state of the art path finding-based approaches.

References

1. Huchard, M., Hacene, M.R., Roume, C., Valtchev, P.: Relational concept discovery in structured datasets. Ann. Math. Artif. Intell. 49(1-4), 39–76 (2007)
2. Quinlan, J.R.: Learning logical definitions from relations. Machine Learning 5, 239–266 (1990)
3. Muggleton, S., Raedt, L.D.: Inductive logic programming: Theory and methods. J. Log. Program. 19/20, 629–679 (1994)
4. Tran, T.N., Satou, K., Ho, T.-B.: Using Inductive Logic Programming for Predicting Protein-Protein Interactions from Multiple Genomic Data. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 321–330. Springer, Heidelberg (2005)
5. Dolsak, B.: Finite element mesh design expert system. Knowl.-Based Syst. 15(5-6), 315–322 (2002)
6. Dzeroski, S., Jacobs, N., Molina, M., Moure, C., Muggleton, S., Laer, W.V.: Detecting traffic problems with ilp. In: Page, D.L. (ed.) ILP 1998. LNCS, vol. 1446, pp. 281–290. Springer, Heidelberg (1998)
7. Muggleton, S., Raedt, L.D., Poole, D., Bratko, I., Flach, P.A., Inoue, K., Srinivasan, A.: Ilp turns 20 - biography and future challenges. Machine Learning 86(1), 3–23 (2012)
8. Alphonse, É., Osmani, A.: On the connection between the phase transition of the covering test and the learning success rate in ilp. Machine Learning 70(2-3), 135–150 (2008)
9. Richards, B.L., Mooney, R.J.: Learning relations by pathfinding. In: Swartout, W.R. (ed.) AAAI, pp. 50–55. AAAI Press, The MIT Press (1992)
10. Gonzalez, J.A., Holder, L.B., Cook, D.J.: Graph-based relational concept learning. In: Sammut, C., Hoffmann, A.G. (eds.) ICML, pp. 219–226. Morgan Kaufmann (2002)

11. Gao, Z., Zhang, Z., Huang, Z.: Learning relations by path finding and simultaneous covering. In: Burgin, M., Chowdhury, M.H., Ham, C.H., Ludwig, S.A., Su, W., Yenduri, S. (eds.) CSIE (5), pp. 539–543. IEEE Computer Society (2009)
12. Washio, T., Motoda, H.: State of the art of graph-based data mining. SIGKDD Explorations 5(1), 59–68 (2003)
13. Santos, J.C.A., Tamaddoni-Nezhad, A., Muggleton, S.: An ILP System for Learning Head Output Connected Predicates. In: Lopes, L.S., Lau, N., Mariano, P., Rocha, L.M. (eds.) EPIA 2009. LNCS, vol. 5816, pp. 150–159. Springer, Heidelberg (2009)
14. Gonzalez, J.A., Holder, L.B., Cook, D.J.: Graph-based concept learning. In: Russell, I., Kolen, J.F. (eds.) FLAIRS Conference, pp. 377–381. AAAI Press (2001)
15. Yoshida, K., Motoda, H.: Clip: Concept learning from inference patterns. Artif. Intell. 75(1), 63–92 (1995)
16. Matsuda, T., Motoda, H., Yoshida, T., Washio, T.: Knowledge Discovery from Structured Data by Beam-Wise Graph-Based Induction. In: Ishizuka, M., Sattar, A. (eds.) PRICAI 2002. LNCS (LNAI), vol. 2417, pp. 255–264. Springer, Heidelberg (2002)
17. Ong, I.M., de Castro Dutra, I., Page, D.L., Santos Costa, V.: Mode Directed Path Finding. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 673–681. Springer, Heidelberg (2005)
18. Gao, Z., Zhang, Z., Huang, Z.: Extensions to the relational paths based learning approach rpbl. In: Nguyen, N.T., Nguyen, H.P., Grzech, A. (eds.) ACIIDS, pp. 214–219. IEEE Computer Society (2009)
19. Kavurucu, Y., Senkul, P., Toroslu, I.H.: Concept discovery on relational databases: New techniques for search space pruning and rule quality improvement. Knowl.-Based Syst. 23(8), 743–756 (2010)
20. Cook, D.J., Holder, L.B.: Graph-based data mining. IEEE Intelligent Systems 15(2), 32–41 (2000)
21. Goutte, C., Gaussier, É.: A probabilistic interpretation of precision, recall and *F*-score, with implication for evaluation. In: Losada, D.E., Fernández-Luna, J.M. (eds.) ECIR 2005. LNCS, vol. 3408, pp. 345–359. Springer, Heidelberg (2005)
22. `http://www.cs.utexas.edu/ftp/mooney/forte/` (accessed December 09, 2012)

Applications of Concurrent Access Patterns in Web Usage Mining

Jing Lu[1], Malcolm Keech[2], and Cuiqing Wang[3]

[1] Southampton Solent University, Southampton UK, SO14 0YN
[2] University of Bedfordshire, Park Square, Luton UK, LU1 3JU
[3] Shenyang University of Chemical Technology, Shenyang China, 110142
Jing.Lu@solent.ac.uk, Malcolm.Keech@beds.ac.uk,
wangcuiqing@126.com

Abstract. This paper builds on the original data mining and modelling research which has proposed the discovery of novel structural relation patterns, applying the approach in web usage mining. The focus of attention here is on concurrent access patterns (CAP), where an overarching framework illuminates the methodology for web access patterns post-processing. Data pre-processing, pattern discovery and patterns analysis all proceed in association with access patterns mining, CAP mining and CAP modelling. Pruning and selection of access patterns takes place as necessary, allowing further CAP mining and modelling to be pursued in the search for the most interesting concurrent access patterns. It is shown that higher level CAPs can be modelled in a way which brings greater structure to bear on the process of knowledge discovery. Experiments with real-world datasets highlight the applicability of the approach in web navigation.

Keywords: web access patterns (WAP) post-processing, concurrent access patterns (CAP), CAP mining and modelling, WAP pruning, knowledge discovery.

1 Introduction

Web usage mining is the process of applying data mining techniques to the discovery of usage patterns from web data to understand and better serve the needs of user navigation on the Internet [1]. The method and algorithms for web usage mining are normally divided into three stages: data collection and pre-processing, pattern discovery and patterns analysis. Depending on the ultimate goals and the desired outcomes, there are different types of pattern discovery and analysis techniques such as association rules generation, sequential patterns mining, cluster analysis and visitor segmentation, and classification and prediction based on web/user transactions [2].

Association rules generation aims to discover unordered correlation among the *frequent* items in a transaction database. In the context of web usage mining, a transaction is a group of web page accesses and an item is a single web page. Therefore, association rules refer to the sets of pages that are accessed together under a minimum support threshold, even though they may not be directly linked with each other.

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 339–348, 2013.

Sequential patterns mining attempts to identify frequently observed sequential occurrence of items across *ordered* transactions (or sequence databases) over time [3]. This technique has been applied in the web usage context to discover web access patterns (WAP) and to capture frequent navigation paths among user trials. WAP-tree was introduced to facilitate development of algorithms for mining access patterns from pieces of web logs [4]. Analysis of these access patterns allows Internet-based organisations to understand user preferences and predict future visit patterns.

Sometimes the scale and level of detail of access patterns found makes it difficult to identify specific navigation trails in web logs. Structural relation patterns have been introduced to extend the search for more complex patterns hidden behind large sequences of data [5]. Discovering these patterns is based on post-processing of sequential patterns mining results – those sequential patterns supported by the same data sequence have been called *concurrent patterns* and could be useful for analysing clickstreams. While much of the research in web mining has its focus on *efficiency*, this paper will pursue concurrent access patterns (CAP) mining and modelling methods which both discover and represent new structures not found elsewhere.

Some related work is highlighted in the next section to provide the relevant background on web access patterns mining and structural relations, culminating in the formal definition of concurrent access patterns. Following the novel framework for WAP post-processing in section 3, a CAP mining and graph construction methodology is presented with a worked example to illustrate the approach taken. An experimental evaluation using real datasets is given in section 4 which showcases the effectiveness of CAP mining and modelling at generating new and stimulating results. The paper draws to a close by summarising and making brief conclusions.

2 Related Work

This section will describe two types of related work to provide background and further motivation – the latter is from the authors' previous research on sequential patterns post-processing.

2.1 Web Access Patterns

Web access patterns have been defined by Pei et al. based on the problem statement of sequential patterns mining [4]. In general, a web log can be regarded as a sequence of user identifier and event pairs. Each piece of web log is a sequence of events from one user or session in chronological order.

Let $P = \{p_1, p_2, \ldots, p_t\}$ be a set of t items (e.g. web pages). An *access sequence* $AS = \langle as_1, as_2, \ldots, as_m \rangle$ is an ordered list of itemsets (web pages), where $as_i \in P$, $1 \leq i \leq m$. The number of items in a sequence is known as the *length* of the access sequence. A sequence $AS_1 = \langle X_1, X_2, \ldots, X_u \rangle$ is *contained* in $AS_2 = \langle Y_1, Y_2, \ldots, Y_v \rangle$ if $u \leq v$ and $X_i \subseteq Y_j$ for all i, $1 \leq i \leq u$ and corresponding j, $1 \leq j \leq v$, and it is denoted by $AS_1 \angle AS_2$.

A *Web Access Sequence Database* (WASD) is a set $\{AS_1, AS_2, \ldots, AS_n\}$, where each AS_i $(1 \leq i \leq n)$ is an access sequence. The *support* in WASD of any given AS is defined as $Sup_{\mathrm{WASD}}(AS) = |\{AS_i: AS \subseteq AS_i\}|/n$, where $|\ldots|$ denotes the number of sequences. AS is

called an *Access Pattern* in WASD with respect to a minimum support threshold *minsup* ($0<minsup\leq 1$) if $Sup_{WASD}(AS)\geq minsup$. Web access patterns mining thus discovers a set of patterns from a given WASD under a user-specified *minsup*.

Example 1. Given a small web log that recorded user access to seven web pages labelled {*a*, *b*, *c*, *d*, *e*, *f*, *g*} respectively and let WASD={<*abcfge*>, <*adcbef*>, <*abfe*>, <*bfg*>}. Table 1 shows the set of all access patterns mined with a *minsup* of 50% and this will be used as a running worked example throughout the paper.

Table 1. Access patterns from a sample WASD

Sequence	Access Patterns Supported by each Sequence (SuppAP), *minsup*=50%
AS_1=*abcfge*	*a,b,c,e,f,g,ab,ac,ae,af,be,bf,bg,ce,cf,fe,fg,**abe**,**abf**,**ace**,**acf**,afe,bfe,bfg,abfe*
AS_2=*adcbef*	*a,b,c,e,f,ab,ac,ae,af,be,bf,ce,cf,**abe**,**abf**,**ace**,**acf***
AS_3=*abfe*	*a,b,e,f,ab,ae,af,be,bf,fe,abe,abf,afe,bfe,abfe*
AS_4=*bfg*	*b,f,g,bf,bg,fg,bfg*

2.2 Structural Relations and Concurrent Access Patterns

Structural relation patterns have been defined as a general designation of patterns that consists of sequential patterns, *concurrent* patterns, *exclusive* patterns, *iterative* patterns and their composition [5]. This sub-section provides the necessary background for concurrent patterns in the web access context.

Following the notation used in the previous sub-section, new ordering relationships based on access patterns can be predicated as follows: given a *Web Access Sequence Database* WASD = {AS_1, AS_2, …, AS_n}, let α, β be two of the access patterns mined from WASD with minimum support threshold *minsup* and assume that α, β are not contained in each other. With regard to a particular sequence $AS\in$ WASD, access patterns α and β have a *concurrent* relationship if and only if both of them have occurred in *AS*, i.e. $(\alpha\angle AS)\wedge(\beta\angle AS)$ is true. This is represented by $[\alpha+\beta]_{AS}$, where the notation '+' represents the concurrent relationship [5].

