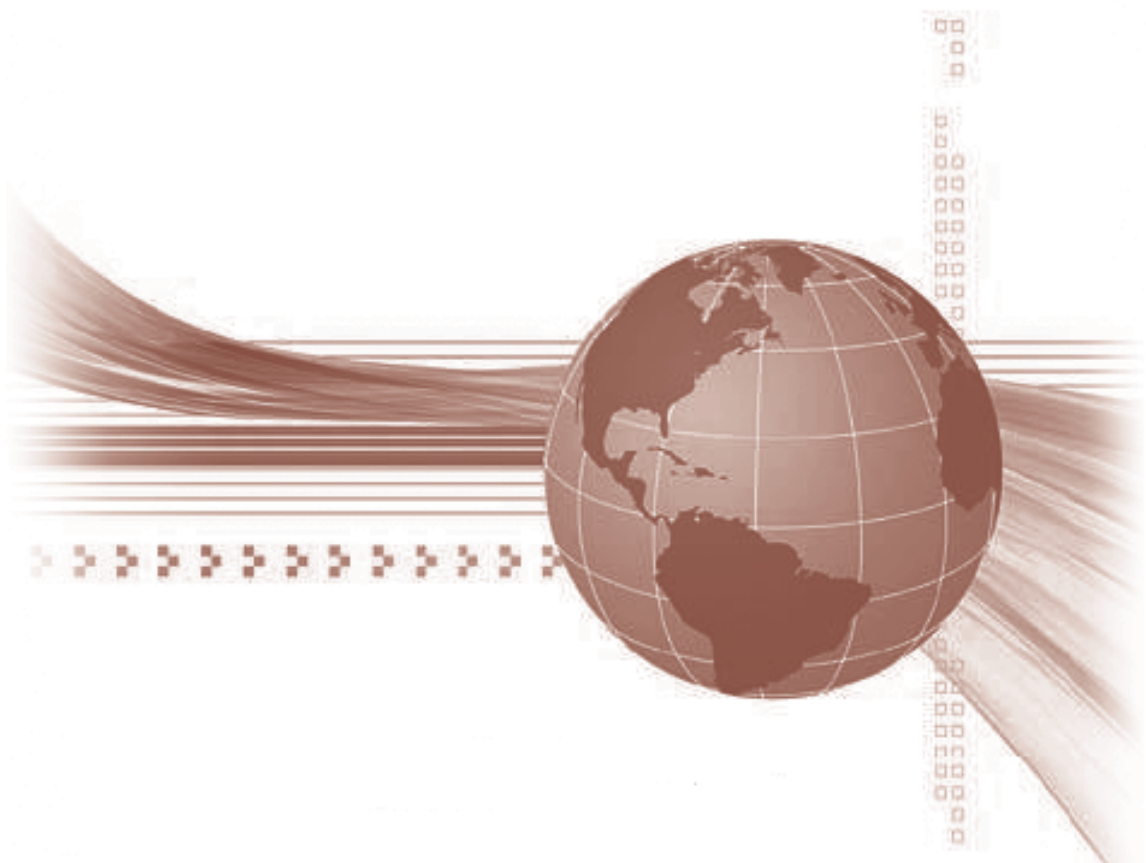
STUDIA UNIVERSITATIS
BABEŞ-BOLYAI



INFORMATICA

1/2013

STUDIA

UNIVERSITATIS BABEŞ-BOLYAI
INFORMATICA

No. 1/2013

January - March

EDITORIAL BOARD

YEAR **Volume 58 (LVIII) 2013**
MONTH **MARCH**
ISSUE **1**

S T U D I A
UNIVERSITATIS BABEŞ-BOLYAI
INFORMATICA

1

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

SUMAR – CONTENTS – SOMMAIRE

G. Coman, M. Frenţiu, *Academician Professor Dimitrie D. Stancu at His 85-th Anniversary* 5

M.R. Farahani, *A Good Drawing of Complete Bipartite Graph $K_{9,9}$, Whose Crossing Number Holds Zarankiewicz Conjectures* 21

M. Antal, *On the use of Elo rating for adaptive assessment* 29

Zs. Marian, *A Study on Association Rule Mining Based Software Design Defect Detection* 42

B. Pop, *Building an Automated Task Delegation Algorithm for Project Management and Deploying it as SaaS* 58

I.V. Pop, *A Content Ontology Design Pattern for Software Metrics* 71

T. Ban, *Fuzzy Computing for Complexity Level of Evaluation Tests* 81

F. Crăciun, S. Motogna, B. Pârv, *Aspect Towards a Region-based Calculus for Energy-Aware Programming* 93

V. Varga, H. Greblă, A. Andreica, *Decision Support System for Babes-Bolyai University* 102

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVIII**, Number 1, 2013

ACADEMICIAN PROFESSOR DIMITRIE D. STANCU AT HIS 85-TH ANNIVERSARY

GHEORGHE COMAN AND MILITON FRENŢIU

Professor **Dimitrie D. Stancu** was born on February 11, 1927, in a very poor farmer family from the village Călacea, situated not far from Timişoara, the capital of Banat, in the south-west part of Romania. Soon he remained orphan, and he was forced to work as a shepherd engaged by a rich family. Consequently he was able to go to primary school only when he was nine years old. His oldest brother, who worked in a painter workshop in Arad city, brought him in Arad at the "Regina Maria" orphanage in 1937, where he was included also in the elementary school. Then he entered at the Gymnasium from the district "New Arad". In the period 1943–1947 he studied at the prestigious Lyceum "Moise Nicoară" from Arad, the same place where other two academicians, Tiberiu Popoviciu and Caius Iacob, studied some years earlier.

In 1947 he began a four-year study at the "Victor Babeş" University from Cluj, the capital of the Romanian province "Transylvania". He was a very good student, remarked by his teachers, consequently in his third year was named teaching assistant. In 1951, after his graduation, he was appointed assistant at the Department of Mathematical Analysis, University of Cluj. During his studies he was under the influence of professor Tiberiu Popoviciu, a great master of Numerical Analysis and Approximation Theory, who has stimulated him to do research work in mathematics, and was his PhD supervisor. He obtained the Ph.D. in mathematics in 1956. In a normal succession he advanced up to the rank of full professor in 1969.

In the academic year 1961–1962 Professor D. D. Stancu had an excellent opportunity to be a fellowship at the Numerical Analysis Department, University of Wisconsin, Madison, conducted by late professor Preston C. Hammer. Returning from America, he was named deputy dean at the Faculty of Mathematics of "Babeş-Bolyai" University from Cluj. And in this period he became closed to the future Computer Science section of our University. He taught for the first time in Romania, the programming language Fortran. Also, he created a new chair, Numerical and Statistical Calculus, and was its chief.

Here he gathered a group of teachers interested in this domain who were implied later in teaching the students of computer science section. The group of theachers interested in Computer Science has aroused in this department.

Professor Stancu has taught several courses at his University: MathematicalAnalysis, Numerical Analysis, Approximation Theory, Informatics, Probability Theory and Mathematical Statistics, as well as Constructive Theory of Functions.

Professor D.D. Stancu has important mathematical contributions in various areas of numerical analysis, approximation theory, numerical differentiation, orthogonal polynomials, numerical quadratures and cubature, Taylor-type expansions, approximation by linear positive operators, representations of remainders in linear approximation formulas, probabilistic methods for construction and investigation of linear operators of approximation, use of interpolation and calculus of finite differences in probability theory and mathematical statistics. After 1959, he became interested in approximation theory by means of sequences of linear positive operators. His results are well appreciated in the scientific community. There are more than 50 papers published in different mathematical journals, containing in their titles the name of D. D. Stancu.

Professor Stancu has participated at many international scientific meetings of mathematics, organized in Romania (Cluj, Bucharest, Iaşi, Timişoara), Germany (Stuttgart, Hannover, Hamburg, Goettingen, Dortmund, Munster, Siegen, Berlin,Wurzburg and Oberwolfach), Italy (Roma, Napoli and Potenza), England (Lancaster and Durham), Hungary (Budapest), France (Paris), Bulgaria (Sofia and Varna), Poland (Warsaw), Czech Republic (Brno) and USA (Nashville, S.C.-Columbus, OH, Columbia, S.C.). Professor D. D. Stancu has participated at different events in USA, organized by the American Mathematical Society. He has presented contributed papers at several regional meetings of this Society, from Milwaukee, Chicago and New York. Also, on the base of an official invitation from the Society for Industrial and Applied Mathematics (SIAM), at a "SIAM Symposium on Approximation" organized in Gatlinburg, Tennessee (October 21–26, 1963). In May 2000 he was invited to participate at the International Symposium "Trends in Approximation Theory" dedicated the 60th birthday of Professor L. L. Schumaker, held in Nashville, Texas, where he presented a paper in collaboration with professor Wanzer Drane, from Columbia, S.C.

Since 1961 he is a member of the American Mathematical Society and a reviewer of the international journal "Mathematical Reviews". Also, he is a member of the German society: "Gesellschaft für Angewandte Mathematick und Mechanik" as well as a reviewer of the international journal "Zentralblatt für Mathematik". For many years he was a member of the Editorial Board

of the Italian mathematical journal "Calcolo", published now by "Springer-Verlag", in Berlin.

In Romania he is the Editor in Chief of the journal published by the Romanian Academy: "Revue d'Analyse Numerique et de Theorie de l'Approximation". In 1996 Professor D. D. Stancu has organized at the University "Babeş-Bolyai", Cluj-Napoca, an "International Conference on Approximation and Optimization", in conjunction with the Second European Congress of Mathematics, held in Budapest, where participated around 150 mathematicians from 20 countries around the world. The Proceedings of this ICAOR conference were published in two volumes, having the title: "Approximation and Optimization", by Transilvania Press, Cluj-Napoca, Romania, in 1997.

In May 9–11, 2002, there was organized by "Babeş-Bolyai" University, Cluj-Napoca, the "International Symposium on Numerical Analysis and Approximation Theory", dedicated to the 75th anniversary of Professor D. D. Stancu. In the period 5–8 July, 2006 there was organized in Cluj-Napoca an "International Conference on Numerical Analysis and Approximation Theory". Professor D. D. Stancu was an honorary chair of this Conference.

He had a large numbers of doctoral students from Romania, Germany and Vietnam. Here are the persons that own their PhD to professor D.D.Stancu: **Grigor Moldovan** (1971), **Stefan Maruster** (1974), Ioan Gansca (1975), Trung Du Hoang (1976), Ioan Mihoc (1976), Ştefan Cobzaş (1978), Maria Micula (1978), Octavian Dogaru (1979), Dumitru Acu (1980), Aurel Gaidici (1980), Horst Kramer (1980), Maria Mihoc (1981), Ioan Gavrea (1982), Petru Blaga (1983), Adrian Diaconu (1983), Crăciun Iancu (1983), Ioan Şerb (1983), Elvira Kramer (1984), Constantin Manole (1984), Traian Mureşan (1984), **Zoltan Kasa** (1985), **Leon Ţâmbulea** (1985), Cristina Cismaşiu (1986), Tiberiu Vladislav (1986), Maria Dumitrescu (1989), **Teodor Toadere** (1989), **Dumitru Dumitrescu** (1990), Ioana Chiorean (1994), Octavian Agratini (1995), Alexandra Ciupa (1995), Reiner Dünnbeil (1996), Alexandru Bărbosu (1997), Vasile Miheşan (1997), Emil Popa (1998), Gabriela Vlaic (1998), Emil Cătinaş (1999), Daria Dumitraş (1999), Silvia Toader (1999), Andrei Vernescu (2000), Maria Crăciun (2005). Those marked are computer science teachers and this aspect constitutes the third connection of Professor D.D.Stancu to Computer Science.

In 1995, for his scientific achievements, the "Lucian Blaga" University from Sibiu has awarded him the scientific title of Doctor Honoris Causa. Few years ago the "North University" of Baia Mare, from which he had several doctoral students, has distinguished him with the same title.

In 1999 professor D. D. Stancu was elected as **Honorary Member of the Romanian Academy**.

Editing this article we can state that the intensive work and his important results obtained in Numerical Analysis, Approximation Theory and Probability Theory has brought him recognition and appreciation in our country and abroad. The important academic activity of professor D. D. Stancu culminated in three fundamentals books on Numerical Analysis [108, 109, 113] "Analiză Numerică şi Teoria Aproximării" (1438 pages), written under his supervision.

Now, on celebrating his 85-th birthday, we wish to D.D. Stancu and his family "Many Happy Returns of the Day", and a long life in health and happiness.

Annex A: Selected Publications

[1] Stancu, D.D., *Contribution to the partial numerical differentiation of functions of two and several variables* (Romanian), Bul. Sti. al Acad. R.P. Române, Secţ. Mat. Fiz., 8, 1956, 733-763.

[2] Stancu, D.D., *A study of the polynomial interpolation of functions of several variables, with application to the numerical differentiation and integration; methods for evaluating the remainders*, PhD Thesis (Romanian), University of Cluj, 1956, 192 pag.

[3] Stancu, D.D., *Consideration on the polynomial interpolation formulas for functions of several variables*(Romanian), Bul. Univ. Babeş-Bolyai Cluj, 1, 1957, 43-82.

[4] Stancu, D.D., *Contributions to the numerical integration of functions of several variables* (Romanian), Acad. R. P. Române, Fil. Cluj, Stud. Cerc. Mat., 8, 1957, 75-101.

[5] Stancu, D.D., *Generalization of some interpolation formulas for functions of several variables and certain considerations on the numerical integration formula of Gauss* (Romanian), Bul. Sti. al Acad. R.P. Române, Secţ. Sti. Mat. Fiz., 9, 1957, 287-313.

[6] Stancu, D.D., *A generalization of the Gauss-Christoffel quadrature formula* (Romanian), Bul. Sti. al Acad. R.P. Române, Secţ. Sti. Mat. Fiz., 8, 1957, no.1, 1-18.

[7] Stancu, D.D., *The generalization of certain interpolation formulae for the functions of many variables* (Romanian), Bul. Inst. Politeh. Iaşi, 3, 1957, no.1-2, 31-38.

[8] Stancu, D.D., *Sur une classe de polinomes orthogonaux et sur des formules générales de quadrature à nombre minimum de termes.* (French), Bull. Math. Soc. Sci. Mat. Phys., R. P. Roumaine, 1 (49), 1957, 479-498.

[9] Stancu, D.D., *On the Hermite interpolation formula and on some of its applications* (Romanian), Stud. Cerc. Mat. al Acad. R. P. Române, Fil. Cluj, 8, 1957, 339-355.

[10] Stancu, D.D., *On some general numerical integration formulas* (Romanian), Stud. Cerc. Mat., 9, 1958, 209-216.

[11] Stancu, D.D., *On numerical integration of functions of two variables* (Romanian), Stud. Cerc. Sti. Mat. Al Acad. R. P. Române, Fil. Iaşi, 9, 1958, no.1, 5-21.

[12] Stancu, D.D., *A method for constructing quadrature formulas of high degree of exactness* (Romanian), Com. Acad. R. P. Române, 8, 1958, 349-358.

[13] Stancu, D.D., *On the Gaussian quadrature formulas*, Studia Univ. Babeş-Bolyai, Cluj, 1, 1958, 71-84, (Romanian, Russian and French summaries).

[14] Stancu, D.D., *A method for constructing cubature formulas for functions of two variables* (Romanian), Stud. Cerc. Mat. al Acad. R. P. Române, Fil. Cluj, 9, 1958, 351-369.

[15] Stancu, D.D., *Some Taylor developments for functions of several variables* (Russian), Rev. Math. Pures Appl., 4, 1959, 249-265.

[16] Stancu, D.D., *On the approximation by Bernstein type polynomials of functions of two variables* (Romanian), Com. Acad. R. P. Române, 9, 1959, 773-777.

[17] Stancu, D.D., *On some general quadrature formulas of type Gauss-Christoffel* (French), Mathematica (Cluj), 1 (24), 1959, no. 1, 167-182.

[18] Stancu, D.D., *On a proof of the Weierstrass theorem* (Romanian), Bul. Inst. Politehn. Iaşi, 5 (9), 1959, no. 1-2, 47-50.

[19] Stancu, D.D., *The integral expression of the remainder in a formula of Taylor type for functions of two variables* (Romanian), Stud. Cerc. Mat., Acad. R. P.Române, Fil. Cluj, 11 , 1960, 177-183.

[20] Stancu, D.D., *On some Bernstein type polynomials* (Romanian), Acad. R. P. Române, Fil. Iaşi, Stud. Cerc. Sti. Mat., 11, 1960, 221-233.

[21] Stancu, D.D., *On the approximation of functions of two variables by polynomials of Berstein type. Some asymptotic estimation* (Romanian), Stud. Cerc. Mat., Acad. R. P. Române, Fil. Cluj, 11, 1960, 171-176.

[22] Stancu, D.D., *Sur l'approximation des dérivées des fonctions par les dérivées correspondantes de certaines polynomes du type Bernstein* (French), Mathematica (Cluj), 2 (25), 1960, 335-348.

[23] Stancu, D.D., *The expression of the remainder in some numerical partial differentiation formulas* (Romanian), Acad. R. P. Române, Fil. Cluj, Stud. Cerc. Mat., 11, 1960, 371-380.

[24] Stancu, D.D., *On the calculation of the coefficients of a general quadrature formula* (Romanian), Studia Univ. Babeş-Bolyai Ser. I Math. Phys., 1960, no. 1, 187-192.

[25] Stancu, D.D., *Some Bernstein polynomials in two variables and their applications*, Dokl. Akad. Nauk SSSR, 134, 48-51 (Russian); translated as Soviet. Math. Dokl., 1, 1961, 1025-1028.

[26] Stancu, D.D., *On the integral representation of the remanider in Taylor's formula in two variables* (Romanian), Stud. Cerc. Mat., Acad. R. P. Române, Fil. Cluj, 13, 1962, 175-182.

[27] Stancu, D.D., *On the remainder in the approximation formulae by Bernstein's polynomials*, Notices Amer. Math. Soc., 9, 1962, no. 1, 26.

[28] Stancu, D.D., *The remainder of certain linear approximation formulas for two variables*, Notices Amer. Math. Soc.,1962, no. 2, 207.

[29] Stancu, D.D., *A method for obtaining polynomials of Berstein type of two variables*, Amer. Math. Monthly, 70, 1963, 260-264.

[30] Stancu, D.D., *Quadrature formulas with simple Gaussian nodes and multiple fixed nodes*, Math. Comp., 17, 1963, 384-394.

[31] Stancu, D.D., *Evaluation of the remainder term in approximation formulas by Bernstein polynomials*, Math. Comp., 17, 1963, 270-278.

[32] Stancu, D.D., *Generalizations of an inequality of G. G. Lorentz*, An.Sti. Univ. "Al. I. Cuza", Iaşi, Secţ. mat.-fiz., 9, 1963, 49-58.

[33] Stancu, D.D., *On the moments of some discrete random variables* (Romanian), Studia Univ. Babeş-Bolyai, Ser. Math. Phys., 9, 1964, no.2, 35-48.

[34] Stancu, D.D., *The remainder of certain linear approximation formulas in two variables*, SIAM Numer. Anal., Ser. B, 1, 1964, 137-163.

[35] Stancu, D.D., Straud, A.N., *Quadrature formulas with multiple Gaussian nodes*, SIAM Numer. Anal., Ser. B, 2, 1965, 129-143.

[36] Stancu, D.D., *On the automatic programming of digital computers*, Gaz. Mat., Ser. A, 70, 1965, 170-173, (Romanian).

[37] Stancu, D.D., *On the international algorithmic language ALGOL 60*, Gaz. Mat., Ser. A, 70, 1965, 361-368, 401-408, 475-481.

[38] Stancu, D.D., *A general interpolation formula*, Acad. R. P. Române, Fil. Cluj, Institutul de Calcul, Proceedings of the Colloquium on Convex Functions, with Applications to Numerical Calculus, Cluj, 1965, 92-93.

[39] Stancu, D.D., *On Hermite's osculatory interpolation formula and on some generalization of it*, Mathematica Cluj, 8 (31), 1966, 373-391.

[40] Stancu, D.D., *On the monotoncity of the sequence formed by the first order of the Bernstein polynomials*, Math. Z., 98, 1967, 46-51.

[41] Stancu, D.D., *A method for computing the moments of the multinomial and multiple Poisson distributions*, Studia Univ. Babeş-Bolyai, Ser.Math.-Phys., 12, 1967, no.1, 49-54.

[42] Stancu, D.D., *On the moments of Pólya distribution*, Acad. R. P. Române, Fil. Cluj, Institutul de Calcul, Proceedings of the Colloquium of Approximation Theory, Cluj, 1967, (Romanian).

[43] Stancu, D.D., *On the moments of negative order of the positive Bernoulli and Poisson variables*, Studia Univ. Babeş-Bolyai, Ser. Math.-Phys., 13, 1968, no. 1, 27-31.

[44] Stancu, D.D., *On a new positive linear polynomial operator*, Proc. Japan Acad., 44, 1968, 221-224.

[45] Stancu, D.D., *Approximation of functions by a new class of linear polynomials operators*, Rev. Roumaine Math. Pures Appl., 14, 1968, 1173-1194.

[46] Stancu, D.D., *On the Markov probability distribution*, Bull. Math. Soc. Sci. Math., R. S. Roumanie, 12 (61), 1968, no. 4, 203-208.

[47] Stancu, D.D., *Use of probabilistic methods in the theory of uniform approximation of continuous functions*, Rev. Roumaine Math. Pures Appl., 14, 1969, 673-694.

[48] Stancu, D.D., *A new class of uniform approximating polynomial operators in two and several variables*, Proceedings of the Conference on Constructive Theory of Functions, Budapest, August 24 – September 3, 1969, 443-455.

[49] Stancu, D.D., *On a generalization of the Bernstein polynomials* (Romanian), Studia Univ. Babeş-Bolyai, Ser. Math. Phys., 14, 1969, no.2, 31-45.

[50] Stancu, D.D., *Recurrence relations for the central moments of certain discrete probability laws* (Romanian), Studia Univ. Babeş-Bolyai, Ser.Math.-Mech., 15, 1970, no. 1, 55-62.

[51] Stancu, D.D., *Probabilistic methods in the theory of approximation of functions of several by linear positive operators*, 1970, Approximation Theory (Proc. Sympos., Lancaster, 1969), 329-342, Academic Press,London.

[52] Stancu, D.D., *Approximation properties of a class of linear positive operators*, Studia Univ. Babeş-Bolyai, Ser. Math.-Mech., 15, 1970, fasc. 2, 33-38.

[53] Stancu, D.D., *Approximation of functions of two and several variables by a class of polynomials of Bernstein type* (Romanian), Stud. Cerc.Mat., 22, 1970, 334-345.

[54] Stancu, D.D., *On the distribution functions for the multidimensional Bernoulli and Poisson probability laws* (Romanian), Stud. Cerc. Mat., 22, 1970, 675-681.

[55] Stancu, D.D., *Two classes of positive linear operators*, An. Univ. Timişoara, Ser. Sti Mat., 8, 1970, 213-220.

[56] Stancu, D.D., *On the approximation of functions of two variables by means of a class of linear operators*, In: Constructive Theory of Functions

(Proc. Int. Conf. Varna, 1970; eds. B. Penkov, D. Vacov), 327-336, Sofia, Izdat. Bolgar. Akad. Nauk., 1972, MR. 52, no. 3823.

[57] Stancu, D.D., *On the remanider of approximation of functions by means of a parameter-dependent linear polynomial operator*, Studia Univ. Babeş-Bolyai, Ser. Math.-Mech., 16, 1971, no. 2, 59-66.

[58] Stancu, D.D., *A new generalization of the Meyer-König and Zeller operators*, Anal. Univ. Timişoara, Ser. Sti Mat., 10, 1972, 207-214.

[59] Stancu, D.D., *Approximation of functions of two variables by means of some new classes of positive linear operators*, Numerische Methoden der Approximationstheorie, Band 1 (Tagung, Math. Forschungsinst., Oberwolfach, 1971), 187-203, Internat. Schriftenreihe Numer. Math., Band 16, Birkhäuser, Basel, 1972.

[60] Stancu, D.D., *On the approximation of functions of two variables by means of a class of linear operators.* Constructive theory of functions (Proc. Internat. Conf., Varna, 1970) (Russian), 327-336, Izdat. Bolgar. Akad. Nauk., Sofia, 1972.

[61] Stancu, D.D., *A new class of uniform approximation polynomial operators in two and several variables*, Proceedings of the Conference on the Constructive Theory of Functions (Approximation Theory) (Budapest, 1969), 443-355, Akad'emiai Kiad'o, Budapest, 1972.

[62] Stancu, D.D., *Evaluation of the remainder in certain approximation procedures by Meyer-König and Zeller-type operators*, Internat. Schriftenriehe Nummer. Math., Band 26, Birkhäuser, Basel, 1975.

[63] Stancu, D.D., *The use of linear interpolation for the construction of a class of Bernstein polynomials* (Romanian), Stud. Cerc. Mat., 28, no. 3, 369-379, 1976.

[64] Stancu, D.D., *Use of Biermann's interpolation formula for constructing a class of positive linear operators for approximating multivariante functions*, Constructive theory of functions of several variables (Proc. Conf., Math. Res. Inst., Oberwolfach, 1976), 267-276, Lecture Notes in Math., Vol 571, Springer, Berlin, 1977.

[65] Stancu, D.D., *Approximation of bivariate functions by means of some Bernstein-type operators*, Multivariate approximation (Sympos., Univ. Durham, 1977), Multivariate approximation, Proc. Sympos. Durham, 1977, ed. D. C. Handscomb, Academic Press, London-New York, 1978, 189-208.

[66] Stancu, D.D., *On the precision of approximation of differentiable functions by means of linear positive operators*, Itinerant seminar on functional equations, approximation and convexity, Univ. Babe¸s-Bolyai, Cluj-Napoca, 1978, 74-75, (Romanian).

[67] Stancu, D.D., *An extremal problem in the theory of numerical quadratures with multiple nodes*, Proceedings of the Third Colloquium on Operations

Research (Cluj-Napoca, 1978), 257-262, Univ. Babeş-Bolyai, Cluj-Napoca, 1979.

[68] Stancu, D.D., *Representation of the remainder in an approximation formula of Favard-type*, Itinerant seminar on functional equations, approximation and convexity, Univ. Babeş-Bolyai, Cluj-Napoca, 1979, 185-190 (Romanian).

[69] Stancu, D.D., *Linear interpolation, with applications to numerical approximation*, Gaz. Mat. 84, 1979, no. 11, 401-404.

[70] Stancu, D.D., *Application of divided difference to the study of monotonicity of the derivatives of the sequence of Berstein polynomials*, Calcolo, 16, 1979, no. 4, 431-445, 1980.

[71] Stancu, D.D., *A study of the remainder in an approximation formula using a Favard-Szász type operators*, Studia Univ. Babeş-Bolyai, Ser.Math., 25, 1980, no. 4, 70-76.

[72] Stancu, D.D., *Representation of the remainder in some linear approximation formulas*, Itinerant seminar on functional equations, approximation and convexity, Univ. Babeş-Bolyai, Cluj-Napoca, 1980, 127-129 (Romanian).

[73] Stancu, D.D., *A generalization of the Schonenberg approximating spline operator*, Studia Univ. Babeş-Bolyai, Ser. Math., 26, 1981, no. 2, 37-42.

[74] Stancu, D.D., *On a generalization of the Tiberiu Popoviciu quadrature formula of maximum degree of exactness*, Itinerant seminar on functional equations, approximation and convexity, Univ. Babeş-Bolyai, Cluj-Napoca, 1981, 383-394, (Romanian).

[75] Stancu, D.D., *Quadrature formulas constructed by using certain linear positive operators*, In: Numerical Integration, (Proc. Conf. Math. Res. Inst. Oberwolfach, 1981, ed. G. Mämerlin; ISNM 57), Basel-Boston-Stuttgart: Birkhäuser, 1982, 241-251, no. 65003.

[76] Stancu, D.D., *Procedures of numerical integration of functions obtained by means of some linear positive operators*, Itinerant seminar on functional equations, approximation and convexity, Univ. Babeş-Bolyai, Cluj-Napoca, 1982, 333-337.

[77] Stancu, D.D., *On the representation by divided and finite difference of some linear positive operators constructed by means of probabilistic methods.* Itinerant seminar on functional equations, approximation and convexity (Cluj-Napoca, 1983), 159-166, Univ. Babeş-Bolyai, Cluj-Napoca, 1983.

[78] Stancu, D.D., *Approximation of functions by means of a new generalized Bernstein operator*, Calcolo, 20 (1983), no. 2, 211-229.

[79] Stancu, D.D., *A note on a multiparameter Bernstein-type approximating operator*, Mathematica (Cluj), 26 (49), 1984, no. 2, 153-157.

[80] Stancu, D.D., *Bivariate approximation by some Berstein-type operators*, Proceedings of the Colloquium on approximation and optimization, Cluj-Napoca, October 27, 1984, 25-34.

