Juhani Karhumäki | Yuri Matiyasevich
Aleksi Saarela (Eds.)

Proceedings of the Fourth Russian Finnish Symposium on Discrete Mathematics

Turku Centre *for* Computer Science

TUCS Lecture Notes
No 26, May 2017

Proceedings of the Fourth Russian Finnish Symposium on Discrete Mathematics

Organized in Turku on 16.-19.5.2017

Juhani Karhumäki, Yuri Matiyasevich, Aleksi Saarela (editors)

Preface

The fourth RuFiDiM conference, Russian-Finnish Symposium on Discrete Mathematics, took place in Turku in May, from 16th til 19th, 2017. This meeting was organized as a part of research activities between Steklov Institute of Mathematics of St. Petersburg and Department of Mathematics and Statistics of University of Turku. The goal of the conference series is to increase cooperation between Finnish and Russian mathematicians in discrete mathematics, but the symposium is open for a broader international audience. In the present event there were contributions from 10 different nations.

RuFiDiM 2017 consisted of six invited talks and 25 contributed presentations. The invited speakers were Volker Diekert (University of Stuttgart), Alexandr Kostochka (University of Illinois at Urbana-Champaign), Alexei Miasnikov (Stevens institute of Technology), Igor Potapov (University of Liverpool), Aleksi Saarela (University of Turku) and Jouko Väänänen (University of Helsinki). The program was chosen by the international program committee. Abstracts or extended abstracts of the lectures are presented in these preproceedings.

The organizers are grateful to the supporters of the symposium: Magnus Ehrnrooth Foundation, Turku University Foundation, Finnish Academy of Sciences (Mathematics Fund), and the University of Turku and Turku Centre for Computer Science.

Turku and St. Petersburg, April 2017

Juhani Karhumäki, Yuri Matiyasevich, Aleksi Saarela

Organization

Scientific committee:

- Juhani Karhumäki
- Jarkko Kari
- Tero Laihonen
- Vesa Halava
- Alexander Okhotin
- Svetlana Puzynina
- Yuri Matiyasevich
- Mikhail Volkov
- Dmitry Karpov
- Ilya Ponomarenko
- Vladimir Mazalov

Organizing committee:

- Juhani Karhumäki
- Yuri Matiyasevich
- Markus Whiteland

Contents

I Invited lectures

II Contributed talks

Part I
Invited lectures
Solving equations in free partially commutative monoids: developments over the past 20 years

Volker Diekert

Formale Methoden der Informatik, Universität Stuttgart, Germany

Abstract

Abstract. The topic WordEquation describes an exciting and active research field in discrete mathematics and combinatorics on words. Classical results by Makanin showed that the satisfiability problem for equations is decidable in free monoids (resp. free groups). Since then there have been two main directions: first, improve (or determine) its complexity, and second, relate the problem to the (existential) theory in a larger class of monoids or groups.

The focus in this note is on free partially commutative monoids and groups. The satisfiability problem for equations in trace monoids was shown to be decidable by Matiyasevich in 1996 and published in 1997. Since then, over the past twenty years there was considerable progress. For example, today we know that the full solution set of a given word equation over a trace monoid forms an EDT0L language and that an effective description as an EDT0L language can be constructed in quasi-linear space. Moreover, the results hold also for free partially commutative groups.

Survey

Trace monoids are used in computer science as a basic algebraic structure to model concurrency. This goes back to pioneering work of Keller [12] and Mazurkiewicz [17]. The latter paper also coined the term *trace monoid* to denote a finitely generated free partially commutative monoid. In combinatorics, trace monoids were investigated even earlier in connection with MacMahon's Master Theorem [2]. Free partially commutative groups are known as RAAGs (*right-angled Artin groups*), see [24] for the recent interest in RAAGs. Still another notation for RAAG is *graph group* [9]. This notation reflects that the setting for a free partially commutative monoid (resp. group) is given by an undirected graph (Γ, I). The vertices $a \in \Gamma$ are the generators and one adds a defining relation $ab = ba$ whenever $ab \in I$ is an edge. In the following $M(\Gamma, I) = \Gamma^* / \{ab = ba \mid ab \in I\}$ denotes a trace monoid and $G(\Gamma, I) = F(\Gamma) / \{ab = ba \mid ab \in I\}$ denotes a graph group where $F(\Gamma)$ is a free group over Γ and Γ is finite. In order to study equations over groups we assume that Γ is equipped with an *involution*. For a set, this is a mapping $x \mapsto \overline{x}$ such that $\overline{\overline{x}} = x$, for a monoid we additionally demand that $\overline{xy} = \overline{y}\,\overline{x}$. In the following, we assume that a generating set Γ is endowed with an involution. This is no restriction, since the identity is an involution on every set, but the notation of *free group* differs slightly from the standard one: $F(\Gamma)$ becomes $\Gamma^* / \{a\overline{a} = 1 \mid a \in \Gamma\}$. Such groups are called *specular* in [1] because there are elements of order 2 (if Γ has self-involuting elements).

An *equation* (over a monoid with involution M which is generated by some set Γ) is a pair (U, V), frequently written as $U = V$, where U, V are words in $(\Gamma \cup \mathcal{X})^*$. Here, $\mathcal{X}$ a set of variables. A *solution* of $U = V$ is a of variables by elements in M such that the resulting equation becomes an identity in M.

The satisfiability problem for free monoids and groups was solved affirmatively by Makanin in his seminal papers [14, 15]. It took twenty years from [14] when in 1997

Matiyasevich published his solution to the satisfiability problem in trace monoids, [16, 7]. His strategy was based on an induction on the size of commutation relations (that is the size of I) and a reduction to solve word equations with regular constraints. A problem which was already solved positively in the Habilitation thesis of Schulz [22] thereby generalizing [14]. The extension of Schulz's method to graph groups $G(\Gamma, I)$ did not work directly. A first step was to show that the satisfiability problem for equations with rational constraints in free groups is decidable [5]. The other obstacle was not the group structure, but the involution. More precisely, "cet obscur objet du désir" was a *normal form* for traces which respects the involution. Once the desired object was found, known techniques in trace theory did the rest: the satisfiability was shown to be decidable for trace monoids with involution (and hence also for graph groups) in [8].

The next step beyond satisfiability is to look for an algorithmic description of all solutions. In the case of free groups such a description was given by Razborov [21]. His description became known as a *Makanin-Razborov diagram*, a major tool in the positive solution of Tarski's conjectures about the elementary theory in free groups [13, 23]. Using sophisticated arguments from geometric and combinatorial group theory a generalization of Razborov-Makanin diagrams to graph groups was established by [3], but no algorithmic description of all solutions was known for trace monoids.

Clearly, the complexity of the problems is of interest, too. Makanin's algorithms were famous for the difficulty of the termination proof and also for the extremely high complexity. A breakthrough on the complexity was obtained by Plandowski and Rytter [20], who recognized compression as a key tool for solving word equations. Indeed, Plandowski showed via compression that the satisfiability problem for word equations is in PSPACE [18]. It took more than a decade before Jeż lowered the complexity down to $\mathsf{NSPACE}(n \log n)$ in [10]. (Very recently, Jeż proved $\mathsf{NSPACE}(n)$: the set of solvable word equations is context-sensitve [11].) More importantly, his "recompression technique", simplified all existing proofs for solving word equations; and it provided an effective representation of all solutions. A similar representation was given earlier by Plandowski [19]. The technique of Jeż is based on two simple rules: compress large powers a^λ and pairs ab into fresh letters. Here, a, b denote letters which occur in a given solution of an equation. Reading the process backwards the process can be described by applying endomorphisms of a free monoid over an extended alphabet defined by $a_\lambda \mapsto a^\lambda$ resp. $c \mapsto ab$ and leaving all other letters invariant. This observation led to a remarkable structural property of the full solution set of equations in trace monoids and graph groups: they are EDT0L. As a corollary, it is decidable whether or not a given equation has at least one, finitely many or infinitely many solutions in $\mathsf{NSPACE}(n \log n)$, see [4] for the special case of free monoids and free groups; and see [6] for its generalization to partial commutation.

References

[1] V. Berthé, C. D. Felice, V. Delecroix, F. Dolce, J. Leroy, D. Perrin, C. Reutenauer, and G. Rindone. Specular sets. In F. Manea and D. Nowotka, editors, *Combinatorics on Words - 10th International Conference, WORDS 2015, Kiel, Germany, September 14-17, 2015, Proceedings*, volume 9304 of *Lecture Notes in Computer Science*, pages 210–222. Springer, 2015.

[2] P. Cartier and D. Foata. *Problèmes combinatoires de commutation et réarrangements.* Number 85 in Lecture Notes in Mathematics. Springer-Verlag, Heidelberg, 1969.

[3] M. Casals and I. Kazachkov. On systems of equations over partially commutative groups. *Memoirs Amer. Math. Soc.*, 212:1–153, 2011.

[4] L. Ciobanu, V. Diekert, and M. Elder. Solution sets for equations over free groups are EDT0L languages. *International Journal of Algebra and Computation*, 26:843–886, 2016. Conference abstract in ICALP 2015, LNCS 9135 with full version on ArXiv e-prints: abs/1502.03426.

[5] V. Diekert, C. Gutiérrez, and Ch. Hagenah. The existential theory of equations with rational constraints in free groups is PSPACE-complete. *Information and Computation*, 202:105–140, 2005. Conference version in STACS 2001, LNCS 2010, 170–182, 2004.

[6] V. Diekert, A. Jeż, and M. Kufleitner. Solutions of Word Equations Over Partially Commutative Structures. In Y. R. Ioannis Chatzigiannakis, Michael Mitzenmacher and D. Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 127:1–127:14, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[7] V. Diekert, Yu. Matiyasevich, and A. Muscholl. Solving word equations modulo partial commutations. *Theoretical Computer Science*, 224:215–235, 1999. Special issue of LFCS'97.

[8] V. Diekert and A. Muscholl. Solvability of equations in free partially commutative groups is decidable. *International Journal of Algebra and Computation*, 16:1047–1070, 2006. Conference version in Proc. ICALP 2001, 543–554, LNCS 2076.

[9] C. Droms. Graph groups, coherence and three-manifolds. *Journal of Algebra*, 106(2):484–489, 1985.

[10] A. Jeż. Recompression: a simple and powerful technique for word equations. *J. ACM*, 63(1):4:1–4:51, 2016. Conference version in Proc. STACS 2013.

[11] A. Jeż. Word equations in linear space. *ArXiv e-prints*, Feb. 2017.

[12] R. M. Keller. Parallel program schemata and maximal parallelism I. Fundamental results. *J. ACM*, 20(3):514–537, 1973.

[13] O. Kharlampovich and A. Myasnikov. Elementary theory of free non-abelian groups. *J. of Algebra*, 302:451–552, 2006.

[14] G. S. Makanin. The problem of solvability of equations in a free semigroup. *Math. Sbornik*, 103:147–236, 1977. English transl. in Math. USSR Sbornik 32 (1977).

[15] G. S. Makanin. Equations in a free group. *Izv. Akad. Nauk SSR*, Ser. Math. 46:1199–1273, 1983. English transl. in Math. USSR Izv. 21 (1983).

[16] Yu. Matiyasevich. Some decision problems for traces. In S. Adian and A. Nerode, editors, *Proceedings of the 4th International Symposium on Logical Foundations of Computer Science (LFCS'97), Yaroslavl, Russia, July 6–12, 1997*, volume 1234 of *Lecture Notes in Computer Science*, pages 248–257, Heidelberg, 1997. Springer-Verlag. Invited lecture.

[17] A. Mazurkiewicz. Concurrent program schemes and their interpretations. DAIMI Rep. PB 78, Aarhus University, Aarhus, 1977.

[18] W. Plandowski. Satisfiability of word equations with constants is in PSPACE. *J. ACM*, 51:483–496, 2004. Conference version in Proc. FOCS'99.

[19] W. Plandowski. An efficient algorithm for solving word equations. In J. M. Kleinberg, editor, *STOC*, pages 467–476. ACM, 2006.

[20] W. Plandowski and W. Rytter. Application of Lempel-Ziv encodings to the solution of word equations. In K. G. Larsen et al., editors, *Proc. 25th International Colloquium Automata, Languages and Programming (ICALP'98), Aalborg (Denmark), 1998*, volume 1443 of *Lecture Notes in Computer Science*, pages 731–742, Heidelberg, 1998. Springer-Verlag.

[21] A. A. Razborov. *On Systems of Equations in Free Groups.* PhD thesis, Steklov Institute of Mathematics, 1987. In Russian.

[22] K. U. Schulz. Makanin's algorithm for word equations — Two improvements and a generalization. In K. U. Schulz, editor, *Word Equations and Related Topics*, volume 572 of *Lecture Notes in Computer Science*, pages 85–150, Heidelberg, 1991. Springer-Verlag.

[23] Z. Sela. Diophantine geometry over groups VIII: Stability. *Ann. of Math.*, 177:787–868, 2013.

[24] D. Wise. *From Riches to Raags: 3-Manifolds, Right-Angled Artin Groups, and Cubical Geometry.* American Mathematical Society, 2012.

Recent results on disjoint and longest cycles in graphs

Alexandr Kostochka

The goal of the talk is to discuss recent refinements of some results on cycle structure of graphs from the sixties.

Let $V_{\geq t}(G)$ (respectively, $V_{\leq t}(G)$) denote the set of vertices in G of degree at least (respectively, at most) t. Dirac and Erdős in 1963 extended the Corrádi-Hajnal Theorem as follows: If $k \geq 3$ and G is a graph with $|V_{\geq 2k}(G)| - |V_{\leq 2k-2}(G)| \geq k^2 + 2k - 4$, then G has k disjoint cycles. They also showed a series of examples of graphs G with $|V_{\geq 2k}(G)| - |V_{\leq 2k-2}(G)| = 2k - 1$ that do not have k disjoint cycles.

We show that each graph G with $|V_{\geq 2k}(G)| - |V_{\leq 2k-2}(G)| \geq 3k$ has k disjoint cycles. This is sharp if we do not impose restrictions on $|V(G)|$. We also show that if $|V(G)| \geq 19k$, then the result holds with $2k$ instead of $3k$. This is joint work with Kierstead and McConvey.

The theorems of Erdős and Gallai from 1959 on the most edges in n-vertex graphs that do not have paths/cycles with at least k vertices were sharpened later by Faudree and Schelp, Woodall, and Kopylov. Let $h(n,k,a) = \binom{k-a}{2} + a(n-k+a)$. The strongest result (by Kopylov) was: if $t \geq 2$, $k \in \{2t+1, 2t+2\}$, $n \geq k$, and G is an n-vertex 2-connected graph with at least $\max\{h(n,k,2), h(n,k,t)\}$ edges, then G contains a cycle of length at least k, unless $G = H_{n,k,t} := K_n - E(K_{n-t})$. We prove stability versions of these results. In particular, if $k \geq 3$ is odd, $n \geq k$ and the number of edges in an n-vertex 2-connected graph G with no cycle of length at least k is greater than $\max\{h(n,k,3), h(n,k,t-1)\}$, then G is a subgraph of $H_{n,k,t}$ or of $H_{n,k,2}$. This is joint work with Füredi, Luo and Verstraëte.

Non-commutative discrete optimization: geometry, compression, and complexity

Alexei Miasnikov

Decision problems in matrix semigroups: limitations and algorithms

Igor Potapov

Department of Computer Science, University of Liverpool, Email: potapov@liverpool.ac.uk.

Matrices and matrix products play a crucial role in a representation and analysis of various computational processes [6, 12, 13]. However, many simply formulated and elementary problems for matrices are inherently difficult to solve even in dimension two, and most of these problems become undecidable in general starting from dimension three or four [1, 2, 4, 6, 7, 14]. Let us given a finite set of square matrices (known as a generator) which is forming a multiplicative semigroup S. The classical computational problems for matrix semigroups are:

- Membership (Decide whether a given matrix M belong to a semigroup S) and two special cases such as: Identity (i.e. if M is the identity matrix) and Mortality (i.e. if M is the zero matrix) problems
- Vector reachability (Decide for a given vectors u and v whether exist a matrix M in S such that $M \cdot u = v$)
- Scalar reachability (Decide for a given vectors u, v and a scalar L whether exist a matrix M in S such that $u \cdot M \cdot v{=}L$)
- Freeness (Decide whether every matrix product in S is unique, i.e. whether it is a code) and some variants of the freeness such as finite freeness problem, the recurrent matrix problem, the unique
- factorizability problem, vector freeness problem, vector ambiguity problems, etc.

The undecidability proofs in matrix semigroups are mainly based on various techniques and methods for embedding universal computations into matrix products. The case of dimension two is the most intriguing since there is some evidence that if these problems are undecidable, then this cannot be proved directly using previously known constructions. Due to a severe lack of methods and techniques the status of decision problems for 2×2 matrices (like membership, vector reachability, freeness) is remaining to be a long standing open problem not only for matrices over algebraic, complex, rational numbers but also for integer matrices.

Recently, a new approach of translating numerical problems of $2x2$ integer matrices into variety of combinatorial and computational problems on words and automata over group alphabet and studying their transformations as specific rewriting systems [8, 9] have led to a few results on decidability and complexity for some subclasses:

- The membership problem for $2x2$ nonsingular integer matrices is decidable [16]. The algorithm relies on a translation of numerical problems on matrices into combinatorial problems on words. It also makes use of some algebraic properties of well-known subgroups of GL(2, Z) and various new techniques and constructions that help to convert matrix equations into the emptiness problem for intersection of regular languages.
- The Identity problem in SL(2,Z) is NP-complete [5, 3]. Our NP algorithm is based on various new techniques that allow us to operate with compressed word representations of matrices without explicit exponential expansion.

- The vector reachability problem over a finitely generated semigroup of matrices from SL(2,Z) and the point to point reachability (over rational numbers) for fractional linear transformations, where associated matrices are from SL(2,Z) are decidable [15].

Finally our new techniques have been applied to show that the freeness problem is co-NP-hard [11] as well as to study the complexity of other freeness problems such as finite freeness problem, the recurrent matrix problem, the unique factorizability problem, vector freeness problem, vector ambiguity problems, etc [10].

References

[1] Paul Bell and Igor Potapov. On undecidability bounds for matrix decision problems. *Theoretical Computer Science*, 391(1-2):3–13, 2008.

[2] Paul C. Bell, Mika Hirvensalo, and Igor Potapov. Mortality for 2x2 matrices is NP-hard. In Branislav Rovan, Vladimiro Sassone, and Peter Widmayer, editors, *Mathematical Foundations of Computer Science 2012*, volume 7464 of *Lecture Notes in Computer Science*, pages 148–159. Springer Berlin Heidelberg, 2012.

[3] Paul C. Bell, Mika Hirvensalo, and Igor Potapov. The Identity Problem for Matrix Semigroups in SL2(Z) is NP-complete. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 187-206, 2017.

[4] Paul C. Bell and Igor Potapov. On the undecidability of the identity correspondence problem and its applications for word and matrix semigroups. *Int. J. Found. Comput. Sci.*, 21(6):963–978, 2010.

[5] Paul C. Bell and Igor Potapov. On the computational complexity of matrix semigroup problems. *Fundam. Inf.*, 116(1-4):1–13, January 2012.

[6] Vincent D. Blondel, Emmanuel Jeandel, Pascal Koiran, and Natacha Portier. Decidable and undecidable problems about quantum automata. *SIAM J. Comput.*, 34(6):1464–1473, June 2005.

[7] Julien Cassaigne, Vesa Halava, Tero Harju, and François Nicolas. Tighter undecidability bounds for matrix mortality, zero-in-the-corner problems, and more. *CoRR*, abs/1404.0644, 2014.

[8] Christian Choffrut and Juhani Karhumaki. Some decision problems on integer matrices. *RAIRO-Theor. Inf. Appl.*, 39(1):125–131, 2005.

[9] Yuri Gurevich and Paul Schupp. Membership problem for the modular group. *SIAM J. Comput.*, 37(2):425–459, May 2007.

[10] Sang-Ki Ko, and Igor Potapov Vector Ambiguity and Freeness Problems in SL(2,Z), Theory and Applications of Models of Computation: 14th Annual Conference, TAMC 2017, Bern, Switzerland, April 20-22, 2017, Proceedings, 2017, LNCS Springer, 373–388.

[11] Sang-Ki Ko, and Igor Potapov. Matrix Semigroup Freeness Problems in SL(2,Z). SOFSEM 2017: Theory and Practice of Computer Science: 43rd International Conference on Current Trends in Theory and Practice of Computer Science, 2017, LNCS Springer, 268–279.

[12] Joël Ouaknine, João Sousa Pinto, and James Worrell. On termination of integer linear loops. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pages 957–969. SIAM, 2015.

[13] Joël Ouaknine and James Worrell. On the positivity problem for simple linear recurrence sequences,. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part II*, pages 318–329, 2014.

[14] M. S. Paterson. Unsolvability in 3x3 matrices. *Studies in Applied Mathematics*, 49(1):pp.105–107, 1970.

[15] Igor Potapov and Pavel Semukhin. Vector reachability problem in SL(2, Z). MFCS 2016. *CoRR*, abs/1510.03227, 2015. http://arxiv.org/abs/1510.03227.

[16] Igor Potapov and Pavel Semukhin. Decidability of the membership problem for 2 x 2 integer matrices. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 170–186, 2017.

Studying Word Equations by Geometric and Algebraic Methods

Aleksi Saarela

Department of Mathematics and Statistics, University of Turku, 20014 Turku, Finland
amsaar@utu.fi

We discuss two well-known questions on word equations, two methods for analyzing them, and the possible connections between these topics.

The first question we are interested in concerns equations of the form

$$x_0^k = x_1^k \cdots x_n^k, \tag{1}$$

where $x_0, \ldots, x_n$ are words and k is a positive integer. A simple example would be $(ababa)^k = (ab)^k a^k (ba)^k$, which holds for $k \in \{1, 2\}$, but not for any larger k. These equations and their generalizations are connected to many topics: Pumping properties of formal languages, test sets, Sturmian words, and constructions of large independent systems of word equations. They seem like simple equations with a lot of structure, but some simple questions about them have been open for a long time. Specifically, if (1) holds for three positive values of k, does this imply that the words $x_0, \ldots, x_n$ commute, that is, $x_i x_j = x_j x_i$ for all i, j? In some form, this question has been open for at least about two decades. A special case appeared in an article by Hakala and Kortelainen [4]. Since then, there have been many articles and partial results, see, e.g., the article by Holub [5]. Recently, the question was finally answered by Saarela [10], so we have the following theorem.

Theorem 1. *Let $x_0, \ldots, x_n$ be words. If $x_0^k = x_1^k \cdots x_n^k$ for three positive integers k, then the words $x_0, \ldots, x_n$ commute.*

Let us now outline the ideas behind the proof. We can assume that our alphabet Γ is a subset of $\mathbb{R}$ (this is not a restriction; we can assign numerical values to the letters in any way we like, as long as no two letters get the same value). This allows us to define $\Sigma(w)$ to be the sum of the letters of a word $w \in \Gamma^*$, that is, if $w = a_1 \cdots a_n$ and $a_1, \ldots, a_n \in \Gamma$, then $\Sigma(w) = a_1 + \cdots + a_n$. Words w such that $\Sigma(w) = 0$ are called *zero-sum words*. The *prefix sum word* of w is the word $\mathrm{psw}(w) = b_1 \cdots b_n$, where $b_i = \Sigma(a_1 \cdots a_i)$ for all i. The word w can be represented as a plane curve by connecting the points $(0, 0), (1, b_1), \ldots, (n, b_n)$. The properties of this curve can then be studied, leading to a geometric way of analyzing words.

Theorem 1 can be proved by first normalizing the alphabet so that x_0 becomes a zero-sum word. If all x_i are zero-sum, we can compress them by writing them as products of minimal zero-sum words. After compression and normalization (possibly repeated several times), either the words x_i are unary, which is a trivial case, or x_0 is zero-sum but at least one x_i is not, in which case we can analyze the curves of the x_0^k and $x_1^k \cdots x_n^k$ and see that they can be equal for at most two values of k. This leads to a proof of Theorem 1.

The second question we are interested in is the following: For a given n, what is the maximal size of an independent system of constant-free word equations on n variables? It is known that every system of word equations is equivalent to a finite subsystem and, consequently, every independent system is finite. This is known as *Ehrenfeucht's compactness property*. It was conjectured by Ehrenfeucht in a language theoretic setting, formulated in terms of word equations by Culik and Karhumäki [2], and proved by Albert and Lawrence [1] and independently by Guba [3]. If $n > 2$, no finite upper bound for the

size of independent systems is known. The largest known independent systems have size $\Theta(n^4)$ [7]. The difference between the best known lower and upper bounds is particularly striking in the case of three variables: The largest known independent systems consist of just three equations, but it is not even known whether there exists a constant c such that every independent system has size c or less.

There have been some recent advances regarding this topic. The first nontrivial upper bound was proved by Saarela [9]: The size of an independent system on three variables is at most quadratic with respect to the length of the shortest equation in the system. This bound was improved to a linear one by Holub and Žemlička [6]. A logarithmic bound was proved by Nowotka and Saarela [8]. The first two results can be generalized for more than three variables if the assumption of independence is replaced by a stronger variation. The third result, on the other hand, is specific to the case of three variables.

Ehrenfeucht's compactness property can be proved by using polynomials and Hilbert's basis theorem. Also the proofs in [9] and [6] are based on polynomials. If we keep the assumption that letters are numbers, then a word $a_0 \cdots a_n$ can be represented by the polynomial $a_0 + a_1 X + \cdots + a_n X^n$. Studying these polynomials, together with using linear algebra over the field of rational functions and over the field of real numbers, leads to results about independent systems of word equations.

There are also some potential connections between the topics we have presented. For example, the sizes of independent systems of three-variable word equations could possibly be studied by using the geometric method. On the other hand, zero-sum words and prefix sum words could be studied by using the algebraic method.

References

[1] M. H. Albert and J. Lawrence. A proof of Ehrenfeucht's conjecture. *Theoret. Comput. Sci.*, 41(1):121–123, 1985.

[2] Karel Culik, II and Juhani Karhumäki. Systems of equations over a free monoid and Ehrenfeucht's conjecture. *Discrete Math.*, 43(2–3):139–153, 1983.

[3] V. S. Guba. Equivalence of infinite systems of equations in free groups and semigroups to finite subsystems. *Mat. Zametki*, 40(3):321–324, 1986.

[4] Ismo Hakala and Juha Kortelainen. On the system of word equations $x_1^i x_2^i \cdots x_m^i = y_1^i y_2^i \cdots y_n^i$ $(i = 1, 2, \cdots)$ in a free monoid. *Acta Inform.*, 34(3):217–230, 1997.

[5] Štěpán Holub. Local and global cyclicity in free semigroups. *Theoret. Comput. Sci.*, 262(1–2):25–36, 2001.

[6] Štěpán Holub and Jan Žemlička. Algebraic properties of word equations. *J. Algebra*, 434:283–301, 2015.

[7] Juhani Karhumäki and Wojciech Plandowski. On the defect effect of many identities in free semigroups. In Gheorghe Paun, editor, *Mathematical aspects of natural and formal languages*, pages 225–232. World Scientific, 1994.

[8] Dirk Nowotka and Aleksi Saarela. A connection between one-unknown word equations and constant-free three-unknown word equations. In *Proceedings of the 20th DLT*, volume 9840 of *LNCS*, pages 332–343. Springer, 2016.

[9] Aleksi Saarela. Systems of word equations, polynomials and linear algebra: A new approach. *European J. Combin.*, 47:1–14, 2015.

[10] Aleksi Saarela. Word equations where a power equals a product of powers. In *Proceedings of the 34th STACS*, volume 66 of *LIPIcs*, pages 55:1–55:9. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017.

On the logic of dependence and independence

Jouko Väänänen

I will give a quick introduction to team semantics i.e. semantics in which meaning of formulas is defined relative to a set of valuations rather than relative to a single valuation. I use this semantics in the context of propositional logic and describe a hierarchy of extensions of classical propositional logic obtained by adding distinguished relations between propositional atoms, such as dependence, inclusion and independence. I give semantic characterizations as well as complete axiomatizations of several such extensions, and will also list some open problems. This is joint work with Fan Yang.

Part II
Contributed talks
Decision Problems and Applications of Rational Sets of Regular Languages

Sergey Afonin

Moscow State University

Abstract

In this paper we consider known results on decision problems for finitely generated semigroups and rational sets of regular languages and prove some statements on various topics. In particular, we prove undecidability of the equivalence problem for rational set of finite languages, automaticity of finitely generated semigroups of factorial languages, and provide an algorithm for automatic presentation construction in case of regular factorial languages. We also discuss possible applications of rational sets of regular languages.

1 Introduction

Regular languages and finite automata are among basic concepts of computer science. Most decision problems for regular languages are effectively decidable and corresponding algorithms are widely used in software and hardware applications. In this paper we consider decision problems for finite or infinite *sets* of regular languages. This paper has a threefold purpose: we collect known result on this topic, prove some new statements, and formulate questions for future research.

The main object of this paper is the *rational set of regular languages*. Let us consider a finite set of regular languages over some alphabet. As regular languages are closed under concatenation, which is an associative operation, the set of all finite concatenations of elements of this finite set forms a (finitely generated) semigroup. A set of regular languages is called *rational*, if it is a rational subset of a finitely generated semigroup of languages. Rational sets can be effectively presented by means of regular language substitutions. Let $\varphi : \Delta \to \mathrm{Reg}(\Sigma)$ be a regular language substitution, where Σ and Δ are finite alphabets, and $\mathrm{Reg}(\Sigma)$ stands for the set of all regular languages over Σ. We can extend φ to words in a natural way: $\varphi(uv) = \varphi(u)\varphi(v)$ for all $u, v \in \Delta^+$. Every regular language K in Δ defines a rational set of languages $\mathcal{R} = \{\varphi(w) \mid w \in K\}$.

There are two kinds of problems addressed in this paper. First, we consider decision problems such as checking membership of a regular language in a given set, checking the finiteness of a rational set defined by the given representation, checking the equivalence of two rational sets, or checking minimality of a representation. The second kind of problems is related to the word problem for semigroups of regular languages and motivated by the following question. Given a regular language substitution $\varphi : \Delta \to \mathrm{Reg}(\Sigma)$ and two words $u, v \in \Delta^+$ one can decides that $u = v$ by checking the equivalence of non-deterministic finite automata $\varphi(u)$ and $\varphi(v)$. The question is: Is it possible to *precompute* some information for the given substitution φ, such that subsequent word equivalences can be efficiently processed? This leads us to the notion of an automatic presentation of a semigroup of regular languages.

Automatic semigroups are finitely generated semigroups such that multiplication by generators is realized by finite automata (formal definition presented in the next section). Many naturally appearing semigroups, e.g. finitely generated subsemigroups of a free semigroup are automatic [9]. If an automatic structure of a semigroup is given, then word equivalence $u = v$ may be checked in quadratic time with respect to the length of u and v. Note, that complexity does not depend on number of states of corresponding automata, so an automatic structure for the semigroup associated with a language substitute φ is a basis for efficient equivalence checks. It is not known whether semigroups of regular languages are automatic or not. We prove autmaticity in case of factorial languages.

The layout of the paper is the following. In the next section we introduce the notation. In Section 3 basic decision problems for rational sets of regular languages are described. The structure of finitely generated semigroups is considered in Section 4. In particular, we prove that semigroups of factorial languages are automatic. Applications of rational sets of regular languages in such areas as database systems, cryptography, and software testing frameworks are briefly discussed in Section 5. The list of open questions concludes the paper.

2 Definitions and Notation

Let $\mathrm{Reg}(\Sigma)$ denote the set of all regular languages over alphabet Σ. As regular languages are closed under concatenation by definition and concatenation is an associative operation the set $(\mathrm{Reg}(\Sigma), \cdot)$ forms a semigroup. In the sequel we will only consider semigroups with concatenation as a product and this operation will not be explicitly denoted. Every finite set $V \subset \mathrm{Reg}(\Sigma)$ defines a (finitely generated) subsemigroup $\langle V \rangle$ of $\mathrm{Reg}(\Sigma)$. Let Δ be a fresh alphabet such as $|\Delta| = |V|$ and $\varphi : \Delta \to \mathrm{Reg}(\Sigma)$ be a regular language substitution defined as $\varphi(\delta_i) \mapsto V_i$. We can extend φ to a homomorphism between Δ^+ and $\langle V \rangle$ in a natural way: $\varphi(uv) = \varphi(u)\varphi(v)$ for all $u, v \in \Delta^+$. We call a set of regular languages over Σ *rational* if it is a rational subset of a finitely generated subsemigroup of $\mathrm{Reg}(\Sigma)$. Thus, any rational set of regular languages may be represented as a pair (K, φ), where $K \in \mathrm{Reg}(\Delta)$ and $\varphi : \Delta^+ \to \mathrm{Reg}(\Sigma)$.

We now define automatic semigroups [9]. Let S be a semigroup, A be a finite set, L be a regular language over A, and $\psi : A^+ \to S$ be a homomorphism with $\psi(L) = S$. The pair (A, L) is called an *automatic structure* for S if

1. $L_= = \{(u, v) \mid u, v \in L, \psi(u) = \psi(v)\}$ is regular;
2. $L_a = \{(u, v) \mid u, v \in L, \psi(ua) = \psi(v)\}$ is regular for each $a \in A$.

A semigroup S is called *automatic* if it has an automatic structure (A, L) for some A and L. Let us note that a semigroup may have an automatic structure with respect to one generating set, but not with respect to another. It is also known that an automatic semigroup S always has an automatic structure (A, L) such that L is the set of *unique normal forms*, i.e. $\psi(u) \neq \psi(v)$ for all $u, v \in L$.

If an automatic structure of a semigroup is given, then word equivalence $u = v$ may be checked by the following procedure. First, find two words $n(u), n(v) \in L$ such that $\psi(u) = \psi(n(u))$ and $\psi(v) = \psi(n(v))$. Let $u = a_1 \dots a_k \in A^+$. The word $n(u)$ can be found as follows. Find a word $n(a_1) \in L$ using the automaton L_{a_1}: the pair $(\varepsilon, n(a_1))$ is in L_{a_1}. Then find the pair $(n(a_1), n(a_1 a_2))$ is in L_{a_2}, and so on. We have $u = v$ iff $(n(u), n(v)) \in L_=$ (or just $n(u) = n(v)$, if L is the set of unique normal forms).

3 Decision Problems for Rational Sets

One can consider various subclasses of rational sets of languages, e.g., sets of finite languages, factorial languages, or subclasses of regular languages. In this section we consider the general case.

3.1 Membership and Approximation of Languages

Let $\mathcal{R} = (K, \varphi)$ be a rational set of regular languages over Σ and $L \in \mathrm{Reg}(\Sigma)$. The membership problem for $\mathcal{R}$ and L is the set $\{w \in K \mid \varphi(w) = L\}$. By *approximation* of a language L by $\mathcal{R}$ we mean some subset $A \subseteq K$ such that the language $\tilde{L} = \bigcup\limits_{w \in A} \varphi(w)$ satisfies either $\tilde{L} \subseteq L$ (approximation from below), or $L \subseteq \tilde{L}$ (approximation from above).

Membership of a language L in a semigroup or rational set of languages is a non-trivial question only if the empty word ε belongs to some of the generators. Otherwise, one can establish an upper bound for the length of a decomposition (word w such that $\varphi(w) = L$) using lengths of shortest words in L and the generators.

Theorem 1 (Calvanese et al. [7], K. Hashiguchi [13])**.** *Let $\varphi : \Delta^+ \to \mathrm{Reg}(\Sigma)$ be a regular language substitution. For any regular language $L \subseteq \Sigma^*$ the* maximal rewriting

$$M_\varphi(L) = \{w \in \Delta^+ \mid \varphi(w) \subseteq L\}$$

is a regular language over Δ.

This theorem states that the maximal approximation from below of a language is regular. The approximation is called *exact* if $\bigcup\limits_{w \in M} \varphi(w) = L$.

The regularity of the membership problem for a semigroup of regular languages is a simple corollary from Theorem 1 and Higman's lemma.

Theorem 2. *Let $\varphi : \Delta^+ \to \mathrm{Reg}(\Sigma)$ be a regular language substitution and w be a word in Δ^+. The membership problem*

$$[w] = \{u \in \Delta^+ \mid \varphi(u) = \varphi(w)\}$$

is a regular language over Δ.

Proof. Let $w = \delta_{i_1} \dots \delta_{i_m}$ be a word over Δ, $A \subseteq \Delta$, and $\Delta_0 = \{\delta \in \Delta \mid \varepsilon \in \varphi(\delta)\}$. By $E(w, A)$ we denote the regular language $(A^*\delta_{i_1}A^*\delta_{i_2} \dots A^*\delta_{i_m}A^* \cap M_\varphi(\varphi(w))$.

For every $u \in \Delta^+$ and $v \in E(u, \Delta_0)$ we have $\varphi(u) = \varphi(v)$. Indeed, let $\delta \in \Delta_0$. Consider the word $v = u_1\delta u_2$, where $u_1, u_2 \in \Delta^*$ and $u = u_1u_2$. We have

$$\varphi(u) \subseteq \varphi(v) \subseteq \varphi(u).$$

The first inclusion is due to $\varepsilon \in \varphi(\delta)$ while the second one follows from the definition of the language $E(u, \Delta_0)$.

By Theorem 1 the set $M_\varphi(w) = \{u \in \Delta^+ \mid \varphi(u) \subseteq \varphi(w)\}$ is regular. Clearly, $[w] \subseteq M_\varphi(w)$. We prove now that there exists a finite subset $F \subseteq [w]$ satisfying

$$[w] = \bigcup_{u \in F} E(u, \Delta_0).$$

Note that if u is a scattered subword of v then $E(v, \Delta_0) \subseteq E(u, \Delta_0)$, so without loss of generality we may assume that if the language F contains a word v then it does not

contain subwords of v. The finiteness of F follows immediately from Higman's lemma which states that in every infinite sequence $\{u_i\}_{i\geqslant 1}$ of words over a finite alphabet there exist indices i and j, such that u_i is a scattered subword of u_j. □

As a trivial corollary we have

Theorem 3. *The membership problem for rational set $\mathcal{R} = (K, \varphi)$ is a regular language.*

Finding a minimal approximation of L from above, as a natural counterpart to the maximal approximation of Theorem 1, is nontrivial. Let $\mathcal{R}$ be a semigroup, i.e. $K = \Delta^+$. The set Δ^+ may be represented as a disjoint union of three regular languages: $R_C = M_\varphi(L)$, $R_I = \{w \in \Delta^+ \mid \varphi(w) \cap L \neq \varnothing\}$, and $E = \Delta^+ \setminus (R_I \cup R_I)$. Minimal approximation of L by $\mathcal{R}$ from above is the union of R_C and some subset of R_I. In general, there exists infinitely many languages $R \subsetneq R_I$ such that $\varphi(R) = \varphi(R_I)$. One can propose an algorithm to construct a minimal representation in the spirit of Theorem 2 where a random word w is drawn form R_I and the set R_I replaced by $R_I \setminus [w]$, but the finiteness of this procedure is not guaranteed. To the best of our knowledge, no algorithm for construction of minimal approximation is known.

3.2 Equivalence and Inclusion

Equivalence and inclusion problems for rational sets of finite languages are undecidable due to the following Theorem.

Theorem 4. *The problem of checking equivalence of two rational sets of finite languages is undecidable.*

Proof. The proof is based on a reduction to the equivalence problem of finite substitutions. Let φ_1 and φ_2 be *finite substitutions*, i.e. homomorphisms between Δ^+ and semigroups of finite languages. The equivalence problem of finite substitutions on a regular language L asks for an algorithm to check that

$$\varphi_1(w) = \varphi_2(w) \text{ for all } w \in L.$$

It is known that this is undecidable[19] for $L = xy^*z$.

Let $\Delta = \{x_1, y_1, z_1, x_2, y_2, z_2\}$, φ be a finite substitution, and rational sets $\mathcal{R}_1 = (K_1, \varphi)$ and $\mathcal{R}_2 = (K_2, \varphi)$ are given by languages $K_1 = x_1y_1^*z_1$ and $K_2 = x_2y_2^*z_2$.

Rational sets $\mathcal{R}_1$ and $\mathcal{R}_2$ are equal if and only if finite substitutions φ_1 and φ_2 (induced by φ in a natural way) are equal on the language xy^*z. If $\varphi_1(xz) = \varphi_2(xz)$ and $\varphi_1(xyz) = \varphi_2(xyz)$, then by considering the length of the longest word in the image $\varphi(w)$, we have that for all $k \neq m \Rightarrow \varphi_1(xy^kz) \neq \varphi_2(xy^mz)$ for all $k, m > 1$. □

Let us note, that the inclusion problem is obviously decidable for finitely generated semigroups of regular languages. It is sufficient to check membership of all generators of one semigroup in another. It could be interesting to establish decidability of equivalence of a rational set and a semigroup.

3.3 Properties of a Representation

A rational set of regular languages may be represented in infinitely many ways, in general. One representation could be more appropriate than another. Let us call a representation $\mathcal{R} = (K, \varphi)$ *K-minimal* if for every $K' \subset K$ the set $\mathcal{R}' = (K', \varphi)$ is a proper subset of $\mathcal{R}$, and *Δ-minimal* if for every representation (K', φ') of $\mathcal{R}$ inequality $|\Delta'| < |\Delta|$ holds.

Theorem 5. *It is undecidable whether or not given representation (K, φ) is K-minimal.*

Proof. Trivially follows from undecidability of checking that given set of regular languages forms a code in the monoid of languages [18]. □

In [1] it was shown that for a given semigroup $\mathcal{S} = (\Delta^+, \varphi)$ one can find a representation with minimal number of generators. The search space for generators consists of all possible intersections of maximal factors of given generators, i.e. intersections of factors of $\{\varphi(\delta) \mid \delta \in \Delta\}$. Let us recall that a language F is called a *maximal factor* of L if there exists a natural number m such that F is a component of some maximal solution to the equation $L = X_1 \dots X_m$. In the case of general rational set $\mathcal{R} = (K, \varphi)$ the Δ-minimal set of generators for corresponding semigroup $\mathcal{S} = (\Delta^+, \varphi)$ may not be minimal for $\mathcal{R}$. For example, if $K = (\delta_1\delta_2)^+$ then the set can be represented using single generator.

Theorem 6. *There exists an algorithm that computes a Δ-minimal representation for a given rational set $\mathcal{R} = (K, \varphi)$.*

Sketch. Consider the sequence of finite sets $F_1, F_2, \dots$ defined as $F_n = \{\varphi(w) \mid w \in K \wedge |w| \leqslant n\}$. Every Δ-minimal representation for $\mathcal{R}$ is a representation for all sets in this sequence. Using the technique of maximal factors it is possible to compute the finite set of Δ-minimal representation for any set F_n. There exist number N, that depends on the number of states of finite automata representing K, such that for every $n > N$ every Δ-minimal representation of F_n is a Δ-minimal representation of $\mathcal{R}$. □

Let us note, that representation of rational sets may me used as a description complexity measure of finite set of languages. Given a complexity measure C of regular languages, e.g. state complexity of corresponding deterministic automata, we can define complexity of a regular language substitution φ as $C(\varphi) = \sum_{\delta \in \Delta} C(\varphi(\delta))$. The complexity of a finite set $\mathcal{F} = \{L_1, \dots, L_n\}$ of regular languages may be defined as

$$C(\mathcal{F}) = \min_{\varphi, R_1, \dots, R_n} \left(C(\varphi) + \sum_{i=1}^{n} C(R_i) \right),$$

where minimum is taken over all languages $R_i \subset \Delta^*$ such that $\cup_{w \in R_i} \varphi(w) = L_i$. Informally, we are looking for a set of regular languages, a basis, such that every element in $\mathcal{F}$ can be represented in terms of the basis and the total complexity of both the basis and representation languages is minimal.

3.4 Closure Properties

Closure properties for rational sets of regular languages were investigated in [16]. It was shown that rational sets are closed under:

concatenation	$\mathcal{R}_1 \cdot \mathcal{R}_2 = \{L_1 \cdot L_2 \mid L_1 \in \mathcal{R}_1, L_2 \in \mathcal{R}_2\}$,
Kleene star	$\mathcal{R}^* = \{\mathcal{R} \cup \mathcal{R} \cdot \mathcal{R} \cup \dots \mid L \in \mathcal{R}\}$, and
union	$\mathcal{R}_1 \cup \mathcal{R}_2 = \{L \mid L \in \mathcal{R}_1 \vee L \in \mathcal{R}_2\}$.

These results follow immediately from the closure properties of regular languages. For example, if $\mathcal{R} = (K, \varphi)$, then $\mathcal{R}^*$ can be represented as (K^*, φ).

Rational sets of regular languages are not closed under the complement. If we consider the set of all languages over the same alphabet, then non-closure follows from the cardinality argument. If the complement is defined as $\neg\mathcal{R} = \mathrm{Reg}(\Sigma) \setminus \mathcal{R}$, then one should note that every rational set contains finitely many *prime languages*, while the complement contains infinitely many prime languages.

Closure under intersection is left open in [16]. It is not clear whether or not emptiness of intersection is decidable.

4 Automaticity of Some Classes of Semigroups of Regular Languages

A semigroup $\mathcal{S}$ may be defined in a number of ways. For example, a multiplication table, or an algorithm that efficiently computes the product of semigroup elements, may be given (this is exactly how we defined finitely generated semigroups of regular languages above). Alternatively, a set of *defining relations* may be given (we refer the reader to [17] for the general result on semigroup theory). Let us consider a semigroup $\mathcal{S}$ generated by languages

$$x = (a+b)^*a, \quad y = \varepsilon + a + b, \text{ and } z = b^*$$

over $\Sigma = \{a, b\}$ and construct a presentation for it. First of all, note that the language z is a star, so $z^k = z^p$ for all $k, p \geqslant 1$. Second, since x is the set of all words over Σ that end with the letter a and both y and z contain the empty word we have equations $(y+z)^k x = x$. Finally, the language $xz = (a+b)^*ab^*$ is the set of all words over Σ that contain at least one letter a, so we have a relations $x(yz)^k zy^p = xz$ for all $k, p \geqslant 0$. The semigroup $\mathcal{S}$ satisfies no other relations. Thus, the semigroup $\mathcal{S}$ has a presentation

$$\langle x, y, z \mid yx = x, zx = x, z^2 = z, xzy = xz, xy^k z = xz \ (k \geqslant 1)\rangle.$$

Checking equivalence of two words $u, v \in \{x, y, z\}^+$ of a semigroup defined by a regular language substitution leads to equivalence check of two NFAs. The complexity of the word problem for semigroup given by a presentation depends on words length. If the semigroup has a "nice" presentation then the equivalence may be checked more efficiently.

While it is well known that the word problem for finitely presented semigroups is undecidable in general, there exist classes of semigroups that have efficiently solvable word problem. Two examples are rational semigroups [20] and automatic semigroups [9]. In both cases the word problem can be solved by the means of finite automata. For rational semigroups the set of all equal words is recognized by a (two-tape) finite automaton. Automatic semigroups are finitely generated semigroups with rational cross-section (set of unique normal forms, i.e. $L \subseteq \Delta^+$ such that $\mathcal{S} = \varphi(L)$ and $\varphi(u) \neq \varphi(v)$ for all $u, v \in L$) such that multiplication by each generator is recognized by a finite automaton (the precise definition is presented in the next section). The set of such automata is known as automatic structure. Having such automatic presentation the word problem is solvable in linear time for rational semigroups and in quadratic time for automatic semigroups. Let us note that rational semigroups form a proper subclass of automatic semigroups [14].

There are two fundamental questions that should be answered:

- What is the structure of finitely generated semigroups of regular languages? Are they automatic? and
- If it is known that a semigroup defined by a finite set of regular languages is automatic, is it possible to construct a corresponding automatic structure *algorithmically*?