Definition 1 (Concurrence). *The **concurrence** of access patterns α and β is defined as the fraction of sequences that contains both of the access patterns. This is denoted by* $concurrence(\alpha,\beta) = |\{AS_k:(\alpha\angle AS_k)\wedge(\beta\angle AS_k)\}|/n$, *where* $AS_k\in WASD$, $1\leq k\leq n$ *and n is the total number of access sequences.*

The user-specified *minsup* provides the threshold for frequency measurement when mining frequent itemsets, sequential patterns and web access patterns. Another fractional value, the minimum *concurrence* threshold, *mincon* ($0<mincon\leq 1$) is used to check the concurrent relationships of access patterns.

Definition 2 (Concurrent Access Patterns). *Let mincon be the user-specified minimum concurrence. The concurrence of access patterns* $ap_1, ap_2, \ldots, ap_r$ *is defined as* $concurrence(ap_1, ap_2, \ldots, ap_r) = |\{AS_k:[ap_1+ap_2+\ldots+ap_r]\}|/n$, *where* $AS_k\in WASD$ *and* $1\leq k\leq n$. *If* $concurrence(ap_1, ap_2, \ldots, ap_r)\geq mincon$ *is satisfied, then* $ap_1, ap_2, \ldots$ *and* ap_r *are called **concurrent access patterns**. This is represented by* $CAP_r = [ap_1+ap_2+\ldots+ap_r]$, *where there is no particular order for the access patterns.*

Example 2. Consider WASD={<*abcfge*>, <*adcbef*>, <*abfe*>, <*bfg*>} from Example 1 and assume a *mincon* of 50%. For the access patterns *abe*, *abf*, *ace* and *acf* shown in bold in Table 1, according to Definition 1, *concurrence*(*abe*,*abf*,*ace*,*acf*) = 2/4 = 50%. Therefore, together they constitute the concurrent access pattern given by CAP_4 = [*abe*+*abf*+*ace*+*acf*].

Definition 3 (Maximal CAPs). *The concurrent access patterns represented by* $CAP_k = [a_1+a_2+\ldots+a_k]$ *are contained in* $CAP_{k+m} = [b_1+b_2+\ldots+b_{k+m}]$ *if* $a_i \angle b_j$ *for all i,* $1 \leq i \leq k$ *and corresponding j,* $1 \leq j \leq (k+m)$. *This is denoted by* $CAP_k \angle CAP_{k+m}$. *Concurrent access patterns are called* ***maximal CAPs*** *if they are not contained in any other concurrent patterns.*

Note that the CAP_4 from Example 2 is also maximal, where this type of pattern represents a navigation trail associated with the most frequently accessed patterns, displaying concurrency beyond the user-specified threshold.

3 Web Access Patterns Post-processing

With the successful implementation of efficient and scalable algorithms for mining web access patterns, it is natural to consider extending the scope of previous study to more structured data mining for enhanced knowledge discovery in web usage.

3.1 Framework

Fig. 1 introduces a novel framework for *web access patterns post-processing* which can be described through its four rows. First, depending on the nature of the log files, data pre-processing involves different types of tasks such as data fusion and cleaning, user/web page identification and data transformation [2]. The 1st row completes with access patterns mining using traditional sequential patterns mining techniques.

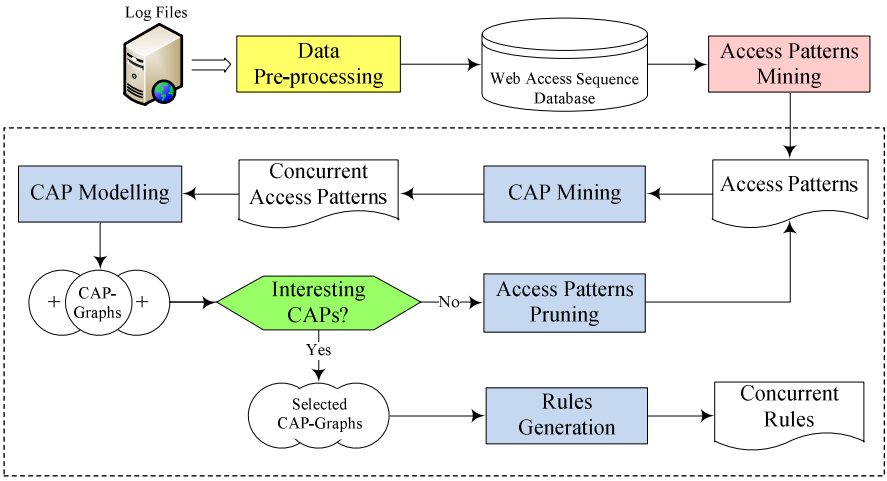


Fig. 1. WAP post-processing framework

The initial cycle of concurrent pattern discovery proceeds through CAP mining and modelling (2nd row of Fig. 1), which yields CAP-Graphs for analysis (3rd row, see sub-section 3.3). If the CAPs are deemed uninteresting, then access patterns pruning is undertaken before another cycle of pattern discovery is instigated. If at this stage the patterns analysis is successful, through selection of more interesting CAP-Graphs, then concurrent rules can be generated (4th row) and the process completes.

3.2 From WAP to CAP Mining

The method used to mine concurrent access patterns is derived from that for concurrent sequential patterns [5] and has been highlighted previously in the web mining context [6]. It is illustrated step-by-step below with the running worked example.

Step 1. Calculation of Access Patterns Supported by each Sequence.

Access patterns which are supported by a sequence AS_i (i.e. $AS_i \in$ WASD, $1 \le i \le n$) are computed and denoted by: $SuppAP(AS_i) = \{ap: ap \in AP \wedge ap \angle AS_i\}$.

The results of this for Example 1 are shown in the second column of Table 1.

Step 2. Determination of Concurrent Access Patterns.

Each $SuppAP(AS_i)$ can be viewed as a transaction, i.e. the unordered set of access patterns supported by data sequence AS_i. Thus, the problem of finding the concurrent access patterns which satisfy the specified minimum concurrence (*mincon*) becomes one of mining frequent itemsets under *minsup=mincon*.

To calculate CAPs with *mincon*=50% means finding the groups of access patterns which occur together in at least 50% of the data sequences. In Example 1, both data sequences AS_1 and AS_2 support the common set of access patterns which constitutes CAP = [*a+b+c+e+f+ab+ac+ae+af+be+bf+ce+cf+abe+abf+ace+acf*].

Step 3. Finding Maximal Concurrent Access Patterns.

According to the containing relationship among sequences, the CAPs need to be simplified in order to deduce the maximal concurrent patterns. Using Definition 3, these can be obtained by deleting the concurrent access patterns which are contained by other CAPs, then deleting the access patterns in particular CAPs (in turn) which are contained by other access patterns within the same CAP.

For example, for the CAP from the previous step, the following contained relationships exist: $a \angle af \angle acf$, $b \angle ab \angle abe$, $c \angle ce \angle ace$, ... ; therefore this concurrent access pattern can be reduced to the maximal CAP_4 = [*abe+abf+ace+acf*].

3.3 CAP Modelling and Knowledge Representation

The use of graphical models in data mining has motivated the development of a sequential patterns graph, SPG that explores the inherent relationships among these patterns. The idea was adapted in [7] for modelling concurrent sequential patterns.

Each page view can be represented as a node in a graph and the directed edge between two nodes indicates a sequential relation, i.e. user navigation. The first node for each path is called a *start* node and the last is called a *final* node. A node with two or more incoming sequential relations applied to the paths is called a *synchronizer* node, while *fork* nodes allow independent execution between concurrent paths.

Without dwelling on the detail, in this context concurrent access patterns graph (CAP-Graph) is thus a graphical representation of CAPs which maps a function from a set of directed edges to a set of pairs of nodes (see [7]). The definition of CAP-Graph is an extension of that for sequential patterns graph and therefore the method to construct SPG can be adapted for CAP modelling.

For the running worked example, and following initialisation, the intermediate construction and iteration phases proceed as illustrated in Fig. 2. By taking sequential patterns in turn, nodes and directed edges are added step-by-step, culminating in the final CAP-Graph for CAP_4=[*abe+abf+ace+acf*].

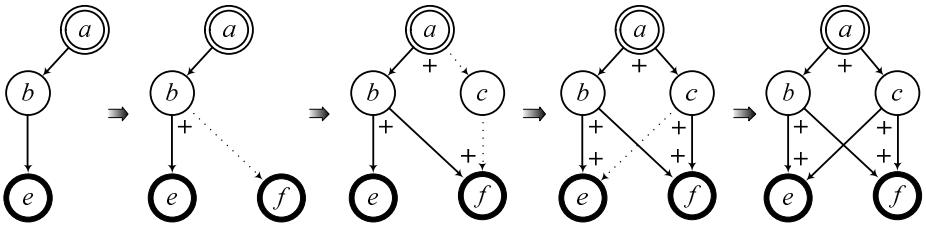


Fig. 2. Modelling CAP_4=[*abe+abf+ace+acf*] using adapted SPG method

We conclude WAP post-processing by indicating another way to simulate knowledge representation here through *concurrent rules*. For example, the following rules can be generated based on the above CAP-Graph: $a \Rightarrow [b+c] \Rightarrow [e+f]$. This may be used to predict visit patterns: after accessing page *a*, visitors are likely to access pages *b* and *c* (in no particular order), followed by further visiting pages *e* and *f*.

4 Experiments and Evaluation

The empirical analysis of the proposed CAP mining and modelling methods was performed on real-world datasets to test their effectiveness as well as to further illustrate the framework and process for web access patterns post-processing.

4.1 Pre-processing Real Datasets

The initial focus is on pre-processing real datasets available from published sources, namely *BMS-WebView-1* and *msnbc.com*.

BMS-WebView-1 was the first of the three KDD CUP 2000 datasets, which is sometimes called "Gazelle" [8]. This dataset contains clickstream data from a former web retailer, *Gazelle.com*, and has been used widely to assess the performance of frequent patterns mining. The particular file here contains 59,602 sequences and has been pre-processed already, which is convenient, although the inherent meaning of the 497 items no longer features in the competition web site.

The msnbc.com anonymous web dataset was drawn from [9]. The original data comes from Internet Information Server (IIS) logs for msnbc.com and contains 989,818 data sequences in total for one day. Each sequence in the dataset corresponds

to page views of a user during that 24-hour period and each event (item) in the sequence corresponds to a user's request for a page.

For web access patterns mining, the original ordering of the pages is important. If a page appears twice in succession in the same sequence, then only the first request will remain following pre-processing. There are many such sequences for the msnbc dataset which contain repetitive/adjacent items or single items only. Therefore, pre-processing has been extended for the experiments which reduces data sequences by 60%. The final file has been divided fairly equally into four datasets called *msnbc1.dat* to *msnbc4.dat*, with each processed separately, to make computation more manageable.

4.2 BMS-WebView-1

All experiments have been conducted on a 2.5GHz Intel Core i5 processor with 4GB main memory running Windows 7. Appropriate use has been made of *PrefixSpan* for access patterns mining, an open-source data mining package available from [10].

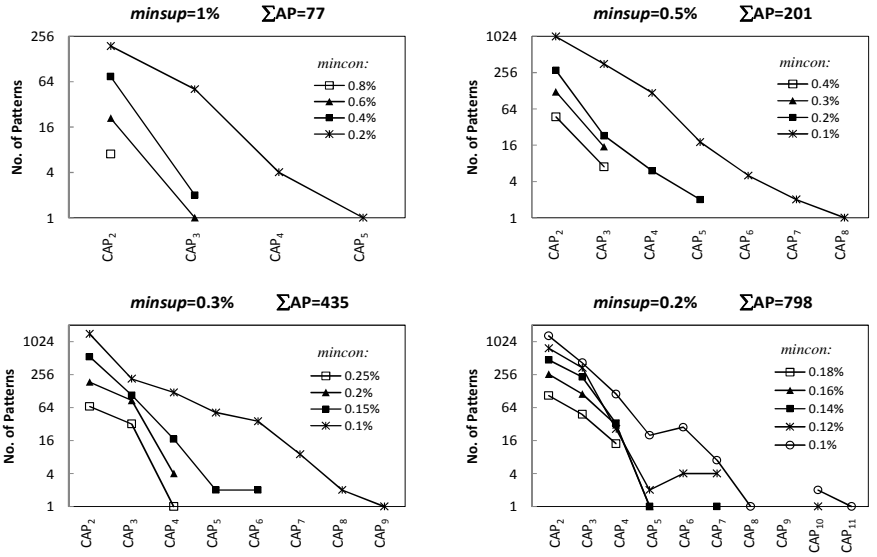


Fig. 3. CAPs mined from BMS-WebView-1 under various *minsup* and *mincon*

Several tests have been performed on BMS-WebView-1 across a range of *minsup* values – no CAPs were found beyond a 2% threshold. A summary of the nature of these results is shown in Fig. 3 when *minsup*=1%, 0.5%, 0.3% and 0.2%. It is noted that the number of patterns increases as *mincon* thresholds decrease within the same *minsup*, while no CAPs have been found for this dataset when *mincon*=*minsup*.