[81] Stancu, D.D., *Generalized Bernstein approximating operators.* Itinerant seminar on functional equations, approximation and convexity (Cluj-Napoca, 1984), 185-192, Univ. Babeş-Bolyai, Cluj-Napoca, 1984.

[82] Stancu, D.D., *Probabilistic approach to a class of generalized Bernstein approximating operators*, Anal. Numér. Théor. Approx., 14 (1985), no.1, 83-89.

[83] Stancu, D.D., *Bivariate approximation by some Bernstein-type operators*, Procedings of the coloquium on approximation and optimization, pp.25-34, Univ. Babeş-Bolyai Cluj-Napoca, 1985.

[84] Stancu, D.D., *On the representation by divided differences of the remainder in Bernstein's approximation formula*, Seminar of numerical and statistical calculus (Cluj-Napoca, 1984-1985), 103-110, Univ. Babeş-Bolyai Cluj-Napoca, 1985.

[85] Stancu, D.D., *On a class of multivariate linear positive approximating operators*, Studia Univ. Babeş-Bolyai, Ser. Math., 31 (1986), no. 4, 56-64.

[86] Stancu, D.D., *On some spline-type operators of approximation*, Studia Univ. Babeş- Bolyai, Math., 32 (1987), no. 4, 47-54.

[87] Stancu, D.D., Stancu, Felicia, *Quadrature rules obtained by means of interpolatory linear positive operators*, Rev. Anal. Numér. Théor. Approx., 21 (1992), no. 1, 75-81.

[88] Stancu, D.D., *On the integral representation of the remainders in approximation formulae by means of interpolatory linear positive operators*, Research Seminar on Numerical and Statistical Calculus, 69-80, Univ. Babeş-Bolyai, Cluj-Napoca, 1993.

[89] Stancu, D.D., Occoriso, M. R., *Mean-value formulae for integrals obtained by using Gaussian-type quadratures*, Proceedings of the Second International Conference in Functional Analysis and Approximation Theory (Acquafredda di Maratea, 1992), Rend. Circ. Mat. Palermo (2) Suppl. no. 33, 1993, 463-478.

[90] Stancu, D.D., *On the monotonicity properties of a sequence of operators of Meyer-König and Zeller type*, Studia Univ. Babeş-Bolyai, Ser.Math., 39, 1994, no. 2, 97-106.

[91] Stancu, D.D., *On the beta approximating operators of second kind*, Rev. Anal. Numér. Théor. Approx., 24, 1995, no. 1-2, 231-239.

[92] Stancu, D.D., *A note on the remainder in a polynomial approximation formula*, Studia Univ. Babeş-Bolyai, Math., 41, 1996, no. 2, 95-101.

[93] Stancu, D.D., *Representation of remainders in approximation formulae by some discrete type linear positive operators*, Proceedings of the Third

International Conference on Functional Analysis and Approximation Theory, Vol. II (Acquafredda di Maratea, 1996), Rend. Circ. Mat. Palermo (2) Suppl., no. 52, Vol. II, 1998, 781-791.

[94] Stancu, D.D., *Professor Gheorghe Coman at his 60th anniversary*, Studia Univ. Babeş-Bolyai Math., 41, 1996, no. 2, 1-8.

[95] Stancu, D.D., *Approximation properties of a class of multiparameter positive linear operators*, Approximation and optimization, Vol. I (Cluj-Napoca, 1996, 107-120), Transilvania, Cluj-Napoca, 1997.

[96] Stancu, D.D., Cismaşiu, C., *On an approximating linear positive operator of Cheney-Sharma*, Rev. Anal. Numér. Théor. Approx., 26, 1997, no. 1-2, 221-227.

[97] Stancu, D.D., *The remainder in the approximation by a generalized Berstein operator: a representation by a convex combination of second-order divided differences*, Calcolo, 35, 1998, no. 1, 53-62.

[98] Gori, L., Stancu, D.D., *Mean-value formulae for integrals involving generalized orthogonal polynomials. Dédié au Professeur Dr. D.D. Stancu a l'ocassion de son 70e anniversaire*, Rev. Anal. Numér. Théor. Approx., 27, 1998, no. 1, 107-115.

[99] Stancu, D.D., Occorisio, M. R., *On approximation by binomial operators of Tiberiu Popoviciu type. Dédié au Professeur Dr. D. D. Stancu a l'ocassion de son anniversaire*, Rev. Anal. Numér. Théor. Approx., 27, 1998, no. 1, 167-181.

[100] Stancu, D.D., *The evaluation of the remainders in approximation formulas by linear positive operators of interpolatory type*, Gen. Math., 6, 1998, 85-88.

[101] Stancu, D.D., Vernescu A. D., *Approximation of bivariate functions by means of a class of operators of Tiberiu Popoviciu type*, Math. Rep. (Bucureşti), 1, 51, 1999, no. 3, 411-419.

[102] Stancu, D.D., *On the use of divided differences in the investigation of interpolatory positive linear operators*, Studia Sci. Math. Hungar., 35, 1999, no. 1-2, 65-80.

[103] Stancu, D.D., Vernescu A., *On some remarkable positive polynomial operators of approximation*, Rev. Anal. Numér. Théor. Approx., 28, 1999, no. 1, 85-95.

[104] Stancu, D.D., Giurgescu, P., *On the evaluation of remainders in some linear approximation formulas*, RoGer 2000-Braşov, 141-147, Schr. reihe Fachbereichs Math. Gerhard Mercator Univ., 485, Gerhard-Mercator-Univ. Duisburg, 2000.

[105] Stancu, D.D., Stancu, F., *Approximation by a binomial operator depending on several parameters*, 4th International Conference on Functional

Analysis and Approximation Theory, Acquatredda di Maratea, September 22-28, 2000.

[106] Stancu, D.D., *Numerical integration of functions by Gauss-Turán-Ionescu type quadratures*, Mathematical contributions of D. V. Ionescu, 59-68, Babeş-Bolyai Univ. Dept. Appl. Math., Cluj-Napoca, 2001.

[107] Stancu, D.D., Drane, J. W., *Approximation of functions by means of the power of operators S(f; i; r; s)*, Trends in approximation theory (Nashville, TN, 2000), 401-405, Innov. Appl. Math., Vanderbilt Univ. Press, Nashville, TN, 2001.

[108] Stancu, D.D., Coman, Gh., Agratini, O., Trâmbiţaş, R., *Analiză Numerică şi Teoria Aproximării , Vol. I* (Romanian), Numerical Analysis and approximation theory-Vol. I, Presa Universitară Clujeană, Cluj-Napoca, 2001.

[109] Stancu D.D., Coman, Gh., Blaga, P., *Analiză Numerică şi Teoria Aproximării, vol. II* (Romanian), Numerical Analysis and Approximation Theory, Vol. II, Presa Universitară Clujeană, Cluj-Napoca, 2002.

[110] Stancu D.D., Simoncelli, A.C., *Compound poweroid operators of approximation*, Proceedings of the Fourth International Conference on Functional Analysis and Approximation Theory, Vol.II (Potenza, 2000), Rend. Circ. Mat. Palermo (2) Suppl. 2002, no. 68, part II, 845-854.

[111] Stancu D.D., *Methods for construction of linear positive operators of approximation*, Proceedings of the International Symposion Dedicated to the 75th Anniversary of D. D. Stancu, Cluj-Napoca, May 9-11, 2002, pg. 23-45.

[112] Stancu D.D., *On approximation of functions by means of compound power of operators*, in Mathematical Analysis and Approximation Theory, (edited by Lupaş A., Gonska H., Lupaş Luciana), Proc. RoGer-2002, Sibiu, 2002, 259-271.

[113] Stancu D.D. (Editor) , Agratini O., Chiorean Ioana, Coman Gh., Trâmbiţaş R., *Analiză Numerică şi Teoria Aproximării, vol. III* (Romanian), Numerical Analysis and Approximation Theory, Vol. III, Presa Universitară Clujeană , Cluj-Napoca, 2003.

[114] Stancu D.D., Stoica E.I., *On the use of Abel-Jensen type combinatorial formulas for construction and investigation of some angebraic polynomial operators of approximation*, Studia UBB Mathematica, LIV (2009), no. 4, 167-182.

Annex B : Scientific Papers Containing in the Title the Name of D. D. STANCU

1. Abel, U., Asymptotic Approximation with Stancu Beta Operators, Rev. Analyse Numer. et Theorie de l'Approximation, vol.27(1998).

2. Acu, D., Boncut, M., Cismaşiu, C., Evaluation of the remainder in an approximation formula of D.D. Stancu, Calcolo, vol. 35(1998), 191-196.

3. Acu, D., On the D.D. Stancu method of parameters, Studia Univ. Babeş-Bolyai, Mathematica, vol. 47(1997), 1-7.

4. Agratini, O., Approximation Properties of a Class of Operators of Stancu-Kantorovich Type, Babeş-Bolyai Univ. Preprint nr.1, Fac. Math., Cluj-Napoca, pp. 3-12, 1994.

5. Agratini, O., On the monotonicity of a sequence of Stancu-Bernstein type operators, Studia Univ. Babeş-Bolyai, Mathematica, vol. 41(1996), 17-23.

6. Agratini, O., Application of divided differences to the study of monotonicity of a sequences of D.D. Stancu polynomials, Rev. Analyse Numerique et Theorie de l'Approximation, vol. 25(1996), pp. 3-10.

7. Agratini, O., On Simultaneous Approximation by Stancu-Bernstein Operators, Proceedings of ICAOR, vol. I, Transilvania Press, Cluj-Napoca, 1997, pp. 157-162.

8. Agratini, O., Linear Combinations of D.D. Stancu polynomials, Rev. Analyse Numer. et Theorie de l'Approximation, vol. 27(1998).

9. Agratini, O., Stancu modified operators revisited, Rev. Analyse Numer. Theorie de l'Approximation, vol. 31(2002).

10. Altomare, F., Campiti, M., Korovkin-type Approximation Theory and its Applications & 5.2.7: Stancu Operators (301-306); & 6.1: Stancu-Schnabl operators (385-399, 419-420, 423-479); Bernstein-Stancu operators (117- 123, 220-221), Walter de Gruyter, Berlin, New-York, 1994.

11. Bărbosu, D., On a blending operator of Bernstein-Stancu type, Proc. Annual Conference Romanian Soc. Math. Sci., Bucharest, May 29-June 1, 1997, pp. 217-222.

12. Bărbosu, D., The approximation of the Bögel-continuous functions using the Bernstein-Stancu polynomials, Bul. Sti. Univ. Baia Mare, vol. 8(1992), 11-18.

13. Bărbosu, D., On the approximation of three variate B-continuous functions using Bernstein-Stancu type operators, Bul. Stiint. Univ. Baia Mare, Ser. B, vol 9(1993), pp. 9-18.

14. Bărbosu, D., L'approximation par des polynomes du type Bernstein-Stancu des fonctions Boegel-continues, Bul. Stiinţ. Univ. Baia Mare, Ser. B, vol. 8(1992), pp. 11-18.

15. Birkhoff, G., Schultz, M.H., Varga, R.S., Piecewise Hermite Interpolation in One and Two Variables with Applications to Partial Differential Equations, Numerische Mathematik, vol. 11(1968), pp. 232-236 & 6. Piecewise Hermite Interpolation: Method of Stancu and Simonsen.

16. Campiti, M., Limit semigroup of Stancu-Muhlbach operators associated with positive projections, Ann. Sc. Norm. Pisa, Cl. Sci. IV, Ser. 19(1992), nr.1, pp. 51-67.

17. Campiti, M., A generalization of Stancu-Muhlbach operators, Constructive Approximation, vol. 7(1991), nr. 1, pp. 1-18.

18. Campiti, M., Metafune, G., Solutions of abstract Cauchy problems approximated by Stancu-Schnabl-type operators, Atti Accad. Sci. Torino, Cl. Sci. Fis. Mat. Natur. vol. 130(1996), pp. 81-93.

19. Chen, W.Z., Tian, J.S., On approximation Properties of Stancu Operators of Integral Type, Acta Sci. Natur. Univ. Amoien vol. 26(1987), pp.270-276.

20. Chen, W.Z., Gu, S.M., Approximation Theorems for Stancu-type operators, Xiamen Daxue Xuebno Ziran Kexue Ban vol. 32(1993), pp. 679-684.

21. Chen,W., On two dimensional Stancu-Mühlbach operator, Approximation Theory Appl. vol. 13(1997), nr. 3, pp. 70-77.

22. Della Vecchia, B., On Stancu operator and its Generalizations, Inst. Applied Mathematics, Napoli, Rapp. Tecnico nr. 47(1988).

23. Della Vecchia, B., On the Approximation of Functions by means of the Operators of D.D. Stancu, Studia Univ. Babeş-Bolyai, Mathematica, vol. 37(1992), pp. 3-36.

24. Della Vecchia, B., Mache, D.H., On Approximation Properties of Stancu-Kantorovich Operators, Rev. Analyse Numer. et Theorie de l'Approximation, vol. 27(1998).

25. De la Cal, J., On Stancu-Mühlbach operators and some connected problems concerning probability distributions, J. Approx. Theory vol. 74(1993), pp.56-68.

26. Di Lorenzo, A., Occorsio, M.R., Polinomi di Stancu, Ins. Applicazioni della Matematica, Napoli, Rapp. Tecnico nr. 121(1995), 42 pag.

27. Finta, Z., On some Propertes of Stancu Operator, Rev. Analyse Numer. et Theorie de l'Approximation vol. 27(1998).

28. Felbecker, G., Ueber Verallgemeinerte, Stancu-Mühlbach Operatoren, Z. Angew. Math. Mech. 53, Sonderheft, T188-T189(1973).

29. Frenţiu, M., On the Asymptotic Aspect of the Approximation of Functions by means of the D.D. Stancu operators, Babeş-Bolyai University, Preprint nr. 8(1987), pp. 57-64.

30. Gonska, H.H., Meier, J., Quantitative Theorems and Approximation by Bernstein-Stancu Operators, Calcolo vol. 21(1984), pp. 317-335.

31. Gonska, H.H., On approximation in spaces of continuous functions (& 4.2 Approximation by bivariate Bernstein-Stancu operators), Bull. Austral. Math. Soc. vol. 28(1983), 411-432.

32. Gori, L., Stancu, D.D., Mean Value Formulae for Integrals Involving Generalized Orthogonal Polynomials, Rev. Analyse Numer. et Theorie de l'Approximation, vol. 27(1998).

33. Gunttner, R., Beitrage zur Theorie der Operatoren vom Bernstein schen Typ (& 5.2 Die Operatoren von Stancu), Doctoral Dissertation, Techn. Univ. Clausthal, 1974.

34. Horova, I., Budikova, M., A note on D.D. Stancu Operators, Ricerche di Matematica vol. 44(1995), pp. 397-407.

35. Horova, I., Gegenbauer Polynomials, optimal Kernels and Stancu Operators, Approximation Theory and Function Series, Bolyai Math. Society, Math. Studies, 5, Budapest, 1996, 227-235.

36. Jiang, T., On the rate of convergence of Stancu-Sikkema-Bernstein operator and Stancu-Sikkema-Kantorovich operator for functions of bounded variation, Math. Appl. vol. 3(1990), nr. 2, pp. 89-90.

37. Jiang, G.J., Approximation properties of Stancu-Kantorovich operators, Journal of Shandong-Mining-Institute, vol. 9(1990), nr. 2, pp. 193-196.

38. Lorentz, G.G., Approximation of Functions, Chelsea Publ. Comp. New-York, 1986, (In Preface to the Second Edition: "I am grateful to D.D.Stancu, Cluj, Romania, for many corrections").

39. Mastroianni, G., Occorsio, M.R., Sulle derivete dei Polinomi di Stancu, Rend. Accad. Sci. M.F.N., Ser. IV, vol. 45(1978), pp. 273-281.

40. Mastroianni, G., Occorsio, M.R., Una Generalizzazione dell'Operatore di Stancu, Rend. Accad. Sci. M.F.N., Ser. IV, vol 45(1978), 495-511.

41. Mühlbach, G., Verallgemeinerung der Bernstein und Lagrange Polynome. Bemerkungen zu einer Klasse linearer Polynom-operatoren von D.D. Stancu, Rev. Roumanie Math. Pres Appl., vol. 15(1970). pp. 1235-1252.

42. Occorsio, D., Simoncelli, A.C., Generalized Stancu-Polya Curves, Rev. Analyse Numer. et Theorie de l'Approximation, vol. 27(1998).

43. Raşa, I., Vladislav, T., Some properties of Bernstein and Stancu Operators, Approximation and Optimization, Proceedings of ICAOR, vol. I, Transilvania Press, Cluj-Napoca, 1997, pp. 345-350.

44. Rus, I.A., Iterates of Stancu operators, via contraction principle, Studia Babeş-Bolyai, Mathematica, vol.47, No.4/2002.

45. Vlaic, G., Bivariate Approximation by D.D. Stancu type Polynomials on a Triangle, Approximation and Optimization, Proceedings of ICAOR, vol.II, Transilvania Press, Cluj-Napoca, 1997, pp. 247-252.

46. Vlaic, G., Multivariate Approximation by Stancu type operators on a simplex, Bul. St. "Politehnica", Timişoara, vol. 41(55)(1996), pp. 247-252.

47. Vlaic, G., On a bivariate Multiparameter Approximation Operator of D.D.Stancu, Studii şi Cercetări Matematice, Academia Română, 1997.

48. Walz, G., Trigonometric Bezier and Stancu polynomials over intervals and triangles, Comput. Aided Geom. Design, vol. 14(1997), pp. 393-397.

49. Zhang, C., Asymptotic formulae for approximation error for smooth functions by Stancu operator on Simplex, J. Math, Res. Expo, vol. 17(1997), nr. 4, pp. 573-577.

50. Zhao, J., Lsp saturation for Stancu-Kantorovich operators, J. Math. Wuhan Univ., vol. 8(1988), nr. 3, pp. 257-262.

51. Xue, Y., L'approximation by the iterates of Stancu-Kantorovich operators, J. Math. Res. Expo. vol. 17, nr. 4, 1997, pp. 570-572.

52. Xiong, J., Yang. R., Cao, F., Approximation theorems by Stancu-Kantorovich polynomials on a simplex, J. Qufu Norm. Univ. Nat. Sci., vol. 19(1993), nr. 4, pp. 29-34.

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

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

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

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVIII**, Number 1, 2013

A GOOD DRAWING OF COMPLETE BIPARTITE GRAPH $K_{9,9}$, WHOSE CROSSING NUMBER HOLDS ZARANKIEWICZ CONJECTURES

MOHAMMAD REZA FARAHANI

Abstract. There exist some Drawing for any graph $G = (V, E)$ on plan. An important aim in Graph Theory and Computer science is obtained a best drawing of an arbitrary graph. Also, a draw of a non-planar graph G on plan generate several edge-cross. A good drawing (or strongly best drawing) of G is consist of minimum edge-cross.

The crossing number of a graph G, is the minimum number of crossings in a drawing of G in the plane, denoted by $cr(G)$. A crossing is a point of intersection between two edges. The crossing number of the complete bipartite graph is one of the oldest crossing number open problems.

In this paper, we present a good drawing of complete bipartite graph $K_{9,9}$. This drawing is able to developed on $K_{n,n}$, $\forall n \leq 9$ and implies that the crossing number of these graphs hold Zarankiewicz conjecture. $\forall n, m \in \mathbb{N}$ Zarankiewicz conjecture is equal to

$$cr(K_{n,m}) \stackrel{?}{=} Z(m,n) = [\frac{m}{2}][\frac{m-1}{2}][\frac{n}{2}][\frac{n-1}{2}].$$

1. Introduction

Let $G = (V, E)$ be a simple finite connected graph with the vertex set $V(G)$ and the edge set $E(G)$. $|V(G)| = n$, $|E(G)| = e$ are the number of vertices and edges.

For each vertex v of a graph G, let $N_G(v) := \{u \in V(G) | uv \in E(G)\}$ be the neighborhood of v in G. The degree of v, denoted by $deg(v)$, is $|N_G(v)|$. Let $\Delta(G)$ be the maximum degree of a vertex of G.

The crossing number of a graph G, denoted by $cr(G)$, is the minimum number of crossings in a drawing of G in the plane.

A drawing of a graph represents each vertex by a distinct point in the plane, and represents each edge by a simple closed curve between its endpoints, such

Received by the editors: November 20, 2012.

2010 *Mathematics Subject Classification.* 05C10, 05C35.

Key words and phrases. Complete graph, Complete bipartite graph, Crossing number, Best drawing.

that the only vertices an edge intersects are its own endpoints, and no three edges intersect at a common point (except at a common endpoint). A drawing is convex if in addition the vertices are in convex position. A crossing is a point of intersection between two edges (other than a common endpoint). A drawing with no crossings is crossing-free. A graph is planar if it has a crossing-free drawing, see [4, 12, 22] for surveys. For example look at Figure 1, (planar graph K_4 and non-planar graphs K_5, $K_{3,3}$).

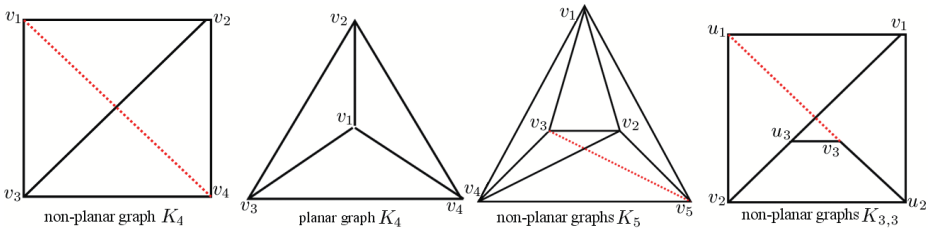


FIGURE 1. Figures of K_4, K_5 and $K_{3,3}$ on the plan

The crossing number is an important measure of the non-planarity of a graph [18]. Computing the crossing number is NP-hard [5], and remains so for simple cubic graphs [9, 13]. Moreover, the exact or even asymptotic crossing number is not known for specific graph families, such as complete graphs [14], complete bipartite graphs [11,14,16] and Cartesian products [1, 2, 6-8, 10, 15, 16, 19-21, 23, 24].

Determining the crossing number of the complete bipartite graph is one of the oldest crossing number open problems. It was first posed by Turan and known as Turan's brick factory problem. In 1954, Zarankiewicz conjectured [24] that it is equal to

$$cr(K_{n,m}) \stackrel{?}{=} Z(m,n) = [\frac{m}{2}][\frac{m-1}{2}][\frac{n}{2}][\frac{n-1}{2}].$$

He even gave a proof and a drawing that matches the lower bound, but the proof was shown to be flawed by Richard Guy [7]. Then in 1970 D.J. Kleitman proved that Zarankiewicz conjecture holds for $Min(m;n) \leq 6$ [10]. In 1993 D.R. Woodall proved it for $m \leq 8$; n$\leq$ 10 [23]. Previously the best known lower bound in the general case for all $m, n \in \mathbb{N}$ was the one proved by D.J. Kleitman [10]:

$$cr(K_{n,m}) \geq \frac{1}{5}\,(m(m-1))\,[\frac{n}{2}][\frac{n-1}{2}].$$

Now, we have the better lower bound [11]

$$cr(K_{n,m}) \geq \frac{1}{5}\,(m(m-1))\,[\frac{n}{2}][\frac{n-1}{2}] + 9.9 \times 10^{-6} m^2 n^2.$$

for sufficiently large m and n.

Upper bounds on the crossing number of general families of graphs have been less studied. Obviously $cr(K_{n,m}) \leq \binom{|E(G)|}{2}$ for every graph G.

2. Drawing of complete Bipartite graph $K_{9,9}$

D.R. Woodall [10] used an elaborate computer search to show that Zarankiewicz conjecture holds for $K_{7,7}$ and $K_{7,9}$. Thus, one of the smallest unsettled instance of Zarankiewicz conjecture is $K_{9,9}$. For further research see paper series [8, 10, 11, 17-21].

So, we focus on the best drawing of complete bipartite graph $K_{9,9}$ and compute its crossing number for this drawing. In continue, we claim that this drawing is a best drawing for $K_{9,9}$ and $cr_D(K_{9,9})$ hold Zarankiewicz conjecture. By according the Figure 5. Also we show that by similar drawing for $K_{7,7}$ which is a best drawing of it and hold Zarankiewicz conjecture, it is maybe another proof of $cr_D(K_{7,7})$.

Before beginning present of this drawing, we give some definitions that will be used throughout the paper.

Definition 1. The crossing number $cr(G)$ of a graph G is the smallest crossing number of any drawing of G in the plane, where the crossing number cr of a drawing D is the number of non-adjacent edges that have a crossing in the drawing.

Definition 2. A good drawing a graph G is a drawing where the edges are non-self-intersecting where each two edges have at most one point in common, which is either a common end vertex or a crossing.

Clearly a drawing with minimum crossing number must be a good drawing (or for strongly a best drawing) and obviously a good drawing of planar graph G is the crossing-free drawing.

Definition 3. Suppose $V = \{v_1, v_2, ..., v_n\}$ is the vertex set of an arbitrary graph G. Then $E(G)$ (the edge set of G) is consist of $e_{i,j}$, such that v_i is adjacent with v_j $(\forall i, j \in \mathbb{Z}_n = \{1, 2, ..., n\})$. Now, Pair-Cross Matrix of G $(CR(G) = [cr_{i,j}]_{i,j \in \mathbb{Z}_n =})$ presents the number of all cross on the edge $e_{i,j}$.

It's obvious that, if v_i, v_j be the non-adjacent vertices, then $cr_{i,j} = 0$. Since, there exist many different drawing for a graph G, therefore we have a Pair-Cross Matrix $CR_D(G)$ for any drawing D of G. Also, it's obvious that all Pair-Cross Matrix $CR_D(G)$ are symmetric and the members on the original diameter are equal to zero.

$$CR_D(G) = \begin{bmatrix} 0 & cr_{1,2} & cr_{1,3} & . & . & . & cr_{1,n} \\ cr_{2,1} & 0 & cr_{2,3} & . & . & . & cr_{2,n} \\ cr_{3,1} & cr_{3,2} & 0 & . & . & . & cr_{3,n} \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ cr_{n,1} & cr_{n,2} & cr_{n,3} & . & . & . & 0 \end{bmatrix}_{n\times n} \begin{matrix} \rightarrow cr_{v_1} \\ \rightarrow cr_{v_2} \\ \rightarrow cr_{v_3} \\ . \\ . \\ . \\ \rightarrow cr_{v_n} \end{matrix}$$

$$\begin{matrix} \downarrow & \downarrow & \downarrow & & & & \downarrow \\ cr_{u_1} & cr_{u_2} & cr_{u_3} & . & . & . & cr_{u_n} \end{matrix} \tag{1}$$

Example 1. By according to Figure 1, we see that the drawing D_1 is the crossing-free drawing of K_4. So Pair-Cross Matrix of K_4 will be equal to

$$CR_{D_1}(K_4) = 0$$

and also

$$CR_{D_2}(K_4) = \begin{matrix} v_1 \rightarrow \\ v_2 \rightarrow \\ v_3 \rightarrow \\ v_4 \rightarrow \end{matrix} \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}_{4\times 4}$$

$$\begin{matrix} \uparrow & \uparrow & \uparrow & \uparrow \\ v_1 & v_2 & v_3 & v_4 \end{matrix} \tag{2}$$

Example 2. Similar Above (see Figure 1), Pair-Cross Matrix of $K_5, K_{3,3}$ on the best drawing D will be equal to

$$CR_{D_3}(K_5) = \begin{matrix} v_1 \rightarrow \\ v_2 \rightarrow \\ v_3 \rightarrow \\ v_4 \rightarrow \\ v_5 \rightarrow \end{matrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}_{5\times 5}$$

$$v_1 \quad v_2 \quad v_3 \quad v_4 \quad v_5 \tag{3}$$

and

$$CR_{D_4}(K_{3,3}) = \begin{matrix} v_1 \rightarrow \\ v_2 \rightarrow \\ v_3 \rightarrow \\ u_1 \rightarrow \\ u_2 \rightarrow \\ u_3 \rightarrow \end{matrix} \begin{bmatrix} 0 & 0 & 0 & |0 & 0 & 1 \\ 0 & 0 & 0 & |0 & 0 & 0 \\ \underline{0} & \underline{0} & \underline{0} & |\underline{1} & \underline{0} & \underline{0} \\ \overline{0} & \overline{0} & \overline{1}| & 0 & 0 & 0 \\ 0 & 0 & 0| & 0 & 0 & 0 \\ 1 & 0 & 0| & 0 & 0 & 0 \end{bmatrix}_{6\times 6}$$

$$v_1 \quad v_2 \quad v_3 \quad u_1 \quad u_2 \quad u_3 \tag{4}$$

Corollary 1. The summation of all members of $CR_D(G)$ implies that is equal to the crossing number $CR_D(G)$ of a graph G on the drawing D. In other words

$$CR_D(G) = \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} cr_{i,j} = \frac{\sum_{i=1}^{n} cr_{v_i}}{4} \tag{5}$$

Definition 4. Let $V_1 = \{v_1, v_2, ..., v_n\}$ and $V_2 = \{u_1, u_2, ..., u_m\}$ be two partitions of $V(K_{m,n})$, where $V(K_{m,n})$ is the vertex set of the complete bipartite graph $K_{m,n}$. Now, the Pair-Cross Matrix $CR^*_D(K_{m,n})$ presents the number of all cross on the edge $e_{i,j} = v_iu_j$ as follow:

$$CR^*_D(K_{m,n}) = V_1\{\overbrace{\left[cr_{v_iu_j}\right]}^{V_2}{}_{n\times m} \tag{6}$$

We redefine this matrix for $K_{m,n}$, because by rewrite Definition 3 for $G = K_{m,n}$ then

$$CR_D(K_{m,n}) = \begin{array}{c} V_1 \rightarrow \\ V_2 \rightarrow \end{array} \begin{array}{c} \\ \left[\begin{array}{cc} 0 & CR^*_D(K_{m,n}) \\ CR^*_D(K_{m,n})^t & 0 \end{array}\right]_{(m+n)\times(m+n)} \\ \begin{array}{cc} V_1 & V_2 \end{array} \end{array} \tag{7}$$

and $CR^*_D(K_{m,n}) = CR^*_D(K_{m,n})^t$.