It is clear that semigroups of regular languages have some specific properties that are induces in some way by structure of automata corresponding to semigroup generators. In particular, in [2] the authors considered finiteness conditions for semigroups of regular languages. It was found that a finitely generated semigroup $\mathcal{S} = \langle V_1, \ldots, V_k\rangle$ is finite if and only if for every set of non-repeating indexes $\{i_1, \ldots, i_m\}$ $(m \leqslant k)$ there exists a natural number p such that $(V_{i_1} \cdot \ldots \cdot V_{i_m})^p = (V_{i_1} \cdot \ldots \cdot V_{i_m})^{p+1}$. In contrast, the semigroup given by presentation $S = \langle \Delta \mid x^2 = x^3 \text{ for all } x \in \Delta^+\rangle$ is infinite if $|\Delta| \geqslant 2$ [6]. It is also known that finitely generated commutative semigroups of regular languages over a one letter

alphabet are *rational* [3], but there exist commutative finitely presented semigroups that are not automatic [15].

In this section we prove that semigroups of factorial languages are automatic and provide an algorithm for building an automatic presentation in case of regular factorial languages.

4.1 Non-rationality of Semigroups of Regular Languages

Theorem 7. *Let $\varphi : \Delta^+ \to \mathrm{Reg}(\Sigma)$ be a regular language substitution. The set*

$$\mathrm{Ker}(\varphi) = \{(u, v) \mid u, v \in \Delta^+ \ \varphi(u) = \varphi(v)\}$$

is not an effectively constructable regular language.

Proof. We show that if the set $\mathrm{Ker}(\varphi)$ is regular then the equivalence problem for a rational set of regular languages, i.e. the problem to decide whether or not two given rational sets $\mathcal{R}_1 = (K_1, \varphi)$ and $\mathcal{R}_2 = (K_2, \varphi)$ are equal as sets of languages over Σ, is decidable.

For a given regular language $K \in \Delta^*$ by $\overline{K}$ let us denote the closure of K with respect to φ:

$$\overline{K} = \varphi^{-1}(\varphi(K)) = \{u \in \Delta^+ \mid \exists v \in K \ \varphi(u) = \varphi(v)\}.$$

The equality $\mathcal{R}_1 = \mathcal{R}_2$ holds if and only if $\overline{K_1} = \overline{K_2}$. Now, suppose that the set $\mathrm{Ker}(\varphi)$ is regular, i.e. there exists a finite automaton M that recognizes this language. By standard direct product construction of automata M and K we construct the automaton that recognizes the language $\{v \in \Delta^+ \mid \exists u \in K \ (u, v) \in \mathrm{Ker}(\varphi)\}$. Thus, if $\mathrm{Ker}(\varphi)$ is regular then so is $\overline{K}$ for every regular language $K \in \Delta^*$ and equality of rational set is decidable.

We have a contradiction with Theorem 4, so the set $\mathrm{Ker}(\varphi)$ is not regular in general. □

We can not state that $\mathrm{Ker}(\varphi)$ is not regular because it is possible that $\overline{K}$ is regular but not constructable.

4.2 Factorial Languages

A language is called *factorial* if it is closed under taking factors of its elements, i.e. $w \in L$ implies $v \in L$ for all v such that $w = \alpha v \beta$ where $\alpha, \beta \in \Sigma^*$. It is clear that factorial languages are closed under concatenation. A factorial language L is called *indecomposable* if $L = XY$ implies $L = X$ or $L = Y$ for all factorial languages. A decomposition of non-empty factorial language L to factorial languages $L = L_1 \dots L_k$ is called *canonical* if for all $i = 1, \dots, k$ languages L_i are non-empty and indecomposable, and for any factorial language $L_i' \subset L_i$ we have $L \neq L_1 \dots L_{i-1} L_i' L_i \dots L_k$. Canonical decomposition of $L = \{\varepsilon\}$ contains only one component $L_1 = \{\varepsilon\}$. The canonical decomposition of L is denoted as $\overline{L}$.

Canonical decompositions were studied in [5, 10, 4]. It was found that a canonical decomposition of each factorial language exists and unique, and all entries in the canonical decomposition of regular factorial language are also regular. The proof of uniqueness of decomposition is non-constructive if arbitrary factorial languages are considered. For regular languages canonical decompositions can be computed. In order to describe how canonical decompositions change under concatenation we need additional notation. For factorial language L let $\Pi(L) = \{a \in \Sigma \mid La \subseteq L\}$, and $\Lambda(L) = \{a \in \Sigma \mid aL \subseteq L\}$. For a subset $\Lambda \subseteq \Sigma$ and factorial language A define $L_\Lambda(A)$ as a factorial closure of the language $A \setminus \Lambda A$. Similarly, $R_\Lambda(A)$ denotes the factorial closure of $A \setminus A\Lambda$.

Theorem 8 ([10])**.** *Let A and B be factorial languages with $\overline{A} = A_1 \dots A_k$ and $\overline{B} = B_1 \dots B_m$. Denote $\Pi = \Pi(A)$ and $\Lambda = \Lambda(B)$. Then the canonical decomposition of the concatenation AB can be found as follows:*

1. *If* $\Lambda \setminus \Pi \neq \emptyset$ *and* $\Pi \setminus \Lambda \neq \emptyset$, *then* $\overline{AB} = \overline{A} \cdot \overline{B}$.
2. *If* $\Lambda = \Pi$ *and* $A_k \neq \Lambda^*$, $B_1 \neq \Lambda^*$, *then* $\overline{AB} = \overline{A} \cdot \overline{B}$.
3. *If* $\Lambda = \Pi$ *and* $A_k = \Lambda^*$, *then* $\overline{AB} = A_1 \dots A_{k-1}\overline{B}$. *Symmetrically, if* $\Lambda = \Pi$ *and* $B_1 = \Lambda^*$, *then* $\overline{AB} = \overline{A}B_2 \dots B_m$.
4. *If* $\Pi \subsetneq \Lambda$, *then* $\overline{AB} = \overline{R_\Lambda(A)} \cdot \overline{B}$. *Symmetrically, if* $\Lambda \subsetneq \Pi$, *then* $\overline{AB} = \overline{A} \cdot \overline{L_\Pi(B)}$.

In order to prove automaticity of semigroups of factorial languages we first choose appropriate set of generators, and then show that the set of canonical decompositions forms a regular language and multiplication can be performed by finite automata.

Let us denote by $\mathcal{F}(L)$ the set of languages in the canonical decomposition $\overline{L}$. For any set of factorial languages $\mathcal{L}$ by $\mathcal{F}(\mathcal{L})$ denote the set of all possible factors of canonical decompositions of $\mathcal{L}$ members, i.e. $\mathcal{F}(\mathcal{L}) = \bigcup\limits_{L \in \mathcal{L}} \mathcal{F}(L)$.

Lemma 9. *Let* $\mathcal{S}$ *be a finitely generated semigroup of factorial languages. The set* $\mathcal{F}(\mathcal{S})$ *is finite.*

Proof. According to Theorem 8 new languages only emerge in case 4. It is clear that for any factorial language A and $\Lambda_1, \Lambda_2 \subset \Lambda$ the equality $R_{\Lambda_1}(R_{\Lambda_2}(A)) = R_{\Lambda_2}(R_{\Lambda_1}(A)) = R_{\Lambda_1 \cup \Lambda_2}(A)$ holds, so the canonical decompositions of these languages coinside. Similarly, L_{Π_1} and L_{Π_2} do commute, as well as R_Λ and L_Π.

Let the semigroup $\mathcal{S}$ be generated by a regular language substitution $\varphi : \Delta^+ \to \mathrm{Reg}(\Sigma)$ and consider a word $w = p\delta s$ in $\Delta^*\delta\Delta^*$. Let δ occur in w only once. The language $\varphi(\delta)$ generates some elements into the canonical decomposition of $\overline{\varphi(w)}$ but these elements are completely defined by $\Pi(\varphi(p))$ and $\Lambda(\varphi(s))$, that are just some subsets of Σ. So, due to finiteness of the alphabet Σ, every semigroup generator yields only finitely many factors of canonical decompositions. □

The set $\mathcal{F}(\mathcal{S})$ will be the set of generators for automatic structure of a semigroup $\mathcal{S}$. By $F(\mathcal{S})$ let us denote a finite alphabet such that $|F| = |\mathcal{F}(\mathcal{S})|$.

Lemma 10. *Let* $\mathcal{S}$ *be a finitely generated semigroup of factorial languages. The set of canonical decompositions is a regular language over* $F(\mathcal{S})$.

Sketch. Let $\mathcal{S}$ be generated by a regular language substitution $\varphi : G^+ \to \mathrm{Reg}(\Sigma)$. For each generator $g \in G$ and all possible subsets $\Pi, \Lambda \subseteq \Sigma$ construct F-words corresponding to canonical decompositions $R_\Lambda(L_\Pi(\varphi(g)))$.

Given a generator $g \in G$ and two subsets $\Pi, \Lambda \subseteq \Sigma$ we can construct a regular expression that describes the language $N \subseteq F^*gF*$ such that substitution of $\overline{R_\Lambda(L_\Pi(\varphi(g)))}$ instead of g in any word from N gives a canonical decomposition. This is possible because the set of G-words satisfying $\Lambda(\varphi(w)) = \Lambda$ is regular (factorial languages contain the empty word, so $\Lambda(A_1 \dots A_k)$ equals to $\Lambda(A_i)$, where i is the maximal index such that $\Lambda(A_j) \subseteq \Lambda(A_i)$ for all $j < i$). Similarly, the set of words satisfying $\Pi(\varphi(w)) = \Pi$ is regular. Constructed G-words may be replaced with corresponding F-words because there are only finitely many possible subsets of Σ. □

We turn now to semigroup product operation. Let $A = A_1 \cdots A_k$. In [10] it was shown that the canonical decomposition of $A' = R_\Lambda(A)$ can be obtained by deleting $\{\varepsilon\}$ entries from the decomposition $A' = \overline{A'_1} \cdots \overline{A'_k}$, where languages A'_k are obtained by the following procedure. Starting from $\Lambda_k = \Lambda$ for all i from k to 1:

$$\begin{array}{rcll} A'_i & = & R_{\Lambda_i}(A_i) \text{ and } \Lambda_{i-1} = \Lambda(A'_i), & \text{if } A_i \not\subseteq \Lambda_i^*; \\ A'_i & = & \{\varepsilon\} \text{ and } \Lambda_{i-1} = \Lambda_i, & \text{otherwise.} \end{array}$$

A symmetric procedure is defined for decomposition $A' = L_\Pi(A)$.

Let us consider an automaton $\mathcal{A}$ recognizing the language L_g for some semigroup generator g (a factorial language that is not necessarily in $\mathcal{F}(\mathcal{S})$). This automaton synchronously reads pair of words (u, v) (each time it reads one letter of the first and second words) where both u and v are F-words representing canonical decompositions of some languages, and it should verify that v is the canonical decomposition of $\varphi(ug)$. This can clearly be done because the only thing that $\mathcal{A}$ should track if two words become different is the set of possible subsets of Σ that can produce such changes. Due to finiteness of Σ the semigroup multiplication can be verified by a finite automaton. Summing up, we can state the following.

Theorem 11. *Finitely generated semigroups of factorial languages are automatic.*

Let us note that the described construction relies on possibility of construction a canonical decomposition of factorial language. Although it is always exists, an algorithm for construction of such decomposition is only known for regular factorial languages (elements of decomposition are recognized by subautomata of an automaton recognizing the language).

5 Applications

Rational sets and finitely generated semigroups of regular or finite languages appear in such applications as database query processing, cryptography, and software testing frameworks. In this section we briefly identify these applications.

5.1 Semistructured Databases

Let us illustrate possible applications of the decision problems considered in this paper in the context of view based query processing in semistructured databases.

The problem of view based query processing plays an important role in many database applications, including information integration, query optimization, mobile computing and data warehousing. In its general form, the problem is stated as follows. Given a query over database schema and a set of materialized views over the same schema (i.e. a set of queries with precomputed answers – *view extensions*), is it possible to answer the newly arrived query using answers to the views? This question has been intensively studied for various data models and different assumptions on views semantic (e.g., [12, 11, 8, 21]). The main approaches to view based query processing are *query rewriting* and *query answering* (see [12] for a survey). In query rewriting approach, given a query Q and a set of views $\mathbf{V}$ one should construct a rewriting R such that $Q(D) = R \circ \mathbf{V}(D)$ for each database instance D. As $\mathbf{V}(D)$ are assumed known, the query rewriting R can be thought as an algorithm that describes how result of the query can be computed from the views.

In semistructured databases a query, in its simplest form, may be considered as a regular language over appropriate alphabet and query rewriting as a concatenation of given views. In these settings every set of views $\mathbf{V}$ defines a semigroup of queries that can be answered using that views. This is a semigroup of regular languages and a query can be answered if and only if its regular language belongs to the semigroup generated by $\mathbf{V}$. Further, one can define additional restrictions on the structure of admissible rewriting. For example, one could eliminate rewritings that use some sequence of views concatenations. If such restrictions are stated as a regular language, then the set of admissible rewritings is rational set of languages and query rewriting coincides with the membership problem for rational set of languages.

The above mentioned examples lead to the membership problem for rational sets of regular languages. Given two sets of views, say $\mathbf{V}_1$ and $\mathbf{V}_2$, can we compare these sets by their expressive power? For example, is it true that every query answered by $\mathbf{V}_1$ can be answered by $\mathbf{V}_2$? This is an inclusion problem.

Further, two rewritings are equivalent if they define the same language. Such query equivalence is clearly decidable because there exists an algorithm that can construct finite automata representing two concatenations and equivalence of finite automata is decidable. Nevertheless, the complexity of this straightforward algorithm grows exponentially with respect to both the number of states of automata representing members of $\mathbf{V}$ and the length of concatenations (this is because concatenation yields nondeterministic automata (NFA) and equivalence check of NFAs is P-SPACE complete). The question is: Given a set $\mathbf{V}$ of regular languages can we *precompute* some information that can speed up subsequent equivalence checks? This leads us to the notion of an automatic presentation of a semigroup of regular languages.

5.2 Cryptography

Algorithms for checking membership in a rational set have relatively high upper bounds. One of the key components is the limitedness property of distance automata, which is doubly exponential with respect to the number of states in the input automata. One can construct a public key cryptosystem on top of *language factorization*, more precisely, on *rational subset membership* problem for a finitely generated semigroup of regular languages. The ciphertext for a single bit of a message is a finite automaton, which represents zero or one depending on its rational subset membership. The membership problem may be considered as a generalization of the finite power property of a regular language and the only known solution is based on PSAPCE-complete problem of testing limitedness of distance automata.

An intruder should factorize the language using limitedness property of some distance automaton, while the intendant recipient could use additional information to speed up the processing, e.g. automatic presentation of finitely generated subsemigroup of a free semigroup.

5.3 Software Testing

In [16] the authors state that rational sets of regular languages, including the finite languages, form a theoretical basis for a query-driven program testing paradigm coverage specification language FQL. A program is represented as a directed graph, control-flow automaton, with edges marked by assignment operators, function calls, functions returns and some other operators. Alphabet Σ consists of these graph labels. Each language over Σ constitutes a test goal. During the analysis of the program one needs to construct test cases that cover all important properties of the program. This question corresponds to inclusion problems of rational sets, that was proved to be undecidable in case of finite languages.

6 Open Problems

As a possible direction of future research we consider the following.

- Emptiness of intersection for semigroups and rational sets of finite or arbitrary regular languages.

- Construct an example of semigroup such that $Ker(\varphi)$ is not regular.
- Computability of minimal upper approximation of a regular language by elements of a semigroup or a rational set of languages.
- Establish automaticity of semigroups of finite languages.

References

[1] Sergey Afonin. The view selection problem for regular path queries. In Eduardo Sany Laber and Claudson Bornstein, editors, *Proceedings of the LATIN 2008*, volume 4957 of *Lecture Notes in Computer Science*, pages 121–132. Springer, 2008.

[2] Sergey Afonin and Elena Khazova. Membership and finiteness problems for rational sets of regular languages. *International Journal of Foundations of Computer Science*, 17(3):493–506, 2006.

[3] Sergey Afonin and Elena Khazova. On the structure of finitely generated semigroups of unary regular languages. *Int. J. Found. Comput. Sci.*, 21(5):689–704, 2010.

[4] Sergey V. Avgustinovich and Anna E. Frid. A unique decomposition theorem for factorial languages. *IJAC*, 15(1):149–160, 2005.

[5] Sergey V. Avgustinovich and Anna E. Frid. Canonical decomposition of a regular factorial language. In *Computer Science Symposium in Russia, CSR 2006*, volume 3967 of *Lecture Notes in Computer Science*, pages 18–22. Springer, 2006.

[6] J.A. Brzozowski, K. Culik, and A. Gabrielian. Classification of noncounting events. *Journal of computer and system sciences*, 5:41–53, 1971.

[7] D. Calvanese, G. De Giacomo, M. Lenzerini, and M.Y. Vardi. Rewriting of regular expressions and regular path queries. *Journal of Computer and System Sciences*, 64:443–465, May 2002.

[8] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. Answering regular path queries using views. In *ICDE*, pages 389–398, 2000.

[9] Colin M. Campbell, Edmund F. Robertson, Nikola Ruškuc, and Richard M. Thomas. Automatic semigroups. *Theoretical Computer Science*, 250(1–2):365–391, January 2001.

[10] Anna E. Frid. Simple equations on binary factorial languages. *Theor. Comput. Sci.*, 410(30-32):2947–2956, 2009.

[11] Gosta Grahne and Alex Thomo. Query containment and rewriting using views for regular path queries under constraints. In *PODS '03: Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 111–122, New York, NY, USA, 2003. ACM Press.

[12] Alon Y. Halevy. Theory of answering queries using views. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 29(4):40–47, 2000.

[13] K. Hashiguchi. Representation theorems on regular languages. *Journal of computer and system sciences*, 27:101–115, 1983.

[14] Michael Hoffmann, Dietrich Kuske, Friedrich Otto, and Richard M. Thomas. Some relatives of automatic and hyperbolic groups. In *Semigroups, Algorithms, Automata and Languages*. World Scientific Pub Co Pte Lt, nov 2002.

[15] Michael Hoffmann and Richard M. Thomas. Automaticity and commutative semigroups. *Glasgow Mathematical Journal*, 44:167–176, 2002.

[16] Andreas Holzer, Christian Schallhart, Michael Tautschnig, and Helmut Veith. Closure properties and complexity of rational sets of regular languages. *Theoretical Computer Science*, 605:62 – 79, 2015.

[17] J.M. Howie. *Fundamentals of semigroup theory*. London Mathematical Society monographs. Clarendon, 1995.

[18] Karhumaki Juhani and Saarela Aleksi. The unique decipherability in the monoid of regular languages is undecidable. *Fundamenta Informaticae*, 110(1-4):197–200, 2011.

[19] Juhani Karhumäki and Leonid P. Lisovik. The equivalence problem of finite substitutions on ab*c, with applications. In *ICALP '02: Proceedings of the 29th International Colloquium on Automata, Languages and Programming*, pages 812–820, London, UK, 2002. Springer-Verlag.

[20] Jacques Sakarovitch. Easy multiplications I: The realm of Kleene's theorem. *Inf. Comput.*, 74(3):173–197, 1987.

[21] Luc Segoufin and Victor Vianu. Views and queries: determinacy and rewriting. In *PODS '05: Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 49–60, New York, NY, USA, 2005. ACM Press.

Some Space-Time Characteristics of Relational Structures*

Elena Bozhenkova[1,2] *Irina Virbitskaite*[1,2]

[1] A.P. Ershov Institute of Informatics Systems, SB RAS
6, Acad. Lavrentiev avenue, 630090, Novosibirsk, Russia

[2] Novosibirsk State University
2, Pirogov avenue, 630090, Novosibirsk, Russia

Abstract

Analysis techniques of concurrent and distributed systems that are grounded on the behavioural relations have become popular recently. Relational structures consisting of systems events and relations on them generalize many well-known behavioral models for concurrency and are helpful in dealing with topics like the specification of priorities, error recovery, treatment of simultaneity, etc. It is widely accepted that space-time concurrency axioms (including K-density, N-density, etc.) allow avoiding inconsistency between syntactic and semantic representations of concurrent processes and excluding unreasonable processes represented by the concurrent models. The intention of this paper is to generalize concurrency axioms and study their interrelations in the framework of the class of relational structures with distinct, irreflexive relations on countable sets of systems events.

1 Introduction

Causality (precedence), concurrency (independence), and conflict (mutual exclusion) are the most fundamental behavioral relations between actions in concurrent and distributed systems. To describe and analyse these and other relations, a variety of behavioral process models were put forward in concurrency theory, among them are causal nets, partial order sets, occurrence nets, event structures, relational structures, etc. It is worth stressing that analysis techniques of concurrent and distributed systems that are grounded on the behavioural relations have become popular recently [24].

Causal nets [21] model system behaviors by occurrences of conditions (local states) and by occurrences of events (actions) which are partially ordered. The partial order is interpreted as a kind of causal dependency relation. Also, causal nets are endowed with a symmetric, but in general non-transitive, concurrency relation — absence of the causality. Partial order sets (posets) [5] do not discriminate between conditions and events. Models with partial order based causality are useful in treating topics like fairness, confusion, etc.

Event structures, first defined in [28], consist of a set of events, together with a causality relation (presented as a partial order) and symmetric conflict relation, which satisfy the principles of finite causes and forward hereditary conflict, respectively. Two events that are neither in the causality relation nor in conflict relation are considered to be concurrent. As shown in [20], there is a close connection between this type of event structures and occurrence nets — causal nets extended by adding forward hereditary conflict. In the literature, several modifications and generalisations of the original definition of event structures can be found, often depending on the domain of application [7, 10, 19, 29]. In some of the modifications (e.g., in [6, 14, 22]), the conflict relations are not necessarily symmetric, allowing for, amongst other things, the description of weak causality.

*This work is supported in part by DFG (project CAVER, grant BE 1267/14-1)

It is known that some aspects of concurrent behavior (e.g., the specification of priorities, error recovery, treatment of simultaneity, etc.) are to some extent problematic to be dealt with using only partially ordered causality based models. One way to cope with the problems is to utilize the model of a so called relational structure — a set of elements (systems events) with a number of different kind relations on it. The authors of the papers [12, 13, 15, 16] have proposed and carefully studied a subclass of the model where general causal concurrent behavior is represented by a pair of relations instead just one, as in the standard partial order approach. Depending on the assumptions and goals for the chosen model of concurrency, the pair of the relations are interpreted in two versions: either as partially ordered causality and irreflexive weak causality (not in general a partial order) or as a symmetric and irreflexive mutex relation (non-simultaneity) and irreflexive weak causality (herewith, the relations may not be completely distinct). The approaches allow modeling and studying concurrency at different levels of consideration: from abstract behavioral observations such as step sequence executions to system level models such as elementary Petri nets and their generalizations with inhibitor arcs and mutex arcs.

We next draw the attention to the fact that in contrast to the standard physical theories which model systems by the continuum, Petri proposed concurrent structures (with causal nets as a special case thereof) as a combinatorial representation of a space-time. In the structures, notions corresponding to the relativistic concepts of world line and causal cone can be defined by means of concurrency and causal dependence relations, respectively. As a result, the density notion of the continuum model can be replaced by several properties — so called concurrency axioms (including K-density, N-density, etc.). K-density is based on the idea that at any time instant, any sequential subprocess of a concurrent structure must be in some state or changing its state. N-density can be viewed as a sort of local density. Furthermore, it has turned out that concurrency axioms allow avoiding inconsistency between syntactic and semantic representations of concurrent processes and excluding unreasonable processes represented by the concurrent structures. The power and limitations of concurrency axioms in the context of causal nets [3, 4] and posets [5, 8, 21] were widely studied to get a better understanding of the connections of causality and concurrency relations between systems events. In contrast to these treatments, the authors of the paper [18] dealt with causality and concurrency on cyclic processes represented by net models which do not require an underlying partial order of causality. In the setting of event structures, modifications of concurrency axioms and their interrelations were studied in [25, 26]. The paper [11] formulated conditions under which the coincidence between some generalizations of concurrency axioms and K-density occurs in the framework of occurrence nets. In more recent work (see [1, 2] among other), algebraic and orthomodular lattices were generated from occurrence nets with and without forward conflicts. Also, an alternative characterization of K-density is given on the basis of a relation between maximal sets of pairwise causally related elements and closed sets w.r.t. a closure operator, defined starting from the concurrency relation.

This paper is a component to the paper [27] where further generalizations of K-density were defined and sufficient conditions for their interrelations and characterizations in term of closed sets w.r.t. a closure operator, defined from an arbitrary symmetric relation, were formulated, in the context of relational structures. Here, we intend to extend classical results concerning the relationships between K- and N-density to their generalizations and modifications, in the framework of the class of relational structures with distinct, irreflexive relations on countable sets of elements (systems events).

2 Relational Structures

In this section, we define a model of relational structures and its properties. Before doing so, we need auxiliary notions and notations which will be useful throughout the text.

Given a set X and a relation $R \subseteq X \times X$,

- R is *cyclic* iff there exists a sequence of distinct elements $x_1, \ldots, x_k \in X$ $(k > 1)$ such that $x_j \ R \ x_{j+1}$ $(1 \leq j \leq k-1)$ and $x_k \ R \ x_1$,
- R is *acyclic* iff it is not cyclic,
- R is *asymmetric* iff $(x \ R \ x') \Rightarrow \neg(x' \ R \ x)$, for all $x, x' \in X$,
- R is *antisymmetric* iff $(x \ R \ x') \wedge (x \neq x') \Rightarrow \neg(x' \ R \ x)$, for all $x, x' \in X$,
- R is *symmetric* iff $(x \ R \ x') \iff (x' \ R \ x)$, for all $x, x' \in X$,
- R is *transitive* iff $(x \ R \ x') \wedge (x' \ R \ x'') \Rightarrow (x \ R \ x'')$, for all $x, x', x'' \in X$,
- R is *irreflexive* iff $\neg(x \ R \ x)$, for all $x \in X$,
- $R^\alpha = R \cup \mathrm{id}_X$ (the reflexive closure of R),
- $R^\beta = R \cup R^{-1}$ (the symmetric closure of R),
- $R^\gamma = R^\beta \cup \mathrm{id}_X$ (the reflexive and symmetric closure of R),
- $R^\delta = (R \setminus \mathrm{id}_X) \setminus (R \setminus \mathrm{id}_X)^2$ (the irreflexive, intransitive relation), if R is a transitive relation, and $R^\delta = R$, otherwise,

Notice that a relation is asymmetric iff it is both antisymmetric and irreflexive; a transitive relation is asymmetric iff it is irreflexive; if a relation R is irreflexive and transitive, then it is acyclic and antisymmetric, i.e. a (strict) partial order, and, moreover, R^δ is the immediate predecessor relation. Given elements $x_1, x_2 \in X$, and subsets $A \subseteq X' \subseteq X$, let $[x_1 \ R \ x_2] = \{x \in X \mid x_1 \ R^\alpha \ x \ R^\alpha \ x_2\}$, ${}^R A = \{x' \in X \mid \exists x \in A : (x' \ R^\alpha \ x)\}$, and A is a *(maximal) R-clique of X'* iff A is a (maximal) set containing only pairwise $R \cup \mathrm{id}_{X'}$-related elements of X'.

Definition 1. A *relational structure* is a tuple $S = (E, V_1, \ldots, V_n)$ $(n \geq 1)$, where

- E is a countable set of elements,
- $V_1, \ldots, V_n \subseteq E \times E$ are irreflexive relations such that
  - $\bigcup_{1 \leq i \leq n} V_i^\beta = (E \times E) \ \setminus \ \mathrm{id}_E$, where id_E is identity on E,
  - $V_i^\beta \cap V_j^\beta = \emptyset$, for all $1 \leq i \neq j \leq n$.

From now on, we shall use P, Q, and R to denote the unions of the form $\bigcup_{i \in \mathcal{V}} V_i$ and call them *connectives* of S. Here, $\mathcal{V} \subseteq \{i \mid 1 \leq i \leq n\}$.

Example 2. A simple example of a relational structure with four relations is shown in Fig. 1. Assume that V_1 is an irreflexive and transitive relation (a strict partial order), V_2 is an asymmetric relation, and V_3 and V_4 are irreflexive and symmetric relations. We can interpret the relation V_1 as causality dependence, V_2 as asymmetric conflict [22, 6], V_3 as synchronous concurrency (simultaneity), and V_4 as asynchronous concurrency (independence).

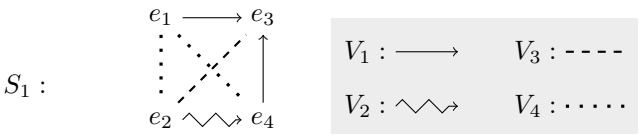

Figure 1:

Clearly, any poset (causal net) can be represented by a relational structure with two relations corresponding to causality (partial order) and concurrency (absence of partial order). Furthermore, any Winskel's prime event structure (occurrence net) with forward hereditary conflict [20] and Boudol and Castellani's flow event structure with forward and backward non-hereditary conflict [7] can be seen as a relational structure with three relations, one of them (causality) is transitive, and the other two (concurrency and conflict) are symmetric, whereas any Boldan et al's asymmetric event structure [6] with distinct causality and conflict — as a relational structure with an asymetric relation corresponding to asymmetric conflict.

Consider two auxiliary properties of relational structures which will be useful in further considerations. We shall call a relational structure S with its connectives P and Q:

- *P-transitive (P-irreflexive, P-symmetric*, respectively) iff P is transitive (irreflexive, symmetric, respectively);
- *P-finite* iff any P-clique of E is finite;
- *$\bigtriangledown_{PQ}$-free* iff there are no distinct elements e_1, e_2, and e_3 in E such that $e_1\ P^\beta\ e_2\ Q^\beta\ e_3\ R^\beta\ e_1$, where $R = \bigcup_{1\le i\le n} V_i \setminus (P \cup Q)$.

Lemma 3. *Given connectives $P \subseteq Q \subseteq R$ of a relational structure S and a maximal R-clique $\widetilde{E}$ of E, any maximal P-clique E' of $\widetilde{E}$ is a maximal P-clique of any maximal Q-clique E'' of $\widetilde{E}$, if $E' \subseteq E''$, and, moreover, at least one such the maximal Q-clique exists.*

Lemma 4. *Given distinct connectives $P \cup Q \subseteq R$ of a $\bigtriangledown_{PQ}$-free relational structure S and a maximal R-clique $\widetilde{E}$ of E, any maximal P-clique of any maximal $P \cup Q$-clique of $\widetilde{E}$ is a maximal P-clique of $\widetilde{E}$.*

3 Concurrency Axioms

In order to get a better understanding of the interaction of causality and concurrency, Petri [21] introduced the notion of K-density requiring non-empty intersections of maximal w.r.t. causality sets with maximal w.r.t. concurrency sets, in the context of causal nets. In [17], a direct analog of K-density under the name of L-density was proposed on acyclic nets with causality and symmetric hereditary conflict relations. Another analog under the name of R-density in the context of maximal w.r.t. concurrency and conflict substructures of event structures was dealt with in the paper [25]. In the framework of occurrence nets, a generalization of K-density called M-density in the paper [17] demand intersections of maximal w.r.t. causality and concurrency subnets with maximal w.r.t. causality and conflict subnets, to be maximal w.r.t. causality sets. Extensions of the concept of M-density to substructures being maximal w.r.t. different combinations of causality, concurrency, and conflict of event structures were put forward in the paper [26]. The author of [11] formulated

conditions under which K-density and B-density, requiring non-empty intersections of maximal w.r.t causality and conflict subnet with maximal w.r.t. concurrency sets, coincide, in the framework of occurrence nets. Our aim in this section is to give and study a single uniform definition generalizing K-density and its above-mentioned modifications, in the setting of relational structures.

Definition 5. Given a relational structure S and a maximal $(P \cup Q)$-clique $\widetilde{E}$ of E,

- $\widetilde{E}$ is K_{PQ}-*dense* iff for any maximal P-clique E' of $\widetilde{E}$ and for any maximal Q-clique E'' of $\widetilde{E}$, $E' \cap E''$ is a (unique) maximal $(P \cap Q)$-clique of $\widetilde{E}$,
- S is K_{PQ}-*dense* iff any maximal $(P \cup Q)$-clique $\widetilde{E}$ of E is K_{PQ}-dense.

Clearly, K-density (L-density, R-density, B-density, M-density, respectively) of a Winskel's prime event structure is K_{V_1,V_2}-density (K_{V_1,V_3}-density, K_{V_2,V_3}-density, $K_{V_1\cup V_3,V_2}$-density, $K_{V_1\cup V_2,V_1\cup V_3}$-density, respectively) of the corresponding relational structure with V_1 being partial order causality, V_2 — symmetric concurrency, and V_3 — symmetric conflict.

Theorem 6. *Given a relational structure S and its distinct connectives P, Q, and R,*

(i) S is K_{PQ}-dense, $\bigtriangledown_{PR}$- and $\bigtriangledown_{QR}$-free, and P- or Q-finite $\Longrightarrow$ S is $K_{\widetilde{P}\widetilde{Q}}$-dense,

(ii) S is K_{PQ}-dense $\Longleftarrow$ S is $K_{\widetilde{P}\widetilde{Q}}$-dense, $\bigtriangledown_{PQ}$-free, and P- or Q or R-finite,

where $\widetilde{P} = (P \cup R)$ and $\widetilde{Q} = (Q \cup R)$.

Proof. Notice that $E \neq \emptyset$, as the connectives under consideration are distinct.

(i) Suppose a contrary, i.e. S is not $K_{\widetilde{P}\widetilde{Q}}$-dense. Then, there exists a maximal $(P\cup Q\cup R)$-clique $\widetilde{E}$ of E with a maximal $(P \cup R)$-clique E_1 and a maximal $(Q \cup R)$-clique E_2 such that $A = E_1 \cap E_2$ is not a maximal $(P \cap Q)$-clique of $\widetilde{E}$. Let $B = E_1 \setminus A$ and $C = E_2 \setminus A$. Clearly, $B \cap C = \emptyset$. Due to the maximality of E_1 and E_2, it holds that $B \neq \emptyset$ and $C \neq \emptyset$. Consider two auxiliary

Lemma A. Given a maximal P-clique $B' \subseteq B$ of E_1 and a maximal Q-clique $C' \subseteq C$ of E_2, there is $\widetilde{b} \in B'$ and $\widetilde{c} \in C'$ such that $\widetilde{b}\ R^{\beta}\ \widetilde{c}$.

Lemma B. Given a P-clique $B' \subseteq B$ of E_1 and a Q-clique $C' \subseteq C$ of E_2 such that $A \cup \{e\}$ is an R-clique of $\widetilde{E}$ forall $e \in B' \cup C'$, it holds:

(a) $\exists b \in B'$ such that $C' \cup \{b\}$ is a Q-clique of $\widetilde{E} \Longrightarrow$
$\exists \widetilde{c} \in C \setminus C'$ such that $b\ P^{\beta}\widetilde{c}$, and, moreover, $A \cup \{\widetilde{c}\}$ is an R-clique of $\widetilde{E}$;

(b) $\exists c \in C'$ such that $B' \cup \{c\}$ is a P-clique of $\widetilde{E} \Longrightarrow$
$\exists \widetilde{b} \in B \setminus B'$ such that $c\ Q^{\beta}\widetilde{b}$, and, moreover, $A \cup \{\widetilde{b}\}$ is an R-clique of $\widetilde{E}$.

Suppose that A is a maximal R-clique of E_1 or E_2. By Lemma 4 and $\bigtriangledown_{PR}$- and $\bigtriangledown_{QR}$-freeness of S, A is a maximal R-clique of $\widetilde{E}$, contradicting our assumption. So, A is not a maximal R-clique of E_1 and E_2. Then, there is $b_1 \in B$ such that $A \cup \{b_1\}$ is an R-clique of E_1. As $\{b_1\}$ is a P-clique of E_1 and $\{b_1\} \cup \emptyset$ is a Q-clique of $\widetilde{E}$, there is $c_1 \in C \setminus \emptyset$ such that $b_1\ P^{\beta}\ c_1$ and $A \cup \{c_1\}$ is R-clique of E_2, by Lemma B(a). As $\{b_1\}$ is a P-clique of E_1 and $\{b_1\} \cup \{c_1\}$ is a P-clique of $\widetilde{E}$, then there is $b_2 \in B \setminus \{b_1\}$ such that $c_1\ Q^{\beta}\ b_2$ and $A \cup \{b_2\}$ is R-clique of E_1, by Lemma B(b). Again, as $\{c_1\}$ is a Q-clique of E_2 and $\{b_2\} \cup \{c_1\}$ is a Q-clique of $\widetilde{E}$, there is $c_2 \in C \setminus \{c_1\}$ such that $b_2\ P^{\beta}\ c_2$ and $A \cup \{c_1\}$ is R-clique of E_2, by Lemma B(a). Due to $\bigtriangledown_{PR}$-freeness and $\bigtriangledown_{QR}$-freeness of S, we get that $b_1\ P^{\beta}b_2$ and $c_1\ Q^{\beta}c_2$. Two cases are admissible.

1. $\exists b_i\ (i \leq 2) : c_2\ Q^\beta\ b_i$. A P-clique $\{b_1, b_2\}$ of E_1 can be extended to a maximal P-clique B' of E_1 ($B' \subseteq B$), and a Q-clique $\{c_1, c_2\}$ of E_2 can be extended to a maximal Q-clique C' of E_2 ($C' \subseteq C$). By Lemma A, there is $b_3 \in B'$ and $\widetilde{c_3} \in C'$ such that $b_3\ R^\beta\ \widetilde{c_3}$. Due to $\bigtriangledown_{PR}$-freeness of S, we get that $c_1\ Q^\beta\ b_3\ Q^\beta\ c_2$ and $A \cup \{b_3\}$ is R-clique of E_1. Moreover, by $\bigtriangledown_{PR}$-freeness of S, we get that $b_1\ P^\beta\ \widetilde{c_3}\ P^\beta\ b_2$ and $A \cup \{\widetilde{c_3}\}$ is R-clique of E_2.

   Since $\{b_3\} \cup \{c_1, c_2\}$ is a Q-clique of $\widetilde{E}$, there is $c_3 \in C \setminus \{c_1, c_2\}$ such that $b_3\ P^\beta\ c_3$ and $A \cup \{c_3\}$ is R-clique of E_2, due to Lemma B(a). By $\bigtriangledown_{PR}$-freeness of S, $c_1\ Q^\beta\ c_3\ Q^\beta\ c_2$, and, moreover, $\neg(c_2\ R^\beta\ b_1)$. Also, $\widetilde{c_3}\ R^\beta\ c_3$.

2. $\forall b_i\ (i \leq 2) : c_2\ P^\beta\ b_i$. Since $\{b_1, b_2\} \cup \{c_2\}$ is a P-clique of $\widetilde{E}$, there is $b_3 \in B \setminus \{b_1, b_2\}$ such that $c_2\ Q^\beta\ b_3$ and $A \cup \{b_3\}$ is R-clique of E_1, by Lemma B(b). Due to $\bigtriangledown_{PR}$-freeness of S, we get that $b_1\ P^\beta\ b_3\ P^\beta\ b_2$, and, moreover, $\neg(b_3\ R^\beta\ c_1)$. Two cases are admissible.

   (a) $\exists c_i\ (i < 3) : b_3\ P^\beta\ c_i$. A P-clique $\{b_1, b_2, b_3\}$ of E_1 can be extended to a maximal P-clique B' of E_1, and a Q-clique $\{c_1, c_2\}$ of E_2 can be extended to a maximal Q-clique C' of E_2. By Lemma A, there is $c_3 \in C'$ and $\widetilde{b_4} \in B'$ such that $c_3\ R^\beta\ \widetilde{b_4}$. Due to $\bigtriangledown_{PR}$-freeness of S, we get that $b_1\ P^\beta\ c_3\ P^\beta\ b_2$ and $A \cup \{c_3\}$ is R-clique of E_2, $A \cup \{\widetilde{b_4}\}$ is R-clique of E_1.

   (b) $\forall c_i\ (i < 3) : b_3\ Q^\beta\ c_i$. As $\{b_3\} \cup \{c_1, c_2\}$ is a Q-clique of $\widetilde{E}$, there is $c_3 \in C \setminus \{c_1, c_2\}$ such that $b_3\ P^\beta\ c_3$ and $A \cup \{c_3\}$ is R-clique of E_2, due to Lemma B(a). By $\bigtriangledown_{PR}$-freeness of S, we get that $c_1\ Q^\beta\ c_3\ Q^\beta\ c_2$.

By repeating infinitely many times the above reasonings, we get an infinite P-clique $b_1, b_2, \ldots$ and an infinite Q-clique $c_1, c_2, \ldots$ of $\widetilde{E}$, contradicting to either P- or Q-finiteness, respectively, of S.

(*ii*) Suppose a contrary, i.e. S is not K_{PQ}-dense. (Notice that $P \neq \emptyset \neq Q$.) Then, there exists a maximal $(P \cup Q)$-clique $\widehat{E}$ of E with a maximal P-clique B and a maximal Q-clique C such that $B \cap C$ is not a maximal $(P \cap Q)$-clique of $\widehat{E}$. Hence, $B \cap C = \emptyset$, because P and Q are distinct connectives. Consider an auxiliary

Lemma C. $\forall b \in B\ \exists c \in C : b\ P^\beta\ c$ and $\forall c \in C\ \exists b \in B : c\ Q^\beta\ b$.

By Lemma 3, there exists a maximal $(P \cup Q \cup R)$-clique $\widetilde{E}$ of E such that $\widehat{E}$ is a maximal $P \cup Q$-clique of $\widetilde{E}$. As S is $\bigtriangledown_{PQ}$-free, B is a maximal P-clique and C is a maximal Q-clique of $\widetilde{E}$, due to Lemma 4. According to Lemma 3, there exists a maximal $(P \cup R)$-clique E_1 of $\widetilde{E}$, such that B is a maximal P-clique of E_1, and a maximal $(Q \cup R)$-clique E_2 of $\widetilde{E}$, such that C is a maximal Q-clique of E_2. Notice that $E_1 \neq B$ and $E_2 \neq C$. As S is $K_{\widetilde{P}\widetilde{Q}}$-dense, $A = E_1 \cap E_2$ is a maximal R-clique of $\widetilde{E}$. By Lemma 3, A is a maximal R-clique of E_1 and E_2. Suppose that either $A \cap B \neq \emptyset$ or $A \cap C \neq \emptyset$. Due to Lemma C, this implies that either for $b \in A \cap B \subseteq E_2$ there exists $c \in C$ such that $b\ P^\beta\ c$, or for $c \in A \cap C \subseteq E_1$ there exists $b \in B$ such that $c\ Q^\beta\ b$, respectively, contradicting either to E_2 being a maximal $(Q \cup R)$-clique of $\widetilde{E}$ or to E_1 being a maximal $(P \cup R)$-clique of $\widetilde{E}$, respectively, because the connectives P, Q, and R are distinct. Therefore, $A \cap B = \emptyset$ and $A \cap C = \emptyset$.

As $E \neq \emptyset$ and B is a maximal P-clique of $\widetilde{E}$, $B \neq \emptyset$. Take $b_1 \in B$. Due to Lemma B, we get the following: for b_1 there exists $c_1 \in C$ such that $b_1\ P^\beta\ c_1$, because $B \cap C = \emptyset$; for c_1 there exists $a_1 \in A$ such that $c_1\ Q^\beta\ a_1$, because $A \cap C = \emptyset$; for a_1 there exists $b_2 \in B$ such that $a_1\ R^\beta\ b_2$, because $A \cap B = \emptyset$. By repeating infinitely many times the above reasonings, we get infinite sequences of elements $b_1, b_2, \ldots$ from B, $c_1, c_2, \ldots$ from C, and $a_1, a_2, \ldots$ from A such that $b_i\ P^\beta\ c_i\ Q^\beta\ a_i\ R^\beta\ b_{i+1}$ $(i \geq 1)$. As $\widetilde{E}$ is $\bigtriangledown_{PQ}$-free, we have: $b_i\ P^\beta\ a_j$, if $i \leq j$; $b_i\ R^\beta\ a_j$, if $i > j$; $b_i\ P^\beta\ c_j$, if $i \leq j$; $b_i\ Q^\beta\ c_j$, if $i > j$; $a_i\ R^\beta\ c_j$, if $i < j$;

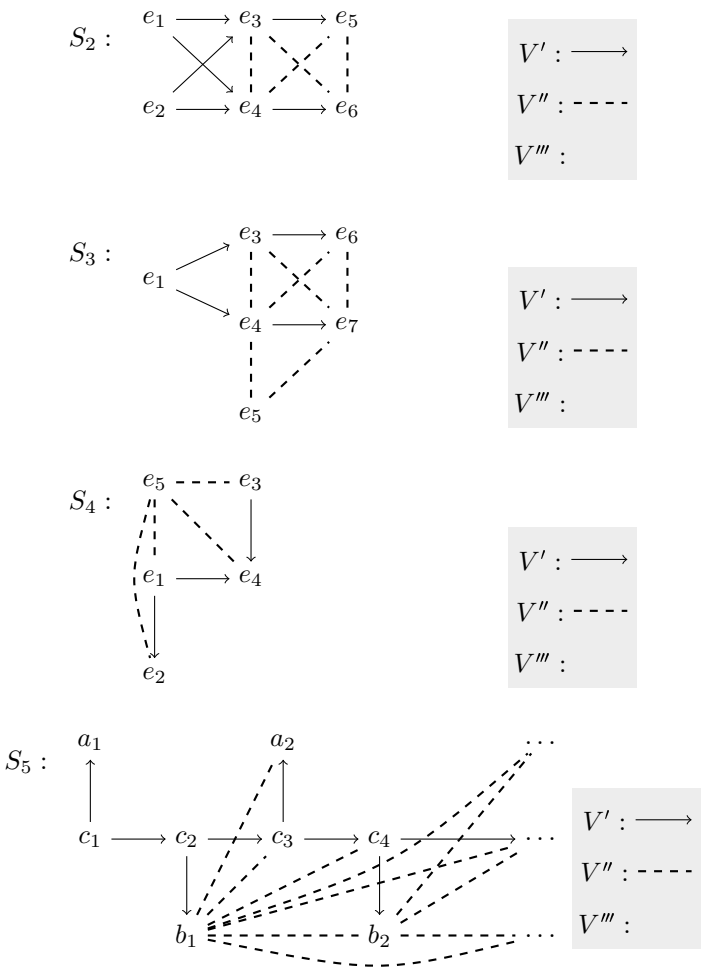

Figure 2:

$a_i\ Q^\beta\ c_j$, if $i \geq j$ $(i, j \geq 1)$. Then, $i \neq j \Rightarrow b_i \neq b_j$, $a_i \neq a_j$, $c_i \neq c_j$ $(i, j \geq 1)$, because P, Q, and R are distinct connectives. So, we get a contradiction to either P- or Q- or R-finiteness of S. □

Example 7. Consider the relational structures S_2–S_5, with the transitive relation V' and the symmetric relations V'' and V''', shown in Fig. 2. It is easy to check that $S_2 = (E_2, V', V'', V''')$ is $K_{V''V'''}$-dense, $\bigtriangledown_{V'V''}$-free, $K_{\widetilde{V}''\widetilde{V}'''}$-dense, and finite, where $\widetilde{V}'' = V' \cup V''$ and $\widetilde{V}''' = V' \cup V'''$. It is not difficult to verify that the relational structure $S_3 = (E_3, V', V'', V''')$ is $K_{V''V'''}$-dense and finite but neither $\bigtriangledown_{V'V'''}$-free, because $e_1\ V'\ e_4\ V''\ e_5\ V'''\ e_1$, nor $K_{\widetilde{V}''\widetilde{V}'''}$-dense, because in E_3 the intersection of the maximal $\widetilde{V}''$-clique $\{e_5, e_4, e_7\}$ with the maximal $\widetilde{V}'''$-clique $\{e_1, e_4, e_7\}$ is not a maximal V'-clique, where $\widetilde{V}'' = V' \cup V''$, $\widetilde{V}''' = V' \cup V'''$. The relational structure $S_4 = (E_4, V', V'', V''')$ is $\bigtriangledown_{V'V''}$-free and finite but neither $K_{V'V'''}$-dense, because in the maximal $(V' \cup V''')$-clique $\widetilde{E}' = \{e_1, e_2, e_3, e_4\}$ of E_4 the intersection of the maximal V'-clique $\{e_1, e_4\}$ with the maximal V'''-clique $\{e_2, e_3\}$ is empty, nor $K_{\widetilde{V}'\widetilde{V}'''}$-dense, because the intersection of

maximal $\widetilde{V}'$-clique $\{e_1, e_4, e_5\}$ with maximal $\widetilde{V}'''$-clique $\{e_2, e_3, e_5\}$ of E_4 is not a maximal V''-clique of E_4. It is not difficult to make sure that $S_5 = (E_5, V', V'', V''')$ is $K_{\widetilde{V}'\widetilde{V}''}$-dense but neither $\bigtriangledown_{V'V''}$-free nor finite, where $\widetilde{V}' = V' \cup V'''$ and $\widetilde{V}'' = V'' \cup V'''$. It is straightforward to demonstrate that the maximal $V'V''$-clique $\{b_1, b_2, b_3, \ldots, c_1, c_2, c_3, \ldots\}$ of E_5 is not $K_{V'V''}$-dense. Hence, S_5 is not $K_{V'V''}$-dense.