It is significant that *higher level* CAPs can be mined for BMS-WebView-1 under decreasing *minsup* and *mincon*, although there are limits for these experiments. For example, when *minsup*=0.1%, the number of access patterns mined reaches nearly 4,000 and run time grows exponentially for CAP mining under the smaller *mincon*. Likewise,

when *minsup*=0.3%, the number of CAPs exceeds 2,000 once *mincon*≤0.08, which suggests a natural cut-off point for presentational purposes here.

Two novel and interesting CAP-Graphs are selected from BMS-WebView-1 results and highlighted in Fig. 4, where CAP_9 and CAP_{11} patterns demonstrate that complex structural relations can be discovered and modelled from web usage data.

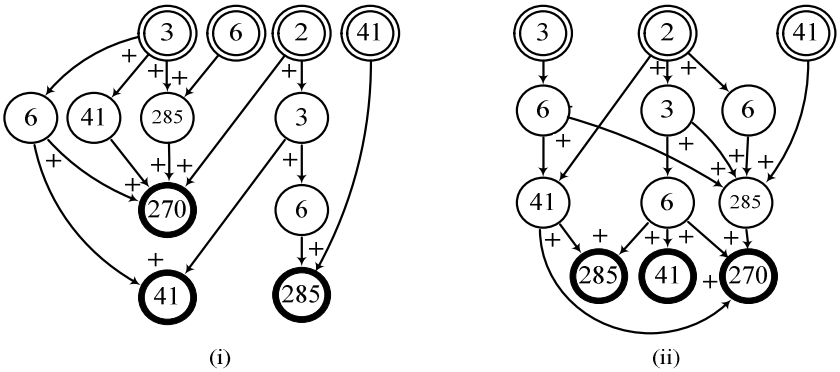


Fig. 4. Sample (i) CAP_9 when *minsup*=0.3% and (ii) CAP_{11} when *minsup*=0.2%; *mincon*=0.1%

4.3 msnbc.com

The second set of tests was performed on the msnbc dataset, where no CAPs were found for *minsup*≥7%. Following preliminary access patterns mining with *minsup*=5%, several CAPs were discovered with repetitive formats and modelling results confirmed these CAPs were not interesting. msnbc is a dense dataset containing only 17 different items and this causes significant repetition in the access patterns. Therefore, to improve the mining results, WAPs have been pruned by removing repetitive and adjacent items while deleting single length patterns not useful in the context of concurrency.

CAP mining thus resumes based on the pruned WAPs before CAP modelling once more helps to determine whether there are any interesting patterns. This sub-section presents selected results from the *msnbc1* dataset only, although experiments were also performed on *msnbc2-4* which were essentially equivalent. Initial tests with *minsup*=5% show a progression of the highest level patterns from a single CAP_2 to a pair of CAP_3 to a single CAP_4 then a single CAP_5 as *mincon* decreases from 4% to 1%.

Varying the *minsup* threshold below 5% increases the number of access patterns as well as the potential for CAP discovery. This is best illustrated for *minsup*=1%, where a representative sample of the highest level CAP-Graphs can be selected, as in Fig. 5.

The results are shown *with meaning* this time, where original codes are transposed into the corresponding page category. For *mincon*=1% there was a choice of 21 CAP_3 and the two highlighted in (i) and (ii) give a flavour for the concurrent patterns mined. For *mincon*=0.75% and 0.5% there were only single CAP_5 and CAP_6 respectively, as in (iii) and (iv), each displaying a more interesting structure. And for *mincon*=0.25% there was a choice of four CAP_8, where the two patterns in (v) and (vi) demonstrate that complex structural relations can again be found from real web data.

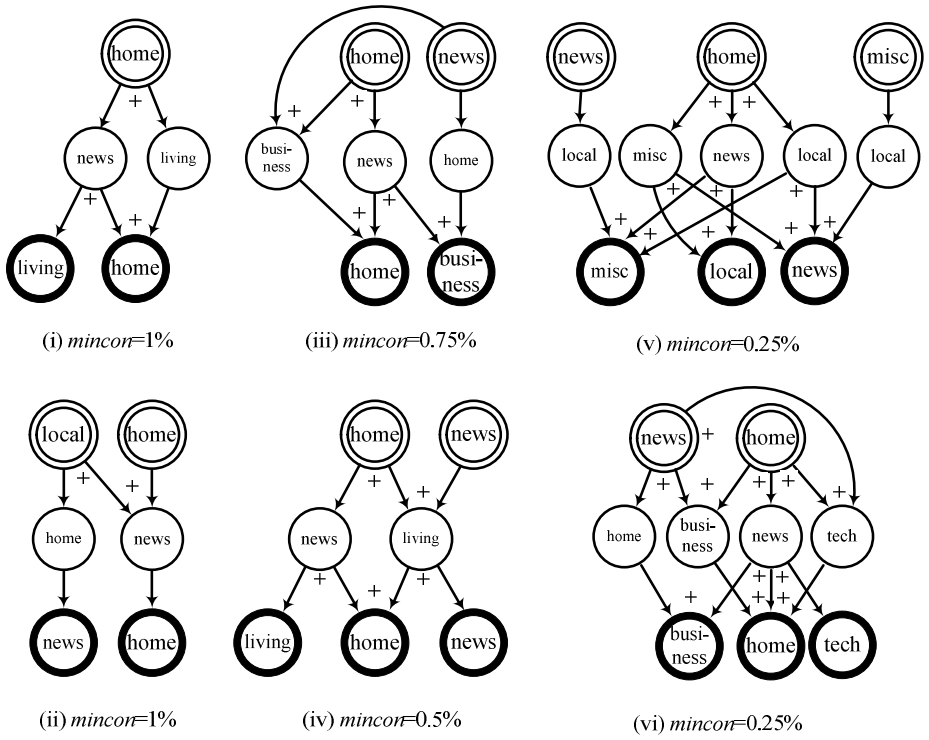


Fig. 5. Examples of CAP-Graphs for msnbc1.dat when *minsup*=1% under various *mincon*

5 Conclusion

The focus of this work has been on the application of original data mining and modelling research to the web mining context. In particular, concurrent access patterns are defined and explored through an approach to web access patterns post-processing which employs a novel framework for knowledge discovery and analysis in Fig. 1. Constituent data mining and modelling techniques yield concurrent access patterns and graphs allowing the question to be raised: are these CAPs interesting? This calls to some extent for a subjective judgement, but careful inspection of CAP-Graphs provides enough evidence to inform a suitable answer.

Experiments have been performed on a number of real-world datasets and two are reported here: BMS-WebView-1 and msnbc.com. Our clear motivation is to evaluate the effectiveness of the techniques while illustrating the stages of the WAP post-processing approach. Pre-processing is required for one of the datasets and initial access patterns mining undertaken for both. The results of CAP mining and modelling are always interesting up to a point, but there can still be many CAP-Graphs to consider and they sometimes exhibit unhelpful repetition. This is where access patterns pruning can come in, which was especially useful for the second dataset.

The extent to which higher level CAPs can be discovered is first examined for BMS-WebView-1, where results are reported in Fig. 3 for $0.2\% \leq minsup \leq 1\%$. The connectivity and structure of the sample CAP-Graphs shown in Fig. 4 highlights the potential for knowledge representation through concurrent patterns, albeit with no attached meaning here for this web retail dataset. Pre-processing is required for msnbc.com before it is divided into four for individual investigations across $1\% \leq minsup \leq 5\%$. When the lower *minsup* value is explored in tandem with decreasing *mincon* then, following access patterns pruning, the progressive nature of structural relations modelled unfolds in Fig. 5. The higher level CAPs convey the increasingly rich navigation of real pages viewed in this media context.

Future work is being considered in three directions: first, while concurrent rules are suggested in this paper, a specific study of their role and generation would be useful. Second, while CAP modelling presents a stimulating visualisation of mining results, there can be many CAP-Graphs to view for large-scale datasets – a more automated approach to CAP selection would be worthwhile. And finally, while the WAP post-processing framework does not limit the number of iterations in pursuit of the most interesting CAPs, a single additional cycle is illustrated in experiments here. Criteria could be developed for further access patterns pruning informed by other types of constraint relevant to concurrency and applied more widely.

References

1. Srivastava, J., Cooley, R., Deshpande, M., Tan, P.N.: Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. SIGKDD Explorations 1(2), 12–23 (2000)
2. Liu, B.: Web Data Mining – Exploring Hyperlinks, Contents, and Usage Data. Book series: Data-Centric Systems and Applications. Springer, Heidelberg (2011)
3. Agrawal, R., Srikant, R.: Mining Sequential Patterns. In: 11th International Conference on Data Engineering, pp. 3–14. IEEE Computer Society Press, Taipei (1995)
4. Pei, J., Han, J., Mortazavi-asl, B., Zhu, H.: Mining Access Patterns Efficiently from Web Logs. In: Terano, T., Liu, H., Chen, A.L.P. (eds.) PAKDD 2000. LNCS, vol. 1805, pp. 396–407. Springer, Heidelberg (2000)
5. Lu, J., Chen, W.R., Adjei, O., Keech, M.: Sequential Patterns Post-Processing for Structural Relation Patterns Mining. International Journal of Data Warehousing and Mining 4(3), 71–89 (2008)
6. Lu, J., Keech, M., Chen, W.R.: Concurrency in Web Access Patterns Mining. In: International Conference on Data Mining, vol. 58, pp. 600–609. WASET, Venice (2009)
7. Lu, J., Chen, W.R., Keech, M.: Graph-based Modelling of Concurrent Sequential Patterns. International Journal of Data Warehousing and Mining 6(2), 41–58 (2010)
8. Kohavi, R., Brodley, C., Frasca, B., Mason, L., Zheng, Z.: KDD-Cup 2000 Organizers' Report: Peeling the Onion. SIGKDD Explorations 2(2), 86–98 (2000)
9. UCI KDD Archive, `http://kdd.ics.uci.edu/databases/msnbc/msnbc.html`
10. IlliMine System Package, `http://illimine.cs.uiuc.edu/`

Cell-at-a-Time Approach to Lazy Evaluation of Dimensional Aggregations*

Peter Thanisch[1], Jyrki Nummenmaa[1], Tapio Niemi[2], and Marko Niinimäki[2]

[1] School of Information Sciences, University of Tampere, Finland
[2] Helsinki Institute of Physics, University of Helsinki, Finland
{pt,jyrki}@sis.uta.fi, {tapio.niemi,man}@cern.ch

Abstract. We present a lazy evaluation technique for computing summarized information from dimensional databases. Our technique works well with a very large number of dimensions. While the traditional approach has been to preprocess analysis models from which the user selects the data of interest, in our approach only the cells required by the user are calculated using a cell-by-cell computation strategy.

1 Introduction

In dimensional databases, dimension members classify numeric measure data for summarization. The number of potential summarizations is exponential in the number of dimensions. As the number of dimensions increases, increased processing time means that building a cube and pre-computing aggregations becomes an increasingly inefficient option [2]. In many business intelligence (BI) applications, for aggregation queries, i.e. queries in which the user specifies slicers and aggregation levels, a good approximation, rather than the exact answer, suffices [9]. Hence approximation techniques based on statistical compression of the raw data [4] can provide better performance. Good approximations might suffice when looking at the "big picture", but once users identify the particularly interesting results (which may be just a few cells in a pivot table) they will want the exact answers, rather than approximations.

