Anul LIV **2009**

S T U D I A
UNIVERSITATIS BABEŞ-BOLYAI

INFORMATICA

2

Redacția: M. Kogălniceanu 1 • 400084 Cluj-Napoca • Tel: 0264.405300

SUMAR – CONTENTS – SOMMAIRE

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

PROFESSOR LEON TAMBULEA AT HIS SIXTIES

FLORIAN M. BOIAN AND MILITON FRENŢIU

Professor Leon Tambulea graduated the Faculty of Mathematics-Mechanics, section of Computer Science 37 years ago, on June 1972. Due to his remarkable results as a student he has been offered a position of Assistant Professor at our faculty. Since then we have been colleagues, and Leon has actively participated in the development of this new scientific field of Computer Science for all these years.

During these years he stepped through all the didactic positions: Assistant Professor since 1972, Lecturer since 1990, Associate Professor since 1992 and, since 1995, full Professor in Informatics Systems. He obtained his Ph.D. degree in 1985, under the supervision of Prof.Dr. D.D.Stancu, with the thesis Mathematical Models For Information Structures. Since 2001 he has been a Ph.D. supervisor himself.

As a professor, he is known for a high level of scientific knowledge at his courses. Also, he proved to be a distinguished pedagogue, very appreciated by his students. He introduced and taught many courses for the first time in our university, in a period without the Internet, and scarce bibliography resources. He was and continues to be an active participant at the huge evolution of Computer Science in our faculty during the last 37 years.

Professor Tambulea has important contributions in many fields of Computer Science, and in the education of many generations of computer scientists. Briefly we mention the following:

- Didactic activities in 21 disciplines (lectures, seminaries, laboratories);
- 16 published books, university manuals;
- 66 published scientific papers;
- Many conference participations in Computer science.

He has participated at all activities of our faculty and university:

- Director or member of many research grants, or industrial contracts;
- Director of endowment grants, having as main beneficiaries our students;
- Participation in many committees and juries of the students competitions in Computer Science;
- Participation in many committees of school children in olympiads;

- Executive Editor of the journal "Studia Universitatis Babes-Bolyai", series Mathematica-Informatica until 1996;
- Member in the Editorial Board of the journal "Studia Universitatis Babes-Bolyai", series Informatica since 1996;
- Director of a master program in Databases;
- Member and organizer of many examination teams: license, dissertations, PhD;
- Promoter of the organization of our computers network;
- Created many computer programs for managing the various information about the students of our faculty.

The main research interest of Professor Tambulea is databases. He has been, as well, interested in computer graphics and web programming, as we may notice from the list of published papers.

Due to his qualities of good colleague and organizer, Professor L. Tambulea was elected dean of the faculty for three different terms. Moreover, Professor L. Tambulea fulfilled many administrative managerial activities:

- Director of the Babes-Bolyai University Computer Center (1992-1993);
- Vice-Dean of the Faculty of Mathematics and Computer Science (1992-1996);
- Head of Chair of Information Systems (1996-2000, 2000-2004, 2004-2007);
- Dean of Faculty of Mathematics and Computer Science (1996-2000, 2000-2004, since 2008);
- Member of many working groups of the Ministry of National Education: CNCSIS (2004-2005), ONBSS (2006-2008), CNATDU (2000-2003, since 2006).

We, and all his colleagues do appreciate his hard work and effort to teach our students and to develop our didactic and scientific activity in Computer Science.

We wish Professor Tambulea a happy and healthy long life, full of achievements, for the benefit of all faculty members and for his family.

Scientific Activity

Journal and Conference Papers.

(1) Tambulea L., and A. Sabau, From Databases to Semantic Web, STUDIA UNIVERSITATIS BABES-BOLYAI, Special Issue for International Conference KEPT 2009, Knowledge Engineering Principles and Techniques, Cluj-Napoca, July 2-4, 2009, pp.85-88.

(2) Tambulea L., and M. Horvat, Redistributing fragments into a distributed database, Int. J. of Computers, Communications and Control, Vol. 3 (2008), Proceedings of ICCCC 2008, pp. 11-16.
(3) Tambulea L., and M. Horvat, Dynamic Distribution Model in Distributed Database, Int. J. of Computers, Communications Control, ISSN 1841-9836, E-ISSN 1841-9844, Vol. III (2008), Suppl. issue: Proceedings of ICCCC 2008, pp. 512-515.
(4) Tambulea L., and I. Gansca, Generalized Cylinders Surfaces, Studia Univ. Babes-Bolyai, Informatica, Volume LII, Number 2, 2007, 69-78.
(5) Tambulea L., and M. Frentiu, Professor Florian Mircea BOIAN at his sixties, Studia Univ. Babes-Bolyai, Informatica, Volume LII, Number 2, 2007, 3-12.
(6) Tambulea L., and H. F. Pop, Management of Web Pages Using XM, Documents, KEPT 2007 - Knowledge Engineering: Principles and Technologies, International Conference, Babes-Bolyai University, Cluj-Napoca, June 6-8, 2007, 236-243.
(7) Tambulea L., and H. F. Pop, Cooperative Model For Web Sites Authoring, International Workshop in Collaborative Systems, Cluj-Napoca 2006, Annals of the Tiberiu Popoviciu Seminar, Cluj-Napoca, 28 29 octombrie 2006, 329-336.
(8) Dumitrescu Dan, Sas Laura, Serban Gabriela, Campan Alina, Darabant Sergiu, Pop Horia, Tambulea Leon, Cooperative Learning for Distributed Data Mining, International Workshop "Collaborative Support Systems in Business and Education", 27-29 octombrie 2005, 432 440.
(9) I. Gansca, Willhem F. Bronsvoort, G. Coman, Tambulea L., Self-intersection avoidance and integral properties of generalized cylinders, Computer Aided Geometric Design 19 (2002), pp. 695707.
(10) I.Gansca, Tambulea L., Curves of Bezier type with shape parameters, Numerical analysis and approximation theory, 2002, 187-198.
(11) Gh.Coman, I.A.Rus, Tambulea L., Professor Dimitrie D. Stancu, at his 75th birthday anniversary. Studia Babes-Bolyai, Mathematica, XLVII,4,2002, 3-12 .
(12) I.Gansca, Tambulea L., Parabolic Blanding Surfaces. Automation Computers Applied Mathematics, vol.10, no.1-2, 2001, 56-61.
(13) M.Frentiu, D.Dumitrescu, B.Parrv, H.F.Pop, Tambulea L., Algoritmi utili existenti in reteaua Universitatii. Research Seminars, Seminar of Computer Science, Preprint no.2, 1999, 111-122.
(14) Tambulea L., The Fields of Computer Science, "Babes-Bolyai" University of Cluj-Napoca, Research Seminars, Seminar of Computer Science, Preprint no.2, 1997, 19-22.

(15) Gh.Coman, Tambulea L., I.Gansca, Multivariate Approximation. Applications. Seminar on Numerical and Statistical Calculus, Preprint nr. 1, 1996, 29-60.
(16) Gh.Coman, I.Gansca, Tambulea L., Remodelling Given Polynomial Bezier Curves and Surfaces, Studia Babes-Bolyai, Mathematica, XLIV, 1, 1998, 29-38.
(17) I.Gansca, Gh.Coman, Tambulea L., Rational Bezier Curves and Surfaces with Independent Coordinate Weights, Studia Babes-Bolyai, Mathematica, XLIII, 2, 1998, 29-38.
(18) Gh.Coman, I.Gansca, Tambulea L., Remodelling given Bzier spline curves and surfaces. Studia Babes-Bolyai, Mathematica, XLIII, 1, 1998, 29-38.
(19) I.Gansca, Gh.Coman, Tambulea L., Contributions to Rational Bezier Curves and Surfaces. In Proceedings of the International Conference on Approximation and Optimization, Cluj-Napoca, July 29 - Aug.1, 1996.
(20) I.Gansca, Gh.Coman, Tambulea L., Generalizations of Bezier Curves and Surfaces. In "Curves and Surfaces in Geometric Design", P. J. Laurent, A. le Mehaute, and L. L. Shumaker (eds), A K PETERS, Wellesley MA, 1994, 169-176.
(21) Gh.Coman, I.Gansca, Tambulea L., Surfaces Generated by Blending Interpolation. Studia Babes-Bolyai, Mathematica, XXXVIII, 3, 1993.
(22) Gh.Coman, I.Gansca, Tambulea L., Blending Approximation. In Preceedings of the 9-th Romanian Symposium on Computer Science, 12-13 November, 1993, Iasi (ed.V.Felea, G.Ciobanu), 126-139.
(23) Gh.Coman, I.Gansca, Tambulea L., New Interpolation Procedure in Triangles. Studia Babes-Bolyai, Mathematica, XXXVII, 1, 1992, 37-45.
(24) Gh.Coman, Tambulea L., Bivariate Birkhoff interpolation of scattered data. Studia Babes-Bolyai, Mathematica, XXXVI, 2, 1991, 77-86.
(25) I.Gansca, Gh.Coman, Tambulea L., On a Bezier Surface (III). Bulletins for Applied Mathematics, 765 (1991), 191-198.
(26) Gh.Coman, I.Gansca, Tambulea L., Some new roof-surfaces generated by blending interpolation technique. Studia Babes-Bolyai, Mathematica, XXXVI, 1, 1991, 119-130.
(27) Gh.Coman, Tambulea L., On some interpolation procedures of scatered data. Studia Babes-Bolyai, Mathematica, XXXV, 2, 1990, 90-98.
(28) Tambulea L., Binary trees, an Euler's problem and finite sequences of numbers. Studia Babes-Bolyai, Mathematica, XXXV, 3, 1990, 83-94
(29) I.Gansca, Gh.Coman, Tambulea L., On the Shape of Bezier Surfaces. Studia Babes-Bolyai, Mathematica, XXXV, 3, 1990, 37-42.

(30) Gh.Coman, I.Gansca, Tambulea L., Some practical application of blending interpolation. Itinerant Seminar on Functional Equations, Approximation and Conexity, Cluj-Napoca 1989, Preprint no.6, 1989, 5-22.
(31) Tambulea L., Consecutive retrieval with minimum redundance. Studia Univ.Babes-Bolyai, Mathematica, XXXIII, 3, 1988, 56-60.
(32) Gh.Coman, Tambulea L., and A Shepard-Taylor approximation formula. Studia Univ.Babes-Bolyai, Mathematica, XXXIII, 3, 1988, 65-73.
(33) Z.Kasa, Tambulea L., Binary Trees and Number of States in Buddy Systems. Annales Universitatis Scientirum Budapestinensis de Roland Eotvos Nominate, Sectio Computatorica, Tom.VII, 1987.
(34) Tambulea L., The Storing of Data Collections in Accordance with the Consecutive Retrieval Property. Studia Univ. Babes-Bolyai, Mathematica, XXXII, 3, 1987, 53-66.
(35) D.L. Dumitrascu, P. Groza, Tambulea L., Risk Factors in Ulcer Disease. A Case Control Computer Processed Study. Revue Romaine de Morphologie, d'Embrionologie et de Physiologie, serie Physiologie, 24, 1, 1987, 15-21.
(36) D.L. Dumitrascu, P. Groza, D. Dumitrascu, Tambulea L., Non- Meteorological Factors Associated to the Periodicity of Realapses in Ulcer Disease. Clujul Medical, 1987, vol.LX, 3. 236-240.
(37) Fl.Boian, M.Frentiu, Z.Kasa, Tambulea L., Fortran can be improved. Studia Univ.Babes-Bolyai, Mathematica, XXXII, 3, 1987, 15-16.
(38) Tambulea L., The collection of recordings with the minimum number of blocks. Mathematica, 26 (49), 1984, 81-84.
(39) Tambulea L., Data organization according to the property of consecutive retrieval. Revue d'analyse numerique et de la theorie d'approximation, 9, 2, 1980, 269-281.
(40) Tambulea L., and C.Kalic Sur la resolution de quelques problemes aux limites par la methodes des elements finis. Itinerant Seminar on Functional Equations, Approximation and Convexity, Cluj-Napoca 1983, 169-174.
(41) Tambulea L., and A system for drawing curves and surfaces. Preprint no.5, 1992, 65-69, Research Seminars, Seminar on Computer Science.
(42) E.Radu, Tambulea L., Graphic Modelling of some Combined Effects in High Atmospheric Density Distribution. Preprint no.4, 1991.
(43) Tambulea L., and A new method for storing binary trees. Preprint no.9, 1989, Seminar on Computer Science, Research Seminars.
(44) Gh.Coman, Tambulea L., On the complexity of some scattered date interpolation procedures. Preprint no.10, 1989, Research Seminars, Seminar on Complexity of Algorithms.

(45) Tambulea L., Interactive graphic system. Preprint no.9, 1988, 33-34, Seminar on Computer Science, Research Seminars.
(46) Tambulea L., and Towards a new standard Fortran. Preprint no.2, 1986, Seminar on Computer Science, Cluj-Napoca, 1-20 (in colab.cu: Fl.Boian, M.Frentiu, Z.Kasa)
(47) C.Kalic, Tambulea L., Calcul automatique des formules de cubature. Preprint no.7, 1984, 59-69.
(48) Tambulea L., and Pop, H.F., A formal approach of web sites management, Proceedings of the Symposium Zilele Academice Clujene, Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, June 2006, p. 66-72.
(49) I.Gansca, E.Zetea, Tambulea L., Asupra generarii suprafetelor de tip cupola. In lucrarile celei de a VI-a editie a Conferintei Grafica 2000, 435-440, Craiova 2000.
(50) I.Gansca, E.Zetea, D.Dragan, Tambulea L., Suprafete cu plan director generate de curbe lantisor. In lucrarile celei de a VI-a editie a Conferintei Grafica 2000, 189-194, Craiova 2000.
(51) E.Zetea, I.Gansca, A.Tripa, Tambulea L., Suprafete deformabile cu plan director, generate de cisoide, In lucrarile celui de al IV-lea Simpozion de Geometrie descriptiva si grafica computerizata, Vol.I, 207-212, Editura Bren, Bucuresti 1998.
(52) I.Gansca, E.Zetea, D.Dragan, Tambulea L., Suprafete cu plan director generate de ramuri antisimetrice de parabole. In lucrarile celui de al IV-lea Simpozion de Geometrie descriptiva si grafica computerizata, Vol.I, 145-150, Editura Bren, Bucuresti 1998.
(53) P.Alexa, A.Ioani, Tambulea L., Programarea matematica in calculul placilor plane. In lucrarile simpozionului national "Aplicatii ale informaticii in cercetarea si proiectarea de constructii", Sibiu 1979.
(54) Tambulea L., Model de organizare a colectiilor de date. In culegerea de lucrari "Informatica pentru conducere. Progrese in informatica romaneasca", 1980
(55) V.Ille, A.Ioani, P.Alexa, H.Criveanu, I.Magyarosi, A.Giurgiu, Tambulea L., Procedeu de masurare si inregistrare continua a deplasarilor experimentale. In lucrarile celui de al II-lea simpozion national de tensometrie, Cluj-Napoca 1980, 118-128.
(56) Tambulea L., Determinarea numarului de RC-multimi maxime. In lucrarile celui de al IV-lea Colocviu de Informatica, Iasi, 27-29 octombrie 1983, 129-137.
(57) Fl.Boian, Z.Kasa, D.Oprean, Tambulea L., Automatizarea procesului de luare a deciziilor. Revista economica, supliment 50/1984, 4-6.

(58) Tambulea L., Proprietatea de regasire consecutiva pentru multimi de intrebari maximale si conexe. In lucrarile celui de al V-lea Colocviu de Informatica, Iasi 18-19 octombrie 1985, 270-279.
(59) Fl.Boian, M.Frentiu, Z.Kasa, L.Erdo, A.Szen, Tambulea L., Simularea automatelor programabile. In lucrarile simpozionului "Informatica si aplicatiile sale", Cluj-Napoca 1985, 44-51.
(60) Tambulea L., Model optim de organizare a colectiilor de date. In lucrarile simpozionului "Informatica si aplicatiile sale", Cluj-Napoca 1985. 85-88.
(61) R.Ciupa, S.Ciupa, Tambulea L., Computer utilisation at biological wareform analysis with applications at blood flow in arteries. In lucrarile simpozionului de "Matematici si aplicatii", Timisoara, 12 noiembrie 1985, 170-173.
(62) Fl.Boian, Z.Kasa, Tambulea L., Pachetul de programe Admitere. Produse informatice nr.1, Litografiat Cluj-Napoca 1986, 43 pagini.
(63) Fl.Boian, M.Frentiu, Z.Kasa, Tambulea L., STAF - Sistem de programe pentru elaborarea statelor de functii. Cluj-Napoca 1987.
(64) R.Ciupa, S.Sfrinjeu, Tambulea L., Model matematic de studiu a circulatiei sanguine arteriale. In lucrarile sesiunii de comunicari "Realizari in domeniul electronicii profesionale", Snagov-Parc, 3-5 septembrie 1987, pag.IV.48-51.
(65) Fl.Boian, M.Frentiu, Z.Kasa, Tambulea L., Sistem de programe pentru elaborarea statelor de functii. In lucrarile sesiunii stiintifice a Centrului de calcul al Universitatii din Bucuresti, 20-21 februarie 1987, 438-443.
(66) Tambulea L., Program pentru rezolvarea grafica a problemelor de programare liniara. In Lucrarile seminarului de "Didactica Matematicii", Vol.6 (1990), 343-348.

Books.

(1) Fl.Boian, M.Frentiu, I.Lazar, Tambulea L., Informatica de baza. Presa Universitara Clujeana, Cluj-Napoca, 2005, 225 pagini .
(2) Tambulea L., FoxPro pentru programatori, Editura Promedia Plus Cluj-Napoca. Editia I-a in 1995. Editia a II-a in 1996.
(3) Tambulea L., Access pentru programatori, Editura Promedia Plus Cluj-Napoca 1996.
(4) P. Blaga, Z. Kasa, Tambulea L., Culegere de algoritmi, Litografiat Cluj-Napoca 1977.
(5) Gr. Moldovan, F.Landa, D. Oprean, Tambulea L., Scheme logice si programe Cobol, Litografiat Cluj-Napoca 1979.
(6) Z. Kasa, C.Tartia, Tambulea L., Culegere de probleme de teoria grafelor, Litografiat Cluj-Napoca 1979.

(7) S. Groze, Fl. Boian, M. Frentiu, Tambulea L., Bazele informaticii. Culegere de probleme pentru lucrarile de laborator, Litografiat Cluj-Napoca 1982.
(8) Fl. Boian, Gh. Coman, M.Frentiu, Tambulea L., Elemente de informatica. Curs litografiat Cluj-Napoca 1989.
(9) Tambulea L., Structuri de date si banci de date, curs litografiat in 1992.
(10) Tambulea L., Baze de date, Univ. "Babes-Bolyai" Cluj-Napoca, Editia I: 1998, Editia II: 1999, Editia IIII: 2000, Editia IV: 2001, Editia V: 2002, Editia VI: 2003.
(11) Fl. Boian, Gh.Coman, M. Frentiu, Tambulea L., Informatica pentru elevi. Editura Microinformatica, Cluj-Napoca 1992, Editia a II-a in 1992, Editia a III-a in 1993.
(12) F.Boian, Gh. Coman, S. Groze, M. Frentiu, Z. Kasa, T. Toadere, Tambulea L., Manualul incepatorului in programare Pascal. Editura Albastra, Cluj-Napoca 1995.
(13) P. Blaga, Z. Kasa, N. Lupsa, Tambulea L., Memento FELIX C-256, Litografiat Cluj-Napoca 1995.
(14) M. Frentiu, Z. Kasa, C. Tartia, Tambulea L., Utilizarea calculatorului personal PRAE-M, Litografiat Cluj-Napoca 1986.
(15) Tambulea L., MATH-I. Operating and User's Guide. Scientific Program Library for Romanian Personal Computers. ITCI 1987.

Babes-Bolyai University, Department of Computer Science, Str. Mihail Kogalniceanu Nr 1, RO-400084 Cluj-Napoca

E-mail address: florin@cs.ubbcluj.ro

E-mail address: mfrentiu@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

RELATIONAL DATABASES AND RESOURCE DESCRIPTION FRAMEWORK

LEON ŢÂMBULEA AND ANDREEA SABĂU

Abstract. A large number of Web pages are generated from relational databases. Many papers appeared lately, that mention the advantages offered by Semantic Web in a possible global query of the informational sources over the Internet. It seems that one of the current challenges is to use databases (great amount of data, but with local usage) by specialized programs in the context of future development of the Internet (in Semantic Web). This paper contains a formalization of types of triplets that can appear in Resource Description Framework (RDF) documents. An automatic conversion method of information from a relational database into RDF documents in described based on the presented model.

1. Introduction

The Web is one of the most popular and richest sources of information. According to [17], in November 2009 were received about 240 million answers from world's Web sites. Some Netcraft statistics mention the fact that on every site there are on average 273 Web pages, and on the Internet there are about 65 billion Web pages, which contain a very large amount of information.

The source of data for a Web page is generally a relational database. These Web pages (static pages or dynamic pages) are built at certain moments of time or as an answer to some events that appear in dynamic pages forms. In addition, there are also many Web sites that offer a lot of documentation to be consulted (tutorials, articles, research reports, books), which represents a great quantity of static information.

The Web pages which are generated from different data sources or built by users are written in different languages and are organized in different formats. Most of them are html or xml documents. The documents that can be found

Received by the editors: December 7, 2009.

2010 *Mathematics Subject Classification.* 68P05, 68P20, 68T30.

1998 *CR Categories and Descriptors.* E.1. [**Data**]: Data Structures; H.2.1. [**Information Systems**]: Database Management – *Logical Design* H.3.5. [**Information Systems**]: Information Storage And Retrieval – *Online Information Services*.

Key words and phrases. Semantic Web, RDF, Knowledge.

on the Internet do not have the contained information structured according to a certain pattern; therefore obtaining some data from different document is not an easy task. The search engines are able to perform search operations within the Web using key words, but they are not able to perform reasoning or to interpret the results of the search (the obtained documents) [1, 2, 11]. Today, such reasoning over a page or document can be performed only by a human user. The Web documents should meet certain requirements, have certain structure, or contain also some metadata with a unique format, in order to allow a program (software agent) to perform reasoning.

Adding such metadata in Web pages or creating documents containing a unique pattern for metadata will lead to a new generation of Web (Web 3.0 or Semantic Web). There are more proposed methods for organizing this metadata. The simplest model is RDF (Resource Description Framework), which was already declared as standard by W3C [14].

2. Relational Databases and RDF

Most data that is used in generating Web pages is extracted from databases. The relational databases had a rapid evolution after their first formalization [5]. The storage of data in relational databases benefits of a lot of advantages, like:

- optimization of the access to stored data,
- control of the concurrent access to data,
- implementation of different security procedures.

Different objects are used in management of relational databases, objects which are usually included in programming languages or environments. These objects are using drivers or providers (like ODBC, JDBC, OLEDB, SQL-NCLI, MSDAORA etc.) which make possible the management of data (stored on different management systems) from different programming languages and operating systems.

The metadata of a RDF document is specified as a sequence of instructions, where such an instruction is a triplet (subject, predicate, object). This kind of instruction is similar to a simple sentence from a natural language. For example, having the sentence "The paper appears in Studia Informatica", the subject can be considered to be "the paper", "appears" can be seen as predicate (or property), and "Studia Informatica" is the correspondent for the object of an instruction. The objects from a sentence can be described by an instruction, within which the same item can be subject. For example: "Studia Informatica has the home-page at http://www.cs.ubbcluj.ro/∼studia-i/".

If a RDF instruction is wanted to be used in a wider context (ideally - a global context in Web's space), a described subject (or an object), or a predicate
(property) should be uniquely identified on Web. Therefore, we can say they are *unique resources*. This uniqueness over the Web can be guaranteed by an URI (Uniform Resource Identifier).

So far, we considered very large sources of information, which are relational databases with local usage from dedicated programs, and RDF documents, with included metadata for Web documents and with the possibility of complex queries integration. Having these sources, the raised problem is to convert a relational database (or a part of data from a relational database, a part which is considered to be useful and not confidential to be published) into a RDF document [1, 6, 8, 9, 10, 15]. For example, [12, 13] are offering very large sized RDF documents to be used.

3. Relational Databases

A relational database consists of a number of components that can be defined, modified, or deleted using SQL commands. In order to retrieve data from a relational database, only the relations (tables) and views are useful. Let's consider:

$$B = \{T_i, i := 1, m\} \cup \{V_j, j := 1, n\}$$

the relations and views of a relational database. Each table $T_i, i := 1, m$, has a schema defined by an SQL statement:

$$\text{CREATE TABLE } T_i(column\_definition[, column\_definition]... \\ [, table\_constraints])$$

where $column\_definition$ is given by:

$$column\_name\ column\_type[(length)][column\_constraints]$$

Each database view $V_j, j := 1, n$, can be defined using an SQL statement containing one or more $SELECT$ statements (retrieving data from one or more sources). Also, for a view, the column list (its structure) is known.

The definitions of a relational database's components are stored in a dictionary which can be also consulted by users with proper rights.

The constraints that are defined when the tables are created have the role of keeping the correctness of stored data. These rules are checked every time some data is added, modified, or deleted. If only a data retrieval is performed, there is no reason to check the integrity constraints (only, possibly, the user rights). When some complex queries are executed, especially those who need join operations to be performed over two or more tables, the indexes created on

primary key columns and foreign key columns are usually used for optimizing the retrieval of data.

4. RDF

A RDF document [3, 6, 7, 14] is given by a set of triplets (s, p, v), where s = subject, p = predicate (property), and v = the property's value (a literal value or an object). If the property's value is an object, it should be described in the same manner as a subject; therefore these two notions (subject, object) are identical.

Such a triplet represents information with the following meaning: "The object (subject) s has the property (predicate) p with value v". Therefore, the predicates are similar to binary operators, because they associate a subject s with a value v: $p(s, v)$.

In order to use the objects and their properties in a (as large as possible) context, eventually outside the current document, a unique identification of these is necessary. Such an identification is accomplished using an URI (for example: http://address or ftp://address), in this manner obtaining a *unique resource*. Using such identification items, some relations (links) between resources from different documents may be established. Thus, one property (defined in a RDF document) associates a value (a literal or an object, defined in the same document or in another) to an object (also defined in the same or another document).

When a new resource is defined, an ID attribute may be used, having a value which will identify that resource. Therefore, the value of this ID attribute will be used when the corresponding resource is used (for example: in retrieving the value of one of its properties).

A set of RDF triplets can be stored in different ways:

(1) Using an oriented graph, where the vertices represent objects (subjects) or literals, and the edges represent the connections between objects and values, having the name (identification) of the property as label; the vertices are labeled using the ID of the corresponding resource or the value of the literal.
(2) In an XML document.
(3) Other formats: Notation 3 (N3) [18], Turtle [19], etc.

In order to specify the value of a property which can be decomposed in more pairs (property, value), a *blank node* will be used (as an intermediary resource useful only within the current document).

The value of a property may be:

(a) a value that can be specified by a string (a literal),

(b) a resource (a complex object) having different properties and associated values; the blank nodes are used in this case.

These cases can be graphically represented as a graph (see figure 1). In figure 1.c a blank node is used.

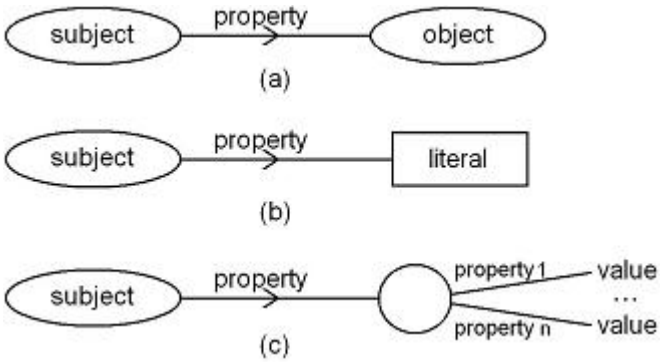


FIGURE 1. The graphical representation of the associations between properties and their values.

In order to formalize these concepts, the following notations will be used:

R = the set of URIs (resource identifiers)
V = the set of literals (the basic values which will not be described by other triplets)
P = the set of properties, $P \subset R$

Let T be the set of triplets within a RDF document, $T \subset R \times P \times (V \cup R)$. An oriented graph corresponding to set T can be built:

$$G_T = (N, E, L),$$

where:

- N = the set of vertices (resources or literals), $N = \{r \mid (r, p, v) \in T\} \cup \{v \mid (r, p, v) \in T\}$. One node $n \in N$ is labeled with the ID of the corresponding resource or with the value of the corresponding literal. In the graph G_T, all nodes corresponding to resources are distinct.
- E = the set of edges, $E = \{(r, v) \mid (r, p, v) \in T\}$.
- L = the set of labels associated to edges of G_T, $L : E \to P$, $L(r, v) = p$, for $(r, p, v) \in T$.

The properties (and their values) are defined by the set of triplets from a RDF document, for a set of resources. This association is free from restrictions. In order to restrict the association of properties to a resource, a pattern should

be defined for that resource. The definition of such a pattern can be done using a *class* (or a *class hierarchy*). A set of *properties* can be associated to a class once defined. Such definitions (for classes, properties, but also for sub-classes, sub-properties) can be built using the triplets (r, p, v) that were mentioned before.

The definition of classes and properties can be done using RDF schemas (noted RDFS, which is an extension of RDF). As mentioned in [6], RDF and RDFS are namespaces with the same meaning, but used with different names because of historical reasons. Properties defined in these two namespaces (rdf:$property$ or $rdfs$:$property$) are used in order to define resources.

A defined resource can be a class, a property, a sub-class, a sub-property, or an instance or a defined class. Such an instance of a class can make use of the properties associated to its class (thus - a model), but also can add some new properties.

Let (r, p, v) be a triplet which defines a resource, where r represents the resource (class, sub-class, property, sub-property). We may have the following definitions:

- $p = rdf$:$type$, $v = rdfs$:$Class$, where the property is indicating that the type of the resource is defined, and the value shows the fact that this resource is a class;
- $p = rdfs$:$subClassOf$, where p is mentioning that the type of the resource is a sub-class (therefore - is inheriting another class), $v =$ reference to a resource that is a class (and which is inherited);

For the triplets $(r, rdf$:$type, rdfs$:$Class)$, $(c1, rdf$:$type, rdfs$:$Class)$, $(r, rdf$:$subClassOf, c1)$ the graph represented in figure 2 can be built.

- $p = rdf$:$type$, $v = rdf$:$Property$, indicating that the defined resource is a property;
- $p = rdfs$:$subPropertyOf$, where p is showing that the resource is a sub-property, $v =$ reference to a resource of type property;
- $p = rdfs$:$domain$, or $p = rdfs$:$range$, $v =$ reference to a resource; in this case the domain or the range of the defined resource is indicated. The domain indicates the subject (the resource) to which it is associated, and the range is the type of the value;

The following relations result from these definitions:

$rdfs$:$Class \in R$
rdf:$type$, $rdfs$:$subClassOf$, rdf:$Property$, $rdfs$:$subPropertyOf \in P$
$rdfs$:$domain$, $rdfs$:$range \in P$

The following two situations specify the type of values for resource r, for the triplet (r, p, v):

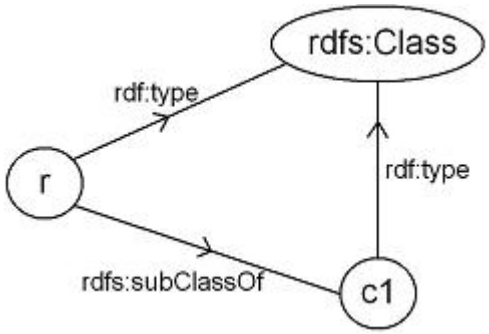


FIGURE 2. The oriented graph corresponding to triplets $(r, rdf{:}type, rdfs{:}Class)$, $(c1, rdf{:}type, rdfs{:}Class)$, $(r, rdf{:}subClassOf, c1)$.

- $p = rdf{:}type$, $v = rdfs{:}Literal$; in this case the value of the resource is a literal (a string);
- $p = rdf{:}type$, $v =$ reference to a resource which is a class or a sub-class; in this case the resource r is an instance of the class indicated by v.

A number of rules of deduction can be defined over a set of triplets, and these rules ensure obtaining new triplets [16]. Some of these rules may be:

(1) Determining the instances of a class:
$$(x, p, y),\ (p, rdfs{:}domain, z) \Rightarrow (x, rdf{:}type, z)$$
$$(x, p, y),\ (p, rdfs{:}range, z) \Rightarrow (y, rdf{:}type, z)$$
$$(x, rdf{:}type, y),\ (y, rdfs{:}subClassOf, z) \Rightarrow (x, rdf{:}type, z);$$
(2) Determining the transitive closure of a property by repeatedly using the following rule:
$$(x, rdfs{:}subPropertyOf, y), \quad (y, rdfs{:}subPropertyOf, z) \quad \Rightarrow$$
$$(x, rdfs{:}subpropertyOf, z);$$
(3) Determining the transitive closure of a class by repeatedly using the following rule:
$$(r, rdfs{:}subClassOf, y), \quad (y, rdfs{:}subClassOf, z) \quad \Rightarrow$$
$$(x, rdfs{:}subClassOf, z).$$

The existing definitions in a set of triplets T or the definitions that can be deduced by rules of deduction must comply with some restrictions, among which:

(a) There cannot be cycles, thus:
$$\forall p, (p, rdfs{:}subPropertyOf, p) \notin T,$$
$$\forall c, (c, rdfs{:}subClassOf, c) \notin T.$$
(b) A property may have one or more domains, but only one range:

$$(r, rdfs{:}range, x) \Rightarrow \nexists y, y \neq x, (p, rdfs{:}range, y).$$

A set of restrictions that violates at least a restriction is called *inconsistent*. Such a set may provide erroneous data to different queries.

5. Conversion of Relational Databases to RDF

There are a lot of papers that contain studies about conversion of relational databases to RDF documents. A working group was established at W3C (Incubator Group) in order to synthesize the papers from this domain, and some of its results were published in 2009 in [15].

Some results from [1, 2, 4, 6, 8, 9, 10, 15] were used for the following remarks:

1. A first proposal for converting relational databases to RDF is obtained in a natural manner. One RDF class is built corresponding to each data source $S[A_1, A_2, ..., A_n]$ (table or view) contained in a relational database. Such a class has the name of the source, and the columns of the source generate the properties of that class. Therefore, for the source S the following triplet will be obtained:

$$(S, rdf{:}type, rdfs{:}Class),$$

and for each column $A_i, i = 1, n$, the RDF document will contain the following triplets (see figure 3):

$$(S.A, rdf{:}type, rdf{:}Property),$$
$$(S.A, rdfs{:}domain, S),$$
$$(S.A, rdfs{:}range, rdfs{:}Literal).$$

Each line (record) from S will generate an instance of the class generated by S through a blank node (the type of this node is the RDF class having the name of the source S). The ID for this node may be automatically generated (as for a column of type auto-increment that exists in many relational database systems).

The structure (schema) of each of the sources from a relational database can be obtained from the database dictionary; therefore such a conversion can be automatically obtained. Some options can be used to such a conversion, like:

(a) Some source columns may change their name when they are transformed into properties. The new names may be taken from vocabularies that already exist and are specialized for certain domains. In this case, some associations between column names and the new RDF property names would be necessary.
(b) It is possible that not all tables and views, or not all data from these sources to be necessary in a conversion from relational database to

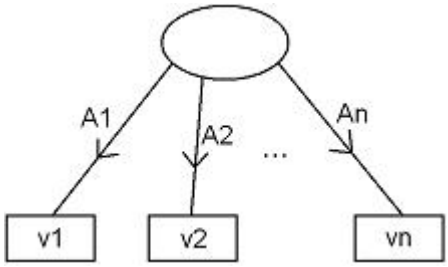


FIGURE 3. The conversion of a data source $S[A_1, A_2, ..., A_n]$ (from a relational database) to RDF document.

RDF. Therefore, in order to perform a conversion, a list of tables, views, and $SELECT$ statements can be used in order to establish the actual sources of data.

(c) We usually define a primary key for a table from a database. This primary key may consist of one or more columns and is used to uniquely identify a record within the corresponding table. In order to use the primary key's values in identifying the blank nodes from the RDF document, some strings may be generated from these values (for example: $S.v$, where S is the table's name and v is the key's value). If the primary key contains more than one column, then the values of all these columns may be concatenated. If the columns use to uniquely identify the rows within a table are not needed in the conversion to RDF, then they may not be extracted in the RDF document.

2. In a relational database, a foreign key constraint can be defined between two tables:

- table $T1$ with a primary key PK,
- table $T2$ with a foreign key FK, which references PK from $T1$.

These restrictions are usually created in the relations normalization process.

A query engine (of a relational database system) is able to optimize the queries that have to compute join between $T1$ and $T2$, using the indexes defined on the primary columns and foreign key columns, respectively. On the other side, the RDF documents that are obtained after a conversion of these tables are not efficient in performing queries on them. If the conversion method previously described is used in conversion, a join operation between the obtained documents is not efficient, because there are no indexes to be used. In order to improve the search, an auxiliary property of class $T2$ may

be defined. This property would have the domain to $T2$ class, the range as $T1$ class, and the name of the foreign key constraint.

3. In order to model $m : n$ (many-to-many) relationships between two tables $T1$ and $T2$ in a relational database, an auxiliary table is created, $T3$. The third table materializes the conceptual $m : n$ relationship, and two $1 : n$ (one-to-many) relationships are obtained in database (between $T1$ and $T3$, and between $T2$ and $T3$).

As an example, the four tables represented in figure 4 are considered to exist in a relational database.

Departments

id	name
1	Math
2	Comp science

Courses

id	title
c1	course1
c2	course2
c3	course3

Students

id	name	dept
s1	stud1	1
s2	stud2	2
s3	stud3	2

CS

idc	ids
c1	s1
c2	s1
c1	s2
c3	s2
c2	s3
c3	s3

FIGURE 4. Example of four tables within a relational database.

The following foreign key constraints are created:

$FK\_CS\_Courses$: $CS(idc)$ references $Courses(id)$
FK_CS_Studs: $CS(ids)$ references $Students(id)$
FK_Studs_Dept: $Students(dept)$ references $Departments(id)$

In figure 5 are represented three of the fourteen blank nodes that are obtained corresponding to records from the four tables, using the method previously described and having one property for a 1:n relationship.

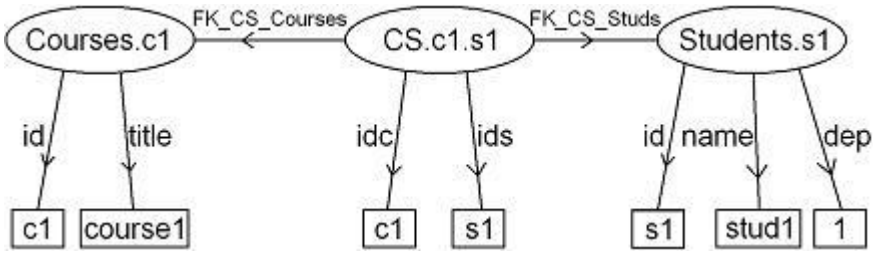


FIGURE 5. Three blank nodes that are obtained after the conversion of tables $Courses$, $Departments$, $Students$, and CS.

The properties *idc* and *ids* can be removed, because the table CS was created only in order to represent the $m : n$ relationship. Their values can be obtained by using the properties $FK\_CS\_Courses$ and FK_CS_Studs. Once the properties *idc* and *ids* are removed, one blank node is obtained, without properties corresponding to columns of table that generated it. This is the reason why this blank node can be also removed; therefore the two existing properties ($FK\_CS\_Courses$ and FK_CS_Studs) change their domain, as it is represented in figure 6.

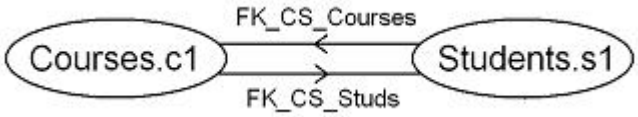


FIGURE 6. The two blank nodes obtained after removal of blank node corresponding to CS table.

6. CONCLUSIONS

The RDF documents are very large data warehouses that can be used by queries in Semantic Web. But, in order to obtain them, the conversion of data from other sources (usually from relational databases) is necessary. This paper analyzed the main characteristics of relational databases and RDF, important for such a conversion, and presented three conversion techniques.