Example 3. By according to Figure 1, it is obvious that modified Pair-Cross Matrix of $K_{3,3}$ is

$$CR^*_D(K_{3,3}) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{8}$$

Corollary 2. The summation of all members of $CR^*_D(K_{m,n})$ is equal to the crossing number $CR_D(K_{m,n})$ of a complete bipartite graph on the drawing D. Thus

$$CR_D(G) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m} cr^*_{i,j}. \tag{9}$$

2.1. **Method.** In this subsection, we achieve a good drawing D of complete bipartite graph $K_{9,9}$ and conclude the crossing number $CR_D(K_{9,9})$ of it. We start this process with an arbitrary drawing D of complete bipartite graph $K_{9,9}$ and we make Pair-Cross Matrix $CR^*_D(K_{9,9})$ by according to the drawing D. So, we find a large member of $CR^*_D(K_{9,9})$ ($\Delta cr = Max\{cr^*_{i,j} | i, j \in \mathbb{Z}_9$) and we redraw the correlate edge with it, such that decrease the number of

cross on the correlate edge. Notice that this change must n't many increase other members of $CR^*_D(K_{9,9})$. By repeat this process several times, upshot we will have a good drawing of complete bipartite graph $K_{9,9}$. See Figure 3 (For look Figure 3, attention to Appendix 1.) and Pair-Cross Matrix $CR^*_D(K_{9,9})$ is equal to

$$CR^*_D(K_{9,9}) = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \end{array} \begin{array}{c} \begin{array}{ccccccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 & v_8 & v_9 \end{array} \\ \left(\begin{array}{ccccc:cccc} 16 & 12 & 8 & 4 & 0 & 0 & 4 & 8 & 12 \\ 12 & 9 & 6 & 3 & 0 & 0 & 3 & 6 & 9 \\ 8 & 6 & 5 & 4 & 3 & 4 & 5 & 6 & 7 \\ 4 & 3 & 4 & 5 & 6 & 8 & 7 & 6 & 5 \\ 0 & 0 & 3 & 6 & 9 & 12 & 9 & 6 & 3 \\ \hdashline 0 & 0 & 4 & 8 & 12 & 16 & 12 & 8 & 4 \\ 4 & 3 & 5 & 7 & 9 & 12 & 10 & 8 & 6 \\ 8 & 6 & 6 & 6 & 6 & 8 & 8 & 8 & 8 \\ 12 & 9 & 7 & 5 & 3 & 4 & 6 & 8 & 10 \end{array} \right) \end{array} \begin{array}{l} \\ \longrightarrow 64 = 4(4)^2 \\ \longrightarrow 48 = 3(4)^2 \\ \longrightarrow 48 = 3(4)^2 \\ \longrightarrow 48 = 3(4)^2 \\ \longrightarrow 48 = 3(4)^2 \\ \longrightarrow 64 = 4(4)^2 \\ \longrightarrow 64 = 4(4)^2 \\ \longrightarrow 64 = 4(4)^2 \\ \underset{+}{\longrightarrow} 64 = 4(4)^2 \\ \left(\frac{512}{2}\right) = 256 = (4)^2(4)^2 \end{array}$$

By refer to Figure 3 of complete bipartite graph $K_{9,9}$, it is obvious to see that this figure is symmetric (near to symmetric) and the vertices are in the two opponent cycles (eight vertices as a common set in one of cycles and one remaining vertex is a center of another cycle). As well as, these two cycles and their covered vertices have stated in a mirror (See close up view of $K_{9,9}$ in Figure 2).

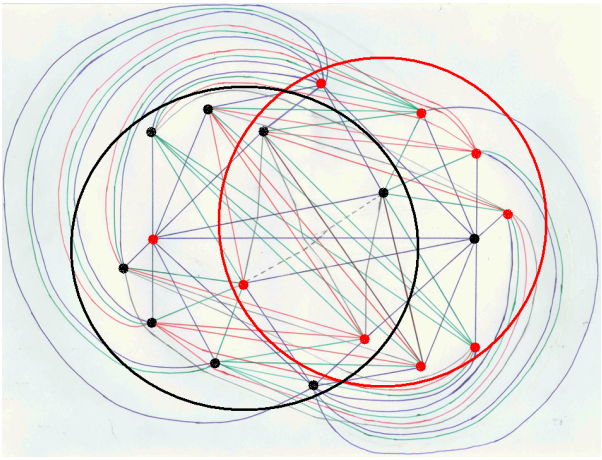


FIGURE 2. The close up view of $K_{9,9}$ with two cycles that covered vertices (black and red cycles).

Now, by according to the matrix $CR^*_D(K_{9,9})$ and Figure 3, if we redraw an edge e_{uv}, then we increase the crossing number cr^*_{uv}) obviously. But, an important point is number 4 and its multiples in the matrix $CR^*_D(K_{9,9})$. Number

4 is important, since $4 = \left[\frac{9}{2}\right]$. On the other hand, number 3 is important in the matrix $CR^*_D(K_{7,7})$, since $3 = \left[\frac{7}{2}\right]$ similarly. See complete bipartite graph $K_{7,7}$ in Figure 4 (on Appendix 2) and Pair-Cross Matrix $CR^*_D(K_{7,7})$ as follow:

$$CR^*_D(K_{7,7}) = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \end{array} \begin{array}{c} \begin{array}{ccccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \end{array} \\ \left(\begin{array}{ccccccc} 9 & 6 & 3 & 0 & 0 & 3 & 6 \\ 6 & 4 & 2 & 0 & 0 & 2 & 4 \\ 3 & 2 & 2 & 2 & 3 & 3 & 3 \\ 0 & 0 & 2 & 4 & 6 & 4 & 2 \\ 0 & 0 & 3 & 6 & 9 & 6 & 3 \\ 3 & 2 & 3 & 4 & 6 & 5 & 4 \\ 6 & 4 & 3 & 2 & 3 & 4 & 5 \end{array}\right) \end{array} \begin{array}{l} \longrightarrow \quad 27 = 3(3)^2 \\ \longrightarrow \quad 18 = 2(3)^2 \\ \longrightarrow \quad 18 = 2(3)^2 \\ \longrightarrow \quad 18 = 2(3)^2 \\ \longrightarrow \quad 27 = 3(3)^2 \\ \longrightarrow \quad 27 = 3(3)^2 \\ \longrightarrow + 27 = 3(3)^2 \\ \hline \left(\frac{162}{2}\right) = 81 = (3)^2(3)^2 \end{array}$$

3. Conclusions

In this report, we drawing $K_{9,9}$ in the plan with 256 crossing number. We obtained this drawing by draw $K_{9,9}$ step to step, such that we choose a large Cr on the Pair-Cross Matrix and redraw it for decrease crossing number. In fact, this work is quite tentative and experience, in other words, is handwork. In other way, we can drawing $K_{9,9}$ by add two vertices to a best drawing $K_{8,8}$ (Readers know that this graph have 144 crossing points in best drawing or $Cr(K_{8,8}) = 144$), and also we can obtain a best drawing $K_{8,8}$ by add two vertices to best drawing $K_{7,7}$ ($Cr(K_{7,7}) = 81$). In other words, For $h = 3, ..., 9$; we can draw all complete graphs $K_{h,h}$, that the crossing number of them hold Zarankiewicz conjecture.

4. Acknowledgement

The author is thankful to Prof. Farhad Shahrokhi of Department of Computer Science, University of North Texas (Denton, USA) and Dr. Mehdi Alaeiyan of Department of Mathematics, Iran University of Science and Technology (IUST) for their valuable comments and suggestions.

References

[1] J.Adamsson, and R.B. Richter, *Arrangements, circular arrangements and the crossing number of $C_7 \times C_m$. J. Combin. Theory Ser. B*, 90 (1), (2004), 21-39.

[2] D. Bokal, *On the crossing numbers of Cartesian products with paths $C_7 \times C_m$. J. Combin. Theory Ser. B*, 97(3) (2007), 381-384.

[3] V. Dujmovic, K. Kawarabayashi, B. Mohar and D.R. Wood. *Improved Upper Bounds on the Crossing Number. Manuscript.* December 3, (2007).

[4] P. Erdos, R.P. Guy, *Crossing number problem, American Mathematical Monthly* 80 (1973), 52-58.

[5] M. R. Garey and D. S. Johnson, *Crossing number is NP-complete, SIAM J. Algebraic and Discrete Methods.* 4 (1983), 312-316.
[6] L.Y.Glebsky, G. Salazar, *The crossing number of* $cr(C_n \times C_m) = (m-2)n$ *is as conjectured for* $n \geq m(m+1)$*, J. Graph Theory.* 47 (2004), 53-72.
[7] R.K. Guy, *The decline and fall of Zarankiewicz's theorem, Proof techniques in Graph Theory*, F. Harary, ed. Academic Press, New York (1969), 63-69.
[8] R.K. Guy and T. Jenkins, *The toroidal crossing number of* $K_{m,n}$*, J. Combinatorial Theory.* 6 (1969), 235-250.
[9] P. Hlineny, *Crossing number is hard for cubic graphs. Mathematical Foundations of Computer Science*, Lecture Notes in Computer Science 3153, Springer, Berlin, (2004), 772-782.
[10] D.J. Kleitman, *The crossing number of* $K_{5m,n}$*, J. Combinatorial Theory.* 9 (1970), 315-323.
[11] N.H. Nahas, *On the crossing number of* $K_{m,n}$*. Electron. J. Combin*, 10:N8, (2003).
[12] J. Pach, G. Toth, *Which crossing number is it anyway? J. Combin. Theory Ser. B*, 80(2), (2000), 225-246.
[13] M.J. Pelsmajer, M. Schaefer, and D. Stefankovic. *Crossing number of graphs with rotation systems.* Proc. 15th International Symp. on Graph Drawing (GD '07), *Lecture Notes in Comput. Sci, Springer*, to appear. Tech. Rep. 05-017, School of Computer Science, Telecommunications and Information Systems, DePaul University, Chic ago, U.S.A, (2005).
[14] R.B. Richter and C. Thomassen. *Relations between crossing numbers of complete and complete bipartite graphs, The American Mathematical Monthly.* 104(2) (1997), 131-137.
[15] R.B. Richter, c. Thomassen. *Intersections of curve systems and the crossing number of* $C_5 \times C_5$*. Discrete Comput. Geom*, 13(2), (1995), 149-159.
[16] R.B. Richter and J. Siran, *The crossing number of* $K_{3,n}$ *in a surface, J. Graph Theory.* 21 (1996), 51-54.
[17] F. Shahrokhi, L. A. Szekely, O. Sykora, and I. Vrto. *Drawings of graphs on surfaces with few crossings. Algorithmica*, 16(1), (1996), 118-131.
[18] L.A. Szekely, *A successful concept for measuring non-planarity of graphs: the crossing number, Discrete Mathematics.* 27 (2004), 331-352.
[19] P. Turan. *A note of welcome, J. Graph Theory.* 1 (1977) 7-9.
[20] K. Urbanik, *Solution du probleme pose par P.Turan concerning graphs, Colloq. Math.* 3 (1955), 200-201.
[21] V. Vassilevska, *On the crossing number of* $K_{9,9}$*, Manuscript.*
[22] I. Vrto, *Crossing numbers of graphs: A bibliography. http://www.i.savba.sk/ imrich.* December, (2010).
[23] D.R. Woodall, *Cyclic-order graph and Zarankiewicz's crossing number conjecture, J. Graph Theory.* 17 (1994), 657-671.
[24] K. Zarankiewicz, *On a problem of P. Turan concerning graphs, Fund. Math.* 41 (1954), 137-145.

Department of Applied Mathematics, Iran University of Science and Technology (IUST), Narmak, Tehran 16844, Iran
E-mail address: Mr_Farahani@MathDep.iuat.ac.ir

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVIII**, Number 1, 2013

ON THE USE OF ELO RATING FOR ADAPTIVE ASSESSMENT

MARGIT ANTAL

Abstract. In this paper, we present a new item response model for computerized adaptive testing: Item Response Theory combined with Elo rating. Adaptive test systems require a calibrated item bank and item calibration methods are usually based on Item Response Theory (IRT). However, these methods require item pretesting on large sample sizes, which is very expensive. Hence, this paper presents alternative methods for item difficulty calibration.

Results show that combining IRT with Elo rating is an alternative model for adaptive item sequencing, which offers not only estimations for abilities, but for item difficulties too. The new adaptive item sequencing model was compared with IRT on artificial data. Results show that the new method is able to estimate the ability of the examinee, although more items are required compared to IRT. Hence, this method is recommended for test systems where adaptation to the user knowledge level is a requirement, but the duration of the measurement is less important, i.e. practice systems.

1. Introduction

There are more and more adaptive e-learning systems trying to fulfill specific needs of learners. These systems can be adapted to user knowledge, user interests or user individual traits [2], to name a few specific elements of a user model. In this paper we study adaptive test systems where item sequencing is adapted to user knowledge. These systems are known as Computerized adaptive testing (CAT) systems. Many organizations test their examinees using CAT tools (e.g. TOEFL, GMAT, GRE). The CAT tools used in these examinations are not free software, therefore they cannot be deeply evaluated nor compared. However, Economides and Roupas [4] succeed in performing a rough comparison of these CAT systems based on demo versions of these applications. These CAT tools select items from a calibrated item pool. The item

Received by the editors: December 10, 2012.

2010 *Mathematics Subject Classification.* 97Q70.

1998 *CR Categories and Descriptors.* K.3 [**Computers and Education**]: Computer and Information Science Education – *Computer science education* .

Key words and phrases. Item Difficulty, Item Response Theory, Performance Assessment.

pool calibration is usually performed by the means of Item response theory (IRT, [12]).

There are a lot of free and commercial software designed for item calibration (BILOG-MG, MULTILOG, PARAM-3PL etc.) by means of IRT and based on item pre-test data. Unfortunately, quality item pre-testing is very costly, thus educational institutions cannot afford it. This includes both the cost of item prestest and maintenance of such a system as well. Moreover, the item parameter estimates obtained through item pretest are very sensitive to examinees. Stocking in paper [11] conducted a study in order to analyze optimal examinee quality for accurate item parameter estimation. She found that item calibration is very sensitive to examinee quality; hence inappropriate examinees can easily lead to incorrect item parameters. She also concluded that a broad distribution of abilities may provide more information than a bell-shaped distribution.

The first objective of this paper is to search for alternative item parameter estimation methods, which are comparable with IRT item calibration estimates. In this paper we restrict ourselves to the 1PL IRT model, which uses only one item parameter, namely item difficulty. Very few studies are to be found in this field. Wauters et. al. in paper [15] conducted a study in this direction and ranked six alternative item difficulty estimation methods by their correlates to the IRT based estimates. [14] is a previous version of this study. [9] applied the Elo rating for on the fly item difficulty estimation. There are a few studies about teachers or experts ability to estimate item difficulty, concluding that teachers are not very accurate in this task [6], [7], [15].

The second objective of this paper is to introduce a new adaptive item sequencing method based on Elo Rating System (ERS), which is able to estimate both the examinees' ability and item difficulties. In our ERS method we combined Elo rating based ability estimation with IRT based next item selection method. Even though the Elo rating for adaptive item sequencing was studied by other researchers too [1], [9], [15], a detailed comparison with IRT is missing. We compared our new item sequencing method to IRT in order to find out the proper test length (number of test items used) for accurate ability estimations.

This paper is structured in five major sections. The next section describes the ERS and the IRT briefly. Section 3 is devoted to the presentation of our item difficulty estimation methods and their comparison. Section 4 compares IRT and ERS based adaptive item sequencing. Section 5 outlines our conclusions.

2. Theoretical background

2.1. **Elo Rating System.** The ERS was introduced for rating chess players by Arpad Elo in 1978 [5]. In this rating system each player has a rating, which represents their relative ability, thus a higher rating indicates a better player. Players are given an initial ability, which is continuously updated based on match results. The ERS system uses simple computations for updating players ratings.

If A, B are the two players with abilities (performances) Θ_A and Θ_B, formulas (1) and (2) are for ability estimations based on the match result. S_A is the match result for player A (0 loss, 0.5 - draw and 1 win) and $E(S_A)$ is the expected match result for player A in formula 3. S_B is the match result for player B, therefore $1 - S_A$ and $E(S_B)$ is the expected match result for player B (4). The K factor in formulas (1) and (2) weighs the performance change of a player during the matches and usually it is a constant.

$$\hat{\Theta}_A = \Theta_A + K(S_A - E(S_A)), \tag{1}$$

$$\hat{\Theta}_B = \Theta_B + K(S_B - E(S_B)), \tag{2}$$

$$E(S_A) = \frac{1}{1 + 10^{\frac{\Theta_B - \Theta_A}{400}}}, \tag{3}$$

$$E(S_B) = \frac{1}{1 + 10^{\frac{\Theta_A - \Theta_B}{400}}}, \tag{4}$$

In adaptive test systems one of the players is the examinee and the other one is the test item. Before answering the test item, the examinee has ability Θ and the test item has difficulty b, both being on the same scale, usually [-3, 3]. In this case a match is an answer given to the item, which can be correct or incorrect. We derive the new formulas for ability (5) and difficulty (6) estimates as follows

$$\hat{\Theta} = \Theta + K(S - E(S)) \tag{5}$$

$$\hat{b} = b - K(S - E(S)) \tag{6}$$

If the examinee gives a correct answer, then S is 1 and 0 otherwise. Thus for every correct answer the proficiency of the examinee increases and the difficulty of the answered item decreases and vice versa. In this study a constant value

0.4 [15] was used for K, although K could be chosen to reflect the uncertainty in ability estimates by making it a function of the number of answered items. For the expected match result $E(S)$ we used the logistic function given in formula (7).

$$E(S) = \frac{1}{1 + e^{-(\Theta - b)}} \tag{7}$$

2.2. **Item Response Teory.** IRT has its roots in Psychometrics and it defines a method for adaptive item selection. In this model each item is characterized by an Item Characteristic Curve (ICC, [10]), which shows the correct answer probability as a function of ability. ICC is usually modeled by a 3 parameter logistic function (8) also known as 3PL, where Θ represents student ability; a, b and c are item parameters representing discrimination, difficulty and guessing probability, and D is a scaling factor.

$$P(\Theta) = c + \frac{1 - c}{1 + e^{-Da(\Theta - b)}} \tag{8}$$

Replacing $c = 0$, $a = 1$ and $D = 1$ in formula (8) we obtain the 1PL model also known as the Rasch model, whose logistic function (9) is identical with (7).

$$P(\Theta) = \frac{1}{1 + e^{-(\Theta - b)}} \tag{9}$$

Items are administered based on their usefulness in ability estimation. For this purpose items are ranked on their item information values and the maximum one is chosen [10]. Item information for the 3PL model is given in formula (10).

$$I_i(\Theta) = \frac{P_i'(\Theta)^2}{P_i(\Theta)(1 - P_i(\Theta))} \tag{10}$$

The item information function for the 1PL model is

$$I_i(\Theta) = P_i(\Theta)(1 - P_i(\Theta)). \tag{11}$$

For new ability estimates we used the formulas (12) and (13) [10],

$$\Theta_{n+1} = \Theta_n + \frac{\sum_{i=1}^{n} S_i(\Theta_n)}{\sum_{i=1}^{n} I_i(\Theta_n)} \tag{12}$$

where $S_i(\Theta)$ is computed using the following formula:

$$S_i(\Theta) = (u_i - P_i(\Theta)) \frac{P_i'(\Theta)}{P_i(\Theta)(1 - P_i(\Theta))} \tag{13}$$

In formula (13) u_i is 1 if the answer for the ith item is correct and 0 otherwise. The standard error shows the accuracy of ability estimation, and after N administered items it can be computed by the following formula

$$SE(\Theta) = \frac{1}{\sqrt{\sum_{i=1}^{N} I_i(\Theta)}}. \tag{14}$$

3. Item difficulty estimation

Item parameters estimation is usually carried out prior to computerized adaptive testing and requires additional costs.

Not only the cost is an impediment but also the continuous addition of new test items, requiring continuous calibration. Stocking [11] observed that item calibration is very sensitive to examinee abilities, therefore non-compliant subjects can lead to incorrect item parameters. A few studies [8], [9], [15] tried to offer alternative and more lightweight solutions to item parameters estimation; moreover, [15] compared 6 alternative estimation methods with IRT-based calibration and found that Proportion Correct method has the strongest correlation with IRT-based difficulty estimates. Proportion Correct method is a simple approach, which estimates the difficulty level of items by dividing the number of incorrect answers by the number of total answers for the given item. In order to grasp the differences between various item difficulty estimation methods we conducted an experiment, which is partially similar to the one conducted by Wauters et al, 2011. We compared three item difficulty estimation methods: IRT, ERS and Proportion Correct using real test data.

3.1. **Participants.** Students from Computer Science, Automation and Applied Informatics and Informatics participated in this study. Data were collected during real examination sessions at Object-oriented programming. There

TABLE 1. Pearson correlation matrix of the item difficulty estimates for the OOP1 dataset

	IRT	ERS	Prop. Correct
IRT	1	0.942	0.994
ERS		1	0.937
Prop. Correct			1

TABLE 2. Pearson correlation matrix of the item difficulty estimates for the OOP2 dataset

	IRT	ERS	Prop. Correct
IRT	1	0.930	0.991
ERS		1	0.931
Prop. Correct			1

are two datasets: OOP1, collected in 2011 from 74 students and and OOP2, collected in 2012 from 63 students.

3.2. **Material and procedure.** The tests were administered using Moodle [16] and both tests consisted of 30 items (single choice, multiple choice, fill in). Students were familiarized with the Moodle test environment as they had already completed a pretest session. A test is considered reliable if repeated administration of the test give the same result. Test reliability was measured using Cronbachs alpha coefficient and we obtained 0.857 and 0.873 for datasets OOP1 and OOP2, respectively.

IRT model parameter calibration was conducted in R [17] using the ltm package. For Proportion Correct we used formula (15),

$$\hat{b_i} = 1 - \frac{n_i}{N_i} \tag{15}$$

where N_i and n_i are the total number of answers, the number of correct answers to the ith item respectively. Finally, for the ERS formulas (5), (6) and (7) were used.

3.3. **Results.** Figure 1 shows item difficulty estimations obtained by IRT, Proportion Correct and ERS methods for items in the OOP1 dataset. IRT estimation was conducted using the *rasch* function of the *ltm* package [17]. Using formula (15), the values obtained by the Proportion Correct method were scaled to the [-3,3] interval in order to be comparable to IRT estimations.

For ERS estimation each item difficulty was initialised by 0 and we used formulas (5), (6) and (7) in the estimation process.

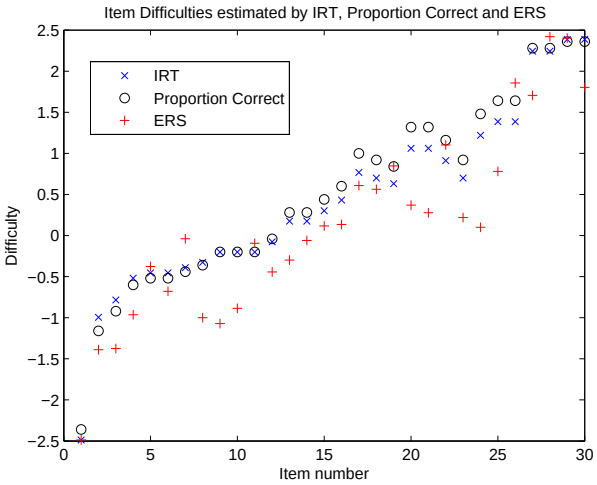


FIGURE 1. Item difficulty estimations by IRT, Proportion Correct and ERS methods (OOP1 dataset)

Tables 1 and 2 show Pearson correlations between the estimated item difficulty parameters using different estimation methods.

Proportion Correct has the strongest correlation with IRT calibration. However, all estimation methods produced difficulty estimates, which highly correlated with the other estimation methods.

4. Ability estimation using adaptive item sequencing

A computerized adaptive test is usually administered using the following steps:

(1) Administer a few items of average difficulty. Score the responses and estimate the person's initial ability level.
(2) Select the item that provides the most information using the person's current ability estimate and score the response.
(3) Re-estimate the person's ability level
(4) If stopping criterion is met, then stop the test, otherwise go to step 2.

The ability estimates obtained by IRT-based adaptive item sequencing are compared with ERS-based one using simulated data.

The only difference between the two methods is the re-estimation formula used for the person's ability, namely (12) for IRT and (5) for ERS. However, the ERS-based sequencing offers a formula for item difficulty estimation (6),

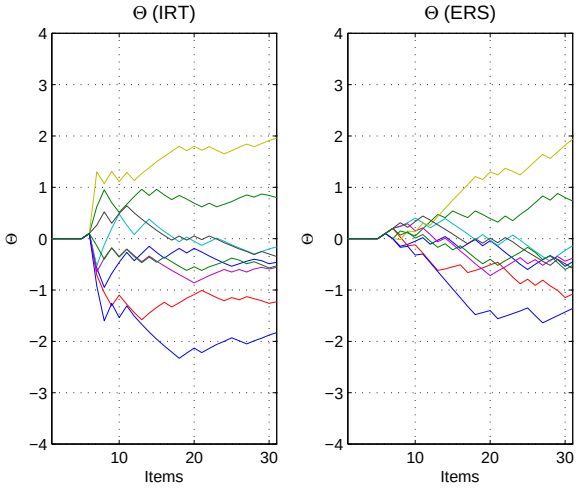


FIGURE 2. Ability estimations by IRT and ERS methods for the first 10 artificial examinees using a 30 item length test

which was not used in this experiment. The next item in both methods is the one having the maximum item information (10) for the person's ability.

There were 1000 artificial examinees simulated using an ideal item bank with 200 items having difficulties uniformly distributed in the [-3, 3] interval. In order to get comparable results, we generated the answer patterns for the examinees using a random number generator with uniform distribution. The first five items were randomly chosen and the estimation process was started after the 5th administered item. For each simulated examinee the same number of items were administered and the same predefined response pattern was used for both methods.

Figure 2 presents the results for the first 10 examinees, whereas the first two subfigures of figure 3 show ability values estimated during adaptive sequencing of 30 items using IRT and ERS methods for the 6th examinee. The third subfigure 3 of figure shows the answers given by the examinee. In this experiment we did not use stopping criteria, consequently all the 30 items were administered.