We propose a technique, based on lazy evaluation, for efficiently computing the exact answer to an aggregation query for any given cell and set of aggregation levels. Hence our technique can be used in conjunction with approximation techniques to allow the user to obtain some exact answers where the approximate answer calls for further examination.

The content of the rest of the paper is as follows. Section 2 discusses related research. Section 3 describes our method. Section 4 covers implementation and performance. Section 5 discusses the case where several related aggregations mst be computed. Section 6 contains concluding remarks.

* This research has been partially supported by the Finnish Academy grant SA 1139590.

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 349–358, 2013.

2 Related Research

Li et al. have noted that a large number of dimensions, alone, does not create performance problems for dimensional databases. Rather, it is the nature of the navigation around the cube space which can cause such problems. For example, regardless of the number of dimensions in the cube, in practice a given query processing operation tends only to involve a small number of dimensions [6]. Li et al. also state that a "typical" dimensional database query is likely to ignore many dimensions (i.e. treating them as irrelevant), fix some dimensions (e.g., using query constants as instantiations), and leave only a few to be manipulated (for drilling, pivoting, etc.) [6].

Fu and Hammer [3] give a special data structure and an efficient algorithm for processing ad-hoc OLAP queries on top of a relational database. Their implementation requires only one scan over a data set. Morfonios et al. [7] review different ROLAP (Relational OLAP) implementations and algorithms for data cubes, focusing on methods in which at least a part of the data cube is precomputed.

Microsoft's Tabular data model [8] presents results in a pivot table from which the user can copy the cube formula for a cell and paste it into a report. just as a formula is copied in a spreadsheet. This is similar to our approach, though the purpose of cube formulas is quite different, since they are used to identify the result, not to compute it.

In the performance tests we report, we used a fully denormalised table. Although this is not essential, Jacobs [5] has pointed out that for 'big data', a fully denormalized table provides the best solution for analytical purposes.

3 Notation and Structures Required for the Technique

We provide notation to describe the roles played by a relation's attributes in pivoting and summarization operations. For notational convenience, we regard the set of dimension schemata as an ordered collection, $< D_1, D_2, \ldots, D_n >$. Where S is a set, we let $|S|$ denote the cardinality of the set.

Definition 1 (Summarization Schema). *A summarization schema is a pair* (D, M) *where*

1. *A dimension schema* D_i $(1 \leq i \leq n)$ *is a sequence* $< A_1^i, A_2^i, \ldots, A_x^i >$ $(x = |D_i|)$ *of attributes, called levels.*
2. $D =< D_1, D_2, \ldots, D_n >$ *is an ordered set of dimension schemata.*
3. $M = \{M_1, M_2, \ldots, M_q\}$ *is a set of measure attributes.*
4. *The members of the collection* $\{D_1, D_2, \ldots, D_n, M\}$ *are pairwise disjoint.*

The order of the attributes in sequence D_i, $< A_1^i, A_2^i, \ldots, A_x^i >$ determines the level in the aggregation hierarchy to which the attribute corresponds. Thus the following set of functional dependencies, F_i, is associated with D_i: $F_i = \{A_g^i \rightarrow A_{g-1}^i | g = 2, 3, \ldots, |D_i|\}$. The attribute set $D_1 \cup D_2 \ldots \cup D_n \cup M$ can be used as a relation schema. We say that a relation over $D_1 \cup D_2 \ldots \cup D_n \cup M$ is a relation over the summarization schema (D, M).

Example 1. Let (D, M) be a summarization schema in which $D =< D_1, D_2 >$ and $M = \{M_1\}$. We let $D_1 = \{A_1^1, A_2^1\}$ and $D_2 = \{A_1^2, A_2^2, A_3^2\}$. Let $r = \{t_1, t_2, \ldots, t_5\}$ be the following relation over the summarization schema (D, M).

	A_1^1	A_2^1	A_1^2	A_2^2	A_3^2	M_1
t_1	b1	c1	d1	e1	f1	10001
t_2	b1	c2	d1	e1	f1	10020
t_3	b2	c3	d1	e1	f2	10300
t_4	b2	c3	d2	e2	f3	14000

Definition 2 (Set of domain values). *Let r be a relation over a summarization schema, (D, M). For $1 \leq i \leq |D_i|, 1 \leq j \leq n$, $DOM_j^i(r)$ denotes the set of domain values in the A_j^i column of r.*

Example 2. Thus, in our running example, we have the following values:

Dimension D_1	Dimension D_2
$DOM_1^1(r) = \{b1, b2\}$	$DOM_1^2(r) = \{d1, d2\}$, $DOM_3^2(r) = \{f1, f2, f3\}$
$DOM_2^1(r) = \{c1, c2, c3\}$	$DOM_2^2(r) = \{e1, e2\}$

Given a relation, r, for the summarization schema, (D, M), we need to pivot the data along a set of axes that correspond to an arbitrary subset of D's dimension schemata. An implementation must be capable of summarising the raw data in r to any of the levels in the aggregation hierarchies associated with those axes. Our approach requires the following items of information as its input.

1. The set of axes required for the analysis. Each axis corresponds to one of the dimension schemata.
2. For each of the required axes, the level in the axis aggregation hierarchy required for the user's analysis.
3. The measure attribute to be used.
4. A summary function (e.g. AVG, SUM or COUNT) for use with the measure attribute. As with any pivot table, the user must select a summary function in order that a single numerical quantity can be associated with that cell.
5. Our approach provide one item of summarised information at a time. Thus, for each axis, a slicer must be specified. Each slicer must be a member of the domain of the attribute associated with aggregation hierarchy level for the axis, as specified in Item 2 above.

In our approach, we define three structures which are used to hold this information: a rollup vector (for items one and two), a summarization state and a slicer vector (for the last item in the list). We now define those three structures.

Definition 3 (Rollup Vector). *Let $D =< D_1, D_2, \ldots, D_n >$ be an ordered collection of n dimension schemata. A* rollup *vector for D is an ordered n-tuple of integers, $L =< L_1, L_2, \ldots, L_n >$, where for $i = 1, 2, \ldots, n, 0 \leq L_i \leq |D_i|$.*

The interpretation of the values of the elements of L is as follows.

- $L_i = 0$ *denotes that the user's analysis does not require an axis corresponding to dimension schema* D_i.
- $1 \leq Li \leq |D_i|$ *denotes that the user's analysis requires an axis corresponding to dimension schema* D_i.

Furthermore, assume that $L_i = l$, *and* r *is the relation of interest. Then the axis corresponding to dimension schema* D_i *shall be labelled by the elements of* $DOM_l^i(r)$, *i.e. members of the attribute column* A_l^i *in* r.

Definition 4 (Summarization Structure). *Let* (D, M) *be a summarization schema and let* L *be a rollup vector for* D. *A summarization structure is a pair* (D, L), *with an axis for each non-zero element of* L. *For any dimension, say* D_i, *for which the element* $L_i = l$, *of* L *is non-zero, the summarization structure contains an axis corresponding to* D_i. *If* r *is a relation over the summarization schema* (D, M), *that axis is labelled by the elements of the set* $DOM(A_l^i(r))$.

Definition 5 (Summary State). *Let* (D, M) *be summarization schema. A summary state is a 5-tuple* (r, D, L, f, M_j), *where* (D, L) *is a summarization structure,* f *is a summary function,* $M_j \in M$ *is a measure attribute, and* r *is a relation over the cube schema* (D, M).

Informally, the summary state contains all of the information needed to pivot relation r, whilst aggregating the measure data in r with summary function f.

Example 3. Let (r, D, L, f, M_1) be a summary state, where r and $(D, \{M_1\})$ are the relation and summarization schema in our running example, L is the rollup vector $< 1, 2 >$ and f is the summary function SUM. The rollup vector $< 1, 2 >$ denotes that the axis corresponding to dimension D_1 must be rolled up to level 1 in its aggregation hierarchy and the axis corresponding to D_2 must be rolled up to level 2 in its aggregation hierarchy. The data in the cells of the summarization structure is to be aggregated using the summary function SUM. The summarised data associated with the summary state is:

		b1	b2
d1	e1	20021	10300
d2	e2	0	14000

Example 4. Let (r, D, L, f, M_1) be the summary state in Example 3, with the only difference that the rollup vector is $< 1, 0 >$. This new summary state does not contain an axis corresponding to dimension D_1, and it only contains an axis for dimension D_2. That axis is labelled with the elements of $DOM_2^2(r)$. As before, the summary function is SUM, so summarised data is as follows:

b1	b2
20021	24300

For a given rollup vector, the *slicer vector* identifies a specific cell in a summary state by specifying the relevant member in each of the summary states axes,

given the levels in the rollup vector. Permissible values in the slicer vector are constrained by the values in the rollup vector. The slicer for a given dimension must be a single member of the level for that dimension which is specified in the rollup vector.

Before we can define the slicer vector formally, we must introduce some more notation. We use the following notation to denote the set of tuples in a relation r, which have the value x in this attribute column.

Definition 6 (Tuples). *Given a relation, r, a dimension, D_i, $1 \leq i \leq n$, and a level number, l, $1 \leq l \leq |D_i|$, let A_l^i denote the attribute associated with level l of dimension D_i and let $x \in DOM_l^i(r)$. The set of tuples in a relation r, which have the value x in the attribute column denoted by i and l is:*

$$tuples(i,l,x) = \begin{cases} r, & if\ l = 0 \\ \{t \in r | t[A_l^i] = x\}, & if\ l \neq 0 \end{cases}$$

We do not pre-compute the tuples() sets. They are used in a formula to indicate which values in the original table must be summarised in order to obtain the correct aggregated value for a particular cell in the summarization structure.

Example 5. The tuples() sets for our running example is as follows:

tuples(1,1,b1) = {t1,t2}	tuples(1,1,c3) = {t3,t4}	tuples(2,2,e2) = {t4}
tuples(1,1,b2) = {t3,t4}	tuples(2,1,d1) = {t1,t2,t3}	tuples(2,2,e1) = {t1,t2,t3}
tuples(1,1,c1) = {t1}	tuples(2,1,d2) = {t4}	tuples(2,3,f2) = {t3}
tuples(1,1,c2) = {t2}	tuples(2,3,f1) = {t1,t2}	tuples(2,3,f3) = {t4}

Definition 7 (Slicer Vector). *Given the summary state $s = (r, L), L =< L_1, L_2, \ldots, L_n >$, a slicer vector for s is an ordered set $< x_1, x_2, \ldots, x_n >$ where for $i = 1, 2, \ldots, n$, if in the rollup vector, $L_i = 0$, then $x_i = NULL$; otherwise let $L_i = l > 0$, then $x_i \in DOM_l^i(r)$.*

Example 6. Given the rollup vector, $< 1, 2 >$, a valid slicer vector is $< b1, e1 >$.

Given a summary state, $s = (r, L)$, and a slicer vector, we can find the collection of tuples() sets which contain the measure values needed by the summary function in order to compute the value for the cell identified by the slicer vector.

Example 7. In our running example, the rollup vector, $< 1, 2 >$, and the slicer vector, $< b1, e1 >$, give $tuples(1, 1, b1) = \{t_1, t_2\}$ and $tuples(2, 2, e1) = \{t_1, t_2, t_3\}$.

The rollup and slicer vectors identify the cell in the summarization structure for which a summarization operation is to be performed. In our implementation, the actual data for this summarization operation is in the original relation.

Definition 8 (Summary Rowset). *Let $s = (r, D, L, f, M_1)$ be a summary state and let $x =< x_1, x_2, \ldots, x_n >$ be a slicer vector for s. The set $T = \bigcap_{i=1}^{n} tuples(i, L_i, x_i)$ is the summary rowset for s and x .*

In Theorem 1, below, we prove that the summary rowset is precisely the set of rows which contain the measure values needed by the summary function in order to produce its result for the summary cell identified by the slicer vector.