We now move to weaker versions of K-density, namely K-crossing and N-density first defined and studied in the context of causal nets and posets (see [3, 5, 21] among other). K-crossing is known as a property requiring non-empty intersections of maximal w.r.t. causality sets with causal predecessors and successors of the events in maximal w.r.t. concurrency sets. A poset is called N-dense iff every (four-element) N-shaped subposet can be extended to an K-dense subposet by adding element. We also need some kinds of discreteness and combinatorics notions making for establishing relationships between the above-mentioned concurrency axioms. Define these properties in the framework of relational structures.

Definition 8. Given a relational structure S and a maximal $(P \cup Q)$-clique $\widetilde{E}$ of E,

- $\widetilde{E}$ is K_{PQ}-*crossing* iff for any maximal P-clique E' of $\widetilde{E}$ and for any maximal Q-clique E'' of $\widetilde{E}$, $E' \cap {}^{P}E'' \neq \emptyset$ and $E' \cap E''^{P} \neq \emptyset$;
- $\widetilde{E}$ is N_{PQ}-*dense* iff for all distinct elements $e_0, e_1, e_2, e_3 \in \widetilde{E}$, if $(e_0\ P^\beta\ e_1\ Q^\beta\ e_2)$, $(e_0\ Q^\beta\ e_3\ P^\beta\ e_2)$, $(e_0\ P^\beta\ e_2)$, and $(e_3\ Q^\beta\ e_1)$ then $\exists e \in \widetilde{E}$ such that $\{e, e_0, e_2\}$ is a P-clique and $\{e, e_3, e_1\}$ is a Q-clique of $\widetilde{E}$;
- $\widetilde{E}$ is PQ-*combinatorial* iff $|\ [e_1\ P\ e_2] \cap E'\ | < \infty$, for some maximal P-clique E' of $\widetilde{E}$ and for all $e_1, e_2 \in \widetilde{E}$;
- $\widetilde{E}$ is PQ-*discrete* iff $|\ [e_1\ P\ e_2] \cap E'\ | < \infty$, for all maximal P-cliques E' of $\widetilde{E}$ and for all $e_1, e_2 \in \widetilde{E}$;
- S is K_{PQ}-*crossing* (N_{PQ}-*dense*, PQ-*combinatorial*, PQ-*discrete*, respectively) iff any maximal $(P \cup Q)$-clique $\widetilde{E}$ of E is K_{PQ}-crossing (N_{PQ}-dense, PQ-combinatorial, PQ-discrete, respectively).

Proposition 9. *Given a P-transitive, PQ-combinatorial, and N_{PQ}-dense relational structure S with distinct connectives P and Q,*

$$S \text{ is } K_{PQ}\text{-dense} \iff S \text{ is } PQ\text{-discrete and } K_{PQ}\text{-crossing.}$$

Proof. ($\Rightarrow$) Obviously, S is K_{PQ}-crossing. Take an arbitrary maximal $(P \cup Q)$-clique $\widetilde{E}$ of E. As P is transitive and irreflexive and, moreover, Q^γ is symmetric, $\widetilde{E}$ may be considered as a poset. The remaining reasonings follow the lines of those in the proofs of Proposition 4.1 and Theorem 4.5 of [23].

($\Leftarrow$) Suppose a contrary, i.e. S is not K_{PQ}-dense. Then, there exists a maximal $(P \cup Q)$-clique $\widetilde{E}$ of E with a maximal P-clique A and a maximal Q-clique B such that their intersection is not a maximal $(P \cap Q)$-clique of $\widetilde{E}$. Since P and Q are distinct, it holds that $A \cap B = \emptyset$. Due to K_{PQ}-crossing of $\widetilde{E}$, we get $A' = A \cap {}^{P}B \neq \emptyset$ and $A'' = A \cap B^{P} \neq \emptyset$. Notice that $A = A' \cup A''$, otherwise there is some $a \in A$ such that $a \notin {}^{P}B$ and $a \notin B^{P}$, and, hence, $a\ Q^\beta b$, for all $b \in B$, i.e. $a \in B$. From P-transitivity of S and distinctness of P and Q, it follows that $(a'\ P\ a'')$, for all $a' \in A'$ and $a'' \in A''$. By PQ-discreteness of S, there exists a maximal w.r.t. P element a' in A' and minimal w.r.t. P element a'' in A'', i.e. $a'\ P^\delta\ a''$. Clearly, we can find b' and b'' from B such that $a'\ P\ b'$ and $b''\ P\ a''$. Due

to P-transitivity of S and $b'\ Q^{\gamma}\ b''$, we get $\neg(a'\ P^{\beta}\ b'')$, $\neg(b'\ P^{\beta}\ a'')$, and $b' \neq b''$. Hence, $a'\ Q^{\beta}\ b''$ and $b'\ Q^{\beta}\ a''$, contradicting N_{PQ}-density of $\widetilde{E}$. □

Theorem 10. *Given a relational structure with distinct connectives P and Q,*

(i) S is K_{PQ}-dense $\Longrightarrow$ S is N_{PQ}-dense,

(ii) S is K_{PQ}-dense $\Longleftarrow$ S is N_{PQ}-dense, PQ-combinatorial, P-transitive, and Q-finite.

Proof. (i) Let $\widetilde{E}$ be an arbitrary maximal $(P \cup Q)$-clique of E. Take distinct elements e_0, e_1, e_2, e_3 of $\widetilde{E}$ such that $e_0\ P^{\beta}\ e_1\ Q^{\beta}\ e_2\ P^{\beta}\ e_3\ Q^{\beta}\ e_0\ P^{\beta}\ e_2$, and $e_3\ Q^{\beta}\ e_1$. In $\widetilde{E}$, extend the P-clique $\{e_0, e_2\}$ and Q-clique $\{e_1, e_3\}$ to a maximal P-clique A and maximal Q-clique B, respectively. Due to $\widetilde{E}$ being K_{PQ}-dense, these cliques are to intersect in some maximal $(P \cap Q)$-clique of $\widetilde{E}$, containing the only element e because P and Q are distinct connectives. Hence, $\{e_0, e_2, e\}$ is a P-clique of $\widetilde{E}$ and $\{e_1, e_3, e\}$ is a Q-clique of $\widetilde{E}$. So, $\widetilde{E}$ is N_{PQ}-dense.

(ii) Suppose a contrary, i.e. S is not K_{PQ}-dense. Then, there exists a maximal $(P \cup Q)$-clique $\widetilde{E}$ of E with a maximal P-clique B and maximal Q-clique C such that their intersection is not a maximal $(P \cap Q)$-clique of $\widetilde{E}$. Since P and Q are distinct, it holds that $B \cap C = \emptyset$. As S is P-transitive, P-irreflexive, and, moreover, Q^{γ}-symmetric, we can apply Theorem 3.7 of [23] to get that $\widetilde{E}$ is PQ-discrete. Then, either $B \cap^{P} C = \emptyset$ or $B \cap C^{P} = \emptyset$, according to Proposition 9. Since C is a maximal Q-clique of $\widetilde{E}$, it holds that $\widetilde{E} \subseteq {}^{P}C \cup C^{P}$. Then, we have either $B \subseteq (C^{P} \setminus C)$ or $B \subseteq ({}^{P}C \setminus C)$, respectively. As S is PQ-discrete and Q-finite, there exists either the minimal w.r.t. P element $b' \in B$ such that $c'\ P\ b'$ for some $c' \in C$ or maximal w.r.t. P element $b'' \in B$ such that $b''\ P\ c''$ for some $c'' \in C$, respectively. Since $B \subseteq (C^{P} \setminus C)$ or $B \subseteq ({}^{P}C \setminus C)$, respectively, either $b' \in B$ is the minimal w.r.t. P element of B or $b'' \in B$ is the maximal w.r.t. P element of B. Hence, by P-transitivity of S, for all $\widetilde{b} \in B$, it holds that either $c'\ P\ \widetilde{b}$ or $\widetilde{b}\ P\ c''$, respectively. We get a contradiction to the maximality of the P-clique B of $\widetilde{E}$. □

Example 11. Consider the relational structures S_6–S_{11} depicted in Fig. 3.

It is easy to check that $S_6 = (E_6, V', V'')$, with the transitive relation V', is finite, $K_{V'V''}$-dense (hence, $K_{V'V''}$-crossing), $N_{V'V''}$-dense, and $V'V''$-discrete (hence, $V'V''$-combinatorial).

In the relational structure $S_7 = (E_7, V', V'', V''')$, $V' = \{(x_i, x_j) \mid 0 \leq i < j\} \cup \{(x_i, y_j) \mid 0 \leq i, j\} \cup \{(x_i, z_j), (z_j, y_i) \mid 0 \leq i, 1 \leq j, i \leq j\}$ is a transitive relation and $V'' = \{(z_j, x_i), (z_j, y_i), (z_j, z_i), (x_i, z_j), (y_i, z_j), (z_i, z_j) \mid 1 \leq j, j < i\}$ and $V''' = \{(z_0, x_i), (z_0, y_i), (x_i, z_0), (y_i, z_0) \mid 0 \leq i\} \cup \{(z_0, z_i) \mid 0 < i\}$ are symmetric relations. It is not difficult to make sure that S_7 is $K_{V'V'''}$-dense (hence, $K_{V'V'''}$-crossing) but not $V'V'''$-combinatorial (hence, not $V'V'''$-discrete) because a maximal $(V' \cup V''')$-clique $\{x_i \mid 0 \leq i\} \cup \{y_i \mid 0 \leq i\} \cup \{z_0\}$ is not $V'V'''$-combinatorial (hence, not $V'V'''$-discrete).

It is straightforward to demonstrate that the relational structure $S_8 = (E_8, V', V'')$, with the transitive relation V' and symmetric relation V'', is $V'V''$-discrete (hence, $V'V''$-combinatorial) and $N_{V'V''}$-dense but not $K_{V'V''}$-crossing (hence, not $K_{V'V''}$-dense) because in the maximal $(V' \cup V'')$-clique $\widetilde{E} = \{e_1, e_2, \ldots\}$ of S_8 the intersection of its maximal V'-clique $E' = \{e_{2 \cdot k+1} \mid k \geq 0\}$ with its maximal V''-clique $E'' = E''^{V'} = \{e_{2 \cdot k} \mid k \geq 1\}$ is empty.

It is not difficult to verify that $S_9 = (E_9, V', V'', V''')$, with the transitive relation V' and symmetric relation V''', is finite, $K_{V'V'''}$-crossing, and $V'V'''$-discrete. However, S_9 is not $N_{V'V'''}$-dense because in the maximal $(V' \cup V''')$-clique $\widetilde{E} = \{e_2, e_3, e_4, e_5, e_7, e_8, e_9, e_{10}\}$ of E_9 there are distinct elements e_2, e_3, e_4, and e_5 such that $(e_2\ V'^{\beta}\ e_3$

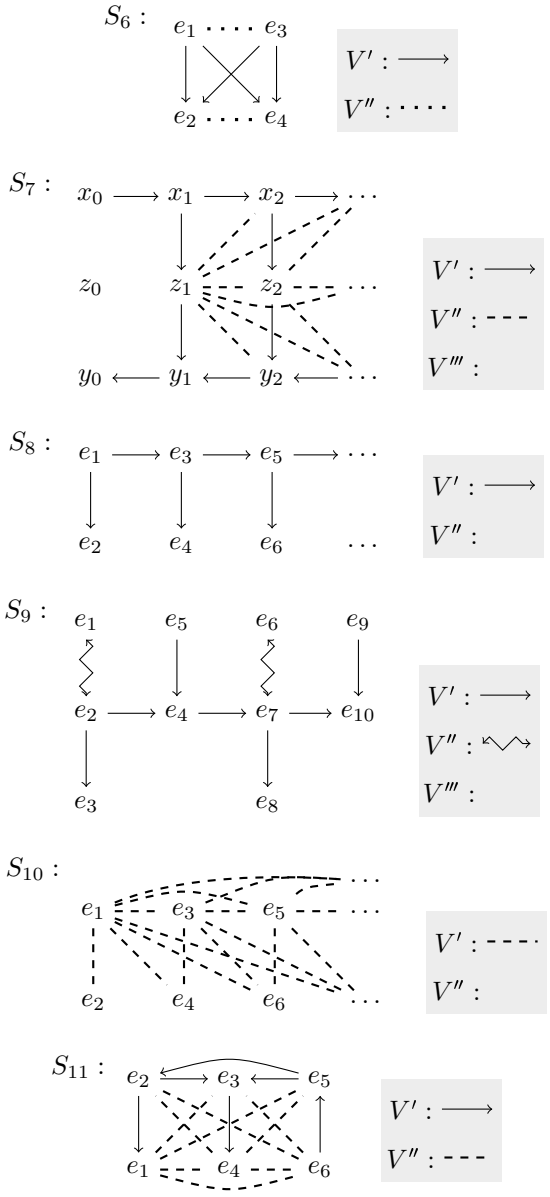
S_6 :
e_1 e_3
e_2 e_4
V' :
V'' :
S_7 :
x_0 x_1 x_2 $\cdots$
z_0 z_1 z_2 $\cdots$
y_0 y_1 y_2 $\cdots$
V' :
V'' :
V''' :
S_8 :
e_1 e_3 e_5 $\cdots$
e_2 e_4 e_6 $\cdots$
V' :
V'' :
S_9 :
e_1 e_5 e_6 e_9
e_2 e_4 e_7 e_{10}
e_3 e_8
V' :
V'' :
V''' :
S_{10} :
e_1 e_3 e_5 $\cdots$
e_2 e_4 e_6 $\cdots$
V' :
V'' :
S_{11} :
e_2 e_3 e_5
e_1 e_4 e_6
V' :
V'' :

Figure 3:

$V'''^{\beta}\ e_4$), ($e_2\ V'''^{\beta}\ e_5\ V'^{\beta}\ e_4$), ($e_2\ V'^{\beta}\ e_4$), and ($e_5\ V'''^{\beta}\ e_3$) but there is no $e \in \widetilde{E}$ such that $\{e_2, e_4, e\}$ is a V'-clique and $\{e_3, e_5, e\}$ is a V'''-clique of $\widetilde{E}$. Furthermore, S_9 is not $K_{V'V'''}$-dense because in the maximal $(V' \cup V''')$-clique $\widetilde{E}$ the intersection its maximal V'-clique $\{e_2, e_4, e_7, e_{10}\}$ with its maximal V'''-clique $\{e_3, e_5, e_9\}$ is empty.

Consider the relational structure $S_{10} = (E_{10}, V', V'')$, with the non-transitive relation V' and symmetric relation V''. It is easy to check that S_{10} is $N_{V'V''}$-dense and $K_{V'V''}$-crossing but not $K_{V'V''}$-dense because in the maximal $(V' \cup V'')$-clique $\widetilde{E} = \{e_1, e_2, \ldots\}$ of E_{10} the intersection of its maximal V'-clique $\{e_{2\cdot k+1} \mid k \geq 0\}$ with its maximal V''-clique $\{e_{2\cdot k} \mid k \geq 1\}$ is empty.

It is straightforward to show that the relational structure $S_{11} = (E_{11}, V', V'', V''')$, with the non-transitive and non-symmetric relation V' and symmetric relation V'', is finite, $N_{V'V''}$-dense, $K_{V'V''}$-crossing but not $K_{V'V''}$-dense because in E_{11} the intersection of its maximal V'-clique $\{e_2, e_3, e_5\}$ with its maximal V''-clique $\{e_1, e_4, e_6\}$ is empty.

References

[1] L. Bernardinello, C. Ferigato, S. Haar, L. Pomello. Closed sets in occurrence nets with conflicts. Fundamenta Informaticae 133(4) (2014) 323–344.

[2] L. Bernardinello, L. Pomello, S. Rombola. Orthomodular algebraic lattices related to combinatorial posets. CEUR-WS.OR 1231 (2014) 241–245.

[3] E. Best. The relative strenghth of K-density. Lecture Notes in Computer Science 84 (1980) 261–276.

[4] E. Best. A theorem on the characteristics of non-sequential processes. Fundamenta Informaticae 3 (1980) 77–94.

[5] E. Best, C. Fernandez. Nonsequential processes — a Petri net view. EATCS Monographs on Theoretical Computer Science 13 (1988).

[6] P. Baldan, A. Corradini, U. Montanari. Contextual Petri nets, asymmetric event structures, and processes. Information and Computation 171(1) (2001) 1–49.

[7] G. Boudol, I. Castellani. Concurrency and atomicity. Theoretical Computer Science 59 (1988) 25–84.

[8] C. Fernandez. Non-sequential processes. Lecture Notes in Computer Science 254 (1986) 95–116.

[9] C. Fernandez, P.S. Thiagarajan. A lattice theoretic view of K-density. Arbeitspapiere der GMD 76 (1983).

[10] R.J. van Glabbeek, G.D. Plotkin. Configuration structures, event structures and Petri nets. Theoretical Computer Science 410(41) (2009) 4111–4159.

[11] S. Haar. Occurrence Net Logics. Fundamenta Informaticae 43(1-4) (2000) 105–127.

[12] R. Janicki, M. Koutny. Invariants and paradigms of concurrency theory. Lecture Notes in Computer Science 506 (1991) 59–74.

[13] G. Juhas, R. Lorenz, S. Mauser. Synchronous + Concurrent = Earlier Than + Not Later Than. In: Proc. ACSD06, IEEE Press, New York (2006) 261–272.

[14] J.-P. Katoen. Quantitative and qualitative extensions of event structures. PhD Thesis, Twente University, 1996.

[15] J. Kleijn, M. Koutny. Process semantics of general inhibitor nets. Information and Computation 190 (2004) 18-69.

[16] J. Kleijn, M. Koutny. Mutex Causality in Processes and Traces of General Elementary Nets. Fundamenta Informaticae 122(1-2) (2013) 119–146.

[17] V. Kotov, L. Cherkasova. On structural properties of generalized processes. Lecture Notes in Computer Science 188 (1984) 288–306.

[18] O. Kummer, M.-O. Stehr. Petri's axioms of concurrency a selection of recent results. Lecture Notes in Computer Science 1248 (1997) 195–214.

[19] R. Langerak. Bundle event structures: a non-interleaving semantics for LOTOS. Formal Description Techniques V. IFIP Transactions C-10 (1993).

[20] M. Nielsen, G. Plotkin, G. Winskel. Petri nets, event structures and domains. Theoretical Computer Science 13 (1981) 85-108.

[21] C.A. Petri. Concurrency. Lecture Notes in Computer Science 84 (1980) 251–260.

[22] G. M. Pinna, A. Poigné. On the nature of events: another perspective in concurrency. Theoretical Computer Science 138 (1995) 425–454.

[23] H. Plünnecke. K-density, N-density and finiteness properties. Lecture Notes in Computer Science 188 (1984) 392–412.

[24] A. Polyvyanyy, M. Weidlich, R. Conforti, M. La Rosa, A.H.M. ter Hofstede. The 4C Spectrum of Fundamental Behavioral Relations for Concurrent Systems. Lecture Notes in Computer Science 8489 (2014) 210–232.

[25] I. Virbitskaite. Some characteristics of nondeterministic processes. Parallel Processing Letter 3(1) (1993) 99–106.

[26] E.N. Bozhenkova, I.B. Virbitskaite. Unified Characterization of Some Properties of Event Structures. In: Proc. Intern. Conf. CONPAR 94 – VAPP VI. Linz, Austria, September 1994. RISC-Linz Rep. Ser. 94-48 (1994) 29–32.

[27] I. Virbitskaite, E. Bozhenkova, E. Erofeev. Space-Time Viewpoints for Concurrent Processes Represented by Relational Structures. CEUR Workshop Proceedings 1492 (2015) 222–233.

[28] G. Winskel. Events in computation. PhD Thesis, University of Edinburgh, 1980.

[29] G. Winskel. Distributed probabilistic and quantum strategies. Electronic Notes in Theoretical Computer Science 298 (2013) 403–425.

On the dynamics of a configuration graph with random vertex degrees and bounded number of edges

Irina Cheplyukova *Yuri Pavlov*

Institute of Applied Mathematical Research, Karelian Research Centre RAS
e-mail: chia@krc.karelia.ru, e-mail: pavlov@krc.karelia.ru

Much attention has been paid to studying random graphs in connection with the wide use of these models for the description of the structure and dynamics of different complex networks, such as the Internet, social, transport and telecommunication networks (see e.g. [1], [2]). One of the most commonly used kinds of random graphs is the configuration model introduced in [3]. Observations of real networks showed ([1], [4]) that their topology could be adequately represented by configuration graphs with the vertex degrees being independently identically distributed random variables following the distribution:

$$\mathbf{P}\{\xi \geq k\} = h(k)k^{-\tau}, \quad k = 1, 2, \ldots, \tag{1}$$

where ξ is the degree of any vertex and $h(k)$ is a slowly varying function.

We consider a random graph consisting of N vertices and let random variables $\xi_1, \ldots, \xi_N$ be equal to the degrees of vertices with numbers $1, \ldots, N$. The vertex degree is the number of its semiedges, i.e. edges for which adjacent vertices are not yet determined. All of semiedges are numbered in an arbitrary order. The sum of vertex degrees in a graph has to be even, and if the sum $\xi_1 + \cdots + \xi_N$ is odd we add an auxiliary vertex with degree one. The graph is constructed by joing all the semiedges pairwise equiprobably to form edges. Because pairing is done without restrictions, multiple edges and loops can appear.

There are many papers where the results describing the limit behaviour of configuration graphs were obtained. The authors of [4] are sure that the function $h(k)$ in (1) does not influence limit results, and that to study the configuration graph one can replace $h(k)$ with the constant 1, then the random variable equal to the vertex degree has the following distribution:

$$p_k = \mathbf{P}\{\xi = k\} = k^{-\tau} - (k+1)^{-\tau}, \quad k = 1, 2, \ldots, \quad \tau > 0. \tag{2}$$

The analysis of conditional random graphs under the condition $\xi_1 + \cdots + \xi_N = n$ was for the first time carried out in [5]. Such conditional graphs can be useful for modeling of networks for which we can estimate the number of communications. They are useful also for studying networks without conditions on the number of links by averaging the results of conditional graphs with respect to the distribution of the sum of degrees. The technique of obtaining results in [5] is based on so called generalized allocation scheme proposed by V.F.Kolchin [6], [7].

In [8], an analogue of the generalized allocation scheme is suggested for solving probabilistic problems of combinatorial nature with a bounded number of elements. The paper [9] deals with conditional random graphs under the natural assumption $\xi_1 + \cdots + \xi_N \leq n$. The limit distributions of some numerical characteristics of such conditional configuration graphs were obtained there.

Recently, there appeared papers arguing that as the network size grows vertex degree distributions of corresponding graphs may became random. In this work we assume that the vertex degrees follow the distribution (2) under the condition that the parameter τ is

a random variable uniformly distributed on the interval $[a,b], 1 < a < b < \infty$. Then, using (2) we can show that random variables $\xi_1, \ldots, \xi_N$ are distributed according to the law

$$p_1 = \mathbf{P}\{\xi_i = 1\} = 1 - \frac{1}{(b-a)\ln 2}\left(\frac{1}{2^a} - \frac{1}{2^b}\right),$$

$$p_k = \mathbf{P}\{\xi_i = k\} = \frac{1}{(b-a)\ln k}\left(\frac{1}{k^a} - \frac{1}{k^b}\right) - \tag{3}$$
$$-\frac{1}{(b-a)\ln(k+1)}\left(\frac{1}{(k+1)^a} - \frac{1}{(k+1)^b}\right),$$

where $k = 2, 3, \ldots; \quad i = 1, 2, \ldots, N$. We consider the subset of random graph realizations with a bounded total sum of vertex degrees: $\xi_1 + \ldots + \xi_N \le n$. Denote by $\eta_1, \ldots, \eta_N$ the random variables equal to the degrees of vertices in such a conditional random graph. It is evident that these random variables are dependent, and for natural $k_1, \ldots, k_N$ such that $k_1 + \ldots + k_N \le n$

$$\mathbf{P}\{\eta_1 = k_1, \ldots, \eta_N = k_N\} = \mathbf{P}\{\xi_1 = k_1, \ldots, \xi_N = k_N | \xi_1 + \ldots + \xi_N \le n\}. \tag{4}$$

We denote by μ_r the number of vertices with degree r. The equation (4) means that for the random variables $\xi_1, \ldots, \xi_N$ and $\eta_1, \ldots, \eta_N$ the analogue of the generalized allocation scheme is valid and we can apply the known properties of this scheme to study conditional random graphs. This way we obtained the limit distributions of μ_r under various types of behaviour of the parameters N, n, r.

We introduce the following notations

$$m = \mathbf{E}\xi_1, \qquad \sigma^2 = \mathbf{D}\xi_1, \qquad H(x) = \sum_{k=2}^{\infty} \frac{1}{k^x \ln k}.$$

The next theorems are proved.

Theorem 1. *Let $N \to \infty$, r is a fixed positive integer and one of the following conditions is satisfied:*

1. $a > 2, (n - Nm)/(\sigma\sqrt{N}) \to \infty$;
2. $a = 2, (n - Nm)/\sqrt{N \ln\ln N} \ge -C > -\infty$;
3. $1 < a < 2, (n - N(b - a + H(a) - H(b))/(b-a))(N/\ln N)^{-1/a} \ge -C > -\infty$,

where C is a positive constant. Then

$$\mathbf{P}\{\mu_r = k\} = \frac{e^{-u^2/2}(1 + o(1))}{\sqrt{2\pi N p_r(1-p_r)}}$$

for nonnegative integers k uniformly with respect to $u = (k - Np_r)/\sqrt{Np_r(1-p_r)}$ lies in any fixed finite interval.

Theorem 2. *Let $N, r \to \infty, N(r^{a+1}\ln r)^{-1} \to \infty$ and one of the following conditions is satisfied:*

1. $a > 2, (n - Nm)/(\sigma\sqrt{N}) \ge -C > -\infty$;

2. $a = 2, (n - Nm)/\sqrt{N \ln \ln N} \geq -C > -\infty$;

3. $1 < a < 2, (n - N(b - a + H(a) - H(b))/(b - a))(N/\ln N)^{-1/a} \geq -C > -\infty$,

where C is a positive constant. Then

$$\mathbf{P}\{\mu_r = k\} = \frac{e^{-u^2/2}(1 + o(1))}{\sqrt{2\pi N p_r (1 - p_r)}}$$

for nonnegative integers k uniformly with respect to $u = (k - Np_r)/\sqrt{Np_r(1 - p_r)}$ lies in any fixed finite interval.

Theorem 3. *Let $N, r \to \infty$ and one of the following conditions is satisfied:*

1. $a > 2, (n - Nm)/(\sigma\sqrt{N}) \geq -C > -\infty$;

2. $a = 2, (n - Nm)/\sqrt{N \ln \ln N} \geq -C > -\infty$;

3. $1 < a < 2, (n - N(b - a + H(a) - H(b))/(b - a))(N/\ln N)^{-1/a} \geq -C > -\infty$,

where C is a positive constant. Then

$$\mathbf{P}\{\mu_r = k\} = \frac{(Np_r)^k}{k!} e^{-Np_r}(1 + o(1))$$

for nonnegative integers k uniformly with respect to $u = (k - Np_r)/\sqrt{Np_r}$ lies in any fixed finite interval.

The study was partially supported by the Russian Foundation for Basic Research, grant 16-01-00005.

References

[1] Faloutsos C., Faloutsos P., Faloutsos M. On power–law relationships of the Internet topology. *Computer Communications Rev.*. Vol. **29** (1999), 251-262.

[2] Hofstad R. Random graphs and complex networks. 2011. *Eindhoven university of technology*.

[3] Bollobas B. A probabilistic proof of an asymptotic formulae for the number of labelled regular graphs. *Eur.J.Comb.* **1** (1980), 311–316.

[4] Reittu H., Norros I. On the power-law random graph model of massive data networks. *Performance Evaluation*. Vol. **55** (2004), 3-23.

[5] Pavlov Yu.L., Cheplyukova I.A. Random graphs of Internet type and the generalized allocation scheme. *Discrete mathematics and applications*. Vol. **18**, issue 5, (2008), 447-464.

[6] Kolchin V.F. Random Graphs. 2010. 2nd ed., Cambridge Univ.Press, Cambridge.

[7] Kolchin V.F. *Random Mapping*. 1986. Springer, New York.

[8] Chuprunov A.N., Fazekas I. An analogue of the generalised allocation scheme: limit theorems for the number of cells containing a given number of particles. *Discrete mathematics and applications*. Vol. **22**, issue 1, (2012), 101-122.

[9] Pavlov Yu.L., Khvorostyanskaya E.V. On the limit distributions of the degrees of vertices in configuration graphs with bounded number of edges. Sbornic. *Mathematics*. Vol. **207**, issue 3, (2016), 400-417.

On small n-uniform hypergraphs with positive discrepancy

Danila Cherkashin[1] *Fedor Petrov*[2]

[1] Chebyshev Laboratory, St. Petersburg State University, 14th Line V.O., 29B, Saint Petersburg 199178 Russia; Moscow Institute of Physics and Technology, Lab of advanced combinatorics and network applications, Institutsky lane 9, Dolgoprudny, Moscow region, 141700, Russia;
St. Petersburg Department of V. A. Steklov Institute of Mathematics of the Russian Academy of Sciences.

[2] Saint Petersburg State University, Faculty of Mathematics and Mechanics; St. Petersburg Department of V. A. Steklov Institute of Mathematics of the Russian Academy of Sciences.

1 Introduction

A hypergraph is a pair (V, E), where V is a finite set whose elements are called vertices and E is a family of subsets of V, called edges. A hypergraph is n-uniform if every edge has size n. A vertex 2-coloring of a hypergraph (V, E) is a map $\pi : V \to \{1, 2\}$.

The *discrepancy* of a coloring is the maximum over all edges of the difference between the number of vertices of two colors in the edge. The *discrepancy* of a hypergraph is the minimum discrepancy of a coloring of this hypergraph. Let $f(n)$ be the minimal number of edges in an n-uniform hypergraph (all edges have size n) having positive discrepancy.

Obviously, if $2 \nmid n$ then $f(n) = 1$; if $2|n$ but $4 \nmid n$ then $f(n) = 3$. Erdős and Sős asked whether $f(n)$ is bounded or not. N. Alon, D. J. Kleitman, C. Pomerance, M. Saks and P. Seymour [1] proved the following Theorem, showing in particular that $f(n)$ is unbounded.

Theorem 1. *Let n be an integer such that $4 \mid n$. Then*

$$c_1 \frac{\log \operatorname{snd}(n/2)}{\log\log \operatorname{snd}(n/2)} \le f(n) \le c_2 \frac{\log^3 \operatorname{snd}(n/2)}{\log\log \operatorname{snd}(n/2)}, \tag{1}$$

where $\operatorname{snd}(x)$ *stands for the least positive integer that does not divide x.*

To prove the upper bound they introduced several quanties. Let $\mathcal{M}$ denote the set of all matrices M with entries in $\{0, 1\}$ such that the equation $Mx = e$ has exactly one nonnegative solution (here e stands for the vector with all entries equal to 1). This unique solution is denoted x^M. Let $d(M)$ be the least integer such that $d(M)x^M$ is integral and let $y^M = d(M)x^M$. For each positive integer n, let $t(n)$ be the least r such that there exists a matrix $M \in \mathcal{M}$ with r rows such that $d(M) = n$ (obviously, $t(n) \le n + 1$ because $d(I_{n+1}) = n$, where I_{n+1} is the $(n + 1) \times (n + 1)$ identity matrix). The upper bound in (1) follows from the inequality $f(n) \le t(m)$ for such m that $\left\lfloor \frac{n}{m} \right\rfloor$ is odd.

Then N. Alon and V. H. Vũ [2] showed that $t(m) \le (2 + o(1)) \frac{\log m}{\log\log m}$ for infinitely many m. However they marked that trueness of inequality $t(m) \le c \log m$ for arbitrary m is not clear.

Our main result is the following

Theorem 2. *Let n be a positive integer number. Then*

$$f(n) \le c \log \operatorname{snd}(n). \tag{2}$$

for some constant $c > 0$.

The main idea of the proof is to find a matrix with determinant $\operatorname{snd}(n)$ and small entries satisfying some additional technical properties.

2 Examples

Example 3. Let us consider the matrix $A = \begin{pmatrix} 3 & 5 \\ 1 & 8 \end{pmatrix}$ and suppose that n is not divisible on $\det A = 19$. Consider the system

$$Ax = \begin{pmatrix} 2n \\ 2n+t \end{pmatrix}. \tag{3}$$

It has the solution if and only if $19 \mid 4n + 3t$ i. e. t has prescribed residue modulo 19. Since n is not divisible on 19, t is not equal to zero modulo 19. So one can choose $-19 < t < 19$ such that t has prescribed residue modulo 19 and t is odd.

Let us construct an n-uniform hypergraph H with positive discrepancy. Let (a, b) be the solution of (3); note that a, b are positive and tend to ∞ with n. Consider disjoint vertex sets $A_1, \dots, A_3$ of size a and $B_1, \dots, B_8$ of size b. If $t < 0$, then consider a vertex set T of size $|t|$ disjoint with all sets A_i, B_j; if $t > 0$ let T be a t-vertex subset of B_1. Let C be a set $B_1 \cup T$ if $t > 0$, otherwise $C := B_1 \setminus T$. The edges of H are listed:

$$A_1 \cup A_2 \cup A_3 \cup B_1 \cup B_2 \cup B_3 \cup B_4 \cup B_5$$

$$A_1 \cup A_2 \cup A_3 \cup B_1 \cup B_2 \cup B_3 \cup B_4 \cup B_6$$

$$A_1 \cup A_2 \cup A_3 \cup B_1 \cup B_2 \cup B_3 \cup B_4 \cup B_7$$

$$A_1 \cup A_2 \cup A_3 \cup B_1 \cup B_2 \cup B_3 \cup B_4 \cup B_8$$

$$A_1 \cup A_2 \cup A_3 \cup B_2 \cup B_3 \cup B_4 \cup B_5 \cup B_8$$

$$A_1 \cup A_2 \cup A_3 \cup B_1 \cup B_3 \cup B_4 \cup B_5 \cup B_8$$

$$A_1 \cup A_2 \cup A_3 \cup B_1 \cup B_2 \cup B_4 \cup B_5 \cup B_8$$

$$A_1 \cup A_2 \cup A_3 \cup B_1 \cup B_2 \cup B_3 \cup B_5 \cup B_8$$

$$A_1 \cup C \cup B_2 \cup B_3 \cup B_4 \cup B_5 \cup B_6 \cup B_7 \cup B_8$$

$$A_2 \cup C \cup B_2 \cup B_3 \cup B_4 \cup B_5 \cup B_6 \cup B_7 \cup B_8$$

$$A_3 \cup C \cup B_2 \cup B_3 \cup B_4 \cup B_5 \cup B_6 \cup B_7 \cup B_8.$$

Obviously, if H has a coloring with discrepancy 0, then $d(B_5) = d(B_6)$, where $d(X)$ is the difference between blue and red verticed in X, because the second edge can be reached by replacing B_5 on B_6 in the first edge. Similarly one can deduce that $d(A_i) = d(A_j)$ and $d(B_i) = d(B_j)$ for all pairs i, j. So one can put $k := d(A_i)$, $l := d(B_i)$. Because of the first edge we have $3k + 5l = 0$. Obviously, k and l are odd numbers, so the minimal solution is $k = 5$, $l = -3$ (or $k = -5$, $l = 3$ which is the same because of red-blue symmetry). But then the last edge gives $|k + 8l| \le |t|$ which contradicts with $|k + 8l| \ge 19 > |t|$.

So we got an example if $19 \nmid n$ of an n-uniform hypergraph with 11 edges and positive discrepancy.

3 Proofs

Proof of Theorem 2. Let us denote $\mathrm{snd}(n)$ by q. We should construct a hypergraph with at most $c\log q$ edges and positive discrepancy. Take m such that $2^m-1\leqslant q\leqslant 2^{m+1}-2$. Then

$$q-(2^m-1)=\sum_{i=0}^{m-1}\varepsilon_i 2^i \quad \text{for some} \quad \varepsilon_i\in\{0,1\},$$

therefore

$$q=\sum_{i=0}^{m-1}\eta_i 2^i, \quad \text{where} \quad \eta_i=1+\varepsilon_i\in\{1,2\}.$$

Assume that q is odd. Consider m vectors in $\mathbb{Z}^m$:

$$v_0=(\eta_0,\dots,\eta_{m-1}),$$

$$v_i=(\eta_0,\dots,\eta_{i-2},\eta_{i-1}+2,\eta_i-1,\eta_{i+1},\dots,\eta_{m-1}) \quad \text{for} \quad i=1,\dots,m-1.$$

Note that the vector $u=(1,2,\dots,2^{m-1})$ satisfies a system of linear equations

$$\langle v_i,u\rangle=q; \quad i=0,\dots,m-1.$$

Choose odd $\delta\in(-q,q)$ such that $x_0:=\frac{n+\eta_{m-1}\delta}{q}$ is integer. Define

$$x_i:=2^i x_0 \quad \text{for} \quad i=1,\dots,m-2; \quad x_{m-1}:=2^{m-1}x_0-\delta,$$

then the vector $x=(x_0,\dots,x_{m-1})$ satisfies $\langle v_i,x\rangle=n$ for $i=1,\dots,m-2$, $\langle v_{m-1},x\rangle=n+\delta$.

In the case $q=2^m\geqslant 8$ we have $n\equiv 2^{m-1} \pmod q$ and $\eta_0=2,\eta_1=\dots=\eta_{m-1}=1$.

Choose $x=(x_0,\dots,x_{m-1})$ so that $\langle v_1,x\rangle=\langle v_{m-1},x\rangle=n+1$ and $\langle v_i,x\rangle=n$ for $i=0,2,3,\dots,m-2$. The solution is given by

$$x_0:=\frac{n+2^{m-1}}{q}; \quad x_1:=2x_0-1; \quad x_i:=2^{i-1}x_1 \quad \text{for} \quad i=2,\dots,m-2; \quad x_{m-1}:=2^{m-2}x_1-1.$$

Now let us construct a hypergraph in the following way: for $i=1,\dots m$ let us take 4 sets A_i^j $(j=1,\dots,4)$ of vertices of size x_i such that all the sets A_i^j are disjoint. Let the edge e_0 be the union of A_i^j over $1\leq i\leq m$ and $1\leq j\leq\eta_i$. By the choice of x_i and η_i we have $|e_0|=n$. Then we add an edge

$$\bigcup_{\substack{1\leq j\leq\eta_i \text{ for } i\neq k\\ j\in R \text{ for } i=k}} A_j^i$$

for every k and for every $R\subset[4]$ such that $|R|=\eta_k$. Clearly there are at most $6m$ such edges. Finally, for every $1\leq k\leq m-1$ we add the edge

$$\bigcup_{\substack{1\leq j\leq\eta_i \text{ for } i\neq k,k-1\\ 1\leq j\leq\eta_i+2 \text{ for } i=k-1\\ 1\leq j\leq\eta_i-1 \text{ for } i=k}} A_j^i.$$

Summing up we have hypergraph with at most $7m$ edges; at most 2 of them have size not equals to n. Let us correct this edges in the simpliest way: if an edge has size less

than n then we add arbitrary vertices; if an edge has size more than n then we exclude arbitrary vertices.

Suppose that our hypergraph has discrepancy 0, so it has a proper coloring π. For every set A_i^j denote by $d(A_i^j)$ the difference between the numbers of red and blue vertices of π in A_i^j. Obviously, $d(A_i^{j_1}) = d(A_i^{j_2})$ because there are edges e_1, e_2 such that $e_1 \Delta e_2 = A_i^{j_1} \cup A_i^{j_2}$. So we may write d_i instead of $d(A_i^j)$.

If q is odd then the vector $d = (d_0, \dots, d_{m-1})$ satisfies

$$\langle v_i, d\rangle = 0 \ \text{ for } \ i = 0, 1, \dots, m-2 \ \text{ and } \ \langle v_{m-1}, d\rangle = s$$

for some odd $s \in (-q, q)$. It implies that

$$d_i = 2^i d_0 \ \text{ for } \ i = 0, \dots, m-2; \ \ d_{m-1} = 2^{m-1} d_0 - s; \ \ 0 = \sum \eta_i d_i = d_0 q - \eta_{m-1} s,$$

but this fails modulo q. A contradiction. In the case $q = 2^m$ we get a similar contradiction, as $(2^{m-1} - 1) \pm 1$ is not divisible by 2^m.

Thus we get a hypergraph on at most $7m = O(\log q)$ edges with positive discrepancy, the claim is proven. □

References

[1] N. Alon, D. J. Kleitman, K. Pomerance, M. Saks, and P. Seymour. The smallest n-uniform hypergraph with positive discrepancy. *Combinatorica*, 7(2):151–160, 1987.

[2] Noga Alon and Văn H. Vũ. Anti-Hadamard matrices, coin weighing, threshold gates, and indecomposable hypergraphs. *Journal of Combinatorial Theory, Series A*, 79(1):133–160, 1997.

Notes about the linear complexity of cyclotomic sequences of order six and corresponding cyclic codes

Vladimir Edemskiy *Nikita Sokolovskiy* *Aleksandra Tsurina*

Novgorod State University, Veliky Novgorod, Russia

Using of cyclotomic classes to construct sequences, which are called cyclotomic sequences, is an important method for sequence design. The linear complexity LC of a sequence is an important parameter in its evaluation as a key stream cipher for cryptographic applications [1]. It may be defined as the length of the shortest linear feedback-shift register that can generate the sequence. The linear complexity of above-mentioned sequences over the finite field of order two was investigated in a lot of articles. Recently, series of papers have examined the linear complexity of cyclotomic sequences over the finite field $GF(q)$ where $q \geq 3$ is a prime number. In particular, the linear complexity of Legendre sequences over the finite field of any order was studied in [7]. Investigating the cyclic codes, C.Ding studied the linear complexity of the series of cyclotomic sequences of order four. Further, the linear complexity of Hall's sextic residue sequences over the finite field of odd prime order was investigated in [3, 4]. In this paper, we derive the linear complexity of a number of balanced cyclotomic sequences of order six over $GF(q)$. In conclusion we give a remark about cyclic codes which could be constructed using the cyclotomic classes of order six.

1 Preliminaries

First, we briefly repeat some of the basic definitions and general information. Let p be a prime of the form $p \equiv 1 \pmod 6$, and let g be a primitive root modulo p. Then the non-zero integers mod p can be partitioned into 6 cosets $H_i, 0 \leq i \leq 5$, each containing $(p-1)/6$ elements, such that H_0 contains the 6th power residues mod p, and the remaining H_i are formed from $g^i H_0$, where the arithmetic is that of $\mathbb{Z}_p$. Here and hereafter $a \bmod p$ denotes the least nonnegative integer that is congruent to a modulo p. Cosets H_i are also called the cyclotomic classes of order 6 with respect to p [1]. We have the following partition

$$\mathbb{Z}_p = H_0 \cup \cdots \cup H_5 \cup \{0\}.$$

Define a sequence $\{s_i\}$ with period p as

$$s_i = \begin{cases} 1, & \text{if } i \bmod p \in H_0 \cup H_1 \cup H_2, \\ 0, & \text{else.} \end{cases} \tag{1}$$

Here we regard it as a sequence over the finite field $GF(q)$ of order q where q is an odd prime, not equal to p. The field $GF(q)$ we identify with the set of integers $\{0, 1, \ldots, q-1\}$.

It is a familiar fact that if $\{s_i\}$ is a sequence of period p, then the linear complexity LC of this sequence is defined by

$$LC = p - \deg \gcd\big(x^p - 1, S(x)\big), \tag{2}$$

where $S(x) = s_0 + s_1 x + ... + s_{p-1} x^{p-1}$ [1].

Let α be a primitive pth root of unity in the extension of $GF(q)$. Then by Blahut's theorem

$$LC = p - |\{v \mid S(\alpha^v), v = 0, 1, \ldots, p-1\}|. \tag{3}$$

So, in order to find the linear complexity of $\{s_i\}$ it is sufficient to find the roots of $S(x)$ in the set $\{\alpha^v, v = 0, 1, \ldots, p-1\}$. To that end, let us introduce the subsidiary polynomials $S_6(x) = \sum_{i \in H_0} x^i$ and $S_3(x) = \sum_{i \in H_0 \cup H_3} x^i$ (here $S_3(x)$ is the polynomial of the characteristic sequence of cubic residues modulo p). The properties of the polynomial $S_6(x)$ were investigated in [3]. In particular, we have that $S_6(\alpha^v) = S_6(\alpha^{g^k})$ for all $v \in H_k, k = 0, 1, \ldots, 5$. So, from our definition it follows that

$$S(\alpha^v) = S_6(\alpha^{g^k}) + S_6(\alpha^{g^{k+1}}) + S_6(\alpha^{g^{k+2}}) \quad \text{for all } v \in H_k, k = 0, 1, \ldots, 5. \tag{4}$$

Hence, by (3), we obtain

$$LC = p - |\{k| \quad S(\alpha^{g^k}) = 0, \quad k = 0, \ldots, 5\}|(p-1)/6 - \Delta, \tag{5}$$

where

$$\Delta = \begin{cases} 1, & \text{if} \quad S(1) = 0, \\ 0, & \text{if} \quad S(1) \neq 0. \end{cases}$$

Besides, since $1 + \alpha + \ldots + \alpha^{p-1} = 0$, it follows that

$$S_3(\alpha) + S_3(\alpha^g) + S_3(\alpha^{g^2}) = -1 \text{ and } S(\alpha^{g^k}) + S(\alpha^{g^{k+3}}) = -1. \tag{6}$$

In conclusion of the section, we list some preliminary results required to prove our theorem. If $p \equiv 1 (\mathrm{mod}\ 3)$ then $4p$ can be expressed as $4p = L^2 + 27M^2; L \equiv 1 (\mathrm{mod}\ 3)$.

Lemma 1. ([6]) *Let $4p = L^2 + 27M^2; L \equiv 1 \pmod 3$. Then $S_3(\alpha), S_3(\alpha^\theta), S_3(\alpha^{\theta^2})$ are roots of the polynomial*

$$z^3 + z^2 - \frac{p-1}{3} z - \frac{3p + Lp - 1}{27}.$$

Denote by $(i, j)_6, i, j \in \mathbb{Z}$ cyclotomic numbers of order 6 [4]. The following statement was proved in [6] (see, also [3]).

Lemma 2. *Let $j, k = 0, \ldots, 5$. Then*

$$S_6(\alpha^{g^j}) S_6(\alpha^{g^k}) = \sum_{f=0}^{5} (k - j, f)_6 S_6(\alpha^{g^f}) + \delta,$$

where

$$\delta = \begin{cases} (p-1)/6, & \text{if } j = k \text{ and } p \equiv 1 \pmod{12} \text{ or } |j - k| = 3 \text{ and } p \equiv 7 \pmod{12}, \\ 0, & \text{otherwise.} \end{cases}$$

2 The linear complexity of a sequence

First of all, we find the equation for the values $S(\alpha^{g^k}), k = 0, \ldots, 5$. By (4) we see $S(\alpha^{g^k})^2 = \left(S_6(\alpha^{g^k}) + S_6(\alpha^{g^{k+1}}) + S_6(\alpha^{g^{k+2}})\right)^2$. Thus, we can use Lemma 2 in this case. The formulae for computing cyclotomic numbers of order six are well-known [4]. Let $\mathrm{ind}_g 2$ be a discreet logarithm to the base g of 2. Applying them and Lemma 2 we obtain the following statement.