REFERENCES

[1] T. Berners-Lee, *Relational Databases and the Semantic Web (in Design Issues)*, http://www.w3.org/DesignIssues/RDB-RDF.html (1998).
[2] C. Bizer, *D2R MAP-A Database to RDF Mapping Language*, In Proceedings of the 12th International World Wide Web Conference, WWW2003, Budapest, Hungary (2003).
[3] S. C. Buraga, *Considerations Regarding the Use of Semantic Web Technologies in the Context of E-business Applications*, Informatica Economica, 3(35) (2005).
[4] K. Byrne, *Populating the Semantic Web - Combining Text and Relational Databases as RDF Graphs*, PhD Thesis, Edinburgh (2008).
[5] E. F. Codd, *A Relational Model of Data for Large Shared Data Banks*, Communications of the ACM, 13(6) (1970), pp. 377-387.
[6] F. Frasincar, G. J. Houben, R. Vdovjak, P. Barna, *RAL: An Algebra for Querying RDF*, World Wide Web, 7(1) (2004), pp. 83-109.
[7] I. Herman, *Introduction to the Semantic Web*, 2nd European Semantic Technology Conference, Vienna, Austria, (2008).
[8] C. Prez De Laborda, *Incorporating Relational Data into the Semantic Web*, PhD Thesis (2006).

[9] J. F. Sequeda, S. Tirmizi, D. Miranker, *SQL Databases are a Moving Target*, Position Paper for W3C Workshop on RDF Access to Relational Databases, Cambridge, USA (2007).
[10] M. Svihla, I. Jelnek, *Two Layer Mapping from Database to RDF*, In Proceedings of Electronic Computers and Informatics (ECI), Slovakia (2004).
[11] L. Ţâmbulea, A. Sabău, *From Databases to Semantic Web*, Studia Universitatis Babeş-Bolyai, Seria Informatica, Special Issue, LIV (2009), for Intl. Conf. KEPT Knowledge Engineering Principles and Techniques Cluj-Napoca (2009), pp. 85-88.
[12] *D2R Server publishing the DBLP Bibliography Database*, http://dblp.l3s.de/d2r/.
[13] *DBpedia*, http://dbpedia.org/About.
[14] *RDF/XML Syntax Specification (Revised 2004)*, W3C Recommendation, http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/.
[15] *A Survey of Current Approaches for Mapping of Relational Databases to RDF*, W3C RDB2RDF Incubator Group, http://esw.w3.org/topic/Rdb2RdfXG/StateOfTheArt.
[16] *RDF Semantics*, http://www.w3.org/TR/rdf-mt/.
[17] *November 2009 Web Server Survey*, http://news.netcraft.com/archives/web_server_survey.html.
[18] *Notation 3*, http://www.5.org/DesignIssues/Notation3.html.
[19] *Turtle - Terse RDF Triple Language*, http://www.dajobe.org/2004/01/turtle.

Babeş-Bolyai University, Faculty of Mathematics and Computer Science, 1 M. Kogălniceanu St., 400084 Cluj-Napoca, Romania
E-mail address: leon@cs.ubbcluj.ro

E-mail address: deiush@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

TOWARDS AUTOMATED EXECUTION OF SECURITY PROTOCOLS FOR WEB SERVICES

BÉLA GENGE

Abstract. Existing solutions for authentication and authorization in Web services make use of technologies such as SAML or WS-Security. These provide a static solution by using a set of predefined protocols. We propose a dynamic approach for the automated execution problem by developing a semantic security protocol model from which security protocol specifications are generated and automatically executed by participants. The proposed model consists of a sequential component, implemented as a WSDL-S specification, and an ontology component, implemented as an OWL specification. The correctness of the proposed model is ensured by using a set of rules and algorithms for generating it based on a protocol model given by the user. We validate our approach by generating and implementing several specifications for existing protocols such as the ISO9798 or Kerberos protocol.

1. Introduction

Security protocols are widely used today to provide secure communication in insecure environments. By examining the literature we come upon various security protocols designed to provide solutions to specific problems [1]. With this large amount of protocols to chose from, distributed heterogenous systems must be prepared to handle multiple security protocols.

Existing technologies, such as the Security Assertions Markup Language [2] (i.e. SAML) or WS-Security [3] provide a unifying solution for the authentication and authorization issues through the use of predefined protocols. By implementing these protocols, Web services authenticate users and provide authorized access to resources. However, despite the fact that existing solutions provide a way to implement security claims, these approaches are rather

Received by the editors: November 17, 2008.

2010 *Mathematics Subject Classification.* 68M14, 68Q55, 94A62.

1998 *CR Categories and Descriptors.* H.3.5 [**Information Systems**]: Information Storage and Retrieval – *Online Information Services* K.6.5 [**Computing Milieux**]: Management of Computing and Information Systems – *Security and Protection*.

Key words and phrases. Security protocols, Automated protocol execution, Web services.

static. This means that in case of new security protocols, services must be reprogrammed.

The solutions developed over the years such as the one described in [4], propose a formal description for protocol specifications. These specifications do not make use of Web service technologies, because of which inter-operability of systems executing the given specifications becomes a real issue. Another approach for automated security protocol implementation is provided in [5], where the specification is constructed as an XML document from which the code is automatically generated. In order to provide automated execution solutions, specifications must not generate code, but must provide an automated implementation solution without any user intervention.

In this paper we propose a semantic security protocol model (SSPM) for generating security protocol specifications that can be automatically executed by participants. The SSPM has two components: a sequential model and an ontology model. The first component is implemented as a WSDL-S [6] specification while the second component is implemented as an OWL [7] specification. The role of the WSDL-S implementation is to describe the message sequences and directions that must be executed by protocol participants. The role of the OWL implementation is to provide semantic information such as the construction, processing and implementation of cryptographic operations (e.g. encryption algorithm, encryption mode, key).

The proposed SSPM is constructed from a security protocol model (SPM) provided by the user. This model describes message sequences, protocol preconditions and effects. Protocol preconditions are used to identify the knowledge required for running the protocol while protocol effects identify the goal of the protocol (e.g. authentication, key exchange).

The construction of the SSPM from a given SPM must maintain the protocol's security properties. For this we propose several generating rules and algorithms that provide a mapping for each component from SPM to SSPM. The correctness of the proposed rules and algorithms results from the one-to-one mapping of each component and from the correctness of SPM.

In order to validate the proposed solution we have generated and implemented several security protocol specifications. From our results we can clearly state that the SSPM contains sufficient information to enable participants to execute the generated specifications.

2. Protocol Model

Protocol participants communicate by exchanging *terms* constructed from elements belonging to the following basic sets: P, denoting the set of role names; N, denoting the set of random numbers or *nonces* (i.e. "number once

used"); K, denoting the set of cryptographic keys; C, denoting the set of certificates and M, denoting the set of user-defined message components.

In order for the protocol model to capture the message component types found in security protocol implementations [2, 3] we specialize the basic sets with the following subsets:

- $\mathsf{P}_{DN} \subseteq \mathsf{P}$, denoting the set of distinguished names; $\mathsf{P}_{UD} \subseteq \mathsf{P}$, denoting the set of user-domain names; $\mathsf{P}_{IP} \subseteq \mathsf{P}$, denoting the set of user-ip names; $\mathsf{P}_U = \{\mathsf{P} \setminus \{\mathsf{P}_{DN} \cup \mathsf{P}_{UD} \cup \mathsf{P}_{IP}\}\}$, denoting the set of names that do not belong to the previous subsets;
- N_T, denoting the set of timestamps; N_{DH}, denoting the set of random numbers specific to the Diffie-Hellman key exchange; $\mathsf{N}_A = \{\mathsf{N} \setminus \{\mathsf{N}_{DH} \cup \mathsf{N}_T\}\}$, denoting the set of random numbers;
- $\mathsf{K}_S \subseteq \mathsf{K}$, denoting the set of symmetric keys; $\mathsf{K}_{DH} \subseteq \mathsf{K}$, denoting the set of keys generated from a Diffie-Hellman key exchange; $\mathsf{K}_{PUB} \subseteq \mathsf{K}$, denoting the set of public keys; $\mathsf{K}_{PRV} \subseteq \mathsf{K}$, denoting the set of private keys;

To denote the encryption type used to create cryptographic terms, we define the following *function names*:

$$\begin{aligned} FuncName ::= {} & sk && (symmetric\, function) \\ | \; & pk && (asymmetric\, function) \\ | \; & h && (hash\, function) \\ | \; & hmac && (keyed\, hash\, function) \end{aligned}$$

The above-defined basic sets and function names are used in the definition of *terms*, where we also introduce constructors for pairing and encryption:

$$\mathsf{T} ::= . \mid \mathsf{R} \mid \mathsf{N} \mid \mathsf{K} \mid \mathsf{C} \mid \mathsf{M} \mid (\mathsf{T}, \mathsf{T}) \mid \{\mathsf{T}\}_{FuncName(\mathsf{T})},$$

where the '.' symbol is used to denote an empty term.

Having defined the terms exchanged by participants, we can proceed with the definition of a *node* and a *participant chain*. To capture the sending and receiving of terms, the definition of nodes uses *signed terms*. The occurrence of a term with a positive sign denotes transmission, while the occurrence of a term with a negative sign denotes reception.

Definition 1. *A* node *is any transmission or reception of a term denoted as* $\langle \sigma, t \rangle$*, with* $t \in \mathsf{T}$ *and* σ *one of the symbols* $+, -$*. A node is written as* $-t$ *or* $+t$*. We use* $(\pm\mathsf{T})$ *to denote a set of nodes. Let* $n \in (\pm\mathsf{T})$*, then we define the function* $sign(n)$ *to map the sign and the function* $term(n)$ *to map the term corresponding to a given node.*

Definition 2. *A* participant chain *is a sequence of nodes. We use* $(\pm\mathsf{T})^*$ *to denote the set of finite sequences of nodes and* $\langle \pm t_1, \pm t_2, \ldots, \pm t_i \rangle$ *to denote an element of* $(\pm\mathsf{T})^*$.

In order to define a participant model we also need to define the preconditions that must be met such that a participant is able to execute a given protocol. In addition, we also need to define the effects resulting from a participant executing a protocol.

Preconditions and effects are defined using predicates applied on terms: $CON\_TERM : \mathsf{T}$, denoting a term that must be previously generated (preconditions) or it is generated (effects); $CON\_PARTAUTH : \mathsf{T}$, denoting a participant that must be previously authenticated (preconditions) or a participant that is authenticated (effects); $CON\_CONF : \mathsf{T}$, denoting that a given term must be confidential (preconditions) or it is kept confidential (effects); $CON\_INTEG : \mathsf{T}$, denoting that for a given term the integrity property must be provided (preconditions) or that the protocol ensures the integrity property for the given term (effects); $CON\_NONREP : \mathsf{T}$, denoting that for a given term the non-repudiation property must be provided (preconditions) or that the protocol ensures the non-repudiation property for the given term (effects); $CON\_KEYEX : \mathsf{T}$, denoting that a key exchange protocol must be executed before (preconditions) or that this protocol provides a key exchange resulting the given term (effects).

The set of precondition-effect predicates is denoted by $\mathsf{PR\_CC}$ and the set of precondition-effect predicate subsets is denoted by $\mathsf{PR\_CC}^*$. Next, we define predicates for each type of term exchanged by protocol participants. These predicates are based on the basic and specialized sets provided at the beginning of this section. We use the $TYPE\_DN : \mathsf{T}$ predicate to denote distinguished name terms, $TYPE\_UD : \mathsf{T}$ to denote user-domain name terms, $TYPE\_IP : \mathsf{T}$ to denote user-ip name terms, $TYPE\_U : \mathsf{T}$ user name terms, $TYPE\_NT : \mathsf{T}$ to denote timestamp terms, $TYPE\_NDH : \mathsf{T}$ to denote Diffie-Hellman random number terms, $TYPE\_NA : \mathsf{T}$ to denote other random number terms, $TYPE\_NDH : \mathsf{T} \times \mathsf{T} \times \mathsf{T} \times \mathsf{P} \times \mathsf{P}$ to denote Diffie-Hellman symmetric key terms $(term, number_1, number_2, participant_1, participant_2)$, $TYPE\_KSYM : \mathsf{T} \times \mathsf{P} \times \mathsf{P}$ to denote symmetric key terms $(term, participant_1, participant_2)$, $TYPE\_KPUB : \mathsf{T} \times \mathsf{P}$ to denote public key terms $(term, participant)$, $TYPE\_KPRV : \mathsf{T} \times \mathsf{P}$ to denote private key terms $(term, participant)$, $TYPE\_CERT : \mathsf{T} \times \mathsf{P}$ do denote certificate terms $(term, participant)$ and $TYPE\_MSG : \mathsf{T}$ to denote user-defined terms.

The set of type predicates is denoted by $\mathsf{PR\_TYPE}$ and the set of type predicate subsets is denoted by $\mathsf{PR\_TYPE}^*$. Based on the defined sets and predicates we are now ready to define the participant and protocol models.

Definition 3. *A* participant model *is a tuple* $\langle prec, eff, type, gen, part, chain\rangle$, *where* $prec \in \mathsf{PR\_CC}^*$ *is a set of precondition predicates,* $eff \in \mathsf{PR\_CC}^*$ *is a set of effect predicates,* $type \in \mathsf{PR\_TYPE}$ *is a set of type predicates,* $gen \in \mathsf{T}^*$ *is a set of generated terms,* $part \in P$ *is a participant name and* $chain \in (\pm\mathsf{T})^*$ *is a participant chain. We use the* MPART *symbol to denote the set of all participant models.*

Definition 4. *A* security protocol model *is a collection of participant models such that for each positive node* n_1 *there is exactly one negative node* n_2 *with* $term(n_1) = term(n_2)$. *We use the* MPROT *symbol to denote the set of all security protocol models.*

3. Semantic Security Protocol Model

In this section we described the proposed semantic security protocol model (SSPM). The proposed model must maintain the security properties of the protocol and must provide sufficient information for participants to be able to execute the protocol.

Protocols are given using their SPM model described in the previous section. Based on this model we must generate the corresponding SSPM from which the specifications can be constructed. The SSPM has two components: the sequential model (SEQM) and the ontology model (ONTM). The first component is implemented as a WSDL-S specification while the second component is implemented as an OWL specification. In the remaining of this section we provide a description of each component and we provide a set of rules to generate SSPM from a given SPM.

3.1. Sequential and Ontology Models.

We use the symbol URI to denote the set of *Uniform Resource Identifiers*, CONC to denote the set of all concepts and CONC^* to denote the set of subsets with elements from CONC.

Definition 5. *An* annotation *is a pair* $\langle uri, c\rangle$, *where* $uri \in \mathsf{URI}$ *and* $c \in \mathsf{CONC}$. *The set corresponding to a* SSPM *is denoted by* ANNOT *and the set of subsets with elements from* ANNOT *is denoted by* ANNOT^*. *A* message *is a pair* $\langle d, a\rangle$, *where* $d \in \{in, out\}$ *and* $a \in \mathsf{ANNOT}$. *We define* MSG *to denote a set of messages and* MSG^* *to denote the set of subsets with elements from* MSG.

Next, we define the sequential model as a collection of preconditions, effects and messages, based on the previous definitions.

Definition 6. *A* sequential model *is a triplet* $\langle s\_prec, s\_eff, s\_msg\rangle$, *where* $s\_prec \in \mathsf{ANNOT}^*$ *is a set of preconditions,* $s\_eff \in \mathsf{ANNOT}^*$ *is a set of effects and* $s\_msg \in \mathsf{MSG}^*$ *is a set of messages.*

The ontology model follows the description of OWL.

Definition 7. *An* ontology model *is a triplet* $\langle conc, propr, inst \rangle$*, where* $conc \in \mathsf{CONC}$ *is a set of concepts,* $propr \in \mathsf{PROPR}$ *is a set of properties and* $inst \in \mathsf{INST}$ *is a set of instances. An element from propr is a pair* $\langle \alpha, \beta \rangle$*, where* α *is a unique id and* β *is a syntactic construction denoting the property name.*

Let $pr_1 = \langle \alpha_1, \beta_1 \rangle$ *and* $pr_2 = \langle \alpha_2, \beta_2 \rangle$*. Then* $pr_1 = pr_2$ *iff* $\alpha_1 = \alpha_2$ *and* $\beta_1 = \beta_2$*. We define the function* $(\_)_{\mathrm{id}}$ *to map the* α *component and the function* $(\_)_{\mathrm{nm}}$ *to map the* β *component of a given property.*

We use PROPR *to denote the set of all properties and* INST *to denote the set of all instances. We use* PROPR^* *to denote the set of all subsets with elements from* PROPR *and* INST^* *to denote the set of all subsets with elements from* INST*.*

In order to handle the previously defined ontology model we define the function $(\_)_{\mathrm{d}} : \mathsf{PROPR} \rightarrow \mathsf{CONC}$ to map the domain concept of a given property, $(\_)_{\mathrm{c}} : \mathsf{PROPR} \rightarrow \mathsf{CONC}$ to map the category concept of a given property, $(\_,\_)_{\mathrm{ci}} : \mathsf{CONC} \times \mathsf{PROPR} \rightarrow \mathsf{INST}$ to map the instance corresponding to a domain concept and property, $(\_)^{\mathrm{s}}_{\mathrm{e}} : \mathsf{CONC} \rightarrow \mathsf{CONC}^*$ to map the set of concepts for which the given concept is parent, $(\_)_{\mathrm{p}} : \mathsf{CONC} \rightarrow \mathsf{PROPR}^*$ to map the set of properties for which the given concept is domain.

3.2. **Generating the Semantic Security Protocol Model.** In order to generate the SSPM of a given SPM, we start with a core ontology model (OM) (figure 1) that contains concepts found in classical security protocols. The core OM was constructed by consulting security protocols found in open libraries such as SPORE [1] or the library published by John Clark [8].

The core ontology is constructed from 7 sub-ontologies. The sub-ontologies that must be extended with new concepts for each SSPM are denoted in figure 1 by interrupted lines, while the permanent sub-ontologies are denoted by continuous lines.

The *SecurityProperty* sub-ontology contains concepts such as *Authentication*, *Integrity*, *Confidentiality*, *Session_key_exchange*. The *TermType* sub-ontology includes concepts related to term types used in security protocol messages such as *SymmetricKey*, *PublicKey* or *ParticipantName*. Concepts related to cryptographic specifications such as encryption algorithms or encryption modes are found in the sub-ontology *CryptoSpec*. In order to model modules needed to extract keys, names or certificates we use the *LoadingModule* sub-ontology. The *ParticipantRole* sub-ontology defines concepts modeling roles handled by protocol participants such as *Initiator*, *Respondent* and *Third_Party*.

The *Knowledge* sub-ontology contains 5 concepts: *PreviousTerm*, *AccessedModule*, *InitialTerm*, *GeneratedTerm* and *DiscoveredTerm*. Each concept defines a class of terms specific to security protocols: terms from previous executions, modules, initial terms, generated terms and discovered terms.

The last sub-ontology is *CommunicationTerm*, which defines two concepts: *SentTerm* and *ReceivedTerm*. The ontology is extended for each SEM-S with concepts that are sent or received. For each concept, functional properties are defined denoting the operations performed on the terms corresponding to concepts. The concepts used to extend the core ontology are specific to each

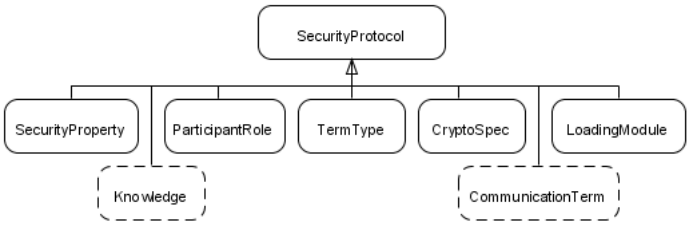


FIGURE 1. Core ontology of SSPM

protocol, however, the defined properties are applied on all constructions. From these properties we mention: *isOfType*, *isEncrypted*, *isStored*, *isVerified*, *isExtracted*, *hasSymmetricAlgorithm*, *hasKey*, *hasLength*.

In order to generate the SSPM from a given SPM we define a set of rules and generating algorithms. The developed rules use the $\_ \leftarrow_{\mathrm{r}} \_$ operator to denote set reunion and the $\_ \leftarrow_{\mathrm{a}} \_$ operator to denote a value transfer.

The first two rules generate the predicate concepts corresponding to preconditions *prec* from a SPM, where the function $gc : \mathsf{T} \rightarrow \mathsf{CONC}$ is used to generate the concept corresponding to a given term and the function *gcc* : $\mathsf{PR\_CC} \rightarrow \mathsf{CONC}$ is used to generate the concept corresponding to a given precondition predicate:

$$\frac{pr \in prec \;\; pr = CON\_TERM(t)}{c \leftarrow_{\mathrm{a}} gc(t) \;\; s\_prec \leftarrow_{\mathrm{r}} \{\langle uri, c\rangle\} \;\; (InitialTerm)^{\mathrm{s}}_{\mathrm{e}} \leftarrow_{\mathrm{r}} \{c\}}\text{pr\_term},$$

$$\frac{pr \in prec \;\; pr \neq CON\_TERM(t)}{s\_prec \leftarrow_{\mathrm{r}} \{\langle uri, gcc(pr)\rangle, \langle uri, gc(t)\rangle\}}\text{pr\_propr}.$$

The rules generating the effects have a similar structure because of the *eff* set. Concatenated terms corresponding to each transmitted or received term are modeled using similar rules. For each sent term the SSPM must provide the construction operations and for each received term the SSPM must provide processing operations.

Processing the received terms is done according to the type of each term and to the knowledge available to the user. The modeled operations introduce constraints on the type and location of knowledge through the following rules.

In the *Knowledge* sub-ontology, each concept has an *isOfType* property attached based on which participants can decide on the operations to execute. For each type, additional properties are defined such as the *hasSymmAlg* or *hasKey* properties for symmetric encrypted terms. The rules based on which these properties are generated are specific to each type.

The remaining generating rules are similar to the presented ones and a complete presentation is out of the scope of this paper. We now provide a brief description of the algorithms that apply the rules we have defined.

The first algorithm generates the preconditions, effects and message sequences of SSPM.

Algorithm 1 Generate preconditions, effects and message sequences

Require: $\langle prec, eff, type, gen, part, chain \rangle \in \mathsf{MPART}$
  for all $pr \in prec$ **do**
    @pr_term(pr), @pr_propr(pr)
  end for
  for all $ef \in eff$ **do**
    @eff_term(pr), @eff_propr(pr)
  end for
  for all $n \in chain$ **do**
    if $sign(n) = +$ **then**
      @msg_tx(n)
    else
      @msg_rx(n)
    end if
  end for

Generating concepts corresponding to the *Knowledge* sub-ontology is done through the use of algorithm 2 and 3. Here, the set of knowledge *KNOW*, corresponding to each executing participant, grows with the construction and reception of each new term. We used the function $mpart : \mathsf{T} \rightarrow \mathsf{T}^*$ to map the set of concatenated terms and the keyword "Exec" to denote the execution of sub-algorithms.

3.3. **Correctness of SSPM.** In the generation process of SSPM from a given SPM, we consider a correct SPM constructed by the user. With the large number of attacks reported in the literature [9], [10], it is vital for new protocol

Algorithm 2 Model positive nodes

Require: $n \in (\pm\mathsf{T}), sign(n) = +$
  for all $t \in mpart(term(n))$ **do**
    Let $c = gc(t)$
    Let $p \Leftarrow$ @con_extr(c)
    if $t \in KNOW$ **then**
      $(p)_{\mathrm{c}} \leftarrow_{\mathrm{a}} c$
    else if $t = \{t'\}_{f(k)}$ **then**
      $(GeneratedTerm)^{\mathrm{s}}_{\mathrm{e}} \leftarrow_{\mathrm{r}} \{c\}$
      Exec $ModelEncryptedGenerated(t)$
    else if $t \in gen$ **then**
      $(GeneratedTerm)^{\mathrm{s}}_{\mathrm{e}} \leftarrow_{\mathrm{r}} \{c\}$
      Exec $ModelPlainGenerated(t)$
    else
      $(DiscoveredTerm)^{\mathrm{s}}_{\mathrm{e}} \leftarrow_{\mathrm{r}} \{c\}$
      Exec $ModelDiscoveredLoaded(t)$
    end if
    $KNOW \leftarrow_{\mathrm{r}} t$
  end for

Algorithm 3 Model negative nodes

Require: $n \in (\pm\mathsf{T}), sign(n) = -$
  for all $t \in mpart(term(n))$ **do**
    if $t \in KNOW$ **then**
      @con_verif
    else if $t = \{t'\}_{f(k)}$ **then**
      if $f = sk \vee (f = pk \wedge TYPE\_KPUB(k, r) \in type, r \in \mathsf{P})$ **then**
        @con_decr
        Exec $ModelEncryptedDiscovered(t)$
      else
        @con_verif
        Exec $ModelEncryptedGenerated(t)$
      end if
    else
      @con_stored
      Exec $ModelDiscovered(t)$
    end if
    $KNOW \leftarrow_{\mathrm{r}} t$
  end for

models to maintain the security properties of protocols for which security properties have been proved to hold.

In order to prove the correctness of the generated SSPM, we consider Γ representing the set of all information included in an SSPM. The information generated by the proposed rules can be divided into three components: mapped information, user-provided information and participant knowledge-based information.

The set of mapped information is denoted by γ_{map} and represents information originating directly from SPM. The set of user-provided information is denoted by γ_{up} and represents information originating from the user (e.g. cryptographic algorithms). The set of knowledge-based information originates from the knowledge available when running the protocol and is denoted by γ_{know}.

By using the above sets $\Gamma = \gamma_{map} \cup \gamma_{up} \cup \gamma_{know}$. The correctness of the information contained in γ_{map} results from the original protocol model, while the correctness of the information contained in γ_{up} results from the assumption that the user provides correct input data.

The information contained in γ_{know} is generated based on the design principles of *fail-stop* [14] protocols. These principles state that the correctness of each received term must be verified and the protocol execution must be stopped immediately in case of invalid terms. By using these principles, the rules we proposed generate verification properties for each received term found in the participant's knowledge set. Protocols that do not follow these rules can not be modeled with our method.

The the correctness of the generated SSPM follows from the correctness of the information generated in the Γ set, constructed from the three sets $\gamma_{map}, \gamma_{up}, \gamma_{know}$ for which the correctness has been discussed above.

4. Experimental Results

In this section we exemplify the construction of a SSPM from a given SPM and we provide a few experimental results from the collection of semantic security protocol models we have implemented.

4.1. **Constructing the SSPM for the "BAN" protocol.**

In order to provide an example for constructing an SSPM for a given SPM, we use the well-known "BAN Concrete Secure Andrew RPC" protocol [1]. This is a two-party protocol providing a session key exchange using symmetric cryptography. The protocol assumes that participants are already in the possession of a long-term key K_{ab}.

In the remaining of this sub-section we provide only the construction of the SSPM for participant A. The SSPM corresponding to participant B can be

similarly constructed because it defines the inverse operations of participant A.

The precondition set $prec_A$ for participant A is $prec_A = \{CON\_TERM(A), CON\_TERM(B), CON\_TERM(K_{ab})\}$ and the effect set eff_A for the same participant is $eff_A = \{CON\_KEYEX(K_{ab})\}$. The set $type_A = \{TYPE\_UD\ (A), TYPE\_UD(B), TYPE\_KSYM(A, B, K_{ab}), TYPE\_KSYM(A, B, K), TYPE\_NA\ (N_a), TYPE\_NA(N_b)\}$ defines the type corresponding to each term and the set $gen_A = \{N_a\}$ defines the terms generated by participant A. The participant name is $part_A = A$ and the participant chain is $chain_A = \langle +(A, N_a), -\{N_a, K, B\}_{sk(K_{ab})}, +\{N_a\}_{sk(K)}, -N_b\rangle$.

By applying the rules and algorithms described in the previous sections we generate the SSPM model. The sequential model is implemented as a WSDL-S specification, while the ontology model is implemented as a OWL specification.

Part of the resulted WSDL-S specification is given in figure 2 and part of the graphical representation of the OWL specification is given in figure 3.

```
...
 <xsd:element name="Msg1Request">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="Term1" type="xsd:base64Binary"
          wssem:modelReference=".../SecProt.owl#SentTerm1">
    </xsd:element>
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
<xsd:element name="Msg2Response">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="EncTerm1" type="xsd:base64Binary"
          wssem:modelReference=".../SecProt.owl#RecvdEncTerm1">
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
...
<wsdl:operation name="Msg1">
 <wsdl:output message="tns:Msg1Request"/>
</wsdl:operation>
<wsdl:operation name="Msg2">
 <wsdl:input message="tns:Msg2Response"/>
</wsdl:operation>
<wssem:effect name="SessionKeyExchange"
      wssem:modelReference=".../SecProt.owl#SessionKey"/>
...
```

FIGURE 2. Sequential model partial implementation

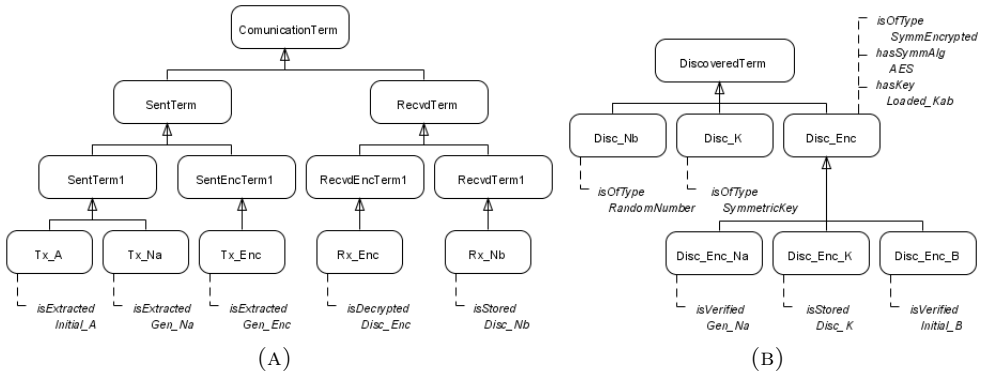


FIGURE 3. Ontology model partial implementation: (a) Communication terms sub-ontology (b) Discovered terms sub-ontology

4.2. **Experimental Results.** In order to prove that the SSPM model contains sufficient information for participants to execute the generated implementations, we generated over 38 WSDL-S and 38 OWL specifications corresponding to initiator and respondent protocol roles.

In order to execute the specifications, messages were encoded and transmitted according to the constructions provided by the WS-Security standard [3]. In the experiments we conducted, participants downloaded the specification files from a public server and they were able to execute the protocols based only on the received descriptions. The participants hardware and software configurations: Intel Dual Core CPU at 1.8GHz, 1GByte of RAM, MS Windows XP.

Part of the experimental results are given in table 1, where the values correspond to milliseconds. The "Spec. proc" column denotes the specification processing time, the "Msg. constr." column denotes the message construction time (for output messages) and the "Msg. proc." column denotes the message processing time (for input messages). The table contains two two-party protocols ("BAN Concrete Andrew Secure RPC", or more simply BAN, and ISO9798) and one three-party protocol (Kerberos). The performance differences between the BAN and ISO9798 protocols are due to the fact that ISO9798 makes use of public key cryptography, while BAN uses only symmetric cryptography.

TABLE 1. Protocol execution timings

Protocol participant	**Spec. proc. (ms)**	**Msg. constr. (ms)**	**Msg. proc. (ms)**	**Total (ms)**
BAN Init.	14.58	11.81	3.68	30.08
BAN Resp.	14.03	2.86	1.62	18.52
ISO9798 Init.	13.07	35.784	23.30	72.16
ISO9798 Resp.	13.51	6.876	12.24	32.63
Kerb. Init. 1	22.63	0.83	0	23.47
Kerb. Init. 2	12.61	0.55	1.58	14.76
Kerb. Init. 3	2.23	3.34	0.94	6.52
Kerb. Resp. 1	19.28	0	0.41	19.69
Kerb. Resp. 2	10.81	3.379	1.67	15.87
Kerb. Resp. 3	5.25	11.41	3.59	20.26

5. Conclusion and Future Work

We developed a novel method for the automated execution of security protocols. Our approach is based on a semantic security protocol model from which security protocol specifications are generated. The sequential component of the proposed model is implemented as a WSDL-S specification while the ontology component is implemented as an OWL specification.

Constructing the SSPM model is not a trivial task and can induce new flaws in correct protocols that can lead to attacks. In order to ensure a correct construction process, we developed several generating rules and algorithms. The proposed rules and algorithms map each component from the input protocol model to a component in the SSPM model. The components from SPM are extended with implementation-specific elements, that do not affect the security properties of the original protocol, as long as correct methods are used to execute the resulted specifications.

In order to prove that the proposed model contains sufficient information for automated execution, we generated and implemented several security protocol specifications. The generated specifications were constructed for well-known security protocols such as the ISO9798 protocol, CCITTX509 or the Kerberos protocol.

As future work we intend to develop a service-based middleware to support secure distribution of these specifications. The middleware will also be able to create new protocols based on already existing protocols and distribute the new specifications to Web services.

References

[1] *Security Protocol Open Repository.* Laboratoire Specification et Verification, http://www.lsv.ens-cachan.fr/spore/, 2008.
[2] *SAML V2.0 OASIS Standard Specification.* Organization for the Advancement of Structured Information Standards, http://saml.xml.org/, 2007.
[3] *OASIS Web Services Security (WSS).* Organization for the Advancement of Structured Information Standards, http://saml.xml.org/, 2006.
[4] L. Mengual, N. Barcia, E. Jimenez, E. Menasalvas, J. Setien, and J. Yaguez. Automatic implementation system of security protocols based on formal description techniques. *Proceedings of the Seventh International Symposium on Computers and Communications*, pages 355–401, 2002.
[5] I. Abdullah and D. Menasc. Protocol specification and automatic implementation using XML and CBSE. *IASTED conference on Communications, Internet and Information Technology*, November 2003.
[6] R. Akkiraju, J. Farrell, J. Miller, M. Nagarajan, M. Schmidt, A. Sheth, and K. Verma. *Web Service Semantics - WSDL-S.* A joint UGA-IBM Technical Note, 2005.
[7] W. W. W. Consortium. *OWL Web Ontology Language Reference.* W3C Recommendation, 2004.
[8] J. Clark, J. Jacob. A Survey of Authentication Protocol Literature: Version 1.0. York University, 17 November 1997.
[9] Gavin Lowe. Some new attacks upon security protocols. In Proceedings of the 9th Computer Security Foundations Workshop, IEEE Computer Society Press, 1996, pp. 162–169.
[10] C. J. F. Cremers. Compositionality of Security Protocols: A Research Agenda. Electr. Notes Theor. Comput. Sci., 142, pp. 99–110, 2006.
[11] C.J.F. Cremers, S. Mauw, E.P. de Vink. Injective Synchronization: an extension of the authentication hierarchy. TCS 6186, Special issue on ARSPA'05, Editors: P. Degano and L. Vigano, 2006, Elsevier.
[12] P. Gutmann. Cryptlib Encryption Toolkit. http://www.cs.auckland.ac.nz/-pgut001/cryptlib/index.html, 2008.
[13] OpenSSL Project. version 0.9.8h, http://www.openssl.org/, 2008.
[14] Gong, L.: Fail-Stop Protocols: An Approach to Designing Secure Protocols. In Proceedings of the 5th IFIP Conference on Dependable Computing and Fault-Tolerant Systems, pp. 44–55 (1995).

"Petru Maior" University of Târgu Mureş, Electrical Engineering Department, Nicolae Iorga str., No. 1, 540088, Târgu Mureş, jud. Mureş, ROMANIA
E-mail address: bgenge@engineering.upm.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

HIERARCHICAL CLUSTERING IN LARGE OBJECT DATASETS - A STUDY ON COMPLEXITY, QUALITY AND SCALABILITY

ADRIAN SERGIU DARABANT AND ANCA GOG

Abstract. Object database fragmentation (horizontal fragmentation) deals with splitting the extension of classes into subsets according to some criteria. The resulting fragments are then used either in distributed database processing or in parallel data processing in order to spread the computation power over multiple nodes or to increase data locality features on each node. In this paper we propose an analysis on the application of hierarchical clustering over object datasets (databases). We use a hierarchical clustering algorithm in order to split the object set into fragments and we analyze their quality based on data accesses in a distributed system. In order to measure the scalability of the algorithm we apply it consecutively to a small, medium and large sized database. We also compare the obtained results with those obtained with other fragmentation algorithms.

1. introduction

Splitting an object database into multiple subsets - usually named fragments - is a process used whenever one needs to distribute computations over a sets of nodes, each containing a subset of the total object database. Fragmentation is used to improve locality features of datasets, processing parallelism or a combination of both. The first approach is used when the entire object set models informations from a real world macro-entity (like an organization) that has a space distributed architecture (an organization with multiple offices located in different geographical areas). The objects are divided into

Received by the editors: November 13, 2009.

2010 *Mathematics Subject Classification.* 68M14, 68P20.

1998 *CR Categories and Descriptors.* C.2.4 [**Computer Systems Organization**]: Computer-Communication Networks – *Distributed Systems*; E.1.2 [**Data**]: Data Structures – *Distributed Data Structures*; H.2.4 [**Information Systems**]: Database Management – *Systems*.

Key words and phrases. clustering, distributed databases.

This work is supported by the Romanian Ministry of Education in the frame of PN2 ID550/2007.

subsets according to the affinity between the real entity modeled by each object and the sets of locations. The main advantage of this approach is that each location (node) already stores the maximum amount of local pertinent information reducing thus the inter-node information exchange requirements. Most applications running in a node will handle local informations in most of the cases.

When parallel processing comes into discussion, one would like that whenever a data intensive request arrives in the (distributed) system it could be divided into multiple sub-requests, each assigned to a different processing node such that the maximum throughput is achieved. In this case data exchange minimization in not the main goal. The main goal is to parallelize the processing by dividing it over as many nodes of the system as possible. A single node is subject to become a processing bottleneck as processing is always divided over the entire system.

When both policies are needed (locality and parallelism) a request is divided in multiple tasks that are assigned to the nodes of the system such that only nodes that can provide data from their local storage to the final result participate in task resolutions.

Experience shows that distributed databases do not evolve at the pace of centralized systems. This is not due to the fact the we lack naturally distributed applications. On the contrary, most of today applications, from ticket reservation to medical patient records management, are inherently distributed. A common sense reason for this lack of development could be the complexity involved in the design and management of distributed architectures. Compared to centralized data stored, most known database distribution techniques require a lot more apriori information about the data that would be stored and applications that will run in the system than in centralized data stores. The existing algorithms for relational database fragmentation [11] are either too complicated to understand and follow or too sensitive to small changes in the inputs, making them difficult to approach by the usual database administrator (DBA).

2. Contributions

Our approach to object dataset fragmentation is based on the idea that a distributed database as described above should be easy enough to design and maintain. In this paper we present a fragmentation method using hierarchical clustering algorithm [1]. The fragmentation quality is evaluated using a partition evaluation function already used in previous work [6, 4, 2, 5, 3]. The main contribution here is to asses the validity of this partitioning method for different sized databases. We also compare our results with those obtained in other

similar works [9, 10, 4]. An application of a variant of the k-means algorithm for clustering in fragmentation is presented in [7]. Most of the other methods are based on variants of the relational database fragmentation algorithms.

We also strive to prove that the requirements of our method are far more easier to accomplish and that the algorithm is almost *automatic* - i.e. it doesn't require a deep analysis of the potential data that will be stored in the database. The algorithm is very simple and its application is almost immediate.

3. Numerical Model

The object data model formalization is based on one of the many equivalent models in literature [8]. Hierarchical clustering splits a set of items (in our case objects) according to their similarities on a common set of features that can be quantified. We could let the set of features be values of the object attributes or method results but this would lead to a classification based only on attributes. Classifying the objects on their static data would certainly lead to a set of fragments, but those fragments would not express at all the dynamics of the system. This is hardly interesting for a distributed database where data is only the static dimension. The dynamic part is represented by the applications that access attributes and send messages to the objects according to the class protocol. Our quantification of the features of each object will not be value/attribute based but application based. The quantification leads to a numerical representation of each object in a vector space. We classify then objects into clusters according to their similarities in behavior in the context of the running applications.