Theorem 1. *Let* $s = (r, D, < L_1, L_2, \ldots, L_n >, f, M_1)$ *be a summary state, and* $x =< x_1, x_2, \ldots, x_n >$ *a slicer vector for* s*. Then a row* $t \in r$ *must be included in an aggregation operation for the cell identified by* x *if and only if*

$$t \in T = \bigcap_{i=0}^{n} tuples(i, L_i, x_i)$$

Proof (ONLY IF). Let the measure to be aggregated be $M_j \in M$. First, consider a tuple, say $t \in r$, such that $t \in T$. From the definition of tuples(), for i=1, 2, ..., n, if $L_i = l \neq 0$ then $t[A_l^i] = x_i$. Thus the value in $t[M_j]$ must be included in the aggregation for the cell defined by the slicer vector x.

[IF] Next, consider a tuple, say $t \in r$, such that $t \notin T$. Thus,there exists h, $1 \leq h \leq n$, such that $t \notin tuples(h, L_h, x_h)$. Suppose that $L_h = 0$. Thus $tuples(h, L_h, x_h) = randt \in r$, a contradiction. Thus $Lh \neq 0$. But in that case, from the definition of tuples(), $t[A_{Lh}^h] \neq x_h$. Thus t has correctly been excluded from the set of rows to be aggregated for this cell.

Theorem 1 is directly applicable wherever the aggregation expression involves summary functions which only require values from the measure column. This leads to an efficient method for identifying the summary rowset.

Example 8. Using the data in our running example, with the summary state and the slicer vector from Example 6, suppose we want to apply the SUM aggregation operator. Using Theorem 1, the summary rowset is $T = tuples(1, 1, b1) \cap tuples(2, 2, e1) = \{t_1, t_2\}$. The sum for this cell is given by the expression $\sum_{t \in T} t_i[M]$. From the data, $t_1[M] = 10001$ and $t_2[M] = 10020$. The sum is 20021.

Thus, given a denormalised [10] table (which contains both the fact data and the dimension data), a rollup vector and a slicer vector, we can use the above formula to compute a summarised value anywhere in the summarization structure directly, without having to create the entire cube. The only data structures used were a conventional relational table and two vectors.

4 Implementation and Performance Results

To demonstrate our technique, we used was a modestly-specified notebook computer with 4GB of memory, a 2.3GHz dual core processor and a 64-bit architecture. The database system was Microsoft SQL Server 2012 [8].

Our synthetic data follows a naming convention for our attribute columns, taking the form <DnLm>, where n is the number of the dimension and m is the level number within the aggregation hierarchy associated with the dimension. All of the attributes have the same data type, namely string. In practice, the

attributes would have a variety of data types, e.g. date and integer. We have implemented the rollup and slicer vectors as ordinary tables in the database. In practice, this information would be captured from the user's application and stored in a data structure. There is a single aggregation function and a single measure attribute in our example. This could be generalised to allow an expression involving several measure attributes and aggregation operations.

It is common in dimensional databases to have one very large dimension table, some medium-sized dimension tables and a large number of small dimension tables [6]. Our synthetic data comprises a hundred and one dimensions:

- dim_001, ..., dim_099: each has 64 members and three aggregation levels.
- dim_100: 2000 members; two aggregation levels.
- dim_101: 500 members; two aggregation levels.

The dimension attributes in a dimension table are intended to correspond to levels in an aggregation hierarchy. When the data is generated, there is a many-to-one relationship between the attribute corresponding to the lower level of the hierarchy to the attribute corresponding to the next-higher level in the hierarchy.

There are 303 columns in the table: 301 dimension attributes, a measure attribute and a primary key column. Our fact table has one million rows. It comprises a single measure and 101 columns, each containing foreign key references to one of the dimension tables.

Given a rollup vector and a slicer vector, Theorem 1 is the basis for a mechanical process to find a set-theoretic expression for the set of tuples which must be aggregated. Our technique uses the pair of vectors as input and generates the code for the SQL query which retrieves the set of records corresponding to the tuples and perform the aggregation on the chosen measure attribute. We have implemented the technique using dynamic SQL. The query generated by the dynamic SQL is give in Figure 1.

```
SELECT SUM(M1)  FROM D1L2 = 'D01L2M01' AND D2L1 = 'D02L1M1' AND
  D3L2 = 'D03L2M01' . . .  AND D100L1 = 'D100L1M1' AND D101L2 = 'D101L2M0';
```

Fig. 1. The implementation of Theorem 1

Once the text of the query has been generated, the query can be executed, using the EXEC command, to perform the required aggregation.

Figure 2 shows the dynamic SQL code for generating a summation query from the rollup vector and slicer vector. The text of the SQL query is built up, step by step, in the variable @sql. It uses the function in Figure 3 to generate a string which contains the name of the unique attribute corresponding to the level number and the dimension number. In general, a lookup table is needed to associate an attribute name with the dimension and level numbers.

We added a column-store index [1] to our database. The average elapsed time to execute the software which generates the dynamic SQL query text, compiles that query and executes it against the database was less than six seconds.

```
CREATE PROCEDURE [dbo].[sp_Build_and_Exec_Query]
       @numberOfDimensions INT,  @measureAttributeName NVARCHAR(100),
       @aggregationFunctionName NVARCHAR(100) AS
BEGIN
    DECLARE @sql NVARCHAR(MAX),  @paramDefs NVARCHAR(MAX),
            @attributeName NVARCHAR(MAX;
    SET @sql = N'SELECT ' + @aggregationFunctionName
             + N' ( '  + @measureAttributeName  + N' ) FROM tblFlat WHERE ';
    DECLARE @dim INT, @level INT, @value NVARCHAR(8),
            @leadingZero NVARCHAR(1), @firstTerm CHAR(3);
    SET @firstTerm = 'YES';  SET @dim = 1;
    WHILE @dim <= @numberOfDimensions BEGIN
        SELECT @level = [lvl] FROM dbo.RollupVector WHERE dimNumber = @dim;
        IF @level > 0 BEGIN
            SELECT @value = [val] FROM dbo.SlicerVector
            WHERE dimNumber = @dim;
            SET @attributeName = dbo.getAttributeName(@level, @dim);
            IF @firstTerm = 'NO' SET @sql = @sql + N' AND ';
            SET @sql = @sql +  + @attributeName + N' = ''' + @value + '''';
            SET @firstTerm = 'NO';
        END
        SET @dim = @dim + 1;
    END
    EXECUTE sp_executesql @sql;
END
```

Fig. 2. The code to generate the summation query

Restricting our attention to the response time of the SELECT query, in successive performance tests, we increased the number of dimensions from 51 to 101 in units of five. Between each performance measurement, all buffer content was removed from the buffer pool. In Figure 4, the response time graph in the test range, i.e. up to 101 dimensions, is linear.

We tested how changing the number of rows affect average response times:

Rows in table	500,000	600,000	700,000	800,000	900,000	1,000,000
Avg. Response time (ms)	1674	1841	2032	1994	2137	2086

To test whether retrieving a group of "nearby" cells affects performance, we retrieved ten cells all with the same rollup vector and with the associated 10 slicer vectors only differing from each other in one dimension. After the first cell is retrieved on cold buffer and cache, the subsequent cells are retrieved with warm buffers and cache. Retrieving the first cell took 3.6 seconds, whereas the average retrieval time for other cells was 115 milliseconds.

5 Processing Several Cells

Typically, a multidimensional query identifies a set of cells, rather than an individual cell, over which the aggregation is to be performed. However, the approach

```
CREATE FUNCTION getAttributeName( @dim INT ) RETURNS NVARCHAR AS
BEGIN
    DECLARE @attributeName NVARCHAR(200), @lvl INT;
    SELECT @lvl = [lvl] FROM dbo.RollupVector WHERE dimNumber = @dim;
    IF @dim <= 9 SET @attributeName =  N'D0' + CAST(@dim AS NVARCHAR)
                                     + N'L' + CAST(@lvl AS NVARCHAR)
    ELSE SET @attributeName =  N'D' + CAST(@dim AS NVARCHAR)
                             + N'L' + CAST(@lvl AS NVARCHAR);
    RETURN @attributeName
END
```

Fig. 3. The function getAttributeName

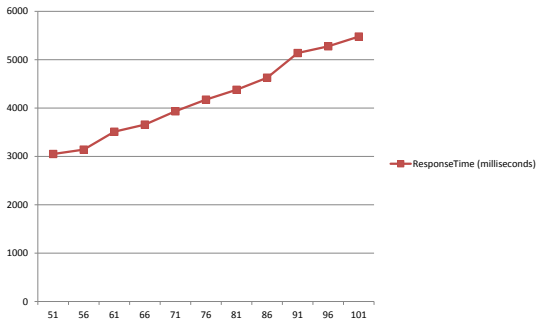


Fig. 4. SELECT query response time by number of dimensions (horizontal axis)

outlined above can only be used to compute the aggregated value one cell at a time. Hence, its performance will become less competitive as the set of cells becomes larger.

Lemma 1. *Disjointness of summary rowsets. Let* $s = (r, D, L, f, M_1)$ *be a summary state,* $L =< L_1, L_2, \ldots, L_n >$*, let* x *and* x' *be different slicer vectors for* s *and let* T *and* T' *be the corresponding summary rowsets for* x *and* x'*. Now,* $T \bigcap T' = \emptyset$*.*

Proof. Suppose, for contradiction, that $T \bigcap T' \neq \emptyset$. Let $t \in T \bigcap T'$. It is given that $x \neq x'$. Thus there exists some i, $1 \leq i \leq n$, such that $x_i \neq x'_i$. Let $l = L_i$ and consider the value in $t[A_l^i]$. Either $t[A_l^i] \neq x_i$ or $t[A_l^i] \neq x'_i$ or both. In the first case $t \notin T$, a contradiction. In the second case, $t \notin T'$, a contradiction. In the third case, $t \notin T$ and $t \notin T'$, a contradiction.

Thus $T \bigcap T' = \emptyset$.

A particular consequence of Lemma 1 is that the calculation of a summary value for different cells does not involve redundant computation.

Theorem 2. *Non redundant computation. The complexity of query processing based on Theorem 1 is O(n), where n is the number of tuples in r.*

Proof. By Lemma 1, a particular input cell is needed only for one aggregation result. Thus a query containing several cells on the same dimension level can be processed without accessing the same input cell twice. Further, in the worst case, it is enough to read each input cell, that is each tuple, once.

In a summarization instance, a summarization cell is null if its associated tuple set is empty. The number of null cells may be extremely large, but the number of non-null cells is limited by the number of rows in the decentralized relation of the dimension data. Each tuple t is in the tuple set of exactly one slicer vector, and that slicer vector can be formed from the values of t. It is possible to compute efficiently the partition of tuples into sets along with all related slicer vectors (i.e. the ones for which a non-null cell exists).

6 Conclusions

We propose a lazy evaluation method for dimensional analysis. Our method avoids unnecessary data access and it can be used e.g. for further analysis when approximate values are initially used. Our method gives satisfactory performance even in the case where there is an extremely large number of dimensions.