Lemma 3. *Let $\{s_i\}$ be defined by (1) and $4p = L^2 + 27M^2; L \equiv 1 \pmod 3$. Then*

1. $S(\alpha^{g^k})^2 + S(\alpha^{g^k}) = M\left(S_3(\alpha^{g^{k+2}}) - S_3(\alpha^{g^k})\right) - (p+1)/4$ *for* $p \equiv 7 \pmod{12}$ *and* $S(\alpha^{g^k})^2 + S(\alpha^{g^k}) = M\left(S_3(\alpha^{g^k}) - S_3(\alpha^{g^{k+2}})\right) + (p-1)/4$ *for* $p \equiv 1 \pmod{12}$ *if* $ind_g 2 \equiv 0 \pmod 3$.
2. $S(\alpha^{g^k})^2 + S(\alpha^{g^k}) = M\left(S_3(\alpha^{g^{k+1}}) - S_3(\alpha^{g^k})\right) - (p+1)/4$ *for* $p \equiv 7 \pmod{12}$ *and* $S(\alpha^{g^k})^2 + S(\alpha^{g^k}) = M\left(S_3(\alpha^{g^k}) - S_3(\alpha^{g^{k+1}})\right) + (p-1)/4$ *for* $p \equiv 1 \pmod{12}$, *if* $ind_g 2 \equiv 1 \pmod 3$.
3. $S(\alpha^{g^k})^2 + S(\alpha^{g^k}) = M\left(S_3(\alpha^{g^{k+2}}) - S_3(\alpha^{g^{k+1}})\right) - (p+1)/4$ *for* $p \equiv 7 \pmod{12}$ *and* $S(\alpha^{g^k})^2 + S(\alpha^{g^k}) = M\left(S_3(\alpha^{g^{k+1}}) - S_3(\alpha^{g^{k+2}})\right) + (p-1)/4$ *for* $p \equiv 1 \pmod{12}$, *if* $ind_g 2 \equiv 2 \pmod 3$.

Before we give the main result of this section, we establish the following lemma for a particular case when $q = 3$.

Lemma 4. *Let* $\{s_i\}$ *be defined by* (1), $4p = L^2 + 27M^2; L \equiv 1 \pmod 3$, *and let* $q = 3$. *Then the linear complexity of* $\{s_i\}$ *equals*

$$LC = \begin{cases} (p-1)/2, & \text{if } p \equiv 1 \pmod{12} \text{ and } M \equiv 0 \pmod 3, \\ p-1, & \text{else.} \end{cases}$$

Proof. First of all, we note that $S(1) = 0$ here.

Let $p \equiv 1 \pmod{12}$. Suppose that there exists $j : 1 \le j < p$ such that $S(\alpha^j) = 0$; then $j \in H_k$ for $0 \le k \le 5$, and by Lemma 3 we see that $M = 0$ over $GF(3)$ or $S_3(\alpha^{g^k}) = S_3(\alpha^{g^f})$ where $f = k+1$ or $f = k+2$. By Lemma 1 the latest equality is impossible for $M \neq 0$ and $q = 3$.

Let $p \equiv 1 \pmod{12}$ and $M \equiv 0 \pmod 3$. Then by Lemma 3 we have that $S(\alpha^{g^k})^2 + S(\alpha^{g^k}) = 0$ for $k = 0, \dots, 5$. The conclusion of Lemma 4 in this case then follows from (5) and (6).

The statement of this lemma for $p \equiv 7 \pmod{12}$ can be proved similarly. □

Theorem 5. *Let* $\{s_i\}$ *be defined by* (1), $4p = L^2 + 27M^2; L \equiv 1 \pmod 3$. *Then:*

(i) $LC = (p-1)/2$ *if* $p \equiv 1 \pmod q$, $p \equiv 1 \pmod{12}$ *and* $M \equiv 0 \pmod q$.

(ii) $LC = (p+1)/2$ *if* $p \equiv -1 \pmod q$, $p \equiv 7 \pmod{12}$ *and* $M \equiv 0 \pmod q$.

(iii) $LC = 2(p+1)/3$ *if* $L \equiv 0 \pmod q$ *and* $p \equiv -9 \pmod q$, $p \equiv 7 \pmod{12}$ *or* $p \equiv 9 \pmod q$, $p \equiv 1 \pmod{12}$.

(iv) $L \ge 5(p-1)/6$ *in other cases.*

Proof. The statement of this theorem for $q = 3$ follows from Lemma 4. Like that, we assume that $q > 3$.

We consider only the first case. Suppose $LC = (p-1)/2$; then by (5) we see $S(1) = 0$ and the polynomial $S(x)$ have three roots in the set $\{\alpha^{g^k}, k = 0, \dots, 5\}$. Since $S(1) = 0$, it follows that $p \equiv 1 \pmod q$. Then by Lemma 3 and (6) we obtain that $M \equiv 0 \pmod q$ or $S_3(\alpha) = S_3(\alpha^g) = S_3(\alpha^{g^2}) = -1/3$. By Lemma 1 the latest equality is impossible for $M \not\equiv 0 \pmod q$.

Let $p \equiv 1 \pmod{12}$ and $M \equiv 0 \pmod q$. Then by Lemma 3 we have that $S(\alpha^{g^k})^2 + S(\alpha^{g^k}) = 0$ for $k = 0, \dots, 5$. The conclusion of this theorem in the this case then follows from (5) and (6).

The other statements of Theorem 5 can be proved similarly, but in a slightly more complicated way. □

The results of direct computing of the linear complexity by Berlekamp-Massey algorithm for $3 \le q \le 23, 5 \le p \le 20000$ confirm the results of Theorem 5.

3 Notes about the cyclic codes

An $[n, k]$ linear code C over $GF(q)$ is a linear subspace of $GF(q)^n$ with dimension k. A code C is called cyclic if $c = (c_0, c_1, \dots, c_{n-1}) \in C$ implies $(c_{n-1}, c_0, c_1, \dots, c_{n-2}) \in C$. As a subclass of linear codes, cyclic codes have applications in consumer electronics, data storage systems, and communication systems as they have efficient encoding and decoding algorithms [2].

We may identify each vector $(c_0, c_1, \dots, c_{n-1}) \in GF(q)^n$ with $c_0+c_1x+\dots+c_{n-2}x^{n-2}+c_{n-1}x^{n-1} \in GF(q)[x]/(x^n-1)$. It is a familiar fact that for every cyclic code C of length n over $GF(q)$, there is a unique monic polynomial $g(x) \in GF(q)[x]$ of the smallest degree such that $C = (g(x))$, where $(g(x))$ is an principal ideal of the ring $GF(q)[x]/(x^n-1)$. This polynomial $g(x)$ is called the generator polynomial of C.

The cyclotomic classes can be also used to construct cyclic codes [2]. Let I be any nonempty subset of $\{0, 1, \dots, 5\}$, $D = \bigcup_{k\in I} H_k$ and $D(x) = \sum_{i\in D} x^i$. The cyclic code over $GF(q)$ with generator polynomial $g(x) = \gcd(x^p - 1, D(x))$ is called the cyclic code of the set D, where the arithmetic is that of $GF(q)$ and is denoted by $C_{GF(q)}(D)$ [2].

It is well-known that $k = n - \deg g(x)$. So, by (2) we have that $k = LC$, where LC is the linear complexity of the characteristic sequence $h(D)$ of D. Note that the correspondence between $h(D)$ and $D(x)$ is one-to-one. Thus, Theorem 5 defines parameters of cyclic codes $[p, k]$ for $D = H_0 \cup H_1 \cup H_2$.

This method can be useful for to derive the linear complexity of other cyclotomic sequences or to determine a dimension of other cyclic codes. For example, if $p = A^2 + 3B^2$ and $D = H_0 \cup H_1$ then the cyclic code $C_{GF(2)}(D)$ has parameters $[p, k]$ where

$$k = \begin{cases} (p-1)/3, & \text{if } B \equiv 0 \pmod{12}, \\ 2(p-1)/3, & \text{if } B \equiv 6 \pmod{12}, \\ p-1, & \text{if } B \not\equiv 0 \pmod{6}. \end{cases}$$

References

[1] Cusick T., Ding C., Renvall A. Stream Ciphers and Number Theory, North-Holland Publishing Co., Amsterdam, 1998.

[2] Ding C. Codes from Difference Sets, World Scientific, 2015.

[3] Edemskiy V., Sokolovskiy N. On the linear complexity of Hall's sextic residue sequences over $GF(q)$. J. Appl. Math. Comput. 1, 1-9 (2016). DOI 10.1007/s12190-016-1010-2

[4] Hall M. Combinatorial Theory, Wiley, New York, 1975

[5] He X., Hu L., Li D. On the $GF(p)$ Linear Complexity of Hall's Sextic Sequences and Some Cyclotomic-Set-Based Sequences. Chin. Ann. Math. 37B(4), 515-522 (2016).

[6] Myerson G. Period polynomials and Gauss sums for finite fields, Acta Arith., 39, 251—264 (1981).

[7] Wang Q., Lin D., Guang X. On the Linear Complexity of Legendre Sequences Over F_q, IEICE Trans. Fundamentals, E97-A (7), 1627- 1630 (2014)

Around the Road Coloring Theorem

Vladimir V. Gusev[1,2] *Elena V. Pribavkina*[1] *Marek Szykuła*[3]

[1] Ural Federal University, Ekaterinburg, Russia, elena.pribavkina@gmail.com
[2] Université catholique de Louvain, Louvain-la-Neuve, Belgium, vl.gusev@gmail.com
[3] University of Wrocław, Wrocław, Poland, msz@cs.uni.wroc.pl

Abstract

An automaton is synchronizing if there exists a word that sends all states of the automaton to a single state. A coloring of a digraph with a fixed out-degree k is a distribution of k labels over the edges resulting in a deterministic finite automaton. The famous road coloring theorem states that every primitive digraph has a synchronizing coloring. We give an overview of recent results and conjectures related to this theorem.

Introduction Let $\mathscr{A} = (Q, \Sigma, \delta)$ be a finite deterministic complete automaton with an alphabet Σ, a set of states Q and a transition function δ. The automaton $\mathscr{A}$ is *synchronizing* if there exist a word u and a state p such that for every state $q \in Q$ we have $q \cdot u = p$, where $q \cdot u$ denotes the image of q under the action of u. Any such word u is called *synchronizing* (or *reset*) word for $\mathscr{A}$. The length of the shortest synchronizing word $\mathrm{rt}(\mathscr{A})$ is called the *reset threshold* of $\mathscr{A}$. Synchronizing automata naturally appear in algebra, coding theory, industrial automation, discrete dynamical systems, etc. A brief survey of the theory of synchronizing automata may be found in [16].

Two fundamental problems about synchronizing automata that were intensively investigated in the last decades are the Černý conjecture and the road coloring problem. The former states that the reset threshold of an n-state automaton is at most $(n-1)^2$ [6]. Despite intensive research efforts it remains open for already half a century. The latter problem states a certain connection between primitive digraphs and synchronizing automata, which we will explain shortly, and was resolved by Trahtman [15] after crucial insight by Culik, Karhumäki, and Kari [7]. We will describe recent results concerning potential generalizations of the road coloring theorem and related problems.

The road coloring theorem The *underlying digraph* $\mathcal{G}(\mathscr{A})$ of an automaton $\mathscr{A}$ is a digraph with Q as the set of vertices, and for each $u \in Q$, $x \in \Sigma$ there is an edge $(u, u \cdot x)$. We allow loops and multiple edges, thus $\mathcal{G}(\mathscr{A})$ has a fixed out-degree equal to the cardinality of the alphabet Σ, i.e., $\mathcal{G}(\mathscr{A})$ is a $|\Sigma|$-out-regular digraph.

Vice versa, given a digraph G with a fixed out-degree k and a finite alphabet Σ with k letters, we can obtain a deterministic finite automaton by distributing the letters of Σ over the edges of G. Any automaton obtained in this way is called a *coloring* of G. A digraph is *primitive* if there exists a number t such that for any two vertices u and v there exists a path from u to v of length exactly t. An automaton is *strongly connected* if its underlying digraph is strongly connected.

Theorem 1 (Road Coloring Theorem [15])**.** *A strongly connected digraph G with a fixed out-degree k has a synchronizing coloring if and only if it is primitive.*

The origin of the terminology is as follows. A digraph G represents a network of one-way roads. A coloring of G defines labels of the roads that can be perceived by drivers. If the coloring is synchronizing then the drivers who are unaware of their current location have the following strategy to relocate themselves: they can simply follow roads labeled by a synchronizing word and their final position will be well defined.

There exists an algorithm working in $O(kn^2)$ time that finds a synchronizing coloring of an aperiodic digraph [2].

Although the road coloring theorem gives an answer for a principal connection between digraphs and synchronizing automata, there are still many basic quantitative questions that remain unanswered.

How many colorings are synchronizing? The *synchronizing ratio* of a k-out-regular digraph G is the number of synchronizing colorings divided by the total number of colorings. Note that a coloring is a mapping from the set of edges to Σ with parallel edges being distinguished. Thus, the total number of colorings of a k-out-regular digraph with n states is always $(k!)^n$. The road coloring theorem can be restated as follows: the synchronizing ratio of a digraph G is greater than zero if and only if G is primitive. In order to derive more precise bounds depending on n, k, a series of computational experiments was performed in [9]. Based on the results obtained for small values of n, k the following conjecture was formulated:

Conjecture 2 ([9]). *The minimum value of the synchronizing ratio among all k-out-regular primitive digraphs with n vertices is equal to $\frac{k-1}{k}$, except for the case $k = 2$ and $n = 6$ when it is equal to $\frac{30}{64}$.*

For $k = 2$, the conjecture has been verified for $n \geq 10$ vertices. Surprisingly, but the conjectured lower bound does not depend on n. It was also shown that it can be achieved:

Theorem 3 ([9]). *There is a 2-out-regular digraph with 6 states and the synchronizing ratio $\frac{30}{64}$. Moreover, for every $n > 3$ and $k \geq 2$ there is a k-out-regular digraph with n vertices and the synchronizing ratio $\frac{k-1}{k}$.*

This result can easily be extended to the cases $n = 2$ and $n = 3$.

For the moment, the conjecture is widely open. In a special class of digraphs a non-trivial lower bound on the synchronizing ratio was obtained in [8] using linear algebraic approach. We will require a few definitions to state the main results. We will denote the adjacency matrix of a primitive digraph G by $\mathcal{A}(G)$. Perron-Frobenius theorem [10, Chapter 8] implies existence of entrywise positive eigenvector $\vec{v}$ of $\mathcal{A}(G)$ associated with the unique largest eigenvalue k, which we will simply call *the eigenvector* of G. The vector $\vec{v}$ can also be seen as the unique stationary distribution of the Markov chain associated with G by assigning the probability $\frac{1}{k}$ for each of the outgoing edges. Let $Q = \{1, \ldots, n\}$ be the set of vertices of G. We will say that the vector $\vec{v}$ is *partitionable* if there exists a partition of $\vec{v}$ into blocks of equal weight b, i.e., a partition $Q_1, \ldots, Q_\ell$ of Q with $\ell > 1$ such that $\sum_{i \in Q_1} \vec{v}[i] = \ldots = \sum_{i \in Q_\ell} \vec{v}[i] = b$. We say that the partition $Q_1, \ldots, Q_\ell$ of $\vec{v}$ into blocks of weight b is *unique* if for every partition $Q'_1, \ldots, Q'_\ell$ of weight b there exists a permutation of $1, \ldots, \ell$ such that $Q_i = Q'_{\sigma(i)}$ for all i. We will denote by $\mathcal{G}(\vec{v})$ the set of digraphs with the eigenvector $\vec{v}$.

Theorem 4 ([8]). *If all partitions of the eigenvector $\vec{v}$ are unique and their number is equal to s, then the synchronizing ratio of every k-out-regular digraph in $\mathcal{G}(\vec{v})$ is at least $\frac{k-s}{k}$.*

Note that for $s = 0, 1$ the digraphs under consideration satisfy Conjecture 2.

Another interesting observation related to the set of attainable synchronizing ratios by digraphs with given n, k was made in [9]. Note that the synchronizing ratio is necessarily divisible by $k!$, but even with this restriction not every value seems to be attainable:

Conjecture 5 ([9]). *For every k and $g \geq 1$, for n large enough there are at least g gaps in the distribution of the number of synchronizing colorings of k-out-regular digraphs with n vertices.*

As a first step towards a solution of this conjecture a series of specific examples were constructed:

Theorem 6 ([9]). *For every integers $d \geq 1$ and $n \geq 3d$ there is a k-out-regular digraph with n vertices and the synchronizing ratio $1 - \frac{1}{k^d}$.*

What is the average number of synchronizing colorings? We say that a digraph G is *totally synchronizing* if its synchronizing ratio is equal to 1, i.e., every coloring is synchronizing. Based on the computational experiments performed in [9] the following surprising conjecture was formulated:

Conjecture 7 ([9]). *For every $k \geq 2$, the fraction of totally synchronizing digraphs among all k-out-regular primitive digraphs with n vertices tends to 1 as n goes to infinity.*

This conjecture can be seen as future refinement of a recent non-trivial theorem stating that a random (at least binary) automaton is synchronizing with high probability [3, 11].

A large class of totally synchronizing digraphs was presented in [8]:

Theorem 8 ([8]). *An entrywise positive integer vector $\vec{v}$ is not partitionable if and only if all digraphs from $\mathcal{G}(\vec{v})$ are totally synchronizing.*

Moreover, relying on this theorem a stronger version of Conjecture 7 was formulated:

Conjecture 9 ([8]). *The eigenvector of a random primitive k-out-regular digraph with n vertices is not partitionable with probability 1 as n goes to infinity.*

The interesting feature of Conjecture 9 is that the statement does not involve synchronizing automata at all.

Despite the fact that road coloring problem gained a lot of attention, the following computational complexity questions remain open:

Problem 10. *Given a k-out-regular digraph G with n vertices, what is the computational complexity of checking whether G is totally synchronizing?*

Problem 11. *Given a k-out-regular digraph G with n vertices, what is the computational complexity of computing the synchronizing ratio?*

What are the reset thresholds of synchronizing colorings? A variety of questions related to optimization of the reset threshold among synchronizing colorings of a given digraph was addressed in the literature.

Conjecture 12 (Hybrid Černý–Road Coloring Problem [1]). *Every primitive k-out-regular digraph with n vertices has a synchronizing coloring with the reset threshold at most $n^2 - 3n + 3$.*

The conjectured bound cannot be lower, since for every n there is a digraph with n vertices such that all of its colorings are isomorphic and their reset threshold is equal to $n^2 - 3n + 3$ [1]. The following series of partial results concerning this conjecture was obtained.

Theorem 13 ([14]). *Every primitive k-out-regular digraph without multiple edges containing a proper prime length cycle admits a synchronizing coloring with the threshold at most $(n-1)^2$.*

Theorem 14 ([4]). *Every primitive k-out-regular digraph without multiple edges and containing a simple cycle of prime length $p < n$ admits a synchronizing coloring with the threshold at most $(2p-1)(n-1)$. Moreover, in the case $p = 2$ with multiple edges allowed, there exists a synchronizing coloring with a synchronizing word of length at most $5(n-1)$.*

Theorem 15 ([5]). *Every primitive Hamiltonian digraph has a coloring with the reset threshold at most $2n^2 - 4n + 1 - 2(n-1)\ln\frac{n}{2}$.*

In the series of papers [12, 13, 17] a significant effort was put into establishing the computational complexity of finding synchronizing colorings with small reset thresholds leading to the following theorem:

Theorem 16 ([17]). *Let k, ℓ be fixed positive integers. The problem of checking whether a given k-out-regular digraph G has a synchronizing coloring with the reset threshold at most ℓ is NP-complete for $\ell \geq 4$ and $k \geq 2$, and solvable in polynomial time in all the other cases.*

Surprisingly, but a similar problem of checking whether a given digraph has a coloring synchronized by a very short fixed word, e.g. *aba*, is also NP-complete:

Theorem 17 ([18]). *Let $\Sigma = \{a, b\}$ and $w \in \Sigma^+$. The problem of checking whether a given 2-out-regular digraph G has a coloring synchronized by w is NP-complete if w is not equal to $a^\ell, b^\ell, a^\ell b, b^\ell a$ for $\ell \geq 1$, otherwise, it is solvable in polynomial time.*

References

[1] D. S. Ananichev, M. V. Volkov, and V. V. Gusev. Primitive digraphs with large exponents and slowly synchronizing automata. *Journal of Mathematical Sciences*, 192(3):263–278, 2013.

[2] M.-P. Béal and D. Perrin. A quadratic algorithm for road coloring. *Discrete Applied Mathematics*, 169:15–29, 2014.

[3] M. V. Berlinkov. On the probability of being synchronizable. In Sathish Govindarajan and Anil Maheshwari, editors, *Algorithms and Discrete Applied Mathematics - Second International Conference, CALDAM 2016, Thiruvananthapuram, India, February 18-20, 2016, Proceedings*, volume 9602 of *Lecture Notes in Computer Science*, pages 73–84. Springer, 2016.

[4] A. Carpi and F. D'Alessandro. The synchronization problem for locally strongly transitive automata. In Rastislav Královič and Damian Niwiński, editors, *Mathematical Foundations of Computer Science 2009: 34th International Symposium, MFCS 2009, Novy Smokovec, High Tatras, Slovakia, August 24-28, 2009. Proceedings*, pages 211–222. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[5] A. Carpi and F. D'Alessandro. Independent sets of words and the synchronization problem. *Advances in Applied Mathematics*, 50(3):339–355, 2013.

[6] J. Černý. Poznámka k homogénnym experimentom s konečnými automatmi. *Matematicko-fyzikálny Časopis Slovenskej Akadémie Vied*, 14(3):208–216, 1964. In Slovak.

[7] Karel Culik, Juhani Karhumäki, and Jarkko Kari. A note on synchronized automata and road coloring problem. *International Journal of Foundations of Computer Science*, (13):459–471, 2002.

[8] V. V. Gusev and E. V. Pribavkina. On synchronizing colorings and the eigenvectors of digraphs. In Piotr Faliszewski, Anca Muscholl, and Rolf Niedermeier, editors, *41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016, August 22-26, 2016 - Kraków, Poland*, volume 58 of *LIPIcs*, pages 48:1–48:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[9] V. V. Gusev and M. Szykuła. On the number of synchronizing colorings of digraphs. In *Implementation and Application of Automata*, volume 9223 of *LNCS*, pages 127–139. Springer, 2015.

[10] C. D. Meyer. *Matrix Analysis and Applied Linear Algebra*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[11] C. Nicaud. Fast synchronization of random automata. In Klaus Jansen, Claire Mathieu, José D. P. Rolim, and Chris Umans, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2016, September 7-9, 2016, Paris, France*, volume 60 of *LIPIcs*, pages 43:1–43:12. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[12] A. Roman. P–np threshold for synchronizing road coloring. In Adrian-Horia Dediu and Carlos Martín-Vide, editors, *Language and Automata Theory and Applications: 6th International Conference, LATA 2012, A Coruña, Spain, March 5-9, 2012. Proceedings*, pages 480–489. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[13] A. Roman and M. Drewienkowski. A complete solution to the complexity of synchronizing road coloring for non-binary alphabets. *Information and Computation*, 242:383 – 393, 2015.

[14] B. Steinberg. The Černý conjecture for one-cluster automata with prime length cycle. *Theoretical Computer Science*, 412(39):5487–5491, 2011.

[15] A. N. Trahtman. The Road Coloring Problem. *Israel Journal of Mathematics*, 172(1):51–60, 2009.

[16] M. V. Volkov. Synchronizing automata and the Černý conjecture. In *Language and Automata Theory and Applications*, volume 5196 of *LNCS*, pages 11–27. Springer, 2008.

[17] V. Vorel and A. Roman. Parameterized complexity of synchronization and road coloring. *Discrete Mathematics & Theoretical Computer Science*, 17(1):283–306, 2015.

[18] V. Vorel and A. Roman. Complexity of road coloring with prescribed reset words. *Journal of Computer and System Sciences*, pages –, 2016.

On the scrambling index of non-negative matrices*

A.E. Guterman *A.M. Maksaev*

Lomonosov Moscow State University, 119991, Moscow, Russia
Moscow Center for Continuous Mathematical Education, 119002, Moscow, Russia

Abstract.

In 2009 Akelbek and Kirkland defined the scrambling index of a nonnegative primitive matrix A, denoted by $k(A)$, which is the smallest positive integer k such that any two rows of A^k have at least one positive element in coincident position. They also obtained an upper bound on scrambling index (see [1]). Scrambling index is closely related with well-known Dobrushin or delta coefficient, denoted by $\tau_1(\cdot)$, which has applications, in particular, to the theory of Markov chains and finite automata (see, for example, [2], [3]).

Due to the correspondence between matrices and directed graphs one can define the notion of scrambling index for primitive digraph G (denoted by $k(G)$).

For vertices u, v of a digraph G the notation $u \xrightarrow{l} v$ is used to indicate that there is a directed walk of the length l from u to v. If digraphs G_1 and G_2 are isomorphic, then we denote it as $G_1 \cong G_2$.

It is easy to see that the definition of scrambling index can be extended without changes to a wider class of directed graphs.

Definition 1. A *scrambling index* of a directed graph G with the set of vertices $V(G)$ is the smallest positive integer k, such that for all vertices $a, b \in V(G)$ there exists a vertex $v \in V(G)$: $a \xrightarrow{k} v$, $b \xrightarrow{k} v$. We denote this number by $k(G)$. If there is no such k, we say that $k(G) = 0$.

A characterization for class of digraphs with $k(G) \neq 0$ is presented below.

Theorem 2. *For an arbitrary digraph G the following conditions are equivalent:*

1. $k(G) \neq 0$.
2. *There exists a primitive subgraph G_1 of G such that for all $v \in V(G)$ there exists $w \in V(G_1)$ for which there exists a directed walk from v to w in G.*

Define two important examples of digraphs.

Definition 3. 1) Wielandt digraph $\mathscr{W}_n$, $n \geqslant 2$, is the digraph with the vertex set $V = \{1, 2, \ldots, n\}$ and edge set $E = \{(1,2),\ (2,3), \ldots, (n-1, n), (n, 1)\} \cup \{(n-1, 1)\}$.

2) Digraph $\mathscr{H}_n$, $n \geqslant 3$, is the digraph with the vertex set $V = \{1, 2, \ldots, n\}$ and edge set $E = \{(i, i+1) \mid 1 \leqslant i \leqslant n-2\} \cup \{(n-1, 1), (n-2, 1), (n, n-1)\}$.

Akelbek and Kirkland in paper [1] have proved the following theorem.

Theorem 4 (see [1], Theorem 3.18)**.** *Let D be a primitive digraph of order $n \geqslant 2$. Then*

$$k(D) \leqslant \left\lceil \frac{(n-1)^2 + 1}{2} \right\rceil.$$

If $n \geqslant 3$ then equality holds if and only if $D \cong \mathscr{W}_n$.

*The work is partially financially supported by the grant RSF 16-11-10075

The purpose of this report is to generalize this result to the non-primitive case, to discuss upper bounds on scrambling index, depending on digraph parameters, and to investigate boundary cases.

Here is a helpful criterion of digraph primitivity.

Theorem 5 (see [4], Theorem 3.2.1)**.** *Digraph G is primitive if and only if G is strongly connected and $k(G) \neq 0$.*

This criterion implies that it is reasonable to consider only not strongly connected digraphs. Following these considerations, we introduce the definition of digraph partition.

Definition 6. We say that G have a $(G_1 \to G_2)$ - partition, if G_1 and G_2 are subgraphs of the digraph G, sets of vertices of which are non-intersect and provide a partition of $V(G)$ (i. e. $V(G_1) \cap V(G_2) = \varnothing, V(G_1) \cup V(G_2) = V(G)$), moreover each edge $e = (v_1, v_2) \in E(G)$ satisfies the following condition: either $e \in E(G_1)$, or $e \in E(G_2)$, or $v_1 \in V(G_1), v_2 \in V(G_2)$.

From geometrical point of view it means that G is partitioned into two non-intersecting components (G_1 and G_2), connected with only edges from G_1 to G_2. Our technique uses the following

Lemma 7. *Let G, G_1, G_2 be arbitrary digraphs and G have a $(G_1 \to G_2)$-partition. Assume that $k(G) \neq 0$. Then $k(G_2) \neq 0$ and $k(G) \leqslant k(G_2) + |G_1|$, where $|G_1|$ is the order of digraph G_1.*

In particular, the following result, which is the generalization of Theorem 4, has been obtained.

Theorem 8. *1) Let G be a not strongly connected digraph of order $n \geqslant 3$. Then*

$$k(G) \leqslant 1 + \left\lceil \frac{(n-2)^2 + 1}{2} \right\rceil.$$

If $n \geqslant 4$, then equality holds if and only if $G \cong \mathscr{H}_n$.

2) Let G be an arbitrary digraph of order $n \geqslant 3$. Then

$$k(G) \leqslant \left\lceil \frac{(n-1)^2 + 1}{2} \right\rceil.$$

Equality holds if and only if $G \cong \mathscr{W}_n$.

Definition 9. We say that a map T preserves the value l ($l \geqslant 0$) of the scrambling index if for all $A \in M_n(\mathbf{B})$ the condition $k(A) = l$ implies that $k(T(A)) = l$. We say that T preserves the scrambling index if it preserves all values $l \geqslant 0$.

Mentioned results give us some information about maps acting on the matrix space over Boolean semiring $\mathbf{B}$ and preserving the scrambling index. Essentially, we got the following result:

Theorem 10. *Let $T\colon M_n(\mathbf{B}) \to M_n(\mathbf{B})$ ($n \geqslant 3$) be a linear operator which preserves the scrambling index, then T is bijective.*

Furthermore, we obtain a complete characterization of all linear bijective scrambling index preservers over arbitrary antinegative semiring with 1 and without zero divisors.

Theorem 11. *Let $\mathscr{S}$ be an antinegative semiring with 1 and without zero divisors, $n \geqslant 3$ and $T\colon M_n(\mathscr{S}) \to M_n(\mathscr{S})$ be a (left) linear map preserving scrambling index. Then there exists a permutation matrix P and a matrix $B \in M_n(\mathscr{S})$, with no zero elements, such that*

$$T(A) = P^T(A \circ B)P \tag{1}$$

for an arbitrary matrix $A \in M_n(\mathscr{S})$. Conversely, each map in the form (1) is the linear map preserving scrambling index.

Here $A \circ B$ denotes the Hadamard or Schur (elementwise) product.

In the talk we also discuss similar results related to linear maps preserving some values of the scrambling index:

Theorem 12. *Let $n \geqslant 3$ and $T\colon M_n(\mathbf{B}) \to M_n(\mathbf{B})$ be a map. Then T is additive bijective map, which preserves the value 1 of the scrambling index, if and only if there exist permutation matrices P and Q, such that*

$$T(A) = PA\,Q.$$

Theorem 13. *Let $n \geqslant 3$ and $T\colon M_n(\mathbf{B}) \to M_n(\mathbf{B})$ be a map. Then T is additive bijective map, which preserves the value 0 of the scrambling index, if and only if there exists a permutation matrix P, such that*

$$T(A) = P^TA\,P.$$

References

[1] M. Akelbek, S. Kirkland. Coefficients of ergodicity and scrambling index, Linear Algebra Appl. 430 (2009) 1111-1130.

[2] E. Seneta. Nonnegative Matrices and Markov Chains, Springer-Verlag, New York, 1981.

[3] A. Paz. Introduction to Probabilistic Automata, Academic Press, New York, 1971.

[4] M. Akelbek. A Joint Neighbour Bound for Primitive Digraphs, Ph.D. Thesis, University of Regina, 2008.

On resolving several objects in the king grid

Anni Hakanen *Tero Laihonen*

Department of Mathematics and Statistics, University of Turku, FI-20014 Turku, Finland
anehak@utu.fi, terolai@utu.fi

The concept of a resolving set was introduced independently by Slater [11] and Harary and Melter [7]. This concept emerges naturally from many diverse areas such as coin weighing problem [1], network discovery and verification [2], and robot navigation [9]. For other recent developments, see [6, 3, 5]. Resolving sets are also related to identifying codes and locating dominating sets which are widely studied — see the list in the web-site [10] for papers on these topics.

Consider a connected, finite, simple, and undirected graph G with vertices V and edges E. Let S be a subset of V. When we think of S as an ordered set $(s_1,\ s_2,\ \ldots,\ s_{|S|})$, we can try to locate an another vertex set X with the distance array

$$\mathcal{D}_S(X) = (d(s_1, X),\ d(s_2, X),\ \ldots,\ d(s_{|S|}, X)),$$

where $d(s_i, X) = \min\limits_{x\in X} d(s_i, x)$ is the shortest distance from s_i to some vertex of X.

Definition 1. The set S is an *ℓ-resolving set* (or *ℓ-set resolving set*) of $G = (V, E)$, where $\ell \leq |V|$, if for every pair of subsets X and Y, with $|X| \leq \ell$ and $|Y| \leq \ell$, we have

$$\mathcal{D}_S(X) \neq \mathcal{D}_S(Y).$$

In other words, an ℓ-resolving set can locate up to ℓ vertices at the same time. Let a surveillance network be modelled by a graph. When we place sensors to the vertices corresponding to the elements of an ℓ-resolving set S, the sensors can locate up to ℓ intruders by sending signals to measure the distance.

Every graph has an ℓ-resolving set for any $\ell \leq |V|$, since V is always such a set. We can simply check which elements of $\mathcal{D}_V(X)$ are 0 and we have located all elements of X. Therefore the existence of resolving sets is not of interest but the size of them is. We denote with $\beta_\ell(G)$ the *ℓ-set-metric dimension* of G, which is the smallest possible cardinality of an ℓ-resolving set of G. An *ℓ-set-metric basis* is an ℓ-resolving set that is of cardinality $\beta_\ell(G)$.

There has been a lot of research on what values $\beta_\ell(G)$ gets with different graphs for $\ell = 1$. For example Khuller, Raghavachari & Rosenfeld proved in 1996 [9] that $\beta_1(G) = 1$ if and only if G is a path and that for a d-dimensional grid graph $\beta_1(G) = d$. Chartrand *et al.* proved in 2000 [6] that for an n-vertex graph $\beta_1(G) = n - 1$ if and only if G is a complete graph. They also gave characterisations for all n-vertex graphs with $\beta_1(G) = n-2$. Resolving several objects has been studied recently in [8]. There the two-dimensional grid graph and the binary hypercube are considered.

Let us denote the path of n vertices by P_n. It was shown in [8] that for the Cartesian product $P_m \Box P_n$ of two paths we have

$$\beta_2(P_m \Box P_n) = \min\{m, n\} + 2.$$

Earlier, it was proven by Khuller *et al.* [9] that

$$\beta_1(P_m \Box P_n) = 2.$$

If two vertices x and y are adjacent, we denote $x \sim y$. Let us denote the *strong product* of two graphs $G = (V, E)$ and $H = (V', E')$ by $G \boxtimes H$, that is, it has as the vertex set

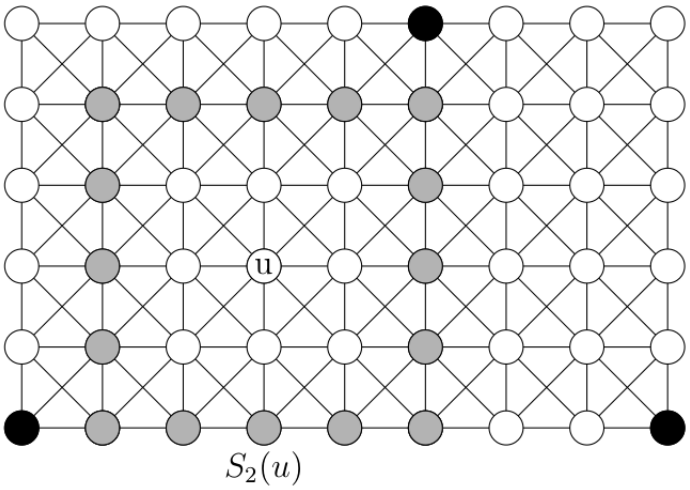

Figure 1: The 9×6 king grid $P_9 \boxtimes P_6$. The black vertices form a 1-set-metric basis for the graph.

the Cartesian product $V \times V'$ and there is an edge between (u_1, u_2) and (v_1, v_2) if one of the following three conditions hold: 1) $u_1 = v_1$ and $u_2 \sim v_2$, 2) $u_1 \sim v_1$ and $u_2 = v_2$ or 3) $u_1 \sim v_1$ and $u_2 \sim v_2$. In this paper, we consider the *king grid* $P_m \boxtimes P_n$. This graph has been studied for related topics, see, for example [4].

The king grid is basically a two-dimensional grid graph with diagonal edges in addition to vertical and horizontal ones (see Figure 1). As such, it mimics the movement of the king on a chess board.

The vertices of a king grid can be thought as $\mathbb{N} \times \mathbb{N}$ lattice points. We can give each vertex two coordinates and write the set of vertices of an $m \times n$ king grid as $\{(i, j) \mid i = 1, \ldots, m,\ j = 1, \ldots, n\}$. Now the distance between the vertices $u = (u_1, u_2)$ and $v = (v_1, v_2)$ is $d(u, v) = \max\{|u_1 - v_1|,\ |u_2 - v_2|\}$.

To ease notations, we define the ith *column* for $i \in [1, m]$ as $C_i = \{(i, j)|\ j = 1, \ldots, n\}$. A *section* is the union of consecutive columns:

$$C_i^j = \bigcup_{k=i}^{j} C_k.$$

We denote with $S_r(u) = \{v \in V|\ d(u, v) = r\}$ the set of vertices that are at the distance of r from the vertex u. Note that if $r \neq r'$, then $S_r(u) \cap S_{r'}(u) = \emptyset$.

For completeness, we first give the following result from [12].

Theorem 2. *Let $P_m \boxtimes P_m$ be an $m \times m$ king grid with $2 \leq m$. Then*

$$\beta_1(P_m \boxtimes P_m) = 3.$$

Sketch of proof. The greatest distance between any two vertices is $m - 1$. Therefore each element of $\mathcal{D}_S(X)$ has m possible values. If $|S| = k$, then there are m^k possible distance arrays. Since $\ell = 1$ no distance array can have more than one zero. Since there are only $m^2 - 1$ acceptable distance arrays of length two but m^2 vertices, we have $\beta_1(P_m \boxtimes P_m) \geq 3$.

If S is a subset of V that contains any three of the graph's corner vertices, it is a 1-resolving set of $P_m \boxtimes P_m$. Therefore $\beta_1(P_m \boxtimes P_m) = 3$. □

The next result considers the 1-set-metric dimension for any king grid. In [12], Rodríquez-Velázquez *et al.* gave a construction giving $\beta_1(P_n \boxtimes P_m) \leq \lceil \frac{n+m-2}{n-1} \rceil$. They also presented a conjecture that this upper bound is optimal for any integers n and m such that $2 \leq n < m$. This was recently proved by Barragán-Ramírez and Rodríquez-Velázquez in [13]. They used the diameter and bipartiteness of graphs. In this paper, we present a direct and simple proof.

Theorem 3. *Let $P_m \boxtimes P_n$ be an $m \times n$ king grid with $2 \leq n < m$. Then*

$$\beta_1(P_m \boxtimes P_n) = \left\lceil \frac{n+m-2}{n-1} \right\rceil.$$

Sketch of proof. Assume that S is a 1-resolving set.

Let first n be even. Each $(n-1) \times n$-section C_i^{i+n-2} contains at least one element of S. Indeed, otherwise we would have $\mathcal{D}(a) = \mathcal{D}(b)$ for $a = (i + \frac{n-2}{2}, \frac{n}{2})$ and $b = (i + \frac{n-2}{2}, \frac{n}{2} + 1)$ — for illustration see Figure 2(i). Moreover, in the both ends of the king grid, the $n \times n$-sections have $|C_1^n \cap S| \geq 2$ and $|C_{m-(n-1)}^m \cap S| \geq 2$ (it is easy to see that one element of S is not enough). Let first $m \geq 2n$. Now let us partition the $m \times n$ king grid as follows (see Figure 3). Take first the two $n \times n$-sections at the both ends of the grid and then divide the middle section into as many disjoint $(n-1) \times n$-sections as possible (there are at most $n-2$ leftover columns outside the sections, in the figure there is one column marked by gray vertices). Now the observations above give $|S| \geq 2 + 2 + \lfloor \frac{m-2n}{n-1} \rfloor$, which equals the conjectured lower bound. In the case $n < m < 2n$, it is easy to show that $|S| \geq 3$.

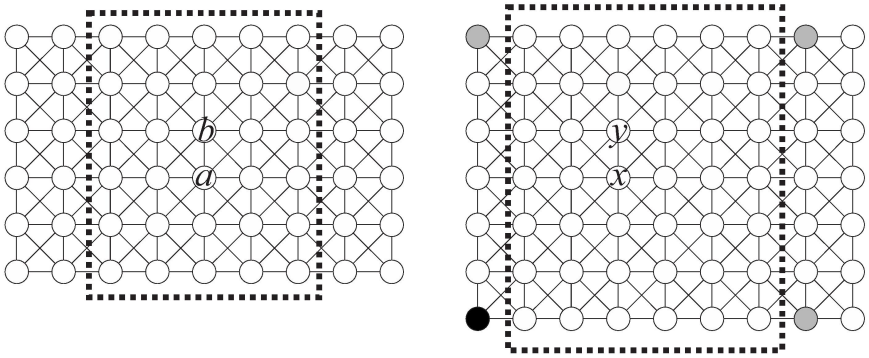

Figure 2: (i) The section C_i^{i+n-2} illustrated for $n = 6$ and the vertices a and b, (ii) The section C_i^{i+n-2} for $n = 7$ and the vertex $(i-1, 1)$ is the black vertex.

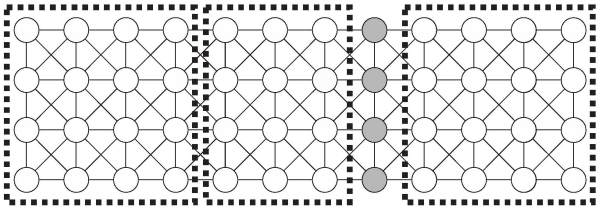

Figure 3: The partition for $n = 4$ and $m = 12$.

The case for n odd goes similarly. Just notice that now the $(n-1) \times n$-section C_i^{i+n-2} can be empty of the elements of S, but in that case the neighbouring columns C_{i-1} on the left and C_{i+n-1} on the right contain both at least two elements of S (see Figure 2(ii)). Indeed, suppose that the section C_i^{i+n-2} is empty. This implies that $(i-1,1)$ belongs to S, since otherwise $\mathcal{D}(x) = \mathcal{D}(y)$ for $x = (i + \frac{n-3}{2}, \frac{n+1}{2})$ and $y = (i + \frac{n-3}{2}, \frac{n+1}{2} + 1)$. In the same way, the vertex $(i-1,n)$ belongs to S. Similarly we can show that $(i+n-1,1)$ and $(i+m-1,n)$ belong to S. Therefore, it is easy to show that in average there is at least one element of S in each of the $(n-1) \times n$-section. For more details, see [14].

□

When $\ell = 2$, the vertices at the frame of the king grid can "hide" behind its neighbour closer to the center. Therefore all vertices at the frame of the grid must be in any 2-resolving set, and it turns out that this condition is sufficient.

Theorem 4. *Let $P_m \boxtimes P_n$ be an $m \times n$ king grid with $2 \leq n \leq m$. Then*

$$\beta_2(P_m \boxtimes P_n) = 2m + 2n - 4.$$

Sketch of proof. Let $u = (u_1,\ u_2)$ be a vertex at the frame of the graph i.e. $u_1 \in \{1, m\}$ or $u_2 \in \{1, n\}$.

Assume that $u_2 = 1$. Let $v = (v_1, v_2) \neq u$ and $u' = (u_1,\ 2)$. Now $d(u,v) = \max\{|u_1 - v_1|,\ |1 - v_2|\}$ and $d(u',v) = \max\{|u_1 - v_1|,\ |2 - v_2|\}$. If $v_2 = 1$, then $|u_1 - v_1| \geq 1$, since $v \neq u$. Now $d(u,v) = d(u',v)$. If $v_2 \geq 2$, then $|2 - v_2| < |1 - v_2|$ and therefore $d(u,v) \geq d(u',v)$.

Let S be a 2-resolving set of $P_m \boxtimes P_n$ and assume that $u \notin S$. Consider two vertex sets $A = \{u'\}$ and $B = \{u, u'\}$. Now $\mathcal{D}_S(A) = \mathcal{D}_S(B)$ since no vertex of S can be closer to u than u' as we saw above. Therefore $u \in S$.

The other cases are handled similarly, namely $u_1 = 1$, $u_1 = m$, and $u_2 = n$.

This shows that all vertices at the frame of the graph must be included in the resolving set. With some effort one can show that these vertices indeed form a 2-resolving set [14]. □

When $\ell \geq 3$, we cannot leave any vertex out of the ℓ-resolving set. If we do, we can always find two sets of vertices that have the same distance array.

Theorem 5. *Let $P_m \boxtimes P_n$ be an $m \times n$ king grid with $2 \leq n \leq m$. Then*

$$\beta_{\geq 3}(P_m \boxtimes P_n) = mn.$$

Proof. In Theorem 4 we saw that the vertices at the frame of the graph must be included in any 2-resolving set. Therefore they must also be in any 3-resolving set. If $n = 2$ or $m = 2$, all vertices are at the frame of the graph and the claim holds.

Let S be a 3-resolving set of $P_m \boxtimes P_n$ where $2 < n \leq m$. Assume that $u = (u_1, u_2) \notin S$ where $u_1 \in [2, m-1]$ and $u_2 \in [2, n-1]$. Let $v = (u_1 - 1, u_2)$ and $w = (u_1 + 1, u_2)$.

Assume that there is a vertex $s = (s_1, s_2) \in S$ such that $d(s,u) < d(s,v)$ and $d(s,u) < d(s,w)$.

- If $s_1 = u_1$, then $d(s,u) < d(s,v)$ implies that $|s_2 - u_2| \leq |s_1 - u_1| = 0$ and therefore $s_2 = u_2$. Now $s = u$ but this is a contradiction, since $u \notin S$.
- If $s_1 < u_1$, then $s_1 - u_1 < 0$ and therefore $|s_1 - u_1 + 1| < |s_1 - u_1|$. In fact $|s_1 - u_1 + 1| = |s_1 - u_1| - 1$. Since $d(s,u) < d(s,v)$,

$$\max\{|s_1 - u_1|,\ |s_2 - u_2|\} < \max\{|s_1 - u_1 + 1|,\ |s_2 - u_2|\}.$$

Now $|s_2 - u_2| < |s_1 - u_1|$, because otherwise $d(s,u) = |s_2 - u_2| = d(s,v)$. But now $|s_2 - u_2| \leq |s_1 - u_1| - 1 = |s_1 - u_1 + 1|$, and therefore $d(s,u) = |s_1 - u_1| > |s_1 - u_1 + 1| = d(s,v)$, which is a contradiction.

- If $s_1 > u_1$, we can just replace v and $|s_1 - u_1 + 1|$ with w and $|s_1 - u_1 - 1|$ in the previous case.

Therefore, every $s \in S$ is as close or closer to either v or w than u. But now $\mathcal{D}_S(A) = \mathcal{D}_S(B)$, where $A = \{v, w\}$ and $B = \{u, v, w\}$. Therefore S cannot be a 3-resolving set if it does not include all vertices of the graph. □

References

[1] N. Alon, D.N. Kozlov and V.H. Vu. The geometry of coinweighing problem, in Proc. 37th Annual Symposium on Foundation of Computer Science, FOCS96, IEEE, 1998, pp. 524–532.

[2] Z. Beerliova, F. Eberhard, T. Erlebach, A. Hall, M. Hoffmann, M. Mihal'ák, and L. S. Ram. Network discovery and verification. Graph-theoretic concepts in computer science, Lecture Notes in Comput. Sci., 3787, pp. 127–138, Springer, Berlin, 2005.

[3] J. Cáceres, C. Hernando, M. Mora, I. M. Pelayo, and M. L. Puertas. On the metric dimension of infinite graphs. Discrete Appl. Math., 160, pp. 2618–2626, 2012.