Let $Class = \{C_i | \, C_i$ is a class in the system$\}$ be the set of all classes. The extension of a class C_i, denoted $Inst(C_i)$ is the set of all instances *(objects)* of that class: $Inst(C_i) = \{O_j | \, O_j$ is an instance of $C_i\}$. We denote by $Q = \{q_1, \ldots, q_t\}$ the set of all queries*(applications)* that will be running in the system. It should be noted that only applications running with some frequency are taken in account for quantification as those are the ones that will be most influenced by the resulting clusters. Considering an SQL based system, each of those applications will have filters of *where clauses* that will filter the accessed objects. Let $Pred = \{p_1, \ldots, p_q\}$ be the set of all atomic predicates Q is defined on. Let $Pred(C) = \{p \in Pred | \, p$ imposes a condition to an attribute of class C or to an attribute of its parent$\}$.

Given the predicate $p : C_1.A_1. \ldots A_n \; \theta value$, $p \in Pred(C_n)$, where class C_i is the complex domain of A_{i-1}, $i = 2..n$, and A_n is an attribute of C_n that has a complex type(another object) or a simple type (scalar). $\theta \in \{<, >, \leq, \geq, =, \neq, \in, \supset, \supseteq\}$ is a filter operator, while $value \in domain(A_n)$

For each object $O_i \in Inst(C_j)$ we can derive a vector having $|Pred(C)|$ dimensions, each corresponding to a predicate and having a value of *one* if the object is selected by that predicate or *zero* otherwise. All object vectors for objects of a class C yield a matrix denoted *object-condition matrix OCM(C)*: $OCM(C) = \{a_{ij}, 1 \leq i \leq |Inst(C)|, 1 \leq j \leq |Pred(C)|\}$, where $Inst(C) = \{O_1, \ldots O_m\}$ and $Pred(C) = \{p_1, \ldots, p_n\}$.

Table 1 shows an example of a object-condition matrix. Each line in the matrix is the object-condition vector of the corresponding class instance. The features (is selected or not) of each object are only qualitative. If we want to integrate some quantitative information about how objects are filtered by a predicate we can add in the percent of objects that are filtered in the same manner by a predicate. Right side of Table 1 shows the CVM.

OCM(C)	$\mathbf{p_1}$	$\mathbf{p_2}$	$\mathbf{p_3}$	$\mathbf{p_4}$
O_1	1	0	1	1
O_2	0	1	0	1
O_3	1	1	0	0
O_4	0	0	0	0
O_5	1	1	0	0
O_6	0	1	0	0

CVM(C)	$\mathbf{p_1}$	$\mathbf{p_2}$	$\mathbf{p_3}$	$\mathbf{p_4}$
O_1	0.5	0.33	0.16	0.33
O_2	0.5	0.66	0.84	0.33
O_3	0.5	0.66	0.84	0.66
O_4	0.5	0.33	0.84	0.66
O_5	0.5	0.66	0.84	0.66
O_6	0.5	0.66	0.84	0.66

TABLE 1. Object-condition and characteristic matrix for a class

A new matrix *(characteristic vector matrix)* $CVM(C) = \{w_{ij} | i = \overline{1..m}, j = \overline{1..n}\}$ having same dimensions is obtained and defined as:

$$w_{ij} = \frac{\sum_{l=\overline{1..m}, a_{lj}=a_{ij}} [(a_{lj}|a_{lj} = 1) + (1 - a_{lj}|a_{lj} = 0)]}{m} \tag{1}$$

4. Hierarchical Clustering fragmentation

We obtain a numerical model that expresses the dynamic behavior of the data in the context of user applications. The only missing thing is a measure of similarity/dissimilarity between objects and an algorithm capable to take as input the OCM or VCM matrices and produce the desired clusters by grouping together only similar objects.

The similarity functions we used for our tests are based on some well known metrics (euclidian and manhattan):

$$d_E(we_i, we_j) = \sqrt{\sum_{k=1}^{n} (we_{ik} - we_{jk})^2}\,,\ d_M(we_i, we_j) = \sum_{k=1}^{n} |we_{ik} - we_{jk}| \tag{2}$$

$$(3)\quad sim_E(O_i,O_j)=1-\frac{d_E(we_i,we_j)}{|Inst(C)|},\ sim_M(O_i,O_j)=1-\frac{d_M(we_i,we_j)}{|Inst(C)|}$$

The similarity functions have values between 0 (meaning that objects are totally dissimilar) and 1 (meaning full similarity). The similarity needs to be bounded so that we could asses 0 and full similarity. The hierarchical clustering algorithm we used is presented bellow. The algorithm starts with $m = |Inst(C)|$ clusters, each containing a single object. The main iteration unifies the two most similar clusters until the number of remaining clusters drops bellow the desired number of clusters.

```
Algorithm HierachicalFragPrimar is
Input:  C-class to cluster, Inst(C) - instances of C, similarity
function sim : Inst(C)xInst(C) ⟶ [0,1], m = |Inst(C)|, k-the number
of desired clusters where 1 < k ≤ m and OCM(C)/CVM(C) − w_ij .
Output:  The set of clusters F = {F_1, ..., F_k}
Begin
    For i=1 To |Inst(C)| do F_i = {w_i};
    F = {F_1, ..., F_m};
    While |F|>k do
       // Find (F_u*, F_v*) with the greatest similarity
       (F_u*, F_v*) := argmax(F_u, F_v)[sim(F_u, F_v)];
       Fnew = F_u* ∪ F_v*;
       F = F − {F_u*, F_v*} ∪ {F_new};
    End While;
End.
```

5. Results

The performance evaluation in the case of fragmentation is generally a complex issue to be dealt with. Normally the generated fragments should be allocated to the nodes of a distributed system. Then the applications are run against the system and various parameters like execution times, data transfer amounts, etc are measured. Since the allocation of the clusters to nodes problem is by itself an entire research subject for which there is no yet a linear solution, one can choose to do a simple allocation schema like: allocate each cluster to the node where it is most used. This is the approach we used in our tests. In order to be able to find the nodes where a cluster is most accessed locally we need to apriori know:

- The objects that the application accesses, for each application and class;
- The number of nodes in the system;
- The frequency of running a given application on a given node;

Given a system with $S = \{S_1, \ldots, S_S\}$ nodes, each application runs with a certain frequency on each node of the system,$freq_{S_j}(q_i)$. We computed the general impact application q_i has on the clustering process as being the sum of all frequencies over all the nodes of the system:

$$freq(q_i) = \sum_{s=1}^{S} freq_{S_s}(q_i)$$

By definition we only consider applications that have $freq(q_i) > 0$. A general frequency, $freq(q_i) = 0$ means that the application is not running in the system - so it is not useful for the clustering process.

We denote by $p_i \in q_j$ the fact that p_i is part of the filters *(where clauses)* that define q_i. We would like to capture the impact predicate p_i has on the clustering process according to its frequency of execution as well. This means we need to weight the OCM/CVM matrices to take into account the frequency. A predicate with a high execution frequency should have larger influence on the clustering process than a predicate with a low execution frequency. Having $Q = \{q_1, \ldots, q_t\}$ and $FreqGen = \{freq_1, freq_2, \ldots, freq_t\}, 0 < freq_i \leq freqQMax, freq_i \in Z$. When a predicate p_j is a filter in more than a single application its frequency is the sum of individual execution frequencies of each application that uses p_j.

$$freq(p_j) = \sum_{i=1, p_j \in q_i}^{t} freq(q_i), 0 < freq(p_j) \leq freqMax \in Z$$

In order to use the predicate frequencies weights in the OCM/CVM matrices we need to shift their values in a well know bounded interval, keeping in the same time their semantic. After the interval shift we can directly weight the OCM/CVM matrices:

$$w_{ij}^{`} = w_{ij} \times \frac{freq(p_j)}{\sum_{i=1, p_j \in q_i}^{t} freqMax} \tag{4}$$

For the numerical evaluation of the proposed model we consider small, medium and large datasets. In order to asses the clustering quality we use a set of applications with a predefined set of frequencies randomly chosen.

We use the partition evaluator proposed in other similar works as [4, 2, 6] for cluster quality:

$$PE(C) = EM^2 + ER^2 \tag{5}$$

The evaluator (PE) computes the cost of accessing local data (EM) and remote data (ER) when running the set of user queries over the fragments of a class. As the value of the cost increases, the quality of fragmentation is lower.

$$EM^2(C) = \sum_{i=1}^{M} \sum_{t=1}^{T} freq_{ts}^2 * |Acc_{it}| * \left(1 - \frac{|Acc_{it}|}{|F_i|}\right) \tag{6}$$

$$ER^2(C) = \sum_{t=1}^{T} \min \left\{ \sum_{s=1}^{S} \sum_{i=1}^{M} freq_{ts}^2 * |Acc_{it}| * \frac{|Acc_{it}|}{|F_i|} \right\} \tag{7}$$

The EM term computes the local irrelevant access cost for all fragments of a class. ER calculates the remote relevant access cost for all fragments of a class. Acc_{it} represents the set of objects accessed by query t from fragment F_i. The value $freq_{ts}$ is the frequency of query t running on site s. In (6) s is the site where F_i is located, while in (7) s is any site not containing F_i. M is the number of clusters for class C, T is the number of queries and S is the number of sites. The fragmentation is better when the local irrelevant costs and the remote relevant access costs are smaller. Each term of PE calculates in fact the average square error of these factors.

The quality of fragmentation expressed as the cost of evaluating queries against the resulting database is expressed in Figure 1:

In Figure 1 we compare the application execution costs for a small database clustered with the k-means algorithm (random initial centroids), hierarchical clustering algorithm, single site database, fully replicated database and the fragmented dataset when using the method exposed in [10]-*Bai* and in [9] - *Bel*. According to the PE measure the single node database and full replication both obtain very high, respectively high costs. This is mostly due to ER term scoring very high in the case of single node database - most of the data is accessed remotely in this case. For the fully replicated case the high score is due to the local irrelevant accesses to data (EM).

We apply the hierarchical and k-means clustering to both OCM and CVM matrices. The better results of the hierarchical clustering are due here to the random initial centroids for the k-means method. Normally the k-means algorithm should perform better in these scenarios, but the random choice of the initial centroids often lead to lost clusters and bad results. Overall,

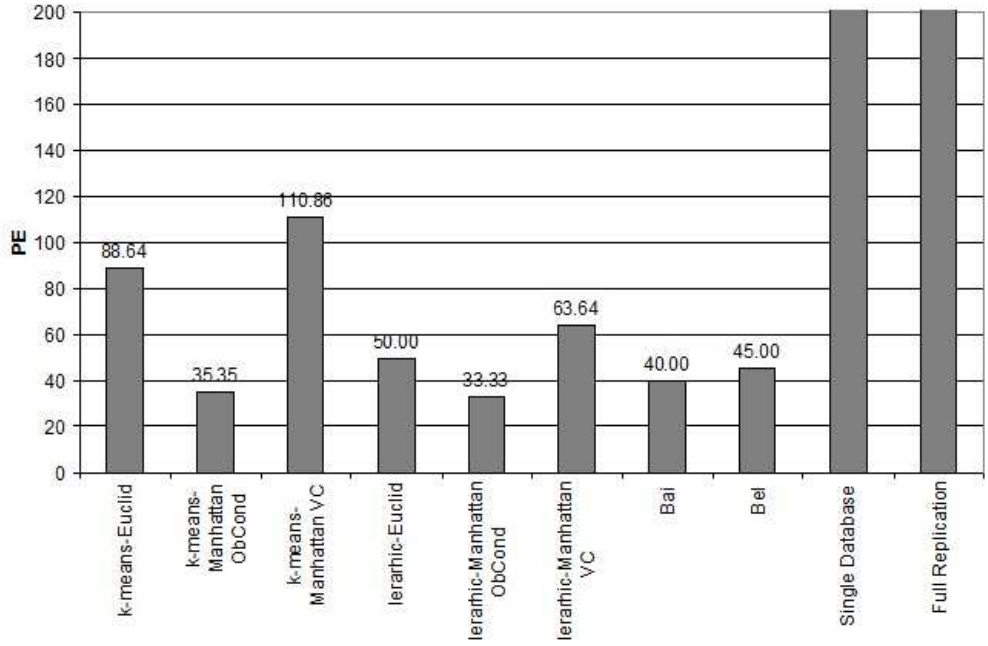


FIGURE 1. Cluster quality evaluation - PE measures.

on small databases the hierarchical clustering performs better than all other methods. The best result is obtained when applying the algorithm on object-condition matrices. The binary selection of predicates yields a better selection than the quantification of *selected/not selected percents* of application objects.

Figure 2 shows the results from a scalability point of view. Three different dataset sizes are tested: small, medium and large datasets. The small dataset context has already been presented in Figure 1. With the significant growth of the dataset, the inefficiency of randomness in centroid choosing fades away and the k-means method takes its place with the smallest cost. Hierarchical clustering scores linearly with the database size, as unifying entire clusters is prone to introducing misplaced objects together with the good ones. At this level hierarchical clustering is too coarse as it does not allow for individual object placing in clusters. Bei and Bel methods are not very affected by the size changes.

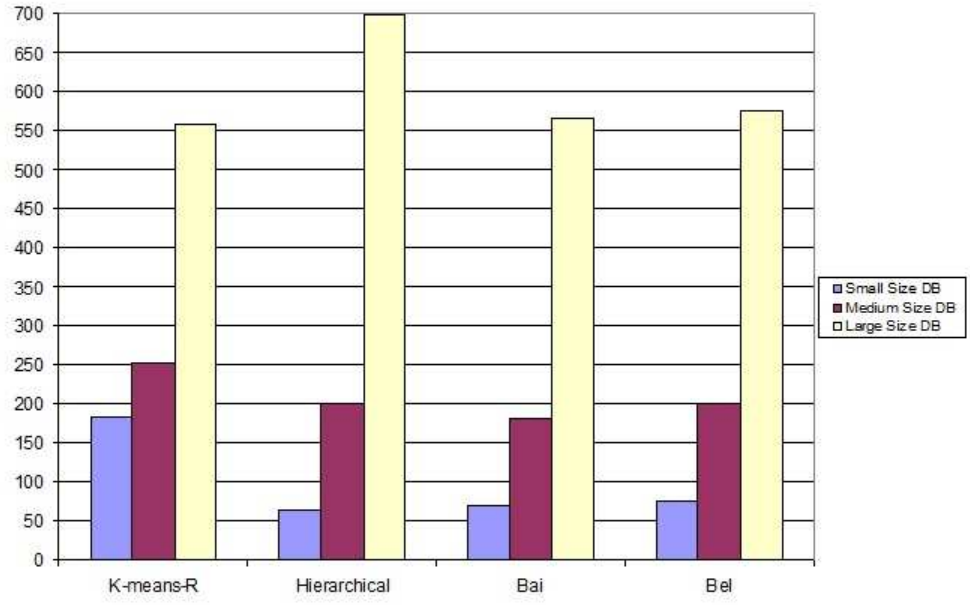


FIGURE 2. Quality evaluation for small, medium and large datasets

6. CONCLUSIONS

Datasets and database fragmentation is a difficult subject to approach from a practical point of view. Existing methods are far too demanding in knowledge and experience for the average DBA to be able to perform them correctly. In this paper we present a dataset fragmentation method based on hierarchical clustering that is easy to apply. Basically, we need as input some quantitative information about the applications that run in the system and the sets of data selected by their filters. Building the OCM/CVM matrices is straightforward and applying the algorithm does not require any additional knowledge from the user. From the performance point of view the algorithm produces clusters comparable in quality with other more elaborate fragmentation methods. We have also shown that our simple method yields good results when evolving the database scale from small to medium.

REFERENCES

[1] Han, J., Kamber, M., Data Mining: Concepts and Techniques, The Morgan Kaufmann Series in Data Management Systems, 2000.

[2] Karlapalem, K., Navathe, S.B., Morsi, M.M.A. - Issues in distribution design of object-oriented databases. In M. Tamer Ozsu, U. Dayal, P. Valduriez, editors, Distributed Object Management, pp 148-164, Morgan Kaufmann Publishers, 1994.
[3] Karlapalem, K., Li, Q., Vieweg, S. - Method Induced Partitioning Schemes in Object-Oriented Databases, In Proceedings of the 16th Int. Conf. on Distributed Computing System (ICDCS'96), pp 377-384, Hong Kong, 1996.
[4] Ezeife, C.I., Barker, K. - A Comprehensive Approach to horizontal Class Fragmentation in a Distributed Object Based System, International Journal of Distributed and Parallel Databases, 33, pp 247-272, 1995.
[5] Karlapalem, K., Li, Q. Partitioning Schemes for Object-Oriented Databases, In Proceedings of the Fifth International Workshop on Research Issues in Data Engineering-Distributed Object Management, pp 42-49, Taiwan, 1995.
[6] Darabant, A. S, Campan, A. - Semi-supervised Learning Techniques: k-means Clustering in OODB Fragmentation, In Proc of the IEEE Intl Conf on Computational Cybernetics ICCC 2004, pag: 333 338, Wien, Austria
[7] Darabant A. S., A new approach in fragmentation of distributed object oriented databases using clustering techniques, in Studia Univ. Babes Bolyai Informatica, Vol I, No 2, pag 91-106, 2005.
[8] Bertino, E., Martino, L. - Object-Oriented Database Systems; Concepts and Architectures, Addison-Wesley, 1993.
[9] Bellatreche,L., Karlapalem, K., Simonet, A. - Horizontal Class Partitioning in Object-Oriented Databases, In Lecture Notes in Computer Science, volume 1308, pp 58-67, Toulouse, France, 1997.
[10] Baiao, F., Mattoso, M. - A Mixed Fragmentation Algorithm for Distributed Object Oriented Databases, In Proc. Of the 9th Int. Conf. on Computing Information, Canada, pp 141-148, 1998.
[11] Tamer, Oszu M., Patrick Valduriez. Principles of Distributed Database Systems, Prentice-Hall, 1998.

Faculty of Mathematics and Computer Science, Babe-s-Bolyai University, Cluj-Napoca, Romania
E-mail address: {dadi,anca}@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

MACRO-ROUTING. PERFORMANCE EVALUATION

SANDA DRAGOŞ AND MARTIN COLLIER

Abstract. QoS routing is used in networks to find feasible paths that simultaneously satisfy multiple (QoS) constraints. The scaling difficulties in conventional shortest-path routing can be addressed using hierarchical routing and state aggregation. State aggregation gives rise to an approximate representation of the network, which can lead to inaccurate path selection. We evaluate in this paper our hierarchical routing protocol, called *Macro-routing*, that we introduced in [4]. *Macro-routing* can distribute the route computation efficiently throughout the network using mobile agents. This allows it to process more detailed information than in conventional hierarchical routing protocols and so increases the likelihood of finding the best path between source and destination.

1. Introduction

The migration of all types of communications services to packet-switched networks (notably the Internet) means that the proportion of traffic generated by real-time applications continues to rise. The Quality of Service (QoS) requirements of such applications can be met either by overprovisioning of the network, or by service differentiation. The latter solution will require ubiquitous network support for QoS routing. QoS routing is the process of identifying efficient paths that can satisfy QoS constraints (e.g. bandwidth, delay, delay variation).

The primary issue for QoS routing solutions in very large networks is *scalability* [10, 9, 2]. Unlike routing protocols such as BGP that are based on relatively static information (e.g. path hop count or AS count), QoS routing requires the frequent updating of dynamic network state information. The update messages consume significant network bandwidth. This amount of state information needs considerable storage space, while processing it requires significant computational power. A large scale deployment of QoS routing

Received by the editors: September 30, 2009.

2010 *Mathematics Subject Classification.* 68M12, 90B20.

1998 *CR Categories and Descriptors.* C.2.1. [**Network Architecture and Design**]: Network topology – *Hierarchical topology*; C.2.2. [**Network Protocols**]: Routing protocols.

Key words and phrases. QoS routing, hierarchical routing.

generates three main types of overhead: *communication*, *computation* and *storage*.

Topology aggregation [8] reduces the amount of routing information and the routing table sizes by orders of magnitude. However, aggressive aggregation methods may have a negative impact on routing performance [7].

2. Hierarchical QoS routing protocols

Private Network-to-Network Protocol (PNNI) [3] is the only QoS-aware hierarchical routing protocol standardized and implemented. It is used in ATM networks and allows up to 104 hierarchical levels. A drawback of PNNI is that the route computation load is distributed unevenly among the network nodes. Also, the aggregation process used in PNNI leads to inaccurate state information advertisements [7, 11] which can result in the inefficient utilization of network resources.

HDP [5] is a proposal for a hierarchical routing protocol within Multiprotocol Label Switching (MPLS) networks. It uses cluster-based server farms as managing nodes which collect all the routing information from their domains and compute them centrally. The setup time of a path is reduced by computing the routes within different domains on one level in parallel, at the expense of an increase in the number of messages [5]. Also, by starting the path computation at the top of the hierarchy and progressing downwards, the aggregation strategies may be used inefficiently as some routing information can already be obsolete by the time the protocol reaches the lower levels of the hierarchy.

In Viewserver [1], the path computation is done by the source node, which gathers centrally all the required routing information by traversing the hierarchy upwards (to find the parent "*view server*") and downwards (to collect detailed routing information about transit and destination domains). However, the setup time is long as the whole path is computed on a single node, and there are scalability concerns regarding the amount of state information gathered.

3. Macro-routing

We proposed in [4] a new protocol that addresses the problem of hierarchical QoS routing within MPLS networks. It is called *Macro-routing* because, being an inter-domain routing protocol, its routing decisions at the higher levels are macro-decisions, as opposed to the detailed or micro-decisions made at the lowest level of the hierarchy.

Macro-routing is capable of both routing and signalling functionalities which are accomplished by the use of mobile agents. Thus, instead of advertising state information, small mobile agents are dispatched to process such information at each node. The advantage of this approach is that the information used to compute routes can be much more detailed that in traditional link-state protocols (e.g., it can feature multiple QoS constraints, or a Full-Mesh aggregated representation). Moreover, by using mobile agents which can replicate at each node and therefore analyse a large set of paths, route computations are done in a distributed and parallel manner which reduces the time required for path setup and distributes the processing burden amongst mobile agents.

3.1. **Protocol description.** The hierarchical organization of *Macro-routing* consists at the lowest level of a number of domains which are typically independent administrative areas. The nodes within such domains are physical network nodes (i.e. routers or switches). Each domain has a *managing node*, which must be able to interpret mobile agent code. It can either be selected from the nodes of the domain (as with PNNI) or it can be a distinct node (as in HDP). Its main function is to maintain an aggregated representation of the domain it is managing.

As the hierarchy is decided administratively, each domain at the lowest level of the hierarchy may choose its own routing strategy. For example it may use standard link-state methods, or may use mobile agents for route discovery. The latter method implies the existence of a mobile agent interpreter on each router or switch. The only *Macro-routing* requirement is that the managing node must contain an Full-Mesh aggregate representation of the managed domain. The maintenance of that aggregate representation is the responsibility of the domain administrator.

For the higher levels of the hierarchy the managing node creates an aggregated representation in four steps:

(1) Each border node and the source and destination nodes activates a mobile agent that floods the domain by replicating itself at each node. Its goal is to find all possible paths to all the other border nodes within the same domain. Each mobile agent records the path it follows and processes the routing information at each node. If one mobile agent is revisiting a node, or the path it has traversed to date does not satisfy the given QoS constraints, it will be discarded. If it reaches another border node it will transmit the path used and its cost to the managing node.
(2) The managing node chooses optimal paths between each pair of border nodes.

(3) A Full-Mesh aggregation topology is created using the selected paths. The costs of the selected paths will become *nodal costs* when computing paths at the next level of the hierarchy.
(4) Some or all of the other computed paths, which have not been selected for the aggregate representation, can be cached for recovery purposes or as alternative paths.

There are three major phases in the *Macro-routing* protocol whereby it finds and selects a QoS path from a given source to a given destination.

3.1.1. *Determination of participant domains.* The first phase involves determining the domains through which the path is likely to pass. It develops in two stages.

In the first stage, the source node initiates an "*upwards search*" in the hierarchy for the lowest level parent node, known as the root node, which has a view of both source and destination, as in HDP and Viewserver.

In the second stage, the parent node initiates a "*downwards search*" in parallel to all its children. Recursively, the nodes reached will continue the search to all their children until they reach the lowest level of the hierarchy. All the physical domains reached by this search will be *participant domains*.

3.1.2. *Path computation.* The second phase involves the determination of the path.

Every managing node in the *participant domains* will create its own aggregate representation by calculating routes between all domain border nodes.

Mobile agents released from each border node traverse the domain multiplying themselves in search of all possible paths that satisfy the QoS constraints. At the end of its journey (i.e. upon reaching a border node different from the starting one) each mobile agent would send the gathered information to the local managing node. Sending the information to the managing node can be done either by piggybacking on legacy messages or using mobile agents.

Starting from the second level of the hierarchy, *nodal costs* will be considered as well as link costs when computing the path cost[1]. The topmost domain will have as border nodes only the aggregated representation of the source and destination domains. The managing node of this domain will determine all the possible paths between its border nodes (the source and the destination) and

[1]Costs include link costs and nodal costs. Nodal costs represent the cost of traversing a virtual node. This nodal cost is zero at the lowest level of the hierarchy.

based on their costs it can determine the optimal path. The other paths found during this process can be used by fast recovery mechanisms or as alternative paths.

3.1.3. *Path reservation and set up.* To accommodate the traffic for which the request has been made, the final path must be set up and the resources reserved. The overall path can be determined by traversing the hierarchy downwards and interrogating all the managing nodes along the chosen path about the detailed sub-paths across their domains.

The path set up and resource reservation can be done either by existing signaling protocols if they can set explicit paths, by existing resource reservation protocols (such as the Resource reSerVation Protocol (RSVP)), or by using suitably programmed mobile agents. Advantages of using mobile agents instead of RSVP for the reservation process are:

- *availability:* Mobile agent support is already available in the nodes, so there is no need to deploy or configure additional software;
- *parallelism:* RSVP has to traverse the path in a sequential manner twice: by using $PATH$ messages, from source to destination, to determine the path and then by using $RESV$ messages, from destination to source, to reserve the path. Mobile agents can do the reservation in a parallel and distributed manner so that, from each hierarchical level, within each domain, a mobile agent can be dispatched to reserve resources corresponding to each logical/aggregated link.
- *hierarchical reservation:* We give the name "hierarchical reservation" to the process of reserving the resources for the overall path in a manner which corresponds with the hierarchical representation used by the routing protocol (i.e. Macro-routing). The first resources to be reserved are those corresponding to the links within the topmost (e.g., level k) domain within the hierarchy. If such resources are not available another level-k path will be selected until there is a path with available resources. Then, the process continues within the $k-1$ level domains which are represented by the virtual nodes traversed by the selected path at level k. The process stops when $k = 0$. The advantage for such reservation strategy is that any unavailable sub-path can be substituted on the spot while all other resources (previously addressed) remain reserved. RSVP, however, performs sequential reservation. This means that any failure is reserving a resource will result in an overall failure, which requires finding a new source-to-destination path and starting the reservation process all over again.

Macro-routing works best in the MPLS context as the separation between the control and forwarding planes allows the coexistence of complex or different routing strategies. Setting up a hierarchical path in an MPLS network is straight-forward. That is because MPLS has the label stack capability. Thus, every sub-path within every domain and every hierarchical level can be treated independently. This can be done either by using a mobile agent-based "hierarchical reservation" or by using any other label distribution protocol.

When all the resources have been successfully reserved and the overall path has been set up, the request is served and the traffic may flow. In the case of resource unavailability or link/node failure, alternative paths which are already computed can be used for a fast recovery.

3.2. **Implementation details.** The *Macro-routing* protocol can be implemented using any mobile agent technology. We have chosen the Wave technology [12]. The main reason for selecting this particular technology is that its syntax make Wave code very compact, perhaps 20 to 50 times shorter that equivalent programs written in C++ or Java [13]. The use of Wave in MPLS networks for routing purposes has already been advocated in [6] to discover multi-point to point trees. Here it is used for hierarchical routing.

The two phases called (*Determination of participant domains* and *Path reservation and set-up*) have a straightforward implementation. The implementation details of the initial *Path computation* phase of the protocol were presented in [4].

4. Test results and performance evaluation

4.1. **The simulation model.** For all tests the *Georgia Tech Internetwork Topology Models* (GT-ITM) [14] was used to generate random network topologies. One example of such topologies is presented in Fig. 1. The corresponding values of number of nodes, number of links and connectivity degree for each topology used in our simulations are shown in Table 1.

One or multiple constraints were associated to each link. The main metrics used were *administrative cost* $\in [1, 15]$ and *hop count*[2] $\in [20, 30]$. For each link, the corresponding metric was randomly chosen.

The results in [4] were obtained by deploying real wave agents on a virtual (i.e., simulated) network running on a single host. Due to the complexity of the virtual networks required to evaluate multi-constrained Macro-routing (hierarchical networks with hundreds of nodes), we developed an application

[2]For the first hierarchical level, the hop count metric would be 1 for each link. However, for the upper hierarchical levels, this hop count can have greater values and differ from one link to another due to the aggregation process.

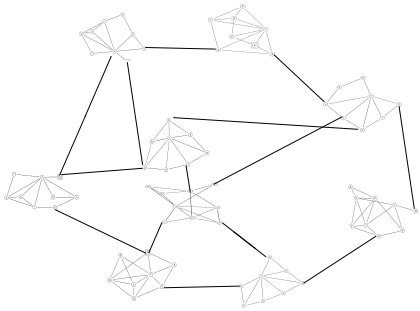


FIGURE 1. Two-level hierarchical topology similar with the ones used in simulation

TABLE 1. Details about two-level hierarchical topologies used in simulation

No. of nodes (N)	**No. of links (L)**	**Connectivity degree ($2L/N$)**
$4 \times 4 = 16$	24	3.00
$9 \times 9 = 81$	124	3.06
$10 \times 10 = 100$	149	3.98
$12 \times 12 = 144$	[216, 372]	[3, 5.166]
$13 \times 13 = 169$	282	3.33
$15 \times 15 = 225$	368	3.27
$20 \times 20 = 400$	600	3.00

to generate a list of the paths the waves would discover, rather than deploying waves on such networks.

4.2. **Macro-routing's communication overhead.** The communication overhead incurred by Macro-routing was evaluated in [4]. It was shown that many of the waves Macro-routing generates within high connectivity topologies end in cycles, and thus do not contribute to route discovery. We introduced a parameter called *lifespan*, which resembles the TTL field used in the IP protocol. Its purpose is to limit the number of *waves* generated during route search by reducing the number of generations which the parent *wave* can produce. The rationale for this is that the law of diminishing returns is assumed to apply - it is unlikely that an exhaustive search of every possible path is necessary to find the optimal path. The modified algorithm is no longer *guaranteed* to find the optimal path (and indeed will find *no* path if the destination is more

than *lifespan* hops away). It was shown, however, that in a representative[3] network comprising $9 \times 9 = 81$ nodes, the optimal paths were found provided that $lifespan \geq 5$.

The following tests further explore the influence of the *lifespan* parameter on the performance of Macro-routing. Before presenting our test results, we first introduce some notation and terminology.

Definition Effort

Macro-routing's *effort* is the ratio of the number of ineffective waves which end up in cycles after n nodes have been visited to the number of waves which might find a path.

Definition Efficiency

Let C_{opt} be the optimal path cost between a specific source and destination, and C_{act} the actual path cost obtained by the (sub-optimal) lifespan-limited Macro-routing algorithm. Macro-routing's *efficiency* is then:

$$E = \frac{C_{opt}}{C_{act}}, \tag{1}$$

Definition Failure, success, best

Let E be Macro-routing's efficiency as defined in Definition 4.2. We define Macro-routing's results as:

- *failure*: no path is found (because of too low a lifespan value): $E = 0$
- *success*: paths satisfying the requirements are found, and: $E > 0$
- *best*: the paths found include the optimal path, i.e. : $E = 1$.

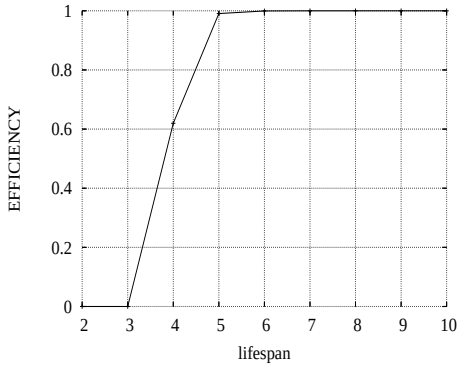
The tests were performed on two-level hierarchical networks with connectivity varying from 3 to 5.166 and $12 \times 12 = 144$ nodes. These topologies were divided into three different classes based on their connectivity degree[4] (c_d) varying in the following intervals:

(1) $c_d \in [3, 3.66]$
(2) $c_d \in [4, 4.5]$
(3) $c_d \in [5, 5.166]$

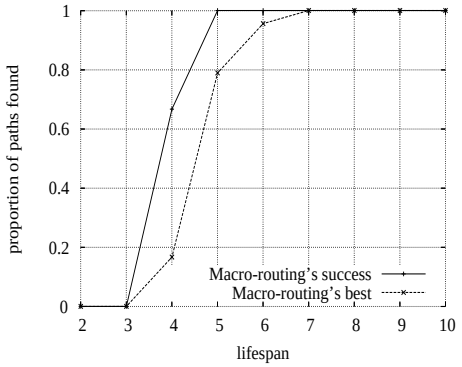
[3]Each domain from the two level hierarchical networks has an average node degree of 3.5 (i.e. $2 \cdot L/N = 3.5$).

[4]We will refer to the average node degree (i.e. $2 \cdot L/N$, where L is the number of links and N is the number of nodes) as *connectivity degree*.

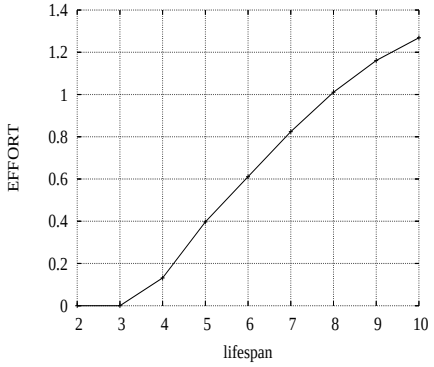
We ran 30 sets of tests on each hierarchical topology class. The mean values of the results obtained across the three classes are depicted in Fig. 2. They show that on the given topology (with $12 \times 12 = 144$ nodes) a lifespan value above 5 does not affect Macro-routing's efficiency in a significant manner (see Fig. 2(a) and 2(b)), while the number of waves is greatly reduced (see Fig. 2(c) for the ratio between the cycle and the alive waves and Fig. 2(d) for the total number of waves/link). Moreover, when the value of lifespan is set to 7 there is no loss in Macro-routing's efficiency, while its overhead (i.e. the number of waves) is considerably reduced.



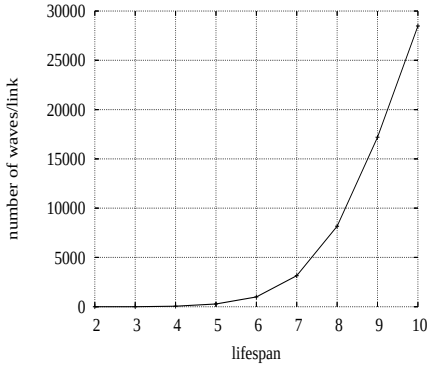
(a) Macro-routing's Mean **Efficiency** (see Definition 4.2)



(b) Mean percentages of **success** and **best** Macro-routing paths found (see Definition 4.2)



(c) Macro-routing's Mean **Effort** (see Definition 4.2)



(d) Mean number of waves/link generated

FIGURE 2. Macro-routing's performance while applying the *lifespan*

The average communication overhead generated by waves, as depicted in Fig. 2(d) is significant for lifespan values above 5, considering the relatively small size of the network. This is because of the high network connectivity in these cases, as can be seen in Fig. 3, which shows the considerable variation in the number of waves generated on three different classes of topologies.

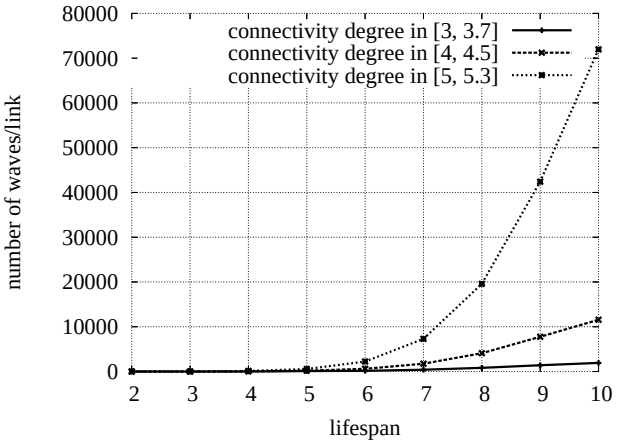


FIGURE 3. Macro-routing's communications overhead on different topologies

The number of waves generated by Macro-routing is overestimated in these results due to the assumption that all paths satisfy the constraints. Demanding constraints would significantly limit the number of compliant paths, and thus the wave population.

5. CONCLUSIONS

In this paper we proposed solutions for *Macro-routing*'s *scalability* in terms of storage, computational and communication overhead introduced by QoS routing. By using mobile agents, *Macro-routing* allows routes to be discovered rapidly without the imprecision introduced by topological state aggregation in other approaches. The price paid for this level of performance is that a large number of mobile agents (implemented as *waves*) traverse the network. However, the level of wave traffic can be restricted by limiting their *lifespan*. Tests run on the two level hierarchical networks showed that there exists a threshold over which this parameter significantly reduces the communication overhead without impairing the performance of our protocol. Moreover, the size of one *wave* packet is significantly smaller (i.e., less than one KB) compared with the link-state packets used to distribute QoS routing information

(see details about the PNNI packet size in [3]). Also, due to the dynamic nature of the routing information, such link-state packets need to be broadcast frequently. In contrast, waves are only dispatched when needed. Thus, it can be tentatively assumed that the communication overhead generated by the wave population is, at worse, no greater than the overhead generated by the link-state packets when centralized QoS mechanisms are used. This quantitative argument will need to be confirmed by quantitative studies, which is a topic for future research.

Acknowledgements

This material is partially supported by the Romanian National University Research Council under award PN-II IDEI 2412/2009.

References

[1] C. Alaettinoglu and A. U. Shankar, *The Viewserver Hierarchy for Interdomain Routing: Protocols and Evaluation*, IEEE Journal of Selected Areas in Communications, 13 (1995), pp. 1396–1410.

[2] G. Apostolopoulos, R. Guerin, S. Kamat, and S. K. Tripathi, *Quality of service based routing: A performance perspective*, in ACM SIGCOMM 98, Vancouver, Canada, August 1998, pp. 17–28.

[3] ATM Forum, *Private network-network interface specification , version 1.0*, Tech. Report af-pnni-0055.000, ATM Forum, March 1996.

[4] S. Dragos and M. Collier, *Macro-routing: a new hierarchical routing protocol*, in Proceedings of the IEEE Global Telecommunications Conference (Globecom), November/December 2004.

[5] M. El-Darieby, D. C. Petriu, and J. Rolia, *A Hierarchical Distributed Protocol for MPLS path creation*, in 7th IEEE International Symposium on Computers and Communications (ISCC'02), July 2002, pp. 920–926.

[6] S. Gonzalez-Valenzuela and V. C. M. Leung, *QoS Routing for MPLS Networks Employing Mobile Agents*, IEEE Network Magazine, 16 (2002), pp. 16–21.

[7] R. Guerin and A. Orda, *QoS-based Routing in Networks with Inaccurate Information: Theory and Algorithms*, in IEEE Conference on Computer Communications (INFOCOM'97), April 1997, pp. 75–83.

[8] F. Hao and E. W. Zegura, *Scalability techniques in qos routing*, Tech. Report GIT-CC-99-04, College of Computing, Georgia Institute of Technology, 1999.

[9] ———, *On Scalable QoS Routing: Performance Evaluation of Topology Aggregation*, in 21st IEEE Conference on Computer Communications (INFOCOM'00), vol. 1, June 2000, pp. 147–156.

[10] S. S. Lee, S. Das, G. Pau, and M. Gerla, *A Hierarchical Multipath Approach to QoS Routing:Performance and Cost Evaluation*, in IEEE International Conference on Communications (ICC'03), Anchorage, AK, USA, May 2003.

[11] W. C. Lee, *Topology Aggregation for Hierarchical Routing in ATM Networks*, ACM SIGCOMM Computer Communication Review, 25 (1995), pp. 82–92.

[12] P. S. Sapaty, *Mobile Processing in distributed and Open Environments*, Wiley, 2000.