References

1. Abadi, D.J., Madden, S.R., Hachem, N.: Column-stores vs. row-stores: how different are they really? In: Proc SIGMOD 2008, pp. 967–980. ACM (2008)
2. Agarwal, S., et al.: On the computation of multidimensional aggregates. In: Proc. VLDB 1996, pp. 506–521. IEEE (1996)
3. Fu, L., Hammer, J.: Cubist: a new algorithm for improving the performance of ad-hoc olap queries. In: Proc. DOLAP 2000, pp. 72–79. ACM (2000)
4. Gao, Y., Ni, Z., Ni, L.: Compression of olap cubes for aggregate queries based on copula approach. In: Proc. BIFE 2010, pp. 67–71. IEEE Computer Society (2010)
5. Jacobs, A.: The pathologies of big data. CACM 52(8) (August 2009)
6. Li, X., Han, J., Gonzalez, H.: High-dimensional olap: a minimal cubing approach. In: Proc. VLDB 2004, pp. 528–539. VLDB Endowment (2004)
7. Morfonios, K., Konakas, S., Ioannidis, Y., Kotsis, N.: Rolap implementations of the data cube. ACM Comput. Surv. 39(4) (November 2007)
8. Russo, A., Ferrari, M.: C Webb. Microsoft SQL Server 2012 Analysis Services: The BISM Tabular Model. Microsoft Press (2012)
9. Shanmugasundaram, J., Fayyad, U., Bradley, P.S.: Compressed data cubes for olap aggregate query approximation on continuous dimensions. In: Proc. ACM SIGKDD 1999, pp. 223–232. ACM (1999)
10. Shin, S.K., Sanders, G.L.: Denormalization strategies for data retrieval from data warehouses. Decis. Support Syst. 42(1), 267–282 (2006)

MAF: A Method for Detecting Temporal Associations from Multiple Event Sequences

Han Liang

Department of Computing Science
University of Alberta
Edmonton, Canada, T6G 2E8

Abstract. In this paper, we propose a two-phase method, called *Multivariate Association Finder* (MAF), to mine temporal associations in multiple event sequences. It is assumed that a set of event sequences, where each event has an id and an occurrence time, is collected from an application. Our work is motivated by the observation that many associated events in multiple temporal sequences do not occur concurrently but sequentially. In an empirical study, we apply our method to two different application domains. Firstly, we use MAF to detect multivariate motifs from multiple time series data. Existing approaches all assume that the univariate elements of a multivariate motif occur synchronously. The experimental results on both synthetic and read data sets show that our method finds both synchronous and non-synchronous multivariate motifs. Secondly, we apply our method to mine frequent episodes from event streams. Current methods often ask users to provide possible lengths of frequent episodes. The results on neuronal spike simulation data show that MAF automatically detects episodes with variable time delays.

1 Introduction

Nowadays, more and more temporal data in the form of event sequences is being generated. Each distinct event sequence consists of events of the same type, where each event has an id and an occurrence time. In practice associated events in different event sequences do often not occur concurrently but with a temporal lag. For example, in human-computer interaction modeling, an event sequence represents actions taken by users during a period of time and the goal is to capture aspects such as user intent and interaction strategy by understanding causative chains of connections between actions.

We propose a two-phase method, called *Multivariate Association Finder* (MAF), to find temporal associations in multiple event sequences. First, we detect bivariate associations from pairs of event sequences by comparing the observed distribution of temporal distances of event occurrences with a null distribution. Second, using the graph of bivariate associations, we search for multivariate associations whose frequencies are statistically significant.

In an empirical study, we applied our method to two application domains. Firstly, we used MAF to detect multivariate time series motifs. In a univariate time series, a *motif* is a set of time series subsequences that exhibit high

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2013, LNCS 8057, pp. 359–371, 2013.

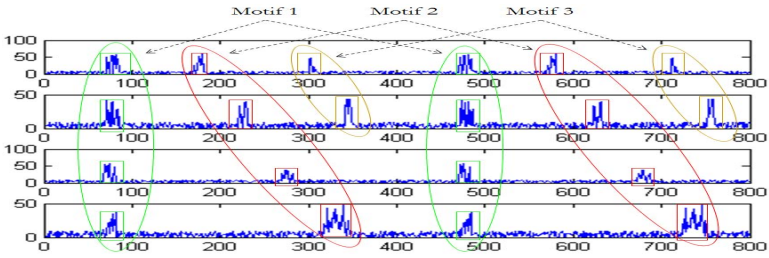


Fig. 1. Illustration of three multivariate motifs. An ellipse represents a multivariate motif occurrence, and a rectangle denotes a univariate element.

similarity and occur frequently in the whole time series [1]. In a d-dimensional multivariate time series containing d univariate time series, a n-dimensional *multivariate motif* is a set of n-dimensional tuples of univariate elements, where the univariate elements from different dimensions have a temporal association, i.e., they occur concurrently as a *synchronous* multivariate motif (e.g., motif 1 in Figure 1) or sequentially as a *non-synchronous* multivariate motif (e.g., motif 2 and motif 3 in Figure 1). Existing methods are all limited by assuming that the univariate elements of a multivariate motif occur synchronously. The experimental results confirm that MAF successfully discovers both synchronous and non-synchronous multivariate motifs. Secondly, we use our method to discover frequent episodes from event streams. Frequent episode discovery is a framework for detecting temporal patterns in symbolic temporal data [3]. The input data of this framework is a sequence of event occurrences with each characterized by an event type and an occurrence time. The detected temporal patterns, referred to as *episodes*, are essentially small, temporally ordered sets of event types. Depending on different types of temporal orders over their event types, episodes are classified into two categories: serial episodes and parallel episodes. A *serial episode* requires its event types to occur sequentially. A *parallel episode* does not require any specific ordering of the event types. Current methods all require users to provide possible lengths of frequent episodes. Our experimental results show that MAF is very effective in detecting episodes with variable lengths.

The reminder of the paper is organized as follows. Section 2 reviews the related work of multivariate motif detection and frequent episode discovery. Section 3 gives basic definitions. Sections 4 describes our method. Sections 5 and 6 present experimental settings and results. Section 7 concludes the paper.

2 Related Work

Multivariate Motif Discovery. Current methods of multivariate motif discovery can be classified into three categories.

(1) Representing a multivariate time series as a set of multi-dimensional points. Methods in this group treat each univariate time series as a dimension

and retrieve a set of d-dimensional points from d equal-length univariate time series. Minnen *et al.* proposed a method that represents data points symbolically based on a vector quantization and uses a suffix tree to locate motif seeds [7]. Their later work located multivariate motifs as regions of high density in the d-dimensional space [6]. Multivariate motifs discovered by these methods must span all dimensions and their univariate elements must be equally sized.

(2) Transforming a multivariate time series into a univariate time series. Tanaka *et al.* used *Principal Component Analysis* (PCA) to transform a multivariate time series into a univariate time series and applied a *Minimum Description Length* (MDL) principle on the projected time series to extract univariate motifs [12]. To handle multivariate time series data, other work in this category extend *Symbolic Aggregate Approximation* (SAX), a technique to reduce the dimensionality of univariate time series subsequences [1]. Minnen *et al.* developed a method that applies SAX on each of the univariate time series and concatenates SAX words from each dimension occurring together in a sliding window [8]. These methods all implicitly assume that the univariate elements in a multivariate motif must be completely synchronous.

(3) Combining a set of univariate motifs into a multivariate motif. Vahdatpour *et al.* constructs a coincidence graph based on the temporal relations of univariate motifs [13]. A graph is initially built, where a vertex represents a univariate motif and the weight of an edge between two vertexes indicates the frequency with which the occurrences of the two corresponding univariate motifs temporally overlap. Starting from the motif with the highest occurrences, a graph clustering algorithm iteratively detects multivariate motifs by comparing the weights of edges connected to this motif to a user-defined threshold. This method allows the univariate motifs to have different lengths and permits that multivariate motifs can span only a subset of dimensions.

Frequent Episode Discovery. The methods applying the framework of frequent episode discovery to neuronal spike data are classified into two categories.

(1) Mining serial and parallel episodes using an Apriori-style procedure. The methods utilize an Apriori-style procedure to detect serial episodes. Mannila *et al.* first presented the framework of frequent episode discovery [3]. The frequency of an episode is defined as the number of sliding windows in which the episode occurs. Laxman *et al.* proposed their work based on a new frequency measurement, which counts the number of non-overlapped occurrences for an episode [2]. These methods all limit their searching scope by requiring users to provide the size of sliding windows or specify an inter-event time constraint for every pair of successive event types in an episode.

(2) Mining statistically significant episodes. The algorithms detect statistically significant serial episodes. Sastry *et al.* designed a statistical test to determine the significance level of discovered frequent episodes, based on the intuition that the interaction between two event sequences can be captured by the conditional probability of observing an event from one sequence after a time delay given that an event has occurred on the other sequence [11]. Patnaik *et al.* presented another

approach by proposing what they so called "*excitatory dynamic networks*" (EDNs), where nodes denote event types and edges represent temporal associations among nodes [9]. The authors also defined their so-called "*fixed-delay episode*", where the time-delay between every pair of event types in an episode is fixed. To obtain the marginal probabilities for an EDN, the occurrence-based frequencies of fixed-delay episodes are used to compute the probabilities for each node. These methods need users to specify an inter-event time-delay for every pair of successive event types in an episode.

3 Background and Definitions

An *event sequence* $\xi =< e_1, e_2, \ldots, e_m >$ is an ordered set of m events e_i. Each e_i in ξ denotes a tuple (e_id, t_i), where e_id represents the event id and t_i is the occurrence time of the event. All event occurrences in ξ are of the same type.

We introduce a *bivariate association* A^d_{ab} $(a \neq b)$, between two event sequences ξ_a and ξ_b, as a subset of the Cartesian product of ξ_a and ξ_b, as following:

Definition 1. *Let ξ_a and ξ_b be two event sequences. A set $A^d_{ab} \subseteq \xi_a \times \xi_b$ is called a bivariate association in (ξ_a, ξ_b) with mean temporal distance d if for all $(e, e') \in A^d_{ab} : t \leq t' \wedge t' - t \sim \Phi(\cdot) \wedge E(t' - t) = d$, and there is a one-to-one correspondence between the sets $\{e|\exists e' : (e, e') \in (A^d_{ab})\}$ and $\{e'|\exists e : (e, e') \in (A^d_{ab})\}$, where t (resp. t') is the occurrence time of event e (resp. e'), $\Phi(\cdot)$ denotes a a known distribution (e.g., uniform or Gaussian) that the temporal distance between two associated events follows, and $E(t' - t) = d$ is the expected temporal difference between associated events in A^d_{ab}.*

In this paper, we assume that the temporal distance of two associated events follows a Gaussian distribution. This assumption holds in many applications. For example, in neuronal spike train analysis, one type of interesting patterns is called *Ordered chains*, which are ordered firing sequences of neurons where times between firing of successive neurons are often assumed to follow a Gaussian.

A *multivariate association* $MA^{d_1 \ldots d_{k-1}}_{1 \ldots k}$ between k event sequences $\xi_1, \ldots, \xi_k$ is defined as:

Definition 2. *Let $\xi_1, \ldots, \xi_k$ be k different event sequences. A set $MA^{d_1 \ldots d_{k-1}}_{1 \ldots k} \subseteq \xi_1 \times \ldots \times \xi_k$ is called a multivariate association in $(\xi_1, \ldots, \xi_k)$ if for all $(e^1, \ldots, e^k) \in MA^{d_1 \ldots d_{k-1}}_{1 \ldots k} : (e^i, e^{i+1})$ is an instance of a bivariate association in (ξ_i, ξ_{i+1}) with mean temporal distance d_i for all $1 \leq i \leq k - 1$.*

For our approach, we assume that the event sequences collected from an application can be modeled as Poisson processes. Two important properties of a Poisson process are (i) the **inter-arrival times** T_i between consecutive event occurrences are independent and follow an exponential distribution with rate $\mu = 1/\lambda$, where λ denotes the intensity of the Poisson process, and (ii) the ith **arrival times** S_i, i.e., the times until the ith event occurrence from the starting point of the process, follow a Gamma distribution with shape parameter $\alpha = i$ and scale parameter $\beta = \lambda$.

4 Proposed Method

Bivariate Association Discovery. To determine whether two event sequences ξ_a and ξ_b are temporally associated, we analyze what we define as *forward distances*, which are the difference in time between the events $e \in \xi_a$ and the events $e' \in \xi_b$ occurring after e.

Definition 3. *The set of forward distances between event sequences ξ_a and ξ_b is given as $FD_{ij} = \{dist | \exists e \in \xi_a \exists e' \in \xi_b, t \leq t' \wedge dist = t' - t\}$, where t (resp. t') is the occurrence time of e (resp. e').*

To compute the forward distances for an event on a sequence ξ_a, to events in a sequence ξ_b, we can think of projecting the event onto sequence ξ_b and denoting the projected position as h. The forward distance from h to its right nearest event on sequence ξ_b can be denoted as Z_1. Since we compute Z_1 for each event of sequence ξ_a, Z_1 can be treated as a random variable whose distribution can be derived from the so-called *Waiting Time Paradox* for Poisson processes [5]. The theorem says that Z_1 follows the same exponential distribution as the inter-arrival times on sequence ξ_b, with mean $\mu = 1/\lambda_b$, and the forward distances Z_j from time h to the jth event after h, can be modeled as the arrival times of the jth event of a Poisson process starting at time h, following a Gamma distribution with shape parameter $\alpha = j$ and scale parameter $\beta = \lambda_b$.