In the second experiment we compared IRT- and ERS-based adaptive sequencing methods from the viewpoint of ability estimates convergence. In order to achieve it we used various fixed length tests and compared the ability estimates obtained by the two methods at the end of tests. For test lengths we set the following values one by one: 10, 15, 20, 25 and 30. For each test length we repeated the experiment described above and obtained the final

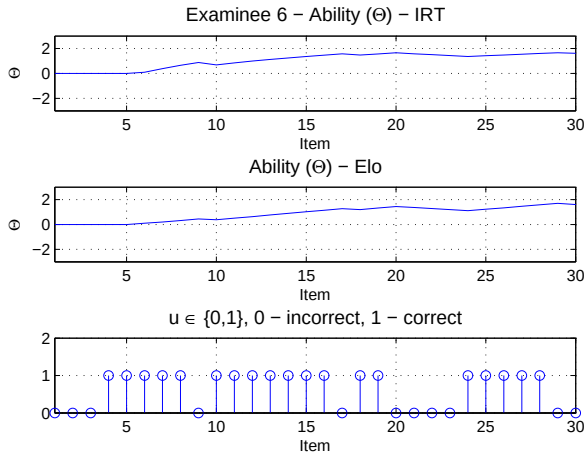


FIGURE 3. Ability estimations by IRT and ERS methods for a given examinee

TABLE 3. Influence of test length on ability estimations with IRT and ERS methods

Test length	$\mu(\Theta_{IRT} - \Theta_{ERS})$	$\sigma(\Theta_{IRT} - \Theta_{ERS})$	$\mu(SE)$	$\sigma(SE)$
10	0.68	0.52	0.46	0.034
15	0.49	0.41	0.35	0.033
20	0.31	0.27	0.30	0.011
25	0.23	0.20	0.26	0.009
30	0.18	0.14	0.23	0.007

ability estimates for IRT and ERS methods after the end of the test. Table 3 summarizes the results of the experiment: the first column contains the test length, the second and the third columns contain the mean and the standard deviation of $\Theta_{IRT} - \Theta_{ERS}$ for the 1000 examinees. Columns 4 and 5 show the mean standard error and its standard deviation computed in the case of IRT estimate. It can be seen that IRT estimates the examinees ability faster than ERS. For the IRT method the standard error is considered a measure of the accuracy of the ability estimate and this value falls below 0.4 for a test containing approximately 15 items. For a 30 item length test the estimated abilities by the two methods are very close, the average difference is 0.18 with a standard deviation of 0.14.

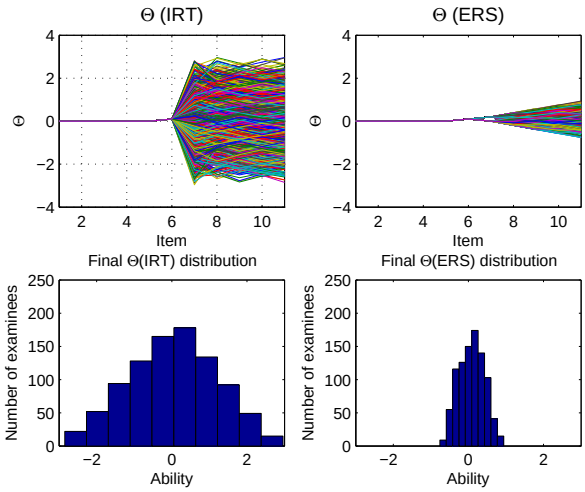


FIGURE 4. Ability estimations by IRT and ERS methods for 1000 artificial examinees using a 10 item length test

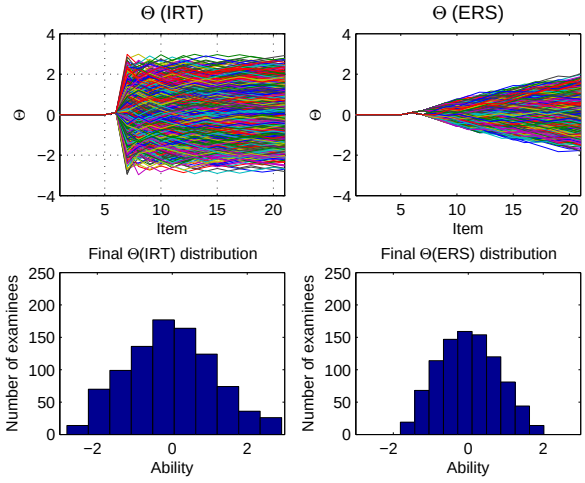


FIGURE 5. Ability estimations by IRT and ERS methods for 1000 artificial examinees using a 20 item length test

Figures 4, 5, 6 show the results of adaptive item sequencing simulations for 10, 20 and 30 items. On the top left figure Θ is estimated by IRT using formula (12), while on the top right figure by ERS using formula (5). Bottom figures

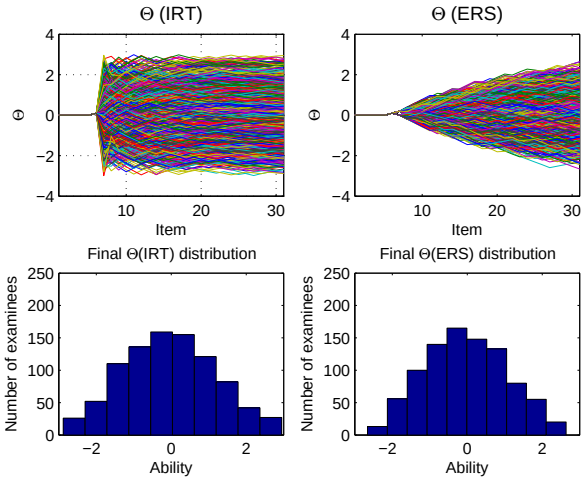


FIGURE 6. Ability estimations by IRT and ERS methods for 1000 artificial examinees using a 30 item length test

show the histograms of ability estimates. These figures show the influence of the test length on ability estimations.

5. CONCLUSIONS

In this paper we presented alternative methods for item difficulty estimation. Using real test data we compared two alternative item difficulty estimation methods to IRT-based estimation. The high correlations between difficulty estimates obtained by IRT, ERS and Proportion Correct methods indicate that any of these can be used in real adaptive test systems. We also presented a new model for computerized adaptive testing: an IRT-based adaptive item sequencing with ERS-based ability estimate. We found that the new model is able to obtain a reliable examinee ability estimate using a test of at least 30 items. This result is in concordance with the finding obtained by van der Maas and Wagenmarkers regarding Elo rating used for ranking chess players, as they concluded that 25 games are needed to obtain a reliable Elo rating for a chess player [13]. Moreover, using this model we can obtain both the ability estimates of the examinees and item difficulty estimates of the test items. One limitation of this study is the size of population used for item difficulty estimation. Therefore we are planning to repeat these estimations as we gather more data.

6. Acknowledgements

This research has been supported by Sapientia Institute for Research Programmes.

References

[1] Brinkhuis, M. J. S., Maris, G. *Dynamic Parameter Estimation in Student Monitoring Systems.* CITO-report (2009).

[2] Brusilovsky, P., Millan, E. *User models for adaptive hypermedia and adaptive educational systems.* In: P. Brusilovsky, A. Kobsa and W. Neidl (eds.): The Adaptive Web: Methods and Strategies of Web Personalization. Lecture Notes in Computer Science, Vol. 4321, Berlin Heidelberg New York: Springer-Verlag, pp. 3-53 (2007).

[3] Cronbach, L.J. *Coefficient Alpha and the internal structure of tests.* Psychometrika 16(3), pp. 297-334 (1951).

[4] Economides, A.A., Roupas, C. *Evaluation of computer adaptive testing systems.* International Journal of Web-Based Learning and Teaching Technologies 2(1), pp. 70-87 (2007).

[5] Elo, A. E., *The rating of chess players, past and present.* B.T. Batsford, Ltd., London (1978).

[6] Impara, J. C., Plake, B. S. *Teachers' ability to estimate item difficulty: A test of the assumptions in the Angoff standard setting method.* Journal of Educational Measurement, 35(1), pp. 69-81 (1998).

[7] Kibble, J. D., Johnson, T. *Are faculty predictions or item taxonomies useful for estimating the outcome of multiple-choice examinations?* Advances in Physiology Education 35, pp. 396-401 (2011).

[8] Kingsbury, G. *Adaptive Item Calibration: A Process for Estimating Item Parameters Within a Computerized Adaptive Test.* In D. J. Weiss (Ed.), Proceedings of the 2009 GMAC Conference on Computerized Adaptive Testing (2009).

[9] Klinkenberg, S., Straatemeier, M., van der Maas, H. L. J. *Computer adaptive practice of Maths ability using a new item response model for on the fly ability and difficulty estimation.* Computers & Education 57(2), pp. 1813-1824 (2011).

[10] Rudner, L. M. *An online, interactive, computer adaptive testing tutorial.* http://echo.edres.org:8080/scripts/cat/catdemo.htm (1998).

[11] Stocking,M. L. *Specifying optimum examinees for item parameter estimation in Item Response Theory.* Psychometrika 55(3), pp. 461-475 (1990).

[12] Van der Linden, W. J., Hambleton, R. K. *Handbook of modern item response theory.* New York, Springer (1997).

[13] Van der Maas, H. L., Wagenmakers, E.-J. *A psychometric analysis of chess expertise* American Journal of Psychology 118(1), pp. 29-60 (2005).

[14] Wauters, K., Desmet, P., Van Den Noortgate, W. *Acquiring Item Difficulty Estimations: a Collaborative Effort of Data and Judgment.* Educational Data Mining EDM pp. 121-128 (2011).

[15] Wauters, K., Desmet, P., Van Den Noortgate, W. *Item difficulty estimation: An auspicious collaboration between data and judgement.* Computers & Education 58(4), pp. 1183-1193 (2012).

[16] http://moodle.org/

[17] http://www.r-project.org/

Sapientia - Hungarian University of Transylvania, Faculty of Technical and Human Sciences, 540053 Tirgu Mures, 540485, Aleea Sighisoarei 1C., Romania
E-mail address: manyi@ms.sapientia.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVIII**, Number 1, 2013

A STUDY ON ASSOCIATION RULE MINING BASED SOFTWARE DESIGN DEFECT DETECTION

ZSUZSANNA MARIAN

Abstract. In this paper we are investigating the effect of parameter variations for a method we have previously introduced for detecting software design defects. This method uses software metrics and relational association rules to find badly designed classes. We perform five different studies, to see the effect of using normalized or original software metric values, the effect of mining only binary or any-length rules, the effect of mining only maximal or all rules and the effect of changing the value of the minimum support for the rules. We are also investigating the changes caused by modifying the value of the parameter that determines which classes to report as having bad design.

1. Introduction

Software systems developed and used in our days are more and more complex, because the tasks they have to solve are getting more and more complex as well. In these systems both maintenance and the addition of new features is a complicated task, which is even more complicated if the system has a bad design. This is why different techniques exist that try to identify design defects in a system, in order to correct them, thus making maintenance easier.

In [3] we have presented a novel method for design defect detection, called *Software Design Defect detection using Relational Association Rules* or SDDRAR. This approach uses software metrics, which are often used for software design defect detection, but it also uses Relational Association Rules, a particular type of association rules, which were not used so far for this task.

We have provided six different open source case studies in [3], to show the effectiveness of our method. We have also presented some similar approaches form literature and compared the SDDRAR method to them. In this paper,

2010 *Mathematics Subject Classification.* 68P15, 68N99.

1998 *CR Categories and Descriptors.* H.2.8 [**Database Management**]: Database Applications - Data Mining; D.2.10 [**Software Engineering**]: Design;

Key words and phrases. Relational association rules, Software metrics, Design defect detection.

we intend to present a study performed with different parameter settings for our method. We want to investigate the following aspects:

- Using the original or normalized values for software metrics.
- Using binary relational association rules or using relational association rules of any length.
- Using only maximal relational association rules, or using all mined rules.
- The effect of modifications for the τ parameter and the minimum confidence on the prediction accuracy.

The rest of the paper is structured as follows: Section 2 presents the theoretical background for the SDDRAR approach, presenting Relational Association Rules (Section 2.1), the used software metrics (Section 2.2), the SDDRAR method (Section 2.3) and the open source projects used for testing (Section 2.4). Section 3 presents the studies performed with different variations of the parameters. Section 4 briefly presents some similar approaches and compares our method to them, while Section 5 concludes the paper.

2. Background

In this section we will present the main theoretical background of the SDDRAR method. A more detailed description was given in [3].

2.1. **Relational Association Rules.** Relational association rules are an extension of regular association rules and are able to capture different types of relationships between record attributes. They were introduced in [16], and can be formally defined in the following way: let $R = \{r_1, r_2, ..., r_n\}$ be a set of instances, where each instance is characterized by a vector of m attributes: $r_i = \{a_1, a_2, ..., a_m\}$. Each attribute a_i takes value from the domain D_i, and the value of an attribute a_i in an instance r_j is denoted by $\Phi(r_j, a_i)$. Between two domains D_i and D_j different relations can be defined, for example equal, less than, greater than, etc. We denote by M the set of all possible relations that can be defined on $D_i x D_j$.

Using the notations presented above, a relational association rule is of the form $(a_{i_1}, a_{i_2}, \ldots, a_{i_l}) \Rightarrow (a_{i_1}\ \mu_1\ a_{i_2}\ \mu_2\ a_{i_3}\ \ldots\ \mu_{l-1}\ a_{i_l})$ where $\{a_{i_1}, \ldots, a_{i_l}\} \subseteq A = \{a_1, a_2, ..., a_m\}$, $a_{i_j} \neq a_{i_k}$, $j, k = 1..l, j \neq k$ and $\mu_i \in M$ is a relation over $D_{i_j} \times D_{i_{j+1}}$, D_{i_j} being the domain of attribute a_{i_j}.

Like regular association rules, relational association rules are also characterized by two values, *support* and *confidence*, defined in the following way:

- If $a_{i_1}, a_{i_2}, \ldots, a_{i_l}$ occur together (are non-empty) in s% of the instances, we call s the *support* of the rule.

- If $R' \subseteq R$ is the set of instances where $a_{i_1}, a_{i_2}, \ldots, a_{i_l}$ occur together and $\Phi(r\prime, a_{i_1})\ \mu_1\ \Phi(r\prime, a_{i_2})\ \mu_2\ \Phi(r\prime, a_{i_3})\ \ldots\ \mu_{l-1}\ \Phi(r\prime, a_{i_l})$ is true for each instance $r\prime \in R'$, we call $c = \frac{|R'|}{|R|}$ the *confidence* of the rule.

Another value that characterizes a relational association rule is its *length*, which is the number of attributes in the rule. The minimum possible length for a rule is 2, while the maximum possible length is the number of attributes.

In a dataset many association rules can be found, so usually only those are mined which have support and confidence higher than some user specified threshold, s_{min} and c_{min}. These rules are also called *interesting*. In [3] we have introduced an A-Priori [1] like algorithm, called *DRAR*, which can find all the interesting relational association rules in a dataset. Also, the algorithm can be configured to find only the maximal interesting relational association rules (rules which cannot be further extended with an attribute, because they will no longer be interesting) in the dataset.

2.2. **Software Metrics.** In order to mine relational association rules from a dataset, one needs a set of instances, where each instance is actually a vector of attributes. In our model, the instances were classes of the software system, while the attributes that characterize these instances are the values of different software metrics. The set of software metrics is denoted by $SM = \{sm_1, sm_2, \ldots, sm_k\}$. So, we consider a software system S as being a set of instances (classes) $S = \{s_1, s_2, \ldots, s_n\}$, where each instance s_i is represented as a k-dimensional vector $s_i = \{s_{i_1}, s_{i_2}, \ldots, s_{i_k}\}$, where s_{i_j} is the value of the software metric sm_j for the instance s_i.

We have identified in [10] a set of 16 software metrics that measure the size, cohesion and coupling in a software system. After performing some statistical analysis on this set in [3], after which we eliminated 4 of them, the set of software metrics became $SM = \{CBO, DAC, ICH, INS, LCC, LCOM1, LCOM2, LCOM4, LCOM5, LD, MPC, NOA\}$. A description of these metrics and the statistical analysis approach performed are presented in [3].

2.3. **The SDDRAR method.** The aim of the SDDRAR method is to identify classes with design defect in a software system, using relational association rules. It has three different steps, which will be presented briefly in this section.

- Data collection and pre-processing.
- Building the SDDRAR model
- Testing

In the first step, *Data collection and pre-processing*, a set S_{good} of well-designed software systems is collected. Next, the k-dimensional representation of the classes (entities) from these software systems is built, denoted by DS, using the

set of software metrics presented in Section 2.2. Then, the already mentioned statistical analysis on the set of initial software metrics is performed.

During the second step, *Building the SDDRAR model*, all interesting relational association rules of any length are discovered in the DS dataset. The relations used between the attributes are $\leq$ and $\geq$ and are not defined for metrics with the value 0. These rules, kept in a set called RAR, will be used to identify classes (entities) with bad design in other software systems.

The last step, *Testing*, is the most complex. In this step, a new software system, S_{new} has to be analyzed to detect those entities that have a bad design. First, the k-dimensional representation of the classes from S_{new} is built (using the same software metrics as above). Then, for each entity $e_i \in S_{new}$ the *number of errors*, $err(e_i)$, is computed, as the number of relational association rules from RAR that are not verified by the k-dimensional representation of the entity. Next, the *percentage of error*, $pe(e_i)$, is computed, as: $pe(e_i) = \frac{err(e_i)}{|RAR|}$. After this, the set of *potential errors*, P_τ, is determined, containing those entities, for which $pe(e_i)$ is greater than a user specified threshold, τ. We have two possible cases:

(1) If this set is empty, a threshold ϵ is computed as the sum between the mean and the standard deviation of the non-zero number of errors for all entities. The SDDRAR method will report as ill structured software entities in S_{new} the ones which have $err(e) > \epsilon$.
(2) If P_τ is not empty, than the average number of errors of the entities from P_τ, avg, is computed and the algorithm will report as ill structured software entities the ones, for which $err(e) > avg$.

2.4. **Open source projects used.** In this section we will shortly present those open source projects that were used in [3] and will be used in this paper, too. First, as presented above, the SDDRAR method needs a set of well designed software systems, S_{good}, in order to build the set of relational association rules. In our current implementation this set contains one single element, the *JHotDraw* [5] software system, built by Erich Gamma and Thomas Eggenschwiler. It is considered an example of good design and use of design patterns. It consists of 173 classes, out of which only 132 are used, because the rest are interfaces, for which the value of some metrics cannot be computed.

We have also used 6 different systems for testing our method: two simple artificial examples and 4 open source systems, taken from the SourceForge repository. For these 4, we have also considered consecutive versions, to see how reported classes and the number of errors change as the project evolves. These 4 systems are the following:

- FTP4J, [4], a Java implementation of a full-featured FTP client. 4 consecutive versions were considered, 1.5, 1.5.1, 1.6, 1.6.1, each having 27 classes (and 8 interfaces which were not included in the analysis).
- ISO8583, [6], an implementation of the ISO 8583 protocol in Java. Three versions were used, 1.5.2, 1.5.3, 1.5.4, each containing 21 classes (and 2 interfaces).
- Profiler4J, [15], a CPU profiler for Java, which supports remote profiling and on-the-fly configuration. From the two jar files of the project, we used four versions of the *agent.jar* file: 1.0-alpha5, 1.0-alpha6, 1.0-alpha7 and 1.0-beta1. The first two versions have 18 classes, while the last two have only 15.
- WinRun4J, [17], a Windows native launcher for Java applications. Five consecutive versions were analyzed, 0.4.0, 0.4.1, 0.4.2, 0.4.3 and 0.4.4. The first three versions have 21 classes, while the last two have 24.

All projects (including JHotDraw) were analyzed using the ASM 3.0 bytecode manipulation framework [2]. We use this framework to extract a representation of a software system containing classes, methods and attributes and the relations between them. This representation is then used to compute the value of the software metrics for the classes.

3. Study on the parameter variations

The SDDRAR method presented in Section 2.3 has many different parameters, whose value can influence the results. For example, the parameters of the relational association rule mining algorithm are the s_{min} and c_{min} values, but also whether the algorithm should find only binary rules or rules of any length, or whether all mined rules should be kept, or only the maximal ones. Related to the software metrics, one parameter is, whether the original or the normalized value of the metrics should be used for the relational association rule mining. And for the SDDRAR algorithm, the value of τ is very important, because this is the one that determines the reported classes. In this section we will present some experiments and analysis of the results, when changing the values of these parameters. The first three experiments were performed with $c_{min} = 0.85$ and $s_{min} = 0.9$.

3.1. **Normalized vs. Original software metrics values.**

The first test we are going to execute is to see how using normalized or non-normalized (called original) software metrics values influences the results of the algorithm.

After computing the values of the software metrics for the JHotDraw system (both normalized and original values), we ran our algorithm to extract the set of maximal relational association rules of any length. The number of

rules for each length and the total number of rules for both cases can be seen in the first two columns of Table 1.

TABLE 1. Number of rules for different rule mining settings for the jHotDraw system.

Length	Normalized, Any-length, Maximal	Original	Binary	All-mined
2	0	0	14	14
3	21	5	0	39
4	4	6	0	22
5	16	160	0	18
6	1	0	0	1
Total	42	171	14	94

We have also divided the relational association rules into simple binary rules, to see which pairs make up the longer rules. The exact pairs are omitted because of lack of space, but analyzing them, we observed that there is only one rule which is common ($LCOM5 \leq LD$), but there are six rules which appear with one relation when using the original values, and appear with the inverse relation, when normalized values are used. Also, the presence of some of the rules for the original metric values is easy to understand. The 12 software metrics used can be divided into two categories: metrics whose value is between 0 and 1 (ICH, INS, LCC, LCOM5 and LD), and metrics whose value can be greater than 1 (CBO, DAC, LCOM1, LCOM2, LCOM4, MPC and NOA). When the metric values are not normalized, it is logical to have many cases when the value of a metric from the first category is less than the value of a metric from the second category. Indeed, if we check the rules for the original values, we can see that for 12 out of the 19 rules this is the case, while for normalized values there is no such rule. This shows that the relational association rules found for the original values uncover a lot less hidden patterns in data than the normalized values.

The mined rules were used to find badly designed classes in the FTP4J software system, version 1.5, with $\tau = 0.8$. The reported classes are presented in the first two columns of Table 2. We can see, that the two variants report very different results. As presented in [3], the *FTPClient* class should be reported, because it is a God Class, having above 3500 lines of code (with comments), 34 attributes (second highest value in the system being 7) and 84 methods (second highest is 15). If we verify the classes reported when using the original metric values, we can see that classes *Base64OutputStream*, *SOCKS4Connector* and *SOCKS5Connector* should not be reported, because

they are not badly designed. The reason why they are reported is that they use no other class from the system, so their CBO and MPC values are 0, causing a lot of errors. The class *FTPFile* can be considered a Data Class (5 attributes, and 11 methods - getters, setters and toString), but most of its errors are because of the CBO and MPC metrics, just like in case of the previous classes, suggesting that there is no pattern in errors for finding Data Classes (and nor for God Classes, because FTPClient was not identified either).

TABLE 2. Classes reported by the SDDRAR method for the FTP4J system for different parameter settings.

Normalized Any-length Maximal	Original	Binary	All-mined
FTPClient	FTPFile Base64OutputStream SOCKS4Connector SOCKS5Connection	FTPClient	FTPClient DirectConnector FTPDataTransfer- Exception

Considering the above presented case study, we can conclude, that it is better to use normalized software metric values, both because they predict bad designs better, and because they can find hidden patterns in the metric values better.

3.2. **Binary vs. Any-length relational association rules.** In this section we will investigate the effect of using only binary, or any-length rules. First, for the *JHotDraw* system, we have generated all maximal rules, once only the binary ones, and after that rules of any length.The number of rules for each length and the total number of rules for both cases can be seen on the first and third column of Table 1.

Dividing the any-length rules in binary pairs, we get the exact same pairs as for the normalized values in the previous Section (which is normal, because they were mined from the same data). They are also the same rules mined when only binary rules are considered. When using these rules to identify badly designed classes in the FTP4J software system, version 1.5 and $\tau = 0.8$, in both cases a single class is identified: *FTPClient* (as it is presented in the first and third column of Table 2). The difference is that when the binary rules are used, case 1 from 2.3 has to be used ($P_{0.8} = \emptyset$). Although the results are the same, both for binary and any-length relational association rules, we think that using longer association rules is better, because the fact that a binary relation can appear more than once in the set of rules, provides kind of a weighting mechanism, making binary relations that appear more than once,

more important. For example, out of those classes from FTP4J which have one error when using the binary rules, some have 2 errors when using longer rules, while others have 9, meaning that they broke relations, which appear in more rules, so they can be considered more important. So, we conclude that it is better to use longer relational association rules.

3.3. **Maximal vs. All-mined relational association rules.** In this section we will investigate the effect of using only the maximal relational association rules, or using all the mined rules, even if later they were extended with other metrics. The number of such rules in the *jHotDraw* system is presented in the last column of Table 1.

The results of using these rules for finding badly designed classes in the *FTP4J* software system, version 1.5, with $\tau = 0.8$ is presented in the last column of Table 2. In case of the all-mined rules, $P_{0.8} = \emptyset$, so case 1 from 2.3 is used.

As already presented above, the identification of the *FTPClient* class is justified, because it is a God Class. The other two classes identified using all the mined rules have the exact same errors, for the CBO, INS, LCOM1 and LCOM4 metrics. Verifying the source code, we can see that these classes are simple classes with 3 or 4 very short (one line) methods, but no design defect can be found in them. So we can conclude that using only maximal rules is better.

3.4. **Changing the value of τ.** In the previous three Sections, we have conducted three different studies, and concluded that it is better to use normalized software metric values for mining relational association rules, and that mining any-length but only maximal rules gives better results. In this Section we will investigate the effect of changing the value of τ, the parameter which determines which classes to report as badly designed. Also, instead of using only one system, we are going to use all four software systems presented in 2.4 with all their versions.

It is expected that lowering the value of τ will result in more classes being reported as having errors, while increasing it will lead to less reported classes. Although, if τ is too high, it might happen that $P_\tau = \emptyset$ (case 1 from 2.3), which can lead to a situation, when an increased τ leads to more reported classes. After this point, increasing the value of τ will not give different results.

During the experiments we varied the value of τ from 0.2 to $0.8 - 0.95$ (depending when we got to the point when there was no use increasing it anymore) with increments of 0.05. The results (identified classes) are presented on Tables 3, 4, 5 and 6. A star after the name of an identified class means that it was identified using case 1 from 2.3. Cases when the result did not change

for more consecutive values of τ are presented with the interval for which the values are true.

TABLE 3. Identified classes for the *FTP4J* project for different values of τ.

τ	1.5, 1.5.1, 1.6, 1.6.1
0.9	*FTPClient**
0.85 - 0.25	*FTPClient*
0.2	*FTPClient* *DirectConnector* *FTPDataTransferException*

From Table 3 we can see that the results are quite stable, the value of τ has to be decreased until 0.2 in order to report classes *DirectConnector* and *FTPDataTransferException*, which, as presented in Section 3.3, are not badly designed.

TABLE 4. Identified classes for the *ISO8583* project for different values of τ.

τ	1.5.2, 1.5.3, 1.5.4
0.95	*MessageFactory**
0.9 - 0.65	*MessageFactory*
0.6 - 02	*MessageFactory* *ISOValue*

From Table 4 we can see that when decreasing the value of τ to 0.6 (and lower) a new class is reported, *ISOValue*. Although this class has some minor design defects (for example calling the getters in the *equals* method, instead of using the fields directly) there are no big problems with it.

Table 5 presents the classes reported for the *Profiler4J* project. In [3] we have presented that the classes reported for $\tau = 0.8$ (*Server* for version 1.0-alpha5, *MemoryInfo* for version 1.0-alpha6 and *Config* for versions 1.0-alpha7 and 1.0-beta1) are justified. *Server* has too high coupling, *MemoryInfo* is a Data Class (which actually disappears after version 1.0-alpha6) and *Config* can be considered a Data Class, too. Besides these three classes, only *ThreadInfo* is reported, but only for low values of τ. Checking the source code, we can see that the class has many static methods, but does not really have errors.

Table 6 presents the classes reported for the WinRun4J system. In [3] we considered the *NativeBinder* class as having smaller design defects (a too long

TABLE 5. Identified classes for the *Profiler4J* project for different values of τ

.

τ	1.0-alpha5	1.0-alpha6	1.0-alpha7, 1.0-beta1
0.9	*Server** *MemoryInfo**	*Config** *MemoryInfo**	*Config**
0.85	*Server*	*Config** *MemoryInfo**	*Config**
0.8 - 0.75	*Server*	*MemoryInfo*	*Config*
0.7 - 0.6	*Server* *MemoryInfo*	*MemoryInfo*	*Config*
0.55 - 0.5	*Server* *MemoryInfo*	*Config* *MemoryInfo*	*Config*
0.45	*Config* *Server* *MemoryInfo*	*Config* *MemoryInfo*	*Config*
0.4 - 0.3	*Config* *Server* *ThreadInfo* *MemoryInfo*	*Config* *MemoryInfo*	*Config* *ThreadInfo*
0.25 - 0.2	*Config* *Server* *ThreadInfo* *MemoryInfo*	*Config* *Server* *ThreadInfo* *MemoryInfo*	*Config* *ThreadInfo*

method, high coupling, etc.), instead of having just one main problem. For $\tau = 0.6$ and less, the class *Launcher* is also reported. It is a class with many methods, and many overloaded methods, which call each other, as suggested by the errors for the ICH metric. High value of LCOM1 metric (and many errors related to this metric) suggest that the class is not really cohesive. Another reported class is *Closure*, but besides having a really long method, this class is fine. Finally, the *FileVerb* class reported for $\tau = 0.45$ and lower, is a Data Class.