[13] S. T. Vuong and I. Ivanov, *Mobile Intelligent Agent Systems: WAVE vs. JAVA*, in 1st Annual Conference of Emerging Technologies and Applications in Communications (etaCOM'96), Portland, Oregon, May 1996.

[14] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee, *How to Model an Internetwork*, in IEEE Conference on Computer Communications (INFOCOM'96), vol. 2, San Francisco, California, USA, March 1996, pp. 594–602.

Babeş-Bolyai University, Department of Computer Science, 400084 Cluj-Napoca, Romania

E-mail address: sanda@cs.ubbcluj.ro

Dublin City University, School of Electronic Engineering, Dublin 9, IRELAND

E-mail address: collierm@eeng.dcu.ie

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

CONCEPTUAL KNOWLEDGE PROCESSING FOR DATABASES. AN OVERVIEW

CHRISTIAN SĂCĂREA AND VIORICA VARGA

Abstract. We present an overview of methods of Conceptual Knowledge Processing and their applications for databases, pointing out recent developments and joint research work in this field. Based on the mathematical theory of Formal Concept Analysis, we show how Conceptual Graphs can be used as a representation tool for both database structure and queries. Also we apply methods of Formal Concept Analysis to mine functional dependencies in relational databases. These methods are then discussed on several examples, presenting two software products developed so far, FCAFuncDepMine and CGDBInterface.

1. Conceptual Knowledge Processing

1.1. **Methods of Conceptual Knowledge Processing.** Formal Concept Analysis started in the early '80s as an attempt to restructure the classical lattice theory. The mathematical part of this theory quickly developed but the power of its expressiveness became clear by dealing with concrete problems of data analysis. Since then, Formal Concept Analysis has been extended to a powerful general framework for Conceptual Knowledge Processing and Representation.

According to [21], Conceptual Knowledge Processing is considered to be an applied discipline dealing with ambitious knowledge which is constituted by conscious reflection, discursive argumentation and human communication on the basis of cultural background, social conventions and personal experiences. Its main aim is to develop and maintain methods and instruments for processing information and knowledge which support rational thought, judgment and action of human beings and therewith promote the critical discourse. The word Conceptual in the name Conceptual Knowledge Processing underlines

Received by the editors: October 30, 2009.

2010 *Mathematics Subject Classification.* 68P15, 03G10.

1998 *CR Categories and Descriptors.* H.2.1 [**Database Management**]: Logical design – *Normal forms*; F.4.m [**Mathematical Logic and Formal Languages**]: Miscellaneous.

Key words and phrases. Conceptual Knowledge Processing, Formal Concept Analysis, Database Theory, Conceptual Graphs.

the constitutive role of the thinking, arguing and communication of human being in order to collect process knowledge. Conceptual Knowledge Processing is grounded on the mathematization of traditional philosophical logic, as a doctrine of concept, judgment and conclusion. The mathematical basis is build on Formal Concept Analysis, a mathematical theory of concepts and concept hierarchies, developed in the last 25 years, which has been proved useful in a large number of applications. Conceptual Knowledge Representation, Conceptual Classification, Analysis of Concept Hierarchies, Conceptual Identification, Conceptual Knowledge Inference, Acquisition and Retrieval are just some methods which can prove their usefulness.

Conceptual Knowledge Processing differs from other data analysis methods in that the emphasis is on recognizing structural similarities ([3]), turning a collection of data into a set of knowledge units called *concepts* and unfolding the subsequent encoded knowledge into a *conceptual hierarchy*.

Throughout this paper, *information* in the scope of Conceptual Knowledge Processing should be understood in the same way as in Devlin's book *Infosense - Turning Information into Knowledge* ([6]). Here, he briefly summarize this understanding in the formulas:

Data	=	signs + syntax
Information	=	Data + Semantics
Knowledge	=	Internalized Information + Possibility to use this information in order to acquire new knowledge.

By this understanding, it becomes clear that Formal Concept Analysis is able to support the representation of information and knowledge ([20]). Moreover, it is important to point out the difference between information and knowledge, difference which constitutes the very essence of knowledge processing vs. data analysis. By this understanding, information is always *contextual*, while knowledge is always *conceptual*.

The mathematical theory of Formal Concept Analysis is based on a set theoretical semantics. Formal Concept Analysis starts with a formalization of contexts. Thus, a formal context is a triple $\mathbb{K} := (G, M, I)$. The set G is called the set of objects, M is the set of attributes, and I is a binary relation between G and M indicating which object has which attribute, called incidence relation. A formal context should be understood as a formalization of a part of reality, containing the entire intended information about collected data.

Since a formal context is just a set of data and related information, we need to unfold a so-called knowledge map, by highlighting the basic knowledge units which are formal concepts. A formal concept is a pair (A, B), consisting of two subsets, A the extension, a subset of G, and B the intension, a subset of

M. The formal extension A contains all objects having the formal attributes collected in the set B. The formal intension contains all attributes valid for all formal objects collected in the set A. Concepts can be ordered in a hierarchy by the *subconcept - superconcept* ordering relation. A concept (A, B) is called a subconcept of (C, D), and (C, D) a superconcept of (A, B), if A is a subset of C or, equivalent, D is a subset of B. We write $(A, B) \leq (C, D)$. A subconcept can be understood as a specialization of the superconcept, while the superconcept is the generalization of its subconcepts. The set of all formal concepts of a given formal context $\mathbb{K}$ is denoted by $\mathcal{B}(\mathbb{K})$. It is called *conceptual hierarchy* or *concept lattice*. Indeed, $\mathcal{B}(\mathbb{K})$ is a complete lattice ordered by the subconcept-superconcept relation. Conceptual hierarchies can be understood as a knowledge map for the information encoded in the formal context. Representing the conceptual hierarchy as a treelike structure enables navigation and activates background knowledge. It makes possible to unfold the conceptual structure of the entire data set in a very precise and coherent way. Conceptual hierarchies are valuable visualization methods for complex knowledge structures, enabling navigation from one knowledge object, i.e, a concept, to another concept. Conceptual hierarchies are also comprehensives knowledge maps, in which navigation is made along lines, from one node to another. Every time a node is hit, the corresponding information is revealed, i.e, what are the objects related to that node, and what are their attributes. By restricting and/or enlarging the set of attibutes or objects navigation becomes dynamic.

Implications in Formal Concept Analysis are an important feature for understanding the internal structure of data. The logic of data is part of the subsequent knowledge which is unfolded in the conceptual hierarchy. Nevertheless, it might be of interest to investigate dependencies between attributes in terms of implications. If (G, M, I) is a formal context, let A and B be subsets of M. Formally, an implication is just a pair (A, B) of subsets of M. We write $A \to B$. We say that the implication $A \to B$ holds in (G, M, I) if and only if every object which has all the attributes from A also has the attributes from B. It can be proven that the implications determine the conceptual hierarchy up to isomorphism and therefore offer an additional interpretation of the conceptual structure.

1.2. **Contextual Logic.** The interplay of the theory of Conceptual Graphs (CG) and Formal Concept Analysis (FCA) proved to be very fruitful in order to formalize the Elementary Logic, defined by I. Kant as "the theory of the three main essential functions of thinking: concepts, judgements and conclusions".

While FCA provides the mathematization of the classical theory of concepts, regarded as units of thoughts, CG are representing a formalization of the theory of judgements and conclusions.

Conceptual graphs can be understood as formal judgements. A conceptual graph is a labeled graph that represents the literal meaning of a sentence. Conceptual graphs express 'meaning in a form that is logically precise, humanly readable, and computationally tractable' ([12]). They serve as an intermediate language for translating computer-oriented formalisms to and from natural languages. With their graphic representation, they serve as a readable, but formal design and specification language.

In particular, they are graphs that consist of concept nodes, which bear references as well as types of the references. The concept boxes are connected by edges, which are used to express different relationships between the referents of the attached concept boxes. Sowa provides rules for formal deduction procedures on conceptual graphs; hence the system of conceptual graphs offers a formalization of conclusions too.

Hence, a conceptual graph is a bipartite graph having to kind of labeled nodes (concepts and relations) having the full description power of first order logic, FCA is a mathematical theory of deriving a conceptual hierarchy from a data table called *formal context*. Later on, these two theories have grown together, FCA being now part of a successful formalization of CG theory.

As Formal Concept Analysis provides a formalization of concepts, and as conceptual graphs offer a formalization of judgements and conclusions, a convincing idea is to combine these approaches to gain a unified formal theory for concepts, judgements and conclusions, i.e., a formal theory of elementary logic. In [19], Wille marked the starting point for a such a theory. There he provided a mathematization of conceptual graphs where the types of conceptual graphs are interpreted by formal concepts of a so-called power context family. The resulting graphs are called concept graphs. They form the mathematical basis for contextual logic.

Interaction with computers becomes more and more important in our daily lifes. The goal of conceptual graphs is to provide a graphical representation for logic which is able to support human reasoning. Possible applications of such a logical representation system have been described in [13], [14]. One interesting application is a consistent, graphical interface for database interaction, which has not been completely developed until today.

2. Conceptual graphs as database interface

The basic idea of using conceptual graphs as query interface to relational databases has been stated in the mid '70s by Sowa [11]. Almost twenty years

later, after a consistent development of database theory, a first attemp to use conceptual graphs for relational databases was stated ([8]), namely it presents the scheme of a relational table as a conceptual graph and queries also as conceptual graphs.

The mathematization of conceptual graphs started by Wille, and continued by Prediger was completed by F. Dau [4]. Here, he studies a calculus for the mathematization of conceptual graphs and their equivalence to first order logic. In database theory (see [1]) the equivalence of the relational calculus and the relational algebra is well known. Modern database languages extend this expressivity, considering especially aggregate functions. F. Dau and J. C. Hereth [5] developed Nested Concept Graphs with Cuts to express aggregate functions and negation.

In our research, we made use of the previous results mentioned above, but we decided that there is need for specialization and slightly modification of the theory developed so far. Hence, we have modeled a new form of conceptual graphs. They enable vizualisation of both the structure of a relational database, and queries, allowing a user friendly representation of queries and structure of the relational database. A software product has been developed, presented at [18] named CGDBInterface, which offers a graphical tool to query an existing relational database. The aim of our software tool is to connect to an existing database by giving the type and the name of the database, a login name and password, then the software offers the structure of the database in form of a conceptual graph. The relation between tables was not represented with conceptual graphs in earlier works. We also have been successful in our attempt to represent the relation between tables too by a conceptual graph, method which has been implemented in the above mentioned software. We illustrate our results by the next examples. The mathematical background of the proposed model for database scheme and queries is under development, since the particular structure of the conceptual graphs we use, imposes some modifications of the theory stated in [4].

Example 1. Let be the next relational `Sales` database scheme. Figure 1 shows the conceptual graph obtained by CGDBInterface software.

```
Customers(CustomerID, CompanyName, ContactName, Address, Phone)
Suppliers(SupplierID, CompanyName, ContactName, Address, Phone)
Orders(OrderID, CustomerID, OrderDate, RequiredDate)
Products(ProductID, ProductName, SupplierID, UnitPrice,
                UnitsInStock, UnitsOnOrder)
OrderDetails(OrderID, ProductID, UnitPrice, Quantity)
```

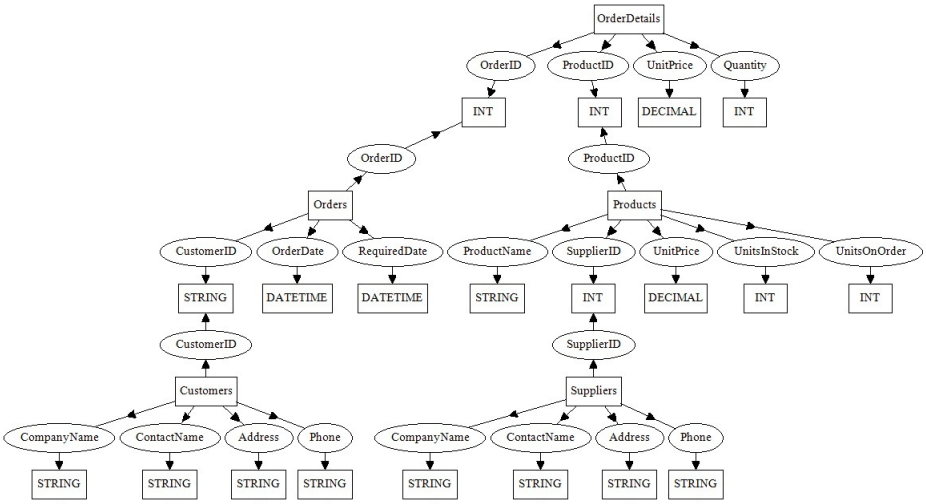


FIGURE 1. Conceptual graph for **Sales scheme**

Queries generated by software CGDBInterface covers all types of queries, starting with simple queries to very complex ones. The software uses hypostatic abstraction to model aggregation and nested queries.

Example 2. The next query is constructed by CGDBInterface, which has a graph editor. The SELECT SQL statement is generated by the software, then the generated query can be executed. In Figure 2 is presented the query, which searches for product names having been ordered for 3th of June, 2009. It gives the OrderID and order quantity too for every product. The selected attributes are marked by '?' in the query conceptual graph. To construct the query we have to join three tables: `Orders, OrderDetails` and `Products. The join attributes connect the tables.`

Example 3. The query in Figure 3 selects for every customer the number of products ordered by him. In order to use relations as objects in other relations, we have to reconsider these relations and to make use of a method introduced by Peirce, called *hypostatic abstractions*. *Hypostatic abstraction*, also known as hypostasis or subjectal abstraction, is a formal operation that takes an element of information, such as might be expressed in a proposition of the form X is Y, and conceives its information to consist in the relation between a subject and another subject, such as expressed in a proposition of the form X has Y-ness. The existence of the latter subject, here Y-ness, consists solely in the truth of those propositions that have the corresponding

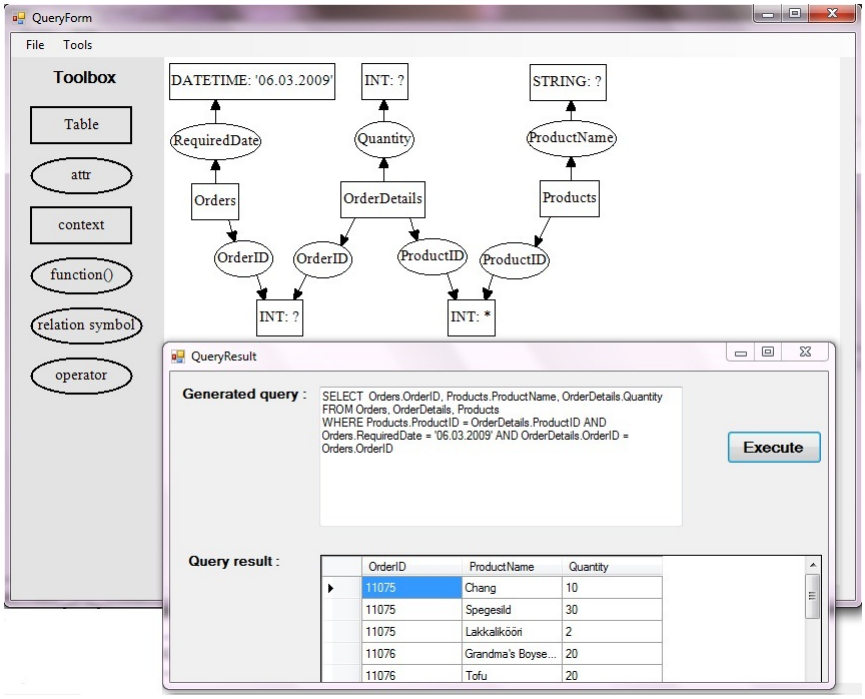


FIGURE 2. Conceptual graph for query involving join operation

concrete term, here Y, as the predicate. The object of discussion or thought thus introduced may also be called a hypostatic object ([9]).

In reasoning, we can consider the elements of a set and their properties, and sometimes the set itself as an element on its own having other properties. This shift in perspective transforms a relation into an object which then may be attached to relations again. In visualization, we use nested concept boxes as rectangle T1, which contains the joins of the tables: `Orders, OrderDetails` and `Products.` The result of the join is a relation too and the aggregation is applied to it.

3. Mining functional dependencies in relational databases

Functional dependencies (FDs shortly) are the most common integrity constraints encountered in databases. FDs are very important in relational database design to avoid data redundancy. Extracting FDs from a relational table is a crucial task to understand data semantics useful in many database applications. The subject of detecting functional dependencies in relational tables was studied for a long time and recently addressed with a data mining

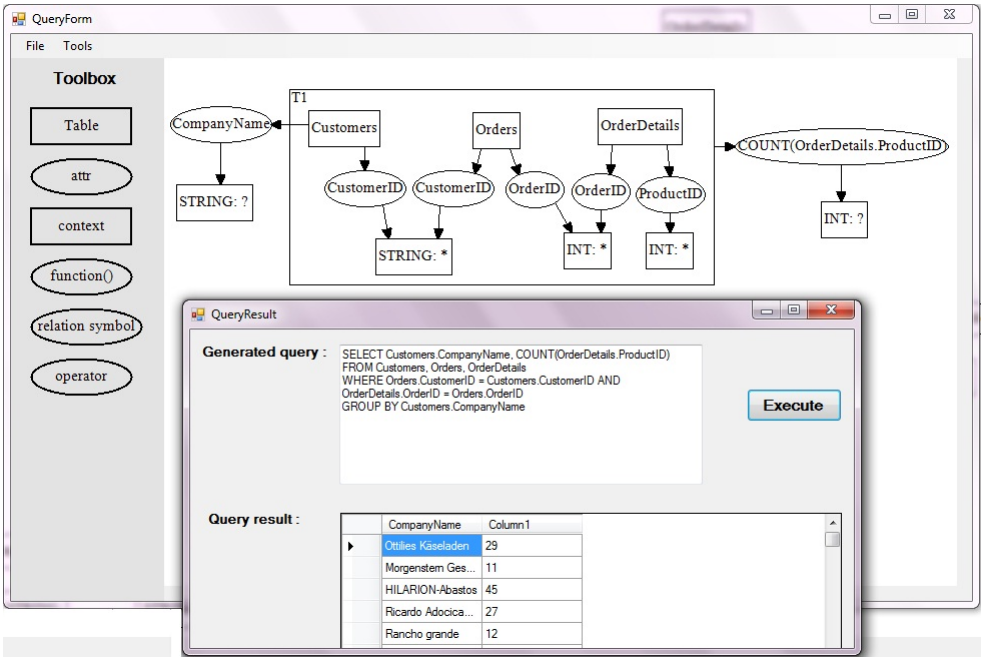


FIGURE 3. Conceptual graph for query involving grouping and aggregate functions

viewpoint. Baixeries [2] gives an interesting framework to mine functional dependencies using Formal Context Analysis.

Hereth [7] presents how some basic concepts from database theory are translated into the language of Formal Concept Analysis. The definition of the formal context of functional dependencies for a relational table can also be found in [7]. Regarding to this definition, the context's attributes are the columns (named attributes) of the table, the tuple pairs of the table will be the objects of the context. In [7] you can find the proposition which asserts that in this context, implications are essentially functional dependencies between the columns of the relational database table.

A detailed analysis and complex examples of the formal context of functional dependencies for a relational table are presented in [15]. The novelty of our method is that it builds inverted index files in order to optimize the construction of the formal context of functional dependencies.

We implemented the method presented in [15] and completed it with a software tool, which analyzes an existing relational database table. Our software named FCAFuncDepMine (see [17]) can connect to a MS SQL Server,

Oracle or MySQL database by giving the type and the name of the database, a login name and password, then the software offers a list of identified table names which can be selected for possible functional dependency examination.

It constructs the formal context of functional dependencies, uses Conexp [22] to build the concept lattice and to determine the implications in this context, which corresponds to functional dependencies in the analyzed table. The software can be used in relational database design and for detecting functional dependencies in existing tables, respectively. A detailed data analysis using software FCAFuncDepMine is presented in [16].

Example 4. Let be a complex example, which is illustrative for our work. The next table:

`OrderDetail [OrderID, CustomerID, OrderDate, CompanyName, Address, Phone, City, Quantity, UnitPrice, ProductID]`

stores orders of different customers together with order details information as product ID, order price and quantity too. `CompanyName` is the name of customer, `Address` is the customer's address and the `City` is his city. In Figure 4 there are the first rows from table `OrderDetail` .

OrderID	CustomerID	OrderDate	CompanyName	Address	Phone	City	Quantity	UnitPrice	ProductID
10248	VINET	1996-07-04 00:00:00.000	Vins et alcools Chevalier	59 rue de l'Abbaye	26.47.15.10	Reims	12	14.00	11
10248	VINET	1996-07-04 00:00:00.000	Vins et alcools Chevalier	59 rue de l'Abbaye	26.47.15.10	Reims	10	9.80	42
10248	VINET	1996-07-04 00:00:00.000	Vins et alcools Chevalier	59 rue de l'Abbaye	26.47.15.10	Reims	5	34.80	72
10249	TOMSP	1996-07-05 00:00:00.000	Toms Spezialitäten	Luisenstr. 48	0251-031259	Münster	9	18.60	14
10249	TOMSP	1996-07-05 00:00:00.000	Toms Spezialitäten	Luisenstr. 48	0251-031259	Münster	40	42.40	51
10250	HANAR	1996-07-08 00:00:00.000	Hanari Carnes	Rua do Paço, 67	(21) 555-0091	Rio de Janeiro	10	7.70	41
10250	HANAR	1996-07-08 00:00:00.000	Hanari Carnes	Rua do Paço, 67	(21) 555-0091	Rio de Janeiro	35	42.40	51
10250	HANAR	1996-07-08 00:00:00.000	Hanari Carnes	Rua do Paço, 67	(21) 555-0091	Rio de Janeiro	15	16.80	65
10251	VICTE	1996-07-08 00:00:00.000	Victuailles en stock	2, rue du Commerce	78.32.54.86	Lyon	6	16.80	22
10251	VICTE	1996-07-08 00:00:00.000	Victuailles en stock	2, rue du Commerce	78.32.54.86	Lyon	15	15.60	57
10251	VICTE	1996-07-08 00:00:00.000	Victuailles en stock	2, rue du Commerce	78.32.54.86	Lyon	20	16.80	65
10252	SUPRD	1996-07-09 00:00:00.000	Suprêmes délices	Boulevard Tirou, 255	(071) 23 67 22 20	Charleroi	40	64.80	20
10252	SUPRD	1996-07-09 00:00:00.000	Suprêmes délices	Boulevard Tirou, 255	(071) 23 67 22 20	Charleroi	25	2.00	33
10252	SUPRD	1996-07-09 00:00:00.000	Suprêmes délices	Boulevard Tirou, 255	(071) 23 67 22 20	Charleroi	40	27.20	60
10253	HANAR	1996-07-10 00:00:00.000	Hanari Carnes	Rua do Paço, 67	(21) 555-0091	Rio de Janeiro	20	10.00	31
10253	HANAR	1996-07-10 00:00:00.000	Hanari Carnes	Rua do Paço, 67	(21) 555-0091	Rio de Janeiro	42	14.40	39
10253	HANAR	1996-07-10 00:00:00.000	Hanari Carnes	Rua do Paço, 67	(21) 555-0091	Rio de Janeiro	40	16.00	49
10254	CHOPS	1996-07-11 00:00:00.000	Chop-suey Chinese	Hauptstr. 29	0452-076545	Bern	15	3.60	24
10254	CHOPS	1996-07-11 00:00:00.000	Chop-suey Chinese	Hauptstr. 29	0452-076545	Bern	21	19.20	55
10254	CHOPS	1996-07-11 00:00:00.000	Chop-suey Chinese	Hauptstr. 29	0452-076545	Bern	21	8.00	74
10255	RICSU	1996-07-12 00:00:00.000	Richter Supermarkt	Grenzacherweg 237	0897-034214	Genève	20	15.20	2
10255	RICSU	1996-07-12 00:00:00.000	Richter Supermarkt	Grenzacherweg 237	0897-034214	Genève	35	13.90	16
10255	RICSU	1996-07-12 00:00:00.000	Richter Supermarkt	Grenzacherweg 237	0897-034214	Genève	25	15.20	36
10255	RICSU	1996-07-12 00:00:00.000	Richter Supermarkt	Grenzacherweg 237	0897-034214	Genève	30	44.00	59
10256	WELLI	1996-07-15 00:00:00.000	Wellington Importadora	Rua do Mercado, 12	(14) 555-8122	Resende	15	26.20	53
10256	WELLI	1996-07-15 00:00:00.000	Wellington Importadora	Rua do Mercado, 12	(14) 555-8122	Resende	12	10.40	77
10257	HILAA	1996-07-16 00:00:00.000	HILARION-Abastos	Carrera 22 con Av...	(5) 555-1340	San Cristóbal	25	35.10	27
10257	HILAA	1996-07-16 00:00:00.000	HILARION-Abastos	Carrera 22 con Av...	(5) 555-1340	San Cristóbal	6	14.40	39
10257	HILAA	1996-07-16 00:00:00.000	HILARION-Abastos	Carrera 22 con Av...	(5) 555-1340	San Cristóbal	15	10.40	77
10258	ERNSH	1996-07-17 00:00:00.000	Ernst Handel	Kirchgasse 6	7675-3425	Graz	50	15.20	2
10258	ERNSH	1996-07-17 00:00:00.000	Ernst Handel	Kirchgasse 6	7675-3425	Graz	65	17.00	5
10258	ERNSH	1996-07-17 00:00:00.000	Ernst Handel	Kirchgasse 6	7675-3425	Graz	6	25.60	32
10259	CENTC	1996-07-18 00:00:00.000	Centro comercial Moct...	Sierras de Granad...	(5) 555-3392	México D.F.	10	8.00	21
10259	CENTC	1996-07-18 00:00:00.000	Centro comercial Moct...	Sierras de Granad...	(5) 555-3392	México D.F.	1	20.80	37
10260	OTTIK	1996-07-19 00:00:00.000	Ottilies Käseladen	Mehrheimerstr. 369	0221-0644327	Köln	16	7.70	41
10260	OTTIK	1996-07-19 00:00:00.000	Ottilies Käseladen	Mehrheimerstr. 369	0221-0644327	Köln	50	15.60	57

FIGURE 4. First rows from table **OrderDetail**

The conceptual lattice for context of functional dependencies in table `OrderDetail` is represented in Figure 5.

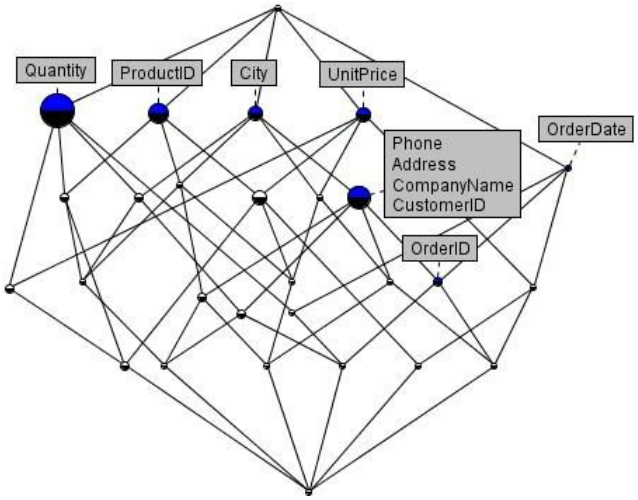


FIGURE 5. Conceptual lattice for context of functional dependencies in table **OrderDetail**

The implications in this lattice, which are functional dependencies in the table can be seen as follows: concept with label `OrderID` is subconcept of concept with labels `Phone, Address, CompanyName, CustomerID` and of concept with label `OrderDate` too. So we can read the following functional dependencies.

$$OrderID \rightarrow CustomerID, CompanyName, Address, Phone$$

```
1 < 3996 > CustomerID ==> CompanyName City Address Phone;
2 < 3996 > CompanyName ==> CustomerID City Address Phone;
3 < 3996 > Address ==> CustomerID CompanyName City Phone;
4 < 3996 > Phone ==> CustomerID CompanyName City Address;
5 < 385 > OrderID ==> CustomerID CompanyName City Address Phone OrderDate;
6 < 385 > City OrderDate ==> CustomerID CompanyName Address Phone OrderID;
7 < 194 > OrderDate ProductID ==> UnitPrice;
8 < 190 > City UnitPrice Quantity ==> CustomerID CompanyName Address Phone OrderID OrderDate ProductID;
9 < 190 > OrderDate UnitPrice Quantity ==> CustomerID CompanyName City Address Phone OrderID ProductID;
10 < 190 > CustomerID CompanyName City Address Phone OrderID OrderDate ProductID UnitPrice ==> Quantity;
```

FIGURE 6. Implications, namely functional dependencies in the table **OrderDetail**

Another implication is: $OrderID \rightarrow OrderDate$. The possible implications given by Conexp software are in Figure 6. The user can make attribute

exploration to decide which implication is valid. The number before the implication can help us. The implications labeled with biger numbers usually are valid. Having these functional dependencies we can propose the decomposition of the table `OrderDetail.` It is clear that the information about customers have to be in a separate table, candidate keys is `Customers` table are: `CustomerID, CompanyName, Phone, Address.` If we introduce the same name for different customers, the `CompanyName` attribute will appear in a concept, which is subconcept of the concept with label `CustomerID, Phone, Address.` The same results if we introduce different companies with the same address. The `CustomerID` attribute is functionally dependent on `OrderID,` so we will design an `Orders` table too. The proposed tables are:

```
Customers(CustomerID, CompanyName, City, Address, Phone)
Orders(OrderID, CustomerID, OrderDate)
OrderDetails(OrderID, ProductID, UnitPrice, Quantity)
```

4. Future work

There are two directions we intend to apply conceptual knowlegde processing. We are developing the mathematical backgound for the schema and query model with conceptual graphs of a relational database.

On the other hand, we intend to analyze semistructured data presented in XML form. Many papers are dedicated to the theory of functional dependency in XML data. Our aim is to construct the context of functional dependencies for an XML tree and generate implications in it, which will be functional dependencies in XML data.

References

[1] Abiteboul, S., Hull, R., Vianu, V.: *Foundations of Databases.* Addison-Wesley, Reading - Menlo - New York (1995)

[2] Baixeries, J.: *A Formal Concept Analysis Framework to Mine Functional Dependencies*, Proceedings of Mathematical Methods for Learning, (2004).

[3] Carpineto, C., Romano, G.: *Concept Data Analysis, Theory and Applications*, Wiley and Sons, 2004.

[4] Dau, F.: *The Logic System of Concept Graphs with Negation And Its Relationship to Predicate Logic*, LNCS, Vol. 2892, Springer Berlin / Heidelberg (2003)

[5] Dau, F., Hereth, J. C.: *Nested Concept Graphs: Mathematical Foundations and Applications for Databases.* In: Ganter, B.; de Moor, A. (eds.): Using Conceptual Structures. Contributions to ICCS 2003. Shaker Verlag, Aachen, (2003), pp. 125-139.

[6] Devlin, K.: *Infosense - Turning Information into Knowledge*, Freeman, New York, 1999.

[7] Hereth, J.: *Relational Scaling and Databases.* Proceedings of the 10th International Conference on Conceptual Structures: Integration and Interfaces LNCS 2393, Springer Verlag (2002) pp. 62-76

[8] Boksenbaum, C., Carbonneill, B., Haemmerle O., Libourel, T.: *Conceptual Graphs for Relational Databases* in Conceptual Graphs for Knowledge Representation., Guy, M. W., Moulin B., Sowa, J. F. eds. Lecture Notes in AI 699, Springer-Verlag, Berlin (1993).

[9] Peirce, C.S.: *The Simplest Mathematics*, in Collected Papers, CP4.235, CP4.227-323, 1902.

[10] Silberschatz, A., Korth, H. F.,Sudarshan, S.: *Database System Concepts*, McGraw-Hill, Fifth Edition, (2005)

[11] Sowa, J. F.: *Conceptual Graphs for a Database Interface.* In: IBM Journal of Research and Development, vol. 20, no. 4, (1976) pp. 336-357.

[12] Sowa, J. F.: *Conceptual Structures: Information Processing in Mind and Machine.* Addison Wesley Publishing Company Reading, (1984).

[13] Sowa, J. F.: *Conceptual graphs summary*, in Nagle, T. E.; Nagle, J. A.; Gerholz, L. and Eklund, P. W. (editors): Conceptual Structures: Current Research and Practice, Ellis Horwood, (1992), pp 3-51.

[14] Sowa, J. F.: *Knowledge Representation: Logical, Philosophical, and Computational Foundations*, Brooks Cole Publishing Co., Pacific Grove, CA. (2000)

[15] Janosi Rancz, K. T., Varga, V.: *A Method for Mining Functional Dependecies in Relational Database Design Using FCA*, Studia Univ. Babeş-Bolyai, Informatica, Vol. LIII, Nr. 1(2008).

[16] Janosi Rancz, K.T., Varga, V., Puskas, J.: *A Software Tool for Data Analysis Based on Formal Concept Analysis*, Studia Univ. Babeş-Bolyai, Informatica, Vol. LIII, Nr. 2(2008).

[17] Varga, V., Janosi Rancz, K. T.: *A Software Tool to Transform Relational Databases in order to Mine Functional Dependencies in it Using Formal Concept Analysis*, Proc. of the Sixth International Conference on Concept Lattices and Their Applications, Olomouc, 21-23 October, 2008. pp. 1-8.

[18] Varga, V., Săcărea, C., Takács, A.: *A Software Tool for Interactive Database Access using Conceptual Graphs*, International Conference Knowledge Engineering Principles and Techniques, KEPT 2009, Cluj-Napoca, July 2-4.

[19] Wille, R.: *Conceptual Graphs and Formal Concept Analysis*, Lecture Notes In Computer Science; Vol. 1257, Proceedings of the Fifth International Conference on Conceptual Structures: Fulfilling Peirce's Dream, Springer Verlag (1997), pp 290 - 303.

[20] Wille, R.: *Conceptual Contents as Information - Basics for Contextual Judgement Logic*, Conceptual Structures for Knowledge Creation and Communication, ICCS 2003, LNAI 2746, Springer, pp. 1-15, 2003.

[21] Wille, R.: *Methods of Conceptual Knowledge Processing*, ICFCA 2006, LNAI 3874, Springer, pp. 1-29, 2006.

[22] Serhiy A. Yevtushenko: *System of Data Analysis "Concept Explorer"*. (In Russian). Proceedings of the 7th National Conference on Artificial Intelligence KII-2000, p. 127-134, Russia, 2000.

Babes-Bolyai University, Faculty of Mathematics and Computer Science, Cluj-Napoca, Romania

E-mail address: csacarea@math.ubbcluj.ro

E-mail address: ivarga@nessie.cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

DEFAULT REASONING BY ANT COLONY OPTIMIZATION

MIHAIELA LUPEA

Abstract. Drawing conclusions from incomplete information by making default assumptions represents default reasoning. Default logics, a class of nonmonotonic logical systems, formalize this type of reasoning using special inference rules, the defaults. During the inferential process, a default theory is extended with plausible conclusions (beliefs) obtaining default extensions. The very high theoretical complexity of the extension computation problem suggests the use of non-deterministic techniques for an efficient computation. In this paper we propose a uniform theoretical approach of the extension computation problem for all default logics (classical, justified, constrained, rational) applying Ant Colony Optimization metaheuristic.

1. Introduction

A lot of applications from Artificial Intellingence domain suppose reasoning with incomplete information. The specificity of this reasoning process, the *nonmonotonicity*, imposes that in the light of new information, some already derived conclusions (which are only consistent, not necessarily true) to be invalidated.

A special case of nonmonotonig reasoning, *default reasoning*, uses reasoning patterns of the form: "in the absence of information to the contrary of... it is consistent to assume that...". In the deductive process, default assumptions are applied in order to derive conclusions, called *beliefs*.

A class of nonmonotonic logical systems, *default logics* was introduce to formalize the default reasoning. Based on first-order logic, default logics use special inference rules, called *defaults*, to model the above nonmonotonic reasoning patterns. The differences among the versions (classical, justified, constrained, rational) of default logic are caused by the semantics of the defaults.

Received by the editors: November 1, 2009.

2010 *Mathematics Subject Classification.* 03B79, 68T15, 68T27, 68T20.

1998 *CR Categories and Descriptors.* I.2[**Artificial Intelligence**]: Logics in artificial intelligence – *default logics, nonmonotonic reasoning*.

Key words and phrases. default logics, nonmonotonic reasoning, ant colony optimization.

Default logics provide a very expressive representation of an incomplete knowledge base (default theory), ruled by laws that are true with a few exceptions, using a simple syntactic formalism (first-order formulas and the defaults). The default reasoning process consists of combining the classical deduction with the defaults in order to derive new facts (beliefs), and obtain the *default extensions*, even if some information are not available.

This great power of the inferential process causes a high level of theoretical complexity. The problem of computing the extensions (classical, justified, constrained, rational) of a default theory is $\sum_2^P = NP^{NP}$-complete, this class belonging to the second level of the polynomial hierarchy of complexity classes based on calculus with oracles. For an efficient computation non-deterministic approaches must be used.

Automated proof systems for default logics proposed in the literature [1, 2, 6, 15, 16] and based on the well known classical theorem proving methods as resolution, connection method, semantic tableaux method, have good performances only for particular classes of default theories and are not efficient for general non-trivial default theories.

In the papers [10, 11, 12] new generation systems for default reasoning were introduced. These are based on heuristics such Genetic Algorithms, Ant Colony Optimization and Local Search, in order to overcome the high complexity and to obtain efficient reasoning systems. ANTDEL [11] is a system which uses Ant Colony Optimization to compute the classical default extension of a default theory that is equivalent to a logic program.

Inspired from the good performances of ANTDEL, in this paper we propose a uniform theoretical approach of the *extension computation problem* (ECP) for all versions (classical, justified, constrained, rational) of default logic using Ant Colony Optimization metaheuristic.

The paper is structured as follows. In Section 2 the main theoretical aspects of default logics are presented. A heuristic approach of the extension computation problem for default logics (classical, justified, constrained, rational) is introduced in Section 3. In section 4 an Ant Colony Optimization based procedure to compute all types of extensions for a default theory is proposed. Conclusions and future work are outlined in Section 5.

2. Default logics

Definition 1. [13] A *default theory* $\Delta = (D, W)$ consists of a set D of *default rules* and W, a set of consistent first-order logic formulas (the *facts*). A *default* has the form $d = \frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}$, where: α is called *prerequisite*, $\beta_1, \ldots, \beta_m$ are called *justifications* and γ is called *consequent*.

A default $d = \frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}$ can be applied and thus derive γ if α is believed and it is consistent to assume $\beta_1, \ldots, \beta_m$ (meaning that $\neg\beta_1, \ldots, \neg\beta_m$ are not believed).

Using the classical inference rules and the defaults, the set of facts, W, can be extended with new formulas, called *nonmonotonic theorems* (*beliefs*), obtaining *extensions*. The set of all the defaults used in the construction of an extension is called the *generating default set* for that extension.

The results from [8] show that default theories can be represented by unitary theories (all the defaults have the syntax: $d = \frac{\alpha:\beta}{\gamma}$), in such a way that extensions (classical, justified, constrained, rational) are preserved. In the paper we will use only unitary default theories and the following notations:
$Prereq(d) = \alpha$, $Justif(d) = \beta$, $Conseq(d) = \gamma$,
$Prereq(D) = \bigcup_{d\in D} Prereq(d)$, $Justif(D) = \bigcup_{d\in D} Justif(d)$,
$Conseq(D) = \bigcup_{d\in D} Conseq(d)$,
$Th(U) = \{A | U \vdash A\}$, the classical deductive closure of the set U of formulas.

Definition 2. [16] A set X of defaults is *grounded* in the set of facts W if there is an enumeration $\langle d_i \rangle_{i\in I}$ of the defaults from X such that:
$\forall i \in I$, $W \cup Prereq(\{d_0, d_1, ..., d_{i-1}\}) \vdash Prereq(d_i)$.

The following theorems provide global characterizations for default extensions using the generating default sets.

Theorem 1. [14] Let (D, W) be a default theory, and let E be a set of formulas. E is a ***classical extension*** of (D, W) if and only if $E = Th(W \cup Conseq(D'))$ for a maximal set $D' \subseteq D$ such that D' is grounded in W and the conditions:

- $\forall d = \frac{\alpha:\beta}{\gamma} \in D'$: $W \cup Conseq(D') \cup \{\beta\}$ is consistent;
- $\forall d = \frac{\alpha:\beta}{\gamma} \notin D'$: $W \cup Conseq(D') \cup \{\beta\}$ is inconsistent **or** $W \cup Conseq(D') \cup \{\neg\alpha\}$ is consistent

are satisfied.