[4] J. Cáceres, C. Hernando, M. Mora, I. M. Pelayo, and M. L. Puertas. Locating-dominating codes: Bounds and extremal cardinalities. Discrete Appl. Math., 220, pp. 38–45, 2013.

[5] J. Cáceres, C. Hernando, M. Mora, I.M. Pelayo, M. L. Puertas, C. Seara, and D.R. Wood. On the metric dimension of Cartesian products of graphs. SIAM J. Discrete Math., 21, pp. 423–441, 2007.

[6] G. Chartrand, L. Eroh, M. A. Johnson, and O. R. Oellermann. Resolvability in graphs and the metric dimension of a graph. Discrete Appl. Math., 105, 99–113, 2000.

[7] F. Harary and R. Melter. On the metric dimension of a graph. Ars Combinatoria, 2, pp. 191–195, 1976.

[8] T. Laihonen. The metric dimension for resolving several objects. Information Processing Letters, 116, pp. 694–700, 2016.

[9] S. Khuller, B. Raghavachari, and A. Rosenfeld. Landmarks in graphs. Discrete Appl. Math., 70, pp. 217–229, 1996.

[10] A. Lobstein. Watching systems, identifying, locating-dominating and discrminating codes in graphs, a bibliography. Published electronically at `http://perso.telecom-paristech.fr/~lobstein/debutBIBidetlocdom.pdf`.

[11] P. Slater. Leaves of trees. Proceedings of the Sixth Southeastern Conference on Combinatorics, Graph Theory, and Computing (Florida Atlantic Univ., Boca Raton, Fla., 1975), pp. 549–559. Congressus Numerantium, No. XIV, Utilitas Math., Winnipeg, Man., 1975.

[12] J. A. Rodríguez-Velázquez, D. Kuziak, I. G. Yero, and J. M. Sigarreta. The metric dimension of strong product graphs. Carpathian J. Math., 31 (2015), No. 2, pp. 261–268.

[13] G. A. Barragán-Ramírez and J. A. Rodríquez-Velázquez. The Local Metric Dimension of Strong Product Graphs. Graphs and Combinatorics, 32 (2016), pp. 1263–1278.

[14] A. Hakanen. Resolving sets and resolving several objects in the finite king grid. Master's thesis, University of Turku, in preparation, 2017.

Two New Classes of Locating-Dominating Codes

Ville Junnila *Tero Laihonen* *Tuomo Lehtilä**

Department of Mathematics and Statistics, University of Turku, FI-20014 Turku, Finland
viljun@utu.fi, terolai@utu.fi and tualeh@utu.fi

1 Introduction

Sensor networks are systems designed for environmental monitoring. Various location detection systems such as fire alarm and surveillance systems can be viewed as examples of sensor networks. For location detection, a sensor can be placed in any location of the network. The sensor monitors its neighbourhood (including the location of the sensor itself) and reports possible irregularities such as a fire or an intruder in the neighbouring locations. Based on the reports of the sensors, a central controller attempts to determine the location of a possible irregularity in the network. Usually, the aim is to minimize the number of sensors in the network. More explanation regarding location detection in sensor networks can be found in [4, 12, 15].

A sensor network can be modeled as a simple, undirected and connected graph $G = (V, E)$ as follows: the set of vertices V of the graph represents the locations of the network and the edge set E of the graph represents the connections between the locations. In other words, a sensor can be placed in each vertex of the graph and the sensor placed in the vertex u monitors u itself and the vertices neighbouring u. In what follows, we present some basic terminology and notation regarding graphs. The *open neighbourhood* of $u \in V$ consists of the vertices adjacent to u and it is denoted by $N(u)$. The *closed neighbourhood* of u is defined as $N[u] = \{u\} \cup N(u)$. A nonempty subset C of V is called a *code* and the elements of the code are called *codewords*. In this paper, the code C (usually) represents the set of locations where the sensors have been placed on. For the set of sensors monitoring a vertex $u \in V$, we use the following notation:

$$I(G, C; u) = I(C; u) = I(u) = N[u] \cap C.$$

We call $I(u)$ the *identifying set* (or the *I-set*) of u. The notation of identifying set can also be generalized for a subset U of V as follows:

$$I(G, C; U) = I(C; U) = I(U) = \bigcup_{u \in U} I(C; u).$$

As stated above, a sensor $u \in V$ reports that an irregularity has been detected if there is (at least) one in the closed neighbourhood $N[u]$. In what follows, we divide into two different situations depending on the capability of a sensor to distinguish whether the irregularity has been spotted in the location of the sensor itself or in its (open) neighbourhood. More precisely, we have the following two cases:

(i) In the first case, we assume that a sensor $u \in V$ reports 1 if the there is an irregularity in $N[u]$, and otherwise it reports 0.

*Research supported by the University of Turku Graduate School (UTUGS).

(ii) In the second case, we assume that a sensor $u \in V$ reports 2 if there is an irregularity in u, it reports 1 if there is one in $N(u)$ and no irregularities in u, and otherwise it reports 0.

Assume first that the sensors work as in (i). Notice then that if the sensors in the code C are located in such places that $I(C;u)$ is nonempty and unique for all $u \in V$, then an irregularity in the network can be located by comparing $I(C;u)$ to identifying sets of other vertices. This leads to the following definition of *identifying codes*, which were first introduced by Karpovsky *et al.* in [11]. For various papers regarding identification and related problems, we refer to the online bibliography [13].

Definition 1. A code $C \subseteq V$ is *identifying* in G if for all distinct $u, v \in V$ we have $I(C;u) \neq \emptyset$ and

$$I(C;u) \neq I(C;v).$$

An identifying code C in a finite graph G with the smallest cardinality is called *optimal* and the number of codewords in an optimal identifying code is denoted by $\gamma^{ID}(G)$.

Let C be an identifying code in G. By the definition, the identifying code C works correctly if there is simultaneously at most one irregularity in the network. However, using the identifying code C, we cannot locate or even detect more than one irregularity in the network. Indeed, there might exist vertices $u, v_1, v_2 \in V$ such that $I(C;u) = I(C;\{v_1, v_2\})$. If now the sensors in $I(C;u)$ output 1 and all the other sensors output 0, then it is deduced that the irregularity is in u. However, as the irregularities could also be in v_1 and v_2, we might determine a false location and more disturbingly not even notice that something is wrong. To overcome this problem, in [7], so called self-identifying codes, which are able to locate one irregularity and detect multiple ones, were introduced. (Notice that in the original paper self-identifying codes are called 1^+-identifying.) The formal definition of self-identifying codes is given as follows.

Definition 2. A code $C \subseteq V$ is called *self-identifying* in G if the code C is identifying in G and for all $u \in V$ and $U \subseteq V$ such that $|U| \geq 2$ we have

$$I(C;u) \neq I(C;U).$$

A self-identifying code C in a finite graph G with the smallest cardinality is called *optimal* and the number of codewords in an optimal self-identifying code is denoted by $\gamma^{SID}(G)$.

In addition to [7], self-identifying codes have also been previously discussed in [9, 10]. In these papers, two useful characterizations have been presented for self-identifying codes. These characterizations are presented in the following theorem.

Theorem 3 ([7, 9, 10])**.** *Let C be a code in G. Then the following statements are equivalent:*

(i) The code C is self-identifying in G.

(ii) For all distinct $u, v \in V$, we have $I(C;u) \setminus I(C;v) \neq \emptyset$.

(iii) For all $u \in V$, we have $I(C;u) \neq \emptyset$ and

$$\bigcap_{c \in I(C;u)} N[c] = \{u\}.$$

As stated earlier, self-identifying codes can locate one irregularity and detect multiple ones. Besides that, the characterization (iii) of the previous theorem also gives another useful property for self-identifying codes. Namely, the location of an irregularity can be determined without comparison to other identifying sets, since for all $u \in V$ the neighbourhoods of the codewords in $I(u)$ intersect uniquely in u.

So far, we have discussed the case where it is assumed that each sensor outputs 1 or 0 depending on whether there is an irregularity in the neighbourhood or not. In what follows, we now focus on the case (ii) where a sensor can also distinguish if the irregularity is on the location of the sensor itself. Then notice that if the sensors in the code C are located in such places that $I(C;u)$ is nonempty and unique for all $u \in V \setminus C$, then an irregularity in the network can be located by comparing $I(C;u)$ to identifying sets of other non-codewords. Indeed, we do not have to worry about vertices in C as an irregularity in such locations is immediately determined by a sensor outputting 2. This leads to the following definition of *locating-dominating codes (or sets)*, which were first introduced by Slater in [14, 16, 17].

Definition 4. A code $C \subseteq V$ is *locating-dominating* in G if for all distinct $u, v \in V \setminus C$ we have $I(C;u) \neq \emptyset$ and

$$I(C;u) \neq I(C;v).$$

A locating-dominating code C in a finite graph G with the smallest cardinality is called *optimal* and the number of codewords in an optimal locating-dominating code is denoted by $\gamma^{LD}(G)$.

Comparing the definitions of identifying and locating-dominating codes, we immediately notice their apparent similarities; in the case of identification we require that the identifying sets $I(u)$ are unique for all vertices and in the case of location-domination the same is required for non-codewords. Therefore, as self-identifying codes are a natural specialization of regular identifying codes, it is obvious to consider if something similar could be done for locating-dominating codes. Indeed, the characterizations of Theorem 3 gives two natural ways to define new types of locating-dominating codes with similar kind of beneficial properties as self-identifying codes have over regular identifying codes. The definitions of these codes are given as follows.

Definition 5. A code $C \subseteq V$ is *self-locating-dominating* in G if for all $u \in V \setminus C$ we have $I(C;u) \neq \emptyset$ and

$$\bigcap_{c \in I(C;u)} N[c] = \{u\}.$$

A self-locating-dominating code C in a finite graph G with the smallest cardinality is called *optimal* and the number of codewords in an optimal self-locating-dominating code is denoted by $\gamma^{SLD}(G)$.

Definition 6. A code $C \subseteq V$ is *solid-locating-dominating* in G if for all distinct $u, v \in V \setminus C$ we have

$$I(C;u) \setminus I(C;v) \neq \emptyset.$$

A solid-locating-dominating code C in a finite graph G with the smallest cardinality is called *optimal* and the number of codewords in an optimal solid-locating-dominating code is denoted by $\gamma^{DLD}(G)$.

In the following theorem, we present characterizations for self-locating-dominating and solid-locating-dominating codes. Comparing these characterizations to the original

definitions of the codes, the differences of the codes become apparent. In particular, it immediate that each self-locating-dominating code is also a solid-locating-dominating code.

Theorem 7. *We have the following characterizations:*

(i) A code $C \subseteq V$ is self-locating-dominating if and only if for all distinct $u \in V \setminus C$ and $v \in V$ we have

$$I(C;u) \setminus I(C;v) \neq \emptyset.$$

(ii) A code $C \subseteq V$ is solid-locating-dominating if and only if for all $u \in V \setminus C$ we have

$$\left(\bigcap_{c \in I(C;u)} N[c] \right) \setminus C = \{u\}.$$

As discussed earlier, self-identifying codes have benefits over regular identifying codes; they detect more than one irregularity and locate one irregularity without comparison to other identifying sets. In what follows, we study the same properties concerning self-locating-dominating and solid-locating-dominating codes:

- Let us begin by considering the ability to locate an irregularity without comparison to other identifying sets. For self-locating-dominating codes, this property immediately follows from the definition. Analogously, the property is obtained for solid-locating-dominating codes by Theorem 7(ii).
- It can also be shown that if C is a self-locating-dominating code in G, then it can detect if there are multiple irregularities. On the other hand, solid-locating-dominating codes may have problems detecting multiple irregularities if they can happen in locations with sensors.

In the paper, our main focus is on the new types of locating-dominating codes. However, we also present some results for regular locating-dominating codes. In Section 2, we consider the different types of locating-dominating codes in the Cartesian product of two complete graphs, which is sometimes also called the rook's graph. Furthermore, in the rook's graphs, we obtain optimal codes for regular location-domination, self-location-domination and solid-location-domination. In Section 3, we consider similar problems in the binary Hamming space (or hypercube) $\mathbb{F}^n$, where n is a positive integer. In particular, we present an infinite family of optimal self-locating-dominating codes and construct regular locating-dominating codes with the smallest known cardinalities; especially proving that $309 \leq \gamma^{LD}(\mathbb{F}^{11}) \leq 320$.

2 Location-domination in the rook's graphs

In this section, we consider the different locating-dominating codes in the Cartesian product of two complete graphs. The Cartesian product of graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is $G_1 \Box G_2 = (V_1 \times V_2, E)$ where $(x,y)(x',y') \in E$ if and only if $x = x'$ and $yy' \in E_2$ or $y = y'$ and $xx' \in E_1$. If K_n and K_m are two complete graphs of order n and m, respectively, then $K_n \Box K_m$ is known as rook's graph and can be viewed as a chess board with m rows and n columns. The closed neighbourhood of a vertex is determined by the movement of a rook in chess. We denote $V(K_n) = \{x_1, \ldots, x_n\}$, $V(K_m) = \{y_1, \ldots, y_m\}$ and kth *row* with $R_k = \{(x_i, y_k) \mid i = 1, \ldots, n, 1 \leq k \leq m\}$ (resp. hth *column* $P_h = \{(x_h, y_i) \mid i = 1, \ldots, m, 1 \leq k \leq n\}$).

Previously identification in rook's graphs has been studied in [6] and [5] and self-identification in [9] and [10]. In what follows, we are going to find optimal locating-dominating, self-locating-dominating and solid-locating-dominating codes in the rook's graphs. In the following theorem, we first begin by considering self-locating-dominating codes.

Theorem 8. *Let $G = K_n \Box K_m$ be a rook's graph with $m \geq n \geq 1$. We have*

$$\gamma^{SLD}(G) = \begin{cases} m, & m \geq 2n, \text{ or } n = 1, \\ 2n, & 2n > m > n \geq 2, \\ 2n-1, & m = n > 2, \\ 4, & n = m = 2. \end{cases}$$

In the following theorem, we will see that optimal solid-locating-dominating codes are mostly of the same size as optimal self-locating-dominating codes. However, this is only a superficial similarity. Actually, the structures of solid-locating-dominating codes vary more and there are more of them. For example, code $R_1 \cup P_1$ is an optimal solid-locating-dominating code when $n = m$ but it is not a self-locating-dominating code.

Theorem 9. *Let $G = K_n \Box K_m$ be a rook's graph with $m \geq n \geq 1$. We have*

$$\gamma^{DLD}(G) = \begin{cases} m, & m \geq 2n \geq 4 \text{ or } n = 2, \\ 2n, & 2n > m > n > 2, \\ 2n-1, & m = n > 2, \\ m-1, & n = 1. \end{cases}$$

Finally, in the following theorem, we present optimal bounds for locating-dominating codes in the rook's graphs.

Theorem 10. *Let $G = K_n \Box K_m$ be a rook's graph with $m \geq n \geq 1$. We have*

$$\gamma^{LD}(G) = \begin{cases} m-1, & m \geq 2n, \\ \left\lceil \frac{2n+2m}{3} \right\rceil - 1, & n \leq m \leq 2n-1. \end{cases}$$

Remark 11. In a forthcoming paper, we will discuss codes for location in graphs which are Cartesian products of several cliques. In particular, we will show that the conjecture of Goddard and Wash [5, Conjecture 4.3] concerning the cardinality of identifying codes in $K_n \Box K_n \Box K_n$ does not hold.

3 Location-domination in the binary Hamming spaces

In this section, we consider self-locating-dominating and solid locating-dominating codes in binary Hamming spaces of length n. A binary Hamming space of length n is a graph with the vertex set $\mathbb{F}^n = \{0,1\}^n$, and two vertices have an edge between them if they differ in exactly one coordinate. Vertices of $\mathbb{F}^n$ are called *words*. The distance $d(x,y)$ is the number of coordinates where words x and y differ. The *minimum distance* of code C is $\min\{d(c_1,c_2) \mid c_1, c_2 \in C\}$. The sizes of optimal self-locating-dominating and solid-locating-dominating codes in $\mathbb{F}^n$ are denoted by $\gamma^{SLD}(\mathbb{F}^n) = \gamma^{SLD}(n)$ and $\gamma^{DLD}(\mathbb{F}^n) = \gamma^{DLD}(n)$, respectively.

In what follows, we first concentrate on the case of self-locating-dominating codes. In particular, we present an infinite family of optimal self-locating-dominating codes in

binary Hamming spaces. This result is based on the following theorem, where we give a characterization for self-locating-dominating codes. The characterization is based on the fact that if a non-codeword has at least three codewords in its I-set, then no other word can have those same codewords in its I-set.

Theorem 12. *A code C is a self-locating-dominating code in $\mathbb{F}^n$ if and only if for each non-codeword w we have $|I(w)| \geq 3$.*

Compare this characterization to an analogous result for self-identifying codes presented in [7]: a code C is self-identifying in $\mathbb{F}^n$ if and only if for each $x \in \mathbb{F}^n$ we have $|I(C; x)| \geq 3$. With the characterization of Theorem 12 we form a lower bound for self-locating-dominating codes.

Theorem 13. *Let $n \geq 3$. We have*

$$\gamma^{SLD}(n) \geq \left\lceil \frac{3 \cdot 2^n}{n+3} \right\rceil.$$

The following cardinalities of optimal self-locating-dominating codes in $\mathbb{F}^n$ are gained with constructions based on linear code similar to the well-known Hamming codes.

Theorem 14. *Let n and k be positive integers such that $n = 3(2^k - 1)$. Then we have*

$$\gamma^{SLD}(n) = 2^{3(2^k-1)-k}.$$

Let $C \subseteq \mathbb{F}^n$ and $D \subseteq \mathbb{F}^m$ be codes. Then the *direct sum* of C and D is defined as $C \oplus D = \{(x, y) \mid x \in C, y \in D\}$. In the following theorem, it is shown that new self-locating-dominating codes can be formed from known ones using a direct sum.

Theorem 15. *Let $C \subseteq \mathbb{F}^n$ be a self-locating-dominating code. Then $D = C \oplus \mathbb{F}$ is also a self-locating-dominating code.*

Let us then concentrate on solid-locating-dominating codes. We will first give a lower bound such that its ratio to $2\frac{2^n}{n+1}$ approaches 1 as n tends to infinity. After that we will give an infinite sequence of solid-locating-dominating codes with the same limit. When we compare the sizes of optimal self-locating-dominating and solid-locating-dominating codes we see from Theorems 13 and 14 that optimal solid-locating-dominating codes are essentially smaller. In the following theorem, we first give a lower bound for solid-locating-dominating codes.

Theorem 16. *Let n be an integer such that $n \geq 5$. Then we have*

$$\gamma^{DLD}(n) \geq \left\lceil \left(1 + \frac{n-1}{n^2+n+2}\right) \frac{2^{n+1}}{n+1} \right\rceil.$$

In the following remark, we briefly compare the previously obtained lower bound to one for locating-dominating codes locating multiple irregularities.

Remark 17. In this paper, we have mainly studied locating-dominating codes which can locate one and detect multiple irregularities. Previously, in [8], so called $(1, \leq l)$-locating-dominating codes of type B ($(1, \leq l)$-LDB codes for short), which can locate multiple irregularities, have been studied. In [8, Theorem 5], the lower bound $\left\lceil \frac{2^{n+1}}{n+1} \right\rceil$ for $(1, \leq 2)$-LDB codes has been achieved. Since it can be shown that every $(1, \leq 2)$-LDB code is also a solid-locating-dominating code, our lower bound in Theorem 16 improves the lower bound for $(1, \leq 2)$-LDB codes in Hamming spaces.

Let us define the *covering radius* of C as $R(C) = \max_{x \in \mathbb{F}^n} \min_{c \in C}\{d(x,c) \mid x \in \mathbb{F}^n, c \in C\}$. When $n \geq 5$, the lower bound in Theorem 16 is attained by choosing as codewords all codewords and their neighbours of a code with covering radius two and minimum distance five. Unfortunately, codes like this are only known when $n = 5$ [2, Theorem 11.2.2]. Using this code, the following theorem is obtained.

Theorem 18. *We have* $\gamma^{DLD}(5) = 12$.

In general, solid-locating-dominating codes can be constructed from codes with covering radius two.

Theorem 19. *Let* $D \subseteq \mathbb{F}^n$ *be a code with* $R(D) = 2$. *Then*

$$C = \{c \in \mathbb{F}^n \mid c \in N[d], d \in D\}$$

is a solid-locating-dominating code.

In [2, Theorem 4.5.8], Struik has constructed an infinite sequence of codes with covering radius two such that we can build on top of it such a sequence of solid-locating-dominating codes that they converge to our lower bound.

Theorem 20. *There exists such a sequence of solid-locating-dominating codes* $(C_n)_{n=1}^{\infty}$ *that*

$$\lim_{n \to \infty} \frac{|C_n|}{2^{\frac{2^n}{n+1}}} = 1.$$

Using direct sum, we can construct new solid-locating-dominating codes from existing ones in a similar fashion as with self-locating-dominating codes.

Theorem 21. *Let* $C \subseteq \mathbb{F}^n$ *be a solid-locating-dominating code. Then* $D = C \oplus \mathbb{F}$ *is also a solid-locating-dominating code.*

Above, we have discussed self-locating-dominating and solid-locating-dominating codes in binary Hamming spaces. In what follows, we briefly consider regular locating-dominating codes. In particular, for certain lengths, we provide locating-dominating codes with the smallest known cardinalities. Previously, locating-dominating codes in $\mathbb{F}^n$ have been considered, for example, in [3, 8]. For future considerations, we first define the mapping $\pi : \mathbb{F}^n \to \mathbb{F}$ as follows:

$$\pi(u) = \begin{cases} 0, & \text{if } u \text{ has an even number of ones;} \\ 1, & \text{otherwise.} \end{cases}$$

Then, in the following theorem, we introduce a novel approach for constructing new locating-dominating codes based on known (suitable) identifying codes.

Theorem 22. *Let* C *be an identifying code in* $\mathbb{F}^n$ *such that* $|I(C;u)| \geq 2$ *for all* $u \in \mathbb{F}^n \setminus C$. *Then*

$$D = \{(\pi(u), u, u + c) \mid u \in \mathbb{F}^n, c \in C\}$$

is a locating-dominating code in $\mathbb{F}^{2n+1}$.

The best known upper bounds on $\gamma^{LD}(\mathbb{F}^n)$ for $1 \leq n \leq 10$ have been presented in [3, Table 3]. For lengths $n > 10$, the smallest known locating-dominating codes are actually identifying codes. (Recall that by the definitions any identifying code is also locating-dominating.) The currently best known upper bounds on $\gamma^{ID}(\mathbb{F}^n)$ can be found in [1]. In the following corollary, we present locating-dominating codes in $\mathbb{F}^n$ with the smallest known cardinalities for the lengths $n = 11$ and $n = 17$. These constructions significantly improve on the known upper bounds $\gamma^{LD}(\mathbb{F}^{11}) \leq \gamma^{ID}(\mathbb{F}^{11}) \leq 352$ and $\gamma^{LD}(\mathbb{F}^{17}) \leq \gamma^{ID}(\mathbb{F}^{17}) \leq 18558$.

Corollary 23. *We have* $\gamma^{LD}(\mathbb{F}^{11}) \leq 320$ *and* $\gamma^{LD}(\mathbb{F}^{17}) \leq 16384$.

With the help of the following theorem, which has been shown in [8, Theorem 7], we can construct new improved locating-dominating codes from codes attained in Corollary 23.

Theorem 24 ([8])**.** *If* $C \subseteq \mathbb{F}^n$ *is a locating-dominating code, then* $C \oplus \mathbb{F}$ *is also a locating-dominating code.*

The smallest currently known upper bounds for locating-dominating codes of lengths $n = 12$ and $n = 18$ are 684 and 35604 respectively [1].

Corollary 25. *We have* $\gamma^{LD}(\mathbb{F}^{12}) \leq 640$ *and* $\gamma^{LD}(\mathbb{F}^{18}) \leq 32768$.

In [8, Theorem 15], a lower bound, which is currently the best known, for $\gamma^{LD}(\mathbb{F}^n)$ has been presented. Applying the lower bound on the lengths $n = 11$, $n = 12$, $n = 17$ and $n = 18$, we obtain that $\gamma^{LD}(\mathbb{F}^{11}) \geq 309$, $\gamma^{LD}(\mathbb{F}^{12}) \geq 576$, $\gamma^{LD}(\mathbb{F}^{17}) \geq 13676$ and $\gamma^{LD}(\mathbb{F}^{18}) \geq 26006$. Thus, comparing the lower bounds to the constructions of the previous corollaries, we can state the obtained codes are rather small.

References

[1] I. Charon, G. Cohen, O. Hudry, and A. Lobstein. New identifying codes in the binary Hamming space. *European J. Combin.*, 31(2):491–501, 2010.

[2] G. Cohen, I. Honkala, S. Litsyn, and A. Lobstein. *Covering codes*, volume 54 of *North-Holland Mathematical Library*. North-Holland Publishing Co., Amsterdam, 1997.

[3] G. Exoo, V. Junnila, T. Laihonen, and S. Ranto. Locating vertices using codes. In *Proceedings of the Thirty-Ninth Southeastern International Conference on Combinatorics, Graph Theory and Computing*, volume 191, pages 143–159, 2008.

[4] N. Fazlollahi, D. Starobinski, and A. Trachtenberg. Connected identifying codes. *IEEE Trans. Inform. Theory*, 58(7):4814–4824, 2012.

[5] W. Goddard and K. Wash. ID codes in Cartesian products of cliques. *J. Combin. Math. Combin. Comput.*, 85:97–106, 2013.

[6] S. Gravier, J. Moncel, and A. Semri. Identifying codes of Cartesian product of two cliques of the same size. *Electron. J. Combin.*, 15(1):Note 4, 7, 2008.

[7] I. Honkala and T. Laihonen. On a new class of identifying codes in graphs. *Inform. Process. Lett.*, 102(2-3):92–98, 2007.

[8] I. Honkala, T. Laihonen, and S. Ranto. On locating-dominating codes in binary Hamming spaces. *Discrete Math. Theor. Comput. Sci.*, 6(2):265–281, 2004.

[9] V. Junnila and T. Laihonen. Collection of codes for tolerant location. In *Proceedings of the Bordeaux Graph Workshop*, pages 176–179, 2016.

[10] V. Junnila and T. Laihonen. Tolerant location detection in sensor networks. Submitted, 2016.

[11] M. G. Karpovsky, K. Chakrabarty, and L. B. Levitin. On a new class of codes for identifying vertices in graphs. *IEEE Trans. Inform. Theory*, 44(2):599–611, 1998.

[12] M. Laifenfeld and A. Trachtenberg. Disjoint identifying-codes for arbitrary graphs. In *Proceedings of International Symposium on Information Theory, 2005. ISIT 2005*, pages 244–248, 2005.

[13] A. Lobstein. Watching systems, identifying, locating-dominating and discrminating codes in graphs, a bibliography. Published electronically at `http://perso.enst.fr/~lobstein/debutBIBidetlocdom.pdf`.

[14] D. F. Rall and P. J. Slater. On location-domination numbers for certain classes of graphs. *Congr. Numer.*, 45:97–106, 1984.

[15] S. Ray, D. Starobinski, A. Trachtenberg, and R. Ungrangsi. Robust location detection with sensor networks. *IEEE Journal on Selected Areas in Communications*, 22(6):1016–1025, August 2004.

[16] P. J. Slater. Domination and location in acyclic graphs. *Networks*, 17(1):55–64, 1987.

[17] P. J. Slater. Dominating and reference sets in a graph. *J. Math. Phys. Sci.*, 22:445–455, 1988.

Location in circulant graphs

Ville Junnila[1] *Tero Laihonen*[1] *Gabrielle Paris*[2*]

[1] Department of Mathematics and Statistics, University of Turku, FI-20014 Turku, Finland
viljun@utu.fi and terolai@utu.fi

[2] LIRIS, University of Lyon, France
claudia.paris-sierra@univ-lyon1.fr

1 Introduction

Let $G = (V, E)$ be a simple, undirected graph with the vertex set V and the edge set E. Let $N[u]$ denote the *closed neighbourhood* of a vertex $u \in V$. A nonempty subset $C \subseteq V$ is called a *code*, and its elements are called *codewords*. The *I-set* (or the *identifier*) of u is defined as

$$I(u) = I(G, C; u) = N[u] \cap C.$$

Let C be a code in G. A vertex $u \in V$ is *covered* or *dominated* by a codeword of C if the identifying set $I(u)$ is nonempty. The code C is *dominating* in G if all the vertices of V is covered by a codeword of C, i.e., $|I(u)| \geq 1$ for all $u \in V$. The code C is *identifying* in G if C is dominating and for all distinct $u, v \in V$ we have

$$I(u) \neq I(v).$$

The definition of identifying codes is due to Karpovsky *et al.* [14], and the original motivation for studying such codes comes from fault diagnosis in multiprocessor systems. The concept of locating-dominating codes is closely related to the one of identifying codes. We say that the code is *locating-dominating* in G if C is dominating and for all distinct $u, v \in V \setminus C$ we have $I(u) \neq I(v)$. The definition of locating-dominating codes was introduced by Slater [17, 19, 20]. For the extensive literature on these codes consult [15]. A code $C \subseteq V$ is *self-identifying* in $G = (V, E)$ if for all vertices $u \in V$ we have

$$\bigcap_{c \in I(u)} N[c] = \{u\}.$$

Self-identifying codes are discussed in [9, 11, 12]. An identifying, locating-dominating or self-identifying code with the smallest cardinality in a given finite graph G is called *optimal*. The number of codewords in an optimal identifying and locating-dominating code in a finite graph G is denoted by $\gamma^{ID}(G)$, $\gamma^{LD}(G)$ and $\gamma^{SID}(G)$, respectively.

In this paper, we focus on studying these codes in so called circulant graphs. Let n and $d_1, \ldots, d_k$ be positive integers such that for all i, $d_i \leq \frac{n}{2}$. The circulant graph $C_n(d_1, d_2, \ldots, d_k)$ is defined as follows: the vertex set is $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$ and the neighbourhood of a vertex $u \in \mathbb{Z}_n$ is

$$N[u] = \{u, u \pm d_1, u \pm d_2, \ldots, u \pm d_k\},$$

where the calculations are done modulo n. Previously, in [1, 3, 5, 7, 10, 16, 18, 21], identifying and locating-dominating codes have been studied in the circulant graphs $C_n(1, 2, \ldots, r)$ ($r \in \mathbb{Z}, r \geq 1$), which can also be viewed as power graphs of cycles of length

*Supported by the ANR-14-CE25-0006 project of the French National Research Agency

n. Recently, in [6], Ghebleh and Niepel studied identification and location-domination in $C_n(1,3)$. They obtained the following results:

$$\lceil 4n/11 \rceil \leq \gamma^{ID}(C_n(1,3)) \leq \lceil 4n/11 \rceil + 1 \text{ and } \lceil n/3 \rceil \leq \gamma^{LD}(C_n(1,3)) \leq \lceil n/3 \rceil + 1.$$

In this paper, we give lower bounds on values of $\gamma^{LD}(G)$ and $\gamma^{SID}(G)$ for the circulant graphs $C_n(1, d-1, d, d+1)$ with $d > 3$ and give code constructions attaining these bounds for infinitely many values of parameters n and d. Moreover, we a give lower bound on $\gamma^{ID}(C_n(1, d-1, d, d+1))$ and we construct a family of identifying codes whose cardinality approaches the lower bound as n tends to infinity. In addition, we give optimal values for self-identifying codes in the circulant graph $C_n(1, n/2)$ when n is even.

2 Lower bounds using an infinite grid

Let us first present the definition of the *infinite king grid* $\mathcal{K}$. Its vertex set is $V = \mathbb{Z}^2$. The edges of the infinite king grid $\mathcal{K}$ are defined in such a way that the closed neighbourhood of $u = (x, y) \in \mathbb{Z}^2$ is

$$N[u] = \{(x', y') \in \mathbb{Z}^2 \mid \max\{|x - x'|, |y - y'|\} \leq 1\}.$$

For estimating the sizes of infinite codes, we need a way to measure them in the grid. For this purpose, we first denote

$$Q_m = \{(x, y) \in \mathbb{Z}^2 \mid |x| \leq m, |y| \leq m\},$$

where m is a positive integer. The *density* of a code $C \subseteq \mathbb{Z}^2$ is then defined as

$$D(C) = \limsup_{m \to \infty} \frac{|C \cap Q_m|}{|Q_m|}.$$

For a finite nonempty set $S \subseteq V$ in a graph $G = (V, E)$, the (local) *density* of a code $C \subseteq V$ in S is defined as $|S \cap C|/|S|$.

Analogously to finite graphs, an identifying, locating-dominating and self-identifying code with the smallest possible density in the infinite king grid is called *optimal*. The densities of optimal codes have been intensively studied and all the exact values are known. The optimal densities can be found in Table 1 together with the references to the papers, where the results have been presented.

	density	reference
ID	2/9	[2, 4]
LD	1/5	[8]
self-ID	1/3	[9]

Table 1: The densities of optimal identifying (ID), locating-dominating (LD) and self-identifying (self-ID) codes in the king grid $\mathcal{K}$.

In the following theorem, we present the connection between identifying, locating-dominating and self-identifying codes in the infinite king grid and the circulant graphs $C_n(1, d-1, d, d+1)$.

Theorem 1. *Let n, d and k be positive integers such that $d \geq 3$. If C is an identifying code in $C_n(1, d-1, d, d+1)$ with k codewords, then there exists an identifying code in the infinite king grid $\mathcal{K}$ with density k/n. Analogous results also hold for locating-dominating and self-identifying codes.*

In the following corollary, we give lower bounds for codes in the circulant graphs $C_n(1, d-1, d, d+1)$. In Section 3, we show that the lower bounds can be attained with locating-dominating and self-identifying codes and that there exists an infinite family of identifying codes approaching the lower bound.

Corollary 2. *Let n and d be positive integers such that $d \geq 3$ and $G = C_n(1, d-1, d, d+1)$. Then we have*

$$\gamma^{LD}(G) \geq \left\lceil \frac{n}{5} \right\rceil, \ \gamma^{ID}(G) \geq \left\lceil \frac{2n}{9} \right\rceil \text{ and } \gamma^{SID}(G) \geq \left\lceil \frac{n}{3} \right\rceil.$$

3 Optimal constructions

In the following theorem, we give optimal locating-dominating codes in the circulant graph $C_n(1, d-1, d, d+1)$. Furthermore, we give an infinite sequence of identifying codes approaching the lower bound in Corollary 2.

Theorem 3. *(i) For $d \equiv 8 \pmod{10}$, $d \geq 8$, $n \geq 4d+6$ and $n \equiv 0 \pmod{10}$, we have*

$$\gamma^{LD}(C_n(1, d-1, d, d+1)) = \frac{n}{5}.$$

(ii) There is a sequence of identifying codes $(C_k)_{k=1}^{\infty}$ in the circulant graphs $C_n(1, d-1, d, d+1)$ with

$$\lim_{k \to \infty} \frac{|C_k|}{n} = \frac{2}{9}.$$

In the next theorem, we will show that the bound on self-identifying codes in Corollary 2 can be reached.

Theorem 4. *If $d \equiv 1 \pmod{3}$, $n \geq 3d+5$ and $n \equiv 0 \pmod{3}$, then*

$$\gamma^{SID}(C_n(1, d-1, d, d+1)) = \frac{n}{3}.$$

In the following theorem, we give optimal self-identifying codes for $C_n(1, n/2)$ for n even.

Proposition 5. *Let $k \geq 5$. The optimal cardinality of self-identifying code in $C_{2k}(1, k)$ is as follows:*

$$\gamma^{SID}(C_{2k}(1,k)) = \begin{cases} \left\lceil 4\frac{k}{3} \right\rceil & \text{if } k \equiv 0 \pmod 3 \text{ or } k \equiv 1 \pmod 3 \\ \left\lceil 4\frac{k}{3} \right\rceil + 1 & \text{if } k \equiv 2 \pmod 3 \end{cases}.$$

It is natural to study also other circulant graphs. For optimal codes in circulant graphs $C_n(1, d)$ and $C_n(1, d-1, d)$, see [13].

References

[1] N. Bertrand, I. Charon, O. Hudry, and A. Lobstein. Identifying and locating-dominating codes on chains and cycles. *European J. Combin.*, 25(7):969–987, 2004.

[2] I. Charon, O. Hudry, and A. Lobstein. Identifying codes with small radius in some infinite regular graphs. *Electron. J. Combin.*, 9(1):Research Paper 11, 25 pp., 2002.

[3] C. Chen, C. Lu, and Z. Miao. Identifying codes and locating-dominating sets on paths and cycles. *Discrete Appl. Math.*, 159(15):1540–1547, 2011.

[4] G. Cohen, I. Honkala, A. Lobstein, and G. Zémor. On codes identifying vertices in the two-dimensional square lattice with diagonals. *IEEE Trans. Comput.*, 50(2):174–176, 2001.

[5] G. Exoo, V. Junnila, and T. Laihonen. Locating-dominating codes in cycles. *Australas. J. Combin.*, 49:177–194, 2011.

[6] M. Ghebleh and L. Niepel. Locating and identifying codes in circulant networks. *Discrete Appl. Math.*, 161(13-14):2001–2007, 2013.

[7] S. Gravier, J. Moncel, and A. Semri. Identifying codes of cycles. *European J. Combin.*, 27(5):767–776, 2006.

[8] I. Honkala and T. Laihonen. On locating-dominating sets in infinite grids. *European J. Combin.*, 27(2):218–227, 2006.

[9] I. Honkala and T. Laihonen. On a new class of identifying codes in graphs. *Inform. Process. Lett.*, 102(2-3):92–98, 2007.

[10] V. Junnila and T. Laihonen. Optimal identifying codes in cycles and paths. *Graphs Combin.*, 28(4):469–481, 2012.

[11] V. Junnila and T. Laihonen. Collection of codes for tolerant location. In *Proceedings of the Bordeaux Graph Workshop*, pages 176–179, 2016.

[12] V. Junnila and T. Laihonen. Tolerant location detection in sensor networks. Submitted, 2016.

[13] V. Junnila, T. Laihonen, and G. Paris. Optimal bounds on codes for location in circulant graphs. Submitted, 2017.

[14] M. G. Karpovsky, K. Chakrabarty, and L. B. Levitin. On a new class of codes for identifying vertices in graphs. *IEEE Trans. Inform. Theory*, 44(2):599–611, 1998.

[15] A. Lobstein. Watching systems, identifying, locating-dominating and discrminating codes in graphs, a bibliography. Published electronically at `http://perso.enst.fr/~lobstein/debutBIBidetlocdom.pdf`.

[16] P. Manuel. Locating and liar domination of circulant networks. *Ars Combin.*, 101:309–320, 2011.

[17] D. F. Rall and P. J. Slater. On location-domination numbers for certain classes of graphs. *Congr. Numer.*, 45:97–106, 1984.

[18] D. L. Roberts and F. S. Roberts. Locating sensors in paths and cycles: The case of 2-identifying codes. *European J. Combin.*, 29(1):72–82, 2008.

[19] P. J. Slater. Domination and location in acyclic graphs. *Networks*, 17(1):55–64, 1987.

[20] P. J. Slater. Dominating and reference sets in a graph. *J. Math. Phys. Sci.*, 22:445–455, 1988.

[21] M. Xu, K. Thulasiraman, and X.-D. Hu. Identifying codes of cycles with odd orders. *European J. Combin.*, 29(7):1717–1720, 2008.

Connected and contractible sets of vertices of a 3-connected graph

Dmitri Karpov

Basic definitions

Before introducing our results let us recall main definitions that we need. We consider undirected graphs without loops and multiple edges and use standard notation.

For a graph G we denote the set of its vertices by $V(G)$ and the set of its edges by $E(G)$. We use notation $v(G)$ for the number of vertices of G.

We denote the *degree* of a vertex x in the graph G by $d_G(x)$.

Let $\mathrm{N}_G(w)$ denote the *neighbourhood* of a vertex $w \in V(G)$ (i.e. the set of all vertices of the graph G, adjacent to w).

For a set of vertices $U \subset V(G)$ we denote by $G(U)$ the *induced subgraph* of the graph G on the set U.

A set $U \subset V(G)$ is *connected*, if the graph $G(U)$ is connected.

Definition 1. 1) Let $R \subset V(G)$. We denote by $G - R$ the graph obtained from G upon deleting all vertices of the set R and all edges incident to vertices of R. The set R is a *cutset*, if the graph $G - R$ is disconnected.

2) A graph G is *k-connected*, if $|V(G)| > k$ and G has no cutset of size less than k.

Introduction and main results

Consider a 2-connected graph G on n vertices, let n_1, n_2 be positive integers with $n_1 + n_2 = n$. It is rather clear, that there exists a decomposition of the vertex set of G into two disjoint connected sets V_1 and V_2, such that $|V_1| = n_1$, $|V_2| = n_2$. (The proof of this fact is an easy exercise on blocks and a kind of mathematical folklore.)

E. Győri [1] in 1976, and L. Lovász [2] in 1977 have independently proved the following beautiful and strong theorem, that generalizes the above fact to $k \geq 2$.

Theorem 2. (E. Győri, L. Lovász.) *Let G be a k-connected graph on n vertices, $v_1, \ldots, v_k \in V(G)$ and positive integers $n_1, \ldots, n_k$ be such that $\sum_{i=1}^{k} n_i = n$. Then there exists a decomposition of the vertex set of G into connected sets $V_1 \ni v_1, \ldots, V_k \ni v_k$, such that $|V_i| = n_i$ for all $i \in \{1, 2, \ldots, k\}$.*

Does there exist a natural condition on graphs of connectivity less than k, that provides a decomposition of its vertex set into k connected subsets of any given sizes?

Let's begin with a counterexample showing that there exist $(k-1)$-connected graphs which cannot have such decompositions for all admissible sizes of sets.

Consider a graph G having a cutset T of size $k-1$. Let $G-T$ has connected components $W_1, \ldots, W_m$, where $m \geq k+1$. Let's try to divide G into disjoint connected sets $V_1, \ldots, V_k$. Only $k-1$ sets of $V_1, \ldots, V_k$ can contain vertices of T. Hence, at least one set V_i is a subset of a component W_j. Let all components have the same size $p \geq 2$. The G cannot be divided into k connected sets of sizes more than p. However, the size

$$v(G) = mp + (k-1) > k(p+1)$$

can be presented as a sum of k integers which are at least $p+1$.

For connected graphs, we have a classic but strong instrument to study graph's structure — blocks and cutpoints. Let's recall the definitions.

Definition 3. Let G be a connected graph.

A vertex $a \in V(G)$ is a *cutpoint* of G, if the graph $G - a$ is disconnected.

A *block* of the graph G is a maximal up to inclusion subgraph, having no cutpoints.

The *interior* $\mathrm{Int}(B)$ of a block B is the set of all its vertices which are not cutpoints of G.

The structure of mutual disposition of blocks and cutpoints of a connected graph can be described by the *tree of blocks and cutpoints* (see [8]). Recall, that the tree of blocks and cutpoints of a graph G is a bipartite graph. Vertices of the first partition are all cutpoints $a_1, \ldots, a_n$ of the graph G, vertices of the second partition are all blocks $B_1, \ldots, B_m$ of the graph G. Vertices a_i and B_j are adjacent if and only if $a_i \in V(B_j)$. It is easy to prove, that this graph is a tree, all leaves of which correspond to blocks (which are called *pendant blocks*).

A trivial consequence of Győri-Lovász Theorem for 2-connected graphs is the following.

Corollary 4. *Let G be a 2-connected graph on n vertices, having exactly two pendant blocks B_1 and B_2. Let $v_1 \in \mathrm{Int}(B_1)$, $v_2 \in \mathrm{Int}(B_2)$ and positive integers n_1 and n_2 be such that $n_1 + n_2 = n$. Then there exists a decomposition of the vertex set of G into two connected sets $V_1 \ni v_1$ and $V_2 \ni v_2$, such that $|V_1| = n_1$, $|V_2| = n_2$.*

It is rather surprising, that for decomposition into 3 connected sets we can prove a more strong and natural result. The condition which provides the existence of decomposition naturally follows from the above counterexample.

Theorem 5. (D. Karpov, 2017.) *Let G be a 2-connected graph on n vertices, such that any its 2-vertex cutset splits G into at most 3 connected components. Let positive integers n_1, n_2 and n_3 be such that $n_1 + n_2 + n_3 = n$. Then there exists a decomposition of the vertex set of G into connected sets V_1, V_2 and V_3, such that $|V_i| = n_i$ for all $i \in \{1, 2, 3\}$.*

Definition 6. A graph G is a subdivision of a graph 0,0,1H, if G can be obtained from H by several operations of substituting an edge by a simple path, which interior consists of *new* vertices. New vertices are distinct and do not belong to $V(H)$ (see figure).

Figure 1: A graph H and its subdivision G

Subdivision of a graph is a construction often used in graph theory. The following statement is a particular case of Theorem 5.

Corollary 7. *Let G be a subdivision of a 3-connected graph and positive integers n_1, n_2, n_3 be such that $n_1 + n_2 + n_3 = v(G)$. Then there exists a decomposition of $V(G)$ into connected sets V_1, V_2, V_3, such that $|V_i| = n_i$ for all $i \in \{1, 2, 3\}$.*

Let's formulate as a conjecture a generalization of Theorem 5 for graphs of greater connectivity.

Conjecture 8. *Let G be a $(k-1)$–connected graph on n vertices, such that any its $(k-1)$-vertex cutset splits G into at most k connected components. Let positive integers $n_1, \ldots, n_k$ be such that $\sum_{i=1}^{k} n_i = n$. Then there exists a decomposition of the vertex set of G into connected sets $V_1, \ldots, V_k$, such that $|V_i| = n_i$ for all $i \in \{1, 2, \ldots, k\}$.*

Let's recall one more conjecture on decomposition of the vertex set of a 3-connected graph. It was stated in 1994 by McCuaig and Ota [6] and mentioned in Mader's survey on connectivity [4].

Definition 9. Let G be a 3-connected graph. A set $W \subset V(G)$ is *contractible*, if the graph $G(W)$ is connected and the graph $G - W$ is 2-connected.

Conjecture 10. *Let $m \in \mathrm{N}$. Then there exists an integer n such that every 3-connected graph G on at least n vertices has a contractible set of m vertices.*

For $m = 1$ this statement is clear, for $m = 2$ it is rather easy and well-known (it was proved by Tutte). The case $m = 3$ was proved by authors of this conjecture [6], the case $m = 4$ was proved by M. Kriesell [7]. For any $m \geq 5$ Conjecture 10 is neither proved nor disproved now.

Our next theorem suggests a new statement on large contractible sets in 3-connected graphs.

Theorem 11. (D. Karpov, 2017.) *Let $m \geq 5$ be a positive integer and G be a 3-connected graph on at least $2m + 1$ vertices. Then G has a contractible set W, such that $m \leq |W| \leq 2m - 4$.*

A particular case of this theorem for $m = 4$ is the following.

Corollary 12. *A 3-connected graph on $n \geq 11$ vertices has a contractible set of 5 or 6 vertices.*

Note, that the analog of conjecture 10 for graphs of greater connectivity fails. It was proved by Mader [5], that for $k \geq 4$ there exist k-connected graphs of any size having no contractible set of given size m. Moreover, one can construct a k-connected graph having no set W of size k (not necessary connected!) such that the graph $G - W$ is $(k-1)$-connected.

References

[1] E. Győri. *On division of graphs to connected subgraphs.* Colloq. Math. Soc. Janos Bolyai, 18. Combinatorics, Keszthely (Hungary) (1976), 485-494.

[2] L. Lovász. *A homology theory for spanning trees of a graph.* Acta Math. Acad. Sci. Hungaricae 30 (1977), 241-251.

[3] W. T. Tutte. *Connectivity in graphs.* Toronto, Univ. Toronto Press, 1966.

[4] W. Mader. *On vertices of degree n in minimally n-connected graphs and digraphs.* Combinatorics, Paul Erdös is Eighty, Budapest, 1996. Vol.2, p.423- 449.

[5] W. Mader. *High connectivity keeping sets in n-connected graphs.* Combinatorica, Vol. 24 (3) (2004) 441-458.