Considering the results for the above presented four systems, we can say that small differences in the value of the τ parameter will not result in big changes in the classes. As we expected, lowering the value of τ will result in more classes reported, but not a lot more. In [3] we have shown that the results given for $\tau = 0.8$ are correct (those classes indeed have problems), and now we presented that most of the other classes reported (for lower values of

TABLE 6. Identified classes for the *WinRun4J* project for different values of τ.

τ	0.4.0 - 0.4.4
0.8 - 0.75	*NativeBinder**
0.7 - 0.65	*NativeBinder*
0.6 - 0.55	*NativeBinder* *Launcher*
0.5	*Closure* *NativeBinder* *Launcher*
0.45 - 0.2	*Closure* *NativeBinder* *FileVerb* *Launcher*

τ) also have design problems to a given extent. This suggests that if there is sufficient time for an analysis, it might be worth lowering the value of τ to get not only the class with the biggest problems, but also other classes with smaller ones.

3.5. **Changing the value of c_{min}.** .

In this section we are going to investigate the effect of changing the value of the minimum confidence (so far we have performed every test with the value of 0.85). While changes in the value of τ influence which classes are reported as having a bad design, but each class had the same number of errors (as shown in the previous section), changes in the value of c_{min} influence which rules are mined, and consequently the number of errors for each class. For these experiments we will use the value of $\tau = 0.8$.

First we wanted to analyze how the number and length of rules changes when the value of c_{min} is increased or decreased. For values between 0.9 and 0.6 with decrements of 0.05 the number of rules is presented in Table 7. We can see that as the minimum confidence decreases, the number of mined rules increases drastically. The maximum length of the rules increases, too.

The results of using the relational association rules mined for the different values of c_{min} for finding badly designed classes in the open source projects are presented on Tables 8, 9, 10, 11. Looking at the tables, we can see that as the number of rules increases, so does the number of reported classes.

In case of the *FTP4J* project (presented on Table 8), besides the well-justified *FTPClient* class, four other classes are reported. Out of these, as

TABLE 7. The number and length of rules for different values of c_{min}.

Confidence	3	4	5	6	7	8	Total
0.9	21	0	0	0	0	0	21
0.85	21	4	16	1	0	0	42
0.8	7	3	132	110	0	0	252
0.75	4	7	113	208	60	0	392
0.7	0	4	132	303	207	30	676
0.65	20	16	111	308	443	146	1044
0.6	25	22	127	506	696	286	1662

TABLE 8. Identified classes for the *FTP4J* system for different values of c_{min}.

c_{min}	1.5 - 1.5.1	1.6 - 1.6.1
0.9 - 0.8	*FTPClient*	*FTPClient*
0.75	*FTPClient* *SOCKS4Connector* *SOCKS5Connector*	*FTPClient* *SOCKS4Connector* *SOCKS5Connector*
0.7	*FTPClient* *DirectConnector* *SOCKS4Connector* *SOCKS5Connector*	*FTPClient* *DirectConnector* *SOCKS4Connector* *SOCKS5Connector*
0.65-0.6	*FTPClient* *DirectConnector* *FTPFile* *SOCKS4Connector* *SOCKS5Connector*	*FTPClient* *FTPFile* *SOCKS4Connector* *SOCKS5Connector*

described in previous Sections, only the class *FTPFile* is reported correctly, because it is a Data Class.

The results given for the *ISO8583* project are really interesting. For the other projects, a class which is reported for a higher value of c_{min} will be reported for the lower values, too (there is one exception for the *FTP4J* project, and one for *Profiler4J*), but here classes keep appearing and disappearing. There are three classes which are reported: *MessageFactory*, *ISOValue* and *ISOType*. We have already presented that *MessageFactory* is correctly identified, but *ISOValue* is not. *ISOType* is actually an enum, with many methods, many of them overloaded, but there are no outstanding problems with it.

TABLE 9. Identified classes for the *ISO8583* system for different values of c_{min}.

c_{min}	1.5.2	1.5.3 - 1.5.4
0.9 - 0.85	*MessageFactory*	*MessageFactory*
0.8 - 0.75	*ISOValue* *MessageFactory*	*ISOValue* *MessageFactory*
0.7	*ISOValue* *ISOType*	*ISOValue* *ISOType*
0.65	*ISOValue* *ISOType* *MessageFactory*	*ISOValue* *ISOType* *MessageFactory*
0.6	*ISOValue* *MessageFactory*	*ISOValue* *ISOType* *MessageFactory*

TABLE 10. Identified classes for the *Profiler4J* system for different values of c_{min}.

c_{min}	1.0-alpha5	1.0-alpha6	1.0-alpha7 1.0-beta1
0.9	*Server*	*MemoryInfo**	*Config**
0.85	*Server*	*MemoryInfo*	*Config*
0.8	*Config* *Server* *MemoryInfo*	*Config* *MemoryInfo*	*Config*
0.75	*Config* *Server* *MemoryInfo* *Response*	*Config* *CFlow* *MemoryInfo* *Response*	*Config* *CFlow*
0.7-0.6	*Config* *Server* *ThreadInfo* *MemoryInfo* *Response*	*Config* *ThreadInfo* *MemoryInfo* *Response*	*Config* *ThreadInfo* *CFlow*

The reason for the fact that some reported classes disappear when the value of c_{min} decreases (and sometimes they appear again) is caused by the fact, that for different values of c_{min} the percentage of binary rules in which a metric appears can change a lot. For example, when $c_{min} = 0.85$, the LD

TABLE 11. Identified classes for the *WinRun4J* system for different values of c_{min}.

c_{min}	0.4.0 - 0.4.4
0.9	*NativeBinder*
0.85	*NativeBinder**
0.8	*NativeBinder* *Launcher*
0.75 - 0.6	*NativeBinder* *FileVerb* *Launcher*

metric appears in 29.26% of the binary rules, while for $c_{min} = 0.6$ this value is only 17.56%. Because of these changes the order of classes based on their number of errors changes too. For example, for version 1.5.2 , for $c_{min} = 0.9$, *MessageFactory* has more errors than *ISOValue* and *ISOType* together. When $c_{min} = 0.8$, *ISOValue* has more errors than *MessageFactory*, and *ISOType* has the least (out of these three). When $c_{min} = 0.65$, *MessageFactory* is the one with the least errors and *ISOValue* has the most. Finally, when $c_{min} = 0.6$, *ISOValue* has again the most, but now *ISOType* has the least errors.

The results for the *Profiler4J* software system are presented on Table 10. Here we also have some classes that were not reported during the previous studies: *Response* and *CFlow*. *Response* is a simple Data Class, with two fields and two getters, while *CFlow* is a class with only one short method and an inner class (which is not included in our analysis).

For the *WinRun4J* project, the reported classes are almost the same as in the previous Section, when the value of τ was changed. As already mentioned there, *FileVerb* is a Data Class andd *Launcher* is not really cohesive.

Verifying the results for all software systems, we can observe that for the lower values of c_{min} (between 0.7-0.6) three out of the four projects report new Data Classes: *FileVerb* in case of *WinRun4J*, *FTPFile* in case of *FTP4J* and *Response* in case of *ISO8583*. This suggests that further analysis might be useful, to see, if it is possible to identify a value or an interval for c_{min}, where it can identify Data Classes in the system.

4. Comparison to Related Work

There are many different approaches that try to identify design defects in software systems, presented in the literature. Most of them use software metrics and predefined thresholds for their values, like Marinescu's *detection strategies* [9], or Munro's method, presented in [14]. In a series of papers,

Moha et. al presents the idea of *rule cards*, which contains both metric values, but also semantic and structural information [12], [11], [13]. Rule cards were extended by Khomh et. al, in order to handle uncertainty [8]. They use Bayesian Belief Networks and assign to each class a probability that it contains a given design defect. In [7] three different search techniques (Harmony Search, Particle Swarm Optimization and Simulated Annealing) are used for finding rules that describe design defects, and can later be applied to classes. These rules are made of software metrics and threshold values for them.

Just like the above presented methods, our approach uses software metrics, but it also uses Relational Association Rules, which, according to our knowledge, have never been used for design defect detection yet. Another important difference between them, is that our method uses the relation between the values of different metrics, while the above presented ones use fixed thresholds, which can be hard to determine, because "good" and "bad" values for a metric usually depend on the size of the software system. On the other hand, the above presented methods are capable of identifying different, well-defined software smells, like Data Class or God Class, while our method can only identify the class with the design problem.

5. Conclusions

In this paper we have presented a study on the effect of changes for different parameters for the SDDRAR method, an approach for detecting design defects in software systems, using software metrics and relational association rules. We have considered five different possible changes, and reported and analysed the results on open source software systems. We showed that it is better to use normalized software metric values for mining relational association rules, that it is better to use any-length association rules (not just binary) and that using maximal rules is better. We have also shown that the parameter values used in [3] for c_{min} and τ (0.85 and 0.8, respectively) are good values, but interesting results could be achieved with different values, too.

The last study performed (when consequences of changes in the value of c_{min} were tested) could be further analyzed. It would also be worth investigating how the binary rules and their number change for different values of c_{min}. We have already identified a possible pattern, that for lower values Data classes are found, but this should be tested.

References

[1] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[2] ObjectWeb: Open Source Middleware, 2012. http://asm.objectweb.org/.
[3] Gabriela Czibula, Zsuzsanna Marian, and Istvan Gergely Czibula. Detecting software design defects using relational association rule mining. *Knowledge and Information Systems*, 2012. Under review.
[4] Ftp4j, 2012. http://sourceforge.net/projects/ftp4j/.
[5] E. Gamma. JHotDraw Project, 2012. http://sourceforge.net/projects/jhotdraw.
[6] Iso8583, 2012. http://sourceforge.net/projects/j8583/.
[7] Marouane Kessentini, Houari Sahraoui, Mounir Boukadoum, and Manuel Wimmer. Search-based design defects detection by example. In *Proceedings of the 14th International Conference on Fundamental Approaches to Software Engineering*, pages 401–415, 2011.
[8] Foutse Khomh, Stéphane Vaucher, Yann-Gaël Guéhéneuc, and Houari Sahraoui. A bayesian approach for the detection of code and design smells. In *Proceedings of the 9th International Conference on Quality Software*, pages 305–314, 2009.
[9] Radu Ma-rinescu. *Measurement and Quality in Object-Oriented Design*. PhD thesis, Politechnica University Timisoara, Faculty of Automatics and Computer Science, 2002.
[10] Zsuzsanna Marian. Aggregated metrics guided software restructuring. In *Proceedings of the 8th IEEE International Conference on Intelligent Computer Communication and Processing*, pages 259–266, 2012.
[11] Naouel Moha. Detection and correction of design defects in object-oriented architectures. In *Doctoral Symposium, 20th edition of the European Conference on Object-Oriented Programming*, 2006.
[12] Naouel Moha, Yann-Gaël Guéhéneuc, and Pierre Leduc. Automatic generation of detection algorithms for design defects. In *Proceedings of the 21st IEEE/ACM International Conference on Automated Software Engineering*, pages 297–300, 2006.
[13] Naouel Moha, Yann-Gaël Guéhéneuc, Anne-Françoise Le Meur, Laurence Duchien, and Alban Tiberghien. From a domain analysis to the specification and detection of code and design smells. *Formal Aspects of Computing*, 22(3–4):345–361, 2010.
[14] Matthew James Munro. Product metrics for automatic identification of "bad smell" design problems in java source code. In *Proceedings of the 11th IEEE International Software Metrics Symposium*, 2005.
[15] Profiler4j, 2012. http://sourceforge.net/projects/profiler4j/.
[16] Gabriela Serban, Alina Câmpan, and Istvan Gergely Czibula. A programming interface for finding relational association rules. *International Journal of Computers, Communications & Control*, I(S.):439–444, June 2006.
[17] Winrun4j, 2012. http://sourceforge.net/projects/winrun4j/.

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

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVIII**, Number 1, 2013

BUILDING AN AUTOMATED TASK DELEGATION ALGORITHM FOR PROJECT MANAGEMENT AND DEPLOYING IT AS SAAS

BOGDAN POP

Abstract. A large number of project management applications currently exist, some focusing on certain features or domains, while others are designed to work in multiple scenarios and domains. Task delegation and resource allocation is one of the crucial parts of project management. Errors in this aspect can result in significant loss of time, resources and deficient project results. No applications currently available on the market automate task delegation; therefore the entire management of the project relies on human experience and is subject to errors. This paper presents the characteristics of a modern, web based, software as a service project management application that focuses on a revolutionary approach to task delegation with the automation of the process. The goal is to reduce, if not eliminate, the necessity of the project manager and to minimize costs and maximize resource usage.

1. Introduction

Project management is the practice of organizing, planning and managing resources in order to achieve specific goals [9]. 27% of a manager's time and more than USD 100 billion is spent each year correcting or working with employees that are not suited for their tasks [11]. Currently, there are more than a few hundred project management software applications, each tackling different aspects of project management. This paper presents a concept application that automatically assigns without human intervention newly created tasks within a project, thus minimizing costs and maximizing resource usage to its fullest. The paper presents the database model, the base algorithm and a deployment scenario as software as a service of the aforementioned concept application. The model tries to provide an algorithm that would reduce, if not

Received by the editors: October 30, 2012.

2010 *Mathematics Subject Classification.* 68M14.

1998 *CR Categories and Descriptors.* C.2.4 Computer Systems Organization [Computer-Communication Networks]: Distributed Systems – *Distributed applications*.

Key words and phrases. nosql, project management, web, application, saas.

eliminate the role of a project manager as known today, allowing the usage of the project manager's knowledge for actual development.

This paper is structured as follows. Section 2 presents some of the most used and known project management applications and some of their main features. Section 3 describes the application concept, its base features and the information flow within those features. Section 4 presents some pros and cons for relational and non-relational databases with regard to current use case and presents why a NoSQL approach is suited for the proposed concept. The 5^{th} section of the paper presents the non-relational database model of the most important data structures used by the application and the algorithm used to automatically assign newly created tasks to users. The 6^{th} section of the paper presents the model that can be used to extend the base application and deploy it as Saas (Software as a Service). The 7^{th} and last section of the paper presents conclusions found as well as future extending possibilities.

2. Project Management Applications Classification

There are a number of factors that can be taken into account while making any classifications for project management applications. Applications can be classified based on the problem they are trying to solve: CRM (client relationship management), collaboration, SCM (supply chain management), bug tracking, issue management or time management. Others can be classified based on the platform or platforms they can run on, such as web, mobile, desktop, or hybrid applications. Some project management applications contain features for dealing with a bundle of the aforementioned problems. Some applications are designed for specific tasks and niches: managing personnel contacts, chatting or sharing scheduling tools. Project management applications can also be classified based on their licensing system, as follows: proprietary, either hosted on premises or not, open source, or SaaS.

2.1. **Project management applications by their licensing system.** Some established open source project management applications include *Trac*, *Launchpad*, *Redmine* or *MantisBT*. *Trac* uses a minimalistic approach to web-based software project management and it is an issue tracking system tailored towards software development projects. *Launchpad* is a collaboration platform that focuses on bug tracking and offers code hosting and reviews, mailing lists, answer tracking or frequently asked questions. It offers a timeline that shows all current and past project events in order, complemented by a roadmap feature that displays future milestones, therefore allowing project managers to easily grasp an overview of the project's progress. *Redmine* combines features from *Launchpad* and *Trac* but it also provides a time tracking feature that supports manual input at project and ticket level.

The most known proprietary project management applications are probably *JIRA*, *Kayako* or *Microsoft Project*. *Kayako*'s main focus is on customer support. It is widely used by web hosting companies of all sizes: resellers to end-point users up to multinational datacenter operating companies. *JIRA* is a project management application mainly used for bug and issue tracking while *Microsoft Project* is designed to assist a project manager in analyzing workloads, tracking progress, managing budgets, assigning tasks and resources.

The most popular applications are probably, due to their distribution channel, the ones offered as services. Some of these software as a service applications are *Basecamp*, *Mavenlink* or *Assembla*.

There are also applications that combine proprietary free and paid solutions. Some have multiple components and only a couple of the central ones require a subscription or a one time fee, leaving the rest of the components free of charge. Two applications that fall within this category are *TeuxDeux* and *getFlow*. Other applications have a reduced lite free version that does not contain all the features of the full paid version. This licensing system is most common in smaller applications that have borrowed their licensing system from the modern application stores designed for mobile devices.

2.2. **Project management applications by their availability platform.** Some project management applications are designed for desktop usage and therefore must be installed on client premises. Some of these applications are *Microsoft Project*, *ConceptDraw Project* or *OmniPlan*. *Kayako* is available as a service on demand or as a downloadable, installable software.

The applications that have seen the largest growth [2] in the past years are the web based ones, due to their high availability, ease of use and lower costs. As they are easier to use, these applications allow for faster and more rapid reactions, which is a key aspect of a successful project management process as described by [5].

JIRA can be used either hosted or installed on the premises. Furthermore, it can be extended through a number of different proprietary applications and services, such as *Confluence*, *FishEye*, *Crucible*, *GreenHopper*, *Bonfire*, *Bonfire*. These extend *JIRA*'s purpose with online wikis, *Subversion*, *Git*, *Mercurial* and *CVS* integration, code reviewing, sprint planning and task tracking for agile teams. They also allow browser screen captures and sharing that facilitates easier tracking, annotating and testing activities.

Basecamp is a project management hybrid system tracking project discussions, files, and events from beginning to end in one place. *Basecamp* has the capability of displaying information about hundreds of projects in a single page including teams, clients, contractors, and vendors. The system can be used to follow along projects development in real-time or by later reviewing

what has happened. *Basecamp* design is simple and easy to use and is bundled with visual timelines, complex access permissions with an ultimate goal to bring responsibility and accountability to the projects managed through it and individuals working on them.

3. Application Concept

As presented in section 2, most of the project management applications require human intervention as they only assist in analyzing workloads, tracking progress, managing budgets, assigning tasks or resources. They do not offer any kind of automation with regards to task management and time management. However, the concept application presented in this paper focuses on an entirely different approach to project management. That is automating project management and eliminating the role of an actual project manager as much as possible.

In order to achieve the highest automation possible, with respect to task and time management, the proposed model has to store information regarding tasks, their types, the people that have completed them, the time required to complete them, personnel availability for future tasks and more. By storing this information one can programmatically determine the best match for a newly added task. By automating this process, the project manager is no longer required and can be part of the actual development team as opposed to only leading it. Moreover, programmatically assigning tasks results in fewer errors compared to those made by a human project manager. Therefore, the development costs and time required to complete projects is reduced to minimum.

The proposed application will show a maximum level of transparency towards clients, allowing access to all information in real time. In order to achieve high transparency and high availability of the entire system and data it withholds, a web application, a hybrid web and mobile system or an internet distributed system is proposed.

Out of the three options, the hybrid web and mobile system is likely the best option. First, it can be developed in two phases, initially with a full web application and secondly with a corresponding mobile application, as it has been the case with numerous successful project management solutions. Secondly, the internet distributed model is prone to bigger costs due to development of multiple applications and systems on different platforms which require different technologies, teams and are harder to maintain.

Apart from the development issues that would arise by the development of a distributed, multi-platform system, another issue could put costs even higher: keeping data consistent while converting it from one format to another.

This would be required given each of the composing platform's characteristics, communication protocols, storage devices and other constraints.

Developing automation of the aforementioned problems can be achieved using a number of evolutionary algorithms, neural networks or swarm intelligence algorithms. However, since the application is likely to store vast amount of information, such algorithms are likely to require a lot of computational time. Given the fact that time management is critical in project management, they may not be best suited for large project management applications.

A different approach, based on Gale-Shapely Courtship Algorithm [1], has been described. A task that has not been attached to a worker selects the worker with the highest preference. If the chosen worker prefers the task to their current task and can finish on time, he or she accepts the task. If the current task is replaced, then it becomes unattached and a callback is placed to attach it again to another worker. The process repeats itself until all tasks have been attached or rejected by the workers [7].

However, although the proposed algorithm has been theoretically proven, no case studies were performed on business groups and it has not been tested in practice. Proper feedback avenues are to be established with testing groups to utilize the mechanisms described in the paper [7].

The proposed model uses basic iterative algorithms to achieve automation by properly storying, sorting and indexing all data available on the project and having it easily accessible from distributed databases. The proposed concept application uses a non-relational database system (NoSQL) for storying the data and automating the process.

4. NoSQL vs SQL

This section presents the database model of the concept application presented in the paper and why a NoSQL approach is better in this scenario. A number of properties, such as atomicity, consistency, isolation, durability (A.C.I.D.) and at a number of guarantees, such as consistency, availability and partition tolerance will be taken into consideration. The three guarantees are known as the CAP or Brewer's theorem [3].

Key properties and guarantees that must be met by the database storage system that the application will use are atomicity, durability, delayed-T consistency, availability and partition tolerance.

In a highly available, partition tolerant system, results are always returned to clients. However, the information returned may be the latest value written in a node A_1, the latest value written in a node A_1 and read from a secondary node A_i $(i \neq 1)$, or a cached value that hasn't expired yet, read from any node A_i $(i = 2, ..., n)$ of the system [10].

Isolation is not a priority requirement since strong consistency cannot be met. No isolation allows two distinct processes to alter the same value in different nodes, which is a potential cause for incorrect information. However, after the set delay time has passed and no new updates are made to the data, the information it withholds becomes consistent based on timestamp ordering. This holds if transactions that started earlier but finished late have a higher priority than those transactions that started later but altered same data sectors sooner. As far as durability is considered, data are not lost, even in the eventuality of a hardware crash since most NoSQL systems are fault tolerant and decentralized.

Before choosing one database system or the other, the application requirements must be presented. First of all, the proposed application will run as software a service and may have a mobile component too. A software as a service application has a huge growth potential and may be required to store vast amount of information from multiple clients: tasks, projects, users, roles, permissions. Hypothetically 20.000.000 tasks, 250.000 users, 30.000 roles, 15.000.000 permissions would be stored if a project's median characteristics were 200 tasks, 10 users, 3 roles, and 150 permissions, with the service hosting 100.000 projects from 25.000 organizations.

Storing so much information on a single SQL server would be difficult, and with high hardware costs. Data could be partitioned across multiple databases and multiple datacenters: users in one server, tasks in other server etc. This would result in increased complexity of the algorithms that access the data. On the other hand, in a NoSQL system, data partitioning can be achieved automatically with little effort, and data accessing does not change as it is abstracted by the NoSQL database system.

SQL systems guarantee information consistency, not high availability as required by the proposed application. Obtaining high availability in SQL database systems is difficult and subject to many failures: communication between nodes or node failures, replication errors etc. With Apache Cassandra however, data is automatically replicated to multiple nodes for fault-tolerance. Replication across multiple data centers is also supported and crashed nodes can be replaced with no downtime. Furthermore, it is decentralized and has no single points of failure and no network bottlenecks. Having these features, Cassandra meets partition tolerance property, which guarantees high availability [8].

5. Database Model and Automation Algorithm

The base model of the application described in section 3 consists of 8 data structures, modeled as columns, column families and super columns. Being a

```
{
    "task name": {
        "keywords":
        [
            { "keyword_name_1": "optional_priority" },
            { "keyword_name_2": "optional_priority" }
        ]
    },
    "task details":[
        { "username": "username value" },
        { "project": "project name" },
        { "timeToComplete": "time" },
        { "completed": "date time" },
        { "deadline": "date time" },
        { "finishedOn": "date time" },
        { "assignedOn": "date time" }
    ]
}
```

FIGURE 1. JSON representation of the Tasks super column

web service the application requires a usernames structure. The users super column contains key value pairs for storying emails, passwords, encryption salts, usernames, contact information, a column family for permissions etc. The tasks super column contains string key value pairs for usernames, project names, time required to complete the tasks, whether the tasks are completed or not, date/time values for deadlines, assignation times or when the tasks were finished, as well as a column family that contains descriptive keywords for each of the tasks. Figure 1 displays the most basic representation of the tasks super column in JSON format.

A user availability column family is also required. It has key value pairs as follows: keys consisted of a string containing the calendar date and a username while the adjacent value holds a column with key name as start time and value as end time. Figure 2 displays the JSON representation of the user availability column family. Project information can be modeled as either a column family or a super column, depending on the information stored. Timelogs are used to store tasks IDs or names, the users working on those tasks and the hours worked on that specific task [10].

The keywordTimeToComplete column family stores a median value of the time required to complete a task by each user. The key name is composed of

```
{
    "DATE_1#username_1":[
        { "startTime_1":"endTime_1" },
        { "startTime_2":"endTime_2" }
    ],
    "DATE_1#username_2":[
        { "startTime_1":"endTime_1" },
        { "startTime_2":"endTime_2" }
    ],
    "DATE_2#username_1":[
        { "startTime_1":"endTime_1" },
        { "startTime_2":"endTime_2" }
    ]
}
```

FIGURE 2. JSON representation of the userAvailability column family

```
{
    { "keyword_1#username_1": "minutes#tasksNo" },
    { "keyword_2#username_1": "minutes#tasksNo" },
    { "keyword_3#username_1": "minutes#tasksNo" },
    { "keyword_4#username_2": "minutes#tasksNo" },
    { "keyword_5#username_2": "minutes#tasksNo" },
    { "keyword_6#username_2": "minutes#tasksNo" }
}
```

FIGURE 3. JSON representation of the keywordTimeToComplete column family

the keyword and the username while the value is composed of the work hours required in minutes and the number of similar tasks the user has worked on. Figure 3 displays this column family in JSON format. This column family together with the following two column families, which are presented in figures 4 and 5, are the basis of the task assignment automation algorithm.

The taskAssign column family stores the following information in the column key name: they keyword, its score and the number of tasks this keyword has been assigned to. The value of the column holds the username that has the certain score for the given keyword. The userScores column family is similar to the previously presented one but it is used to store, track and modify individual user's scores. The key names are composed of a combination of a keyword and a username, while the value holds the score and the number of

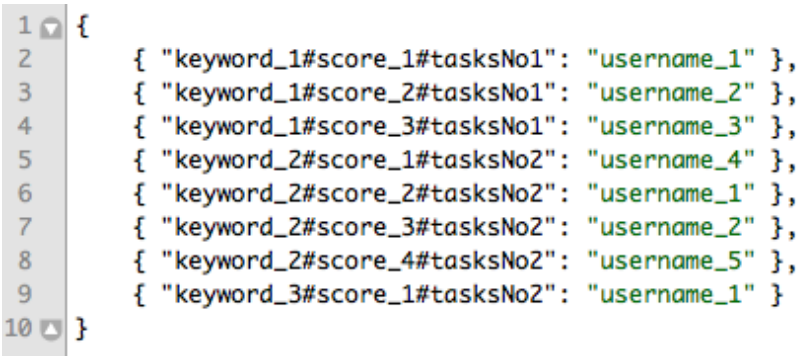


FIGURE 4. JSON representation of the taskAssign column family

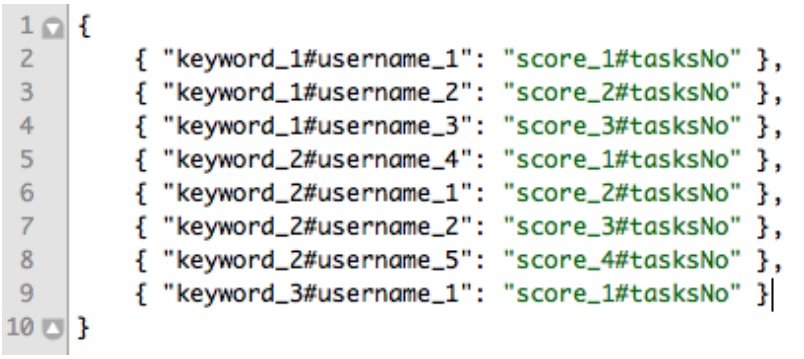


FIGURE 5. JSON representation of the userScores column family

tasks containing the given keyword the user has completed in the past. Whenever one user completes a task, these two columns are updated so that they show the average score per keyword for each user. If two users have the same score, the one that has a lower number of tasks is considered better at the task, as the score was achieved with fewer steps, thus faster [10].