Theorem 2. [6] Let (D, W) be a default theory, and let E, J be sets of formulas. (E, J) is a ***justified extension*** of (D, W) if and only if $E = Th(W \cup Conseq(D'))$ and $J = Justif(D')$ for a maximal set $D' \subseteq D$ such that D' is grounded in W and the conditions:

- $\forall d = \frac{\alpha:\beta}{\gamma} \in D'$: $W \cup Conseq(D') \cup \{\beta\}$ is consistent;

are satisfied.

From the above theorems we remark that a classical/justified default extension is a consistent set and these two logics satisfy the *weak regularity property*, expressed as an individual consistency condition (stronger in justified logic than in classical default logic) for the justifications of the generating defaults.

Theorem 3. [14] Let (D, W) be a default theory, and let E, C be sets of formulas. (E, C) is a ***constrained extension*** of (D, W) if and only if $E = Th(W \cup Conseq(D'))$ and $C = Th(W \cup Conseq(D') \cup Justif(D'))$ for a maximal set $D' \subseteq D$ such that D' is grounded in W and the condition:

$W \cup Conseq(D') \cup Justif(D')$ is a consistent set;

is satisfied.

Theorem 4. [6] Let (D, W) be a default theory, and let E, C be sets of formulas. (E, C) is a ***rational extension*** of (D, W) if and only if $E = Th(W \cup Conseq(D'))$ and $C = Th(W \cup Conseq(D') \cup Justif(D'))$ for a maximal set $D' \subseteq D$ such that D' is grounded in W and the conditions:

- $W \cup Conseq(D') \cup Justif(D')$ is a consistent set;
- $\forall d \in D \backslash D'$ we have:
  $W \cup Conseq(D') \cup \neg Prereq(d)$ is consistent **or**
  $W \cup Conseq(D') \cup Justif(D' \cup d)$ is inconsistent;

are satisfied.

Theorems 3 and 4 show that the *strong regularity property* is common to these logics. According to this property, the reasoning process is guided by a consistent context C containing the actual extension E and the assumptions (justifications) of the applied defaults. For the rational default logic the set of generating defaults must be maximal-active [9] with respect to W and E.

Another important formal property is *semi-monotonicity* which expresses a "monotonicity" with respect to the set of defaults. Justified and constrained default logics have this desirable property (useful in the computation of an extension), with an important consequence: the existence of a justified/constrained extension for any default theory. The existence of a classical/rational extension for a default theory is not guaranteed and semi-monotonicity property is not satisfied by classical/rational default logics.

From theorems 1, 2, 3 and 4 we can conclude that all four types of extensions are deductive closures of the set W (explicit content) and the consequents of the generating default set D'(implicit content).

According to the initial fixed-point definitions of all variants of default logic we have the following definitions:

Definition 3. Let E_1 be a *clas*sical extension, (E_2, J) be a *just*ified extension, (E_3, C_3) be a *cons*trained extension and (E_4, C_4) be a *rat*ional extension of the default theory (D, W). The *generating default sets* are:
$GD_{(D,W)}^{E_1,clas} = \left\{ \frac{\alpha:\beta}{\gamma} \in D | if\ \alpha \in E_1\ and\ E_1 \cup \{\beta\}\ consistent,\ \ then\ \gamma \in E_1 \right\}$
for the classical extension E_1;
$GD_{(D,W)}^{(E_2,J),just} = \left\{ \frac{\alpha:\beta}{\gamma} \in D | if\ \alpha \in E_2\ and\ \forall \eta \in J \cup \{\beta\} : E2 \cup \{\gamma, \eta\}\ consistent \right.$
then $\gamma \in E_2, \beta \in J\}$ for the justified extension (E_2, J);
$GD_{(D,W)}^{(E_3,C_3),cons} = \left\{ \frac{\alpha:\beta}{\gamma} \in D | if\ \alpha \in E_3\ and\ C_3 \cup \{\beta, \gamma\}\, consistent \right.$
$then\ \gamma \in E_3,\ \beta, \gamma \in C_3\}$ for the constrained extension (E_3, C_3);
$GD_{(D,W)}^{(E_4,C_4),rat} = \left\{ \frac{\alpha:\beta}{\gamma} \in D | if\ \alpha \in E_4\ and\ C_4 \cup \{\beta\}\ consistent, then\ \gamma \in E_4, \right.$
$\beta, \gamma \in C_4\}$ for the rational extension (E_4, C_4).

3. A heuristic approach of the extension computation problem

In this section we extend the heuristic approach of the classical extension computation problem from [11, 12] to all types of default extensions: justified, constrained, rational.

The theorems from the previous section show that the problem of finding extensions can be reduced to the problem of finding the generating default sets for those extensions.

In this heuristic approach we need to define a search space for the generating default sets and an evaluation function to compute the fitness of each element of this space according to the definitions of different types of default extensions.

Definition 4. For a default theory (D, W) we define the search space as the set $CGD = 2^D$, representing all possible configurations, called *candidate generating default sets*.

Definition 5. Let (D, W) be a default theory and $X \in CGD$, a candidate generating default set. We define:
- the *candidate extension* associated to X: $CE(X) = Th(W \cup Conseq(X))$;
- the *candidate context* associated to X: $CC(X) = Th(W \cup Conseq(X) \cup Justif(X))$;
- the *candidate support set* associated to X: $CJ(X) = Justif(X)$.

For defining the evaluation function we need four intermediate functions: $f_0^{type}, f_1^{type}, f_2^{type}, f_3^{type}$, where *type=clas* for *clas*sical extensions, *type=just* for *just*ified extensions, *type=cons* for *cons*trained extensions and *type=rat* for *rat*ional extensions.

Using f_0^{type} we check if the candidate extension (for classical and justified default logics) or the candidate context (for constrained and rational default logics) is consistent or not:

$$f_0^{clas}(X), f_0^{just}(X) = \begin{cases} 0 & \text{if } CE(X) \text{ is consistent} \\ 1 & \text{otherwise} \end{cases}$$

$$f_0^{cons}(X), f_0^{rat}(X) = \begin{cases} 0 & \text{if } CC(X) \text{ is consistent} \\ 1 & \text{otherwise} \end{cases}$$

f_1^{type} rates the correctness of the candidate generating default set according to the definitions of different types of default extensions.

$f_1^{type}(X) = \sum_{i=1}^{n} \pi(d_i)$, where $D = \{d_1, d_2, ..., d_n\}$

The table below defines $\pi(d_i) \in Z$, a penalty for each default of D, indicating if a default from X was correctly/wrongly applied and a default from $D - X$ was correctly/wrongly not applied in order to generate the candidate extension $CE(X)$.

$d_i \in X$	C_{pre}	$\mathrm{C}_{justif}^{type}$	$\pi(d_i)$	$d_i = \frac{\alpha_i : \beta_i}{\gamma_i}$
true	true	true	0	correctly applied
true	true	false	k	wrongly applied
true	false	true	k	wrongly applied
true	false	false	k	wrongly applied
false	true	true	k	wrongly not applied
false	true	false	0	correctly not applied
false	false	true	0	correctly not applied
false	false	false	0	correctly not applied

$C_{pre}(X, d_i) : CE(X) \vdash \alpha_i$ is the groundness condition for the applied default d_i.

The conditions C_{justif}^{type}, according to Definition 3, imply the *weak regularity property* for classical/justified default logics and the *strong regularity property* for constrained/rational default logics.

- $C_{justif}^{clas}(X, d_i)$: the set $CE(X) \cup \{\beta_i\}$ is consistent;
- $C_{justif}^{just}(X, d_i)$: $\forall \eta \in CJ(X) \cup \{\beta_i\}$, the set $CE(X) \cup \{\eta, \gamma_i\}$ is consistent;
- $C_{justif}^{cons}(X, d_i)$: the set $CC(X) \cup \{\beta_i, \gamma_i\}$ is consistent;
- $C_{justif}^{rat}(X, d_i)$: the set $CC(X) \cup \{\beta_i\}$ is consistent.

f_2^{type} rates the level of groundness of the candidate generating default set as follows: $f_2^{type}(X) = card(Y)$, where Y is the biggest grounded set $Y \subseteq X \in CGD$.

f_3^{type} checks the groundness property of X:

$$f_3^{type}(X) = \begin{cases} 0 & \text{if } X \text{ is grounded} \\ 1 & \text{otherwise} \end{cases}$$

Definition 6. For a default theory (D, W) the *evaluation function* for a candidate generating default set $X \in CGD$ of an extension of $type \in \{clas, just, cons, rat\}$ is defined by:

$eval^{type} : CGD \longmapsto Z \cup \{\bot, \top\}$ with $\forall z \in Z, \bot < z < \top$

if $f_0^{type}(X) = 1$
  then $eval^{type}(X) = \top$
  else if $f_1^{type}(X) = 0$ and $f_3^{type}(X) = 0$
    then $eval^{type}(X) = \bot$
    else $eval^{type}(X) = f_1^{type}(X) - f_2^{type}(X)$
  endif
endif

The following theorem provides a necessary and sufficient condition for a set of defaults to be a generating set for an extension, using $eval^{type}$.

Theorem 7.[7] Let (D, W) be a default theory. A candidate generating default set $X \in CGD$ generates an extension of $type \in \{clas, just, cons, rat\}$ if and only if $eval^{type}(X) = \bot$.

This evaluation function can be used by different non-deterministic approaches as Genetic Algorithms and Ant Colony Optimization, to evaluate the candidate generating default sets from the search space. The efficiency of these approaches derives from the fact that the search space is not entirely explored. An initial candidate is progressively improved in order to obtain a solution for ECP.

4. Computing default extensions using Ant Colony Optimization

Based on the theoretical considerations of ANTDEL [11, 12], in this section we propose a uniform theoretical approach of the extension computation problem for all versions (classical, justified, constrained, rational) of default logic using Ant Colony Optimization.

Ant Colony Optimization(ACO) is a population-based metaheuristic used successfully to solve difficult problems which can be reduced to finding good paths through graphs.

The collective behavior of ants, seeking for food and cooperating via the environment (the pheromone deposited on the paths), was the inspiration for this optimization technique.

Informally, the extension computation problem is represented as a search problem and it is solved using the ACO metaheuristics as follows:

- Given a default theory, a default graph, representing all the candidate generating default sets, is built. The default rules and two particular vertices: *in* and *out* form the set of vertices. The arcs connect the vertices containing compatible defaults. Each arc is weighted by pheromone which is initialized to 1 and is updated (deposited and evaporated) during the search process.
- An ant colony must find an optimal path from *in* to *out* in the graph, path which corresponds to a generating default set for an extension.
- The ants individually build their paths from *in* to *out* (corresponding to candidate generating default sets), using a probabilistic choice biased on the pheromone deposited on the arcs and a local evaluation function.
- The pheromone evaporates in time and increases on better paths. Therefore, during the optimization process, the paths are progressively improved in order to find an optimal solution (according to the evaluation function from the previous section).

The following definitions formalize the above description using the concepts defined in the previous section.

Definition 7. Let (D, W) be a default theory. The *default graph of* $type \in \{clas, just, cons, rat\}$ associated to the default theory is $G^{type}(D, W) = (D \cup \{in, out\}, A^{type})$. The *arc set*, A^{type}, is defined as follows:

$$A^{type} = \{(in, out)\} \cup \{(d, out), \forall d \in D\} \cup$$
$$\cup \left\{(d, d') \in D^2 | d \neq d' \text{ and } C^{type}(d, d') \text{ is true}\right\} \cup$$
$$\cup \left\{(in, d), \forall d = \frac{\alpha:\beta}{\gamma} \in D | W \vdash \alpha \text{ and } W \cup \{\beta, \gamma\} \text{ consistent}\right\},$$

where: $d = \frac{\alpha:\beta}{\gamma}, d' = \frac{\alpha':\beta'}{\gamma'}$ and the conditions C^{type} are:

$C^{clas}(d, d') = C^{just}(d, d')$:
$W \cup \{\beta, \gamma, \gamma'\}$ *and* $W \cup \{\beta', \gamma, \gamma'\}$ consistent;

$C^{cons}(d, d') = C^{rat}(d, d') : W \cup \{\beta, \beta', \gamma, \gamma'\}$ consistent; Each arc $(i, j) \in A^{type}$ is weighted by a positive real number $\varphi_{i,j}$, called *artificial pheromone.*

In order to decrease the search space, the arc set of the default graph is built using the following observations:

- The arc (in, d) is added to the arc set if d is applicable to W and its application will not lead to a contradiction. This condition is the same for all versions of default logic.

- There is an arc between two defaults d and d' only if they are "compatible", meaning that they can belong together to a candidate generating default set. $C^{clas}(d,d'), C^{just}(d,d'), C^{cons}(d,d'), C^{rat}(d,d')$ express these "compatibility" conditions which are particular cases of the weak/strong regularity properties for the default logics.

Definition 8. For the default theory (D,W) and a path $P=(in,...,out)$ in $G^{type}(D,W)$, $D_P = D \cap P \in CGD$ represents a candidate generating default set of $type \in \{clas, just, cons, rat\}$.

Definition 9. Let P be a path in the default graph $G^{type}(D,W)$ and $d = \frac{\alpha:\beta}{\gamma} \in D - P$.

- d is grounded in P if $W \cup Conseq(D_P) \vdash \alpha$,
- d is *type*-compatible with P if $C^{type}(P,d)$ is true,
  where: $D_P = P \cap D$, $type \in \{clas, just, cons, rat\}$
  $C^{clas}(P,d) : W \cup Conseq(D_P) \cup \{\beta\}$ consistent;
  $C^{just}(P,d) : \forall \eta \in Justif(D_P) \cup \{\beta\} :$
  $W \cup Conseq(D_P) \cup \{\eta, \gamma\}$ consistent;
  $C^{cons}(P,d) : W \cup Conseq(D_P) \cup Justif(D_P) \cup \{\beta, \gamma\}$
  consistent;
  $C^{rat}(P,d) : W \cup Conseq(D_P) \cup Justif(D_P) \cup \{\beta\}$
  consistent
- the local evaluation function is defined by:

$$loc^{type}(P,d) = \begin{cases} 1 & \text{if } d \text{ is grounded in } P \text{ and} \\ & \quad d \text{ is } type\text{-compatible with } P \\ 0 & \text{otherwise} \end{cases}$$

Remarks:

- The compatibility conditions for all types of versions are the apllicability conditions of the defaults and are used to apply one by one the defaults and to build candidate generating default sets (paths from *in* to *out* in the default graph).
- The local evaluation function is used to choose efficiently the next vertex in the path (the next default to be applied) in order to reach the *out* vertex.
- Due to the semi-monotonicity property of justified/constrained default logics applying new defaults will not contradict previously applied defaults. If the $loc^{just/cons}(P,d) = 1$, then $D_P = P \cap D$ is a partial generating default set and P is a "good" path, which will lead to an optimal solution.

- For classical/rational default logics, which do not satisfy the semi-monotonicity property, $loc^{clas/rat}(P,d)=1$ will not guarantee that P is a "good" path, because the application of new defaults can lead to a contradiction and the defaults from $D_P = P \cap D$ can not be generating defaults.

Definition 10. Let (D,W) be a default theory and $P=(in,...,v_i)$ a path in the default graph $G^{type}(D,W)=(V,A^{type})$. The set of *vertices reachable from* v_i is $R(v_i,P)=\left\{v_j \in V-P | (v_i,v_j)\in A^{type}\right\}$. The *attractivity* of each reachable vertex v_j from $v_i \in P$ is

$$at^{type}(v_i,v_j,P)=\frac{\varphi_{i,j} * loc^{type}(P,v_j)}{\sum_{v_k \in R(v_i,P)} \varphi_{i,k} * loc^{type}(P,v_k)}$$

An ant chooses the next vertex on its path using the probability given by the attractivity function.
The following ACO-based procedure, compute default extensions of any type: classical, justified, constrained, rational.

Procedure Extension-Computation-Problem-ACO

Input data:
(D,W) - a default theory
type - the type of default extension
na - the number of ants in the colony
ni - the maximum number of iterations
e - the evaporation coefficient // e=0.01
k - the number of the best paths used for reinforcement
build $G=(V,A,\varphi)$ // the default graph: $G^{type}(D,W)$;
$it \leftarrow 1$; $sol \leftarrow false$;
while ($it <= ni$ and *not sol*) do
  for $i=1$ to na do
    $P[i] \leftarrow$ **path**$(G, type)$;
    $e[i] \leftarrow$ **eval**$^{type}(P[i])$;
  endfor
  order the arrays $e[i], P[i], i=1,na$ ascending with respect to $e[i]$
  $bestP \leftarrow P[1]$;
  if ($e[1]=\perp$) then
    $sol \leftarrow true$; break;
  endif
  $\varphi \leftarrow$**update**(φ, P, k);
  $\varphi \leftarrow$**evaporation**(φ, e);
  $it \leftarrow it+1$;
endwhile
endprocedure

Function path$(G, type)$
$v \leftarrow in$;
$P \leftarrow in$;
while $(v \neq out)$ do
  compute $R(v, P)$; // the vertices reachable from v
  for all $u \in R(v, P)$ do
    compute $at^{type}(v, u, P)$; // attractivity
  endfor
  choose $w \in R(v, P)$ with the probability $at^{type}(v, w, P)$;
  $P \leftarrow P \cup \{w\}$;
  $v \leftarrow w$;
endwhile
return P;
endfunction

The **evaporation** function acts globally decreasing the pheromone on all the arcs: $\varphi(i, j) \leftarrow (1 - e) * \varphi(i, j)$, $\forall (i, j) \in A^{type}$.

In order to improve the paths in the next iterations, the pheromone on the best k paths is increased by the function **update** as follows:
$\varphi(i, j) \leftarrow \varphi(i, j) + 0.9^{k-m}$, $\forall (i, j) \in D \cap P[m], m = 1, \ldots, k$.

We remark that the **update** function can be improved, for justified / constrained extensions, by reinforcing all the partial paths representing partial generating defaults, due to the semi-monotonicity property.

The fact that the existence of classical/rational extensions for a default theory is not guaranteed implies that the execution of the procedure will stop when the maximum number of iterations was done, but we cannot conclude if there is an extension or not.

For justified/constrained logics, in the worst case there is at least one extension for a default theory, corresponding to the path (in, out), which represents $\emptyset$ as the generating default set.

5. Conclusions

In this paper we proposed a uniform theoretical approach of the extension computation problem for all versions (classical, justified, constrained, rational) of default logic using Ant Colony Optimization. Due to the high complexity of this problem for non-trivial default theories, the ECP is solved as a search problem using this metaheuristic.

We are now working at the implementation of an automated system in order to obtain experimental results for non-trivial general default theories and to find the best combinations for the parameters of the proposed procedure.

A first-order theorem prover [6], based on the sematic tableaux method will be used to check the consistency, inconsistency, derivability and groundness, needed for computing the local and general evaluation functions.

References

[1] G. Antoniou, A. P. Courtney, J. Ernst, J., M. A. Williams, "A System for Computing Constrained Default Logic Extensions", *Logics in Artificial Intelligence, Lecture Notes in Artificial Intelligence*, 1126, pp. 237–250, 1996.

[2] P. Cholewinski, W. Marek, M. Truszczynski, "Default reasoning system DeReS", *Proceedings of KR-96*, pp. 518–528, Morgan Kaufmann,1996.

[3] A. Dorigo, E. Bonabeau, G. Theraulaz, "Ant algorithms and stigmergy", *Future Generation Computer Systems*, no. 16, pp. 851–871, 2000.

[4] G. Gottlob, "Complexity results for nonmonotonic logics", *Journal of Logic and Computation*, vol. 2, no. 3, pp. 397–425, 1992.

[5] W. Lukasiewicz, "Considerations on default logic - an alternative approach", *Computational Intelligence*, no. 4, pp. 1–16, 1988.

[6] M. Lupea, *Nonmonotonic Reasoning Using Default Logics, Ph.D. Thesis*, Babes-Bolyai University, Cluj-Napoca, 2002.

[7] M. Lupea, "Computing default extensions - a heuristic approach", *Studia Universitatis Babes-Bolyai, Informatica*, L, no. 2, pp. 49–58, 2005.

[8] W. Marek, M. Truszczynski, "Normal Form Results for Default Logics", *Non-monotonic and Inductive logic, LNAI*, no. 659, pp. 153–174, Springer Verlag, 1993.

[9] A. Mikitiuk, M. Truszczynsky, "Constrained and Rational Default Logics", *Proceedings of IJCAI-95*, pp. 1509–1515, Morgan Kaufman, 1995.

[10] P. Nicolas, F. Saubion, I. Stephan, "GADEL: a genetic algorithm to compute default logic extensions", *Proceedings of European Conference on Artificial Intelligence*, pp. 484–488, 2000.

[11] P. Nicolas, F. Saubion, I. Stephan, "Genes and Ants for Default Logic", *AAAI Technical Report SS-01-01*, 2001.

[12] P. Nicolas, F. Saubion, I. Stephan, "New generation systems for non-monotonic reasoning", *International Conference on Logic Programming and NonMonotonic Reasoning*, pp. 309–321, 2001.

[13] R. Reiter, "A Logic for Default reasoning", *Artificial Intelligence*, no. 13, pp. 81–132, 1980.

[14] T. H. Schaub, *Considerations on Default Logics, Ph.D.Thesis*, Technischen Hochschule Darmstadt, Germany, 1992.

[15] T. H. Schaub, "XRay system: An implementation platform for local query-answering in default logics", *Applications of Uncertainty Formalisms, Lecture Notes in Computer Science*, no. 455, pp. 254–378, Springer Verlag, 1998.

[16] C. Schwind, "A tableaux-based theorem prover for a decidable subset of default logic", *Proceedings of the Conference on Automated Deduction*, Springer Verlag, 1990.

Babeş-Bolyai University, Faculty of Mathematics and Computer Science, Cluj-Napoca, Romania

E-mail address: lupea@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

GENERALIZED FORMAL DEFINITION OF CONFLICT DETECTION AND RESOLUTION

CIPRIAN COSTA

Abstract. In a distributed environment where information that is not read-only is shared between multiple parts of the system, the problem of conflict detection and resolution has to be addressed. In this paper we discuss the issue of generalized conflict definition because this main aspect of the consistency of any distributed system is often treated on a case by case basis. Based on the formal definition we can move on and identify what are the main types of conflict resolution algorithms which is very important for an accurate description of the guarantees offered by a distributed system.

1. Introduction

A lot of effort was channelled into making sure that the conflicts do not appear in a distributed environment mainly because a system that is fully consistent is a simple abstraction and we can reuse many of the concepts that were developed and proven in non-distributed systems. Several types of distributed systems have abandoned this line of thought and moved towards more complicated abstractions, where conflicts are allowed to exist. A simple example are classical desktop database front end applications that have moved from connected environments using pessimistic concurrency to disconnected environments using optimistic concurrency techniques in order to manage the relatively rare but possible conflicts. Why and when these types of systems are created and how to decide which to use is not in the scope of this paper. However, since they gained a lot of traction in recent times, we think that the notion of conflict detection and resolution deserves a unified and formal definition that can be used to better understand these distributed environments.

It is proven by the CAP theorem that availability, consistency and partition tolerance can not be achieved at the same time in a distributed system([2]). Consistency is a desirable property of any system since it makes programming and reasoning on the results of that system simpler, practically eliminating

Received by the editors: December 1, 2009.

2010 *Mathematics Subject Classification.* 68P15, 68M14.

1998 *CR Categories and Descriptors.* H.2.4 [**Systems**]: Subtopic – *Distributed databases*; *Key words and phrases.* distributed systems, conflict.

uncertainty. In order to achieve consistency in a distributed system, a majority of the parts need to agree on every operation that affects the common state. In order to achieve that, a lot of consensus algorithms like Paxos ([6]) were created and found their way into various real live implementations like the Chubby system at Google ([3]).

The problem with consistency is that, according to the CAP theorem, we need to give up on availability or partition tolerance. Since in many distributed systems of large scale such requirements are non-negotiable because of the impact they would have on the perceived quality of the system, new approaches in which conflicts are allowed to temporarily exist in the system have been proposed ([10]).

Because the system is no longer consistent, it is much more difficult to build something on top of it (one may never be sure that the values returned by a part of the system are actually correct), it is important to define what type of guarantees can the system provide. We argue that an important part of this is generated by the way conflicts are detected and resolved. Throughout this paper we analyse existing work in the domain of conflict detection and resolution and propose a generalized conflict detection definition and some criteria that can be used for conflict resolution categorization.

2. Conflict definition

Informally, we define conflicts as situations in which one or more parts of a system have different representations of a shared fact.

Significant research on conflicts has been done in the field of autonomous agents, specifically addressing the problem of distributed constraint satisfaction DCSP ([11]). A constraint satisfaction problem is defined as finding an assignment to a set of variables with the property that a set of given predicates are satisfied by the said assignment. A DCSP is a CSP (constraint satisfaction problem) where the variables and the predicates are distributed among several independent agents and gathering all the information required for solving the CSP on a single node is impossible for various reasons (availability requirements, software incompatibility, etc). In this case, a conflict is defined as an assignment chosen by one of the agents that is valid under all the local predicates but does not hold in at least one of the other predicates.

Another field in which conflicts and conflict resolution algorithms were researched is collaborative editing ([8], [1], [5]). Several aspects related to conflicts and conflict resolution appear in collaborative editing that are not present in DCSPs:

- *Intent* - In the case of distributed agents the software that runs on each agent is well understood and the intent of any operation is always known. In the case of collaborative editing the intent of the user has to be inferred from their actions, it is not a priori known .

- *Operational transformations* - certain conflicts can be transformed in ways that allow them to be applied to the same document and preserve the operations of both users. This situation occurs when a user inserts a character at position 2 while another will insert a character at position 6. If we transform the operation of the second in an insert on position 7, both users can save their edits. This conflict is called a *non-exclusive conflict*. Extensive studies of operational transformations in various scenarios can be found in Sun and all 1998 [7] and Sun and all 2004 [9].

A definition that is often used in these scenarios is based on causality relations because they are supposed to capture the intent of the user. The supposition is that everything the user knows about a document is the cause of the intent, and any change in the cause could possibly alter the intent. From this, if two users are performing an operation having different knowledge of the environment, their actions could be in conflict. Informally, we say that $o_1 \rightarrow o_2$ (o_1 causally precedes o_2) if the user performing o_2 was aware of o_1 before performing o_2. A conflict in this case is defined as a pair of operations with the property that $o_1 \nrightarrow o_2$ and $o_2 \nrightarrow o_1$.

We consider this to be a pessimistic intent preservation. The problem with it is that, in some systems, it could lead to a lot of false positives and the main assumption of these distributed systems is that conflicts are rare and far between. We argue that we need a more generic definition of conflict that would allow a finer grained control over what constitutes intent, what part of the existing state is relevant for an operation and what alternatives are available to merge the two operations thus avoiding a conflict that might lead to an expensive negotiation in order to be resolved.

3. Generalized conflict definition

Let us assume that $x_1 \in D_1$, $x_2 \in D_2$, ..., $x_n \in D_n$ is a set of facts shared between multiple parts of the system. We define an operation as being the tuple

$$o_k = (x_{k_1}, ..., x_{k_p}, state_k, alt_k)$$

where k is the system that executed the operation, x_{k_1}, ... x_{k_p} are assignment on a subset of shared facts, *state* is a representation of the facts and other information that captures the relevant context of the operation, thus defining the intent of the user performing the operation, *alt* is a function defined as $alt : O \times O \rightarrow O \cup \{\oslash\}$ where O is a set of all the possible operations. The function *alt* represents the means of merging two operations into a third with the property that the third is not in conflict with either of the first two. We say that two operations can not be merged if $alt(o_1, o_2) = \oslash$.

In order to define conflicts based on causality relations we can consider *state* to be some sort of vector clocks that summarize the knowledge of the

user when attending the operation, while *alt* would be undefined. But in this definition we have the freedom to alternate *state* from user generated intent description to artificial intelligence algorithms, depending on how the system deals with conflicts and what sort of conflict tolerance it has. The complexity of the conflict definition reflects directly in the guarantees the system is able to make about consistency and other observable metrics, therefore most of the systems will probably stay on the safe side, but still prefer to relax causal dependency.

Since, unlike in the case of the DCSP, we are not trying to solve a set of constraint that may not all fit on an agent but rather satisfy the single constraint of consistency, we can use specific predicates defined on the data existing on each node and define conflicts based on the evaluation of these predicates. We consider *consistency* to be a predicate defined on $O \times O$ that is evaluated to *true* if the two operations can be applied in parallel and preserve the intent of both. We define the conflict between two operations and write $o_1 \otimes o_2$ if

$$\neg consistency(o_1, o_2) \wedge alt(o_1, o_2) = \oslash$$

.

As it can be seen from the definition, there are simplified implementation of the predicates and functions that will lead to all the conflict definitions that were described earlier. Also, the definition allows us to further define and formalize the conflict resolution algorithms.

4. Conflict resolution types

Once a conflict is detected, it must be resolved in order for the system as a whole to function properly. Most of the systems that allow conflicts to appear between parts have a certain built in tolerance for conflicts, but that usually degrades the quality of the service or renders it useless. Since the degradation is acceptable because the cost of avoiding it altogether is prohibitive, the system must ensure that the conflicts are resolved as quick as possible, or, at the very least, ensure that the conflicting situation will not exist forever in the condition of normal functioning of the system.

We identify the following characteristics of any conflict resolution algorithm that can be used in order to classify and better understand the consequences of using them:

- *Convergence* - The guarantee that the system will converge from a conflict state to a conflict free state in a bounded interval of time. It is very difficult to work with a system that does not guarantee convergence in at least some cases. While such systems where perpetual conflict situations are accepted could exist, usually there are guarantees like "system converges if it is partition free, all the components are up and no additional conflicts are added". It is very important for a conflict

resolution algorithm to clearly state the conditions under which it will converge, since convergence under any conditions is impossible.

- *Performance* - One of the reasons to implement conflict tolerance in the first place is to decrease the cost of the system, therefore performance is an important characteristic. The performance level can be measured in many ways and the relevance of these metrics depend on the specific application. Possible metrics include the time, the network resources that are used, the number and type of resources that are involved (for example an algorithm that requires a human to make decisions will probably be more expensive than one that relies entirely on computers).
- *Impact on non-conflicting state* - Most of these systems accept conflicts because they are supposed to be rare, and an optimistic approach to resolving them is applicable. However, if the enforcement of the optimistic concurrency will affect all operations, including the majority that will be conflict free, the price might be prohibitive. One might argue that this is just another possible type of performance metric, but we would rather consider it as a separate category because performance is strictly bound to the process of resolving a conflict situation.
- *Intention preservation* - This characteristic takes into consideration the source of the conflict. Usually conflicts are not introduced by errors in the system but by users or systems working independently and having only a partial knowledge of the environment. In such cases, it is important to determine how and to what degree is the intent of each state that is in conflict being preserved by the conflict resolution algorithm. For example, if a document contains a circle, a user can alter the document to become a "happy face" while another could change the circle to a square. The result of having a square "happy face" will not preserve the intent of either user.

5. Conclusions and future work

In this paper we unified approaches from various communities involved in the research of distributed systems and created a generic definition of what a conflict is. Another contribution is the enumeration of several aspects that are important in the conflict resolution phase. As a system can not remain in a conflict state forever, all systems need to address the issue of conflict resolution, therefore, identifying the characteristics of a conflict resolution algorithm is of utmost importance.

We have used the conflict detection and resolution analysis from this paper in order to implement the JStabilizer ([4]) system and model the object oriented framework so that it accommodates a wide range of possible usage

patterns. We will continue to develop the JStabilizer framework and implement more test cases that will refine and reinforce the validity and generality of the definitions and concepts included in this paper.

References

[1] Pauline M. Berry, Tomás Uribe, Neil Yorke-Smith, Cory Albright, Emma Bowring, Ken Conley, Kenneth Nitz, Jonathan P. Pearce, Bart Peintner, Shahin Saadati, and Milind Tambe. Conflict negotiation among personal calendar agents. *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems - AAMAS '06*, page 1467, 2006.

[2] E.A. Brewer. Towards robust distributed systems. *Proceedings of the Annual ACM Symposium on Principles of Distributed Computing*, 19:710, 2000.

[3] T.D. Chandra, R. Griesemer, and J. Redstone. Paxos made live: an engineering perspective. *Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing*, page 407, 2007.

[4] Costa Ciprian. JStabilizer code repository (http://code.google.com/p/jstabilizer/), 2009.

[5] S. Citro, J. McGovern, and C. Ryan. Conflict management for real-time collaborative editing in mobile replicated architectures. *Proceedings of the thirtieth Australasian conference on Computer science-Volume 62*, page 124, 2007.

[6] Leslie Lamport. The part-time parliament. *ACM Transactions on Computer Systems (TOCS)*, 16:133–169, 1998.

[7] C Sun, X Jia, Y Zhang, Y Yang, and D Chen. Achieving convergence, causality preservation, and intention preservation in real-time cooperative editing systems. *ACM Transactions on Computer Human Interaction*, 5:63–108, 1998.

[8] Chengzheng Sun and David Chen. Consistency maintenance in real-time collaborative graphics editing systems. *Interactions*, 9:1–41, May 2002.

[9] D. Sun, S. Xia, C. Sun, and D. Chen. Operational transformation for collaborative word processing. *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, 6:446, 2004.

[10] Werner Vogels. Eventually consistent. *ACM Queue Communications*, 2008.

[11] M. Yokoo, E.H. Durfee, T. Ishida, and K. Kuwabara. The distributed constraint satisfaction problem: Formalization and algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 10:673685, 1998.

Babes-Bolyai University, Department of Computer Science, Cluj-Napoca, Romania

E-mail address: costa@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

PROPOSAL OF A SET OF OCL WFRS FOR THE ECORE META-METAMODEL

VLADIELA PETRAŞCU, DAN CHIOREAN, AND DRAGOŞ PETRAŞCU

Abstract. Specifying a complete set of OCL Well Formedness Rules (WFRs) is essential for the well-definedness of any (meta)modeling language's abstract syntax. Within this paper, we report on the definition of a set of OCL WFRs for the Ecore meta-metamodel. We use some relevant WFRs for the Ecore generics accompanied by meaningful test cases, in order to illustrate our proposal and its advantages over related approaches.

1. Introduction

The highest level of abstraction in any model-driven approach, be it Model Driven Architecture (MDA) [5], Model Driven Engineering (MDE) [14], or Language Driven Development (LDD) [8], is represented by the metamodeling language - the language used for defining all modeling languages, itself included. This is level M3 of the classical four-level modeling architecture promoted by the Object Management Group (OMG). It is well known that a complete definition of any language should include formal representations of its *abstract syntax*, *concrete syntax*, and *semantics*; for modeling languages, the abstract syntax is generally given in terms of a metamodel (which describes the concepts used by the language and their relationships), which should be accompanied by appropriate Well Formedness Rules (WFRs, that further constrain the legal instantiations of metamodel concepts). It follows that the abstract syntax of a metamodeling language should be defined by means of its meta-metamodel and associated WFRs. OMG's Meta Object Facility (MOF) [3], Eclipse Modeling Framework's (EMF's) Ecore [15], and eXecutable Metamodelling Facility's (XMF's) XCore [8] are probably the best known meta-metamodels today.

Received by the editors: November 1, 2009.

2010 *Mathematics Subject Classification.* 68N30.

1998 *CR Categories and Descriptors.* D.2.4 [**SOFTWARE ENGINEERING**]: Software/Program Verification – *Programming by contract, Class invariants, Validation*; D.2.11 [**SOFTWARE ENGINEERING**]: Software Architectures – *Languages (e.g., description, interconnection, definition)* .

Key words and phrases. Model Driven Engineering (MDE), meta-metamodel, Object Constraint Language (OCL), Well Formedness Rules (WFRs), Ecore.

Having a complete set of metamodel WFRs is of utmost importance for the well-definedness of any modeling language. This is due to the fact that the graphical formalism used to represent the metamodel itself (that of class diagrams) is not powerful enough so as to capture all the constraints that govern the individual meta-concepts, as well as their inter-relationships. Even more, these WFRs should be formalized, in order to allow the existing tools to validate the models against them; a model is considered to be valid/correct if and only if it conforms to both its metamodel and associated WFRs. The language used in order to formalize the WFRs should be the Object Constraint Language (OCL) [7], for the following three reasons at least. First of all, OCL is the standard formalism. Second, defining the WFRs as OCL invariants is preferred to implementing them directly in the repository code, since the OCL expressions are more compact and intelligible compared to their equivalents in a programming language. Third, today we can benefit from a powerful tool support, regarding both the evaluation of OCL constraints on snapshots and their automatic translation into code. The Object Constraint Language Environment (OCLE) [6] and EMF [1] with Model Development Tools (MDT)-OCL [4] are notable examples in this respect.

The arguments above are even stronger when it comes to metamodeling languages. Their abstract syntax is used in defining the metamodels of all possible modeling languages. There has to be possible to check/ensure the correctness of all these metamodels which are to be reused by instantiation in thousands of modeling applications.

Still, a study that we have carried out on the three above mentioned meta-metamodels, MOF, Ecore, and XCore, has revealed that the goal of having a correct and complete set of OCL WFRs for each of them is far from being reached. The closest to this aim is Ecore, whose repository code contains a set of WFRs implemented directly in Java. In case of the OMG MOF standard, a great number of OCL specifications used in describing the Core UML Infrastructure (which is part of MOF) are wrong. As for XCore, it only contains two such WFRs, specified using the OCL-like language XOCL.

Given this state of facts, our overall aim has been to define a complete set of OCL WFRs for each of the three meta-metamodels, as well as to test and validate them on relevant metamodel examples (finding appropriate test models even before or simultaneously with defining the OCL constraints - test driven (meta(-meta))modeling - is highly important). Within this context, the current paper reports on the definition and validation of such a complete set of OCL WFRs for Ecore.

The rest of the paper is organized as follows. Section 2 reviews the state of facts regarding the Ecore WFRs, which provides the context and motivation of our work. Section 3 testifies our contribution, using some relevant WFR for

the Ecore generics. Related work is summarized in Section 4. We conclude the paper in Section 5, giving some hints on future work.

2. Ecore WFRs. State of Facts

Compared to the other meta-metamodels that we have studied, namely MOF and XCore, Ecore has a special status, that may be described by the following:

- Ecore is, beyond any doubt, the best known EMOF (Essential MOF) implementation. However, Ecore does not match EMOF exactly. On the one side, the approach taken with Ecore is more pragmatic and implementation-oriented. On the other side, starting with EMF 2.3, Ecore includes constructs for modeling with generics [11]; this is considered to be a departure from EMOF, which does not currently provide such support.
- Due to the framework it ships with (EMF), Ecore is definitely the most tested meta-metamodel.
- The Ecore repository includes a set of WFRs that allow validating the metamodels that instantiate it. These rules are implemented within the EcoreValidator class. However, although the code does contain comments, these do not reflect all implementation decisions. Especially in case of those rules used to check the correctness of parameterized types, the code complexity is increased and the lack of detailed comments and examples is disturbing. The fact that, as stated in [11], "The design of Ecore's support for generics closely mirrors that of Java itself" is expected to help in this respect. Still, the tests that we have run have shown that there are differences among the two, regarding both the declaration of generic types and their correct instantiation.
- The Ecore implementation witnesses the fact that the value of meta-metamodel level WFRs has been acknowledged. However, even though EMF integrates an OCL plugin (MDT-OCL), we have not found any OCL equivalent of the implemented constraints.
- The paper [9] proposes some OCL WFRs that may be used in validating the Ecore generics; this is actually the only paper concerning the OCL formalization of Ecore WFRs that we have found in the literature. However, even though they are a good starting point and comparison base, the OCL specifications described there are far from complete and not entirely correct.

Within this context, our goal has been to identify, classify and specify in OCL a complete set of WFRs for Ecore, complete at least with respect to the rules already implemented in EMF. We report on accomplishing this goal in the following sections using some relevant constraints concerning generics.

The OCL specifications are preceded by their informal equivalents and are accompanied by relevant test cases used for their validation.