[6] W. McCuaig, K. Ota. *Contractible triples in 3-connected graphs.* J.Comb.Theory Ser.B, Vol.60, 1994, p.308-314.

[7] M. Kriesell. *Contractible subgraphs in 3-connected graphs.* J.Comb.Theory Ser.B, Vol.80, 2000, p.32-48.

[8] F. Harary, *Graph theory,* Addison-Wesley, 1969.

[9] D.V.Karpov, A.V.Pastor. *The structure of a decomposition of a triconnected graph.* Zapiski Nauchnykh Semenarov POMI, **391** (2011), p. 90-148, in Russian. English translation: Journal of Mathematical Sciences, **184**, issue 5 (2012), p. 601-628.

[10] D. V. Karpov. *The Decomposition Tree of a Biconnected Graph.* Zapiski Nauchnykh Semenarov POMI, **417**, (2013), p. 86-105, in Russian. English translation: Journal of Mathematical Sciences, **204**, Issue 2 (2015), pp 232-243.

[11] D. V. Karpov. *Minimal biconnected graphs.* Zapiski Nauchnykh Semenarov POMI, **417** (2013), pp. 106-127, in Russian. English translation: Journal of Mathematical Sciences, **204**, issue 2 (2015), p. 244-257.

Width Hierarchies for Quantum and Classical Ordered Binary Decision Diagrams with Repeated Test

Kamil Khadiev[1,2,*] *Rishat Ibrahimov*[2]

[1] University of Latvia, Riga, Latvia
[2] Kazan Federal University, Kazan, Russia
kamilhadi@gmail.com, rishat.ibrahimov@yandex.ru

Abstract

We consider quantum, nondterministic and probabilistic versions of known computational model Ordered Read-k-times Branching Programs or Ordered Binary Decision Diagrams with repeated test (k-QOBDD, k-NOBDD and k-POBDD). We show width hierarchy for complexity classes of Boolean function computed by these models and discuss relation between different variants of k-OBDD.

Keywords: quantum computing, quantum OBDD, OBDD, Branching programs, quantum vs classical, quantum models, hierarchy, computational complexity, probabilistic OBDD, nondeterministic OBDD

1 Introduction

Branching programs are one of the well known models of computation. These models have been shown useful in a variety of domains, such as hardware verification, model checking, and other CAD applications (see for example the book by I. Wegener [Weg00]). It is known that the class of Boolean functions computed by polynomial size branching programs coincide with the class of functions computed by non-uniform log-space machines.

One of important restrictive branching programs are oblivious read once branching programs, also known as Ordered Binary Decision Diagrams (OBDD) [Weg00]. It is a good model of data streaming algorithms. These algorithms are actively used in industry, because of rapidly increasing of size of data which should be processed by programs. Since a length of an OBDD is at most linear (in the length of the input), the main complexity measure is "width", analog of size for automata. And it can be seen as nonuniform automata (see for example [AG05]).

In the last decades quantum model of OBDD came into play [AGK01], [NHK00], [SS05], [Sau06]. The topic of read-k-times quantum model of OBDD (k-QOBDD) has been under a lot of interest lately. k-QOBDD can be explored from automata point of view. And in that case we can found good algorithms for two way quantum classical automata in paper [AW02] of Ambainis and Watrous. Other automata models, that have relation with k-QOBDD are restart and reset quantum automata [YS10].

One of the interesting questions, which has been explored is hierarchy of complexity classes for classical and quantum k-OBDDs. These models have two main characteristics of complexity: number of layers (k) and width. Hierarchy for numbers of layers was investigated in papers [BSSW98], [Kha16], [KK], [AKK17]

On the other hand, there are only few work on width hierarchy. For example, width hierarchy for deterministic k-OBDD is presented in [Kha15]. Width hierarchies for classical and quantum 1-OBDDs are discussed in [AGKY14], [AGKY16], [KK].

*Partially supported by ERC Advanced Grant MQC. The work is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University

In this paper we prove width hierarchy for nondeterministic, probabilistic k-OBDD and quantum k-OBDD with natural order of input bits. We considered sub linear width and hierarchies based on results and lower bounds from [AK13], [Kha16], [AKK17].

The paper is organized in following way. In Section 2 we present definitions. Section 3 contains Hierarchy results for classical models and Section 4 for quantum one. We discuss relation between different models in Section 5.

2 Preliminaries

Ordered read ones Branching Programs (OBDD) are well known model for Boolean functions computation. As general source different models of branching programs we give the book by I. Wegener [Weg00].

A branching program over a set X of n Boolean variables is a directed acyclic graph with two distinguished nodes s (a source node) and t (a sink node). We denote such program $P_{s,t}$ or just P. Each inner node v of P is associated with a variable $x \in X$. *Deterministic* P has exactly two outgoing edges labeled $x = 0$ and $x = 1$ respectively for such node v.

The program P computes the Boolean function $f(X)$ ($f : \{0,1\}^n \to \{0,1\}$) as follows: for each $\sigma \in \{0,1\}^n$ we let $f(\sigma) = 1$ if and only if there exists at least one $s - t$ path (called *accepting* path for σ) such that all edges along this path are consistent with σ.

A branching program is *leveled* if the nodes can be partitioned into levels $V_1, \ldots, V_\ell$ and a level $V_{\ell+1}$ such that the nodes in $V_{\ell+1}$ are the sink nodes, nodes in each level V_j with $j \leq \ell$ have outgoing edges only to nodes in the next level V_{j+1}. For a leveled $P_{s,t}$ the source node s is a node from the first level V_1 of nodes and the sink node t is a node from the last level $V_{\ell+1}$.

The *width* $w(P)$ of a leveled branching program P is the maximum of number of nodes in levels of P, i.e., $w(P) = \max_{1 \leq j \leq \ell} |V_j|$. The *size* of branching program P is a number of nodes of program P.

A leveled branching program is called *oblivious* if all inner nodes of one level are labeled by the same variable. A branching program is called *read once* if each variable is tested on each path only once. An oblivious leveled read once branching program is also called Ordered Binary Decision Diagram (OBDD). OBDD P reads variables in its individual order $\pi = (j_1, \ldots, j_n)$, $\pi(i) = j_i$, $\pi^{-1}(j)$ is position of j in permutation π. We call $\pi(P)$ the order of P. Let us denote natural order as $id = (1, \ldots, n)$. Sometimes we will use notation id-OBDD P, it means that $\pi(P) = id$. Let $width(f) = \min_P w(P)$ for OBDD P which computes f and $id{-}width(f)$ is the same but for id-OBDD.

The Branching program P is called k-OBDD if it consists from k layers, where i-th ($1 \leq i \leq k$) layer P^i of P is an OBDD. Let π_i be an order of P^i, $1 \leq i \leq k$ and $\pi_1 = \cdots = \pi_k = \pi$. We call order $\pi(P) = \pi$ the order of P.

Nondeterministic OBDD (NOBDD) is nondeterministic counterpart of OBDD. Probabilistic OBDD (POBDD) can have more than two edges for node, and choose one of them using probabilistic mechanism. POBDD P computes Boolean function f with bounded error $0.5 - \varepsilon$ if probability of right answer is at least $0.5 + \varepsilon$.

Let us discuss a definition of quantum OBDD (QOBDD). It is given in different terms, but it is straightforward to see that it is equivalent, see [AGK+05], [AGK01] for more details.

For a given $n > 0$, a quantum OBDD P of width w, defined on $\{0,1\}^n$, is a 4-tuple $P = (T, |\psi\rangle_0, Accept, \pi)$, where

- $T = \{T_j : 1 \leq j \leq n \text{ and } T_j = (G_j^0, G_j^1)\}$ are ordered pairs of (left) unitary matrices

representing the transitions applied at the j-th step, where G_j^0 or G_j^1, determined by the corresponding input bit.

- $|\psi\rangle_0$ is initial vector from w-dimensional Hilbert space over field of complex numbers. $|\psi\rangle_0 = |q_0\rangle$ where q_0 corresponds to the initial node.
- $Accept \subset \{1, \ldots, w\}$ is accepting nodes.
- π is a permutation of $\{1, \ldots, n\}$ defining the order of testing the input bits.

For any given input $\sigma \in \{0,1\}^n$, the computation of P on σ can be traced by a vector from w-dimensional Hilbert space over field of complex numbers. The initial one is $|\psi\rangle_0$. In each step j, $1 \leq j \leq n$, the input bit $x_{\pi(j)}$ is tested and then the corresponding unitary operator is applied: $|\psi\rangle_j = G_j^{x_{\pi(j)}}(|\psi\rangle_{j-1})$, where $|\psi\rangle_{j-1}$ and $|\psi\rangle_j$ represent the state of the system after the $(j-1)$-th and j-th steps, respectively, where $1 \leq j \leq n$.

In the end of computation program P measure qubits. The accepting (return 1) probability $Pr_{accept}(\sigma)$ of P_n on input σ is $Pr_{accept}(\nu) = \sum_{i \in Accept} v_i^2.$, for $|\psi\rangle_n = (v_1, \ldots, v_w)$. We say that a function f is computed by P with bounded error if there exists an $\varepsilon \in (0, \frac{1}{2}]$ such that P accepts all inputs from $f^{-1}(1)$ with a probability at least $\frac{1}{2} + \varepsilon$ and P_n accepts all inputs from $f^{-1}(0)$ with a probability at most $\frac{1}{2} - \varepsilon$. We can say that error probability is $\frac{1}{2} - \varepsilon$.

Let k-$\mathbf{QOBDD}_w$ be a set of Boolean functions which can be computed by bounded error k-QOBDDs of width w. k-id-$\mathbf{QOBDD}_w$ is same for bounded error k-QOBDDs with order $id = (1, \ldots, n)$. k-$\mathbf{NOBDD}_w$ and k-$\mathbf{POBDD}_w$ is similar classes for k-NOBDD and bounded error k-POBDD.

3 Width Hierarchies on Classical k-OBDD

Firstly, let us discuss required definitions.

Let $\pi = (X_A, X_B)$ be a partition of the set X into two parts X_A and $X_B = X \backslash X_A$. Below we will use equivalent notations $f(X)$ and $f(X_A, X_B)$. Let $f|_\rho$ be a subfunction of f, where ρ is mapping $\rho : X_A \to \{0,1\}^{|X_A|}$. Function $f|_\rho$ is obtained from f by applying ρ. Let $N^\pi(f)$ be number of different subfunctions with respect to partition π. Let $\Theta(n)$ be the set of all permutations of $\{1, \ldots, n\}$. Let partition $\pi(\theta, u) = (X_A, X_B) = (\{x_{j_1}, \ldots, x_{j_u}\}, \{x_{j_{u+1}}, \ldots, x_{j_n}\})$, for permutation $\theta = (j_1, \ldots, j_n) \in \Theta(n), 1 < u < n$. We denote $\Pi(\theta) = \{\pi(\theta, u) : 1 < u < n\}$. Let $N^\theta(f) = \max_{\pi \in \Pi(\theta)} N^\pi(f), N(f) = \min_{\theta \in \Theta(n)} N^\theta(f)$.

Secondly, let us present existing lower bounds for nondeterministic and probabilistic k-OBDDs.

Lemma 1 ([AK13]). *Let function $f(X)$ is computed by k-OBDD P of width w, then $N(f) \leq w^{(k-1)w+1}$.*

Lemma 2 ([Kha16]). *Let function $f(X)$ is computed by k-NOBDD P of width w, then $N(f) \leq 2^{w\left((k-1)w+1\right)}$.*

Lemma 3 ([Kha16]). *Let function $f(X)$ be computed by bounded error k-POBDD P of width w, then*

$$N(f) \leq (C_1 k(C_2 + \log_2 w + \log_2 k))^{(k+1)w^2}$$

for some constants C_1 and C_2.

Thirdly, let us define *Shuffled Address Function* ($SAF_{k,w}$) from [Kha15] based on definition of well known Pointer Jumping Function, see [BSSW98], [NW91].

Definition 4 (Shuffled Address Function)**.** Let us define Boolean function $SAF_{k,w}(X) : \{0,1\}^n \to \{0,1\}$ for integer $k = k(n)$ and $w = w(n)$ such that

$$2kw(2w + \lceil \log k \rceil + \lceil \log 2w \rceil) < n. \tag{1}$$

We divide input variables to $2kw$ blocks. There are $\lceil n/(2kw) \rceil = a$ variables in each block. After that we divide each block to *address* and *value* variables. First $\lceil \log k \rceil + \lceil \log 2w \rceil$ variables of block are *address* and other $a - \lceil \log k \rceil + \lceil \log 2w \rceil = b$ variables of block are *value*.

We call $x_0^p, \ldots, x_{b-1}^p$ *value* variables of p-th block and $y_0^p, \ldots, y_{\lceil \log k \rceil + \lceil \log 2w \rceil}^p$ are *address* variables, for $p \in \{0, \ldots, 2kw - 1\}$.

Boolean function $SAF_{k,w}(X)$ is iterative process based on definition of following six functions:

Function $AdrK : \{0,1\}^n \times \{0, \ldots, 2kw - 1\} \to \{0, \ldots, k - 1\}$ obtains firsts part of block's address. This block will be used only in step of iteration which number is computed using this function:

$$AdrK(X, p) = \sum_{j=0}^{\lceil \log k \rceil - 1} y_j^p \cdot 2^j (mod\ k).$$

Function $AdrW : \{0,1\}^n \times \{0, \ldots, 2kw - 1\} \to \{0, \ldots, 2w - 1\}$ obtains second part of block's address. It is the address of block within one step of iteration:

$$AdrW(X, p) = \sum_{j=0}^{\lceil \log 2w \rceil - 1} y_{j+\lceil \log k \rceil}^p \cdot 2^j (mod\ 2w).$$

Function $Ind : \{0,1\}^n \times \{0, \ldots, 2w - 1\} \times \{0, \ldots, k - 1\} \to \{0, \ldots, 2kw - 1\}$ obtains number of block by number of step and address within this step of iteration:

$$Ind(X, i, t) = \begin{cases} p, & \text{where } p \text{ is minimal number of block such that} \\ & AdrK(X, p) = t \text{ and } AdrW(X, p) = i, \\ -1, & \text{if there are no such } p. \end{cases}$$

Function $Val : \{0,1\}^n \times \{0, \ldots, 2w - 1\} \times \{1, \ldots, k\} \to \{-1, \ldots, w - 1\}$ obtains value of block which have address i within t-th step of iteration:

$$Val(X, i, t) = \begin{cases} \sum_{j=0}^{b-1} x_j^p (mod\ w), & \text{where } p = Ind(X, i, t), \text{ for } p \geq 0, \\ -1, & \text{if } Ind(X, i, t) < 0. \end{cases}$$

Two functions $Step_1$ and $Step_2$ obtain value of t-th step of iteration. Function $Step_1 : \{0,1\}^n \times \{0, \ldots, k - 1\} \to \{-1, w \ldots, 2w - 1\}$ obtains base for value of step of iteration:

$$Step_1(X, t) = \begin{cases} -1, & \text{if } Step_2(X, t - 1) = -1, \\ 0, & \text{if } t = -1, \\ Val(X, Step_2(X, t - 1), t) + w, & \text{otherwise.} \end{cases}$$

Function $Step_2 : \{0,1\}^n \times \{0, \ldots, k - 1\} \to \{-1, \ldots, w - 1\}$ obtain value of t-th step of iteration:

$$Step_2(X,t) = \begin{cases} -1, & \text{if } Step_1(X,t) = -1, \\ 0, & \text{if } t = -1 \\ Val(X, Step_1(X,t), t), & \text{otherwise.} \end{cases}$$

Note that address of current block is computed on previous step.
Result of Boolean function $SAF_{k,w}(X)$ is computed by following way:

$$SAF_{k,w}(X) = \begin{cases} 0, & \text{if } Step_2(X, k-1) \leq 0, \\ 1, & \text{otherwise.} \end{cases}$$

Let us discuss complexity properties of the function:

Lemma 5 ([Kha15])**.** *For integer $k = k(n)$, $w = w(n)$ and Boolean function $SAF_{k,w}$, such that inequality (1) holds, the following statement is right: $N(SAF_{k,w}) \geq w^{(k-1)(w-2)}$.*

Lemma 6 ([Kha15])**.** *There is $2k$-OBDD P of width $3w+1$ which computes $SAF_{k,w}$*

Let us present Lemma 5 in more useful form:

Corollary 7. *For integer $k = k(n)$, $w = w(n)$ and Boolean function $SAF_{k,w}$, such that inequality (1) holds, the following statement is right: $N(SAF_{k,w}) \geq w^{kw/6}$.*

3.1 Hierarchy Results for Classical Models

Hierarchy for deterministic OBDD is already known:

Theorem 8 ([Kha15])**.** *For integer $k = k(n), w = w(n)$ such that $2kw(2w + \lceil \log k \rceil + \lceil \log 2w \rceil) < n, k \geq 2, w \geq 64$ we have k-**OBDD**$_{\lfloor w/16 \rfloor - 3} \subsetneq k$-**OBDD**$_w$.*

Let us discuss hierarchies for nondeterministic and probabilistic models.

Theorem 9. *For $w \geq 8$ we have k–**NOBDD**$_{\sqrt{w}/2} \subsetneq k$–**NOBDD**$_{3w+1}$*

Proof. It is clear that k–**NOBDD**$_{\sqrt{w}/2} \subseteq$k–**NOBDD**$_{3w+1}$. Let us proof inequality of these classes. By Lemma 6 we have $SAF_{k,w} \in 2k$-**NOBDD**$_{3w+1}$. Let us show that $SAF_{k,w} \notin 2k$-**NOBDD**$_{\sqrt{w}/2}$.

$$\frac{N(SAF_{k,w})}{2^{\sqrt{w}/2(1+(2k-1)\sqrt{w}/2)}} \geq$$

$$\geq \frac{w^{kw/6}}{2^{\sqrt{w}/2(1+(2k-1)\sqrt{w}/2)}} =$$

$$= 2^{\frac{kw}{6}\log w - \frac{\sqrt{w}}{2} - \frac{1}{4}w(2k-1)} =$$

$$= 2^{\frac{\sqrt{w}}{2}(\frac{k\sqrt{w}}{3}\log w - 1 - k\sqrt{w} + \frac{\sqrt{w}}{2})} =$$

$$= 2^{\frac{\sqrt{w}}{2}(\frac{k\sqrt{w}}{3}(\log w - 3) + \frac{\sqrt{w}}{2} - 1)} > 1$$

Therefore $SAF_{k,w} \notin 2k$-OBDD$_{\sqrt{w}/2}$, due to Lemma 2.
And k–**NOBDD**$_{\sqrt{w}/2} \neq$k–**NOBDD**$_{3w+1}$ □

Theorem 10. *For $\sqrt{w}/(\log_2 k \log_2 w) \geq 1$ we have*
*k–**POBDD**$_{\sqrt{w}/(\log_2 k \log_2 w)} \subsetneq k$–**POBDD**$_{3w+1}$*

Proof. It is clear that k−**POBDD**$_{\sqrt{w}/(\log_2 k \log_2 w)} \subseteq$k−**POBDD**$_{3w+1}$. Let us proof inequality of these classes. By Lemma 6 we have $SAF_{k,w} \in 2k$-**POBDD**$_{3w+1}$. Let us show that $SAF_{k,w} \notin 2k$-**POBDD**$_{\sqrt{w}/(\log_2 k \log_2 w)}$.

$$\frac{N(SAF_{k,w})}{(C_1 k(C_2 + 0.5\log_2 w - \log_2\log_2 k - \log_2\log_2 w + \log_2 k))^{(2k+1)w/(\log_2 k \log_2 w)^2}} \geq$$

$$\geq \frac{w^{kw/6}}{(C_1 k(C_2 + 0.5\log_2 w - \log_2\log_2 k - \log_2\log_2 w + \log_2 k))^{(2k+1)w/(\log_2 k \log_2 w)^2}} =$$

$$= 2^{\frac{kw}{6}\log w - ((2k+1)w/(\log_2 k \log_2 w)^2)\log(C_1 k(C_2 + 0.5\log_2 w - \log_2\log_2 k - \log_2\log_2 w + \log_2 k))} \geq$$

$$\geq 2^{kw((\log w)/6) - (1/(\log_2 k \log_2 w)^2)\log(C_1 k(C_2 + 0.5\log_2 w - \log_2\log_2 k - \log_2\log_2 w + \log_2 k))} > 1$$

Therefore $SAF_{k,w} \notin$k−**POBDD**$_{\sqrt{w}/(\log_2 k \log_2 w)}$, due to Lemma 3.
And k−**POBDD**$_{\sqrt{w}/(\log_2 k \log_2 w)} \neq$k−**POBDD**$_{3w+1}$ □

4 Width Hierarchies for Quantum k-OBDD

Let us present existing lower bound for k-OBDDs.

Lemma 11 ([AKK17]). *Let function $f(X)$ is computed by bounded error k-QOBDD P of width w, then $N^{\pi}(f) \leq w^{C\cdot(kw)^2}$ for some $C = const$.*

Then let us discuss Boolean function *Matrix Xor Pointer Jumping*, which complexity property allow to show hierarchy.

Definition 12. Firstly, let us present version of PJ function which works with integer numbers. Let V_A, V_B be two disjoint sets (of vertices) with $|V_A| = |V_B| = d$ and $V = V_A \cup V_B$. Let $F_A = \{f_A : V_A \to V_B\}$, $F_B = \{f_B : V_B \to V_A\}$ and $f = (f_A, f_B) : V \to V$ defined by $f(v) = f_A(v)$, if $v \in V_A$ and $f = f_B(v)$, $v \in V_B$. For each $j \geq 0$ define $f^{(j)}(v)$ by $f^{(0)}(v) = v$, $f^{(j+1)}(v) = f(f^{(j)}(v))$. Let $v_0 \in V_A$. The functions we will be interested in computing is $g_{k,d} : F_A \times F_B \to V$ defined by $g_{k,d}(f_A, f_B) = f^{(k)}(v_0)$.

Definition of *Matrix XOR Pointer Jumping function* looks like *Pointer Jumping function*.

Firstly, we introduce definition of $MatrixPJ_{2k,d}$ function. Let us consider functions $f_{A,1}, f_{A,2}, \cdots f_{A,k} \in F_A$ and $f_{B,1}, f_{B,2}, \cdots f_{B,k} \in F_B$.

On iteration $j+1$ function $f^{(j+1)}(v) = f_{j+1}(f^{(j)}(v))$, where

$$f_i(v) = \begin{cases} f_{A,\lceil \frac{i}{2} \rceil}(v) \text{ if } i \text{ is odd} \\ f_{B,\lceil \frac{i}{2} \rceil}(v) \text{ if } i \text{ is even} \end{cases}.$$

$MatrixPJ_{2k,d}(f_{A,1}, f_{A,2}, \cdots f_{A,k}, f_{B,1}, f_{B,2}, \cdots f_{B,k}) = f^{(k)}(v_0)$.

Secondly, we add XOR-part to $MatrixPJ_{2k,d}$ (note it $XMPJ_{2k,d}$). Here we take $f^{(j+1)}(v) = f_{j+1}(f^{(j)}(v)) \oplus f^{(j-1)}(v)$, for $j \geq 0$

Finally, we consider boolean version of these functions. Boolean function $PJ_{t,n} : \{0,1\}^n \to \{0,1\}$ is boolean version of $g_{k,d}$, where we encode f_A in a binary string using $d \log d$ bits and do it with f_B as well. The result of function is parity of binary representation of result vertex.

In respect to boolean $MXPJ_{2k,d}$ function we encode functions in input in following order $f_{A,1}, \ldots, f_{A,k}, f_{B,1}, \ldots, f_{B,k}$. Let us describe process of computation on Figure 1. Function $f_{A,i}$ is encoded by $a_{i,1}, \cdots a_{i,d}$, for $i \in \{1 \cdots k\}$. And $f_{B,i}$ is encoded by $b_{i,1}, \cdots b_{i,d}$, for $i \in \{1 \cdots k\}$. Typically, we will assume that $v_0 = 0$.

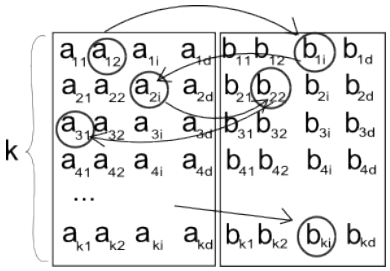

Figure 1: Boolean function $XMPJ_{k,d}$

Next we discuss complexity properties of $MXPJ_{2k,d}$.

Lemma 13 ([AKK17])**.** *For $kd \log d = o(n)$ following is right: $N^{id}(MXPJ_{2k,d}) \geq d^{\lfloor d/3-1 \rfloor (k-3)}$*

Lemma 14 ([AKK17])**.** *There is exact k-id-QOBDD P of width d^2 which computes $MXPJ_{2k,d}$.*

Let us present Lemma 13 in more useful form:

Corollary 15. *For integer $k = k(n)$, $w = w(n)$, $kd \log d = o(n)$ and Boolean function $MXPJ_{2k,d}$, the following statement is right: $N^{id}(MXPJ_{2k,d}) \geq d^{dk/16}$*

4.1 Hierarchy Results for Quantum Models

Now we can prove hierarchy results for k-QOBDDs .

Theorem 16. *We have k-id-**QOBDD**$_{\sqrt{d/C_1 k}} \subsetneq k$-id-**QOBDD**$_{d^2}$ for some $C_1 = const$.*

Proof. It is clear that k-id-**QOBDD**$_{\sqrt{d/C_1 k}} \subseteq k$-id-**QOBDD**$_{d^2}$. Let us proof inequality of these classes.

Due to Lemma 14, $MXPJ_{2k,d} \in k$-id-**QOBDD**$_{d^2}$. Let us show that $MXPJ_{2k,d} \notin k$-id-**QOBDD**$_{\sqrt{d/Ck}}$.

$$\frac{N^{id}(MXPJ_{2k,d})}{\sqrt{d/C_1 k}^{C(k\sqrt{d/C_1 k})^2}} \geq$$

$$\geq \frac{d^{dk/16}}{\sqrt{d/C_1 k}^{C(k\sqrt{d/C_1 k})^2}} =$$

$$= 2^{\frac{dk}{16} \log d - C(k\sqrt{d/C_1 k})^2 \log \sqrt{d/C_1 k}} =$$

$$= 2^{\frac{k}{16}\left(d \log d - \frac{16Ckd}{2C_1 k} \log d/C_1 k\right)}$$

Let $C_1 = 8C$ for C from Lemma 11, then we have

$$\frac{N^{id}(MXPJ_{2k,d})}{\sqrt{d/C_1k}^{C(k\sqrt{d/C_1k})^2}} \geq 2^{\frac{k}{16}\log C_1 k} > 1.$$

Therefore $MXPJ_{2k,d} \notin k$-id-$\mathbf{QOBDD}_{\sqrt{d/Ck}}$, due to Lemma 11.

And k-id-$\mathbf{QOBDD}_{\sqrt{d/C_1k}} \neq k$-id-$\mathbf{QOBDD}_{d^2}$. □

5 Discussion on Relation Between Models

You can see some existing discussion between models in [AKK17], [AGKY16], [AGK+05], [Gai15], [GY17], [IKY17]. Here we will present some relations on fixed k.

Relation between models follows from Theorems 9, 10:

Theorem 17. *There are Boolean function f, such that:*
$f \in k$-$\mathbf{OBDD}_{3w+1}$,k–$\mathbf{NOBDD}_{3w+1}$,k–$\mathbf{POBDD}_{3w+1}$;
$f \notin k$-$\mathbf{OBDD}_{\lfloor w/16 \rfloor -3}$,$k$–$\mathbf{NOBDD}_{\sqrt{w}/2}$,$k$–$\mathbf{POBDD}_{\sqrt{w}/(\log_2 k \log_2 w)}$

Let us compare classical models and quantum models. Firstly, let us discuss classical complexity properties of $MXPJ_{2k,d}$ function.

Lemma 18. *There is k-id-OBDD P of width d^2 which computes $MXPJ_{2k,d}$.*

Proof. Let us construct such k-id-OBDD P.

By the definition of function $MXPJ_{2k,d}$ input separated into $2dk$ blocks by $t = \lceil \log_2 d \rceil$ bits. Blocks encode integers $a_{i1}, a_{i2} \cdots a_{id}$ for $i \in \{1, \cdots k\}$ in the first part of input; and $b_{i1}, b_{i2} \cdots b_{id}$ for $i \in \{1, \cdots k\}$ in the second part (see Figure 1). Let elements of block representing a_{ij} be $X^{i,j} = (x_0^{i,j}, \ldots, x_{t-1}^{i,j})$ for $i \in \{1, \cdots k\}, j \in \{1, \cdots d\}$ and elements of block representing b_{ij} be $Y^{i,j} = (y_0^{i,j}, \ldots, y_{t-1}^{i,j})$ for $i \in \{1, \cdots k\}, j \in \{1, \cdots d\}$

Let us discuss i-th layer. On the first level we have d^2 nodes, each of them corresponds to pair (u, v), for $u, v \in 0, \ldots, d-1$ for storing $f^{(2i-3)}$ and $f^{(2i-2)}$. At first P skips all blocks except $x^{i,f^{(2i-2)}}$. Then it will compute XOR of bits of the block and u of pair. In the end of the block P leads node corresponding to $(f^{(2i-1)}, f^{(2i-2)})$. After that the program skip all other blocks of first part and all blocks of first part except $y^{i,f^{(2i-1)}}$. Then it computes XOR with of bits of the block and v of pair. In the end of the block P leads node corresponding to $(f^{(2i-1)}, f^{(2i)})$.

On the last layer after computing $f^{(2k)}$, all nodes which XOR result is 1 leads 1-sink. □

Lemma 19. $MXPJ_{2k,d} \notin k$-*id*-$\mathbf{OBDD}_{d/32}$.

Proof. Let us apply Lemma 1

$$\frac{MXPJ_{2k,d}}{(d/32)^{(k-1)d/32+1}} \geq \frac{d^{dk/16}}{(d/r)^{(k-1)d/32+1}} \geq$$

$$\geq 2^{\frac{kd}{16}\log d - 2\log(d/32)kd/(32)} = 2^{kd(\frac{\log d}{16} - 2(\log d - 5)/32)} > 1$$

Therefore $MXPJ_{2k,d} \notin k$-id-$\mathbf{OBDD}_{w/32}$, due to Lemma 1. □

Lemma 20. $MXPJ_{2k,d} \notin k\text{-}id\text{-}\mathbf{NOBDD}_{\sqrt{(d\log d)/33}}$.

Proof. Let us apply Lemma 2. Let $r = \sqrt{33d/\log d}$

$$\frac{MXPJ_{2k,d}}{2^{((k-1)d/r+1)d/r}} \geq \frac{d^{dk/16}}{2^{((k-1)d/r+1)d/r}} \geq$$

$$\geq 2^{\frac{kd}{16}\log d - 2(kd/r)d/r} = 2^{kd(\frac{\log d}{16} - 2d/r^2)} > 1$$

Therefore $MXPJ_{2k,d} \notin k\text{-id-}\mathbf{NOBDD}_{\sqrt{(d\log d)/33}}$, due to Lemma 2. □

Lemma 21. $MXPJ_{2k,d} \notin k\text{-}id\text{-}\mathbf{POBDD}_{\sqrt{d/\log k}}$.

Proof. Let us apply Lemma 3. Let $r = \sqrt{d/\log k}$

$$\frac{MXPJ_{2k,d}}{(C_1 k(C_2 + \log_2 d - \log_2 r + \log_2 k))^{(k+1)d^2/r^2}} \geq$$

$$\geq \frac{d^{dk/16}}{(C_1 k(C_2 + \log_2 d - \log_2 r + \log_2 k))^{(k+1)d^2/r^2}} \geq$$

$$\geq 2^{\frac{kd}{16}\log d - 2(kd^2/r^2)(C_3 + \log_2 k + \log_2(C_2 + \log_2 d + \log_2 k))} =$$

$$= 2^{2kd(\frac{\log d}{32} - d(C_3 + \log_2 k + \log_2(C_2 + \log_2 d + \log_2 k))/r^2)} > 1$$

Therefore $MXPJ_{2k,d} \notin k\text{-id-}\mathbf{POBDD}_{d/\log k}$, due to Lemma 3. □

Then base on these lemmas we can get following result:

Theorem 22. *There is Boolean function f, such that:*
$f \in k\text{-}id\text{-}\mathbf{OBDD}_{d^2}, k\text{-}id\text{-}\mathbf{NOBDD}_{d^2}, k\text{-}id\text{-}\mathbf{POBDD}_{d^2}, k\text{-}id\text{-}\mathbf{QOBDD}_{d^2}$;
$f \notin k\text{-}\mathbf{OBDD}_{\lfloor d/32 \rfloor - 3}$, $k\text{-}id\text{-}\mathbf{NOBDD}_{\sqrt{(d\log d)/33}}$, $k\text{-}id\text{-}\mathbf{POBDD}_{\sqrt{d/\log k}}$, $k\text{-}id\text{-}\mathbf{QOBDD}_{\sqrt{d/C_1 k}}$

References

[AG05] Farid Ablayev and Aida Gainutdinova. Complexity of quantum uniform and nonuniform automata. In *Developments in Language Theory*, volume 3572 of *LNCS*, pages 78–87. Springer, 2005.

[AGK01] Farid Ablayev, Aida Gainutdinova, and Marek Karpinski. On computational power of quantum branching programs. In *FCT*, volume 2138 of *LNCS*, pages 59–70. Springer, 2001.

[AGK+05] Farid Ablayev, Aida Gainutdinova, Marek Karpinski, Cristopher Moore, and Christopher Pollett. On the computational power of probabilistic and quantum branching program. *Information and Computation*, 203(2):145–162, 2005.

[AGKY14] Farid Ablayev, Aida Gainutdinova, Kamil Khadiev, and Abuzer Yakaryılmaz. Very narrow quantum obdds and width hierarchies for classical obdds. In *Descriptional Complexity of Formal Systems*, volume 8614 of *Lecture Notes in Computer Science*, pages 53–64. Springer, 2014.

[AGKY16] F. Ablayev, A. Gainutdinova, K. Khadiev, and A. Yakaryılmaz. Very narrow quantum obdds and width hierarchies for classical obdds. *Lobachevskii Journal of Mathematics*, 37(6):670–682, 2016.

[AK13] Farid Ablayev and Kamil Khadiev. Extension of the hierarchy for k-OBDDs of small width. *Russian Mathematics*, 53(3):46–50, 2013.

[AKK17] Farid Ablayev, Kamil Khadiev, and Aliya Khadieva. Lower bound and hierarchies for quantum ordered read-k-times branching programs. 2017. Submitted to DCFS 2017.

[AW02] Andris Ambainis and John Watrous. Two–way finite automata with quantum and classical states. *Theoretical Computer Science*, 287(1):299–311, 2002.

[BSSW98] Beate Bollig, Martin Sauerhoff, Detlef Sieling, and Ingo Wegener. Hierarchy theorems for kobdds and kibdds. *Theoretical Computer Science*, 205(1):45–60, 1998.

[Gai15] A. F. Gainutdinova. Comparative complexity of quantum and classical obdds for total and partial functions. *Russian Mathematics*, 59(11):26–35, 2015.

[GY17] Aida Gainutdinova and Abuzer Yakaryılmaz. Nondeterministic unitary obdds. *arXiv preprint arXiv:1612.07015*, 2017. Accepted by CSR 2017.

[IKY17] Rishat Ibrahimov, Kamil Khadiev, and Abuzer Yakaryılmaz. Exact affine obdds. *arXiv preprint arXiv:1703.07184*, 2017. Submitted to DCFS 2017.

[Kha15] K. Khadiev. Width hierarchy for k-obdd of small width. *Lobachevskii Journal of Mathematics*, 36(2), 2015.

[Kha16] K. Khadiev. On the hierarchies for deterministic, nondeterministic and probabilistic ordered read-k-times branching programs. *Lobachevskii Journal of Mathematics*, 37(6):682–703, 2016.

[KK] Kamil Khadiev and Aliya Khadieva. Reordering method and hierarchies for quantum and classical ordered binary decision diagrams.

[NHK00] Masaki Nakanishi, Kiyoharu Hamaguchi, and Toshinobu Kashiwabara. Ordered quantum branching programs are more powerful than ordered probabilistic branching programs under a bounded-width restriction. In *COCOON*, volume 1858 of *LNCS*, pages 467–476. Springer, 2000.

[NW91] Noam Nisan and Avi Widgerson. Rounds in communication complexity revisited. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 419–429. ACM, 1991.

[Sau06] Martin Sauerhoff. Quantum vs. classical read-once branching programs. In Matthias Krause, Pavel Pudlák, Rüdiger Reischuk, and Dieter van Melkebeek, editors, *Complexity of Boolean Functions*, number 06111 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2006. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

[SS05] Martin Sauerhoff and Detlef Sieling. Quantum branching programs and space-bounded nonuniform quantum complexity. *Theoretical Computer Science*, 334(1-3):177–225, 2005.

[Weg00] Ingo Wegener. *Branching Programs and Binary Decision Diagrams: Theory and Applications*. SIAM, 2000.

[YS10] Abuzer Yakaryılmaz and A. C. Cem Say. Succinctness of two-way probabilistic and quantum finite automata. *Discrete Mathematics and Theoretical Computer Science*, 12(2):19–40, 2010.

Nucleolus as Tournament Solution

Aleksei Yu. Kondratev *Vladimir V. Mazalov*

Russian Academy of Sciences, Karelian Research Center, Institute of Applied Mathematical Research, Pushkinskaya. 11, 185910 Petrozavodsk, Karelia, Russia
{kondratev,vmazalov}@krc.karelia.ru
http://mathem.krc.karelia.ru/

Keywords: tournament solution, adjacency matrix, majority rule, iterated uncovered set, social choice, characteristic function, nucleolus, Banks set

Aggregating individual preferences of electors on a set of candidates (alternatives) is an important problem in social choice theory. Simple majority rule is the most natural way to determine a binary relation on the set of alternatives [12]. For any pair of the candidates, a winner is the one who collects more than half of votes, otherwise a draw is declared. The Condorcet paradox lies in the fact that in the case of two or more alternatives the group binary relation through the majority rule may not be transitive. It means that there is no maximal element (Condorcet winner) even if a personal preference of each voter is transitive. Thereby, in the voting problem the concept of a maximal element requires a generalization.

Simple majority rule defines a *dominance relation* which is usually presented in the form of an *adjacency matrix* H with values of zeros, units and one half that match wins, losses or draws in the case of pairwise comparisons. A *weak tournament* T is the set of m alternatives $A = \{a_1, \ldots, a_m\}$ and the dominance relation on this set, which is determined by the adjacency matrix, i. e., $T = (A, H)$. The weak tournament without ties is simply denoted as the *tournament*. Formally, assume that $H(a, a) = 0, a \in A$. Note that the dominance relation is complete and reflexive, but not transitive.

A rule $S(T)$ which determines a nonempty subset of winners in the weak tournament is called a *tournament solution*. A *tournament ranking procedure* aims at assigning ranks from 1 to m to all alternatives. Note that such ranking can be non strict. Any tournament ranking procedure determines the certain tournament solution in a natural way, by choosing the candidates with the highest ranks.

We say that a candidate a *dominates* b if $H(a, b) = 1$. The *dominion* of the candidate a is denoted by $D(a) = \{b : H(a, b) = 1\}$. On the contrary, the *dominators* of the candidate a is denoted by $\overline{D}(a) = \{b : H(a, b) = 0\}$. We say that the candidate a *covers* b if $D(a) \supseteq D(b)$, $\overline{D}(a) \subseteq \overline{D}(b)$, and $H(a, b) = 1$. The covering relation is transitive, reflexive and asymmetric, but not complete.

The *uncovered set* $UC(T)$ consists of all uncovered alternatives [2, 4, 7, 14, 15]. For the uncovered set UC we can also calculate the second-order uncovered set $UC(UC) = UC^2$, and so on, up to the smallest set UC^∞, which is known as the *iterated uncovered set*.

The *minimal covering set* $MC(T)$ is an inclusion-minimal subset $MC \subseteq A$, such that any candidate $a \notin MC$ is covered in the set $MC \cup \{a\}$, i. e., $a \notin UC(MC \cup a, H)$ (see [5, 11]). The *Banks set* $BA(T) \subseteq A$ consists of maximal elements of all inclusion-maximal transitive subsets of alternatives [1, 2].

There exist a method (maximal lottery) of probabilistic determining winners in the tournament which is based on noncooperative game theory. In this method a zero-sum game with the payoff matrix $G = H - H^t$ is considered. Corresponding tournament solution, the *bipartisan set* $BP(T)$, is defined as a support of the unique Nash equilibrium in mixed strategies [6, 10, 11].

It is well-known that in any tournament $BP \subset MC \subset UC^\infty \subset UC$, as well as $BA \subset UC$. To learn more about tournament solutions you can in the recently published book [3].

As a particular example we consider a tournament where there are $m = 5$ candidates and the adjacency matrix has the following form.

The adjacency matrix in Example 1.

	a	b	c	d	e
a		0	1	1	1
b	1		0	0	1
c	0	1		1	0
d	0	1	0		1
e	0	0	1	0	

In this example the candidate a covers e, so the uncovered set equals $UC = \{a, b, c, d\}$. In the subtournament $\{a, b, c, d\}$ the candidate c covers d, thus, the iterated uncovered set and the minimal covering set are the same, $MC = UC^{\infty} = \{a, b, c\}$. The inclusion-maximal transitive subsets of alternatives are acd, ade, aec, bae, cdb, dbe, hence, the Banks set equals $BA = \{a, b, c, d\}$.

We propose to apply cooperative game theory for the ranking of alternatives in a weak tournament. In papers [8, 9] we use the Shapley value. Define the value of the characteristic function $v(K)$ for the cooperative game as the guaranteed payoff in the constant-sum game of the coalition $K \subseteq A$ against the counter coalition $A \setminus K$ by the formula

$$v(K) = \max_{a \in K} \min_{b \in A \setminus K} H(a, b), \quad K \subseteq A, \quad v(\emptyset) = 0, \quad v(A) = 1.$$

From the definition it follows that in the case of a weak tournament the ch. f. v is non-negative and monotonic, that is $0 \le v(S) \le v(K) \le 1$ for any $S \subseteq K \subseteq A$. Moreover, in the case of a tournament it takes the values 0 or 1, and is superadditive, that is $v(S) + v(K) \le v(S \cup K)$ for any $S \cap K = \emptyset$. Calculate the ch. f. v for all $2^m - 1$ subsets of alternatives in example 1.

The characteristic function v in Example 1.

K	$\emptyset$	a	b	c	d	e	ab	ac	ad	ae	bc	bd	be	cd	ce	de
v	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0

abc	abd	abe	acd	ace	ade	bcd	bce	bde	cde
1	1	1	**1**	**1**	0	**1**	0	0	0

abcd	abce	abde	acde	bcde
1	1	1	1	1

The *excess* for the coalition $K \subseteq A = \{a_1, \ldots, a_m\}$ denotes by $e(x, K) = v(K) - \sum_{a_i \in K} x_i$, where $x = (x_1, \ldots, x_m)$ is the imputation such that $x_i \ge 0$ for any candidate $a_i \in A$, and $\sum_{i=1}^{m} x_i = v(A) = 1$. The vector $e(x) = (e(x, K))^{K \subseteq A}$ includes all $2^m - 1$ excesses. The *nucleolus* is the imputation which provides a minimum of the vector of excesses (see [13]). The *nucleolus support* is the tournament solution $NU(A, H) = \{a \in A : N_a > 0\}$. Denote by $RN(T)$ the tournament ranking procedure which sequentially calculates the nucleolus for the candidates with zero scores. In example 1 the nucleolus support is strictly contained in the Banks set, $\{a, b, c\} = NU(T) \subsetneq BA(T) = \{a, b, c, d\}$.

The ranking in Example 1.

	a	b	c	d	e	ranking
RN	1/3	1/3	1/3	0	0	$a = b = c > d > e$
	-	-	-	1	0	

Theorem 1. *For every tournament (without ties), the ranking of alternatives according to the nucleolus $N(T)$ for the characteristic function v, satisfies:*
1) $N_b = 0$ for any candidate $b \notin UC$.
2) The nucleolus does not depend on covered alternatives, i. e., $N = N(UC^\infty)$.
3) The support of the nucleolus is a refinement of the iterated uncovered set, $NU \subset UC^\infty$.
4) There exists a tournament T, where $NU(T) \subsetneq MC(T)$.
5) There exists a tournament T, where $NU(T) \subsetneq BA(T)$.

Acknowledgments. This work is supported by the Russian Humanitarian Science Foundation (grant 15-02-00352_a) and the Russian Fund for Basic Research (project 16-51-55006 China_a).

References

[1] Banks, J. S. (1985). Sophisticated voting outcomes and agenda control. Social Choice and Welfare, 1(4), 295-306.

[2] Brandt, F. (2011). Minimal stable sets in tournaments. Journal of Economic Theory, 146(4), 1481-1499.

[3] Brandt, F., Conitzer, V., Endriss, U., Lang, J., Procaccia, A., editors. Handbook of Computational Social Choice. Cambridge University Press, 2016.

[4] Brandt, F., Geist, C., Harrenstein, P. (2016). A note on the McKelvey uncovered set and Pareto optimality. Social Choice and Welfare, 46(1), 81-91.

[5] Dutta, B. (1988). Covering sets and a new Condorcet choice correspondence. Journal of Economic Theory, 44(1), 63-80.

[6] Dutta, B., Laslier, J. F. (1999). Comparison functions and choice correspondences. Social Choice and Welfare, 16(4), 513-532.

[7] Fishburn, P. C. (1977). Condorcet social choice functions. SIAM Journal on applied Mathematics, 33(3), 469-489.

[8] Kondratev, A., Mazalov, V. (2015). The Ranking Problem of Alternatives as a Cooperative Game. arXiv preprint arXiv:1511.04437.

[9] Kondratev, A., Mazalov, V. (2017). A Ranking Procedure with the Shapley Value. In: Nguyen N., Tojo S., Nguyen L., Trawinski B. (eds) Intelligent Information and Database Systems. ACIIDS 2017. Lecture Notes in Computer Science, vol 10192. Springer, Cham.

[10] Laffond, G., Laslier, J. F., Le Breton, M. (1993). The bipartisan set of a tournament game. Games and Economic Behavior, 5(1), 182-201.

[11] Laslier, J. F. Tournament solutions and majority voting. Springer Verlag, 1997.

[12] May, K. O. (1952). A set of independent necessary and sufficient conditions for simple majority decision. Econometrica: Journal of the Econometric Society, 680-684.

[13] Mazalov, V. Mathematical game theory and applications. John Wiley & Sons, 2014.

[14] Miller, N. R. (1980). A new solution set for tournaments and majority voting: Further graph-theoretical approaches to the theory of voting. American Journal of Political Science, 68-96.

[15] Moulin, H. (1986). Choosing from a tournament. Social Choice and Welfare, 3(4), 271-291.

On robustness of configuration graphs in random environment

Marina Leri *Yuri Pavlov*

Institute of Applied Mathematical Research, Karelian Research Centre RAS
e-mail: leri@krc.karelia.ru, e-mail: pavlov@krc.karelia.ru

Random graphs are widely used for modeling of the topology and dynamics of complex networks such as the Internet, social and telecommunication networks (see, e.g. [1, 2]). Observations of real networks showed [1, 3, 4] that vertex degrees of such graphs can be considered to be independent identically distributed random variables following the power-law distribution. Thus, one of the most appropriate graphs for modeling the networks are so called configuration graphs [5] with random independent vertex degrees. Let ξ be a random variable equal to any vertex degree. This random variable takes natural values equal to the number of vertex semiedges, i.e. edges for which adjacent vertices are not yet specified. All of the semiedges are numbered in an arbitrary order. The sum of vertex degrees of any configuration graph has to be even, otherwise one extra semiedge is added to an equiprobably chosen vertex. Graph construction is completed by joining all semiedges one to another pairwise equiprobably to form edges.

In [4] the authors showed that in the considered graph models the distribution of the random variable ξ can be defined as follows:

$$\mathbf{P}\{\xi = k\} = k^{-\tau} - (k+1)^{-\tau}, \tag{1}$$

where $k = 1, 2, \ldots$, $\tau > 0$. It has been noted in many papers that for real networks the parameter τ can be regarded to be fixed and lying in the interval $(1, 2)$ (see, e.g. [1, 2, 3, 4]). But in some problems, as has been shown by our recent results [6, 7], models with the parameter $\tau > 2$ are not without interest.