As key names are alphabetically ordered and stored in Cassandra, by using the above model, especially for the keywordTimeToComplete, taskAssign, userScores column families, fast and sorted access is guaranteed towards the information in the database. This means no A.I algorithms are required to pull and assign the best user suited for newly added tasks. Moreover, having the information sorted within the database, the time required to perform the assign operation is low, as searching is fast. This in turn makes the application highly responsive.

```
def function addNewTask(task)

  foundUsers = null
  iteration = 1
  taskAssigned = false

  while (taskAssigned == false && iteration < 10) do

    resetOverallScore(foundUsers)

    for keyword in task.keywords do

      foundUsers.push( getNUsersWithBestScoreFrom_taskAssign(keyword,Start = 10*iteration - 9, End = 10 * iteration) )

      for user in foundUsers
        userScore = GetScoreForKeywordFrom_userScores(keyword,user)
        user.overallScore += userScore
      end

    end

    foundUsers.sort_by { |overallScore, atr1, .. , atrn| overallScore }

    count = 0
    while (taskAssigned == false and count < foundUser.size) do
      if foundUser[count].isAvailable?
        addTaskToUser(foundUser[count],task)
        taskAssigned = true
        return true
      end
    end

    iteration = iteration + 1
  end

  return false

end
```

FIGURE 6. Task Assign Algorithm

When a new task is added, in order to assign it to the best suited user, the algorithm will loop through the taskAssign column family for each keyword, starting from top to bottom, from the best score to the lowest possible. It will compute the overall score for each user and if the user score for a given keyword is null it will count as 0. The user with the best score is assigned the task if and only if the cross checking of the userAvailability column results in a valid response, that is the given user has enough available time to complete the task before the deadline. If the user is not able to complete the task on time, the next user with the best score lower than the previous user's score is selected. If no user can complete the task on time, a message is displayed to the client that is trying to add the new task and two options are provided. The first option is to extend the deadline, which triggers a resume of the above algorithm. The second one is to notify the project manager of the issue and solely in this case human task assignment and decision is required [10].

```
{
    { "project_1#keyword_1#username_1": "minutes#tasksNo" },
    { "project_1#keyword_1#username_2": "minutes#tasksNo" },
    { "project_1#keyword_2#username_1": "minutes#tasksNo" },
    { "project_1#keyword_2#username_2": "minutes#tasksNo" },
    { "project_1#keyword_3#username_1": "minutes#tasksNo" },
    { "project_1#keyword_3#username_2": "minutes#tasksNo" },
    { "project_2#keyword_1#username_1": "minutes#tasksNo" },
    { "project_2#keyword_1#username_2": "minutes#tasksNo" }
}
```

FIGURE 7. JSON representation of the keywordTimeToComplete column family optimized for SaaS

6. Deploying the Application as a SaaS

The model described in section 5 has a few limitations that block the possibility of deploying the application as software as a service. This is due to the fact that all data structures are global and contain no information regarding the projects they are part of, except the task description data structure. For example, information in the keywordTimeToComplete column family (Figure 3) is stored and indexed based on the keyword name and username tuple.

As hypothetically issued in section 4, with 100.000 projects one could have more than 100.000 duplicate keyword names and plenty duplicate usernames. Therefore, searching the keywordTimeToComplete column family for one project would require going through other projects as well. This increases computation time and has a detrimental effect on the entire user experience.

To compensate and to further optimize data indexing, storage and accessing, limiting the amount of time required to retrieve needed information and therefore improving efficiency and user experience, one could easily prefix the database access keys with a project identifier as shown in Figure 7. This way searching would be kept through one project alone, separate from the rest of the projects.

Similar prefix would also be added to taskAssign, userScores and any data structure that does not have a global scope. Furthermore, given the automatic partitioning and distribution model of Apache Cassandra, this model would also allow local, close to client storage for each project. For example, the entire SaaS application would be distributed across 3 continents. Clients from US would access data from US servers; clients from EU would access servers from EU and clients from Australia would access data from servers located there.

7. CONCLUSIONS

The proposed concept is viable as it reduces the role of the project menager. The few cases when a project manager's input would be required within the application are few. One case would hold when deadlines for newly created tasks are really short and the client is unwilling to extend them. A solution for this issue would be to introduce a new user (developer) to the project and have the task assigned to the new person. This however requires human interaction within and outside of the application. Another case when the project manager would be required would be when the algorithm cannot find a single user that can complete the task (noted by $task_x$), as nobody in the system has completed a similar task before.

Future developments may include solving the above issues and enhancements of the current model, as follows: assign tasks based on user's skill set as defined in their profile, not just by their performance, mobile apps integration, automatic time tracking based on proximity sensors or based on straightforward interfaces within the mobile applications, taking user preference into consideration when automatically assigning the tasks, or even employee satisfaction / happiness while doing a specific task based on reports collected from coworkers. This is important because studies have shown that misinterpreted user preference and lack of employee satisfaction inhibits productivity [6].

The presented model limits the accessibility of the software to projects that don't require hardware tools, or that require hardware tools that are by their own definition always available. A huge improvement for the algorithm would be a component that tracks availability, cost and logistics for hardware equipments required by individuals working on the project.

Lastly, the model proposed is eventually consistent and highly available, which means that some information may not be available to all users at a given point in time due to external factors. The algorithm is therefore prone to errors with regards to task assignation when such events occur. This is due to the fact that not all the information would be available to make the proper, exact estimations. Future work should include comprehensive tests that would yield the severity of the algorithm's deviation from its best result when such events do happen.

References

[1] D. Gale, *The two-sided matching problem. Origin, development and current issues*, International Game Theory Review, Vol 3, Nos. 2 & 3, p. 237-252, 2001

[2] Gartner, *Worldwide Software-as-a-Service Revenue to Reach $14.5 Billion in 2012*, Gartner Newsroom, March, 2012

[3] S. Gilbert, N. Lynch, *Brewers Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services*, MIT, 2002

[4] M. Herlihy, J. Wing, *Linearizability: A Correctness Condition for Concurrent Objects*, ACM Transactions on Programming Languages and Systems, Vol 12, No 3, 1990
[5] L. Ireland, *Future Trends in Project Management*, PrezSez 08-2008, 2008
[6] J. M. Ivancevich, *High and low task simulation jobs: a causal analysis of performance-satisfaction relationships*, Academy of Management Journal, Vol 22, No 2, p 206-222, 1979
[7] B. Lagesse, *A game-theoretical model for task assignment in project management*, IEEE International Conference on Management of Innovation and Technology, Singapore, 2006
[8] A. Lakshman, P. Malik, *Cassandra - A Decentralized Structured Storage System*, The 3rd ACM SIGOPS International Workshop on Large Scale Distributed Systems and Middleware, Big Sky Resort MT, LADIS, 2009
[9] S. Nokes, I. Major, et al., *The definitive guide to Project Management: Every executives fast-track to delivering on time and on budget*, Prentice Hall, 2004
[10] B. Pop, *NoSQL Model Design for Automating Project Management*, IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR 2012), Cluj-Napoca, 2012
[11] J. Skabelun, *Are non-performers killing your bottom line?*, Credit Union Executive, Vol 31, No 13, 2005

Department of Computer Science, Babeş-Bolyai University, 1 M. Kogălniceanu St., 400084 Cluj-Napoca, Romania
E-mail address: popb@cs.ubbcluj.ro, bogdan.pop@webraptor.eu

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVIII**, Number 1, 2013

A CONTENT ONTOLOGY DESIGN PATTERN FOR SOFTWARE METRICS

IONEL VIRGIL POP

Abstract. This paper presents a content ontology design pattern for the representation of software metrics, in software engineering ontologies, called OOPMetrics. This content ontology design pattern is designed to ease the detection of software design flaws based on the metrics that are defined in the ontology that uses it. We also present a case study that shows how an ontology that uses this pattern may be queried in order to detect these design flaws. In particular, we will focus on the God Class design flaw.

1. Introduction

In the field of ontology engineering, content ontology design patterns were introduced in [12]. They are a class of ontology design patterns and are useful in reusing concepts over many ontologies. In the field of software engineering, it was shown by Marinescu in [17, 18], that software metrics gathered through code analysis can help in detecting design flaws in software systems.

Sometimes we need to represent software metrics in a software engineering ontology, because software engineering ontologies often make use of software metrics in their content. For this purpose, we present in this paper a content ontology design pattern for software metrics, called OOPMetrics. The content ontology design pattern that is proposed in this paper is not extracted from a particular ontology, but is based on the best practices of using software metrics in the semantic web, mostly from the approach taken in [16] to query knowledge about software metrics. By not having such a content ontology design pattern it is clearly seen from the first stages of ontology development that the naming and relationships among different components that interact

Received by the editors: November 29, 2012.

2010 *Mathematics Subject Classification.* 68N30, 68Q55, 68T30, 97P20.

1998 *CR Categories and Descriptors.* D.2.8 [**Software**]: Software Engineering – *Metrics*; I.2.4 [**Computing Methodologies**]: Artificial Intelligence – *Knowledge Representation Formalisms and Methods*.

Key words and phrases. Content Ontology Design Patterns, Software Metrics, Reverse Engineering.

in a software metrics ontology is very difficult to decide. This is especially the case if one wants a unified system of components with that of other ontologies or wants to query the software metrics ontology to extract software design flaws based on the metrics.

Although OOPMetrics is described here for the first time in a scientific article, we also made available a short description of OOPMetrics online in the pattern collection from the ODP Portal at [23] with a link to it's implementation [22]. However the short description available there was mainly extracted automatically from the pattern's implementation based on it's annotations. And the ontological elements listed there were also extracted automatically from it's implementation.

This paper is structured as follows: after this introductory section, the next one deals with the definitions of the terms used throughout the paper and presents the related work that was done so far in this area. Section 3 describes our proposed content ontology design pattern. Section 4 presents a case study on behavioral god classes. In section 5 we draw some conclusions and present the work that we intend to do in the future.

2. Background and Related Work

The notion of ontology has many definitions in literature. One of the earliest definitions that were given, that is also suitable for the context in which ontologies are described in this paper, is the following:

Definition 2.1. *"An ontology defines the basic terms and relations comprising the vocabulary of a topic area as well as the rules for combining terms and relations to define extensions to the vocabulary."*[20]

Knowledge patterns were described in [5], an updated version of [4]. Later, this notion was extended and ontology design patterns were described, based on [12, 24] as providing *"a modeling solution to solve a recurring ontology design problem"*[21].

There are six classes of Ontology design Patterns (OP), as they were classified in [13]: Structural OPs, Correspondence OPs, Content OPs (CPs) or Content (or conceptual) Ontology Design Patterns (CODePs), Reasoning OPs, Presentation OPs and finally Lexico-Syntactic OPs. We will focus on CPs in this paper as the other ontology design patterns are beyond the scope of this paper.

In Gangemi's article [12] the notion of a Conceptual (or Content) Ontology Design Pattern (CODeP) is described for the first time. In [13] the following definition is given:

Definition 2.2. *CODePs "encode conceptual, rather than logical design patterns. In other words, while Logical OPs solve design problems independently of a particular conceptualization, CPs propose patterns for solving design problems for the domain classes and properties that populate an ontology, therefore addressing content problems."*[13]

Software maintenance and the evolution of software systems were first addressed by Lehman in [14]. Software maintenance accounts for some 70 percent of the total expenditure required in the life cycle of a software system [15].

Software metrics ontologies, especially those that use an appropriate content ontology design pattern, may be queried to find design flaws in the software system. *"The presence of design flaws in a software system has a negative impact on the quality of the software, as they indicate violations of design practices and principles, which make a software system harder to understand, maintain, and evolve"*[9].

Possible design flaws in object-oriented software design were presented by Selvarani et al. in [26]. According to them, they are:

- In the case of *improper coupling:* **Shotgun Surgery** (at class level) and **Wide Subsystem Interface** (at subsystem level);
- In the case of *low cohesion:* **Feature Envy** (at method level) and **Misplaced Class** (at subsystem level);
- In the case of *improper distribution of complexity:* **God Class** (at class level), **God Method** (at method level) and **God Package** (at subsystem level);
- In the case of *flaws related to data abstraction:* **Data Class, Refused Bequest** (both at class level).

Marinescu [17, 18] has done notable work in the area of re-engineering software systems and the current work is based on his papers. Particularly we use the method presented in [17] to show how design flaws may be detected.

Li et al. [16] have brought their contribution in integrating software metrics data using semantic web techniques.

In [7], Şerban has proposed a quantitative evaluation methodology for object-oriented design, based on static analysis of the source code, and described by a conceptual framework. Serban's conceptual framework has four layers of abstraction [7, 8]: Object-Oriented Design Meta-Model, Formal Definitions of Object-Oriented Design Metrics, Specifications of the Assessment Objectives and Measurement Results Analysis. In [8], a case study for her approach was presented, involving God Class design flaw detection.

A number of content ontology design patterns were designed recently. The ODP Portal [25] has the descriptions for a collection of such content ontology

design patterns that were designed for many domains. At present we have however not found any other content ontology design pattern for the software engineering domain on this portal, besides our OOPMetrics [22, 23] content ontology design pattern, even though this is a large and perhaps the only comprehensive collection of content ontology design pattern descriptions on the web.

3. Description of the Proposed Content Ontology Design Pattern

The name of the pattern described in this section is Object-Oriented Programming Metrics Pattern (OOPMetrics). This is a content ontology design pattern for software metrics. Ontologies that use our content ontology design pattern may be queried to detect flaws in design, based on the value of the metrics.

In the description of our content ontology design pattern we will follow some of the steps of describing a pattern that were used in describing software design patterns in [11], such as: intent, motivation, applicability, structure, participants, collaboration, consequences and implementation.

3.1. **Intent.** The goal of this content ontology design pattern is to represent object-oriented software metrics especially for the purpose of detecting flaws in the design of software systems based on these metrics. This may be useful for re-engineering the software system.

3.2. **Motivation.** We consider a context where we have a properly designed ontology for object-oriented software metrics that is based on the OOPMetrics content ontology design pattern. Now let us consider the following scenario: find which class is a God Class based on it's metrics. By using a simple query over the ontology, we can find the God Class if it exists, because we have used a content ontology design pattern that facilitated this. Of course it would have been possible to query it without using a content ontology design pattern to design the ontology. However, in this situation, every software metrics ontology will have to define the concept of software metric in it's own way. Thus, software metrics ontologies would have to be queried in various, different ways that are not necessarily optimized to detect God Classes. In order to have a unified content for the software metrics ontologies, we need a content ontology design pattern like OOPMetrics.

3.3. **Applicability.** The domain of applicability for the OOPMetrics content ontology design pattern is software engineering, more particularly software metrics. This content ontology design pattern has very good applicability in

detecting design flaws in software systems based on metrics. Mainly, we have identified the following competency questions:

- What are the software metrics for a certain project or package or class or method?
- Knowing the necessary software metrics, is there a design flaw in the software system?

3.4. **Structure, Participants and Collaboration.** Figure 1 represents the diagram of the OOPMetrics content ontology design pattern. Additionally, table 1. and table 2. provide some of the elements that do not appear on the diagram.

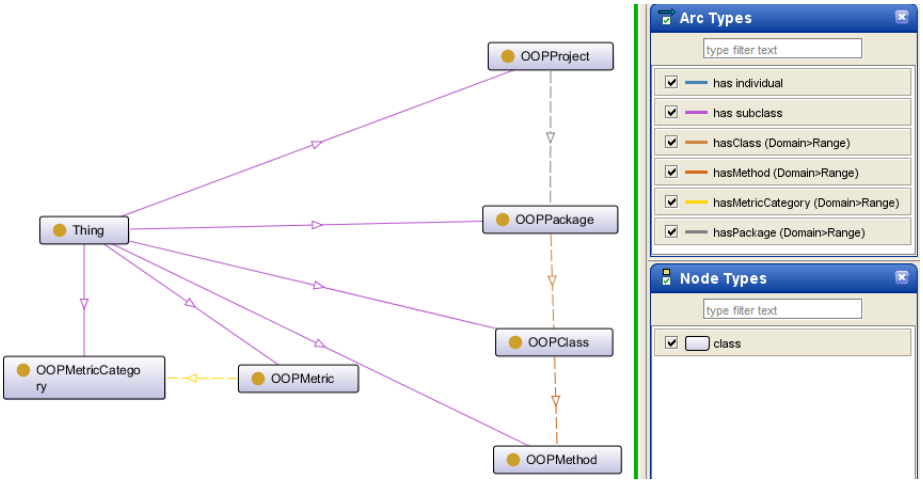


FIGURE 1. The OOPMetrics Diagram

The conceptual elements behind the OOPMetrics content ontology design pattern are classes, data properties and object properties. The following classes are defined:

- **OOPProject**: This class represents a software project;
- **OOPPackage**: This class represents the "package" concept found in object-oriented programming;
- **OOPClass**: This class represents the "class" concept found in object-oriented programming;
- **OOPMethod**: This class represents the "method" concept found in object-oriented programming;
- **OOPMetricCategory**: A (computed) software metric belongs in an OOPMetricCategory. Therefore this class represents the category in

which it belongs to (or what kind of software metric it is). A software metric is therefore represented separately in two distinct classes. These are: OOPMetricCategory (for defining a software metric) and OOPMetric (for computing a software metric) for flexibility reasons;
- **OOPMetric**: This class represents the concept of a (computed) software metric found in object-oriented programming.

In table 1, we show the data properties that are specific to OOPMetrics.

Data Property	**Domain**	**Range**
hasFloatValue	OOPMetric	float
hasIntegerValue	OOPMetric	integer
hasLongName	OOPMetricCategory	string
hasName	OOPMetricCategory	string
hasTag	OOPMetricCategory	integer

TABLE 1. Data Properties of the OOPMetrics pattern

In table 2, we show the object properties that characterize OOPMetrics.

Object Property	**Domain**	**Range**
hasPackage	OOPProject	OOPPackage
hasClass	OOPPackage	OOPClass
hasMethod	OOPClass	OOPMethod
hasMetricCategory	OOPMetric	OOPMetricCategory
hasMetric	UnionOf: OOPProject, OOPPackage, OOPClass, OOPMethod	OOPMetric

TABLE 2. Object Properties of the OOPMetrics pattern

3.5. **Consequences.** The OOPMetrics content ontology design pattern allows ontology engineers and software engineers to design software metrics ontologies easier and in a more unified way. Also, this content ontology design pattern was designed in such a way that software design flaws can be easily detected based on software metrics by using it.

3.6. **Implementation.** It is essential for a content ontology design pattern to be implemented in a particular ontology language in order for the content to be imported later when an ontology is created. This content ontology design pattern was implemented in the Web Ontology Language (OWL) [1], using the ontology editor and knowledge acquisition system Protégé [10]. The implementation is available at [22]. This implementation of the OOPMetrics content ontology design pattern can be imported in Protégé and used in designing software metrics ontologies.

4. Case Study

In this section, a case study will be presented regarding how a software metrics ontology that uses the OOPMetrics pattern may be queried, in order to detect software design flaws. The focus in this case study will be on the God Class design flaw.

In [17] three metrics were used to detect God Classes: Weighted Methods Per Class (WMC) that is defined in [3] and may use various complexity measures such as McCabe's cyclomatic complexity [19], Tight Class Cohesion (TCC), defined in [2] and Access to Foreign Data (ATFD), described in [17]. In [17], it was also explained that high values of WMC and ATFD and low values of TCC may lead to God Classes.

In [26] a formula was presented to detect God Classes. The following formula is an example of how to detect suspects, that is based on [26], but with absolute values:

$$\textbf{GodClass(C)} = \textbf{(WMC(C) >100) and (ATFD(C) >1) and (TCC(C) <0.5)}.$$

We must emphasize the fact that it is beyond the scope of this paper to prove that a detection strategy, such as the one based on the metrics used here to detect God Classes, actually works, because this has already been shown in [17, 18]. Therefore, this paper does not attempt to restate Marinescu's approach by presenting a case study for large software projects with hundreds or thousands of classes to show how such a detection strategy may work. Instead, the point of this case study is to show how the concepts and the relationships defined in the OOPMetrics pattern can be used in practice, in a SPARQL query. Thus, the query in figure 2 shows how an ontology that uses the OOPMetrics pattern may be queried by users, if the ontology was designed using this pattern.

In the query example from figure 2, a hypothetical ontology that uses the OOPMetrics content ontology design pattern is queried. The reason why a hypothetical ontology was chosen here and not a real world ontology is to show that any ontology that uses this content ontology design pattern

may be queried in a similar manner in order to detect which classes are God Classes. However, in order to make sure that this query is correct, we have tested this query in Protégé with an ontology that we have designed using the OOPMetrics pattern.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX oopmetrics: <http://www.cs.ubbcluj.ro/~ivpop/ontologies/
oopmetrics.owl#>

SELECT DISTINCT ?class
WHERE {
    ?class rdf:type oopmetrics:OOPClass .

    ?class oopmetrics:hasMetric ?metric1 .
    ?metric1 rdf:type oopmetrics:OOPMetric .
    ?metric1 oopmetrics:hasMetricCategory ?wmc .
    ?wmc rdf:type oopmetrics:OOPMetricCategory .
    ?wmc oopmetrics:hasTag 1 .
    ?metric1 oopmetrics:hasIntegerValue ?v1 . FILTER (?v1 > 100)

    ?class oopmetrics:hasMetric ?metric2 .
    ?metric2 rdf:type oopmetrics:OOPMetric .
    ?metric2 oopmetrics:hasMetricCategory ?atfd .
    ?atfd rdf:type oopmetrics:OOPMetricCategory .
    ?atfd oopmetrics:hasTag 2 .
    ?metric2 oopmetrics:hasIntegerValue ?v2 . FILTER (?v2 > 1)