3. A Complete Set of OCL WFRs for Ecore

Taking the EMF implementation as a reference, we have defined a complete set of OCL WFRs for Ecore, which we have tested and validated on relevant examples using OCLE. The entire set can be found at [2]. Within this section, we detail on one such WFR, regarding the Ecore generics. Choosing this particular constraint for exemplification purposes is due to both its complexity level (since it is a non-trivial WFR) and the fact that it allows a close comparison with related work described in [9].

3.1. **Generics in Ecore.** Figure 1 shows that part of the Ecore metametamodel that ensures the generic modeling support it provides. As previously mentioned, this has been introduced starting with EMF 2.3, the newly added concepts being `ETypeParameter` and `EGenericType`. We briefly explain and exemplify these concepts in the following.

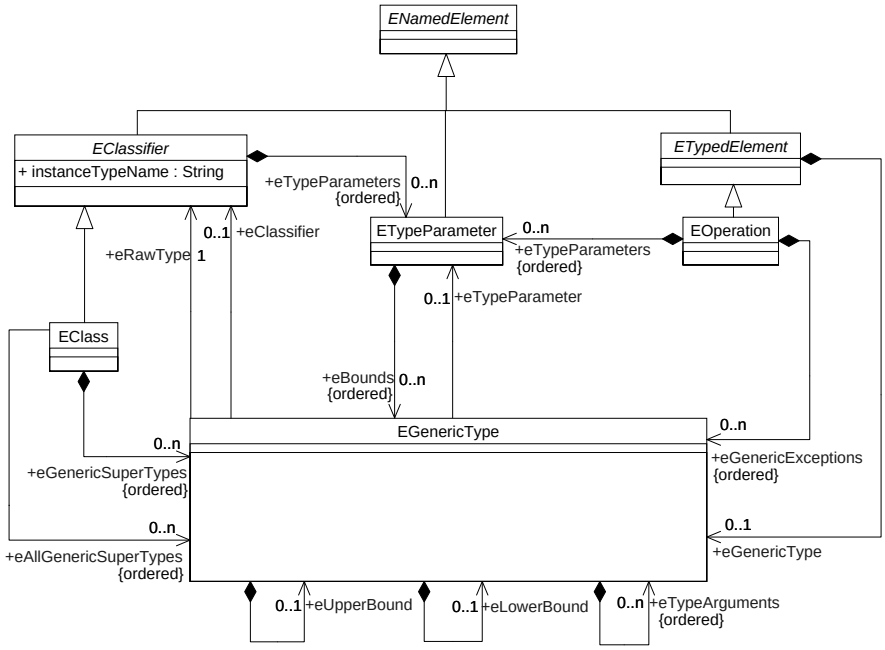


Figure 1. Ecore generics

Similar to Java, Ecore supports generic type and operation declarations, as well as generic type instantiations (also known as parameterized types).

An ETypeParameter instance stands for a type parameter used by either a generic classifier or a generic operation declaration. That particular ETypeParameter is contained by its corresponding EClassifier or EOperation instance. This is denoted by the composition relationships EClassifier - ETypeParameter and EOperation - ETypeParameter from Figure 1, which are mutually exclusive (xor constraint). As an example, the Java generic type declaration interface Collection<T> would be modeled in Ecore by means of an EClass instance named Collection, having its interface attribute set to true, and whose eTypeParameters sequence contains a single ETypeParameter instance, named T.

Type parameters may have bounds, as indicated by the composition relationship between ETypeParameter and EGenericType. For a Java type declaration such as

```
1 class OrderedList<T extends Comparable<T>> { ... }
```

the eBounds sequence owned by the type parameter T contains a single EGenericType instance, namely Comparable<T>.

An EGenericType instance may denote one of the following: a type parameter reference, a (generic) type invocation, or a wildcard. This is reflected by its associations to ETypeParameter and EClassifier, respectively. The two associations are mutually exclusive; there is a WFR specifying that an EGenericType instance cannot be simultaneously associated to both an eTypeParameter and an eClassifier. In case it has an eTypeParameter, then it is a type parameter reference, if it has an eClassifier, then it is a (generic) type invocation, and when both are missing, it is a wildcard. An EGenericType instance denoting a generic type invocation may specify type arguments (see the eTypeArguments role name); in case it does not specify any type arguments, then it is used as a raw type, the reason being that of ensuring compatibility with the previous, non-generic EMF releases. Wildcards may specify a lower or an upper bound (see the corresponding unary compositions of EGenericType). To exemplify all these, let us consider the following Java interface definition:

```
2 interface List<T> extends Collection<T>
3 {
4   boolean add(T elem);
5   boolean addAll(Collection<? extends T> col);
6   ...
7 }
```

The listing above contains a generic type declaration for List. In the equivalent Ecore model (Figure 2), this would be modeled by means of an EClass instance named List, which contains an ETypeParameter instance with the

name `T`. The newly declared type specifies a generic supertype, `Collection<T>`. The latter is modeled using an `EGenericType` instance that corresponds to a generic type invocation with a type argument; the referred classifier is `Collection` and the contained type argument `T`. At its turn, this type argument is an `EGenericType` instance that corresponds to a type parameter reference, the referenced type parameter being the `ETypeParameter` instance `T`. The fourth line of the listing contains another `EGenericType` instance that corresponds to a type parameter reference, only this time it is used not as a type argument, but as the type of the `EParameter` instance `elem`. The type of `col` in line 5 denotes a generic type invocation; the referenced classifier is again `Collection` and the type argument is `? extends T`. The latter is an `EGenericType` instance that corresponds to an upper bounded wildcard; it has no `eClassifier` or `eTypeParameter` and it specifies an `eUpperBound`, namely `T`.

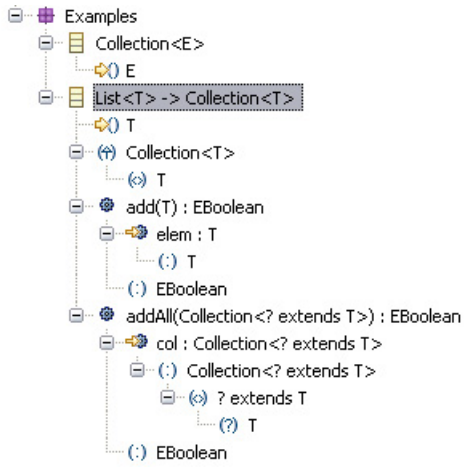


FIGURE 2. Ecore model for `List<T>`. EMF tree-editor screenshot

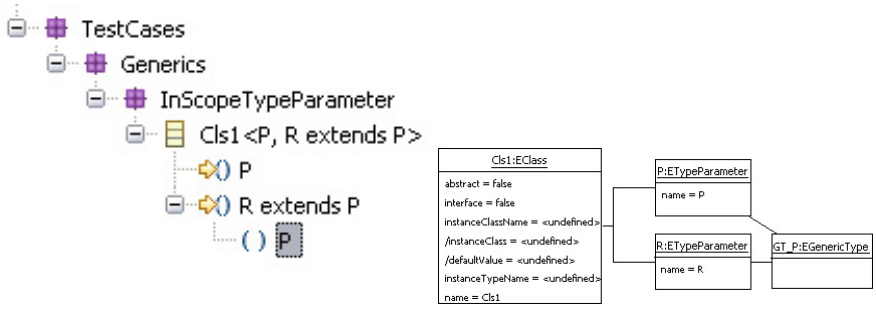
`EGenericType` instances can play various roles in an Ecore model, each kind of usage being constrained by suitable WFRs. Such an instance can be exactly one of the following:

1. A generic supertype of a class, as shown by the composition relationship `EClass-EGenericType`; `Collection<T>` in line 2 of the listing above is such an example;
2. The type of a typed element (attribute, reference, operation, parameter), as shown by the composition relationship between `ETypedElement` and `EGenericType`; an example is `T` in line 4 above;
3. A bound of a type parameter, as shown by the composition relationship `ETypeParameter-EGenericType`; `Comparable<T>` in line 1 above is such an example;
4. One of the type arguments of a generic type invocation, fact denoted by the unary composition relationship of `EGenericType` owning the `eTypeArguments` role; `T` from `Collection<T>` in line 2 above is a good example;
5. The upper or lower bound of a wildcard, as shown by the other two unary compositions of `EGenericType`; `T` from `Collection<? extends T>` in line 5 above is an example of an upper bound usage of a generic type;

6. An exception type, fact denoted by the composition between `EOperation` and `EGenericType`.

3.2. **A WFR for Generics in Ecore.** Equipped with this knowledge regarding the Ecore generics, we seek to provide an OCL specification for the following informal WFR: "Assuming that a generic type denotes a type parameter reference, the referenced type parameter must be in scope and must not be a forward reference. The type parameter is in scope if its container is an ancestor of this generic type within the corresponding Ecore containment tree". We give a few examples in the following, in order to ensure a deeper understanding of the rule and to set up some test cases for its validation. Assuming a closer familiarity of the reader with Java than Ecore, we start with the Java equivalent of each chosen example, followed by OCLE and EMF snapshots for the corresponding Ecore model.

As a first example, let us consider the Java declaration `class Cls1<P, R extends P>`. The generic type declaration for `Cls1` uses `P` and `R` as type parameters, the latter being upper bounded by the former. This is a valid generic declaration since the referenced type parameter `P` is in scope and is not a forward reference (`P` being declared prior to `R`). The equivalent Ecore model consists of an `EClass` instance named `Cls1` which contains two `ETypeParameter` instances named `P` and `R` (Figure 3 shows the corresponding EMF and OCLE snapshots). The type parameter `R` has a bound, which is an `EGenericType` instance that references the type parameter `T`. The OCLE snapshot shows explicitly the `EGenericType` instance used as a bound (`GT_P`) and its link to the referenced type parameter. Within the EMF tree, the bound appears as a direct descendent of the type parameter it bounds, being labeled with the name of the referenced type parameter.



(A) EMF snapshot (B) OCLE snapshot

FIGURE 3. Ecore model for Example 1: `Cls1<P, R extends P>`

For the second example, consider the Java declarations `class Cls2<Q>` and `class Cls3<S, T extends Q>`. The second one is obviously not valid, since the bound of `T` references type parameter `Q`, which is out of scope. The corresponding OCLE snapshot is shown in Figure 4; its EMF equivalent is missing since the framework constrains a referenced type parameter to be chosen from the list of those in scope. Therefore, it is impossible to model such a case using the EMF tree-like editor. However, this erroneous situation could still occur if the model were loaded from an XMI file instead of being created directly with the editor.

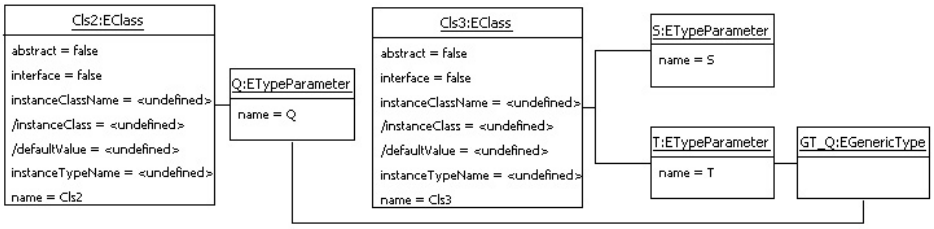
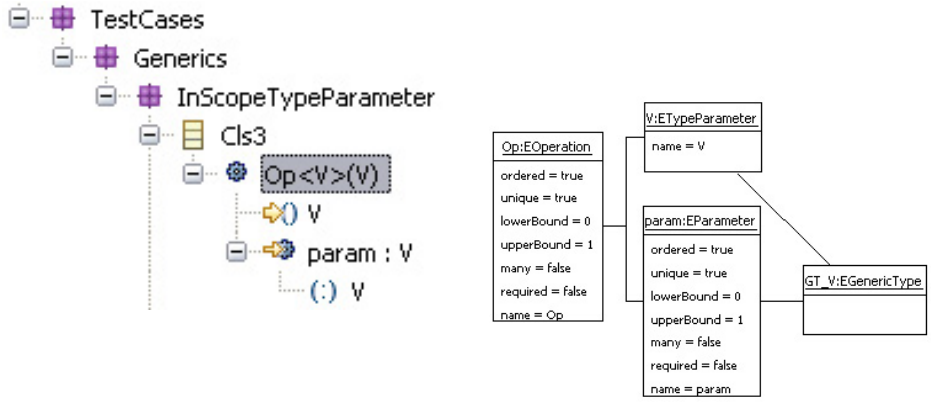


FIGURE 4. OCLE snaphot for Example 2: `Cls2<Q>, Cls3<S, T extends Q>`

In both examples described above, the container of each type parameter has been a classifier. Let us now consider the following Java generic operation declaration `<V> void Op(V param)`. The equivalent Ecore model has at its root an `EOperation` instance, `Op`, whose `eTypeParameters` sequence contains only the type parameter `V`. `Op` owns a single parameter, `param`, whose type is an `EGenericType` instance that references the type parameter `V`. The EMF and OCLE snapshots are illustrated in Figure 5. This is again a valid model with respect to the WFR under consideration.

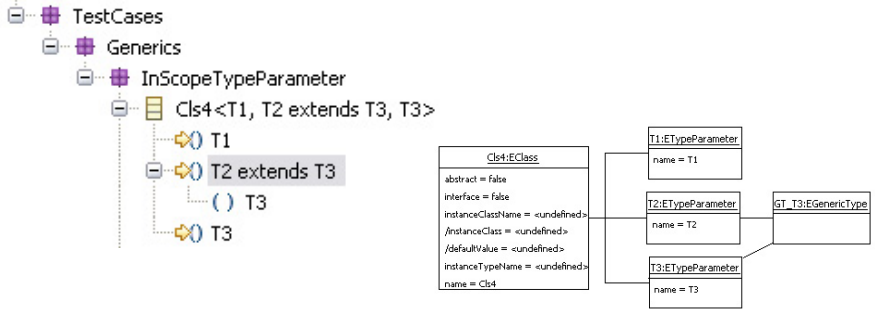
The fourth and last example we take is again a generic class declaration, of the form `class Cls4<T1, T2 extends T3, T3>`. Such a declaration is not valid, since the bound of `T2` performs a forward referencing of the type parameter `T3`. The equivalent snapshots are given in Figure 6.

The constraints in Listing 1 formalize the WFR stated at the beginning of this subsection. The OCL specification has been splitted in two invariants defined for the `EGenericType` context, namely `InScopeTypeParameter` and `NotForwardReference`; as their names indicate, the former enforces the type parameter referenced by a generic type to be in scope, while the latter checks for forward referencing. As in programming, the splitting of large constraints into smaller pieces is a good modeling practice. This way, the constraints become easier to write and their comprehensibility is enhanced. Even more, this also provides valuable support in localizing exactly and in real time the cause of a constraint violation during model checking activities.



(A) EMF snapshot (B) OCLE snapshot

FIGURE 5. Ecore model for Example 3: `<V> void Op(V param)`



(A) EMF snapshot (B) OCLE snapshot

FIGURE 6. Ecore model for Example 4: `Cls4<T1, T2 extends T3, T3>`

```
1  context EGenericType
2   -- The referenced type parameter must be in scope, i.e.,
3   -- its container must be an ancestor of this generic type ...
4   inv InScopeTypeParameter:
5    self.isTypeParameterReference() implies
6    self.ancestors()->includes(self.eTypeParameter.eContainer())

8  context EGenericType
9   -- ... and must not be a forward reference.
10  inv NotForwardReference:
11   (self.isTypeParameterReference() and self.isUsedInATypeParameterBound())
12   implies
```

```
13   (let refParameter : ETypeParameter = self.eTypeParameter
14    let boundedParameter : ETypeParameter = self.boundedTypeParameter()
15    let paramSeq:Sequence(ETypeParameter)=
16      (if refParameter.eContainer().oclIsKindOf(EClassifier)
17       then refParameter.eContainer().oclAsType(EClassifier).eTypeParameters
18       else refParameter.eContainer().oclAsType(EOperation).eTypeParameters
19       endif)
20    let posRefParameter : Integer = paramSeq->indexOf(refParameter)
21    let posBoundedParameter : Integer =
22      (if paramSeq->includes(boundedParameter)
23       then paramSeq->indexOf(boundedParameter)
24       else -1
25       endif)
26    in
27    ( (posBoundedParameter <> -1) implies
28      ( (posRefParameter < posBoundedParameter) or
29        ( (posRefParameter = posBoundedParameter) and
30          (not boundedParameter.eBounds->includes(self))
31        )
32      )
33    )
34   )
```

LISTING 1. `EGenericType` invariants prohibiting invalid type parameter references

We do not insist on the OCL specification for `InScopeTypeParameter` (lines 1 to 6 of Listing 1), since it carefully matches the comments it goes along with. However, we detail the three query operations it makes use of, namely `isTypeParameterReference()`, `eContainer()` and `ancestors()`. Their OCL definitions are provided in Listing 2.

The core query here is `ancestors()`, which should compute all parents of an arbitrary object from within the Ecore containment tree to which the object belongs. The returned set should include the object's direct container, the direct container of the latter, and so on. In case the particular object is the root of the tree, then the empty set is returned. In order to provide its intended functionality, `ancestors()` makes use of `eContainer()`, which returns the direct container of an arbitrary object.

The `eContainer()` operation is given a default definition for the root of the Ecore modeling hierarchy, `EObject`, which is overriden in all its descendants, according to the composition relationships they are involved in (see Figure 7). The default implementation returns `Undefined(EObject)` (Listing 2, lines 12-13) and most of the overridings simply perform a one-step navigation of a composition relationship (Listing 2, lines 52-65). However, the composition relationships which involve `ETypeParameter` and `EGenericType` are uni-directional in the Ecore model, therefore the OCL expressions for

`eContainer()` in these two particular cases (Listing 2, lines 15-50) are more complex and unefficient, due to the calls to `allInstances()`.

```
1  context EGenericType
2   def: isTypeParameterReference() : Boolean =
3         not self.eTypeParameter.isUndefined()

5  context EObject
6   def: ancestors() : Set(EObject) =
7        let empty : Set(EObject) = Set{} in
8        if self.eContainer().isUndefined() then empty
9        else Set{self.eContainer()}->union(self.eContainer().ancestors())
10       endif

12 context EObject
13  def: eContainer() : EObject = oclUndefined(EObject)

15 context EGenericType
16  def: eContainer() : EObject =
17   let cls=EClass.allInstances->any(c|c.eGenericSuperTypes->includes(self))
18   let param=ETypeParameter.allInstances()->any(p|p.eBounds->includes(self))
19   let te=ETypedElement.allInstances()->any(t|t.eGenericType=self)
20   let gt1=EGenericType.allInstances()->any(g|
21            g.eTypeArguments->includes(self))
22   let gt2=EGenericType.allInstances()->any(g|g.eLowerBound=self)
23   let gt3=EGenericType.allInstances()->any(g |g.eUpperBound=self)
24   let op=EOperation.allInstances()->any(o|
25           o.eGenericExceptions->includes(self))
26   in
27   (if not cls.isUndefined() then cls
28    else if not param.isUndefined() then param
29         else if not te.isUndefined() then te
30              else if not gt1.isUndefined() then gt1
31                   else if not gt2.isUndefined() then gt2
32                        else if not gt3.isUndefined() then gt3
33                             else if not op.isUndefined() then op
34                                  else oclUndefined(EObject)
35                                  endif
36                             endif
37                        endif
38                   endif
39              endif
40         endif
41    endif)

43 context ETypeParameter
44  def: eContainer() : EObject =
45   let classifier = EClassifier.allInstances()->any(c |
46                      c.eTypeParameters->includes(self))
47   in
48   (if not classifier.isUndefined() then classifier
49    else EOperation.allInstances()->any(o | o.eTypeParameters->includes(self))
50    endif)
```

```
52 context EPackage
53  def: eContainer() : EObject = self.eSuperPackage

55 context EClassifier
56  def: eContainer() : EObject = self.ePackage

58 context EStructuralFeature
59  def: eContainer() : EObject = self.eContainingClass

61 context EOperation
62  def: eContainer() : EObject = self.eContainingClass

64 context EParameter
65  def: eContainer() : EObject = self.eOperation
```

LISTING 2. Query operations used by `InScopeTypeParameter`

`ETypeParameter`, for instance, is involved in two composition relationships (with `EClassifier` and `EOperation`, see Figure 1 for reference), both of which are unidirectional, navigable only from container to part (we did not manage to find the rationale for this design decision). Therefore, the direct container of a type parameter is always either a classifier or an operation. However, this container cannot be accessed through a simple navigation. The only way to identify it involves searching that particular type parameter within the `eTypeParameters` collections of all classifiers and operations that belong to the current model (which explains the use of `allInstances()` in lines 45, 49 of Listing 2). That operation or classifier which includes the searched type parameter within its `eTypeParameters` collection is its direct container. There will definitely be at most one such classifier or operation, since the considered relationships are compositions. Therefore, the use of the undeterministic `any()` in lines 45, 49 of Listing 2 is completely safe; it should produce the same result no matter the tool used. The overriding of `eContainer()` for `EGenericType` can be justified in a similar manner, only this time the number of composition relationships involved, therefore the complexity, is greater.

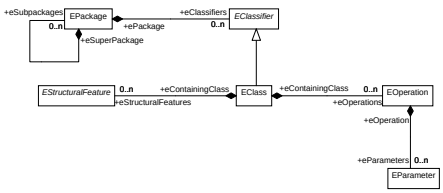


FIGURE 7. Ecore containment relationships

The evaluation of `InScopeTypeParameter` using OCLE has ended successfully for the first, third and fourth of the test examples considered above,

while failing for the second, in accordance with the results given by the EMF EcoreValidator and the Java compiler. In case of the first test example, the constraint is evaluated for the `EGenericType` instance `GT_P`, which references the `ETypeParameter` instance `P`. The computed set of `ancestors()` of `GT_P` contains the `ETypeParameter` instance `R` (its direct container, of which it is a bound) and the `EClass` instance `Cls1` (the direct container of `R`). Also, the direct container (`eContainer()`) of `P` is `Cls1`. Since the latter belongs to the `ancestors()` set, the constraint evaluates to true on `GT_P`. The evaluation results for the second and fourth examples (`false` and `true`, respectively) can be explained in a similar way. In case of the third one, the invariant is evaluated on the `EGenericType` instance `GT_V`, which references type parameter `V`. The `ancestors()` of `GT_V` are its direct container, `param`, (`GT_V` being the `eGenericType` of `param`) and the `EOperation` instance `Op` (the direct container of `param`); the `eContainer()` of `V` is `Op`, therefore the required inclusion takes place, which ends successfully the evaluation.

In order to simplify the reading, we will use the phrase "generic type" instead of "`EGenericType` instance" from here on.

The invariant that completes the proposed WFR's OCL definition, `NotForwardReference`, rules out all generic types which reference type parameters that are in scope, but are declared afterwards. This situation can only occur when the generic type is contained in a type parameter bound, either of a classifier or of an operation. The employed query operations, specifically `isUsedInATypeParameterBound()` and `boundedTypeParameter()`, are defined in Listing 3.

```
1  context EGenericType
2   def: isUsedInATypeParameterBound() : Boolean =
3    -- checks whether self is involved in defining a type parameter bound
4    self.ancestors()->exists(o | o.oclIsTypeOf(ETypeParameter))

6  context EGenericType
7   def: boundedTypeParameter() : ETypeParameter =
8    -- returns the type parameter in whose bound self is used
9    self.ancestors()->any (o |
10     o.oclIsTypeOf(ETypeParameter)).oclAsType(ETypeParameter)
```

LISTING 3. Query operations used by `NotForwardReference`

A generic type is said to be used in a type parameter bound if and only if there is an `ETypeParameter` instance among its ancestors. This is expressed by means of line 4 of Listing 3 above. If that particular `ETypeParameter` instance is its direct container, then the generic type identifies itself with the bound, otherwise it is contained at a certain level in this bound. If a generic type is involved in defining the bound of a type parameter, then this will be the only `ETypeParameter` instance among its ancestors (since no containment,

direct or not, is possible among type parameters). Therefore, the use of `any()` within the OCL expression which returns the type parameter in whose bound the current generic type is involved (line 9, Listing 3) is safe.

Resuming to the OCL definition of `NotForwardReference` (lines 8-34, Listing 1), we should make clear that a forward reference can only happen when a generic type, let us call it `GT`, references a type parameter, let us call it `T`, which is declared at a later time compared to the moment of use of `GT`. Since type parameters can only be declared within a generic classifier or operation definition, and assuming that the referenced type parameter is in scope (out of scope type parameters are ruled out by the first invariant), it follows that `GT` can only be involved in defining a bound for a type parameter owned by the same classifier or operation that owns `T`. This explains line 11 from the definition of `NotForwardReference`. Following this, the invariant computes the referenced type parameter (`refParameter`), the bounded type parameter (`boundedParameter`), and the sequence of all parameters owned by the direct container of `refParameter` (`paramSeq`). For the invariant to evaluate to `true`, `refParameter` should be declared prior to `boundedParameter` in `paramSeq` (line 28, Listing 1), or the two should be the same type parameter (line 29, same listing). In the latter case, however, it is prohibited for a type parameter to bound itself (line 30). Therefore, a situation such as the following `class Cls5<P1, P2 extends P2>` is not allowed, since `P2` bounds itself. Still, a declaration of the kind `Cls6<P3, P4 extends Cls<P4>>`, in which `P4` is involved in defining its own bound, is valid.

From the test examples above, the one intended to capture forward referencing was the fourth. There, the `NotForwardReference` invariant will be evaluated for the generic type `GT_T3`, which bounds type parameter `T2` and references type parameter `T3`. The sequence of all parameters having the same container as the referenced one evaluates to `Seq{T1,T2,T3}`, from which it is obvious that the position of the referenced type parameter (3) is greater than the one of the bounded parameter (2). Therefore, the boolean expression in lines 28-32 of Listing 1 evaluates to `false`, and so does the whole invariant.

4. Related work

As already mentioned in Section 2, the only benchmarks we have for comparing our work with are the EMF implementation of the EcoreValidator and the paper [9].

4.1. **The EMF EcoreValidator.** We have already made clear in Section 1 which are the advantages derived from using OCL, instead of a programming language, in formalizing WFRs. In addition, in Section 2, we have pointed out some of the drawbacks of the current EMF implementation of Ecore's WFRs. Among them, there have been mentioned some discrepancies between

the Java specification of generics and the corresponding WFRs implemented by the EMF EcoreValidator. We will take one such example in the following, so as to justify a new OCL WFR that we propose for the Ecore generics and that should also be implemented by the EcoreValidator.

Concerning the correct declaration of generic types and methods, the Java Language Specification [10] (pp. 50) states the following constraints: "Type variables have an optional bound, T & I_1 ... I_n. The bound consists of either a type variable, or a class or interface type T possibly followed by further interface types I_1, ..., I_n. ... It is a compile-time error if any of the types I_1 ... I_n is a class type or type variable. The order of types in a bound is only significant in that ... and that a class type or type variable may only appear in the first position."

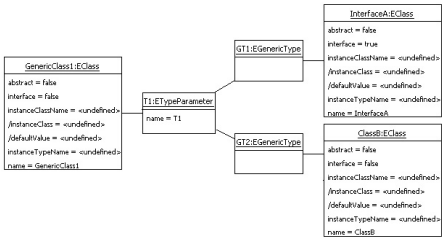


FIGURE 8. OCLE snaphot corresponding to `GenericClass1 <T1 extends InterfaceA & ClassB>`

Therefore, a generic type declaration of the kind

```
1 class GenericClass1 <T1 extends InterfaceA & ClassB>
```

where `InterfaceA` is an interface type and `ClassB` is a class type, gives the following compile-time error in a Java environment "The type ClassB is not an interface; it cannot be specified as a bounded parameter". However, by modeling the exact same type in EMF and validating it, the validation completes successfully.

In a similar manner, the declaration

```
2 class GenericClass2 <T1, T2 extends T1 & InterfaceA>
```

generates the Java compile-time error "Cannot specify any additional bound InterfaceA when first bound is a type parameter", while its equivalent Ecore model validates successfully under EMF.

This is due to the fact that the `EcoreValidator` class does not include code for checking the above mentioned constraints. Therefore, we propose the following OCL WFR for the Ecore generics:

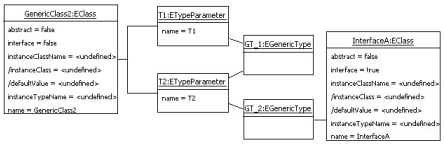


FIGURE 9. OCLE snaphot corresponding to `GenericClass2 <T1, T2 extends T1 & InterfaceA>`

```
1  context ETypeParameter
2   inv ValidBounds:
3    -- If a type parameter has bounds and the first bound is a type parameter
4    -- reference, then there are no other bounds.
5    (self.eBounds->notEmpty() and
6     self.eBounds->first().isTypeParameterReference()
7     implies self.eBounds->size() = 1
8    )
9    and
10   -- If there are at least two bounds,
11   -- then all except (maybe) the first one should refer to interface types.
12   (self.eBounds->size() >= 2
13    implies Sequence{2..self.eBounds->size()}->select(i |
14      not self.eBounds->at(i).hasInterfaceReference())->isEmpty()
15   )
```

LISTING 4. The `ValidBounds` OCL WFR

The aforementioned WFR makes use of the following query operations:

```
1  context EGenericType
2    def: hasClassifierReference() : Boolean =
3      not self.eClassifier.isUndefined()

5    def: hasClassReference() : Boolean =
6      self.hasClassifierReference() and self.eClassifier.oclIsTypeOf(EClass)

8    def: hasInterfaceReference() : Boolean =
9      self.hasClassReference() and self.eClassifier.oclAsType(EClass).interface

11   def: isTypeParameterReference() : Boolean =
12     not self.eTypeParameter.isUndefined
```

LISTING 5. Query operations used by `ValidBounds`

By evaluating the proposed WFR on the OCLE snapshots given in Figures 8 and 9, which correspond to the declarations in lines 1, respectively 2 above, the obtained result is `false` in both cases, in accordance with the Java specification.

4.2. **The approach taken in** [9]**.** In the following, we will focus on comparing our work with the one described in [9]. The above mentioned paper aims at stating a set of OCL constraints that allow checking whether (1) a given generic type declaration or (2) a corresponding instantiation with type arguments (a so called parameterized type) are well formed or not. For further reference and comparison, we provide in Listing 6 the OCL code for the `consistentTypeParameters` WFR, meant to accomplish the first goal above (the OCL specification used for accomplishing goal number 2 is omitted, since it is not directly comparable with the WFRs included in this paper). Its corresponding informal specification, as can be deduced from the paper, would be the following: "The type parameters of any classifier should have non-empty, distinct names. The bounds of a type parameter (if any) can reference either a type parameter or a classifier (they cannot be wildcards). If the bound references a type parameter, then the referenced parameter should be in scope; if it references a classifier, it should be a valid type invocation (either non-generic or generic, possibly raw)."

```
1  context EClassifier
2   inv consistentTypeParameters:
3    allDifferent(eTypeParameters.name) and
4    eTypeParameters->forAll(tp | tp.isConsistent(eTypeParameters))

6  context ETypeParameter::isConsistent(
7                              tpsInScope:Collection(ETypeParameter)):Boolean
8   def: self.name <> '' and (self.eBounds->isEmpty() or
9        self.eBounds->forAll(tr | tr.isConsistentTypeReference(tpsInScope)))

11 context EGenericType::isConsistentTypeReference(
12                            tpsInScope:Collection(ETypeParameter)):Boolean
13  def: not isWildcard() and
14       ( (self.isReferenceToTypeParameter() and
15          tpsInScope->includes(self.eTypeParameter))
16         xor
17         (self.isReferenceToClassifier() and
18          self.eClassifier.isValidTypeInvocation(self.eTypeArguments))
19       )

21 context EGenericType::isReferenceToTypeParameter():Boolean
22  def: eClassifier->isEmpty() and
23       not eTypeParameter->isEmpty() and eTypeArguments->isEmpty()

25 context EGenericType::isReferenceToClassifier():Boolean
26  def: not eClassifier->isEmpty() and eTypeParameter->isEmpty()
```

LISTING 6. The `consistentTypeParameters` constraint from [9]

The set of constraints described in [9] can be analysed with respect to both its declared purpose and our final goal of defining a complete set of WFRs for Ecore in general, and Ecore generics in particular.

Regarding the first criterion, there are certain shortcomings concerning these constraints, that we mention briefly in the following:

(1) The proposed constraints are incomplete with respect to their intended purpose. On the one side, those meant to check the well formedness of a generic type declaration only constrain the bounds of a type parameter to reference parameters from within the same type declaration, without prohibiting forward references (lines 14-15, Listing 6). However, forward referencing is not allowed, neither in EMF not in Java (whose generics' model inspired the one in Ecore). On the other side, for the WFRs that check the correct instantiation of a generic type definition (which are not reproduced here), only a skeleton is given. The OCL expressions for `captureConversion(...)` and `isSuperTypeOf(...)` (which are the core queries of these WFRs, both in matter of complexity and functionality) are missing from the paper and no reference to them is provided. As a consequence, it is impossible to evaluate on snapshots the correctness or efficiency of these WFRs.
(2) There is some redundancy in the OCL specification. The `isConsistentTypeReference(...)` query operation (starting in line 11 above) states that a generic type used in a parameter bound (1) should not be a wildcard and (2) shoud reference either a classifier or a type parameter. The latter implies the former, so it should be enough to only impose (2) as a constraint. Moreover, the `isConsistent(...)` operation (starting in line 6) requires any type parameter to have a nonempty name. However, in Ecore, `ETypeParameter` inherits `ENamedElement`, and the latter owns a WFR that checks the well formedness of its `name` attribute (well formed implies not empty).
(3) As a matter of style, the OCL specification patterns state that the use of `forAll` on collections should be avoided. The corresponding constraints should be rewritten to use either `reject` or `select`, thus allowing an easy identification of the cause of an evaluation-time error.

With respect to defining a complete set of WFRs for the Ecore generics, those described in [9] are only a small subset. They are only focused on the definition and instantiation of generic classifiers; generic operations are not taken into account. Moreover, even if a given generic type is a valid instantiation of a certain generic classifier, depending on its usage, it may be further constrained. There are various ways of using such a generic type (we have detailed on that in Subsection 3.1), with several constraints that result thereof. Here are some examples: a generic type used as a generic supertype should have a classifier that refers to a class; there may not be two different instantiations of the same generic classifier among the generic supertypes of a class; the classifier of a generic type that types an attribute should be a data

type instance, while the one used for a reference should be a class instance, and so on.

The WFRs that we have described here are part of a broader set of OCL WFRs meant to cover all constraints that apply to the Ecore concepts, generics included. Their expressions have been written following OCL specification patters that provide for an easy debugging in case of an evaluation-time error; therefore all usages of `forAll()` have been replaced with eqivalent `select()`/`reject()` rephrasings. Lines 13-14 of Listing 4 are a proof of this. Unefficient OCL constructs have only been used when there has been no other option (see the discussion related to the use of `allInstances()`), and the safety of using undeterministic constructs such as `any()` has been justified whenever the case. All WFRs have been tested and validated using OCLE.

The OCL WFRs that we have defined for generics take into account both generic classifier and generic operation declarations, as well as all previously mentioned usages of a generic type. From a completeness perspective, the WFR described in Subsection 3.2 is stronger than its equivalent part from Listing 6, since it checks for forward referencing, and this aligns it with both the EMF implementation and the Java specification; the one proposed in Subsection 4.1 is missing from the EMF implementation, while being enforced by the Java specification.

5. Conclusions and Future Work

In this paper we have reported on the definition of a set of OCL WFRs for Ecore. We have used medium-complexity WFRs for generics in order to illustrate our work and relate it to existing approaches in the literature. As far as we know, this is the first attempt to provide an OCL formalization of the well-formedness rules of Ecore. In a broader context, we have emphasized the importance of OCL WFRs in defining the abstract syntax of a (meta)modeling language.

Further work includes specifying complete sets of OCL WFRs for the other two meta-metamodels under study, MOF and XCore. This should lead to the identification of a "core" set of constraints used by all meta-metamodels. The definition of some relevant metamodels to help us in validating the proposed WFRs is also considered (in this respect, in [12], [13] we have already proposed and tested such a metamodel for components, named ContractCML).

Acknowledgements

This research has been realized in the framework of the IDEI research project "Frame based on the extensive use of metamodeling for the specification, implementation and validation of languages and applications", code

ID_2049, financed by the Romanian National University Research Council (CNCSIS).

References

[1] Eclipse Modeling Framework (EMF). http://www.eclipse.org/modeling/emf/.
[2] Frame Based on the Extensive Use of Metamodeling for the Specification, Implementation and Validation of Languages and Applications (EMF_SIVLA) project homepage. http://www.cs.ubbcluj.ro/∼chiorean/CUEM_SIVLA.
[3] Meta Object Facility (MOF) 2.0. http://www.omg.org/spec/MOF/2.0/.
[4] Model Development Tools (MDT) OCL. http://www.eclipse.org/modeling/mdt/?project=ocl.
[5] Model Driven Architecture (MDA). http://www.omg.org/mda/.
[6] Object Constraint Language Environment (OCLE). http://lci.cs.ubbcluj.ro/ocle/.
[7] Object Constraint Language (OCL) 2.0. http://www.omg.org/spec/OCL/2.0/.
[8] Tony Clark, Paul Sammut, and James Willans. *Applied Metamodeling. A Foundation for Language Driven Development*. Ceteva, 2008.
[9] Miguel Garcia. Rules for Type-checking of Parametric Polymorphism in EMF Generics. In Wolf-Gideon Bleek, Henning Schwentner, and Heinz Züllighoven, editors, *Software Engineering 2007 – Beiträge zu den Workshops*, volume 106 of *GI-Edition Lecture Notes in Informatics*, pages 261–270, 2007.
[10] James Gosling, Bill Joy, Guy Steele, and Gilad Bracha. *The Java Language Specification, Third Edition*. Addison-Wesley Longman, May 2005.
[11] Ed Merks and Marcelo Paternostro. Modeling Generics with Ecore. In *EclipseCon 2007*, 5-8 March 2007. http://www.eclipsecon.org/2007/index.php.
[12] Vladiela Petraşcu, Dan Chiorean, and Dragoş Petraşcu. ContractCML - a Contract Aware Component Modeling Language. *SYNASC 2008 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, pages 273–276, 2008.
[13] Vladiela Petraşcu, Dan Chiorean, and Dragoş Petraşcu. Component Models' Simulation in ContractCML. *Proceeding of Knowledge Engineering: Principles and Techniques (KEPT 2009), Studia. Universitatis Babes-Bolyai. Informatica, Special Issue*, pages 198–201, 2009.
[14] Douglas C. Schmidt. Model-Driven Engineering. *Computer*, 39(2):25–31, 2006.
[15] Dave Steinberg, Frank Budinsky, Marcelo Paternostro, and Ed Merks. *EMF: Eclipse Modeling Framework (2nd Edition)*. Addison-Wesley Professional, December 2008.

Babeş-Bolyai University, 1 Mihail Kogălniceanu, Cluj-Napoca, Romania
E-mail address: {vladi, chiorean, petrascu}@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

ON THE DEBTS' CLEARING PROBLEM

CSABA PĂTCAŞ

Abstract. The debts' clearing problem is about clearing all the debts in a group of n entities (eg. persons, companies) using a minimal number of money transaction operations. First we will model the problem using graph theoretical concepts and we will reduce the problem to a minimum cost maximum flow problem in a bipartite graph. Because our cost function is not the usual one, a classical minimum cost maximum flow algorithm cannot be applied in this case. We propose a solution using the dynamic programming method with $\Theta(2^n)$ space and exponential time complexity. We will also examine other solving possibilities.

1. Introduction

In this article we will discuss an original problem proposed in 2008 by the author at the qualification contest of the Romanian national team of informatics for the Central European Olympiad of Informatics and Balkan Olympiad of Informatics. The problem of debts' clearing is one, that arises in real life situations as well. In a group of persons that know each other it is not uncommon to borrow some amount of money to an acquaintance for a period of time. This process is also happening among different banks, or even countries. As money transactions are time and money sensitive operations, it is desirable to clear the debts in a minimal number of money transaction operations.

Our article studies some methods to solve this task, conjectured to be NP-complete.

Received by the editors: November 1, 2008.

2010 *Mathematics Subject Classification.* 05C85, 90C39.

1998 *CR Categories and Descriptors.* G.2.2 [**Graph theory**]: Network problems – *Network flow*.

Key words and phrases. NP-completeness, dynamic programming, network flow.

This paper has been presented at the 7th Joint Conference on Mathematics and Computer Science (7th MaCS), Cluj-Napoca, Romania, July 3-6, 2008.