Knowing the distribution of the individual forward distances, we can express the distribution of all forward distances x from all events on a finite sequence ξ_a to all events on a finite sequence ξ_b as a mixture of these individual distributions:

$$f_n(x) = \sum_{j=1}^{N} W_j \times g(x, j, \lambda_b), \tag{1}$$

where $f_n(x)$ is our expected null distribution of forward distances (i.e., when there is no temporal association), N is the number of individual distribution components (equal to the number of forward distances the first event on sequence ξ_a has), $g(x, j, \lambda_b)$ is the Gamma distribution that Z_j follows, and W_j represents the weight of the jth component. Figure 2 illustrates how we estimate the weights for each component density from the properties of the involved Poisson processes. In the figure, S_n is the arrival time of the last event on sequence ξ_b and there are k events on sequence ξ_a that occur before S_n. S_{n-j+1} denotes the arrival time of the jth last event on sequence ξ_b. Every event on ξ_a that occurs before S_{n-j+1} will have all forward distances to events on ξ_b up to and including their jth right neighbor. However, every event on ξ_a after S_{n-j+1} will not have a distance to their jth right neighbor and will not contribute a distance to Z_j. If T_j denotes the time interval between S_{n-j+1} and S_n, its expected length $E(T_j)$ is $(j-1)/\lambda_b$. The expected number of events on sequence ξ_b that are in time interval $E(T_j)$ can be estimated by $(j-1)\lambda_a/\lambda_b$. Let N_j represent the number of distances in Z_j; its expected number $E(N_j)$ can be estimated as $k - [(j-1)\lambda_a/\lambda_b]$. Hence, we can estimate each weight W_j by $E(N_j)/\sum_{i=1}^{N} E(N_i)$.

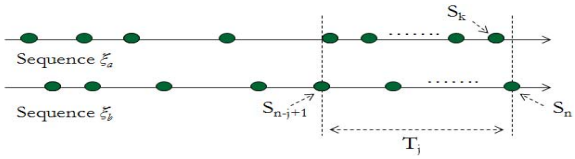


Fig. 2. Determining the Weights of Individual Gamma Distribution Components

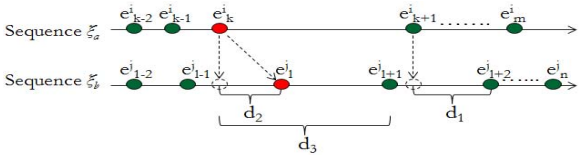


Fig. 3. Three Categories of Forward Distances

Given that the temporal distance of two associated events follows a Gaussian distribution, we can derive a theoretical distribution function of forward distances if two event sequences are temporally associated. Figure 3 illustrates a general scenario of two event sequences where embedded pairs of associated events occur. In this figure, the red points linked by the directed dashed line represent a pair of truly associated events and the green points stand for event occurrences that are independently distributed. If there exist associated pairs of events between two event sequences, three categories of forward distances can be generated by events from different sequences.

The first category of forward distances consists of the distances from an event occurrence that is independently distributed on sequence ξ_a, e.g., the distance d_1 between events e^i_{k+1} and e^j_{l+2} in Figure 3. Since e^i_{k+1} is not temporally associated with any event on sequence ξ_b, its projected position h will be independently distributed regarding all of the events on sequence ξ_b. From the *Waiting Time Paradox* for Poisson processes we learn that the forward distance from position h to its jth right nearest event occurrence on sequence ξ_b follows a Gamma distribution with shape parameter $\alpha = j$ and scale parameter $\beta = \lambda_b$. Therefore, the distribution of the forward distances computed at e^i_{k+1} can be described by a mixture of these Gamma distribution components, which is exactly the expected null distribution $f_n(x)$. Hence, we can use $f_n(x)$ to characterize the distribution of the forward distances in the first category.

The forward distances in the second category are the temporal distances of associated event occurrences, e.g., the distance d_2 between events e^i_k and e^j_l in Figure 3. Since the temporal distance of two associated events follows a Gaussian distribution, if there exists more than one temporal association between two event sequences, we can describe the distribution of distances generated from associated pairs of events as a mixture of Gaussian distributions:

$$f_g(x) = \sum_{p=1}^{N} W_p \times Gaussian(x, \mu_p, \delta_p), \tag{2}$$

where N represents the number of Gaussian components (equal to the number of temporal associations existing in the data), $Gaussian(x, \mu_p, \delta_p)$ denotes the pth Gaussian component with mean μ_p and standard deviation δ_p. W_p denotes the weight of the pth Gaussian component and it can be estimated by $k_p / \sum_{p=1}^{N} k_p$, where k_p represents the number of forward distances following this Gaussian.

The forward distances in the last category are from associated events on ξ_a to events independently distributed on ξ_b, e.g., the distance d_3 from e_k^i to e_{l+1}^j in Figure 3. Since e_k^i is only temporally associated with e_l^j on ξ_b, the projected position of e_k^i is independently distributed regarding all of the events but e_l^j. We can use $f_n(x)$ to describe the distribution of these distances approximately.

Based on the previous analysis, we can derive a theoretical distribution of forward distances generated from two event sequences as:

$$f_t(x) = (\frac{T - \sum_{p=1}^{N} k_p}{T}) f_n(x) + (\frac{\sum_{p=1}^{N} k_p}{T}) f_g(x), \tag{3}$$

where T denotes the total number of forward distances computed between the two sequences, N represents the number of temporal associations and c denotes the number of distances computed at each event occurrence. If k_p denotes the number of temporal distances in the pth temporal association, we can calculate $c \times \sum_{p=1}^{N} k_p$ distances from the associated events on sequence ξ_a, so approximately $T - \sum_{p=1}^{N} k_p$ distances follow the expected null distribution $f_n(x)$ and around $\sum_{p=1}^{N} k_p$ distances follow $f_g(x)$. Finally, we determine the normalized weight for each of the distribution components in $f_t(x)$ by dividing the number of distances following this component by the total of observed distances.

The proposed approach for bivariate association detection is based on the observation that the mean and the standard deviation of a Gaussian distribution can be estimated by using the zero-crossing points of its second derivative curve. We treat bivariate association discovery as a least squares curve-fitting problem, where we adjust the means and the standard deviations of the Gaussian components in the theoretical function to optimally fit the curve of the actual, observed distribution of forward distances. We estimate the observed distribution by using a kernel density estimation method. A statistically significant region is identified if there exists in the observed distribution a bell-shaped portion with a statistically significant higher count of forward distances than expected.

We design a statistical test to determine the probability that a region R contains the observed number $ON(R)$ of forward distances under the null hypothesis (forward distances are distributed according to $f_n(x)$). The probability, P_R, that a randomly chosen forward distance falls into R, under the null hypothesis, can be derived as following:

$$P_R = \int_l^u f_n(x) dx \tag{4}$$

where l and u denote the lower and upper bound of R, respectively. Given n observed forward distances, the distribution of the test statistic $ON(R)$ under the null hypothesis can be modeled by a Bernoulli experiment, repeated independently n times with a success probability of P_R. Consequently, P_R follows a

binomial distribution with parameters n and P_R. Let α_0 be a significance level. Let α be the probability that we observe $ON(R)$ under the null hypothesis. R is statistically significant at significance level α_0 if $\alpha \leq \alpha_0$.[1]

Algorithm 1 is used to detect statistically significant regions from the observed distribution. We first generate the second derivative curve of the observed distribution and obtain the pairs of zero-crossing points from the resulting curve. For each pair (x_l, x_r), we use our statistical test to check if the region $[x_l, x_r]$ in the observed distribution is "*statistically significant*", assuming that all the forward distances are distributed according to $f_n(x)$. Suppose m pairs are statistically significant. We will then approximate $f_t(x)$ by using m Gaussian distributions. For each statistically significant pair (x'_l, x'_r), we estimate the mean of the Gaussian distribution by averaging the values of these two points:

$$\mu = \frac{x'_l + x'_r}{2}. \tag{5}$$

Accordingly, we determine the standard deviation of this Gaussian by taking half of the absolute value of the difference between the two points:

$$\sigma = \frac{|x'_l - x'_r|}{2}. \tag{6}$$

Once these parameters have been initially estimated, we determine the number of forward distances following each Gaussian distribution in $f_t(x)$ by using the system of linear equations:

$$\sum_{x_j \in U} f_t(x_j) \equiv \sum_{x_j \in U} f_o(x_j), \tag{7}$$

where x_j stands for a forward distance and U denotes the set of the distances calculated from two event sequences. After assigning initial values to the Gaussian distributions' parameters in $f_t(x)$, we apply the *Levenberg-Marquardt* algorithm [4] to adjust these parameters. Finally, we check each of the regions defined by the Gaussian parameters in $f_t(x)$ and output it if statistically significant.

Given a statistically significant region R, we assume that there is a true bivariate association A^d_{ab} contained in the two corresponding sequences ξ_a and ξ_a; we estimate the mean temporal distance d of A^d_{ab} as the mean μ of the Gaussian distribution that the distances inside R follow, and we use the standard deviation σ of this Gaussian to define the range where we extract the actual associated event occurrences. For an event occurrence on ξ_a, we only consider the event occurrences on ξ_b that are in the time interval $[\mu - 3 \times \sigma, \mu + 3 \times \sigma]$.

Multivariate Association Discovery. If two event sequences ξ_a and ξ_b are temporally associated their relationship can be expressed as: $\xi_a \overset{50}{\rightarrow} \xi_b$, where the directed line represents the temporal order of these two event sequences and the number (i.e. 50) above the directed line denotes the expected mean temporal distance of their associated pairs of events. We construct a directed graph based on discovered bivariate associations. In the graph, a vertex denotes an event

[1] To avoid false positives, we perform a Bonferroni adjustment by setting the significance level to α_0/m, where m denotes the number of tests.

Algorithm 1. Approximation Algorithm (AA)

Input: $f_o(x)$ - the observed distribution of forward distances; $f_n(x)$ - the expected null distribution of forward distances
Output: S - a set of statistically significant regions

1: Generate the second derivative curve of $f_o(x)$ and obtain the pairs of zero-crossing points on the curve;
2: **for** each pair of zero-crossing points (x_l, x_r) **do**
3: Check if the region $[x_l, x_r]$ in $f_o(x)$ is "statistically significant", assuming that all the forward distances are distributed according to $f_n(x)$;
4: Suppose m regions are statistically significant. Estimate the parameters of the m Gaussian distributions by using Equations 5, 6 and 7;
5: Approximate $f_t(x)$ using these Gaussian distributions and their parameters are adjusted by the *Levenberg-Marquardt* algorithm;
6: **for** each Gaussian distribution $\Phi(\mu, \sigma)$ in $f_t(x)$ **do**
7: **if** the region $[\mu - \sigma, \mu + \sigma]$ in $f_o(x)$ is statistically significant **then**
8: Store pair (μ, σ) into S;
9: **return** S

sequence and an edge represents the association between two event sequences. We search for all of the pathes existing between any of two vertexes V_a and V_b and store them in a result set. We rank these pathes in terms of the number of vertexes (or the path length) each path has. Starting from the longest path, for two consecutive pairs of neighbor vertexes (e.g., given a path $V_1 \rightarrow V_2 \rightarrow V_3$, the pair $V_1 \rightarrow V_2$ and the pair $V_2 \rightarrow V_3$ are consecutive) we check how many occurrences on the shared sequence (e.g., the sequence denoted by V_2) are used by both pairs. We stop checking when the count is smaller than a threshold. A path is outputted as a multivariate motif if it can pass the verification on each of its internal vertexes. We prune all of the short-cuts (e.g., the path $V_1 \rightarrow V_3$ is a short-cut of the path $V_1 \rightarrow V_2 \rightarrow V_3$) of a path from the result set. We start a new search for the longest path never processed in the result set and repeat the previous steps until all of the pathes in the result set have been processed.