    ?class oopmetrics:hasMetric ?metric3 .
    ?metric3 rdf:type oopmetrics:OOPMetric .
    ?metric3 oopmetrics:hasMetricCategory ?tcc .
    ?tcc rdf:type oopmetrics:OOPMetricCategory .
    ?tcc oopmetrics:hasTag 3 .
    ?metric3 oopmetrics:hasFloatValue ?v3 . FILTER (?v3 < 0.5)
}
```

FIGURE 2. SPARQL Query

In the query from figure 2 it is considered that an individual WMC, exists in the ontology, of type OOPMetricCategory for which the value of the hasTag data property is 1, an individual ATFD, exists in the ontology, of type OOP-MetricCategory for which the value of the hasTag data property is 2, and an

individual TCC, exists in the ontology, of type OOPMetricCategory for which the value of the hasTag data property is 3. They could have been referred to by using the hasName or hasLongName properties in a similar manner.

5. Conclusions and Future Work

Content ontology design patterns provide a very convenient way of reusing components in an ontology. This paper has described a content ontology design pattern for software metrics. Then a case study was presented that showed how an ontology that uses such a content ontology design pattern may be queried.

Besides quering the ontology that uses the OOPMetrics content ontology design pattern to extract knowledge like in the case study presented in this paper, it is also possible, through rules, to inference knowledge using reasoners such as Pellet [6]. This allows for further exploitation of the OOPMetrics content ontology design pattern for even more advanced purposes. In the future we intend to exploit the advantages of using reasoners to infer knowledge over ontologies that use this content ontology design pattern by adding rules to it.

References

[1] Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F., and Stein, L. A., OWL Web Ontology Language Reference, W3C Recommendation 10 february 2004, Dean, M. and Schreiber, G., eds., `http://www.w3.org/TR/owl-ref/`.

[2] Bieman, J. M., and Kang, B. K., Cohesion and reuse in object-oriented systems, In *Proceedings of the 1995 Symposium on Software Reusability*, ACM, USA, 1995, pp. 259–262.

[3] Chidamber, S., and Kemerer, C., A metrics suite for object-oriented design, *IEEE Trans. Softw. Eng. 20*, 6 (1994), 476–493.

[4] Clark, P., Thompson, J., and Porter, B., Knowledge patterns, In *KR'2000 (Proc 7th Int Conf)*, A. Cohn, F. Giunchiglia, and B. Selman, Eds. Kaufmann, 2000, pp. 591–600.

[5] Clark, P., Thompson, J., and Porter, B., Knowledge patterns, In *Handbook on Ontologies*, s. Staab and R. Studer, Eds. Springer, 2003, pp. 191–207.

[6] Clark&Parsia, Pellet: OWL 2 reasoner for java, `http://clarkparsia.com/pellet`, 2004.

[7] Şerban, C., A conceptual framework for object-oriented design assessment, In *Fourth UKSim European Modelling Symposium on Computer Modelling and Simulation (EMS)*, 2010, pp. 90–95.

[8] Şerban, C., God class design flaw detection in object oriented design. a case study, *Studia Univ. Babeş-Bolyai, Informatica LVI*, 4 (2011), 33–38.

[9] D'Ambros, M., Bacchelli, A., and Lanza, M., On the impact of design flaws on software defects, In *Proceedings of the 10th International Conference on Quality Software*, IEEE Computer Society, USA, 2010, pp. 23–31.

[10] Stanford Center for Biomedical Informatics Research (BMIR), The Protégé ontology editor and knowledge acquisition system. http://protege.stanford.edu/.
[11] Gamma, E., Helm, R., Johnson, R., and Vlissides, J., *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, USA, 1994.
[12] Gangemi, A., Ontology design patterns for semantic web content, In *Proceedings of the Fourth International Semantic Web Conference*. Springer, 2005, pp. 262–276.
[13] Gangemi, A., and Presutti, V., Ontology design patterns, In *Handbook on Ontologies, Second Edition*, International Handbooks on Information Systems. Springer, 2009, pp. 221–243.
[14] Lehman, M. M., The programming process, IBM Res. Rep. RC 2722, 1969.
[15] Lehman, M. M., Programs, life cycles, and laws of software evolution, *Proc. IEEE 68*, 9 (1980), 1060–1076.
[16] Li, Y., F., and Zhang, H., Integrating software engineering data using semantic web technologies, In *Proceedings of 8th Working Conference on Mining Software Repositories (MSR 2011)*, ACM, USA, 2011, pp. 211–214.
[17] Marinescu, R., Detecting design flaws via metrics in object-oriented systems, In *39th International Conference and Exhibition on Technology of Object-Oriented Languages and Systems (TOOLS 39)*, IEEE Computer Society, USA, 2001, pp. 173–182.
[18] Marinescu, R., Detection strategies: Metrics-based rules for detecting design flaws, In *In Proc. IEEE International Conference on Software Maintenance* (2004).
[19] McCabe, T. J., A complexity measure, *IEEE Trans. Softw. Eng. 2*, 4 (1976), 308–320.
[20] Neches, R., Fikes, R., Finin, T., Gruber, T., Patil, R., Senator, T., and Swartout, W., Enabling technology for knowledge sharing, *AI Magazine 12*, 3 (1991), 36–56.
[21] Noppens, O., and Liebig, T., Ontology patterns and beyond - towards a universal pattern language, In *Proceedings of the Workshop on Ontology Patterns (WOP 2009)*. 2009, pp. 179–186.
[22] Pop, I. V., Oopmetrics (owl file), http://www.cs.ubbcluj.ro/~ivpop/ontologies/oopmetrics.owl.
[23] Pop, I. V., Submissions:oopmetrics, http://ontologydesignpatterns.org/wiki/Submissions:OOPMetrics.
[24] Presutti, V., and Gangemi, A., Content ontology design patterns as practical building blocks for web ontologies, In *Proc. of the 27th Int. Conf. on Conceptual Modeling (ER)*. 2008, pp. 128–141.
[25] NeOn Project, Ontology design patterns . org (odp), http://ontologydesignpatterns.org.
[26] Selvarani, R., Banu, W., and Prasad, K., Quantifying the design quality of object oriented system the metric based rules and heuristic, In *National Conference on Advanced Software Engineering*. Bangalore, 2008, pp. 54–62.

E-mail address: popionelvirgil@yahoo.com

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

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVIII**, Number 1, 2013

FUZZY COMPUTING FOR COMPLEXITY LEVEL OF EVALUATION TESTS

TIBERIU BAN

Abstract. Students tend to make mistakes in evaluation tests that follow a pattern which can be mined; this allows the level of complexity for a given set of tasks in a test paper to be computed in advance. The goal of this paper is to present a mathematical way to compute the level of complexity for a given set of tasks based on a fuzzy model as an improvement from the crisp model. This indicator can be effectively used to predict in advance the degree in which the association rules already mined will trigger chains of mistakes in a given set of evaluation tasks.

1. Background

This paper focuses on mining existing data association between chains of mistaken items from a evaluation test. The main rationale is that if a specific subset of items are mistaken, then there is a computable chance with a specific threshold of confidence level that this chain will trigger one or several other test items to be mistaken as well. After the point where enough data association rules have been already mined, this information can be used in analysing a new set of test items in order to determine the extent to which chain of items can trigger other items to be solved incorrectly. This will be presented with the aid of an indicator called Complexity Level of a set of tasks within a test paper.

The field of Data Mining and its main concepts are taken into consideration as presented in [10] and [11] and continues the path of Data Association presented in [3] and [6], as well as the Discovery of Significant Rules techniques presented in [13]. The APriori algorithm used are presented in [4] and also [12]. Applications of Data Analysis Technique in the field of detecting and mining mistakes made by students are also covered in [9], as well as adaptation of

Received by the editors: January 15, 2013.

2010 *Mathematics Subject Classification.* 68P15.

1998 *CR Categories and Descriptors.* H.2.8. [**Database Applications**]: DataMining – *Data Association Methods*.

Key words and phrases. Data Mining, Data Association, APriori, Complexity of Test Paper, Mathematical Model of Mistakes.

fuzzy approach to this domain of study, as well as applications of Data Analysis Technique in assessing other performance criteria from students projects in [8].

The Complexity Level will present the degree to which existing chains of item tasks can cascade trigger more items to be also mistaken in turn. The higher the value of a complexity level, the lower the score for a statistically average student will be. The Complexity Level is only relevant if its value is relative, not absolute, ranging between 0 and 1.

Allthough it builds on several widely used techniques of Data Association and APriori Algorithm, this application is original in its nature of applying the steps from a supervised learning approach to the known methodology. Also, the field of study is opened to fuzzy approach that comes as a natural extension of the crisp model, each element is taken into account not only with its support and confidence level but also the degree of membership to fuzzy classes. This approach builds on the fiability of the model, making it more robust. The algorithm and the data processing techniques are adapted to the field of study with respect to the robustness of the approach.

Definition 1: An *evaluation test* is defined as a set of tasks.

Each task can either be solved correctly or solved incorrectly. A correctly solved task will be graded with the entire score for the particular task. An incorrectly solved task will be graded with zero score.

Definition 2: A *mistake* is defined as a task that is solved incorrectly.

The first mathematical model that was presented in [1] had a crisp approach to it, thus making it a simplified model. In the first mathematical model each task was considered as having the same amount of points as total score.

The crisp approach consisted in the fact that it was taken into consideration the possibility to divide the tasks from a given evaluation paper in distinct subsets, with each subset consisting of items that had a support level above a predefined threshold.

Also, in order to keep the mathematical model crisp, more assumptions and restrictions have been in place [2] such as each student is present at all tests and each student had the same number of items presented.

The structure of the paper has two main parts. The first part sets the theoretical basis of the Fuzzy Mathematical Model in Section 2 and a review of the current notions in Data Association used in Section 3 and the adaptation of APriori algorithm to the business domain in Section 4. The second part presents an experiment made on real data instances that is presented in Section 5 and the results in Section 6.

2. The Fuzzy Model for Computing the Level of Complexity of an Evaluation Test

The first mathematical model presented in [1] is not sustainable in real life situation due to the fact data regarding the itemset each question belongs to is not crisp. A problem arises in the fact that the same question can be included in multiple itemsets. A good way of handling this is refering to a new fuzzy approach to computing the complexity of a test paper.

Definition 3: The *complexity level of a test paper* is defined as the relative indicator best estimate of the percent an average student is likely to lose out of the total amount of points for the given test paper.

Before presenting a formula to compute this indicator, here are a few theoretical observations:

Let's consider a test paper that consists of four questions labeled A, B, C and D. For this test let's consider a sample of 10 papers with the list of mistaken questions from Table 1.

Paper No.	Mistaken Items
1.	A, B
2.	A, B
3.	A, B
4.	D
5.	D
6.	A
7.	A
8.	D
9.	n/a
10.	A

Table 1. Ten data instances - Mistakes gathered from test papers

After running the APriori algorithm described in the following sections the candidate itemsets discovered are given in Table 2.

These candidate itemsets can be visualized in a lattice structure. In this lattice the void itemset acts as the general infimum and the candidate itemset with all the items as the general supremum for the given lattice. The lattice would be constructed as follows, with respect to [5]:
Let $\mathbf{C}$ be the set of candidate itemsets. $\forall C_i, C_j \in \mathbf{C}$,
$inf(C_i, C_j) = C_i \cap C_j \, and \, sup(C_i, C_j) = C_i \cup C_j$.

In order to further clarify this notion, for the previous example we will show the corresponding lattice. With respect to style and clarity and also in order to focus only on the important nodes in the lattice, we will only

Itemset	Support Level
{A}	0.6
{B}	0.3
{C}	0.2
{D}	0.3
{A, B}	0.3
{C, D}	0.2

TABLE 2. Candidate itemsets for Table 1

implement the nodes that correspond to candidatesets with nonzero support. Figure 1 illustrates the lattice generated with the candidatesets. Figure 2 adds more information to Figure 1, by also displaying the corresponding support level for each candidateset.

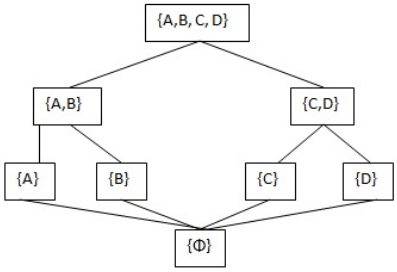


FIGURE 1. Lattice of candidatesets

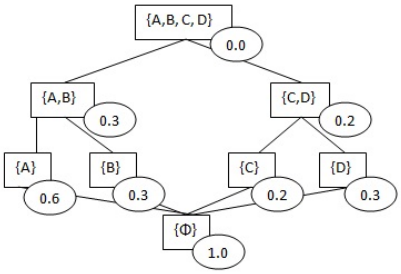


FIGURE 2. Lattice of candidatesets with support values

However, the itemset {B} can be pruned down because of the fact that itemset {B} never appeared isolated and is always as subset of {A, B}. For the same reason candidate itemset {C} can also be pruned down. The remaining itemsets are given in Table 3.

Itemset	**Support Level**
{A}	0.6
{D}	0.3
{A, B}	0.3
{C, D}	0.2

TABLE 3. Candidate itemsets after prunning down

Question A belongs also to the itemset {A} as well as to the itemset {A, B}. Same situation occurs for question D which belongs also to itemset {D} as well as to itemset {C, D}.

In a fuzzy approach we need to build a membership function that will estimate the degree of membership for a given question to an itemset. This process is with respect to the number of occurences of the question to each of the itemsets.

With respect to [14], let's consider the following definitions:

Definition 4: A *fuzzy set* is a "class" with a continuum of grades of membership.

Definition 5: Let X be a space of points (objects), with a generic element of X denoted by x. Thus, X = {x}. A fuzzy set(class) A in X is characterized by a *membership (characteristic) function $f_A(x)$* which associates with each point in X a real number in the interval [0, 1], with the value of $f_A(x)$ representing the "'grade of membership" of x in A.

The membership function in the first iteration is represented in Table 4.

Question	**Itemset**	**Membership value**
A	{A}	0.5
A	{A, B}	0.5
B	{A, B}	1
C	{C, D}	1
D	{D}	0.33
D	{C, D}	0.66

TABLE 4. Values for Membership Function on Candidate Itemsets

In order to clarify, let x be an item (i.e. mistake in a test). Let X be a candidateset that is a fuzzy set. The grade of membership of x in X is computed

as the normalized "contribution" of supp(x) towards computing supp(X). This approach is derived from supervised learning techniques in such manner that it uses known computable values from already known data instances in order to gain knowledge to be used in future data instances.

Definition 6: The *membership function* is defined as $\mu : Q \times I \to [0, 1]$, where Q is the set of tasks from the evaluation test and I is the total set of candidate itemsets. The value $\mu(q, i)$ represents the membership value of the question q to the itemset i

The following definitions extend the work from [2] towards the fuzzy sets.

Definition 7: The *cost function* is defined as $C : I \to \mathbf{R}$, where I is the total set of candidate itemsets. $C(i)$ represents the total amount of score that can be graded in the event that all the items from the itemset i would be solved correctly.

Definition 8: The *support function* is defined as $sup : I \to [0, 1]$, where I is the set of all candidate itemsets.

Definition 9: The *indicator TotalScore* is defined as the sum of costs of all tasks in the set of tasks Q. TotalScore will be computed as follows:

$TotalScore = \sum_{q \in Q} C(\{q\})$

Definition 10: The *Complexity Level* for a given evaluation test will be computed with the following formula:

$Complexity = \frac{\sum_{i \in I, q \in i} \mu(q,i) \times sup(i) \times C(i)}{TotalScore}$

3. Association Rules Notions

The aim of this section is to review the main concepts of Association Rules as well as the working algorithm used to mine valid association rules from a large set of data instances. We reccomend [12] for further reading.

As presented in [12], Association Rules are similar in nature to Classification Rules. This technique aims to extract valid knowledge from existing dataset in form of rules like the following:

$X_1 \wedge X_2 \wedge ... \wedge X_n \to Y[C, S]$

where $X_1, X_2, ..., X_n$ are attributes from the dataset. If in a single item these attributes have a distinct combination of values, there is a computable probability to predict a distinct value for attribute Y. Naturally, on the right hand side of the rule there can be more than a single attribute, thus making the discovery process more difficult because of the nature of predicting more variables in a single rule.

A brute force algorithm might be able to attempt to have every combination of a subset of attributes for both the left hand side and the right hand side of the association rule and to consider this a hypothesis that needs to be validated, but such attempt would require an enormous computing power

gone to waste, considering only very few of these artificial hypothesis would check out as valid after confronting them to existing items in the dataset.

A better technique would be to prune down rules and their branches that are constructed from these rules by adding more items on both sides of the rule. The criteria on which such pruning can be safely done is the coverage of the rule (the number of insances the rule can correctly predict) and the accuracy (the proportion of the number of items or instances from the dataset the rule can be applied to).

As first mentioned [4] and later refined in [10], the coverage of a rule is expressed by its support level, while the accuracy is computed based on its confidence level.

In order to define more clearly these two indicators, let's consider an association rule in the form of $\{itemi_1, ..., itemi_k\} \rightarrow \{itemj_1, ..., itemj_l\}$ and let's define the support set of the association rule, as presented in [10] the set defined by the reunion of the items both in the left hand side and the right hand side of the rule

$\{itemi_1, ..., itemi_k, itemj_1, ..., itemj_l\}$

Let's formally define the support level as follows [10]:

Definition 10. The *support level of a subset* $\{itemi_1, itemi_2, ..., itemi_k\}$ is the percentage where all the items from the given subset were present in the same transaction, out of the total number of transactions.

Also, let's formally define the confidence level of an association rule as follows:

Definition 11. The *confidence level of an association rule*

$\{itemi_1, ..., itemi_k\} \rightarrow \{itemj_1, ..., itemj_l\}$ represents the percentage of transactions where both set of items were present, out of the total number of transactions where the first set of items

$\{itemi_1, ..., itemi_k\}$ were present.

In order not to generate association rules that are too weak or that apply extremely infrequent, two more indicators are needed: minimum support level (called minsup) and minimum confidence level (called minconf).

According to [10], these two indicators should first be set a bit restrictive in order to avoid generating too many association rules and then slowly relaxing the value of the minimum confidence level, until an acceptable number of association rules are determined.

Basically, we only need to focus on association rules with high coverage. We will refer to the items from the support set of a specific association rules, regardless of the position of each item, either left hand side or right hand side. We would only seek combination of attributes that have a minimum coverage, more precise the support level to be above the preset threshold of minsup.

In order to discover Association Rules between mistakes belonging to a specific test paper the APriori algorithm can be used, since all its perquisites are met [6].

4. The Algorithm Used for Generating Rules Efficiently

There are several algorihms available that carry on the task of generating association rules with a specified minimum support and meeting the minimum confidence level. Each of these algorithms follow two general steps. Apriori algorithm has been chosen for this particular experiment. The basics for the APriori algorithm has been presented first in [3].

4.1. **General Algorithm for Generating Association Rules.** The algorithm described in detail in [12] starts by generating all one item sets with the given minimum coverage. Then it uses these sets as base in order to generate all two items sets, three item sets until either all items available in the attribute list are included in the itemset. Since this case would imply a hard partition of all available attributes in the left hand side and the right hand side of an association rule, this case is less likely to occur in every dataset.

As stated both in [3] and also in [4] another condition to end the first step of the algorithm after generating a k-item set meeting the minimum support level, no other (k+1)-item set can be obtained by adding an extra item to any existing k-item sets. This condition is more likely to be the real marker to stop this step and consider the existing item sets of up to k items to be candidates for support set of valid association rules.

In order to generate a (k+1)-item set, an extra item is added to an existing k-item set generated at the previous iteration. Even more, in order to prune down unnecessary computing effort in generating all (k+1)-item sets, each such set

$\{itemi_1, itemi_2, ...itemi_{k+1}\}$ needs to have all of its k item subsets already in the list of valid generated k-item sets at the previous iteration. If any k item subsets does not meet the minimum support level, then the (k+1) item set can not meet the minimum support level either.

A strategy to avoid unnecesary generations of (k+1) item sets is presented in [12]. All k-item sets are to be sorted using the same criteria, either alphabetically or ascending if the items are coded using numbers.

If there are two k item sets $S1 = \{itemi_1, itemi_2, ..., itemi_{(k-1)}, itemi_k\}$
$S2 = \{itemi_1, itemi_2, ..., itemi_{(k-1)}, itemi_{k'}\}$
that have k-1 common items and exactly one different item,

$Card(S_1 \cap S_2) = k - 1$ and
$Card(S_1 - S_2) = 1$ and
$Card(S_2 - S_1) = 1$,

then a new set S' can be obtained by joining the two sets S1 and S2,

$S' = S1 \cup S2$ and

$Card(S') = k + 1$

Furthermore, in order to avoid generating the same (k+1) item set out of several distinct pairs of k item sets, we could only take the k item sets that have the first (k-1) items in their intersection set.

For instance if we have the following 3-item sets that already meet the minimum support level and their items have been ordered alphabetically

$\{A, B, C\}, \{A, B, D\}, \{A, C, D\}, \{A, C, E\} and \{B, C, D\}$

the union of the two that have the same first two items identical

$\{A, B, C\} \cup \{A, B, D\} = \{A, B, C, D\}$

is to be considered, since any other union like

$\{A, B, D\} \cup \{A, C, D\} = \{A, B, C, D\}$

would end up returning the same 4-item candidate that we already collected.

Considering this same example, the following union

$\{A, C, D\} \cup \{A, C, E\} = \{A, C, D, E\}$

which does not meet the minimum support level and is not a valid 4-item candidate because a 3 item subset $\{C, D, E\}$ does not meet the minimum support level and was not included in the 3-item sets generated at the previous iteration.

The process of checking a (k+1) item set whether it meets the minimum support level or not is simplified even more by using hash tables. Each item is to be removed in turn and the remaining k-item set is checked whether it is part of the valid k item sets already known.

Finally, any (k+1) item sets needs to have its support level actually being computed, because the above method only tells for sure if it does not.

5. Presenting the experiment

A test consisting of 36 questions was presented to 75 students, each student having received the same set of test items, with a random factor for the order in which the items were presented. Each student received the same amount of time consisting of 45 minutes to complete the test.

Each test item consisted of one multiple choice question, with 4 distinct answer choices. The students were instructed that only one of the choices is correct for any given question. The 36 questions covered general topics of computer science, such as hardware devices, software concepts, operating system concepts, measurement units in Computer Science, general networking concepts, user safety and ergonomy concepts.

The main rationale for this experiment was the hypothesis already stated in [1] and in more details in [2] that Data Association Methods can be used

in relevant data gathered from results of test papers, as follows. Association rules exists between test items such as if a student incorrectly solves a set of given test items, then there is a computable chance that the same student will incorrectly solve a different set of given test items.

The algorithm chosen for the experiment was APriori algorithm as described in [12], without the benefits given by the usage of hash tables for easier access to candidate itemsets. The confidence level used in the experiment was 75 percent and the minimum support level was set at 20 percent.

Each test paper was recorded as a data instance with an identifier alongside 36 relevant attributes. Each attribute had a correspondent question in the test paper. Considering the focus of interest is in tracking mistakes made, a question that had been mistakenly answered on the test has been coded with a value of 1, marking a mistake occured in the current data instance. In the same manner, a value of 0 marks no mistake occured in the current data instance for the respective question.

The goal is to use these data instances (records) over the first phase of APriori algorithm in order to check the main rationale, the grouping of mistakes (belonging in the same candidateset) to actually have a logical reasoning. This step is adapted from supervised learning techniques, where each candidateset with enough support would have to pass a human validation to see if the candidateset actually can be justified by the logic of the domain of the test.

6. Results of the Experiment

After adapting both the database structure to reduce the number of passes through the dataset and adjusting the minimum support and minimum confidence several candidate itemsets were generated, that had a very good support level.

Such candidate itemsets contained items that clearly belonged to the same general topic. Some of the generated candidate sets are presented in Table 5.

Other such candidate itemsets were also present but with lower support count. Out of each such candidate itemset, several association rules were formed, but the number of test papers analysed was insufficient to determine without doubt whether some test items were more important than others. As future work, an additional number of over a hundred test papers with the same 36 questions will be added to the existing 75 records.

The way the test items were grouped in the same candidate itemset clearly supports the main working hypothesis that mistaked test items do follow association rules, in this case based on the knowledge level of a distinct topic covered by the test.

Topic	**Questions - same candidate set**
Hardware Topic	Which of the following can improve a computer's performance? What is Hard Disk formatting used for? Which of the following devices is an input device? Which of the following devices is both an input and output device?
Software Topic	Which of the following programmes is a software application per se? Which of the following is a function of the operating system? What type of software controls resource allocation in a computer?
Network Topic	Which of the following is not a feature of online commerce? Which of the following is the main advantage of using a computer network? What is World Wide Web? Which of the following statements on the Internet is true?

TABLE 5. Sample of Generated Candidate Itemsets

7. Conclusions and Future Work

The experiment itself largely confirmed the working rationale first taken into consideration in [1]. Several false association had to be removed for not actually having a sustainability in the logic of the data itself. This confirms that a supervised learning approach is the correct way to analyse this particular problem.

The next logical step in terms of Future Work is the adaptation to be able to implement Fuzzy Association Rules and be able to compute the complexity level for a given set of data instances. This step needs to be adapted to supervised learning approach. A comparison between available data association rules is also a valid direction for study.

Also, having non crisp, continuous data values for attributes will open to new APriori adaptations to be taken into consideration, such as the Algorithm for Discovery of Arbitrary Length Ordinal Association (DOAR) presented in [7]. Moreover, having supervised learning data association rules already

known, there is another direction of study in terms of matching new data instances to the already learned association rules and studying the outliers with a Fraud Detection approach in mind.

References

[1] Ban, T. - **"Concept Paper: Generating and Assessing Test Papers Complexity using Predictions in Evolutionary Algorithms"**, Knowledge Engineering: Principles and Techniques Conference (KEPT) 2009, Cluj-Napoca,

[2] Ban, T. - **"Using Predictions in Data Mining for Improving Students' Performance in Tackling Online Test Papers"**, Acta Universitatis Apulensis Special Issue, International Conference on Theory and Applications in Mathematics and Informatics (ICTAMI) 2009, Alba-Iulia

[3] Agrawal, R. & Imielinski, T. & Swami, A - **"Mining Association Rules between Sets of Items in Large Databases"**, Proceedings of the 1993 ACM SIGMOD international conference on Management of data, Washington D.C., May 26-28, pages 207 - 216

[4] Agrawal, R. & Srikant, R. - **"Fast Algorithms for Mining Association Rules"**, Proceedings of 20th VLDB Conference, Santiago, Chile, 1994, pages 487 - 499

[5] Birkhoff, G. - **"Lattice Theory"**, American Mathematic Society Colloq. Publ., Vol. 25, New York 1948, republished 1967

[6] Kotsiantis, S. & Kanellopoulos, D. - **"Association Rules Mining: A Recent Overview"**, GESTS International Transactions on Computer Science and Engineering, Vol.32 (1), 2006, pages 71 - 82

[7] Campan, A. & Serban, G. & Truta, T. M. & Marcus, A. - **"An Algorithm for the Discovery of Arbitrary Length Ordinal Association Rules"**, Proceedings of the 2006 International Conference on Data Mining, DMIN 2006, Las Vegas, Nevada, USA, June 26 - 29, 2006, CSERA Press 2006, pages 107 - 113

[8] Frentiu, M. & Pop, H. F. - **"A Study of Dependence of Software Attributes Using Data Analysis Techniques"**, Studia Universitatis Babes-Bolyai, Informatica, Volume XLVII, Number 2, 2002, pages 53 - 66

[9] Frentiu, M. & Pop, H. F. - **"Detecting Mistakes in Students Software Measurement Projects"**, Proceedings of Symposium "Colocviul Academic Clujean de Informatica", 2005, pages 105 - 111

[10] Pang-Ning, T. & Steinbach, M. & Kumar, V. - **"Introduction to Data Mining"**, Addison Wesley US Edition, 2005

[11] Sumathi, S. & Sivanandam, S.N. - **"Introduction to Data Mining and its Applications"'**, Studies in Computational Intelligence, Volume 29, 2006, pag 161 - 164

[12] Witten, I. & Frank, E. - **"Data Mining Practical Machine Learning Tools and Techniques"**, Morgan Kaufmann Publishers - Elsevier, 2005

[13] Webb, G. - **"Discovering Significant Rules"**, Proceedings of KDD'06 International Conference, Philadelphia, Pennsylvania, USA, August 20 - 23, 2006, pages 434 - 443

[14] Zadeh, L. A. - **"'Fuzzy Sets"**, Information and Control 8, 1965, pages 338 - 353

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

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

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVIII**, Number 1, 2013

TOWARDS A REGION-BASED CALCULUS FOR ENERGY-AWARE PROGRAMMING

FLORIN CRĂCIUN, SIMONA MOTOGNA, AND BAZIL PÂRV

Abstract. Energy efficiency has become one of the most critical software metric both for cloud computing servers as well as for mobile phones, ipads or sensor networks. Much of the research on the reducing energy consumption has been focused on low-power architectures, operating systems and compiler optimizations. Recent studies have been started to explore how programming language technologies can help reason about energy management. In this context our paper discusses how our region calculus used before to manage the heap memory can be adapted to control the energy consumption.

1. Introduction

In the recent decade, energy-aware computing systems (e.g. mobile devices, wireless sensor nodes, cloud computing servers) had a rapid evolution. The transformation of mobile devices (especially smartphones and iPads) into general-purpose computing platforms had an important impact on considering the energy as a first class design constraint for many software applications. Saving energy can extend the battery lifetime and increase the mobility or can reduce the maintenance costs of data-centers.

Much of the research on energy management has been focused on the optimizations for the *energy-aware execution* of the software programs. The optimization techniques (see [KM08] for a survey) have been developed at different layers of the compute stack (e.g. digital circuits, architecture, operating systems and compilers). They are mainly dynamic approaches based on online monitoring or offline profiling.

Recent studies have been started to explore how programming language technologies can help reason about energy management. A new paradigm *energy-aware programming* has been proposed in order to aid developers to

Received by the editors: March 16, 2013.

2010 *Mathematics Subject Classification.* 68Q60, 68N30.

1998 *CR Categories and Descriptors.* D.2.4 [**Software**]: Software engineering – *Software/Program Verification*.

Key words and phrases. energy-aware programming, region calculus, program analysis.

write energy-efficient programs in the first place. Exposing energy considerations at the programming language level can enable a new set of energy optimizations and can allow the program to have a direct control of the energy management techniques from the lower layers of the compute stack.

This paper analyses the new paradigm approaches and discusses a possible unification of them under a general region calculus for energy consumption control. Section 2 introduces the concepts of our previous work on using region calculus for memory management. Section 3 presents the challenges of the new paradigm. Section 4 discusses our proposal while Section 5 concludes the paper.

2. Region Calculus for Memory Management

Region types have been introduced to manage the heap memory at compile time. Region-based memory management systems allocate each new object into a program specified region, with the entire set of objects in each region deallocated simultaneously when the region is deleted. The first safe region-based memory system has been developed by Tofte and Talpin [TT94, TT97] for a functional language. Later, several projects have investigated the use of region-based memory management for C-like languages (e.g. Cyclone [GMJ$^+$02]) and object-oriented languages [BSBR03].

In our previous work [Cra08, CCQR04, CQC08, SCC08], we have developed an automatic region type inference system for object-oriented paradigm. Our compiler automatically augments unannotated object-oriented programs with regions type declarations and inserts region allocation/deallocation instructions that achieve a safe memory management. Our work uses lexically-scoped regions such that the memory is organised as a stack of regions. Regions are memory blocks that are allocated and deallocated by the construct *letreg r in e*, where the region r can only be used to allocate objects in the program e. All objects allocated into a region have the same lifetime. Dangling references are a safety issue for region-based memory management. Our work allows only non-dangling references which originate from objects placed in a younger region and point to objects placed either in an older region or inside the same region. Relations between regions and non-dangling references conditions are expressed as lifetime constraints between objects regions.

Recently, region assertions have also been used to control the possible aliasing [ABB06] in information flow analyses. A new regional logic [BNR08, RBN12] has been proposed to reason about mutation and separation, via variables of type region (finite sets of object references).

3. Energy-Aware Programming Paradigm

Energy consumption is a combined effect of many hardware components (such as CPUs, caches, DRAMs, I/O devices) which interact in complex ways. Therefore energy consumption control is a challenging task for programmers. Energy-aware programming paradigm proposes different programming models and logical frameworks that can help developers to reason about energy consumption. However developers are assumed to have minimal knowledge about energy consumption. In general the new proposed programming models assure an efficient and correct control of the hardware-level energy management through special programming language constructions (e.g. special annotations for types or special instructions). Analysing the recent approaches proposed for energy-aware programming we have identified the following main directions: programming using controlled approximations [SDF$^+$11, LPMZ11, BC10, CKMR12], programming using phased behaviours [CZSL12], and programming according to the battery energy states [CZSL12, SKG$^+$07].

3.1. **Programming using Controlled Approximations.** The key observation of this programming model is that the programs spend a significant amount of energy guaranteeing correctness. However the programs have portions that are more resilient to errors and portions that are critical and must be protected from errors. Therefore non-critical portions of the programs can save a significant amount of energy by using approximate computations. Approximate computations might consist of approximate storage (e.g. reducing refresh power in DRAM memories [LPMZ11], unreliable registers and data caches), approximate operations (e.g. instructions for approximate integer ALU operations as well as approximate floating point operations) and algorithmic approximation (e.g. approximation of the expensive functions and loops [BC10]).

Distinguishing between the critical and non-critical portions of a program is the main challenging task of this programming model. EnerJ [SDF$^+$11] proposes a type system that isolates the precise portion of the program from the approximate portion. That means it prevents a direct flow of data from approximate to precise variables. It also allows programmers to compose programs from approximate and precise components safely. Later a more complete architectural support for approximate programming has been developed in [ESCB12]. Recently, a relational assertion logic [CKMR12] has been proposed to express and verify the properties of program approximations.

3.2. **Programming using Phased Behaviours.** The key observation of this programming model is that different program fragments have distinct patterns of CPU usage, memory accesses, cache misses, and I/O operations

which lead to a distinct pattern of energy consumption. Therefore a program usually have phased behaviours of energy consumption. The rate of energy consumption is steady within a phase but different across.

The main challenging task of this programming model is to determine the number of phases and the boundary of each of them. Energy types [CZSL12] allow the programmer to specify phased behaviours by using phase type annotations. The type system can enforce the phase distinction (each data and operation must commit to only one phase) and the phase isolation (any cross-phase interaction can be done only with type coercion). Phase type information can control the CPU dynamic voltage and frequency scaling (DVFS). DVFS is based on the observation that is most advantageous to scale down the CPU frequency (such that energy can be saved) when the CPU is least busy (such that the performance is the least affected). The challenging task for applying DVFS is to choose the right scaling point and the right scaling factor. In this case the solution reduces to finding the appropriate boundaries for the phases.

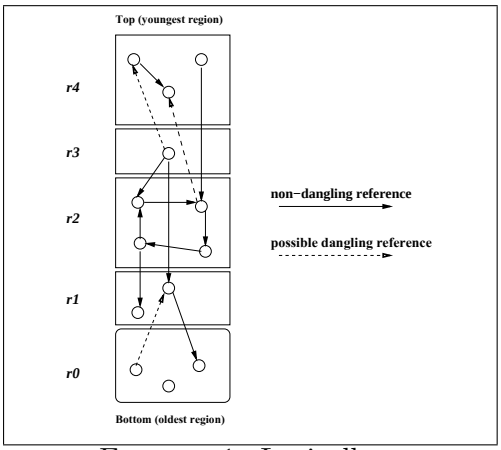


FIGURE 1. Lexically-Scoped Regions

3.3. **Programming according to the Battery Energy States.** The key observation of this programming model is that the different choices to implement an application may consume different levels of energy and be best used in different battery energy states.

The main challenging task of this programming model is to adapt the program to the available energy. Energy types [CZSL12] uses mode type annotations to indicate the expected energy usage context associated with specific data or operations. In [SKG+07] a coordination language is proposed in order to dynamically adapt the system to available energy.

4. A Unified Region Calculus for Energy Management

In this section we propose a general region calculus for both data and code that can unify all three energy-aware programming models presented in Section 3. This proposal extends our region calculus used before to manage the heap memory at compile time [Cra08]. First we illustrate our previous region

calculus by a simple Java code example and then we analyse the modifications of our calculus to support energy-aware programming models.

Our region calculus uses lexically-scoped regions such that the heap memory is organised as a stack of regions, as illustrated in Figure 1. Regions are memory blocks that are allocated and deallocated by the construct *letreg r in e*, where the region *r* can only be used to allocate objects in the program *e*. The older regions (with longer lifetime) are allocated at the bottom of the stack while the younger regions (with shorter lifetime) are at the top.

Dangling references are a safety issue for region-based memory management. Figure 1 shows two kinds of references: non-dangling references and possible dangling references. Non-dangling references originate from objects placed in a younger region and point to objects placed either in an older region or inside the same region. Possible dangling references occur when objects placed in an older region point to objects placed in a younger region. They turn into dangling references when the younger region is deallocated. Using a dangling reference to access memory is unsafe because the accessed memory may have been recycled to store other objects. There are two approaches to eliminating this problem. The first approach allows the program to create dangling references, but uses an effect-based region type system to ensure that the program never accesses memory through a dangling reference. The second approach uses a region type system to prevent the program from creating dangling references at all. Our work has adopted the second approach. Let us consider the example of Figure 2, the Pair object is allocated in region *r4* which is the top of the regions stack. The two fields of the Pair *p* are allocated in two regions *ra* and *rb* which must be older than or the same as *r4*. All the safety requirements are guaranteed by our type system [Cra08]. In addition we have developed the first automatic region type inference system for object-oriented paradigm [Cra08]. Our compiler automatically augments unannotated object-oriented programs with regions type declarations and inserts region allocation/deallocation instructions that achieve a safe memory management.