2. The task

In the following we will give the problem statement.

Let us consider a number of n entities (eg. persons, companies), and a list of m borrowings among these entities. A borrowing can be described by three parameters: the index of the borrower entity, the index of the lender entity and the amount of money that was lent. The task is to find a minimal list of money transactions that clears the debts formed among these n entities as a result of the m borrowings made.

Let us clarify this by the following example:

Let $n = 6$, $m = 5$

List of borrowings:

Borrower	Lender	Amount of money
1	2	10
2	3	10
4	5	5
5	6	5
6	4	5

Solution:

Sender	Reciever	Amount of money
1	3	10

Explanation: The circular borrowings among entities 4, 5 and 6 cancel out each other, and the two borrowings made among 1, 2 and 3 can be cleared using just one money transaction.

3. Reformulation using graph theory

We can reformulate the problem using graph theoretical concepts. For this purpose we need to define some new terms first.

Definition 1. Let $G(V, A, W)$ be a directed, weighted multigraph without loops, $|V| = n$, $|A| = m$, $W : A \to \mathbb{Z}$, where V is the set of vertices, A is the set of arcs and W is the weight function. G represents the borrowings made, so we will call it the **borrowing graph**.

Definition 2. Let us define for each vertex $v \in V$ the **absolute amount of debt** over the graph G: $D_G(v) = \sum_{\substack{v' \in V \\ (v, v') \in A}} W(v, v') - \sum_{\substack{v'' \in V \\ (v'', v) \in A}} W(v'', v)$

Definition 3. $G(V, A, W) \sim G'(V, A', W')$ if and only if:

$D_G(v_i) = D_{G'}(v_i), \forall i = \overline{1,n}$, where $V = \{v_1, v_2, \ldots, v_n\}$

Theorem 4. *The "$\sim$" relation defined above is an equivalence relation.*

Proof.

(1) Reflexivity can be proved trivially: $D_G(v_i) = D_G(v_i), \forall i = \overline{1,n}$
(2) Symmetry is also trivial to prove: $D_G(v_i) = D_{G'}(v_i) \Rightarrow D_{G'}(v_i) = D_G(v_i), \forall i = \overline{1,n}$
(3) Transitivity: $D_G(v_i) = D_{G'}(v_i)$, $D_{G'}(v_i) = D_{G''}(v_i) \Rightarrow D_G(v_i) = D_{G''}(v_i), \forall i = \overline{1,n}$

□

Definition 5. There is an infinite number of G' graphs, that are in "$\sim$" relation with the G borrowing graph. As these G' graphs represent the transactions that are needed to clear the borrowings, we will call them **transaction graphs**.

In the following we will state the problem using the terms defined above:

We are looking for a minimal transaction graph $G_{min}(V, A_{min}, W_{min})$, for which $\forall G'(V, A', W') : G \sim G', |A_{min}| \leq |A'|$ holds.

Figure 1 shows the borrowing graph, that can be associated with the example given in Section 2, while Figure 2 shows the respective minimum transaction graph.

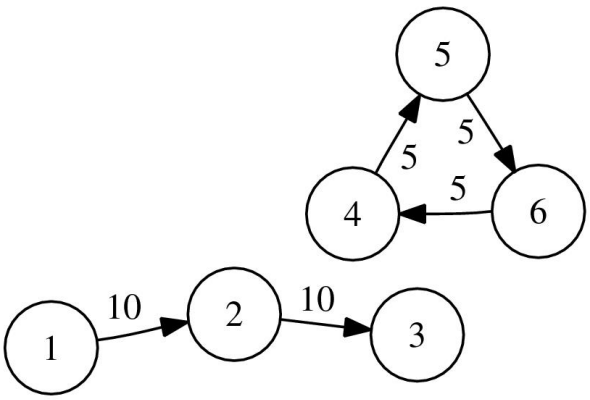


FIGURE 1. The borrowing graph associated with the given example. An arc from node i to node j with weight w means, that entity i must pay w amount of money to entity j.

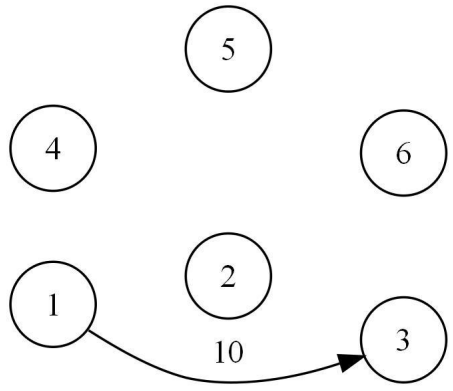


FIGURE 2. The respective minimum transaction graph. An arc from node i to node j with weight w means, that entity i pays w amount of money to entity j.

4. Possible solutions

In this Section we will propose some methods to solve the problem.

4.1. **Graph transformations.** It is clear that the original borrowing graph is also a transaction graph, because of the reflexivity of the "$\sim$" relation ($G \sim G$). This yields the following idea: Is it possible to find a sequence of graphs, such that $G = G_1, G_2, \ldots, G_k = G_{min}$, $G_1 \sim G_2$, $G_2 \sim G_3, \ldots, G_{k-1} \sim G_k$ and G_j is obtained from G_{j-1} using some "elementary transformation operation" for every $j = \overline{2,k}$? Because the "$\sim$" relation is also transitive, it would immediately result, that $G = G_1 \sim G_k = G_{min}$.

Let us enumerate some transformation operations we could use:

- All arcs having weight zero can be discarded.
- It is clear, that if we have multiple arcs between two vertices, these can be united in a single one, having a weight equal to the sum of the original weights.
- It is also easy to see, that if for any two vertices a and b we have $(a, b) \in A$ and also $(b, a) \in A$, the arc having the smaller weight out of those two can be deleted, and its weight can be substracted from the another arc.

These findings suggest the existence of an elementary transformation operation, that can be given in the most general way as follows:

(1) Let $v_{i_1}, v_{i_2}, \ldots, v_{i_p}$ be a sequence of vertices.

(2) Let x be any integer number ($x \in \mathbb{Z}$).
(3) Substract x from all the weights of two consecutive vertices in the sequence ($W(v_{i_{j-1}}, v_{i_j}) := W(v_{i_{j-1}}, v_{i_j}) - x,\ j = \overline{2,p}$). If a weight becomes negative, an edge in the opposite direction should be added. As a result some edges could disappear and new edges could appear.
(4) Add an arc between v_{i_1} and v_{i_p} having weight x.

Theorem 6. *After applying the transformation described above, the resulting graph will be in "$\sim$" relation with the original one.*

Proof.

Let $G(V, A, W)$ be the graph before the transformation and let $G'(V, A', W')$ be the graph after the transformation. Thus we must prove, that $G \sim G'$.

Let us note with d the vector of absolute amount of debts of the graph before the transformation and with d' the vector of absolute amount of debts of the graph after the transformation ($d_i = D_G(v_i), d'_i = D_{G'}(v_i), \forall i = \overline{1,n}$). Thus we must prove, that $d = d'$.

The first two steps of the transformation does not alter the graph. During the third step, the absolute amount of debts do change:

$W(v_{i_1}, v_{i_2}) = W(v_{i_1}, v_{i_2}) - x \to d'_{i_1} = d_{i_1} - x, d'_{i_2} = d_{i_2} + x$

$W(v_{i_2}, v_{i_3}) = W(v_{i_2}, v_{i_3}) - x \to d'_{i_2} = d'_{i_2} - x = d_{i_2} + x - x = d_{i_2}, d'_{i_3} = d_{i_3} + x$

$\vdots$

$W(v_{i_{p-1}}, v_{i_p}) = W(v_{i_{p-1}}, v_{i_p}) - x \to d'_{i_{p-1}} = d'_{i_p} - x = d_{i_{p-1}} + x - x = d_{i_{p-1}}, d'_{i_p} = d_{i_p} + x$

So, when step 3 is completed, the only d values that are changed are d_{i_1} and d_{i_p}, that is $d'_{i_1} = d_{i_1} - x$ and $d'_{i_p} = d_{i_p} + x$. As a result of the fourth step these elements also get back to their original value:

$W(v_{i_1}, v_{i_p}) = W(v_{i_1}, v_{i_p}) + x \to d'_{i_1} = d'_{i_1} + x = d_{i_1} - x + x = d_{i_1}, d'_{i_p} = d'_{i_p} - x = d_{i_p} + x - x = d_{i_p}$ □

An attempt of using two concrete versions of this transformation was the following:

(1) Reduce the number of arcs in all the cycles of the graph, by using the minimal weight of the cycle's arcs as the x value.
(2) The resulting graph has no cycles, thus can be sorted topologically. Try to find a strategy to reduce the number of arcs in the paths of this graph, using the topological order. For instance simplify the longest paths first, in a similar way as the cycles, by using the weights' minimum as the x value.

Unfortunately this strategy doesn't work for all graphs. The order in which we eliminate the cycles and paths does matter, and it is not clear what this order should be. In Figure 3 a borrowing graph is illustrated. If we first apply the transformation to path 2-4-5-7 no more transformations can be made, and we get a graph with 5 arcs, that can be seen in Figure 4. The optimal solution is to apply the transformation to path 1-4-5-6, then to path 3-4-5-8, thus obtaining the graph from Figure 5, which has only 4 arcs.

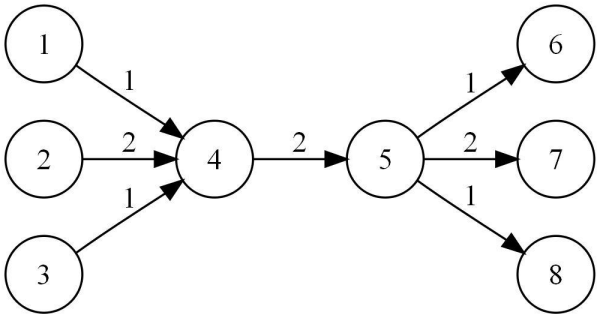


FIGURE 3. Example for a graph in which the order of transformations does matter

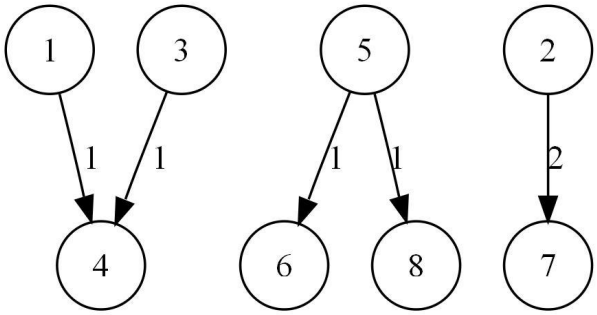


FIGURE 4. Non-optimal solution gained using graph transformations

The general problem with this approach is, that it is not clear in which order to choose which sequences, and what x values to use. The author could not find an algorithm based on this approach, that works for any borrowing graph.

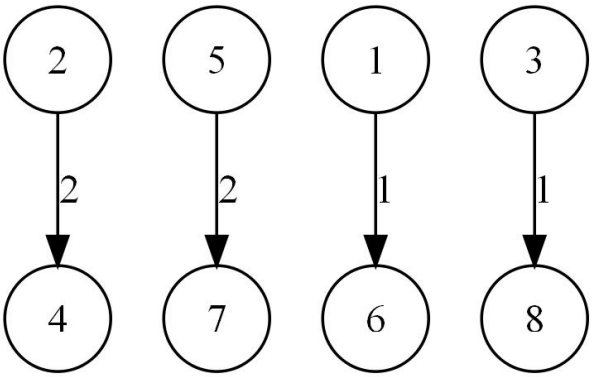


FIGURE 5. Optimal solution gained using graph transformations

4.2. **Reducing the debts' clearing to a flow problem.** A better approach is to try to solve the problem using network flows. Let us construct a bipartite graph $G_b(V_b, A_b)$ from the initial borrowing graph, as follows:

(1) $V_b = V_{left} \cup V_{right}$
(2) $V_{left} = \{v | v \in V, D_G(v) > 0\}$
(3) $V_{right} = \{v | v \in V, D_G(v) < 0\}$
(4) $A_b = \{(a, b) | a \in V_{left}, b \in V_{right}\}$

Using this graph let us construct a flow network $G_f(V_f, A_f, c)$, where $c : A_f \to \mathbb{Z}$ is the capacity function.

(1) We add a source and a sink: $V_f = V_b \cup \{s, t\}$
(2) We add arcs from the source to all the nodes from V_{left} and from all nodes from V_{right} to the sink: $A_f = A_b \cup \{(s, v) | v \in V_{left}\} \cup \{(v, t) | v \in V_{right}\}$
(3) We set the capacities of the arcs as follows:

$$c(i,j) = \begin{cases} D_G(j), & \text{if } i = s, j \in V_{left} \\ -D_G(i), & \text{if } i \in V_{right}, j = t \\ \infty, & \text{if } i \in V_{left}, j \in V_{right} \end{cases}$$

Theorem 7. *Finding the minimal transaction graph of the borrowing graph G is equivalent to finding a maximal flow in G_f with a minimal number of arcs, for which the flow is strictly positive.*

Proof.

Let F be a maximal flow with a minimal number of arcs, for which the flow is strictly positive. From this flow network we can construct the transaction graph $G'(V, A', W')$ in the following way:

(1) $V = V_f/\{s,t\} \cup \{k \in V | D_G(k) = 0\}$
(2) $A' = \{(i,j) | F(i,j) > 0, i \neq s, j \neq t\}$
(3) $W'(i,j) = F(i,j)$

Lemma 8. *The maximal flow F will saturate all arcs outgoing from s and all arcs incoming in t.*

Proof. Each arc $(i,j) \in A$ will increase $D_G(i)$ by $W(i,j)$ and decrease $D_G(j)$ by the same amount $\Rightarrow \sum_{D_G(k)>0} D_G(k) = \sum_{D_G(l)<0} -D_G(l) =: S$. In other words the sum of capacities of all arcs outgoing from s equals to the sum of capacities of all arcs incoming in t. We will prove that the maximal flow in the network has cost S. Let us suppose that the cost of the maximal flow is $S' \neq S$. As the sum of capacities of all arcs outgoing from s is $S \Rightarrow S' < S$ (the maximal flow cannot be greater than S). This means, that we have (at least) two unsaturated arcs (s,i) and (j,t). But the structure of the flow network ($c(i,j) = \infty$) leads to the existence of the augmenting path $s - i - j - t$, which contradicts the assumption, that S' was the cost of the maximal flow. We have proved, that the cost of the maximal flow F is S, which immediately yields the proof of the lemma (we cannot have this maximal flow, unless we saturate the respective arcs). □

Lemma 9. *Let G be the borrowing graph and G' be the graph constructed in the beginning of the proof of Theorem 7. Then $G' \sim G$.*

Proof.
We must prove, that $D_{G'}(v_i) = D_G(v_i) \forall i = \overline{1,n}$. We have two cases:

(1) If $D_{G'}(v_i) > 0$, there is an arc (s, v_i') in the flow network. It results from Lemma 8, that this arc is saturated. From the flow conservation rule ([3]) it results that the sum of the costs of all arcs outgoing from vertex v_i' will be $c(s, v_i') = D_G(v_i)$. From the construction of G' it follows, that there are no arcs incoming in v_i', so $D_{G'}(v_i) = D_G(v_i)$
(2) If $D_{G'}(v_i) < 0$, there is an arc (v_i', t) in the flow network, which is saturated. The rest of the proof, that $D_{G'}(v_i) = D_G(v_i)$ is done in the same manner as in the first case. □

From the fact, that we chose the maximal flow F in a way that minimizes $|A'|$, and from Lemma 9 it results, that we reduced the original problem to a maximal flow problem. □

5. Solving the flow problem

5.1. **Minimal cost maximal flow.** The first idea would be to associate costs $z(e), e \in A_f$ to the arcs of the flow network. Let the arcs between V_{left} and V_{right} have cost 1, and the other arcs have cost 0, and let us use a classical minimum cost maximum flow algorithm to solve the problem:

$$z(i,j) = \begin{cases} 1, & \text{if } i \in V_{left}, j \in V_{right} \\ 0, & \text{otherwise} \end{cases}$$

Figure 6 shows the flow network, that can be associated to the borrowing graph shown in Figure 1. The first number on each arc represents the capacity of the arc, and the second number represents the cost of the arc.

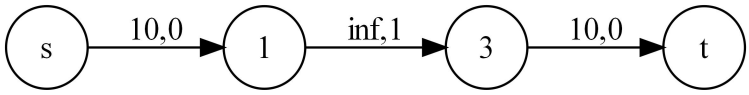


Figure 6. The flow network associated with the example

This approach doesn't work, because the cost of an arc is multiplied by the flow, we don't have fixed costs on the arcs. In a classical minimum cost maximum flow algorithm the cost function looks like: $Cost(e) = f(e) \cdot z(e)$, but in our case this is:

$$Cost(e) = \begin{cases} z(e), & \text{if } f(e) > 0 \\ 0, & \text{otherwise} \end{cases}$$

It can be rewritten to a more simple form, as: $Cost(e) = ind(f(e)) \cdot z(e)$, where ind is the indicator function, defined as follows:

$$ind(x) = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{if } x = 0 \end{cases}$$

In the literature this problem is called Minimum Edge-Cost Flow and is known to be NP-complete([4], problem [ND32]). Even if the problem is constrained so that the capacity of each arc is 2, and the cost of each arc can be 0 or 1, it remains NP-complete. This fact leads us to the following conjecture:

Conjecture 10. *The debts' clearing problem is NP-complete.*

The fact that we have a special case, with all the costs equaling to 0 or 1, and having a complete bipartite graph, gives us a hope that a polynomial solution does exist.

5.2. **Convex flow.** Efficient minimum cost maximum flow algorithms do exist, when the cost function is convex ([1]). Unfortunately this is not our case. Indeed, a function to be convex must satisfy:

$f(t\cdot x+(1-t)\cdot y) \leq t\cdot f(x)+(1-t)\cdot f(y)$ for any x and y from the function domain, and $\forall t \in [0,1]$. For $t=\frac{1}{2}, x=0, y=1$ we get $f(\frac{1}{2}) \leq \frac{f(0)}{2}+\frac{f(1)}{2}$, which is obviously not true in our case, so our cost function is not convex.

5.3. **Nonlinear programming.** We can formulate the problem as a nonlinear programming (NLP) problem:

Minimize $p = \sum_{e\in A_f} ind(f(e))$ subject to

$\sum_{k\in V_{right}} f(i,k) = c(s,i)$

$\sum_{k\in V_{left}} f(k,j) = c(j,t)$

Unfortunately no polinomial algorithm is known for solving any NLP instance. However good approximation algorithms do exist ([6]).

5.4. **Dynamic programming.** We will give a solution using the dynamic programming method. It uses similar techniques to the algorithm discovered independently by Bellman ([2]), respectively Held and Karp ([5]) for solving the Traveling Salesman Problem.

Let us note $n_1 = |V_{left}|$ and $n_2 = |V_{right}|$. Let us define the subproblems of the dynamic programming problem with two parameters i and j, where i is a binary representation of n_1 bits, and j is a binary representation of n_2 bits ($i = \overline{0, 2^{n_1}-1}, j = \overline{0, 2^{n_2}-1}$). A subproblem will have the following meaning:

$dp_{i,j}$ = the minimal number of arcs having strictly positive flow, such that the arcs between s and the nodes determined by the bits of i, and the arcs between the nodes determined by the bits of j and t are all saturated.

The recursive formula to determine the values of the subproblems is the following[1]:

$dp_{i,j} = \min(dp_{i \texttt{ XOR } i', j \texttt{ XOR } j'} + bitcount(i') + bitcount(j') - 1)$, where

(1) i `AND` $i' = i'$

[1]We note by `AND` the bitwise and operation and by `XOR` the bitwise exclusive or operation

(2) j AND $j' = j'$

(3) $\sum_{i' \text{ AND } 2^k \neq 0} c(s,k) = \sum_{j' \text{ AND } 2^k \neq 0} c(k,t)$

(4) $bitcount(x)$ returns the number of bits of x equal to 1.

It can be easily seen, that in the worst case we will need $n_1 + n_2 - 1$ arcs, and in the best case $\max(n_1, n_2)$ arcs. We choose all the possible subsets i' of i and j' of j. The nodes determined by i' and j' can form an "independent subnetwork" only if the respective sums of capacities are equal. In this case we consider the worst case scenario, so we add $bitcount(i') + bitcount(j') - 1$ to the solution not containing these nodes. If we didn't find an independent subnetwork, it means, that the chosen arcs must form a connected graph, thus the minimal number of arcs is $n_1 + n_2 - 1$, so we can't get any better than the worst case scenario.

Let us analyze the performance of the proposed algorithm. The number of subproblems is $2^{n_1} \cdot 2^{n_2} = 2^{n_1+n_2}$, which in the worst case is 2^n. Thus the space complexity of our algorithm is $\Theta(2^n)$. To solve a subproblem (i, j) we need all the pairs (i', j'), such that i' is a subset of i and j' is a subset of j. We can codify any pair (i, i') with a sequence of length n_1 of ternary digits. A digit will be 0, if the respective node is not in i, 1 if it is in i but not in i' and 2 if it is in i' (and thus also in i). The same codification can be done for any (j, j') pair. Thus the number of steps performed by our algorithm is proportional to $3^{n_1} \cdot 3^{n_2} = 3^n$

6. Conclusions and future work

In this article we stated the debt's clearing problem, and analyzed some solving possibilities. An algorithm, that finds the optimal solution using the dynamic programming method was given, and its exponential running time and space complexity was proven.

We didn't analyze deeply the solving possibilities using nonlinear programming strategies. Also, the presented non optimal solutions, such as convex flow, could work in a high percentage of the cases, this possibility must be studied in the future. Approximation algorithms were not considered.

The biggest concern regarding this problem is its NP-completeness. The problem is a special instance of the known NP-complete problem of Minimum Edge-Cost Flow, and seems to be NP-complete too.

References

[1] Ravindra K. Ahuja and Thomas L. Magnanti and James B. Orlin, *Network flows*, Prentice-Hall, 1993.

[2] Bellman, Richard, *Dynamic Programming Treatment of the Travelling Salesman Problem*, Journal of the ACM, 1962.

[3] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein, *Introduction to Algorithms (2nd edition)*, 2001.

[4] Michael R. Garey, David S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*, W. H. Freeman and Company, San Francisco, 1979.

[5] Held, Michael and Karp, Richard M, *A dynamic programming approach to sequencing problems*, Proceedings of the 1961 16th ACM national meeting, 1961.

[6] D. Luenberger, *Linear and nonlinear programming*, Addison-Wesley, New York, 1989.

Department of Computer Science, Babeş-Bolyai University, M. Kogălniceanu 1, 400084 Cluj-Napoca, Romania

E-mail address: patcas@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

A PREDICTOR-CORRECTOR ALGORITHM FOR LINEARLY CONSTRAINED CONVEX OPTIMIZATION

ZSOLT DARVAY

Abstract. In a recent paper we have introduced a new class of search directions for solving linear optimization (LO) problems. These directions are based on an algebraic equivalent transformation of the nonlinear equation from the system which defines the central path. However, from the implementation point of view predictor-corrector algorithms proved to be the most efficient among the class of interior point methods (IPMs). Therefore, we have defined also other variants of this class of algorithms, for example a weighted path-following algorithm, and a predictor-corrector algorithm for LO problems. Recently, the technique of finding search directions has been applied with success for linearly constrained convex optimization (LCCO), by Zhang, Bai and Wang. In this paper we define a new predictor-corrector algorithm for solving LCCO problems. We obtain new search directions by applying the method of algebraic equivalent transformation in this case too. Polynomial complexity of this algorithm is proved.

1. Introduction

The field of IPMs has been very active since Karmarkar published his famous paper [7] in 1984. However, in general one of the significant aspects of determining a new algorithm resides in the method of following the central path. Therefore, search directions play an important role in finding new algorithms. Peng, Roos and Terlaky [10] have defined the notion of self regular functions and, using this concept, they have introduced a new class of search directions for LO. They have extended their results also to complementarity problems (CP), semidefinite optimization (SDO) and second order cone optimization (SOCO), and they have proved polynomial complexity of

Received by the editors: November 20, 2009.

2010 *Mathematics Subject Classification.* 90C25, 90C51.

1998 *CR Categories and Descriptors.* G.1.6. [**Mathematics of Computing**]: Numerical Analysis – *Optimization – Convex programming*;

Key words and phrases. Linearly constrained convex optimization, interior-point methods, predictor-corrector algorithm, equivalent algebraic transformation, centering equation, Newton step.

different large-update algorithms, which use self-regular functions to obtain new directions.

An alternative method has been introduced in [3, 5] by applying algebraically equivalent transformations to the nonlinear centering equation of the system, which defines the central path. The method has been applied with success to convex quadratic optimization (CQO) by Achache [2] and LCCO by Zhang et al. [17]. Recently, the new technique for LO has been extended also to monotone mixed linear complementarity problems (LCPs) by Wang, Cai and Yue [15], and to SDO and SOCO by Wang and Bai [13, 14]. The method of algebraically equivalent transformation has been generalized also to weighted-path-following algorithms. The first results for LO have been given in [4]. Later on, Achache [1] generalized this algorithm to standard LCPs, and Wang et al. [16] to monotone horizontal LCPs. The above mentioned algebraic transformations, followed by a Newton step, resulted in small-update feasible algorithms, and for all of them the best known iteration bounds were obtained. However, Pan, Li and He [9] introduced a large-update infeasible algorithm using a logarithmic transformation. This logarithmic equivalent transformation was mentioned formerly by Tuncel and Todd [12].

Another approach of developing an efficient algorithm is considering predictor and corrector steps. We mention that the first algorithm, that have divided the Newton direction into the affine-scaling and centering direction, is due to Mehrotra [8]. In [6] we have defined a predictor-corrector algorithm obtained by applying an equivalent algebraic transformation, using the square root function, as in [3]. In this paper we extend this algorithm to LCCO.

The notations used in this paper are the following: $\Re^n$ is the set of n-dimentional vectors and $\Re^{m\times n}$ is the set of $m \times n$ matrices. Moreover, $\Re^+$ is the set of nonnegative real numbers, if $x \in \Re^n$ then $diag(x)$ is the diagonal matrix formed by the elements of x, and e is the all-one vector.

This paper is organized in the following way. In the next section we introduce the basic issues regarding the central path, and we discuss the primal-dual algorithm for LCCO. In the third section we define the predictor-corrector algorithm for LCCO and we prove its polynomial complexity. Finally, we present some conclusions in Section 4.

2. Primal-Dual Path-Following Algorithm

Let us consider the following problem

$$(P) \qquad min\ \{f(x) : Ax = b, x \geq 0\},$$

and its Wolfe dual

$$(D) \qquad max\ \{b^T y + f(x) - (\nabla f(x))^T x : A^T y + s - \nabla f(x) = 0, s \geq 0\},$$

where $A \in \Re^{m\times n},\ rank(A) = m,\ b \in \Re^m$ and $f : \Re^n \to R$ is a convex and twice continuously differentiable function. Suppose that the interior point condition (IPC) holds. Thus, there exist (x^0, y^0, s^0) such that

$$\text{(IPC)} \qquad \begin{aligned} Ax^0 &= b, \qquad x^0 > 0, \\ A^T y^0 + s^0 - \nabla f(x^0) &= 0, \qquad s^0 > 0. \end{aligned}$$

The IPC can be assumed without loss of generality, and we may assume $x^0 = s^0 = e$. The optimality condition for the pair (P)-(D) can be written as

$$\text{(1)} \qquad \begin{aligned} Ax &= b, \qquad x \geq 0, \\ A^T y + s - \nabla f(x) &= 0, \qquad s \geq 0, \\ xs &= 0. \end{aligned}$$

If the IPC holds, then for a fixed $\mu > 0$ the system

$$\text{(2)} \qquad \begin{aligned} Ax &= b, \qquad x > 0, \\ A^T y + s - \nabla f(x) &= 0, \qquad s > 0, \\ xs &= \mu e, \end{aligned}$$

has a unique solution, called the *μ-center*. Let us consider the function φ such that

$$\text{(3)} \qquad \varphi \in C^1,\ \varphi : \Re^+ \to \Re^+, \text{ and } \varphi^{-1} \text{ exists.}$$

Then, the system (2) is equivalent to

$$\text{(4)} \qquad \begin{aligned} Ax &= b, \qquad x > 0, \\ A^T y + s - \nabla f(x) &= 0, \qquad s > 0, \\ \varphi\left(\frac{xs}{\mu}\right) &= \varphi(e), \end{aligned}$$

Assume that we have $Ax = b,\ x > 0,\ A^T y + s - \nabla f(x) = 0,\ s > 0$ for a triple (x, y, s). Applying Newton's method for system (4) we get

$$\text{(5)} \qquad \begin{aligned} A\Delta x &= 0, \\ A^T \Delta y + \Delta s - \nabla^2 f(x)\Delta x &= 0, \\ \frac{s}{\mu}\varphi'\left(\frac{xs}{\mu}\right)\Delta x + \frac{x}{\mu}\varphi'\left(\frac{xs}{\mu}\right)\Delta s &= \varphi(e) - \varphi\left(\frac{xs}{\mu}\right). \end{aligned}$$

Denote $$d_x = \frac{v\Delta x}{x}, \qquad d_s = \frac{v\Delta s}{s}.$$

We have

$$\text{(6)} \qquad \mu v(d_x + d_s) = s\Delta x + x\Delta s,$$

$$d_x d_s = \frac{\Delta x \Delta s}{\mu}. \tag{7}$$

The linear system (5) is equivalent to

$$\begin{aligned} \bar{A} d_x &= 0, \\ \bar{A}^T d_y + d_s - \bar{H} d_x &= 0, \\ d_x + d_s &= p_v, \end{aligned} \tag{8}$$

where $V = diag(v)$, $X = diag(x)$, $S = diag(s)$ and we also use the following notations:

$$p_v = \frac{\varphi(e) - \varphi(v^2)}{v\varphi'(v^2)}, \quad \bar{H} = \mu V S^{-1} \nabla^2 f(x) V S^{-1} \quad \text{and} \quad \bar{A} = \frac{1}{\mu} A V^{-1} X.$$

Observe that $\varphi(t) = t$ yields $p_v = v^{-1} - v$, and we obtain the standard primal-dual algorithm. Let $\varphi(t) = \sqrt{t}$. Then we have

$$p_v = 2(e - v). \tag{9}$$

From (5) we obtain

$$\begin{aligned} A\Delta x &= 0, \\ A^T \Delta y + \Delta s - \nabla^2 f(x) \Delta x &= 0, \\ \sqrt{\frac{s}{x}} \Delta x + \sqrt{\frac{x}{s}} \Delta s &= 2(\sqrt{\mu} e - \sqrt{xs}). \end{aligned} \tag{10}$$

The system (10) can be written in the following form:

$$\begin{aligned} A\Delta x &= 0, \\ A^T \Delta y + \Delta s - \nabla^2 f(x) \Delta x &= 0, \\ s\Delta x + x\Delta s &= 2(\sqrt{\mu x s} - xs). \end{aligned} \tag{11}$$

We define a proximity measure to the central path

$$\sigma(xs, \mu) = \frac{\|p_v\|}{2} = \|e - v\| = \left\| e - \sqrt{\frac{xs}{\mu}} \right\|. \tag{12}$$

Denote $q_v = d_x - d_s$. The function f is convex, thus the matrices $\nabla^2 f(x)$ and $\bar{H}$ are symmetric and positive semidefinite. Thus $d_x^T d_s \geq 0$ and

$$\|q_v\| \leq \|p_v\|.$$

We have

$$\sigma(xs, \mu) \geq \frac{\|q_v\|}{2},$$

$$d_x = \frac{p_v + q_v}{2}, \qquad d_s = \frac{p_v - q_v}{2},$$

$$d_x d_s = \frac{p_v^2 - q_v^2}{4}. \tag{13}$$

Algorithm 2.1 *Let $\epsilon > 0$ be the accuracy parameter, $0 < \theta < 1$ the update parameter (default $\theta = \frac{1}{2\sqrt{n}}$), and $0 < \tau < 1$ the proximity parameter (default $\tau = \frac{1}{2}$). Suppose that for the triple (x^0, y^0, s^0) the IPC holds, and let $\mu^0 = \frac{(x^0)^T s^0}{n}$. Furthermore, suppose $\sigma(x^0 s^0, \mu^0) < \tau$.*

begin

$x := x^0;\ y = y^0;\ s = s^0;\ \mu := \mu^0;$

while $x^T s > \epsilon$ **do begin**

$\mu := (1 - \theta)\mu;$

Compute $(\Delta x, \Delta y, \Delta s)$ from (11)

$x := x + \Delta x;$

$y := y + \Delta y;$

$s := s + \Delta s;$

end

end.

The complexity analysis of this algorithm has been given in [17]. The results are similar to the LO analogue [5]. We revisit the following lemmas given in [17].

Lemma 2.1 *Let $x_+ = x + \Delta x$ and $s_+ = s + \Delta s$. Moreover, let $\sigma = \sigma(xs, \mu)$ and suppose that $\sigma < 1$. Then*

$$x_+ > 0 \quad \textit{and} \quad s_+ > 0,$$

hence the full Newton step is strictly feasible.

Proof: Denote $x_+(\alpha) = x + \alpha\Delta x$ and $s_+(\alpha) = s + \alpha\Delta s$ for each $0 \le \alpha \le 1$. We have

$$\frac{1}{\mu} x_+(\alpha) s_+(\alpha) = (1 - \alpha)v^2 + \alpha\left(e - (1 - \alpha)\frac{p_v^2}{4} - \alpha\frac{q_v^2}{4}\right). \tag{14}$$

$$\left\| (1 - \alpha)\frac{p_v^2}{4} + \alpha\frac{q_v^2}{4} \right\|_\infty \le \sigma^2 < 1.$$

Thus, for each $0 \le \alpha \le 1$ we have $x_+(\alpha)s_+(\alpha) > 0$. ∎

Lemma 2.2 *Let $\sigma = \sigma(xs, \mu) < 1$. Then*

$$\sigma(x_+ s_+, \mu) \le 1 - \sqrt{1 - \sigma^2}.$$

Thus the Newton process is quadratically convergent.

Proof: Let $v_+ = \sqrt{\frac{x_+ s_+}{\mu}}$. Using (14) we obtain

$$v_+^2 = e - \frac{q_v^2}{4} \tag{15}$$

$$\min(v_+) = \sqrt{1 - \frac{1}{4}\|q_v^2\|_\infty} \geq \sqrt{1 - \frac{\|q_v\|^2}{4}} \geq \sqrt{1-\sigma^2}. \tag{16}$$

$$\sigma(x_+s_+,\mu) = \left\| \frac{e - v_+^2}{e + v_+} \right\| \leq \frac{\sigma^2}{1+\sqrt{1-\sigma^2}} = 1 - \sqrt{1-\sigma^2}.$$

Hence $\sigma(x_+s_+,\mu) < \sigma^2$, and this proves the lemma. ■

Lemma 2.3 *Let $\sigma = \sigma(xs,\mu)$ and suppose that the vectors x_+ and s_+ are obtained after a full Newton step, thus $x_+ = x + \Delta x$ and $s_+ = s + \Delta s$. We have*

$$(x_+)^T s_+ = \mu(n - \frac{\|q_v\|^2}{4})$$

Hence $(x_+)^T s_+ \leq \mu n$.

Proof: We have

$$\frac{1}{\mu} x_+ s_+ = e - \frac{q_v^2}{4}.$$

Consequently

$$(x_+)^T s_+ = e^T(x_+s_+) = \mu(e^Te - \frac{e^Tq_v^2}{4}) = \mu(n - \frac{\|q_v\|^2}{4}) = \mu(n-\sigma^2).$$

This proves the lemma. ■

Lemma 2.4 *Let $\sigma = \sigma(xs,\mu) < 1$ and $\mu_+ = (1-\theta)\mu$, where $0 < \theta < 1$. Then*

$$\sigma(x_+s_+,\mu_+) \leq \frac{\theta\sqrt{n} + \sigma^2}{1 - \theta + \sqrt{(1-\theta)(1-\sigma^2)}}$$

Moreover, if $\sigma \leq \frac{1}{2}$, $\theta = \frac{1}{2\sqrt{n}}$ and $n \geq 4$ then we have $\sigma(x_+s_+,\mu_+) \leq \frac{1}{2}$.

Proof: Using (15) and (16) we may write

$$\sigma(x_+s_+,\mu_+) = \left\| e - \sqrt{\frac{x_+s_+}{\mu_+}} \right\| = \frac{1}{\sqrt{1-\theta}} \left\| \frac{(1-\theta)e - v_+^2}{\sqrt{1-\theta}e + v_+} \right\| \leq$$

$$\leq \frac{1}{1 - \theta + \sqrt{(1-\theta)(1-\sigma^2)}} \left\| -\theta e + \frac{q_v^2}{4} \right\| \leq \frac{\theta\sqrt{n} + \sigma^2}{1 - \theta + \sqrt{(1-\theta)(1-\sigma^2)}}$$

This implies the first part of the lemma. To prove the second part observe that for $n \geq 4$ and $\theta = \frac{1}{2\sqrt{n}}$ we have $1 - \theta \geq \frac{3}{4}$. Finally, for $\sigma \leq \frac{1}{2}$ a simple calculus yields $\sigma(x_+s_+,\mu_+) \leq \frac{1}{2}$. ■

Lemma 2.5 *Suppose that $x^0 = s^0 = e$. Then Algorithm 2.1 performs at most*

$$\left\lceil \frac{1}{\theta} \log \frac{n}{\epsilon} \right\rceil$$

interior point iterations.

Proof: For the proof we refer to [17], and for the LO variant [5]. ■

Thus, we obtain the following theorem.

Theorem 2.6 *Suppose that $x^0 = s^0 = e$. Using the default values for θ and τ Algorithm 2.1 requires no more than*

$$\left\lceil 2\sqrt{n}\log\frac{n}{\epsilon}\right\rceil$$

interior point iterations. The resulting vectors satisfy $x^T s \leq \epsilon$. ■

3. Predictor-corrector algorithm

The third equation in the system (8) can be written in the form:

$$d_x + d_s = 2e - 2v.$$

Observe that the expression on the right hand side can be viewed as a sum of two terms. Consider the following equations.

$$d_x^a + d_s^a = -2v, \tag{17}$$

$$d_x^c + d_s^c = 2e, \tag{18}$$

and conclude that the standard Newton direction has been breaking down into two steps, the affine-scaling, or predictor one: d_x^a and d_s^a, and the centering, or corrector step: d_x^c and d_s^c. The equations (17) and (18) yield the following systems:

$$\begin{aligned} \bar{A}d_x^a &= 0,\\ \bar{A}^T d_y^a + d_s^a - \bar{H}d_x^a &= 0,\\ d_x^a + d_s^a &= -2v, \end{aligned} \tag{19}$$

and

$$\begin{aligned} \bar{A}d_x^c &= 0,\\ \bar{A}^T d_y^c + d_s^c - \bar{H}d_x^c &= 0,\\ d_x^c + d_s^c &= 2e, \end{aligned} \tag{20}$$

where $\bar{A} = \frac{1}{\mu}Adiag(\frac{x}{v})$ and $\bar{H} = \mu V S^{-1}\nabla^2 f(x) V S^{-1}$. The systems (19) and (20) have unique solutions. Denote by d_x^a, d_s^a, d_x^c and d_s^c these solutions. We have

$$(d_x^a)^T d_s^a \geq 0, \qquad (d_x^c)^T d_s^c \geq 0, \tag{21}$$

and the solution of (8) can be obtained from (19) and (20) as follows

$$d_x = d_x^a + d_x^c,$$

$$d_s = d_s^a + d_s^c.$$

The step direction vectors in the original space are

$$\Delta^a x = \frac{x}{v} d_x^a, \qquad \Delta^a s = \frac{s}{v} d_s^a, \qquad \Delta^a y = d_y^a,$$

$$\Delta^c x = \frac{x}{v} d_x^c, \qquad \Delta^c s = \frac{s}{v} d_s^c, \qquad \Delta^c y = d_y^c.$$

Thus

$$x\Delta^a s + s\Delta^a x = \mu v(d_x^a + d_s^a) = -2\mu v^2 = -2xs, \tag{22}$$

$$x\Delta^c s + s\Delta^c x = \mu v(d_x^c + d_s^c) = 2\mu v = 2\sqrt{xs\mu}, \tag{23}$$

$$\Delta^a x \Delta^a s = \mu d_x^a d_s^a. \tag{24}$$

From (22) we obtain that $(\Delta^a x, \Delta^a y, \Delta^a s)$ is the solution of the system:

$$\begin{aligned} A\Delta^a x &= 0, \\ A^T\Delta^a y + \Delta^a s - \nabla^2 f(x)\Delta^a x &= 0, \\ s\Delta^a x + x\Delta^a s &= -2xs. \end{aligned} \tag{25}$$

Now we are ready to describe the predictor-corrector algorithm.