There are also models where the vertex degree distribution parameter can change with an increase in the number of graph vertices or even have a random behaviour [8]. Let N be the number of graph vertices. In [9] the dynamics of the degree structure of a configuration graph with vertex degree distribution (1) was considered for the case where $N \to \infty$ and the parameter τ is a random variable uniformly distributed on the interval $[a, b]$, $0 < a < b < \infty$. However, it is more natural to suppose that the distribution of the parameter τ on a finite interval is unimodal. In this work we assume that τ has a truncated gamma distribution on the interval $[a, b]$ with the distribution parameters $(2, \lambda)$. The choice of the parameter $\alpha = 2$ of the distribution ensures unimodality. The density of this distribution is as follows:

$$f_\lambda(x) = \frac{\lambda^2 x e^{-\lambda x}}{F_{2,\lambda}(b) - F_{2,\lambda}(a)}, \qquad x \in [a, b], \tag{2}$$

where $F_{2,\lambda}(a)$, $F_{2,\lambda}(b)$ are the values of the gamma distribution function with the parameters 2 and λ at the points a and b, respectively.

The main goal of this paper is to study the robustness of such configuration graphs to destructive influences under the condition that graph evolution takes place in a random environment. This means that vertex degrees follow the distribution (1), where values of the parameter τ are determined separately for each vertex from the distribution with the density (2). Let us note that using the equations (1) and (2) we can average the

distribution of the random variable ξ and deduce that

$$\mathbf{P}\{\xi = k\} = \frac{\lambda^2}{F_{2,\lambda}(b) - F_{2,\lambda}(a)} \int_a^b x e^{-\lambda x} (e^{-x \ln k} - e^{-x \ln (k+1)}) dx, \tag{3}$$

where $k = 1, 2, \ldots$. It is obvious that to study the dynamics of graphs where all vertex degrees have the distribution (3) is much easier as opposed to a random environment. Therefore we have a problem to compare the robustness of configuration graphs in these two cases. This will allow to obtain the conditions under which the study of random graphs in a random environment can be substituted by the study of graphs with vertex degree distribution (3).

We consider two cases of destructive influences: random and targeted. In the case of a random breakdown equiprobably chosen vertices are removed from the graph one after another with all the incident edges. In the case of a targeted attack vertices with the highest degrees are removed sequentially.

It is well known [4] that if the parameter $\tau \in (1, 2)$, then the configuration graph has the only giant connected component which size is proportional to N. As $N \to \infty$, the size of any other connected component is infinitesimal compared to the size of the giant component. If $\tau > 2$, then the graph does not have a giant component, but even in this case the largest component contains far more vertices than any other connected component. Let η be the percentage of vertices in the largest connected component and r be the percentage of vertices removed from the graph. We obtained the dependencies of η on r and λ. In [6] we proposed a criterion of graph destruction. Let us consider a graph to be in destroyed state if the following event occurred: $\mathrm{A} : \{\eta \leq 2\eta_2\}$, where η_2 is the percentage of vertices in the second-sized connected component. Let us denote by $p = \mathbf{P}\{\mathrm{A}\}$ the probability of the event A. Regression dependencies of p on r and λ were found.

The main methods of the study were simulation modelling and methods of screening of significant variables in regression problems.

The obtained dependencies for η and p in the two cases of graph destruction process are given below. In these models we use the following notations: $\xi \sim F[a, b]$ means that vertex degrees follow the distribution (3) defined on the interval $[a, b]$, $\tau \sim \mathrm{G}[a, b]$ means that the destruction takes place in the conditions of random environment. Graph destructions were simulated for graphs sized from 1000 to 10000 vertices, the parameter $0.3 \leq \lambda \leq 2.5$ and the three intervals $[a, b]$: $(1, 2)$, $(1, 3]$ and $[2, 3]$. Power-law configuration graphs with the parameter $\tau \in (1, 2)$ are considered to be a good representation of the AS-level Internet topology [3, 4]. Graphs with the parameter $\tau \in [2, 3]$ are useful for the studies of forest fire models [6, 7] and the interval $(1, 3]$ was chosen as a generalization.

Thus, in the case of random breakdowns the following regression models were obtained:

$$\begin{aligned}
\tau \sim \mathrm{G}(1,2): &\quad \eta = 78.12 - 1.11r,\\
&\quad p = -0.72 - 0.045\lambda + 0.00036r^2,\\
\xi \sim F(1,2): &\quad \eta = 78.07 - 1.11r,\\
&\quad p = -0.7 - 0.035\lambda + 0.00035r^2,\\
\tau \sim \mathrm{G}[2,3]: &\quad \eta = 23.4 + 0.51\lambda - 3.21r^{0.51},\\
&\quad p = 0.088 - 0.077\lambda + 0.0017r^{1.6},\\
\xi \sim F[2,3]: &\quad \eta = 23.04 + 0.52\lambda - 3.03r^{0.52},\\
&\quad p = 0.086 - 0.075\lambda + 0.0016r^{1.62},\\
\tau \sim \mathrm{G}(1,3]: &\quad \eta = 60.74 + 5.35\lambda - 0.96r,\\
&\quad p = -0.55 - 0.08\lambda + 0.00035r^2,\\
\xi \sim F(1,3]: &\quad \eta = 60.9 + 5.74\lambda - 0.98r,\\
&\quad p = -0.5 - 0.087\lambda + 0.00034r^2.
\end{aligned}$$

Determination coefficients R^2 of all the above models are no less than 0.92. The values of r in the models for the probability of graph destruction p are limited as follows: $44.75 + 1.34\lambda \le r \le 69.1 + 0.89\lambda$ for $\tau \sim \mathrm{G}(1,2)$, $44.74 + 1.1\lambda \le r \le 69.6 + 0.7\lambda$ for $\xi \sim F(1,2)$, $-6.5 + 6.9\lambda \le r \le 50.8 + 2.57\lambda$ for $\tau \sim \mathrm{G}[2,3]$, $-6.13 + 6.6\lambda^{1.2} \le r \le 50.35 + 2.35\lambda$ for $\xi \sim F[2,3]$, $40 + 2.6\lambda \le r \le 66.5 + 1.6\lambda$ for $\tau \sim \mathrm{G}(1,3]$ and $38.5 + 3\lambda \le r \le 66.45 + 1.8\lambda$ for $\xi \sim F(1,3]$ with $R^2 = 0.99$ for all the models. It is clear that for the out-of-limits values of r the probability $p = 0$ when r is below the lower limit and $p = 1$ when r is above the upper limit. The same is true for corresponding models given below.
As for the process of targeted attack on vertices with the highest degrees the regression models were as follows:

$$\begin{aligned}
\tau \sim \mathrm{G}(1,2): &\quad \eta = 52.37 - 7.22r - 7.74\ln r,\\
&\quad p = -0.08 - 0.08\lambda + 0.084r^{1.61},\\
\xi \sim F(1,2): &\quad \eta = 52.15 - 7.23r - 7.8\ln r,\\
&\quad p = -0.1 - 0.09\lambda + 0.095r^{1.54},\\
\tau \sim \mathrm{G}[2,3]: &\quad \eta = -3.8 + 1.4\lambda + 3.6r - 5.6\ln r,\\
&\quad p = 1.21 - 0.11\lambda + 0.35\ln r,\\
\xi \sim F[2,3]: &\quad \eta = -3.35 + 1.44\lambda + 3r - 5.4\ln r,\\
&\quad p = 1.24 - 0.13\lambda + 0.36\ln r,\\
\tau \sim \mathrm{G}(1,3]: &\quad \eta = 26.8 + 10.6\lambda - 7.2r - 8.7\ln r,\\
&\quad p = 0.003 - 0.28\lambda + 0.34r^{0.98},\\
\xi \sim F(1,3]: &\quad \eta = 26.2 + 10.6\lambda - 6.9r - 8.8\ln r,\\
&\quad p = -0.034 - 0.3\lambda + 0.4r^{0.92}
\end{aligned}$$

with determination coefficients R^2 being no less than 0.93 and the following limits of r in models for p: $1.04 + 0.44\lambda \le r \le 4.8 + 0.22\lambda$ for $\tau \sim \mathrm{G}(1,2)$, $1.1 + 0.5\lambda \le r \le 4.9 + 0.25\lambda$ for $\xi \sim F(1,2)$, $0.03 + 0.02\lambda \le r \le 0.57 + 0.17\lambda$ for $\tau \sim \mathrm{G}[2,3]$, $0.03 + 0.02\lambda \le r \le 0.54 + 0.18\lambda^{1.5}$ for $\xi \sim F[2,3]$, $-0.024 + 0.86\lambda \le r \le 2.98 + 0.86\lambda$ for $\tau \sim \mathrm{G}(1,3]$ and $0.03 + 0.84\lambda \le r \le 2.8 + 0.9\lambda$ for $\xi \sim F(1,3]$, where $R^2 = 0.99$ for all models.

The results of this work show that the patterns describing the destruction process in a random environment and a model with the vertex degree distribution (3) are practically identical. This means that the study of graph destruction processes in a random environment can be replaced by a simpler case with averaged vertex degree distribution, which, in fact, requires less computational time. Let us also note that on the interval $(1, 2)$ the fraction of vertices in the giant component depends only on the fraction of removed vertices r and does not depend on the parameter λ.

The study is supported by the Russian Foundation for Basic Research, grant 16-01-00005.

References

[1] Durrett R. Random Graph Dynamics. Cambridge: Cambridge Univ. Press. 2007.

[2] Hofstad R. Random Graphs and Complex Networks. Univ. of Technology. 2011. Vol. 1. http://www.win.tue.nl/∼rhofstad/NotesRGCN.pdf

[3] Faloutsos C., Faloutsos P., Faloutsos M. On power-law relationships of the Internet topology. Computer Communications Rev. 1999. Vol. 29, pp. 251-262.

[4] Reittu H., Norros I. On the power-law random graph model of massive data networks. Performance Evaluation. 2004. Vol. 55, pp. 3-23.

[5] Bollobas B. A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. Eur. J. Comb. 1980. Vol. 1, pp. 311-316.

[6] Leri M., Pavlov Y. Power-law random graphs' robustness: link saving and forest fire model. Austrian Journal of Statistics. 2014. Vol. 43, iss. 4, pp. 229-236.

[7] Leri M., Pavlov Y. Forest fire models on configuration random graphs. Fundamenta Informaticae. 2016. Vol. 145, iss. 3, pp. 313-322.

[8] Biaconi G., Barabasi A.-L. Bose-Einstein condensation in complex networks. Physical Review Letters. 2001. Vol. 86, iss. 24, pp. 5632-5635.

[9] Pavlov Y. On random graphs in random environment. Computer Data Analysis and Modeling: Proceedings of the XI International Conference. Minsk, Publishing centre of BSU. 2016. Pp. 177-180.

Trim Strongly Connected Synchronizing Automata and Ideal Languages*

Marina Maslennikova[1] *Emanuele Rodaro*[2]

[1] Ural Federal University, Ekaterinburg, Russia
[2] Dipartimento di Matematica, Politecnico di Milano, Milano, Italy
maslennikova.marina@gmail.com, emanuele.rodaro@fc.up.pt

Introduction

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a *deterministic finite automaton* (DFA), where Q is the *state set*, Σ stands for the *input alphabet*, and $\delta : Q \times \Sigma \to Q$ is the totally defined *transition function* defining the action of the letters in Σ on Q. Let $L[\mathscr{A}]$ denote the language accepted by $\mathscr{A}$. In what follows we will consider only languages which are regular, thus we will drop the term "regular".

A DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ is called *synchronizing* if there exists a word $w \in \Sigma^*$ whose action leaves the automaton in one particular state no matter at which state in Q it is applied, i.e. $\delta(q, w) = \delta(q', w)$ for all $q, q' \in Q$. Any word with this property is said to be *reset* for the DFA $\mathscr{A}$. By $\mathrm{Syn}(\mathscr{A})$ we denote the set of all reset words of $\mathscr{A}$. For a brief introduction to the theory of synchronizing automata we refer the reader to the survey [7].

Recently in a series of papers [3, 2, 6] a language theoretic approach to the study of synchronizing automata has been developed. Recall that a language $I \subseteq \Sigma^*$ is called a *two-sided ideal* (or simply an *ideal*) if I is non-empty and $\Sigma^* I \Sigma^* \subseteq I$. Synchronizing automata can be considered as a special representation of ideal languages. The complexity of such a representation is measured by the *reset complexity* $rc(I)$ which is the minimal possible number of states in a synchronizing automaton $\mathscr{A}$ such that $\mathrm{Syn}(\mathscr{A}) = I$ for a given ideal I. In [3] it has been shown that representation of an ideal language by means of a minimal synchronizing automaton can be exponentially more succinct than its "traditional" representation via minimal automaton recognizing this language.

In this language theoretic approach to synchronizing automata, strongly connected synchronizing automata play an important role. Recall that a DFA is called *strongly connected* if for each pair of different states (p, q) there exists a word mapping p to q. It is interesting to find out whether for every regular ideal language there exists a strongly connected synchronizing automaton whose set of reset words is equal to a given language. Indeed, while the minimal automaton recognizing an ideal language I is always a synchronizing automaton with a unique *sink* state (i.e. a state fixed by all letters), finding examples of strongly connected synchronizing automata $\mathscr{A}$ with $\mathrm{Syn}(\mathscr{A}) = I$ is a non-trivial task. In [6] it is proved that such strongly connected automaton exists for every ideal over alphabet of size at least two. The construction itself is non-trivial and rather technical. Therefore, the importance of the studies of issues like finding more effective constructions of the smallest strongly connected synchronizing automaton for an ideal language is evident. Questions concerning the size of such automaton were considered in [1, 2].

*This work was supported by the Russian Foundation for Basic Research, grant no. 16-01-00795, the Ministry of Education and Science of the Russian Federation, project no. 1.3253.2017, and the Competitiveness Enhancement Program of Ural Federal University.

Main results

In [4] strongly connected synchronizing automata have been characterized via homomorphic images of automata belonging to a particular class $\mathcal{L}(\Sigma)$. The class $\mathcal{L}(\Sigma)$ is formed by all the trim automata $\mathscr{A} = \langle Q, \Sigma, \delta, q_0, \{q_0\}\rangle$ such that $L[\mathscr{A}] = w^{-1}\Sigma^* w$ for some word $w \in \Sigma^*$. Recall that a DFA $\mathscr{A} = \langle Q, \Sigma, \delta, q_0, F\rangle$ is called *trim* whenever it does not contain any state that cannot be reached from the initial state, or state from which no final state can be reached. In the present work we strengthen results of [4, 6].

Theorem 1. *Let I be an ideal regular language over at least binary alphabet. For any word $w \in I$ of minimum length, there is a DFA $\mathscr{B} \in \mathcal{L}(\Sigma)$ with $L[\mathscr{B}] = w^{-1}\Sigma^* w$ and*

$$\Sigma^* w \Sigma^* \subseteq \mathrm{Syn}(\mathscr{B}) = I.$$

The improvement of [4] lies in a consequent analysis of $\mathscr{B}$ which is taken from that paper. In [4] it has been stated that the inclusion $\mathrm{Syn}(\mathscr{B}) \subseteq I$ can be guaranteed. However, this statement did not provide any new information about classes of automata where the minimal strongly connected synchronizing DFA for I could be found. In the present research we show that the equality $\mathrm{Syn}(\mathscr{B}) = I$ holds true. Furthermore, it implies that for every ideal language I over Σ one may construct a strongly connected trim DFA $\mathscr{B}$ from $\mathcal{L}(\Sigma)$ for which I serves as the language of reset words. The last argument strengthens the result of [6], where just the existence of a strongly connected synchronizing DFA for a given ideal language has been proved.

Further we initiate the study of structural properties of $\mathscr{B}$. Let us note that the construction of $\mathscr{B}$ from Theorem 1 is based on a strongly connected synchronizing DFA $\mathscr{A}$ such that $\mathrm{Syn}(\mathscr{A}) = I$. We prove that the resulting automaton $\mathscr{B}$ possesses at most $\sum_i |\delta(Q, w_i)|$ states, where Q is the state set of $\mathscr{A}$, δ stands for the transition function, w_i denotes the prefix of w of length i. This upper bound is shown ro be tight. For instance, if we take $I = \Sigma^{\geq n}$ (the set of all words of length at least n), then $\mathscr{A}$ and $\mathscr{B}$ coincide and, furthermore, $|Q| = 2^n$.

Now it would be interesting to find an algorithm to construct a DFA from $\mathcal{L}(\Sigma)$, for which given language I serves as the language of reset words, without applying the construction of some strongly connected synchronizing DFA for I. Therefore, a deeper understanding of the structure of automata from the class $\mathcal{L}(\Sigma)$ is a crucial point.

Every DFA from $\mathcal{L}(\Sigma)$ recognizes $w^{-1}\Sigma^* w$ for some w. We describe precisely the set of all minimal reset words for the minimal DFA $\mathscr{A}_w$ recognizing $w^{-1}\Sigma^* w$. Let $|w|$ denote the length of w. Since the case $|w| \leq 2$ is trivial, one may suppose that $|w| > 2$. It turns out that u is a minimal reset word for $\mathscr{A}_w$ if and only if u shares one of the following properties:

1) u is a prefix of w but u does not appear in w as an inner factor or proper suffix;

2) u is not a factor of w and $|u| \leq |w|$.

Since the set of all minimal reset words for $\mathscr{A}_w$ is finite, we have that $\mathrm{Syn}(\mathscr{A}_w)$ is a finitely generated ideal language, i.e. $\mathrm{Syn}(\mathscr{A}_w) = \Sigma^* U \Sigma^*$ for some finite set $U \subseteq \Sigma^*$. Such languages have already been viewed as languages of reset words of synchronizing automata in [5]. In the present paper we reveal that the construction of $\mathscr{A}_w$ may be used in order to build a strongly connected DFA from $\mathcal{L}(\Sigma)$ for which given finitely generated ideal language serves as the language of reset words. Furthermore, the procedure of constructing the corresponding automaton for some w may be more effective than the general algorithm building a strongly connected synchronizing DFA for a finitely generated ideal from [1].

References

[1] Gusev, V.V., Maslennikova, M.I., Pribavkina, E.V.: Finitely generated ideal languauges and synchronizing automata. In: J.Karhumäki, A.Lepisto, L.Zamboni (eds.), WORDS 2013, LNCS, vol. 8079, pp. 143–153, Springer (2013)

[2] Gusev, V.V., Maslennikova, M.I., Pribavkina, E.V.: Principal Ideal languages and synchronizing automata. In: V.Halava, J.Karhumäki, Yu. Matiyasevich (eds.), the Special Issue of the RuFiDiM 2012, Fundamenta Informaticae, vol. 132(1), pp. 95–108 (2014)

[3] Maslennikova, M.I.: Reset Complexity of Ideal Languages. In: M. Bieliková (eds.), SOFSEM 2012, Proc. Institute of Computer Science Academy of Sciences of the Czech Republic, vol. II, pp. 33–44 (2012)

[4] Maslennikova M., Rodaro E.: Representation of (left) ideal regular languages by synchronizing automata. In: L. Beklemishev, D. Musatov (eds.) 10th Int. Comp. Sci. Symp. CSR 2015, LNCS, vol. 9139, pp. 325–338, Springer-Verlag, Berlin-Heidelberg (2015)

[5] Pribavkina, E., Rodaro, E.: Synchronizing automata with finitely many minimal synchronizing words. Information and Computation, vol. 209(3), pp. 568-579 (2011)

[6] Reis, R., Rodaro, E.: Regular ideal languages and synchronizing automata. In: J. Karhumäki, A. Lepistö, L. Zamboni (eds.), WORDS 2013, LNCS, vol. 8079, pp. 205–216. Springer, Heidelberg (2013)

[7] Volkov, M.V.: Synchronizing automata and the Černý conjecture. In: Martín-Vide, C., Otto, F., Fernau, H. (eds.) LATA 2008, LNCS, vol. 5196, pp. 11–27. Springer, Heidelberg (2005)

Network analysis based on a typology of nodes

Vladimir Matveenko[1] *Alexei Korolev*[2]

[1] Department of Economics, National Research University Higher School of Economics, 16 Soyuza Pechatnikov Street, St. Petersburg 190121, Russia
vmatveenko@hse.ru

[2] Department of Applied Mathematics and Business Informatics, National Research University Higher School of Economics, 16 Soyuza Pechatnikov Street, St. Petersburg 190121, Russia
danitschi@gmail.com

Abstract

Commonly in network analysis an undirected graph (network) is represented by its adjacency matrix, and while the latter may have an enormous order. We show that in many cases instead of the adjacency matrix a much smaller matrix (referred as type adjacency matrix) may be used. We introduce notions of the types of nodes of graph and of the type adjacency matrix, propose an algorithm for division of the set of nodes into types. We study properties of the type adjacency matrix and some of its applications in the social and economic network analysis.

Keywords: undirected graph, network, type of node, centrality, growth of network, production, knowledge, externality, network game, Nash equilibrium, network formation.

1 Introduction

Commonly in network analysis and its economic and social applications an undirected graph (network) is represented by its adjacency matrix, $\mathbf{A}$. A drawback of the adjacency matrix is that it can have an enormous size, and this may trouble working with graphs possessing big size but simple structure. For example, for the star graph with ν peripheral nodes the adjacency matrix is of size $(\nu+1)\times(\nu+1)$, and this matrix, by itself, does not rely on the simple structure of the graph.

This drawback may be somehow rectified if another kind of matrix, referred further as type adjacency matrix, $\mathbf{T}$, is used. The type adjacency matrix usually has much smaller size in comparison with the adjacency matrix and reflects important features of the structure of the graph in an aggregate form. For example, for the str graph the type adjacency matrix has the size 2×2, independently on the number of peripheral nodes. Due to the small size, the type adjacency matrix can considerably simplify the network analysis.

The presence of the type adjacency matrix corresponds to the fact that the set of the nodes of undirected graph can be divided into disjoint subsets (types) in such way that each node of definite type has definite numbers of neighbors (adjacent nodes) of each type. We will consider such division with the minimal possible number of types.

For example, for the network shown in Figure 1, whose adjacency matrix is

$$\mathbf{A}=\begin{pmatrix}0&1&1&1&1\\1&0&1&1&0\\1&1&0&0&1\\1&1&0&0&1\\1&0&1&1&0\end{pmatrix},$$

the set of nodes is divided into two types: the 1^{st} type consists of node 1, and the 2^{nd} type includes nodes 2, 3, 4, 5 (see Figure 2). The 1^{st} type node is adjacent to 0 first-type nodes

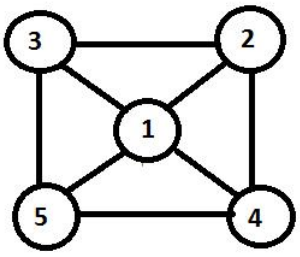

Figure 1: Graph with 5×5 adjacency matrix.

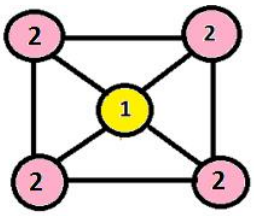

Figure 2: Division of the set of nodes into two types.

and to 4 second-type nodes, while each 2^{nd} type node is adjacent to 1 first-type node and 2 second-type nodes. These numbers of neighbors of different types can be written in form of the type adjacency matrix:

$$\mathbf{T} = \begin{pmatrix} 0 & 4 \\ 1 & 2 \end{pmatrix}.$$

We will see that in many respects the type adjacency matrix may serve as a substitute for the adjacency matrix.

In the present paper we provide a definition of the types of nodes and the type adjacency matrix, propose an algorithm for division of the set of nodes into types and construction of the type adjacency matrix, and study its properties and some of its applications.

2 Types of nodes

Let G be undirected graph (network) of order n and $\mathbf{A}$ be its adjacency matrix. Let us remind that $\mathbf{A}$ is $n \times n$ matrix such that $a_{ij} = a_{ji} = 1$ if in the graph there is an edge connecting nodes i and j, and $a_{ij} = a_{ji} = 0$ otherwise; $a_{ii} = 0$ for all $i = 1, 2, \ldots, n$.

The concept of types of nodes may be explained informally in the following way. The nodes of graph can be colored in S colors in such way that each node of color j has a definite number $l_i(j)$ of neighbors of color i (for each $i = 1, 2, \ldots, S$).

More formally, the set of nodes of graph may be decomposed into minimal number S of disjoint classes $j = 1, 2, \ldots, S$ in such way that any node belonging class j has $l_i(j)$ neighbors from class i (for $i = 1, 2, \ldots S$). The classes will be referred as *types of nodes*. Type j is characterized by vector $\mathbf{l}(j) = (l_1(j); l_2(j); \ldots; l_S(j))$, where $l_i(j)$ is the number of neighbors in class i for each node of class j.

2.1 Type adjacency matrix

Let the graph have S types of nodes. We construct a $S \times S$-matrix $\mathbf{T}$ in the following way. The first row of this matrix is the row vector $\mathbf{l}(1)$, the second row is $\mathbf{l}(2)$, ..., the S-th row is $\mathbf{l}(S)$:

$$\mathbf{T} = \begin{pmatrix} \mathbf{l}(1) \\ \mathbf{l}(2) \\ \dots \\ \mathbf{l}(S) \end{pmatrix} = \begin{pmatrix} l_1(1) & l_2(1) & \dots & l_S(1) \\ l_1(2) & l_2(2) & \dots & l_S(2) \\ \dots & \dots & \dots & \dots \\ l_1(S) & l_2(S) & \dots & l_S(S) \end{pmatrix}.$$

This matrix will be referred as *type adjacency matrix* of the graph. Our type adjacency matrix $\mathbf{T}$ is the same as, in spectral graph theory, a quotient matrix for the symmetric matrix $\mathbf{A}$ (e.g. [6], [2]).

Figures 3 and 4 provide an example of two graphs which have the same size and the same distribution of degrees, but different typology. Their type adjacency matrices are, correspondingly,

$$\begin{pmatrix} 0 & 2 \\ 2 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}.$$

Later we will see that there exist graphs of different size, but with the same typology.

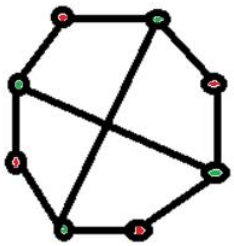

Figure 3: Graph with $\mathbf{l}(1) = (0; 2)$, $\mathbf{l}(2) = (2; 1)$.

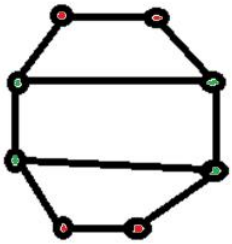

Figure 4: Graph with $\mathbf{l}(1) = (1; 1)$, $\mathbf{l}(2) = (1; 2)$.

2.2 Algorithm

Let us describe an algorithm of subdivision of the set of nodes of undirected graph into types. Let s be a current number of subsets of subdivision. Initially $s = 1$.

Iteration of the algorithm. Consider nodes of the first subset. If all of them have the same numbers of neighbors in each subset $1, 2, \dots, s$, then the first subset is not changed.

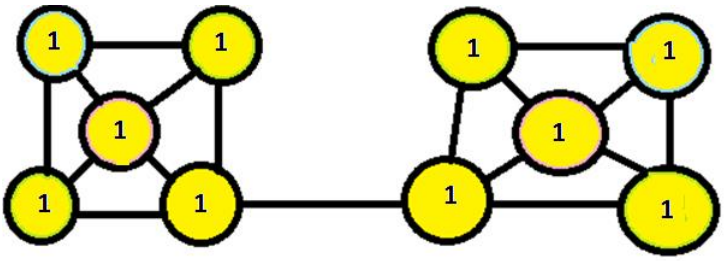

Figure 5: Start of the algorithm: $s = 1$.

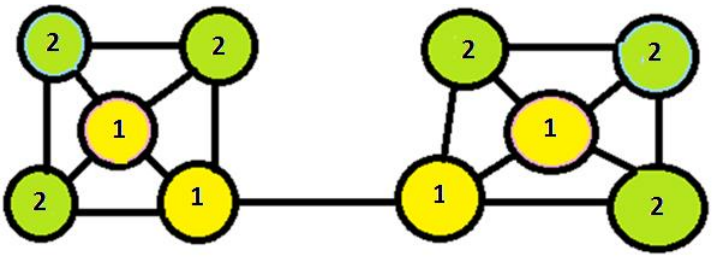

Figure 6: Result of the first iteration: $s = 2$.

In the opposite case, we divide the first subset into new subsets in such way that all nodes of each new subset have the same numbers of neighbors in subsets.

We proceed in precisely same way with the second, the third, ..., the s-th subset. If on the present iteration the number of subsets s has not changed, then the algorithm finishes its work. If s has increased, then the new iteration starts.

The number of subsets s does not decrease in process of the algorithm. Since s is bounded from above by the number n of nodes in the graph, the algorithm converges. It is clear that the algorithm divides the set of nodes into the minimal possible number of classes.

Example. Let us apply the algorithm to the graph depicted in Figure 2. Initially $s = 1$, all nodes constitute the same one set (Figure 5).

After the first iteration we obtain the division corresponding to degrees, depicted in Figure 6. Then, on the first step of the second iteration, we obtain the division depicted in Figure 7. On the second step of the second iteration we obtain the division shown in Figure 8.

On the third iteration nothing changes, and the algorithm stops.

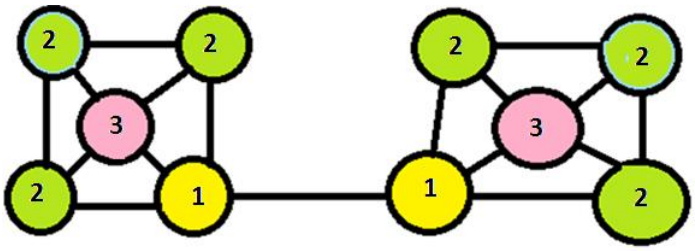

Figure 7: The first step of the second iteration: $s = 3$.

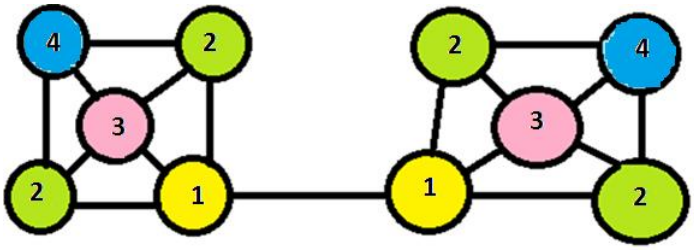

Figure 8: The second step of the second iteration: $s = 4$.

We have obtained a subdivision of the set of nodes of the graph into four types which are characterized by the vectors of numbers of neighbors:

$$\mathbf{l}(1) = (1;2;1;0), \mathbf{l}(2) = (1;0;1;1), \mathbf{l}(3) = (1;2;0;1), \mathbf{l}(4) = (0;2;1;0).$$

The corresponding type adjacency matrix is

$$\mathbf{T} = \begin{pmatrix} \mathbf{l}(1) \\ \mathbf{l}(2) \\ \mathbf{l}(3) \\ \mathbf{l}(4) \end{pmatrix} = \begin{pmatrix} 1 & 2 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 2 & 0 & 1 \\ 0 & 2 & 1 & 0 \end{pmatrix}.$$

2.3 Relation to modular division

Our subdivision of the set of nodes of graph into types is a generalization of the modular aggregation of Allouch [1]. Allouch [1] defines *module* as a subset of nodes, such that each node in a module has the same neighbors. It can be seen that each division into modules is a division into types, but not the opposite.

3 Some applications of the type adjacency matrix in network analysis

3.1 Numbers of routes

It is known that each element a_{ij}^k of the k-th power of the adjacency matrix, $\mathbf{A}^k$, is equal to the number of the k-step routes between nodes i and j. In a parallel way, each element t_{ij}^k of the k-th power of the type adjacency matrix, $\mathbf{T}^k$, shows the number of the k-step routes between any i-th type node and all j-th type nodes.

E.g., for the graph shown in Figure 1 the 3^{rd} power of the adjacency matrix,

$$\mathbf{A}^3 = \begin{pmatrix} 8 & 8 & 8 & 8 & 8 \\ 8 & 4 & 8 & 8 & 4 \\ 8 & 8 & 4 & 4 & 8 \\ 8 & 8 & 4 & 4 & 8 \\ 8 & 4 & 8 & 8 & 4 \end{pmatrix},$$

shows that in the graph there are 8 three-step routes from node 3 to node 2 and 4 three-step routes from node 3 to node 4. The same graph is characterized by the type adjacency matrix

$$\mathbf{T} = \begin{pmatrix} 0 & 4 \\ 1 & 2 \end{pmatrix}.$$

Matrix $\mathbf{T}$ shows, in particular, that the degree of the 1^{st} type node is $0+4=4$, and the degree of any 2^{nd} type node is $1+2=3$. The 3^{rd} power, e.g.,

$$\mathbf{T}^3 = \begin{pmatrix} 8 & 32 \\ 8 & 24 \end{pmatrix},$$

shows that in the graph there are 24 three-step routes from each 2^{nd} type node to all 2^{nd} type nodes.

3.2 Centrality measures

Network analysis uses various centrality measures, among them degree centrality, Katz-Boncicz centralities [12],[4], eigenvector centrality, and α- centrality [5].

The *degree centrality* of the node (or of each node of the type) may be calculated in a similar way by use of either the adjacency matrix, $\mathbf{A}$, or the type adjacency matrix $\mathbf{T}$:

$$C_D(i) = \frac{\sum_{j=1}^{n} a_{ij}}{n-1}, \tag{1}$$

$$C_D(\tilde{i}) = \frac{\sum_{j=1}^{S} t_{\tilde{i}\tilde{j}}}{n-1},$$

where $\tilde{i}$ is the type of node i.

Let us remind how the numbers of routes are used in calculation of the *Katz-Bonacich centrality measures*. Katz [12] and Bonacich [4] propose to use in definition of centrality measure not the degree of node i, as in formula (1), but the total discounted number of routes outgoing from the node i. All such routes are accounted for, and the longer the route is (the more is its length, k), the smaller discounting multiplier, α^k (where $0 < \alpha < 1$), it receives in calculation of the discounted number of routes.

Because, as was already mentioned, the number of the k-step routes is an element of the power of adjacency matrix, $\mathbf{A}^k$, the total discounted number of the routes between nodes i and j (where $i \neq j$) is equal to

$$m_{ij} = \sum_{k=1}^{\infty} \alpha^k a_{ij}^k.$$

Assuming for the sake of simplicity of further calculation that there is a fictitious zero-length route from node i to itself, one obtains the formula

$$\bar{m}_{ii} = \sum_{k=0}^{\infty} \alpha^k a_{ij}^k.$$

Using the notation for the identity matrix, $\mathbf{I} = (\alpha\mathbf{A})^0$, one comes to the formula for the matrix of the total discounted numbers of routes between pairs of nodes:

$$\bar{\mathbf{M}} = \sum_{k=0}^{\infty} (\alpha\mathbf{A})^k = (\mathbf{I} - \alpha\mathbf{A})^{-1}.$$

The total discounted number of routes between node i and all possible nodes j, called sometimes (e.g. [8]) *Bonacich centrality* of node i, is

$$C_B(i) = \sum_{j=1}^{n} \bar{m}_{ij}.$$

The vector of the Bonacich centrality measures for nodes is

$$\mathbf{C}_B = \begin{pmatrix} C_B(1) \\ C_B(2) \\ \dots \\ C_B(n) \end{pmatrix} = \bar{\mathbf{M}} \cdot \mathbf{1} = (\mathbf{I} - \alpha\mathbf{A})^{-1} \cdot \mathbf{1},$$

where $\mathbf{1}$ is the vector of all ones.

We have seen, that in this version of the formula for Bonacich centralities an unwanted 1 enters the measure for each node – it is the number of fictitious "zero-length" routes which was added for the sake of simplicity of calculation. Some authors, to improve this "fault", do subtract these unwanted units; then the formula turns into

$$\mathbf{C}_K = ((\mathbf{I} - \alpha\mathbf{A})^{-1} - \mathbf{I}) \cdot \mathbf{1}.$$

This version is often referred as *Katz centrality*. Both versions are met in the literature, often under similar names, what commonly puzzles readers.

Since only the summary numbers of the routes from node i (but not the numbers of routes to particular nodes) are needed in calculation of both Katz-Bonacich centrality measures, the type adjacency matrix, $\mathbf{T}$, can be, naturally, used instead of the adjacency matrix, $\mathbf{A}$.

The total discounted number of routes from any node of type i to all nodes of type j (where $i, j = 1, 2, \dots, S$; $i \neq j$) is equal to

$$\tilde{m}_{ij} = \sum_{k=1}^{\infty} \alpha^k t_{ij}^k.$$

A fictitious "zero-length" route is interpreted now as a route from a node of type i to nodes of the same type. Accounting for this route, we obtain the following formula for the discounted number of routes from any node of type i to all nodes of the same type:

$$\bar{\tilde{m}}_{ii} = \sum_{k=0}^{\infty} \alpha^k t_{ij}^k,$$

Hence, the Bonacich centrality measure of any node of type i is equal to

$$\tilde{C}_B(i) = \sum_{j=1}^{S} \bar{\tilde{m}}_{ij}.$$

If node i_0 is of type i, then $C_B(i_0) = \tilde{C}_B(i)$. We come to the formula for the matrix of summary discounted numbers of routes between any nodes of type i and all nodes of type j $(i, j = 1, 2, \dots, S)$:

$$\bar{\tilde{\mathbf{M}}} = \sum_{k=0}^{\infty} (\alpha\mathbf{T})^k = (\tilde{\mathbf{I}} - \alpha\mathbf{T})^{-1},$$

where $\tilde{\mathbf{I}}$ is the $S \times S$ unit matrix.

The vector of Bonacich centralities for types is

$$\tilde{\mathbf{C}}_B = \begin{pmatrix} \tilde{C}_B(1) \\ \tilde{C}_B(2) \\ \dots \\ \tilde{C}_B(S) \end{pmatrix} = \bar{\tilde{\mathbf{M}}} \cdot \tilde{\mathbf{1}} = (\tilde{\mathbf{I}} - \alpha\mathbf{T})^{-1} \cdot \tilde{\mathbf{1}},$$

where $\bar{\bar{\mathbf{M}}}, \tilde{\mathbf{I}}, \mathbf{T}$ are $S \times S$-matrices, and $\tilde{\mathbf{1}}$ is the S-vector of all ones. Correspondingly, the vector of Katz centralities for types is

$$\tilde{\mathbf{C}}_K = ((\tilde{\mathbf{I}} - \alpha\mathbf{T})^{-1} - \tilde{\mathbf{I}})\tilde{\mathbf{1}}.$$

As an example, let us calculate the vector of Bonacich centralities for the star graph with ν peripheral nodes. The type adjacency matrix is

$$\mathbf{T} = \begin{pmatrix} 0 & \nu \\ 1 & 0 \end{pmatrix},$$

hence,

$$(\tilde{\mathbf{I}} - \alpha\mathbf{T}) = \begin{pmatrix} 1 & -\alpha\nu \\ -\alpha & 1 \end{pmatrix},$$

$$(\tilde{\mathbf{I}} - \alpha\mathbf{T})^{-1} = \frac{1}{1-\alpha^2\nu}\begin{pmatrix} 1 & \alpha\nu \\ \alpha & 1 \end{pmatrix},$$

$$\tilde{\mathbf{C}}_B = (\tilde{\mathbf{I}} - \alpha\mathbf{T})^{-1} \cdot \tilde{\mathbf{1}} = \frac{1}{1-\alpha^2\nu}\left(1+\alpha\nu; 1+\alpha\right)^T.$$

The positive values of centralities are received under values of parameter α sufficiently small in comparison with the size of the star:

$$\alpha < \frac{1}{\sqrt{\nu}}.$$

In other words, the parameter α has to be lower than the inverse of the Frobenius eigenvalue of matrix $\mathbf{T}$.

We see that the Bonacich centrality of the center of the star is $(1+\alpha\nu)/(1+\alpha)$ times higher than the Bonacich centrality of the peripheral node. If $\alpha \to 1/\sqrt{\nu}$, i.e. if α converges to the inverse of the Frobenius eigenvalue, the ratio of the Bonacich centralities of the center and the periphery converges to $\sqrt{\nu}$, i.e. to the ratio of their eigenvalue centralities.

The following two theorems show that the type adjacency matrix $\mathbf{T}$ may be used instead of the adjacency matrix $\mathbf{A}$ for calculation of *eigenvalue centralities* and α *– centralities*.

Let the first type have n_1 nodes, the second type have n_2 nodes, ..., the S-th type have n_S nodes, so that

$$n = n_1 + n_2 + \ldots + n_S.$$

Let the nodes of graph G be numbered in the following way. The nodes of the first type are numbered from 1 to n_1, the nodes of the second type – from $n_1 + 1$ to $n_1 + n_2$, ..., the nodes of the S-th type – from $n_1 + n_2 + \ldots + n_{S-1} + 1$ to n.

Theorem 1. *1. Let λ be an eigenvalue of the type adjacency matrix $\mathbf{T}$ and $\tilde{\mathbf{g}}$ be corresponding eigenvector*

$$\tilde{\mathbf{g}} = (\beta_1; \beta_2; \ldots; \beta_S)^T.$$

Then λ is also an eigenvalue of the adjacency matrix $\mathbf{A}$ and the corresponding eigenvector is

$$\mathbf{g} = (\beta_1; \beta_1; \ldots; \beta_1; \beta_2; \beta_2; \ldots; \beta_2; \ldots; \beta_S; \beta_S; \ldots; \beta_S)^T$$

(where β_i is repeated n_i times; $i = 1, 2, \ldots, S$)).

2. Let λ^* be the Frobenius eigenvalue of matrix $\mathbf{T}$. Then λ^* is also the Frobenius eigenvalue of matrix $\mathbf{A}$.

Proof. 1. For any $i = 1, 2, \ldots, S$ we have for i-th row of matrix $\mathbf{T}$:

$$\beta_1 l_1(i) + \beta_2 l_2(i) + \ldots + \beta_S l_S(i) = \lambda \beta_i.$$

Then for any $i = 1, 2, \ldots, S$ and $j = 1, 2, \ldots, n_i$ we obtain for the $(n_1+n_2+\ldots+n_{i-1}+j)$-th row of matrix $\mathbf{A}$ the equality

$$\underbrace{\beta_1 + \beta_1 + \ldots + \beta_1}_{l_1(i)} + \underbrace{\beta_2 + \beta_2 + \ldots + \beta_2}_{l_2(i)} + \ldots + \underbrace{\beta_S + \beta_S + \ldots + \beta_S}_{l_S(i)} = \lambda \beta_i.$$

2. Assume that the Frobenius eigenvalue of matrix $\mathbf{A}$ is $\mu \neq \lambda^*$. The first part of the theorem implies that $\mu > \lambda^*$. Let $\mathbf{e}$ be the Frobenius eigenvector of matrix $\mathbf{A}$, and $\hat{\mathbf{e}}$ be the vector constructed in the following way. Each of the first n_1 components of vector $\hat{\mathbf{e}}$ is equal to the maximum of the first n_1 components of vector $\mathbf{e}$; each of the next n_2 components of $\hat{\mathbf{e}}$ is equal to the maximum of the corresponding n_2 components of $\mathbf{e}$;...; each of the last n_S components of $\hat{\mathbf{e}}$ is equal to the maximum of the last n_S components of $\mathbf{e}$. Let $\hat{\mathbf{f}}$ be the S-vector corresponding to $\hat{\mathbf{e}}$ (i.e. i-th component of $\hat{\mathbf{f}}$ ($i = 1, 2, \ldots, S$) is equal to the maximum of n_i corresponding components of $\mathbf{e}$). Evidently,

$$\mathbf{A}\hat{\mathbf{e}} \geq \mu \hat{\mathbf{e}}.$$

Correspondingly,

$$\mathbf{T}\hat{\mathbf{f}} \geq \mu \hat{\mathbf{f}}. \tag{2}$$

But, according to the Perron-Frobenius theorem, since λ^* is the Frobenius eigenvalue, (2) implies that $\lambda^* \geq \mu$. Contradiction!

□ □

Theorem 2. *Let* $\mathbf{A}$ *be adjacency matrix,* $\mathbf{T}$ *be type adjacency matrix,* α *be any number,* $\tilde{\mathbf{e}} = (e_1; e_2; \ldots; e_S)^T$ *be any* S*-vector and let the matrix* $\tilde{\mathbf{I}} + \alpha\mathbf{T}$ *be invertible. Then the matrix* $\mathbf{I} + \alpha\mathbf{A}$ *is also invertible. Let* κ *be the* S*-vector of* α*-centralities,*

$$\kappa = (\tilde{\mathbf{I}} + \alpha\mathbf{T})^{-1}\tilde{\mathbf{e}}. \tag{3}$$

Let $\mathbf{e}$ *be the* n*-vector,* n_1 *first components of which are equal to the component* e_1*;* n_2 *following components are equal to* e_2*; ...; the last* n_S *components are equal to* e_S*. Let* $\mathbf{k}$ *be the* n*-vector of* α*-centralities,*

$$\mathbf{k} = (\mathbf{I} + \alpha\mathbf{A})^{-1}\mathbf{e}. \tag{4}$$

Then the first n_1 *components of vector* $\mathbf{k}$ *are equal to* κ_1*; the next* n_2 *components are equal to* κ_2*; ...; the last* n_S *components are equal to* κ_S*.*

Proof. The system of equations (3) is equivalent to

$$\kappa + \alpha\mathbf{T}\kappa = \tilde{\mathbf{e}}. \tag{5}$$

Writing the system (5) for individual components, we obtain

$$\kappa_i = \alpha(l_1(i)\kappa_1 + l_2(i)\kappa_2 + \ldots + l_S(i)\kappa_S) = \tilde{e}_i, \tag{6}$$

where $i = 1, 2, \ldots, S$. But for each i equation (6) coincides with each of the equations with numbers from $n_1 + n_2 + \ldots + n_{i-1} + 1$ until $n_1 + n_2 + \ldots + n_i$ of the system

$$\mathbf{k} + \alpha\mathbf{A}\mathbf{k} = \mathbf{e},$$

which is equivalent to system (4). The invertibility of the matrix $\mathbf{I} + \alpha\mathbf{A}$ follows from invertibility of the matrix $\tilde{\mathbf{I}} + \mathbf{T}$. □ □

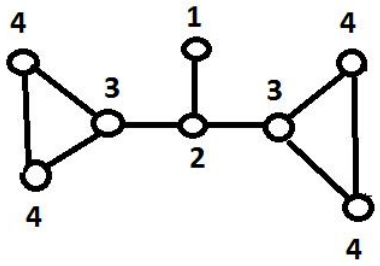

Figure 9: Graph with 2 peripheral components ($c = 2$).

3.3 Growth of a core-periphery network

The graph shown in Figure 9 has the following structure: the central node (type 2), two symmetric peripheral components (with nodes of types 3 and 4) and a node close to the center (type 1). This graph can be described either by the adjacency matrix $\mathbf{M}$ of order 8 or by the type adjacency matrix of order 4:

$$\mathbf{T} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Any of matrices $\mathbf{M}$ and $\mathbf{T}$ can be used for calculation of the eigenvector centralities (which are approximately proportional to 0.25, 0.61, 0.61, 0.43 for the types 1, 2, 3, 4, correspondingly).

Now, let the number of symmetric peripheral components, c, increase (Figure 10). The type adjacency matrix becomes

$$\mathbf{T}(c) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & c & 0 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Despite the increase of the order of the adjacency matrix, $(3c + 2) \times (3c + 2)$, the order of the type adjacency matrix, 4×4, does not change; moreover, the only element which changes in the type adjacency matrix is $c = l_3(2)$ – the number of the 3^{rd} type nodes.