5 Empirical Study on Multivariate Motif Discovery

We apply our method to detect multivariate motifs from multiple time series sequences and compare it with Vahdatpour's method (VAH) [13], which is one of the most effective methods of multivariate motif discovery.

To evaluate the performances of compared methods, we conducted three groups of experiments on synthetic data sets. In each group, we generated a set of univariate time series with a length of 2×10^7 time units, where varying numbers of occurrences of a multivariate motif and "*noise*" univariate motif occurrences were implanted. Both the length of noise univariate motif occurrences and the length of univariate elements in the multivariate motif equaled 20 time units. We used F-measure to evaluate the performances of compared methods.

In the first group of experiments, we created synthetic data sets containing five randomly generated univariate time series, where a 5-variate motif and some noise univariate motif occurrences were implanted. The 5-variate motif consisted of 4 bivariate motif components, each of which had a fixed standard deviation of temporal distances equal to 2.5 time units. We varied both the percentage of 5-variate motif occurrences from 10% to 100%, and the mean temporal distance between 10 and 5000 time units. Table 1(a) shows that MAF not only detected this multivariate motif when its univariate elements temporally overlap, but also found it as its univariate elements had varying temporal lags. Table 1(b) shows that Vahdatpour's method detected nothing when the univariate elements of this implanted multivariate motif were non-synchronous.

Table 1. % of Multivariate Motif Occurrences vs. Mean Temporal Distance

(a) Experimental Results of MAF

Per. \ Mean	10	20	200	1000	5000
10%	0.947	0.945	0.941	0.944	0.942
30%	0.952	0.956	0.949	0.953	0.957
50%	0.965	0.960	0.965	0.963	0.967
70%	0.974	0.973	0.977	0.974	0.972
90%	0.987	0.991	0.988	0.985	0.986

(b) Experimental Results of VAH

Per. \ Mean	10	20	200	1000	5000
10%	0.889	0.496	0.0	0.0	0.0
30%	0.959	0.519	0.0	0.0	0.0
50%	0.974	0.531	0.0	0.0	0.0
70%	0.985	0.553	0.0	0.0	0.0
90%	0.990	0.564	0.0	0.0	0.0

In the second group, we varied both the percentage of 5-variate motif occurrences from 10% to 100%, and the standard deviation of the bivariate motif components between 10 and 100 time units. Each bivariate motif component had now a fixed mean of temporal distances equal to 500 time units. Table 2 shows that our method successfully detected the multivariate motif in most of the cases. Vahdatpour's method detected nothing from all of the cases.

Table 2. % of Motif Occurrences vs. Standard Deviation of Temporal Distances

Per. \ Vari.	2.5	5	10	20	30
10%	0.946	0.923	0.911	0.751	0.592
30%	0.950	0.947	0.938	0.889	0.747
50%	0.966	0.953	0.941	0.926	0.880
70%	0.972	0.964	0.955	0.943	0.904
90%	0.982	0.973	0.965	0.952	0.922

Finally, we generated a complex synthetic data set of ten randomly generated univariate time series, where we implanted five multivariate motifs. Each multivariate motif had 1000 occurrences. We also added 5000 noise univariate motif occurrences to each dimension in the data set. Table 3 lists the properties of the implanted multivariate motifs. MAF obtained scores of 1.0 for the bivariate and the 3-variate motifs, 0.995 for the 5-variate motif, 0.989 for the 8-variate motif and 0.988 for the 10-variate motif. Compared with our method, Vahdatpour's method detected none of the non-synchronous multivariate motifs.

Table 3. The Properties of Implanted Multivariate Motifs

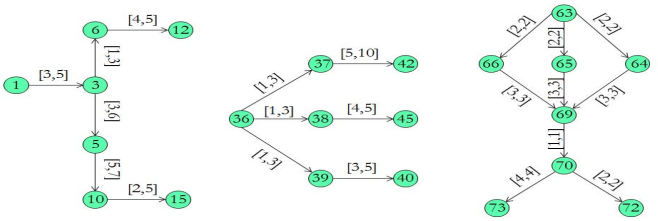
Properties \ Motifs	bi-variate	3-variate	5-variate	8-variate	10-variate
Mean	2	20	800	3000	5000
Standard Deviation	0.5	2.5	10	10	5
Dimensions	1-2	3-5	1-5	1-8	1-10

We further evaluated MAF and Vahdatpour's method using a real-world data set where non-synchronous multivariate motifs exist. The data set consists of recordings of shovel operations provided by an industrial company. We attempted to detect a variety of patterns, such as dig-cycles. The power consumed by three motors (i.e., Crowd power, Hoist power, and Swing power) was recorded as a time series and the power profiles of the motors could provide information about the shovel's activities. MAF detected several non-synchronous multivariate motifs. Compared with our method, Vahdatpour's method detected only a multivariate motif that was a synchronous subset of a larger motif detected by MAF. We are currently in the process of characterizing and interpreting the usefulness of the temporal associations found in this data set.

6 Empirical Study on Frequent Episode Discovery

We also applied our method to discover frequent episodes from neural spike train data and compared it with Patnaik's method (PAT) [9], which is currently the most effective method for detecting temporal patterns from spike train data. We evaluated MAF and Patnaik's method on simulation data collected from a mathematical model of spiking neurons [10]. This model allows for temporal associations with variable time delays of associated spikes, which mimic the situation in conduction pathways of real neurons. Each generated spike train (a sequence of spikes made by a neuron) follows an inhomogeneous Poisson process. We use this model to assess the performances of compared methods in discovering three episodes implanted into several simulation data sets. Figure 4 illustrates these episodes, where nodes denote spike trains and directed arcs represent temporal orders of firing spikes among trains. For each episode the values above a directed arc indicate the range of time delays between associated spikes. Table 4 lists the properties of these data sets. The third column shows the base firing rate of neurons and the fourth column presents the activation probability of a neuron. We arrange these data sets into three groups and use F-measure to evaluate the performances of compared methods.

First, we summarize the experimental results on the data sets of A-group. We created these data sets by changing the base firing rate $\hat{\lambda}_0$ of neurons in the mathematical model. The larger value we assign to $\hat{\lambda}_0$, the more spikes are generated on a train. Table 5(a) shows that MAF successfully detected all the implanted episodes by achieving high scores in all of the cases. Table 5(b) presents that: although Patnaik's method was effective in finding the fixed-delay episode, it detected nothing when the time delay of two associated spikes varies.



(a) Serial Episode (b) Parallel Episode (c) Fixed-delay Episode

Fig. 4. Three Episodes Implanted into Synthetic Data Sets

Table 4. Spike Train Simulation Data Sets

(a)

Name	Length (ms)	Base Fir. Rate $\hat{\lambda}_0$	Act. Prob. ρ
A_1	60000	0.01	0.9
A_2	60000	0.015	0.9
A_3	60000	0.02	0.9
A_4	60000	0.025	0.9
B_5	60000	0.02	0.8
B_6	60000	0.02	0.85

(b)

Name	Length (ms)	Base Fir. Rate $\hat{\lambda}_0$	Act. Prob. ρ
B_7	60000	0.02	0.9
B_8	60000	0.02	0.95
C_9	60000	0.02	0.9
C_{10}	90000	0.02	0.9
C_{11}	120000	0.02	0.9

Table 5. Implanted Episodes vs. Base Firing Rate $\hat{\lambda}_0$

(a) Experimental Results of MAF

Episode Types \ $\hat{\lambda}_0$	0.01	0.015	0.02	0.025
Serial Episode	0.999	0.998	0.997	0.996
Parallel Episode	0.999	0.999	0.998	0.996
Fixed-delay Episode	1.0	1.0	0.999	0.998

(b) Experimental Results of PAT

Episode Types \ $\hat{\lambda}_0$	0.01	0.015	0.02	0.025
Serial Episode	0.0	0.0	0.0	0.0
Parallel Episode	0.0	0.0	0.0	0.0
Fixed-delay Episode	1.0	1.0	1.0	1.0

Table 6. Implanted Episodes vs. Activation Probability ρ

(a) Experimental Results of MAF

Episode Types \ ρ	0.8	0.85	0.9	0.95
Serial Episode	0.997	0.996	0.997	0.997
Parallel Episode	0.998	0.998	0.998	0.997
Fixed-delay Episode	1.0	1.0	0.999	0.999

(b) Experimental Results of PAT

Episode Types \ ρ	0.8	0.85	0.9	0.95
Serial Episode	0.0	0.0	0.0	0.0
Parallel Episode	0.0	0.0	0.0	0.0
Fixed-delay Episode	1.0	1.0	1.0	1.0

Second, we evaluate the two methods using the data sets of B-group. This time we created the data sets by varying the activation probability ρ of a neuron. The larger value we set to ρ, the more occurrences of an episode are implanted into the data. Table 6(a) shows that our method successfully discovered these implanted episodes from the data. Table 6(b) presents that Patnaik's method found neither serial nor parallel episodes.

Finally, we evaluate the performances of these methods as the data set length was varied. Table 7(a) shows that our method worked constantly well by achieving high scores in all of cases. Table 7(b) shows that Patnaik's method still failed to detect either serial or parallel episodes.

Table 7. Implanted Episodes vs. Data Set Length L

(a) Experimental Results of MAF

Episode Types \ Length(ms)	60000	90000	120000
Serial Episode	0.998	0.998	0.997
Parallel Episode	0.999	0.998	0.999
Fixed-delay Episode	0.999	0.999	0.999

(b) Experimental Results of PAT

Episode Types \ Length(ms)	60000	90000	120000
Serial Episode	0.0	0.0	0.0
Parallel Episode	0.0	0.0	0.0
Fixed-delay Episode	1.0	1.0	1.0

7 Conclusion and Future Work

We presented a general, statistical method for finding temporal associations in multiple event sequences. In an empirical study, we first used it to detect multivariate motifs. The results on both synthetic and real data showed that our method found both synchronous and non-synchronous multivariate motifs. We then applied our method to discover frequent episodes from event streams. The results on neuronal spike simulation data presented that our method effectively discovered episodes with variable lengths. In future work, we will investigate other application domains, such as network monitoring.

References

1. Chiu, B., Keogh, E., Lonardi, S.: Probabilistic discovery of time series motifs. In: KDD 2003, pp. 493–498 (2003)
2. Laxman, S., Sastry, P., Unnikrishnan, K.: A fast algorithm for finding frequent episodes in event streams. In: KDD 2007, pp. 410–419 (2007)
3. Mannila, H., Toivonen, H., Verkamo, A.: Discovering frequent episodes in sequences. In: KDD 1995, pp. 210–215 (1995)
4. Marquardt, D.: An algorithm for least-squares estimation of nonlinear parameters. Journal of the Society for Industrial and Applied Mathematics 11, 431–441 (1963)
5. Meester, R.: A Natural Introduction to Probability Theory (2004)
6. Minnen, D., Isbell, C., Essa, I.: Discovering multivariate motifs using subsequence density estimation and greedy mixture learning. In: AAAI 2007, pp. 615–620 (2007)
7. Minnen, D., Starner, T., Essa, I., Isbell, C.: Discovering characteristic actions from on-body sensor data. In: ISWC 2006, pp. 11–18 (2006)
8. Minnen, D., Starner, T., Essa, I., Isbell, C.: Improving activity discovery with automatic neighborhood estimation. In: IJCAI 2007, pp. 2814–2819 (2007)
9. Patnaik, D., Laxman, S.: Discovering excitatory networks from discrete event streams with applications to neuronal spike train analysis. In: ICDM 2009, pp. 407–416 (2009)
10. Raajay, V.: Frequent episode mining and multi-neuronal spike train data analysis. Master's thesis, IISc, Bangalore (2009)
11. Sastry, P., Unnikrishnan, K.: Conditional probability-based significance tests for sequential patterns in multineuronal spike trains. Neural Computation (2010)
12. Tanaka, Y., Iwamoto, K., Uehara, K.: Discovery of time-series motif from multi-dimensional data based on mdl principle. Machine Learning 58, 269–300 (2005)
13. Vahdatpour, A., Amini, N.: Toward unsupervised activity discovery using multi-dimensional motif detection in time series. In: IJCAI 2009, pp. 1261–1266 (2009)

Author Index