Algorithm 3.1

Let $0 < \tau < 1$ be the proximity parameter (default value $\tau = \frac{5}{13}$), $\epsilon > 0$ the accuracy parameter, and $0 < \theta < \frac{1}{2}$ the update parameter (default $\theta = \frac{1}{3\sqrt{n}}$). Assume that for the triple (x^0, y^0, s^0) IPC holds, and let $\mu^0 = \frac{(x^0)^T s^0}{n}$. Furthermore, assume that $\sigma(x^0 s^0, \mu^0) \le \tau$.

begin

$x := x^0$; $s := s^0$; $\mu := \mu^0$;

while $x^T s > \epsilon$ **do begin**

Compute $(\Delta x, \Delta y, \Delta s)$ from (11).

$x := x + \Delta x$;

$s := s + \Delta s$;

Compute $(\Delta^a x, \Delta^a y, \Delta^a s)$ using the system (25).

$x := x + \theta\Delta^a x$;

$s := s + \theta\Delta^a s$;

$\mu := (1 - 2\theta)\mu$;

end

end.

Our first aim is to prove that this algorithm is well defined. We discuss also the complexity of the algorithm. To achieve these goals, in the first lemma we find lower and upper bounds for the components of the vector v.

Lemma 3.1 *Let $x > 0$, $s > 0$, $\mu > 0$, $v = \sqrt{\frac{xs}{\mu}}$ and $\sigma = \sigma(xs, \mu) = \|e - v\|$. Assume that $\sigma < 1$. Then, for all i such that $1 \le i \le n$, we have*

$$1 - \sigma \le v_i \le 1 + \sigma.$$

Moreover, the following inequalities hold

$$\min\left(v^2\right) \geq (1-\sigma)^2, \quad \|v\|^2 \leq n(1+\sigma)^2. \tag{26}$$

Proof: See [6]. ■

The following lemma provides upper bounds for the Euclidian norm and the infinity norm of the product of two vectors with non-negative inner product.

Lemma 3.2 (Extension of the first *uv*-lemma in [11]**)** *Let $\pi \in \Re^n$ and $\zeta \in \Re^n$ are two vectors such that $\pi^T\zeta \geq 0$. Then we have*

$$\|\pi\zeta\|_\infty \leq \frac{1}{4}\|\pi+\zeta\|^2, \quad \|\pi\zeta\| \leq \frac{\sqrt{2}}{4}\|\pi+\zeta\|^2.$$

Proof:

We have $\|\pi+\zeta\|^2 = \|\pi-\zeta\|^2 + 4\pi^T\zeta$, resulting $\|\pi+\zeta\| \geq \|\pi-\zeta\|$. Furthermore

$$\pi\zeta = \frac{1}{4}(\pi+\zeta)^2 - \frac{1}{4}(\pi-\zeta)^2. \tag{27}$$

Thus

$$-\frac{1}{4}(\pi-\zeta)^2 \leq \pi\zeta \leq \frac{1}{4}(\pi+\zeta)^2,$$

and

$$-\frac{1}{4}\|\pi+\zeta\|^2 e \leq -\frac{1}{4}\|\pi-\zeta\|^2 e \leq \pi\zeta \leq \frac{1}{4}\|\pi+\zeta\|^2 e.$$

This proves the first inequality. To prove the second one, observe that from (27) and $\|\pi\zeta\|^2 = e^T(\pi\zeta)^2$ we obtain:

$$\|\pi\zeta\|^2 = \frac{1}{16}e^T\left((\pi+\zeta)^2-(\pi-\zeta)^2\right)^2 \leq \frac{1}{16}e^T\left((\pi+\zeta)^4+(\pi-\zeta)^4\right).$$

Moreover, for each $\xi \in \Re^n$ the $e^T\xi^4 \leq \|\xi\|^4$ inequality holds, therefore

$$\|\pi\zeta\|^2 \leq \frac{1}{16}\|\pi+\zeta\|^4 + \frac{1}{16}\|\pi-\zeta\|^4,$$

and using again the relation $\|\pi-\zeta\| \leq \|\pi+\zeta\|$ we get the second inequality. This proves the lemma. ■

We introduce the followig notations. Let

$$\lambda(\sigma) = \left(1+\sqrt{2}\right)\sigma^2 - 2\left(\sqrt{2}-1\right)\sigma + \sqrt{2}, \tag{28}$$

$$K(\sigma,\theta,n) = (1-\sigma)^2 - \frac{\theta^2 n}{1-2\theta}(1+\sigma)^2, \tag{29}$$

and

$$\Phi(\sigma,\theta,n) = \frac{\lambda(\sigma) - \sqrt{2}K(\sigma,\theta,n)}{1+\sqrt{K(\sigma,\theta,n)}}. \tag{30}$$

Consider the following function

$$\omega(t) = 1 - \sqrt{1-t^2}, \tag{31}$$

defined for every $0 \le t < 1$. For fixed τ introduce the function

$$\Psi_\tau(\sigma) = \frac{1}{(1+\sigma)^2}\left(\sigma^2 - 2\sigma + \sqrt{2}\tau - \frac{\tau^2}{2}\right). \tag{32}$$

We give a sufficient condition for yielding strictly feasible vectors after an affine-scaling step.

Lemma 3.3 *Let $x > 0$, $s > 0$, $\mu > 0$ in such a way, that $\sigma = \sigma(xs, \mu) < 1$. Furthermore, let $0 < \theta < \frac{1}{2}$. Denote $x^+ = x + \theta\Delta^a x$ and $s^+ = s + \theta\Delta^a s$. Then*

$$x^+ > 0 \quad \textit{and} \quad s^+ > 0,$$

if the inequality $K(\sigma, \theta, n) > 0$ holds.

Proof: Let us introduce the notations

$$x^+(\beta) = x + \beta\theta\Delta^a x \quad \text{and} \quad s^+(\beta) = s + \beta\theta\Delta^a s$$

for each real number $0 \le \beta \le 1$. We have the following equality:

$$x^+(\beta)s^+(\beta) = xs + \beta\theta(x\Delta^a s + s\Delta^a x) + \beta^2\theta^2\Delta^a x\Delta^a s.$$

Using the relations (22) and (24) we get:

$$x^+(\beta)s^+(\beta) = (1 - 2\beta\theta)xs + \mu\beta^2\theta^2 d_x^a d_s^a \tag{33}$$

thus we obtain

$$\frac{x^+(\beta)s^+(\beta)}{(1-2\beta\theta)\mu} = v^2 + \frac{\beta^2\theta^2}{(1-2\beta\theta)} d_x^a d_s^a. \tag{34}$$

Therefore

$$\min\left(\frac{x^+(\beta)s^+(\beta)}{(1-2\beta\theta)\mu}\right) \ge \min\left(v^2\right) - \frac{\beta^2\theta^2}{1-2\beta\theta}\left\|d_x^a d_s^a\right\|_\infty.$$

Moreover, for each fixed $0 < \theta < \frac{1}{2}$, the function $\vartheta(\beta) = \frac{\beta^2\theta^2}{1-2\beta\theta}$ defined for $0 \le \beta \le 1$ is strictly increasing, thus

$$\min\left(\frac{x^+(\beta)s^+(\beta)}{(1-2\beta\theta)\mu}\right) \ge \min\left(v^2\right) - \frac{\theta^2}{1-2\theta}\left\|d_x^a d_s^a\right\|_\infty. \tag{35}$$

From Lemma 3.2, using the equality (17) and Lemma 3.1 we obtain

$$\left\|d_x^a d_s^a\right\|_\infty \le \frac{1}{4}\left\|d_x^a + d_s^a\right\|^2 = \|v\|^2 \le n(1+\sigma)^2. \tag{36}$$

Now, using the relation (35) and Lemma 3.1 again, we get

$$\min\left(\frac{x^+(\beta)s^+(\beta)}{(1-2\beta\theta)\mu}\right) \ge K(\sigma, \theta, n) \tag{37}$$

But $K(\sigma,\theta,n) > 0$, and we deduce that for each $0 \leq \beta \leq 1$ the inequality $x^+(\beta)s^+(\beta) > 0$ holds. Therefore the $x^+(\beta)$ and $s^+(\beta)$ functions are not changing sign on the $[0,1]$ interval. We know that $x^+(0) = x > 0$, and $s^+(0) = s > 0$, thus we conclude that $x^+(1) = x^+ > 0$, and $s^+(1) = s^+ > 0$. This proves the lemma. ■

In the next lemma we investigate the modification of the proximity measure after an affine-scaling step, and the update of the parameter μ.

Lemma 3.4 *Let $x > 0$, $s > 0$, $\mu > 0$ such that $\sigma = \sigma(xs,\mu) < 1$. Moreover, let $0 < \theta < \frac{1}{2}$ and assume that $K(\sigma,\theta,n) > 0$. Assume that we obtain the vectors x^+ and s^+ from an affine-scaling step, thus $x^+ = x + \theta\Delta^a x$ and $s^+ = s + \theta\Delta^a s$. Denote $\mu^+ = (1-2\theta)\mu$ and $\sigma^+ = \sigma(x^+s^+,\mu^+)$. Then the inequality*

$$\sigma^+ \leq \Phi(\sigma,\theta,n) \tag{38}$$

holds.

Proof: From Lemma 3.3 we deduce that the affine-scaling step is strictly feasible. Denote

$$v^+ = \sqrt{\frac{x^+s^+}{\mu^+}}.$$

By substituting $\beta = 1$ in the relations (34) and (37) we get

$$\left(v^+\right)^2 = v^2 + \frac{\theta^2}{1-2\theta} d_x^a d_s^a, \tag{39}$$

$$\min\left(v^+\right) \geq \sqrt{K(\sigma,\theta,n)}. \tag{40}$$

Moreover

$$\sigma^+ = \left\|e - v^+\right\| = \left\|\frac{e - \left(v^+\right)^2}{e + v^+}\right\|,$$

so the following inequality holds

$$\sigma^+ \leq \frac{\left\|e - v^2\right\| + \left\|v^2 - \left(v^+\right)^2\right\|}{1 + \min\left(v^+\right)}. \tag{41}$$

Using Lemma 3.2, the equality (17) and Lemma 3.1 we obtain

$$\|d_x^a d_s^a\| \leq \frac{\sqrt{2}}{4}\left\|d_x^a + d_s^a\right\|^2 = \sqrt{2}\|v\|^2 \leq \sqrt{2}n(1+\sigma)^2. \tag{42}$$

Now, from (39) we get

$$\left\|v^2 - \left(v^+\right)^2\right\| \leq \frac{\theta^2}{1-2\theta}\sqrt{2}n(1+\sigma)^2. \tag{43}$$

Observe that $\left\|e-v^2\right\| \leq \sigma + \|v(e-v)\|$, and Lemma 3.1 yields $\|v\|_\infty \leq 1+\sigma$, therefore

$$\left\|e-v^2\right\| \leq \sigma + \|v\|_\infty \|e-v\| \leq \sigma^2 + 2\sigma. \tag{44}$$

Finally, using the relations (40), (41), (43) and (44) we deduce

$$\sigma^+ \leq \frac{\sigma^2 + 2\sigma + \frac{\theta^2}{1-2\theta}\sqrt{2}n(1+\sigma)^2}{1+\sqrt{K(\sigma,\theta,n)}} \tag{45}$$

and this results in (38). Thus, the lemma is proved. ■

The next lemma is devoted to the proximity measure of the vectors obtained by a full Newton step. We use also the results of Lemma 2.2.

Lemma 3.5 *Let $x > 0$, $s > 0$, $\mu > 0$, and $0 < \tau < 1$ in such a way that $\sigma = \sigma(xs, \mu) \leq \tau$. Suppose that the vectors x^+ and s^+ are produced by a full Newton process, thus $x^+ = x + \Delta x$ and $s^+ = s + \Delta s$. Then*

$$\sigma(x^+s^+, \mu) \leq \omega(\tau). \tag{46}$$

Moreover, if $\tau \leq \frac{3}{4}$, then $\sigma(x^+s^+, \mu) < 6 - 4\sqrt{2}$ and $\sigma(x^+s^+, \mu) < \frac{\tau}{\sqrt{2}}$.

Proof: From Lemma 2.2 we obtain

$$\sigma(x^+s^+, \mu) \leq \omega(\sigma).$$

Furthermore, the function $\omega(t)$ is increasing for $0 \leq t < 1$, so the inequality (46) holds. If we assume $\tau \leq \frac{3}{4}$, then a simple calculus yields

$$\sigma(x^+s^+, \mu) \leq \omega(\tau) \leq \omega\left(\frac{3}{4}\right) = 1 - \frac{\sqrt{7}}{4} < 6 - 4\sqrt{2}.$$

For the last relation it is sufficient to prove that the inequality $\omega(\tau) < \frac{\tau}{\sqrt{2}}$ holds. We have $\tau \leq \frac{3}{4} < \frac{2\sqrt{2}}{3}$, thus $3\tau^2 < 2\sqrt{2}\tau$, therefore $\left(\sqrt{2}\tau - 1\right)^2 = 2\tau^2 - 2\sqrt{2}\tau + 1 < 1 - \tau^2$. We obtain

$$\frac{\tau}{1+\sqrt{1-\tau^2}} < \frac{1}{\sqrt{2}},$$

and from this inequality we get $\omega(\tau) < \frac{\tau}{\sqrt{2}}$. This proves the lemma. ■

In the following lemma we provide a sufficient condition, which guarantees that after an affine-scaling step the proximity measure will not exceed the proximity parameter.

Lemma 3.6 *Let τ be fixed such that $0 < \tau \leq \frac{3}{4}$ and let $\mu > 0$. Assume now that $x > 0$ and $s > 0$ are the vectors generated by the full Newton step of Algorithm 3.1. Let x^+ and s^+ be the vectors obtained after the affine-scaling step, thus $x^+ = x + \theta\Delta^a x$ and $s^+ = s + \theta\Delta^a s$. Denote $\mu^+ = (1-2\theta)\mu$*

and $\sigma^+ = \sigma(x^+s^+, \mu^+)$. *Finally, assume that the update parameter satisfies the* $0 < \theta < \frac{1}{2}$ *condition. Then the inequality*

$$\sigma^+ \leq \tau \tag{47}$$

holds if

$$\frac{\theta^2 n}{1-2\theta} \leq \Psi_\tau(\sigma), \tag{48}$$

Moreover, $\Psi_\tau(\sigma) > 0$ *and for each fixed* τ *the function* Ψ_τ *is decreasing on the closed interval* $[0,\ \omega(\tau)]$.

Proof: Let us introduce the notation

$$\psi_\tau(\sigma) = \sigma^2 - 2\sigma + \sqrt{2}\tau - \frac{\tau^2}{2}. \tag{49}$$

Thus

$$\Psi_\tau(\sigma) = \frac{\psi_\tau(\sigma)}{(1+\sigma)^2}.$$

Observe that for $0 < \tau \leq \frac{3}{4}$ we have $1 - \frac{\tau}{\sqrt{2}} \geq 1 - \frac{3\sqrt{2}}{8} > 0$. Therefore

$$\psi_\tau(\sigma) = (1-\sigma)^2 - \left(1 - \frac{\tau}{\sqrt{2}}\right)^2 < (1-\sigma)^2 \tag{50}$$

Suppose that the inequality (48) holds. Then, from (50) results $K(\sigma, \theta, n) > 0$, and we obtain that Lemma 3.4 can be applied. Because $x > 0$ and $s > 0$ are the vectors generated by a full Newton step of Algorithm 3.1, we deduce that there exists the vectors $\tilde{x} > 0$ and $\tilde{s} > 0$ in such a way that, from these vectors we obtain x and s by a full Newton step. Moreover, $\sigma(\tilde{x}\tilde{s}, \mu) \leq \tau$, and applying Lemma 3.5 for the vectors $\tilde{x}$ and $\tilde{s}$ we get the following inequalities

$$\sigma < \frac{\tau}{\sqrt{2}}, \tag{51}$$

$$\sigma < 6 - 4\sqrt{2}. \tag{52}$$

Lemma 3.4 implies that the inequality $\sigma^+ \leq \tau$ holds if $\Phi(\sigma, \theta, n) \leq \tau$. This can be written in the following form

$$\sqrt{2}K(\sigma, \theta, n) + \tau\sqrt{K(\sigma, \theta, n)} + \tau - \lambda(\sigma) \geq 0.$$

Denote $\kappa = \sqrt{K(\sigma, \theta, n)}$ and $\varrho(t) = \sqrt{2}t^2 + \tau t + \tau - \lambda(\sigma)$. Then (47) holds if $\varrho(\kappa) \geq 0$. The next issue is to determine lower and upper bounds of $\lambda(\sigma)$ and using these results to study the sign of the function ϱ. From (52) we get $0 \leq \sigma < 6 - 4\sqrt{2}$, therefore

$$\left(1 + \sqrt{2}\right)\sigma^2 \leq 2\left(\sqrt{2} - 1\right)\sigma,$$

thus $\lambda(\sigma) \leq \sqrt{2}$. Moreover, we have

$$\lambda(\sigma) = \left(1+\sqrt{2}\right)\left(\sigma - \left(3-2\sqrt{2}\right)\right)^2 + 7 - 4\sqrt{2} \;\geq\; 7 - 4\sqrt{2},$$

resulting in

$$7 - 4\sqrt{2} \leq \lambda(\sigma) \leq \sqrt{2}. \tag{53}$$

Let

$$\Delta_{\sigma,\tau} = \tau^2 - 4\sqrt{2}\,\tau + 4\sqrt{2}\,\lambda(\sigma). \tag{54}$$

The roots of the $\varrho(t) = 0$ equation are

$$t_1 = \frac{-\tau - \sqrt{\Delta_{\sigma,\tau}}}{2\sqrt{2}}, \quad t_2 = \frac{-\tau + \sqrt{\Delta_{\sigma,\tau}}}{2\sqrt{2}}.$$

Since $0 < \tau \leq \frac{3}{4}$, from (53) we get $\tau < \lambda(\sigma)$ thus

$$\Delta_{\sigma,\tau} > \tau^2 > 0.$$

We obtain the inequalities $t_1 < 0$ and $t_2 > 0$, and this means that if $\kappa \geq t_2$, then $\varrho(\kappa) \geq 0$ holds, and (47) is satisfied. Using (53) from (54) we get

$$\sqrt{\Delta_{\sigma,\tau}} \leq \sqrt{\tau^2 - 4\sqrt{2}\,\tau + 8} = \sqrt{\left(2\sqrt{2} - \tau\right)^2} = 2\sqrt{2} - \tau,$$

therefore

$$t_2 \leq \frac{2\sqrt{2} - 2\tau}{2\sqrt{2}} = 1 - \frac{\tau}{\sqrt{2}}.$$

We deduce that the inequality (47) holds if $\kappa \geq 1 - \frac{\tau}{\sqrt{2}}$, and this can be written in the form

$$K(\sigma,\theta,n) \geq \left(1 - \frac{\tau}{\sqrt{2}}\right)^2.$$

Using (29) the inequality (48) follows. This proves the first assertion of the lemma. Now, from Lemma 3.5 the inequality $\sigma \leq \omega(\tau)$ holds, therefore we are going to study the functions ψ_τ and Ψ_τ on the $[0,\ \omega(\tau)]$ interval. Observe, that from (50) we have

$$\psi_\tau(\sigma) = \left(\frac{\tau}{\sqrt{2}} - \sigma\right)\left(2 - \frac{\tau}{\sqrt{2}} - \sigma\right).$$

Since

$$2 - \frac{\tau}{\sqrt{2}} - \sigma \;>\; 2\left(1 - \frac{\tau}{\sqrt{2}}\right) > 0$$

we obtain the inequality $\psi_\tau(\sigma) > 0$, which results in $\Psi_\tau(\sigma) > 0$. Taking the derivative of the function ψ_τ we get

$$\left(\psi_\tau\right)'(\sigma) = 2\sigma - 2 < 0.$$

Thus, the function ψ_τ is positive and is decreasing on the interval $[0,\ \omega(\tau)]$. The same is true for the function $\frac{1}{(1+\sigma)^2}$ and this implies the last result of the lemma. ■

In Lemma 3.7 we investigate how will be modified the duality gap after an affine-scaling step.

Lemma 3.7 *Let $x > 0$, $s > 0$ and $\mu > 0$ such that $\sigma = \sigma(xs, \mu) < 1$ and $0 < \theta < \frac{1}{2}$. Assume that x^+ and s^+ are the vectors obtained after the affine-scaling step of Algorithm 3.1. Then the following inequality holds*

$$\left(x^+\right)^T s^+ \leq (1 - 2\theta + 2\theta^2) x^T s \leq (1 - \theta) x^T s.$$

Proof: Substitute $\beta = 1$ in the relation (33). Thus

$$\left(x^+\right)^T s^+ = e^T \left(x^+ s^+\right) = (1 - 2\theta) e^T (xs) + \mu\theta^2 e^T \left(d_x^a d_s^a\right).$$

Since (17) we have

$$d_x^a d_s^a = 2v^2 - \frac{(d_x^a)^2 + (d_s^a)^2}{2}$$

and this leads to

$$e^T (d_x^a d_s^a) = 2e^T \frac{xs}{\mu} - \frac{\|d_x^a\|^2 + \|d_s^a\|^2}{2} \leq \frac{2}{\mu} x^T s.$$

We obtain $\mu\theta^2 e^T \left(d_x^a d_s^a\right) \leq 2\theta^2 x^T s$ and this implies the first inequality of the lemma. Now observe that for $0 < \theta < \frac{1}{2}$ the inequality

$$1 - 2\theta + 2\theta^2 \leq 1 - \theta$$

holds, thus we get the second inequality. This proves the lemma. ■

Lemma 3.8 is devoted to finding an upper bound for the duality gap after a whole iteration (full Newton step followed by an affine-scaling step).

Lemma 3.8 *Let $x > 0$, $s > 0$ and $\mu > 0$ such that $\sigma = \sigma(xs, \mu) < 1$ and $0 < \theta < \frac{1}{2}$. Assume that the vectors x^+ and s^+ are obtained after an iteration of Algorithm 3.1. Moreover, let $\mu^+ = (1 - 2\theta)\mu$. Then the relation*

$$\left(x^+\right)^T s^+ \leq (1 - \theta) n\mu < \frac{n\mu^+}{1 - 2\theta}$$

is satisfied.

Proof: Let $\bar{x}$ and $\bar{s}$ be the vectors obtained by a full Newton step. Using Lemma 2.3 we get $\bar{x}^T \bar{s} \leq n\mu$. Furthermore, from Lemma 3.7 and $1 - \theta < 1$ we obtain

$$\left(x^+\right)^T s^+ = (1 - \theta) \bar{x}^T \bar{s} \leq (1 - \theta) n\mu < \frac{n\mu^+}{1 - 2\theta}.$$

This completes the proof. ■

In the following lemma we analyse the question of the bound on the number of iterations performed by the algorithm. We assume that we would like to approximate the optimal solution with a given precision.

Lemma 3.9 *Let x^k and s^k be the vectors generated by Algorithm 3.1 after k iterations, where $k > 1$. Then for each k satisfying the condition*

$$k \geq 1 + \left\lceil \frac{1}{2\theta} \log \frac{(x^0)^T s^0}{\epsilon} \right\rceil$$

the inequality $(x^k)^T s^k < \epsilon$ holds.

Proof: Let μ^k be the value of μ after k iterations. From Lemma 3.8 results

$$(x^k)^T s^k < \frac{n\mu^k}{1-2\theta} = (1-2\theta)^{k-1} n\mu^0 = (1-2\theta)^{k-1}(x^0)^T s^0.$$

Thus the inequality $(x^k)^T s^k < \epsilon$ holds if

$$(1-2\theta)^{k-1}(x^0)^T s^0 \leq \epsilon.$$

Taking logarithms we obtain

$$(k-1)\log(1-2\theta) + \log((x^0)^T s^0) \leq \log \epsilon$$

and using the relation $-\log(1-2\theta) \geq 2\theta$ we conclude that this inequality is satisfied if

$$2\theta(k-1) \geq \log((x^0)^T s^0) - \log \epsilon = \log \frac{(x^0)^T s^0}{\epsilon}.$$

This implies the lemma. ■

In the following theorem we give a sufficient condition, which guarantees that the algorithm will be well defined. Furthermore, we provide an upper bound for the number of iterations.

Theorem 3.10 *Let $0 < \tau \leq \frac{3}{4}$ and $0 < \theta < \frac{1}{2}$. If*

$$\frac{\theta^2 n}{1-2\theta} \leq \Psi_\tau(\omega(\tau)), \tag{55}$$

then Algorithm 3.1 is well defined and performs at most

$$1 + \left\lceil \frac{1}{2\theta} \log \frac{(x^0)^T s^0}{\epsilon} \right\rceil \tag{56}$$

iterations. The generated vectors satisfy the $x^T s < \epsilon$ inequality.

Proof: As in the proof of Lemma 3.6 let $\tilde{x}$ and $\tilde{s}$ be the vectors at the begining of a new iterate. Furthermore, let x and s be the vectors after the full Newton step. Finaly, denote by x^+ and s^+ the vectors obtained by the affine-scaling step. We have to prove that the interior point condition holds every time a

new iterate begins and the proximity measure is not greater than τ. This assertion will be true if we have the following one. Suppose that $\tilde{x} > 0,\ \tilde{s} > 0$ and $\sigma(\tilde{x}\tilde{s}, \mu) \leq \tau$ then we have to prove that $x^+ > 0$ and $s^+ > 0$ and for the proximity measure we have the inequality $\sigma(x^+s^+, \mu^+) \leq \tau$, where μ^+ denotes the value of the parameter μ at the end of the iteration.

From Lemma 2.1 results $x > 0$ and $s > 0$. Using these relations, from Lemma 3.3 we obtain that the inequalities $x^+ > 0$ and $s^+ > 0$ are satisfied if $K(\sigma, \theta, n) > 0$. Moreover, using Lemma 3.6 the inequality $\sigma(x^+s^+, \mu^+) \leq \tau$ holds if we have the relation (48) and from (50) we deduce that in this case the inequality $K(\sigma, \theta, n) > 0$ is also satisfied.

This means that it is sufficient to prove that the inequality (48) holds. From Lemma 3.5 we deduce

$$\sigma = \sigma(xs, \mu) \leq \omega(\tau).$$

Since the function Ψ_τ is decreasing, we conclude that the inequality (48) is satisfied if the relation (55) holds. Lemma 3.9 implies the upper bound for the number of iterations. This completes the proof. ■

In the next theorem we prove that Algorithm 3.1 is well defined for the default values. From the upper bound on the number of iterations we conclude that this predictor-corrector type algorithm finds an ϵ-solution in polynomial time.

Theorem 3.11 *Let $\tau = \frac{5}{13}$ and $\theta = \frac{1}{3\sqrt{n}}$, where $n \geq 2$. Then Algorithm 3.1 is well defined and requires no more than*

$$\left\lceil 3\sqrt{n} \log \frac{(x^0)^T s^0}{\epsilon} \right\rceil$$

iterations. For the vectors obtained we have the $x^T s \leq \epsilon$ inequality. ■

4. Conclusion

We have introduced a new predictor-corrector algorithm for solving LCCO problems. The method of finding a new search direction is based on an equivalent algebraic transformation of the centering equation from the system, which defines the central path. Polynomial complexity is proved, and the best known iteration bound for small-update methods is obtained.

References

[1] M. Achache. A weighted-path-following method for the linear complementarity problem. *Studia Universitatis Babeş-Bolyai, Series Informatica*, 49(1):61–73, 2004.

[2] M. Achache. A new primal-dual path-following method for convex quadratic programming. *Computational & Applied Mathematics*, 25(1):97–110, 2006.

[3] Zs. Darvay. A new algorithm for solving self-dual linear optimization problems. *Studia Universitatis Babeş-Bolyai, Series Informatica*, 47(1):15–26, 2002.
[4] Zs. Darvay. A weighted-path-following method for linear optimization. *Studia Universitatis Babeş-Bolyai, Series Informatica*, 47(2):3–12, 2002.
[5] Zs. Darvay. New interior point algorithms in linear programming. *Advanced Modeling and Optimization*, 5(1):51–92, 2003.
[6] Zs. Darvay. A new predictor-corrector algorithm for linear programming. *Alkalmazott matematikai lapok*, 22:135–161, 2005. (In hungarian).
[7] N.K. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.
[8] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4):575–601, 1992.
[9] S. Pan, X. Li, and S. He. An infeasible primal-dual interior-point algorithm for linear programs based on logarithmic equivalent transformation. *Journal of Mathematical Analysis and Applications*, 314(2):644–660, 2006.
[10] J. Peng, C. Roos, and T. Terlaky. *Self-Regular Functions: a New Paradigm for Primal-Dual Interior-Point Methods*. Princeton University Press, 2002.
[11] C. Roos, T. Terlaky, and J.-Ph. Vial. *Theory and Algorithms for Linear Optimization. An Interior Approach*. John Wiley & Sons, Chichester, UK, 1997.
[12] L. Tuncel and M.J. Todd. On the interplay among entropy, variable metrics and potential functions in interior-point algorithms. *Computational Optimization and Applications*, 8:5–19, 1997.
[13] G.Q. Wang and Y.Q. Bai. A new primal-dual path-following interior-point algorithm for semidefinite programming. *Journal of Mathematical Analysis and Applications*, 353:339–349, 2009.
[14] G.Q. Wang and Y.Q. Bai. A primal-dual interior-point algorithm for second-order cone optimization with full Nesterov-Todd step. *Applied Mathematics and Computation*, 215:1047–1061, 2009.
[15] G.Q. Wang, X.Z. Cai, and Y.J. Yue. A new polynomial interior-point algorithm for monotone mixed linear complementarity problem. In *ICNC '08: Proceedings of the 2008 Fourth International Conference on Natural Computation*, pages 450–454, Washington, DC, USA, 2008. IEEE Computer Society.
[16] G.Q. Wang, Y.J. Yue, and X.Z. Cai. A weighted-path-following method for monotone horizontal linear complementarity problem. In B.-y. Cao, C.-y. Zhang, and T.-f. Li, editors, *Fuzzy Information and Engineering*, volume 54 of *Advances in Soft Computing*, pages 479–487. Springer-Verlag, Berlin, Heidelberg, 2009.
[17] M. Zhang, Y.Q. Bai, and G.Q. Wang. A new primal-dual path-following interior-point algorithm for linearly constrained convex optimization. *Journal of Shanghai University*, 12(6):475–480, 2008.

Department of Computer Science, Babeş-Bolyai University, 1 M. Kogălniceanu St., 400084 Cluj-Napoca, Romania
E-mail address: darvay@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

A NOTE ON A PROBLEM OF ŢÂMBULEA

ZOLTÁN KÁSA

Abstract. In [6] and [7] L. Ţâmbulea studied the number of positive integers sequences $(s_1, s_2, \ldots, s_{2n+1})$ with the properties: $s_1 = s_{2n+1} = 1$ and $|s_i - s_{i+1}| = 1$ for each i between 1 and $2n$. In this note we discuss the algorithm to code this sequences by Dyck words and several related problems.

In [6] Ţâmbulea studied the problem of the number of elements in the set

$$S_n = \big\{(s_1, s_2, \ldots, s_{2n+1}) \mid s_1 = s_{2n+1} = 1, |s_i - s_{i+1}| = 1 \text{ for } i = 1, \ldots, 2n, \text{ and } s_j \in \mathbf{N}^* \text{ for } j = 2, 3, \ldots, 2n\big\}$$

and proved that this is equal to the Catalan number $C_n = \frac{1}{n+1}\binom{2n}{n}$. In [7] was proved that this number is equal to the number of possibilities to divide a convex $(n+2)$-gon into triangles using noncrossing diagonals. The equivalence with the problem of the number of binary trees with n nodes was proved too.

Stanley collected in [4, 5] more than a hundred problems related to Catalan numbers. This problem is not listed in [4] nor in [5]. In [5] there is a similar problem:

(p^5) Sequences $a_1, \ldots, a_{2n}$ of nonnegative integers with $a_1 = 1, a_{2n} = 0$ and $a_i - a_{i-1} = \pm 1$.

Using the idea of coding elements of sets whose cardinality is a Catalan number, described in [1, 2], each sequence from S_n can be coded as follows. If $s_i - s_{i+1} = 1$ let us put a 0, and if $s_i - s_{i+1} = -1$ let us put a 1 in the code. E. g. the sequence 1,2,3,2,1 can be coded as 0011, and the sequence 1,2,1,2,1 as 0101. It is easy to see that such a code is a Dyck word, which is a binary word with equal number of 0s and 1s, the number of 1s never exceeding the number of 0s in each position from left to right. It is well-known that the number of $2n$-length Dyck words is the Catalan number C_n (see e.g. [8]). In the following algorithm description we consider correct inputs only.

Received by the editors: December, 2009.

2010 *Mathematics Subject Classification.* 05A99, 68R05, 68P30.

1998 *CR Categories and Descriptors.* G.2.1 [**Combinatorics**]: Subtopic – *Combinatorial algorithms*.

Key words and phrases. sequences of integers, Catalan numbers, Dyck words.

```
ENCODING1(s_1, s_2, ..., s_{2n+1})
j ← 0
for i ← 1 to 2n
    do j ← j + 1
       if s_{i+1} − s_i = 1
          then d_j ← 0
          else d_j ← 1
return d_1, d_2, ..., d_{2n}
```

```
DECODING1(d_1, d_2, ..., d_{2n})
s_1 ← 1; j ← 1
for i ← 1 to 2n
    do j ← j + 1
       if d_i = 0
          then s_j ← s_{j−1} + 1
          else s_j ← s_{j−1} − 1
return s_1, s_2, ..., s_{2n+1}
```

These two algorithms prove that between the set S_n and the set of $2n$-length Dyck words there is a bijection.

For the similar problem the codification can be made as follows. Consider the elements of sequence backwards. If $a_{i-1} - a_i = 1$ let us put a 0, else an 1 in the output word. At the end let us add a 1. For example, in the case of $n = 2$, we have two such sequences 1,2,1,0 and 1,0,1,0, the corresponding codes are: 0011 and 0101. Each code is a Dyck word. In the following algorithms we omitted the input verification.

```
ENCODING2(a_1, a_2, ..., a_{2n})
j ← 0
for i ← 2n downto 2
    do j ← j + 1
       if a_{i−1} − a_i = 1
          then d_j ← 0
          else d_j ← 1
d_{2n} ← 1
return d_1, d_2, ..., d_{2n}
```

```
DECODING2(d_1, d_2, ..., d_{2n})
a_{2n} ← 0;
j ← 2n
for i ← 1 to 2n
    do j ← j − 1
       if d_i = 0
          then a_j ← a_{j+1} + 1
          else a_j ← a_{j+1} − 1
return a_1, a_2, ..., a_{2n}
```

For a list of codifications of elements related to Catalan numbers see [3].

Another problem given in [7], which can be included in the set of problems being an interpretation of Catalan numbers, but not listed in [4, 5] is the following. A labelled binary tree is contructed by the rules:

• the root (which has the level 1) is labelled with 1,

• a node with the label i has a left subtree with the root labelled with $i-1$ and a right subtree with the root labelled with $i+1$,

• any subtree having the root labelled with 0 is omitted from tree.

The number of nodes labelled with 1 at level $2n+1$ is the Catalan number C_n. This can be proved easily by coding the paths between the root and nodes with label 1 at the level $2n+1$. Going from root on these paths, let us put 0 in code for a right edge and 1 for a left edge. In the Figure 1 for $n = 2$, we have two such paths with the codes: 0101 and 0011.

The labels of the nodes of such paths from root to the leaf correspond to the elements of sequence S_n, while the edges in the same order represent the bits of

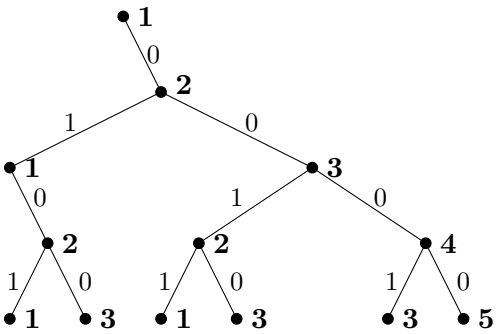


FIGURE 1. The tree corresponding to the set S_2. The two paths from the root to the nodes labelled with 1 at the level 5 are: 1,2,1,2,1 coded as 0101 and 1,2,3,2,1 coded as 0011.

the corresponding code. Such trees can be easily generated, so the generation of elements of S_n and the corresponding Dyck words is a straightforward task.

In future these two problems maybe will get into the list [5] of problems being an interpretation of Catalan numbers.

REFERENCES

[1] Bege, A., Kása, Z., Coding object related to Catalan numbers, Studia Univ. Babes-Bolyai, Informatica, 46, 1 (2001) 31-40.

[2] Kása, Z., Generating and ranking of Dyck words, Acta Univ. Sapientiae, Informatica, 1, 1 (2009) 109-118.

[3] Kása, Z., Coding elements related to Catalan numbers by Dyck words, version January 19, 2009. http://www.ms.sapientia.ro/∼kasa/CodingDyck.pdf

[4] Stanley, R. P., Enumerative combinatorics, Vol. 2, Cambridge University Press, 1999.

[5] Stanley, R. P., Catalan addendum, version August 11, 2009. http://www-math.mit.edu/∼rstan/ec/catadd.pdf

[6] Ţâmbulea, L., Data organization according to the property of consecutive retrieval, Mathematica, Revue d'analyse numérique et de theorie de l'approximation, 9, 2 (1980) 269–281.

[7] Ţâmbulea, L., Binary trees, an Euler's problem and finite sequences of numbers, Studia Univ. Babeş-Bolyai, Mathematica, 35, 3 (1990) 84–94.

[8] Vilenkin, N, Ia., Combinatorial mathematics for recreation, 1972 (translated from Russian).

SAPIENTIA HUNGARIAN UNIVERSITY OF TRANSYLVANIA, RO 400112 CLUJ-NAPOCA, STR. MATEI CORVIN 4, DEPARTMENT OF MATHEMATICS AND INFORMATICS, TÂRGU MUREŞ
E-mail address: kasa@ms.sapientia.ro

IN MEMORIAM: PROFESSOR EMIL MUNTEANU

Professor Emil Muntean, age 76, one of the founders of the computer science school in Cluj-Napoca, died Thursday, November 29, 2009, in Cluj-Napoca following a courageous battle with illness.

He was born on July 31, 1933, in Magura, Hunedoara County. After finishing secondary school in 1952, he studied at the University of Cluj-Napoca. He graduated in Mathematics from Cluj University in 1957. Still being in the fifth year, he was named at the Computing Institut of Academy. Since then, the entire activity of Professor Munteanu is connected with computers. He worked to the construction of MARICA (1959), a Romanian computer built from relays, and to the construction (in 1961) of DACICC-1, the first Romanian transistor-based computer.Then (1967-1969) he worked to the complex project of building DACICC-200. Also, he had contributed to the realisation of some programs.

He obtained his Ph.D. from the Saint Petersburg University, U.S.S.R., in 1964. In 1968 he became the Head of the newly Institut of Computer Technology (ITC). As the Head of the ITC in a pioniering period, he has directed with much competence and inspiration the research activity, to design and to implement high-level software products. He was really a very good organizer.

In 1990 he became full professor at our Faculty, Department of Computer Science, but his teaching activity started long time ago. He used to teach the students of Mathematics various computers related subjects. In the last four years he taught courses in Expert Systems, and Computers Networks. His courses were held at a high scientific and pedagogical level.

From 2000, he became full professor at the "Dimitrie Cantemir" University, Faculty of Economical Science, the Cluj branch. In the same time, professor Muntean had classes of History of Computer Science at our faculty.

There are eight computer scientists who owe their Ph. Degrees to their supervisor, professor Emil Munteanu. In the last years he became interested in spreading computer science knowledge, i.e. he is today a known editor of books in this area. He was the father of Microinformatica Publishers, and in the last years he created Promedia Publishers.

Professor Emil Muntean was a distingushed pedagogue, very appreciated by his students. It was, as well, a pleasure for all of us to have such a generous colleague.