```
letreg r4 in {
  Pair p;
  Object a,b;
  ...
  a = new Object<ra>();
  b = new Object<rb>();
  p = new Pair<r4>(a,b);
  ...
}
```

FIGURE 2. Memory Regions Example

In this proposal we assume that the regions for energy management are manually introduced by the programmers while our energy type systems guarantees the appropriate safety conditions for using energy regions.

4.1. **Programming using Controlled Approximations.** In order to distinguish between the critical and non-critical fragments (both code and data) of the programs we use two kinds of regions: *approximate regions* and *precise regions*. By default, when no region is explicitly given, the code and the data are in a precise region. Therefore the programmers have to introduce the region programming constructions only for the approximate data and computations. In general an approximate region unifies approximate data storage, approximate computation and approximate algorithms.

Let us consider the example from Figure 3 where the region *rApprox* denotes an approximate program fragment. Therefore all the variables declared inside this region (e.g. *a*, *c*), all the memory allocations done inside this region (e.g. *b*), and all the operators computations done inside this region must be approximate. In the case of a function call that is executed inside of an approximate region (e.g. *f(y)*, its computation can be done according to the regions of the function body. However the function call result is stored in an approximate variable (x in our example). Our model is portable, the compiler is entirely responsible for choosing the energy-saving mechanisms for approximate data and computations from an approximate region. The safety requirement that must be guaranteed by our type system is that the approximate data cannot affect precise data. However it is important that data be occasionally allowed to break the strict separation enforced by the type system. Therefore our region model provides a special construction that allows the programmers to control explicitly when approximate data can affect precise state.

```
letreg rApprox in {
  int a,c;
  Object b;
  ...
  b = new Object();
  a = c+a;
  x = f(y);
  ...
}
```

FIGURE 3. Approximate Regions Example

4.2. **Programming using Phased Behaviours.** In this energy programming model our energy regions denote the energy states of the different hardware components whose energy consumption contributes to the energy consumption of the programs. In this proposal we restrict our region calculus to

CPU states such that our energy regions represent the CPU frequencies. For example we can have the regions *rH*, *rM*, and *rL* to denote three different frequencies for a CPU: high, medium and low respectively. The programmers can choose different regions for different program fragments execution according to the characteristics of those program fragments.

Let us consider the example from Figure 4 where the region *rH* is used for CPU intensive operations while *rL* is used for I/O operations. Our *letreg* construction corresponds to two instructions: the first which set the CPU frequency at the beginning of the block and the second which restore the previous CPU frequency at the end of the block. The safety requirement for this model refers to the proper usage of the CPU frequencies. A proper usage of CPU frequencies means to reduce the energy consumption without significantly affecting the execution time. It is very difficult to statically guarantee this safety requirement.

```
letreg rH in {
  c=1;
  while (c < 10000){
        ...
        //CPU intensive operations
        ...
  }
  ...
  letreg rL in {
     ...
     //I/O operations
     ...
  }
  c=1;
  while (c < 10000){
        ...
        //CPU intensive operations
        ...
  }
  ...
}
```

FIGURE 4. CPU Frequency Regions Example

4.3. **Programming according to the Battery Energy States.** In this model the energy regions correspond to the battery energy states. Since the battery state cannot be known at compile time these regions are runtime regions. However we can use a special construction (implemented in a special library) that can check the battery state in order to introduce the regions at the compile time. An illustrative example is given in Figure 5. The programmers can choose the code that will be executed according to the battery status. The safety requirement here is to not allow a transition from a low status region to a high status region without an explicit check/change of the battery status.

5. Concluding Remarks and Future Work

In this paper we analised the new energy-aware programming paradigm, we identified the programming models of the new paradigm and we discussed the challenges of these models. As a solution we proposed a general unified region calculus based on our previous work on region-based memory management. Our intention is to formalize the proposed calculus and to develop an energy type system that can enforce the energy safety requirements for all the programming models of the new paradigm. We also like to develop a compile-time energy evaluation model that can guide the programmers to find the appropriate places for energy regions in a program. Our work is a small step towards designing programming models for energy efficiency.

```
if (batteryState()==High_State){
  letreg rH in {
    ...
  }
}else {
  letreg rL in {
    ...
  }
}
```

Figure 5. Battery State Regions Example

Acknowledgment

This work was possible with the financial support of the Sectoral Operational Program for Human Resources Development 2007-2013, co-financed by the European Social Fund, within the project POSDRU 89/1.5/S/60189 with the title Postdoctoral Programs for Sustainable Development in a Knowledge Based Society.

References

[ABB06] Amtoft, T., Bandhakavi, S., and Banerjee, A. A logic for information flow in object-oriented programs. In *POPL*, pages 91–102. ACM, 2006.

[BC10] Baek, W. and Chilimbi, T. M. Green: a framework for supporting energy-conscious programming using controlled approximation. In *PLDI*, pages 198–209. ACM, 2010.

[BNR08] Banerjee, A., Naumann, D. A., and Rosenberg, S. Regional logic for local reasoning about global invariants. In *ECOOP*, pages 387–411. 2008.

[BSBR03] Boyapati, C., Salcianu, A., Beebee, W., Jr., and Rinard, M. Ownership types for safe region-based memory management in real-time java. In *PLDI*, pages 324–337. ACM, 2003.

[CCQR04] Chin, W.-N., Craciun, F., Qin, S., and Rinard, M. C. Region inference for an object-oriented language. In *PLDI*, pages 243–254. ACM, 2004.

[CKMR12] Carbin, M., Kim, D., Misailovic, S., and Rinard, M. C. Proving acceptability properties of relaxed nondeterministic approximate programs. In *PLDI*, pages 169–180. ACM, 2012.

[CQC08] Craciun, F., Qin, S., and Chin, W.-N. A formal soundness proof of region-based memory management for object-oriented paradigm. In *ICFEM*, pages 126–146. 2008.

[Cra08] Craciun, F. *Advanced Type Systems for Object-Oriented Languages*. PhD Thesis, National University of Singapore, 2008.

[CZSL12] Cohen, M., Zhu, H. S., Senem, E. E., and Liu, Y. D. Energy types. In *OOPSLA*, pages 831–850. ACM, 2012.

[ESCB12] Esmaeilzadeh, H., Sampson, A., Ceze, L., and Burger, D. Architecture support for disciplined approximate programming. In *ASPLOS*, pages 301–312. ACM, 2012.

[GMJ+02] Grossman, D., Morrisett, G., Jim, T., Hicks, M., Wang, Y., and Cheney, J. Region-based memory management in cyclone. In *PLDI*, pages 282–293. ACM, 2002.

[KM08] Kaxiras, S. and Martonosi, M. *Computer Architecture Techniques for Power-Efficiency*. Morgan and Claypool Publishers, 1st edition, 2008.

[LPMZ11] Liu, S., Pattabiraman, K., Moscibroda, T., and Zorn, B. G. Flikker: saving dram refresh-power through critical data partitioning. In *ASPLOS*, pages 213–224. ACM, 2011. ISBN 978-1-4503-0266-1.

[RBN12] Rosenberg, S., Banerjee, A., and Naumann, D. A. Decision procedures for region logic. In *VMCAI*, pages 379–395. 2012.

[SCC08] Stefan, A., Craciun, F., and Chin, W.-N. A flow-sensitive region inference for cli. In *APLAS*, pages 19–35. 2008.

[SDF+11] Sampson, A., Dietl, W., Fortuna, E., Gnanapragasam, D., Ceze, L., and Grossman, D. Enerj: approximate data types for safe and general low-power computation. In *PLDI*, pages 164–174. ACM, 2011.

[SKG+07] Sorber, J., Kostadinov, A., Garber, M., Brennan, M., Corner, M. D., and Berger, E. D. Eon: a language and runtime system for perpetual systems. In *SenSys*, pages 161–174. ACM, 2007.

[TT94] Tofte, M. and Talpin, J.-P. Implementation of the typed call-by-value lambda-calculus using a stack of regions. In *POPL*, pages 188–201. 1994.

[TT97] Tofte, M. and Talpin, J.-P. Region-based memory management. *Inf. Comput.*, 132(1997)(2):109–176.

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

E-mail address: craciunf@cs.ubbcluj.ro,motogna@cs.ubbcluj.ro,bparv@cs.ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVIII**, Number 1, 2013

DECISION SUPPORT SYSTEM FOR BABES-BOLYAI UNIVERSITY

VIORICA VARGA, HOREA GREBLĂ, AND ANCA ANDREICA

Abstract. Babes-Bolyai University is considered from an enterprise perspective in order to build a business intelligence solution that would improve decision-making process in order to achieve a better performance, on both academical and economical level. Our initial proposed design takes into account an enterprise business model, identifying the main processes structured according to Zachman's enterprise architecture framework. A preliminary approach of the multidimensional design is also presented. Some of the possible entities to be considered in the decision process improvement are described in the paper. A collaborative business intelligence approach is inspected as a possible solution for the proposed model.

1. Introduction

Babes-Bolyai University is considered to be the most important employer and one of the most important economical agents in the Cluj-Napoca city. We try to view the university from an enterprise perspective and model its processes accordingly. We focus on some of the core processes and propose a data warehouse design as basis for a Business Intelligence (BI) solution.

The activities that are performed in a university can be seen and modeled as a set o individual business processes. There are business processes that involve a single department with a specific target and there are processes that span across the university. As the university can be perceived as an enterprise from business process perspective, the goals and current economical environment presses the executive board to optimize the activities, improve cost savings and improve services to their clients, no matter if they are the students, industry partners or some other entities.

This paper is organized as follows: the second section describes the nine building blocks of the business model and presents an enterprise architecture

2000 *Mathematics Subject Classification.* 68P15, 68U35.

1998 *CR Categories and Descriptors.* H.4.2. [**Information Systems Applications**]: Types of Systems – *Decision support*.

Key words and phrases. Business Intelligence, Data Warehousing, Decision Support, Information Systems.

framework that we consider suitable for Babes-Bolyai University. The next section presents the multidimensional data model for the BI solution of the educational and research activity. Last section presents the business intelligence solution from its users point of view, proposing a collaborative approach.

This paper presents a preliminary approach for the design part of the BI solution. We intend to implement our data model, load it with data and apply data analysis and data mining techniques to help decision.

2. Business Model's Building Blocks

According to [3] "A business model describes the rationale of how an organization creates, delivers, and captures value". In their work they formulated the building blocks for a business model as follows:

1. Customer Segments: An organization serves one or several Customer Segments. - In our case, Babes-Bolyai University has as main "customer" the student, but it is not limited to that; there are projects having the industry or governmental units as beneficiary and some other cases like projects for professional retraining or life long learning for non student categories.
2. Value Propositions: It seeks to solve customer problems and satisfy customer needs with value propositions. As the society evolves, the knowledge transferred to students needs to be aligned accordingly, the result being the frequent updates in the University curricula and specializations offered to students.
3. Channels: Value Propositions are delivered to customers through communication, distribution, and sales channels. For the university case there are direct channels through courses, online channels through course materials provided, sometimes widely accessible, conferences, seminars, reviews and magazines.
4. Customer Relationships: Customer relationships are established and maintained with each Customer Segment through specific entities inside university: secretarial and academic staff for students, public relations department, research department for external projects, and so on.
5. Revenue Streams: Revenue streams result from Value Propositions successfully offered to customers. The revenue has many forms, from direct taxes collected from students, governmental budget received per student, funds raised, revenues from scientific research projects and other projects, sold magazines, etc.
6. Key Resources: Key resources are the assets required to offer and deliver the previously described elements and our university has a highly

respected teaching and research staff, a good infrastructure and various collaborations with other entities in the country and abroad.

7. Key Activities: The main focus is on teaching and research but also there is an administrative and maintenance department that facilitates teaching and researching in good conditions.
8. Key Partnerships: As the industry is the main beneficiary from the perspective of students as future employees, university developed partnerships with various leading players that share their knowledge early in this stage so that the students acquire more experience.
9. Cost Structure: The business model elements result in the cost structure.

In order to create a complete picture of the enterprise model for the university, we can use an enterprise architecture framework like Zachman's were we can identify and note [7] all the goals, people and technologies that support achieving the goals (see Figure 1, taken from [6]). When building the business

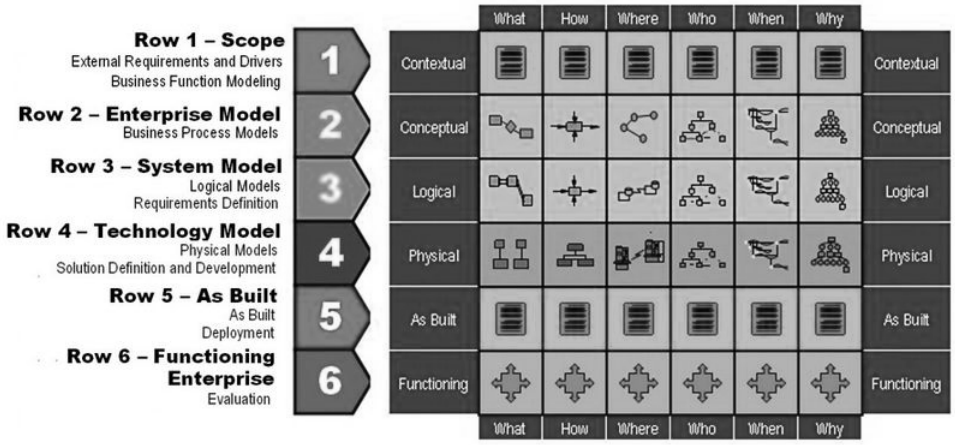


FIGURE 1. Zachman Framework [6]

model we have to take into consideration that the organization is not new on the market, but has long and successful history. Also, the customer is not a client entering a shop for an everyday goods shopping, but a student that invests time and money in developing a profession or employees needing to upgrade the skills in a dynamic economy. In order to compute the profit in a university, we need to visualize more than monetary figures. The numbers of students enrolled might be another success factor, along with the number of graduated students that were employed right after their graduation in a field related to their studies.

The number of research studies successfully finished and implemented have the same importance. Solving problems that increase the standard of living is also a good measure of the success. With all these processes, it is difficult to determine what processes to optimize and where to reduce costs.

In order to make it easier for the executive board, we propose a data warehouse model that would collect all data related to processes inside the university and serve as basis for a business intelligence application that would help in taking appropriate decisions in the optimization process. Our proposed model intends to provide means (at least) to:

- Manage corporate performance
- Provide analytical insight into data
- Allow users to analyze data in a self service manner
- Achieve a balanced academic structure and class schedule
- Support and manage processes for student life-cycle
- Improve admission and grading

3. Modelling the Data Mart

Universities obtain a part of their financial support from student taxes. Therefore, one of the fact tables will be FactStudPayment, which stores the taxes collected from students. Universities have a hierarchical structure consisting of faculties, departments and specializations. The academic staff is organized in departments; students are enrolled to one specialization within a faculty. Usually students are grouped in some formations (DimStudentGroup). Teaching resources are classified into position groups (e.g., professor, lecturer, etc.). Curricular activities are referred as courses. For the data mart these will constitute *dimensions*. Time dimension cannot be missed from a data warehouse. Dimensions usually are not in third normal form. Nearly every dimension has a DateFrom and DateTo attribute, they are not visible, because of space consideration. A student type (StudentType) will change in time, he begin as undergradute, them becomes graduate, he can be a PhD student. This is a Slowly Changing Dimension, which gives us the possibilty to follow student's career. The academic processes can be considered in terms of educational supply with faculties as suppliers of educational services and students as their consumers.

The FactStudPayment fact table has the following dimensionality: student (with the next hierarchy: faculty, specialization, student group), time, taxes.

Students' results at different curricular activities are stored in Catalogs. This is considered fact table, students' graduation at different courses in different faculties, specializations, different semesters can be measured. See the data mart for curricula activities and students payment in Figure 2. The third

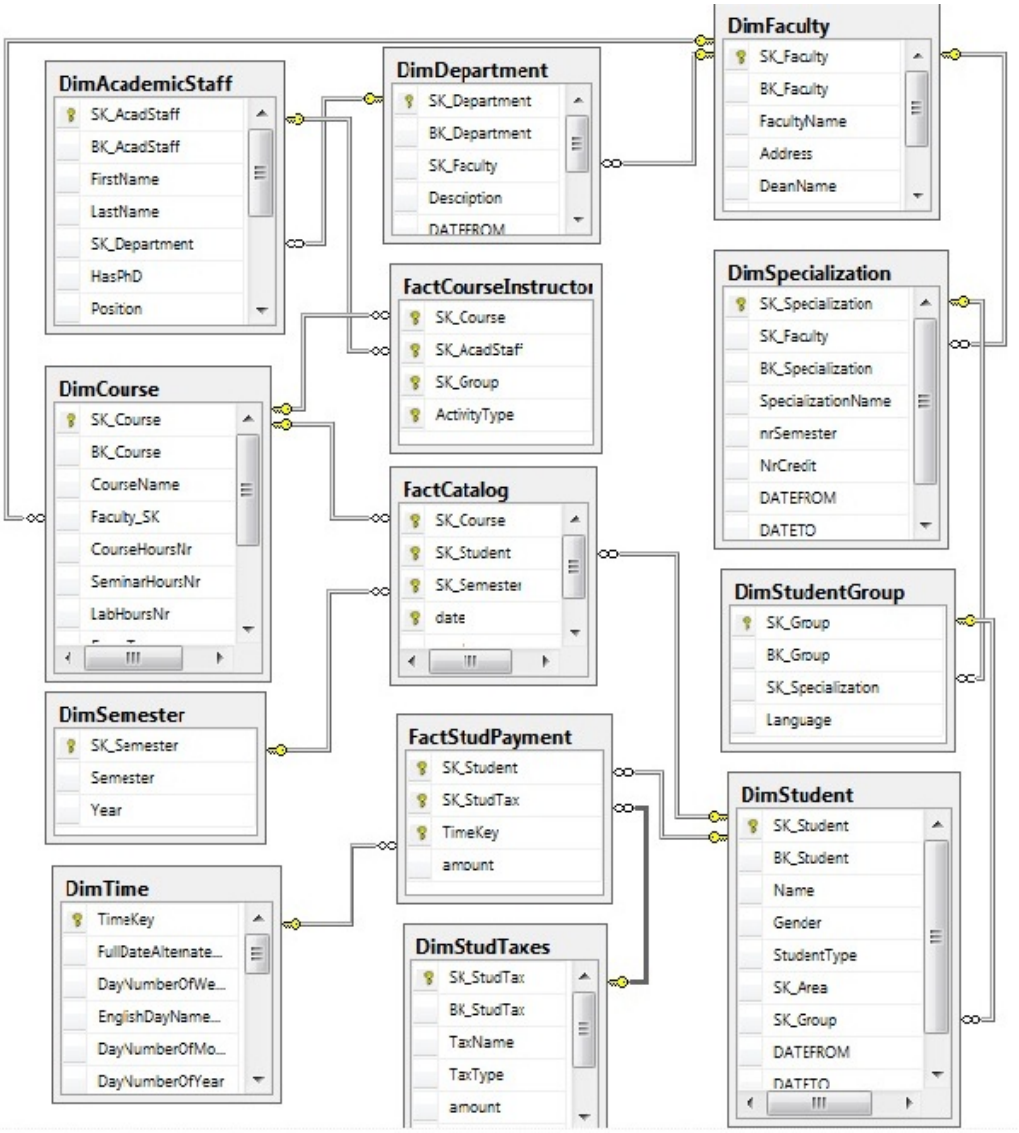


FIGURE 2. Data Mart for Curricula Activity

fact table in the constellation model is FactCourseInstructor. This table stores which course by which instructor for what student group is presented.

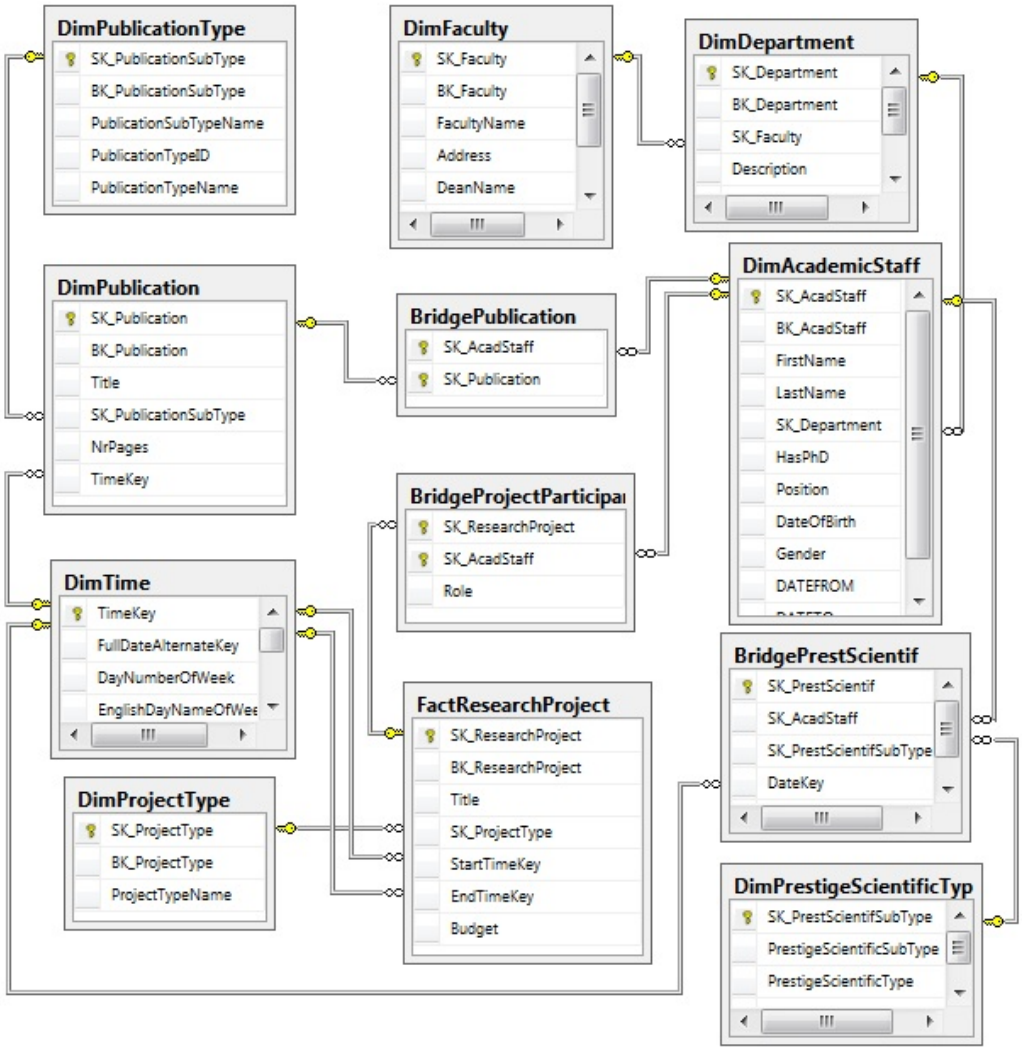


FIGURE 3. Data Mart for Research Activity

The data model for research activity is in Figure 3. Academic staff is involved in research activity which is measured by publications and participation in research projects. Research projects have a budget, so this can be

considered *fact* table. Aggregation functions can be applied for this fact table in the faculty, department hierarchy.

The quality of publications and the number of them are a very important measure of research activity. There is no measurable attribute, so we design a bridge table to store the authors of the publications in BridgeProjectParticipants.

There are other activities, which are important in the scientific prestige of the university, like reviewer in an international journal, getting an international prize, etc. These are stored too in the data model.

4. Sharing and exchange of information

Information offered by the described BI solution has to get to the right people in the organization, so that they can analyze data and make valuable decisions based on that. People that could improve the decision system are not always at the top level of the organization. There are often people at different levels that could bring valuable knowledge in the process of data analysis.

The concept of *Collaborative Business Intelligence* is a quite new trend that refers to such a collaboration between different users in order to improve the decision making system. It represents a merge between business intelligence and social media tools in order to facilitate BI solution users interaction so that they can choose the most relevant data and share it [5].

The BI solution proposed in this paper will transform raw data into meaningful and useful information, but due to the complexity of the organization, analyzing this data and deciding which is the most relevant and the most valuable will require cooperation between different users. Collaborative BI brings the human component next to the organizational information. The process of irrelevant data rejection will be therefore enabled by rating, commenting and sharing.

Some of the advantages of enabling users to interact and provide feedback on reports and dashboards are therefore:

- choice of relevant data
- improved decisions
- taking decisions in a shorter period of time

BI products are being integrated with collaborative platforms like Microsoft SharePoint and IBM Lotus Connections. There is also a development towards cloud based technologies which represent a good approach for collaboration. Mobile technologies also sustain integration of collaborative BI.

Another meaning of collaborative BI is given by data warehouse integration. This comes from the need of interaction with other organizations in the

process of decision making. Indeed, cooperation with other Romanian universities could result in better decisions with high level impact. This would require that each university has its own data warehouse which is to be integrated.

There are three approaches found in the literature for enabling collaborative BI [4] in the sense of data warehouse integration:

- *warehousing approaches* where the integrated data are physically materialized
- *federative approaches* where the integration is only virtual
- *peer-to-peer approaches* when there is no global schema to rely on when integrating different datawarehouses

The first two approaches require the existence of a general schema under which data warehouses coming from different sources could be integrated. In the cited paper [4] there is a new proposed peer-to-peer framework called Business Intelligence Network where peers expose querying functionalities aimed at sharing business information for the decision-making process. The main features of this framework are decentralization, scalability, and full autonomy of peers.

Collaborative business intelligence seems therefore to be a good solution, from both perspectives, for our university BI model. Different tools will be investigated in order to find the most suitable approach for our study case.

5. Conclusions and Future Work

A preliminary research has been done for building a BI solution that could help improve decision-making process for Babes-Bolyai University. This could lead to a better performance of the university at both academical and economical level. We started by identifying the main processes that take place in the university from an enterprise perspective. A preliminary model for the multidimensional design is described. A collaborative approach have also been proposed for the information sharing and exchange in order to improve the decision making system.

The authors intend to study the impact of this solution for the university and to implement the proposed model. The multidimesional model will be extended to graduate students employment, in order to see which are the specializations with the highest employment rates and which is the impact of students' practice for their employment. Also, we intend to apply data mining [2] and data analysis techniques like Formal Concept Analysis for the constructed data warehouse.

6. Acknowledgement.

The author Viorica Varga has been fully supported by Romanian Ministry of Education in the frame Grant CNCSIS PCCE-55/2008.

References

[1] Kimball, R., Ross, M., *The Data Warehouse Toolkit. The Complete Guide to Dimensional Modeling. Second Edition*, Wiley Computer Publishing (2002).

[2] Liu, S., Duffy, A.H.B., Whitfield, R.I., Boyle, I.M., *Integration of decision support systems to improve decision support performance*. Knowledge Information Systems, 22, 3, (2010), pp. 261-286.

[3] Osterwalder, A.,Pigneur, Y., *Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers*. John Wiley & Sons, (2010).

[4] Rizzi, S., *Collaborative Business Intelligence*, M.-A. Aufaure and E. Zimanyi (Eds.): eBISS 2011, LNBIP 96 (2012), pp. 186-205.

[5] Singh Khalsa, R.H., Reason, A., Biere, M., *The new era of collaborative business intelligence*, IBM (2010).

[6] US Department of Veterans Affairs, A Tutorial on the Zachman Architecture Framework (2002).

[7] VA Enterprise Architecture Innovation Team, *Enterprise Architecture: Strategy, Governance, & Implementation*, report Department of Veterans Affairs, August, (2001).

[8] Vinnik, S. Scholl, H. M.: *Decision Support System for Managing Educational Capacity Utilization*. IEEE Transactions on Education, ICECE05, Vol. 50, Issue:2 (2005), pp. 143-150.

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

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

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

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