Below we list the eigenvectors of the type adjacency matrices with the numbers of peripheral components $c = 2, 3, 4, 5, 6$.

$$\mathbf{x}(2) = \begin{pmatrix} 1 \\ 2.44 \\ 2.44 \\ 1.72 \end{pmatrix}, \mathbf{x}(3) = \begin{pmatrix} 1 \\ 2.67 \\ 1.96 \\ 1.22 \end{pmatrix}, \mathbf{x}(4) = \begin{pmatrix} 1 \\ 2.82 \\ 1.71 \\ 0.96 \end{pmatrix}, \mathbf{x}(5) = \begin{pmatrix} 1 \\ 2.96 \\ 1.54 \\ 0.79 \end{pmatrix}, \mathbf{x}(6) = \begin{pmatrix} 1 \\ 3.07 \\ 1.43 \\ 0.68 \end{pmatrix}.$$

It is interesting to observe how the eigenvector centrality of the center (type 2) increases with respect to c, and how at the same time the relative centrality of type 1 (which is a neighbor of the center) increases in relation not only to type 4 but even to type 3. The type adjacency matrix $\mathbf{T}$ allows a simple and visible analysis without any increase in the order of the matrix.

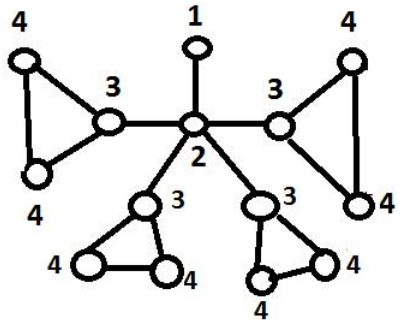

Figure 10: Graph with 4 peripheral components ($c = 4$).

Theorem 3. *With increase of c, if the eigenvector centrality of the* 1^{st} *type node is taken constant equal to 1, then the centrality of the* 2^{nd} *type node increases slower than* $\sqrt{c}$ *and goes to infinity; the centrality of each* 3^{rd} *type node decreases and converges to 1; the centrality of each* 4^{th} *type node decreases faster than* $1/\sqrt{c}$ *and converges to 0.*

Proof. The characteristic equation of the matrix $\mathbf{T}$ is

$$(\lambda - 1)[\lambda^3 - (c+3)\lambda - 2] = 0.$$

To the eigenvalue $\lambda = 1$ the eigenvector $(1, 1, 0, -1/2)^T$ corresponds. The equation for other three eigenvalues is

$$\lambda^3 - (c+3)\lambda - 2 = 0. \tag{7}$$

This equation is of the form

$$\lambda^3 + 3p\lambda + 2q = 0,$$

where

$$p = -\frac{c+3}{3},\ q = -1.$$

Since the discriminant is

$$D = q^2 + p^3 = 1 - \frac{(c+3)^3}{27} < 0$$

(because $c \geq 2$), the equation has three real roots:

$$\lambda_1 = -2rcos\frac{\phi}{3},\ \lambda_2 = 2rcos\frac{\pi - \phi}{3},\ \lambda_3 = 2r\frac{\pi + \phi}{3},$$

where $r = \pm\sqrt{|p|} = \pm\sqrt{(c+3)/3}$ (the sign of r is chosen to coincide with the sign of q); the angle ϕ is determined by the relation

$$cos\phi = \frac{q}{r^3} = \left(\frac{3}{c+3}\right)^{3/2}.$$

Under $c = 2:\ \phi = \arccos(3/5)^{3/2} \approx 1.087$. With increase in c the angle ϕ also increases and $\lim_{c\to\infty}\phi = \pi/2$. Thus, of the three roots of the equation, only the first one is positive:

$$\lambda_1 = 2\sqrt{\frac{c+3}{3}}\cos\left(\frac{1}{3}\arccos\left(\frac{3}{c+3}\right)^{3/2}\right). \tag{8}$$

Under $c = 2$:

$$\lambda_1 = 2\sqrt{\frac{5}{3}} \cos\left(\frac{1}{3} \arccos\left(\frac{3}{5}\right)^{\frac{3}{2}}\right) \approx 2.414 > 1;$$

with increase in c the value λ_1 also increases and $\lim_{c\to\infty} \lambda_1 = +\infty$. Thus, under any c the Frobenius eigenvalue of matrix $\mathbf{T}(c)$ is given by (8). Let us find the general form of the dependence of the coordinates of the eigenvectors corresponding the eigenvalues obtained from equation (7). The eigenvector is defined by the equation

$$\mathbf{T}(c)\mathbf{X} = \lambda\mathbf{X},$$

or, in coordinate form:

$$\begin{cases} \lambda x_1 = x_2, \\ x_1 + cx_3 = \lambda x_2, \\ x_2 + 2x_4 = \lambda x_3, \\ x_3 + (1-\lambda)x_4 = 0. \end{cases}$$

Solving these system of equations and using the fact that $\lambda \neq 1$, we find the eigenvector corresponding to the eigenvalue λ:

$$\mathbf{X} = \left(1, \lambda, \frac{\lambda^2 - 1}{c}, \frac{\lambda + 1}{c}\right)^T.$$

If λ is given by (7), then the components of the vector $\mathbf{X}$ are the values of eigenvector centrality of the nodes of the corresponding types. This implies the demanded result. □
□

4 Game equilibria in a model of production and externalities in network

One more field of application of the type adjacency matrix is analysis of game equilibria in economic networks. An example is the network model of production with knowledge externalities [14], [15].

Behavior of agents in a network structure is defined in much by actions of their neighbors or by information received from them. Network economics and network games theory consider questions of network formation, spreading of information in networks, positive and negative externalities, complementarity and substitutability of activities (see reviews [9], [7], [10]). Externalities, i.e. influence of other agents which does not go through the price mechanism, possess properties of public goods and are not fully paid. Positive externalities, and among them externalities of knowledge and human capital, spring up both in processes of production [16], [13] and consumption [3], and it is important to account for them in economic and sociological analysis, forecasting, and mechanism design.

Matveenko and Korolev [14], [15] continue the line of research of Nash equilibria in networks in presence of positive externalities and introduce several new elements in comparison to the previous literature. Firstly, production externalities are studied; agents' efforts have meaning of investments, in particular, investments into knowledge. The presence of production block allows to interpret game-theoretic concepts of strategic complementarity (supermodularity) and strategic substitutability (submodularity) as, correspondingly, absence and presence of productivity and to analyse these concepts within the same model.

Secondly, the model, for the first time in the network literature, uses the notion of the Jacobian production externality [11], [16], [13] in definition of the concept of game equilibrium. The essence of this notion is that any agent makes her decision staying in a particular environment which depends on actions produced by the agent herself and by her neighbors in network. When making her decision, the agent considers the state of the environment as exogenous; this means that the agent does not take into account that her actions can directly influence the state of the environment. As a simplest example, imagine a game equilibrium in a collective of smokers and non-smokers. A smoker, when making in equilibrium a decision to continue or to give up smoking, makes it staying in an environment relating already to her smoking.

The third novation of the model is the use of dynamic approach. Essentially, the model is a network generalization of the simple two-period model of endogenous growth and knowledge externalities of Romer [16].

We will see that an important place in the model is played by the typology of nodes described above. The typology defines behavior of agents in equilibrium and allows to consider a possibility of transplantation of equilibrium among networks of different size but the same typology.

4.1 Description of the model

There is an undirected graph (network), G, with n nodes $i = 1, 2, \ldots, n$; each node represents an agent. In time period 1 each agent i possesses endowment e of good and can use it partly (or wholly) for consumption in the 1^{st} period, c_i^1, and partly (or wholly) for investment into knowledge, k_i. The investment is immediately transformed into the stock of knowledge and is used in production of good for consumption in the 2^{nd} period, c_i^2. Agent's preferences are described by quadratic utility function

$$U(c_i^1, c_i^2) = c_i^1(e - ac_i^1) + dc_i^2,$$

where a is a satiation coefficient; $d > 0$. It is assumed that under $c_i^1 \in [0, e]$ the utility increases in c_i^1. These assumptions imply that $0 < a < 1/2$. The production in node i is described by function

$$F(k_i, K_i) = gk_iK_i \ (g > 0),$$

which depends on the state of knowledge, k_i, and the environment, K_i. The environment, by definition, is the sum of investments of the agent herself and her neighbors (the agents in the adjacent nodes of the graph, $j \in N(i)$).

Since increase in each of the parameters d and g promotes increase in the 2^{nd} period consumption, we denote $dg = b$ and talk about parameter b as a productivity. We assume $b > a$. If $b > 2a$ we say that *productivity presents*, and if $b < 2a$ we say that *productivity absents*.

Let us consider the following game. Players are the agents $i = 1, 2, \ldots, n$. Strategies of player i are her feasible volumes of investment, $k_i \in [0, e]$. *Nash equilibrium with externalities* (for shortness, *equilibrium* is a profile of players' strategies, $k_1^*, k_2^*, \ldots, k_n^*$, such that each k_i^* solves the agent's problem:

$$\max_{c_1^i, c_2^i, k_i} U(c_i^1, c_i^2)$$

s.t.

$$c_i^1 \leq e - k_i,$$
$$c_i^2 \leq F(k_i, K_i),$$

$$c_i^1 \geq 0, c_i^2 \geq 0, k_i \geq 0,$$

given $K_i = k_i^* + \tilde{K}_i$, where $\tilde{K}_i = \Sigma_{j \in N(i)} k_j^*$ is the sum of investments by the player i's neighbors in the network. If $k_i^* \in (0, e), i = 1, 2, \ldots, n$, the equilibrium is called *inner*.

The first order conditions imply (see details in [14], [15]) that the inner equilibrium (when it exists given values of parameters) is uniquely defined by the system of equations

$$(b - 2a)\mathbf{k} + b\mathbf{A}\mathbf{k} = e(1 - 2a)\mathbf{1},$$

where $\mathbf{k} = (k_1, k_2, \ldots, k_n)^T$, $\mathbf{A}$ is the adjacency matrix of graph G, $\mathbf{1}$ is the vector of all ones. It follows that

$$k^* = (\mathbf{I} - \alpha\mathbf{A}^{-1})\tilde{\mathbf{e}},$$

where $\alpha = b/(2a - b)$, $\tilde{\mathbf{e}} = [e(1 - 2a)/(b - 2a)]\mathbf{1}$. Thus, agents' strategies are defined by their α-centralities in the network.

4.2 Usage of the type adjacency matrix in analysis of game equilibria

An alternative way to find the inner equilibrium is by use of the type adjacency matrix. We have

$$(b - 2a)\hat{\mathbf{k}} + b\mathbf{T}\hat{\mathbf{k}} = e(1 - 2a)\hat{\mathbf{1}}, \tag{9}$$

where $\hat{\mathbf{k}} = (\hat{k}_1, \hat{k}_2, \ldots, \hat{k}_s)^T$ is the vector of investments by types, $\hat{\mathbf{1}}$ is the S-vector with all ones. This implies

$$\hat{\mathbf{k}}^* = (\hat{\mathbf{I}} - \alpha\mathbf{T}^{-1})\hat{\mathbf{e}},$$

where $\alpha = b/(2a - b)$, $\hat{\mathbf{e}} = [e(1 - 2a)/(b - 2a)]\hat{\mathbf{1}}$.

In the inner equilibrium (which is unique) all agents of the same type use the same strategy, i.e. make the same investment (defined by the α-centrality of the type). Moreover, if two networks are characterized by the same type adjacency matrix $\mathbf{T}$, then their inner equilibria do coincide, in the sense that agents in the nodes of the same type make the same investment.

Thus, the characterization of the inner equilibrium in terms of the types of nodes is especially important because it shows a possibility of transplantation of the inner equilibrium from one network into another network with different size but the same typology.

For example, in Figure 11 three graphs are shown which have different sizes but the same typology: the type adjacency matrix is

$$\mathbf{T} = \begin{pmatrix} 1 & 2 \\ 2 & 0 \end{pmatrix}.$$

Transplantation of equilibrium among these graphs is possible.

Generally, let there be two types of nodes characterized by the type adjacency matrix

$$\mathbf{T} = \begin{pmatrix} s_1 & s_2 \\ t_1 & t_2 \end{pmatrix}.$$

Then (9) implies the system of linear equations

$$\begin{cases} (b - 2a + s_1 b)k_1 + s_2 b k_2 = e(1 - 2a), \\ t_1 b k_1 + (b - 2a + t_2 b)k_2 = e(1 - 2a), \end{cases}$$

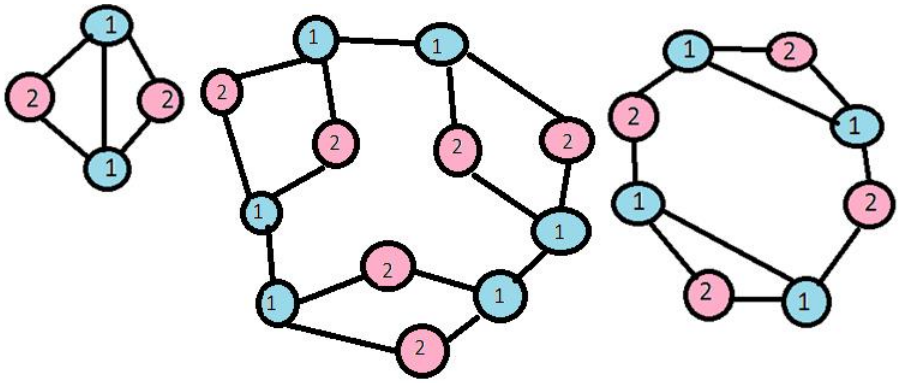

Figure 11: Three graphs with the same typology.

where k_1, k_2 are investments of the types. The solution of the system is the pair

$$k_1 = \frac{e(1-2a)[b-2a+(t_2-s_2)b}{(b-2a)^2+(s_1+t_2)(b-2a)b+(s_1t_2-t_1s_2)b^2}, \tag{10}$$

$$k_2 = \frac{e(1-2a)[b-2a+(s_1-t_1)b}{(b-2a)^2+(s_1+t_2)(b-2a)b+(s_1t_2-t_1s_2)b^2}, \tag{11}$$

If $0 < k_i < e,\ i = 1, 2$, then the values k_1, k_2 define the inner equilibrium of the game.

Here we limit ourselves by several examples of the networks with two types of nodes.

For the chain of four nodes (with the order of types: 2–1–1–2) the type adjacency matrix is

$$\mathbf{T} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}.$$

Formulas (10)-(11) take the form

$$k_1 = \frac{2ae(1-2a)}{6ab-4a^2-b^2},$$

$$k_2 = \frac{e(1-2a)(2a-b)}{6ab-4a^2-b^2}.$$

The conditions of inner equilibrium $0 < k_i < e, i = 1, 2$ are fulfilled under absence of productivity ($b < 2a$.)

A generalization of the previous case is a fan, i.e. a dyad to each node of which a bundle of ν hanging nodes is adjoined. The type adjacency matrix of the fan is

$$\mathbf{T} = \begin{pmatrix} 1 & \nu \\ 1 & 0 \end{pmatrix}.$$

An important example of network with two types of nodes is the star network; let us remind that its type adjacency matrix is

$$\mathbf{T} = \begin{pmatrix} 0 & \nu \\ 1 & 0 \end{pmatrix}.$$

Equations (9)-(10) turn into

$$k_1 = \frac{e(1-2a)[(\nu-1)b+2a]}{\nu b^2-(b-2a)^2},$$

$$k_2 = \frac{2ea(1-2a)}{\nu b^2 - (b-2a)^2}.$$

The pair k_1, k_2 defines inner equilibrium if $0 < k_i < e, i = 1, 2$, i.e. if

$$\begin{cases} \nu b^2 - (b-2a)^2 > 0, \\ \nu b^2 - (b-2a)^2 > (1-2a)[(\nu-1)b + 2a], \\ \nu b^2 - (b-2a)^2 > 2a(1-2a). \end{cases}$$

The third and the first inequalities follow from the second one, and the latter is fulfilled for all ν if $b + 2a > 1$ and $b > (-6a + 1 + \sqrt{36a^2 - 4a + 1})/2$.

The following proposition identifies agents interested in growth of the star network.

Theorem 4. *In star network, if the number of peripheral nodes, ν , increases, then knowledge and utility in the central node decrease under absence of productivity, but increase under presence of productivity. Knowledge and utility in each peripheral node always decrease.*

Proof. Derivative of k_1 in ν (if ν is considered as a continuous variable) is

$$\frac{2bae(1-2a)(b-2a)}{[(b-2a)^2 - \nu b^2]^2}.$$

Hence, knowledge in the central node decreases in ν if $b < 2a$ and increases if $b > 2a$. It is directly seen that k_2 decreases in ν. According to Theorem 2.2 in [15], utility increases in knowledge. □ □

Acknowledgements

This research is supported by the Russian Foundation for Basic Research (project 17-06-00618).

References

[1] Allouch, N. (2016) 'Aggregation in networks', *Queen Mary University of London. School of Economics and Finance.* Available: http://webspace.qmul.ac.uk/nallouch/Modulardecomposition36.pdf

[2] Atik, F., Panigrahi, P. (2015) 'Graphs with few distinct distance eigenvalues irrespective of the diameters', *Electronic J. of Lin. Algebra*, Vol. 29, pp.194–205.

[3] Azariadis, C., Chen, B.-L., Lu, C.-H., Wang, Y.-C. (2013) 'A two-sector model of endogenous growth with leisure externalities', *J. Econ. Theory*, Vol. 148, pp. 843–857.

[4] Bonacich, P. (1972) 'Factoring and weighting approaches to status scores and clique identification', *J. of Math. Sociology*, Vol. 2, pp. 113—120.

[5] Bonacich, P., Lloyd, P. (2001) 'Eigenvector-like measures of centrality for asymmetric relations', *Social Networks*, Vol. 23, pp. 191–201.

[6] Brouwer, A.E., Haemers W.H. (2012) *Spectra of graphs*, Springer, New York.

[7] Galeotti, A., Goyal, S., Jackson, M.O., Vega–Redondo, F., Yariv, L. (2010) 'Network games', *Rev. Econ. Stud.*, Vol. 77, pp. 218–244.

[8] Goyal, S. (2009) *Connections: An Introduction to the Economics of Networks*, Princeton University Press, Princeton.

[9] Jackson, M.O. (2008) *Social and economic networks*, Princeton University Press, Princeton.

[10] Jackson, M.O., Zenou, Y. (2015) 'Games on networks'. In: Young P. and Zamir S. eds. *Handbook of game theory. Vol. 4*, Elsevier Science, Amsterdam.

[11] Jacobs, J. (1969) *The economy of cities*, Random House, New York.

[12] Katz, L. (1953) 'A New Status Index Derived from Sociometric Index', *Psychometrika*, pp. 39–43.

[13] Lucas, R. E. (1988) 'On the mechanics of economic development', *J. Monetary Econ.*, Vol. 22, pp. 3–42.

[14] Matveenko, V., Korolev, A. (2016) 'Equilibria in networks with production and knowledge externalities'. In: Kalyagin V.A., Koldanov P.A., Pardalos P.M., eds. Models, algorithms and technologies for network analysis. *Springer Proceedings in Mathematics and Statistics*, Vol. 156. Springer, pp. 291-331.

[15] Matveenko, V., Korolev, A. (2017) 'Knowledge externalities and production in network: game equilibria, types of nodes, network formation', *Int. J. of Computational Economics and Econometrics*. Forthcoming.

[16] Romer, P. M. (1986) 'Increasing returns and long-run growth', *J. Polit. Econ.*, Vol. 94, pp. 1002–1037.

On critically 3-connected graphs with exactly two vertices of degree 3

*A. V. Pastor**

Steklov Institute of Mathematics at St.Petersburg, Russia
Peter the Great Saint-Petersburg Polytechnic University

Abstract

A graph G is *critically* 3*-connected* if G is 3-connected and $G - v$ isn't 3-connected for any $v \in V(G)$.

It was proved in [2] that every critically 3-connected graph contains at least two vertices of degree 3. Some generalizations of this result for k-connected graphs could be found in [1, 3, 6]. In [2, 3] some examples of critically 3-connected graphs with exactly two vertices of degree 3 were presented. In this paper we describe all such graphs.

1 Some general facts

Let $G(V, E)$ be a critically 3-connected graph and $u, v \in V$ are only two vertices of G of degree 3. We use following notations.

$$\mathfrak{M}_i(G) = \{S \mid S \subset V \cup E, |S| = 3, |S \cap E| = i \text{ and graph } G - S \text{ is disconnected}\},$$

$\mathfrak{M}(G) = \cup_{i=1}^{3} \mathfrak{M}_i(G)$ and
$\mathfrak{M}^+(G) = \cup_{i=0}^{3} \mathfrak{M}_i(G)$. We say that S is a *cut* if $S \in \mathfrak{M}(G)$ and we say that S is a *cutset* if $S \in \mathfrak{M}_0(G)$. It's obvious that any vertex of G is contained contains in at least one cutset.

For any cut $S \in \mathfrak{M}(G)$ we denote $V_0(S) = S \cap V$ and $V(S)$ — the set of all vertices which are contained in S or incident to an edge from S. We say that a set $S \in \mathfrak{M}^+(G)$ is *maximal* if it's impossible to change some vertex $x \in S$ for an edge xy, such that the set $(S \setminus \{x\}) \cup \{xy\}$ is a cut. The set of all maximal cuts and cutsets we denote $\mathfrak{M}^*(G)$.

Lemma 1. *Any cutset $S \in \mathfrak{M}_0(G)$ divides G into exactly two components.*

It's easy to see that any cut $S \in \mathfrak{M}(G)$ divides G into exactly two components too. Let $S \in \mathfrak{M}^+(G)$ and H_1', H_2' be connectivity components of $G - S$. Then we denote $H_i = H_i' \cup V_0(S)$, $T_i = H_i \cap V(S)$ $(i \in \{1, 2\})$. We say that T_1, T_2 are *borders* of S and H_1, H_2 are parts of S-decomposition of G. We denote $\mathrm{Part}(S) = \{H_1, H_2\}$.

Lemma 2. *Let $S \in \mathfrak{M}_0(G)$, $x \in S$ and $d(x) = 3$. Then S don't contain a vertex, adjacent with x.*

2 The case of adjacent vertices of degree 3

Let u, v be adjacent vertices of degree 3. Then lets delete an edge uv. It's easy to see that graph $G - uv$ is 2-connected graph in which any vertex except for u, v are critical. In this case we use the construction of decomposition tree [4] (see also [5]).

*Supported by the Government of the Russian Federation (grant 14.Z50.31.0030)

Theorem 3. 1) *The decomposition tree* $\mathrm{BT}(G)$ *is a chain, which terminal parts are triangles. One of this triangles contains* u *and another contains* v.

2) *All blocks in* $\mathrm{BT}(G)$ *are* K_4 *and all cycles are* C_3 *or* C_4 *(cycles on 3 and 4 vertices respectively).*

3) *Lets call two parts in* $\mathrm{BT}(G)$ neighboring *if they have a common adjacent cutset in* $\mathrm{BT}(G)$. *Then* C_4 *can be neighboring only with* K_4 *and* C_3 *can be neighboring with* K_4 *or* C_3. *Moreover, the terminal triangles can be neighboring only with* K_4*'s.*

4) *Let* $x \in V \setminus \{u, v\}$. *Then* x *can be contained in 2 or 3 parts, this parts are consecutive and if there are 3 such parts, then the middle of them is a triangle.*

5) *If two* K_4*'s are neighboring in* $\mathrm{BT}(G)$, *than two vertices of they common cutset could be adjacent or not adjacent. In all other case two vertices of a cutset in* $\mathrm{BT}(G)$ *are adjacent.*

Corollary 4. *At that case we have* $\Delta(G) \leq 6$ *(*$\Delta(G)$ *is the maximum degree of graph vertices).*

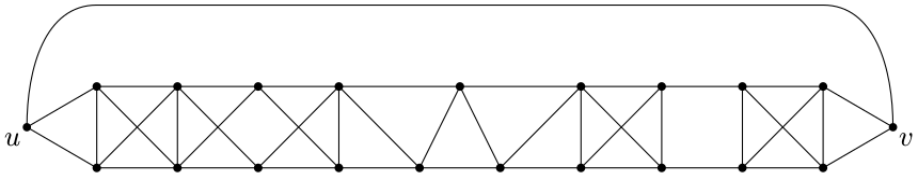

Figure 1: critically 3-connected graph with two adjacent vertices of degree 3.

An example of a such graph you can find on a picture 1.

3 The case of not adjacent vertices of degree 3

In this case we describe the structure of the graph G in terms of the set $\mathfrak{M}^*(G)$.

Lemma 5. *Let* $S, T \in \mathfrak{M}^*(G)$. *Then* $|S \cap T| \leq 1$.

Corollary 6. *All but perhaps one vertices of* T *are contained in the same part of* $H \in \mathrm{Part}(S)$.

In this case we say that T *belongs to* H.

Theorem 7. 1) *We can denote all sets from* $\mathfrak{M}^*(G)$ *by* $S_1, \ldots, S_m$ *and denote* $\mathrm{Part}(S_i) = \{L_i, R_i\}$, *such that for all* $i < j$ *the set* S_i *belongs to* L_j *and* S_j *belongs to* R_i.

2) *In that notations we can without lost of generality say that* $u \in L_1 \subset L_2 \subset \ldots \subset L_m$ *and* $R_1 \supset R_2 \supset \ldots \supset R_m \ni v$.

Lets consider two consecutive sets S_i and S_{i+1} from our sequence. Let P_i and Q_{i+1} are borders of R_i and L_{i+1} respectively. Now there are two different cases: sets S_i and S_{i+1} can have or haven't a common edge.

Theorem 8. *Let* $S_i \cap S_{i+1} = \{xy\}$, *where* $x \in Q_{i+1}$ *and* $y \in P_i$. *Then we have two possibilities.*

1. $R_i \cap L_{i+1} = P_i \cap Q_{i+1} = \{a, b\}$ *and* $ab \in E$.
2. $R_i \cap L_{i+1} = \{a, b, c, d\}$, *where* $P_i = \{a, c, y\}$, $Q_{i+1} = \{b, d, x\}$ *and* $ab, ad, cb, cd \in E$.

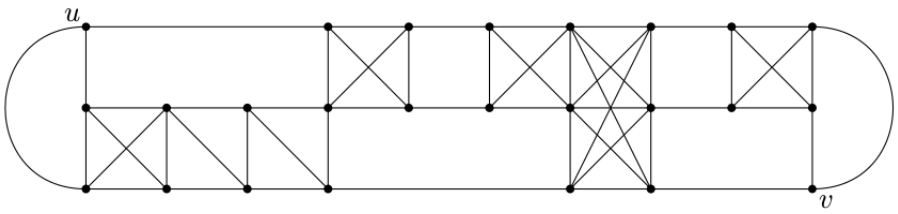

Figure 2: critically 3-connected graph with two non-adjacent vertices of degree 3.

Theorem 9. *Let S_i and S_{i+1} haven't common edges (but may have a common vertex). Then $R_i \cap L_{i+1} = P_i \cup Q_{i+1}$, $|P_i \cap Q_{i+1}| \leq 1$ and $|R_i \cap L_{i+1}| \geq 5$.*

Edge structure of subgraph in all this cases could be easily described.

Corollary 10. *At that case we have $\Delta(G) \leq 10$.*

An example of a such graph you can find on a picture 2.

References

[1] G. Chartrand, A. Kaugars and D. R. Lick,
Critically n-connected graphs,
Proc. Amer. Math. Soc. **32** (1972), 63-68.

[2] R. C. Entringer and P. J. Slater,
A theorem on critically 3-connected graphs,
Nanta Math **11** (1978), no. 2, 141-145.

[3] Y. O. Hamidoune,
On critically h-connected simple graphs,
Discrete Math. **32** (1980), 257-262.

[4] D. V. Karpov,
The tree of decomposition of a biconnected graph,
Zapiski nauchnyh ceminarov POMI, **417** (2013), 86-105.

[5] W. T. Tutte, *Connectivity in graphs*,
Toronto, Univ. Toronto Press, 1966.

[6] H. J. Veldman,
Non-κ-critical vertices in graphs,
Discrete Math **44** (1983), 105-110.

On winning shifts of generalized Thue-Morse substitutions

Jarkko Peltomäki *Ville Salo*

Turku Centre for Computer Science TUCS, Turku, Finland
University of Turku, Department of Mathematics and Statistics, Turku, Finland
r@turambar.org, vosalo@utu.fi

Abstract

The second author introduced with I. Törmä a two-player word-building game [Playing with Subshifts, Fund. Inform. 132 (2014), 131–152]. The game has a predetermined (possibly finite) choice sequence α_1, α_2, ... of integers such that on round n the player A chooses a subset S_n of size α_n of some fixed finite alphabet and the player B picks a letter from the set S_n. The outcome is determined by whether the word obtained by concatenating the letters B picked lies in a prescribed target set X (a win for player A) or not (a win for player B).

The winning shift $W(X)$ of a subshift X is defined as the set of choice sequences for which A has a winning strategy when the target set is the language of X. The winning shift $W(X)$ mirrors some properties of X. For instance, $W(X)$ and X have the same factor complexity. In this paper, we completely describe the winning shifts of subshifts generated by the generalized Thue-Morse substitutions and show that they have a substitutive structure. We show how the description can be used to derive their factor complexity functions.

Keywords: two-player game, winning shift, factor complexity, generalized thue-morse word

1 Introduction

In the paper [8], the second author introduced with I. Törmä a two-player word-building game. The two players, Alice and Bob, agree on a finite alphabet S, a target set X of words over S, game length $n \in \mathbb{N} \cup \{\mathbb{N}\}$, and a choice sequence $\alpha_1\alpha_2\cdots\alpha_n$ (a word) of integers in $\{1, 2, \ldots, |S|\}^n$. On the round j of the game, $1 \leq j \leq n$, Alice first chooses a subset S_j of S of size α_j and then Bob picks a letter a_j from the subset S_j. During the game, Alice and Bob thus together build the word $a_1a_2\cdots a_n$ (finite or infinite). If this built word is in the target set X, then Alice wins, otherwise Bob does. In other words, Alice aims to build a valid word of X while her adversary Bob attempts to introduce a forbidden word.

In studying games of this sort, it would be typical to fix a choice sequence and see what conditions on X guarantee the existence of a winning strategy for one of the players. The work of [8] adopts the opposite point of view: fix a set X and see for which choice sequences Alice has a winning strategy. This set of choice sequences, dubbed as the winning set $W(X)$ of X, turns out to be a very interesting object. First of all, if X is a subshift, then $W(X)$, now called the winning shift of X, is also a subshift, and the set of factors of $W(X)$ of length k is exactly the winning set of factors of X of length k. Actually the winning set $W(X)$ inherits many properties of X. For instance, if X is a regular language, so is $W(X)$, and if X computable, then so is $W(X)$. The most interesting result, which sparked the research in this paper, is the fact that the sets X and $W(X)$ have the same cardinality so, for a subshift X, the winning shift $W(X)$ has the same entropy and factor complexity function as X. Now the winning set $W(X)$ is in a sense simpler than X because it is downward closed: if any letter of a choice sequence in $W(X)$ is downgraded to a

smaller letter, then the resulting word is still in $W(X)$. The winning set $W(X)$ is thus a rearrangement of X to a downward closed set. Indeed, the winning set can be significantly simpler: for instance, the winning set of a Sturmian subshift is the subshift over $\{1, 2\}$ whose words contain exactly one letter 2. Overall, the winning set of X is not just some obscure set loosely related to X but a related object that mirrors properties of X.

Descriptions of the winning shifts for particular subshifts remain largely unknown. In this work, we provide such descriptions for the winning shifts of subshifts generated by generalized Thue-Morse substitutions. Let $b \geq 2$, $m \geq 1$, and σ_i be the substitution defined by $\sigma_i(k) = k1^{i-1}$. The generalized Thue-Morse substitution $\varphi_{b,m}$ is the uniform substitution defined by

$$\varphi_{b,m}(k) = k(k+1)(k+2)\cdots(k+(b-1)),$$

for $k \in \{0, 1, \ldots, m-1\}$, where the letters are intepreted modulo m. The result is that all choice sequences in the language $\mathcal{L}_{b,m}$ of the winning shift $W(\varphi_{b,m})$ of $\varphi_{b,m}$ that have length at least $2b+1$ are obtained from shorter choice sequences by substitution. Namely, if α is in $\mathcal{L}_{b,m}$ ending with a letter that is greater than 1 such that $|\alpha| > 2b$, then $\alpha = \sigma_i(\diamond)\sigma_b(w)a$ for $\diamond wa \in \mathcal{L}_{b,m}$, where $\diamond$ and a are letters and $1 \leq i \leq b$. Further, the words in $\mathcal{L}_{b,m}$ of length at most $2b$ admit a compact description.

The structure of the winning shift of $\varphi_{b,m}$ is quite easy to comprehend, and we apply our results to give a simple derivation of the first difference function of the subshift generated by $\varphi_{b,m}$. This function can in turn be used to derive the factor complexity function. Š. Starosta has derived these functions previously with methods related to so-called G-rich words [7].

2 Notation and Preliminary Results

2.1 Standard Definitions

Here we briefly define word-combinatorial notions; further details are found in, e.g., [5]. An alphabet S is a nonempty finite set of letters, and we denote by S^* the set of finite words over S. The set of words over S of length n is denoted by S^n, and by $S^{\leq n}$ we denote the set of words over S with length at most n. Infinite words over S are sequences in $S^{\mathbb{N}}$. The length of a finite word w is denoted by $|w|$, and the empty word ε is the unique word of length 0. Suppose that w is a word (finite or infinite) such that $w = uzv$ for some words u, z, and v. Then we say that z is a factor of w. If $u = \varepsilon$ (respectively $v = \varepsilon$), then we call the factor z a prefix (respectively suffix) of w. If $u = \varepsilon$ and $z \neq w$, then z is a proper prefix of w; similarly we define a proper suffix of w. We say that z occurs at position $|u|$ of w; the position $|u|$ is an occurrence of the factor z. Thus we index letters from 0. The word $\partial_{i,j}(w)$, where $i + j \leq |w|$, is obtained from the word w by deleting i letters from the beginning and j letters from the end. An infinite word is ultimately periodic if it is of the form $uvvv\cdots$; otherwise it is aperiodic.

A *subshift* X is a subset of $S^{\mathbb{N}}$ defined by some set F of forbidden words:

$$X = \{w \in S^{\mathbb{N}} : \text{no word of } F \text{ occurs in w}\}.$$

We denote by $\mathcal{L}_X(n)$ the set of words of length n occurring in words of X and define the *language* $\mathcal{L}(X)$ of X as the set $\bigcup_{n\in\mathbb{N}} \mathcal{L}_X(n)$. The subshift X is uniquely defined by its language. The function f defined by letting $f(n) = |\mathcal{L}_X(n)|$ is called the *factor complexity function of* X, and it counts the number of words of length n in the language of X. We define the *first difference function* Δ by setting $\Delta(n) = f(n) - f(n-1)$ and $\Delta(0) = 1$. This function measures the growth of the factor complexity function.

A function $\tau\colon S^* \to S^*$ is a called a *substitution* if $\tau(uv) = \tau(u)\tau(v)$ for all $u, v \in S^*$. In this paper, we typically select $S = \{0, 1, \ldots, |S| - 1\}$. If $\tau(s)$ has the same length for every $s \in S$, then we say that τ is *uniform*. We call the images of letters, the words $\tau(s)$, *τ-images*. Consider the language $\mathcal{L}$ defined as the set

$$\bigcup_{s \in S} \{w \in S^* \colon w \text{ occurs in } \tau^n(s) \text{ for some } n \geq 0\}$$

consisting of the factors of the words obtainable by applying τ repeatedly to the letters of S. Let

$$\mathcal{L}(\tau) = \{w \in \mathcal{L} \colon \text{there exists arbitrarily long words } u \text{ and } v \text{ such that } uwv \in \mathcal{L}\}.$$

The subshift generated by τ is simply the subshift with the language $\mathcal{L}(\tau)$ (i.e., we forbid the complement of $\mathcal{L}(\tau)$). The substitution τ is *primitive* if there is an integer n such that $\tau^n(s)$ contains all letters of S for every $s \in S$. The substitution τ is *aperiodic* if the subshift generated by τ does not contain ultimately periodic infinite words. We assume that all substitutions considered are aperiodic.

We say that a word w in $\mathcal{L}(\tau)$ admits an interpretation $(a_0 \cdots a_{n+1}, i, j)$ for letters a_0, …, a_{n+1} by τ if $w = \partial_{i,j}(\tau(a_0 \cdots a_{n+1}))$, $0 \leq i < |\tau(a_0)|$, $0 \leq j < |\tau(a_{n+1})|$, and $a_0 \cdots a_{n+1} \in \mathcal{L}(\tau)$. The word $a_0 \cdots a_{n+1}$ is called an *ancestor* of the word w. We say that (u_1, u_2) is a *synchronization point* of w (for τ) if $w = u_1 u_2$ and whenever $v_1 w v_2 = \tau(z)$ for some $z \in \mathcal{L}(\tau)$ and some words v_1 and v_2, then $v_1 u_1 = \tau(t_1)$ and $u_2 v_2 = \tau(t_2)$ for some words t_1 and t_2 such that $z = t_1 t_2$. We say that τ has *synchronization delay* L if every word in $\mathcal{L}(\tau)$ of length at least L has at least one synchronization point and L is minimal.

Let τ be a uniform substitution of length M with synchronization delay L. Let w in $\mathcal{L}(\tau)$ be a word such that $|w| \geq L$. Suppose that w has an ancestor z, so that $w = \partial_{i,j}(\tau(z))$ with $0 \leq i, j < M$. While w might have several ancestors, the uniformity of τ and the fact that w has at least one synchronization point ensure that the numbers i and j are independent of the chosen ancestor z. In fact, the positions i and j mark a synchronization point of w. All in all, the number i uniquely identifies the positions of w where the τ-images of the letters of any ancestor of w begin at, and we say that w has *decomposition* i mod M.

2.2 Word Games

Next we define precisely the word game in which two players, Alice and Bob, build a finite or infinite word. A *word game* is a quadruple (S, n, X, α), where S is an alphabet, $n \in \mathbb{N} \cup \{\mathbb{N}\}$, the *target set* X is a subset of S^n, and the *choice sequence* α is a word of length n over the alphabet $\{1, 2, \ldots, |S|\}$. It does not matter if the target set X contains words of distinct lengths, we may use $X \cap S^n$ in place of X; this will always be clear from context.

Denote by G the word game (S, n, X, α), and write $\alpha = \alpha_1 \cdots \alpha_n$ for letters α_i. During the round i, $1 \leq i \leq n$, of this game, first Alice chooses a subset S_i of S of size α_i. Then Bob picks a letter a_i from the set S_i. After n rounds have taken place, Alice and Bob have together built a word $a_1 a_2 \cdots a_n$. If $a_1 a_2 \cdots a_n \in X$, then Alice wins the game G and otherwise Bob does. An example is provided at the beginning of the next section, and more examples are found in [8].

Alice's *strategy* for G is a function $s\colon S^{\leq i} \to 2^S$ that specifies which subset she should choose next given the word of length i constructed so far. Similarly we define Bob's strategy as a partial function $s\colon S^{\leq i} \times 2^S \to S$ specifying which letter Bob should pick given the word constructed so far and the subset chosen by Alice. Let s_A and s_B respectively be Alice's strategy and Bob's strategy for the game G. The *play* $p(G, s_A, s_B)$ of the strategy pair

(s_A, s_B) is the word $a_1 a_2 \cdots a_n$ defined inductively by $a_{i+1} = s_B(a_1 \cdots a_i, s_A(a_1 \cdots a_i))$ with $a_1 \cdots a_0 = \varepsilon$. We say that Alice's strategy s is *winning* if $p(G, s, s_B) \in X$ for all Bob's strategies s_B (Alice wins no matter how Bob plays). Analogously Bob's strategy s is winning if $p(G, s_A, s) \notin X$ for all Alice's strategies s_A. If $n \in \mathbb{N}$ or X is a closed set in the product topology of $S^{\mathbb{N}}$ (in particular, if X is a subshift), then a winning strategy always exists for one of the players [4]. In this paper, we consider Bob's strategies only indirectly. Thus whenever we talk about a winning strategy we mean that it is Alice's winning strategy. Similarly by a *winning play* we mean a play by a strategy pair (s_A, s_B) where s_A is Alice's winning strategy.

Like mentioned in the introduction, we are interested in the choice sequences for which Alice has a winning strategy. Given a subset X of S^n, where $n \in \mathbb{N} \cup \{\mathbb{N}\}$, we define the *winning set* $W(X)$ of X as the set

$$\{\alpha \in \{1, \ldots, |S|\}^n : \text{Alice has a winning strategy for the word game } (S, n, X, \alpha)\}.$$

We often omit the alphabet S, it will be clear from the context. For a language $X \subseteq S^*$, we set

$$W(X) = \bigcup_{n \in \mathbb{N}} W(X \cap S^n)$$

and call also this set the winning set of X. If $n = \mathbb{N}$ and X is a subshift, then we call $W(X)$ the *winning shift* of X; if the subshift X is generated by a substitution τ, then we denote its winning shift by $W(\tau)$. Indeed, in [8, Proposition 3.4], the following result was proven.

Proposition 1. *If X is a subshift, then $W(X)$ is a subshift and $\mathcal{L}(W(X)) = W(\mathcal{L}(X))$.*

We abuse notation and write $W(X)$ for $\mathcal{L}(W(X))$, it is always clear from context whether we consider finite words or infinite words. In addition, we have the following observation.

Lemma 2. *Let X and Y be sets containing words of equal length. If $X \subseteq Y$, then $W(X) \subseteq W(Y)$.*

Proof. Alice's winning strategy for a word game with target set X and choice sequence in $W(X)$ is sufficient as it is for her to win in the game with the same choice sequence and target set Y. □

We endow the alphabet $\{1, \ldots, |S|\}$ with the natural order $1 < 2 < \ldots < |S|$. Suppose that u and v are words over this alphabet (finite or infinite), and write $u = u_0 \cdots u_{n-1}$ and $v = v_0 \cdots v_{m-1}$ for letters u_i, v_i. Then $u \leq v$ if and only if $n = m$ and $u_i \leq v_i$ for $i = 0, \ldots, n-1$. The winning set $W(X)$ is *downward closed* with respect to this partial ordering: if $u \leq v$ and $v \in W(X)$, then $u \in W(X)$. This is simply because *downgrading* a letter from the choice sequence only makes Bob's chances of winning slimmer.

Observe that the winning strategies for finite choice sequences ending with the letter 1 are just trivial extensions of winning strategies of shorter choice sequences ending with a letter greater than 1. Thus we define a finite choice sequence ending with 1 to be *reducible* and a choice sequence that is not reducible to simply be *nonreducible*. The infinite words of the winning shift $W(X)$ are obtainable from nonreducible choice sequences by appending infinitely many letters 1 and by taking closure. A rule of thumb for the rest of the paper is that to describe the structure of the winning sets it is enough to study only nonreducible choice sequences.

Finally, we need the next proposition [8, Proposition 5.7] that motivates the presented results.

n		n		n	
1	$\diamond$	9	$\diamond$11111112	17	$\diamond$1111111111111112
2	$\diamond$2	10	$\diamond$111111112 $\diamond$211111112	18	$\diamond$1111111111111112 $\diamond$2111111111111112
3	$\diamond$12	11	$\diamond$1111111112 $\diamond$1211111112	19	$\diamond$111111111111111112 $\diamond$121111111111111112
4	$\diamond$112 $\diamond$212	12	$\diamond$11111111112 $\diamond$11211111112	20	$\diamond$111111111111111112 $\diamond$112111111111111112
5	$\diamond$1112	13	$\diamond$111111111112 $\diamond$111211111112	21	$\diamond$1111111111111111112 $\diamond$1112111111111111112
6	$\diamond$11112 $\diamond$21112	14	$\diamond$1111111111112	22	$\diamond$1111111111111111112 $\diamond$1111211111111111112
7	$\diamond$111112 $\diamond$121112	15	$\diamond$11111111111112	23	$\diamond$1111111111111111111112 $\diamond$1111121111111111111112
8	$\diamond$1111112	16	$\diamond$111111111111112	24	$\diamond$1111111111111111111112 $\diamond$1111112111111111111112

Table 1: The nonreducible choice sequences of the winning shift of the Thue-Morse substitution for lengths 1 to 24. The letter $\diamond$ can be substituted by both of the letters 1 and 2.

Proposition 3. *If $n \in \mathbb{N}$ and $X \subseteq S^n$, then $|W(X)| = |X|$.*

We note that a subset W of $\{0,1\}^n$ can be interpreted as a family of subsets of $\{1,2,\ldots,n\}$ (a so-called set system) by considering a word $w \in \{0,1\}^n$ as the characteristic function of a subset. Proposition 3 has been proven in relation to set systems in [2].

3 Winning Shifts of Generalized Thue-Morse Substitutions

In this section, we describe the winning shifts of generalized Thue-Morse substitutions. Before giving the results in full generality, we look at the winning shift of the Thue-Morse word and explain our ideas through examples.

Let τ be the Thue-Morse substitution: $\tau(0) = 01$, $\tau(1) = 10$. The substitution τ is uniform and primitive, and it is readily proven that it is aperiodic. With an exhaustive search, it is easily established that its synchronization delay is 4 (see also Lemma 5). The fixed point at 0 is the famous Thue-Morse word, which is overlap-free (i.e., it does not contain a factor of the form $auaua$ for a word u and a letter a). For more details on the substitution τ, see for example [6, Section 2.2].

In Table 1, we list nonreducible choice sequences of $W(\tau)$ for lengths 1 to 24.[1] For the

[1] Remember that reducible choice sequences of length n are obtained by padding shorter nonreducible choice sequences with the letter 1.

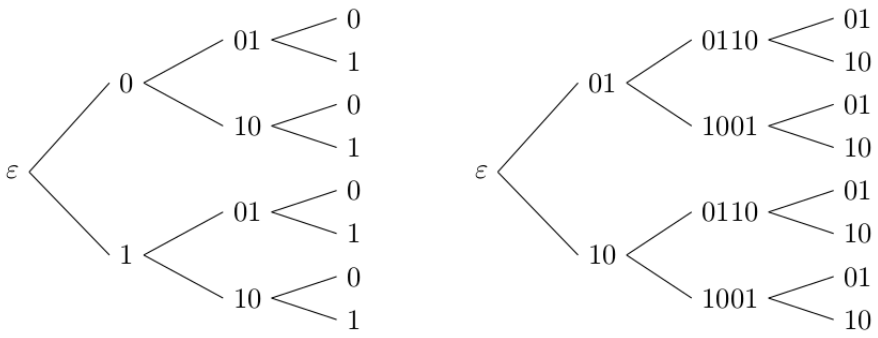

Figure 1: Winning strategies for Alice for the choice sequences 2212 and 21211121 in the case of the Thue-Morse substitution.

choice sequence 2212, Alice has the following winning strategy:

$$\begin{aligned} \varepsilon &\mapsto \{0,1\},\\ 0,1 &\mapsto \{0,1\},\\ 00,10 &\mapsto \{1\},\\ 01,11 &\mapsto \{0\},\\ 001,101,010,110 &\mapsto \{0,1\}, \end{aligned}$$

the other arguments being irrelevant. This strategy is depicted in Figure 1 as a *strategy tree*.

Table 1 contains many patterns. By Proposition 3, the number of nonreducible choice sequences of length n is counted by the first difference function $\Delta(n)$. Based on the data, it seems that $\Delta(n) \in \{2,4\}$ for all $n \geq 1$ and $\Delta(n) = 4$ only if $n = 2^k + \ell + 1$ for $k \geq 1$ and $1 \leq \ell \leq 2^{k-1}$. This is of course readily observed when looking at the factor complexity function; here we see much more: the rule described next confirms the preceding observations.

We observe that a choice sequence α in the winning shift always seems to contain at most three occurrences of 2. Moreover, if α contains exactly three occurrences of 2, then the distance between the two final occurrences is $2^k - 1$ for some $k \geq 1$, and the middle occurrence is preceded by at most 2^{k-1} occurrences of the letter 1. The rule seems to be the following. If $n = 3 \cdot 2^k + 2$, then the only nonreducible choice sequence of length n (up to the difference at the very beginning) is $\diamond 1^{3 \cdot 2^k} 2$. Then the number of 1s increases until there are $2^{k+2} - 1$ of them. Next a third occurrence of 2 can be introduced: the choice sequences of length $2^{k+2} + 2$ are $\diamond 21^{2^{k+2}-1}2$ and $\diamond 1^{2^{k+2}}2$ (the former choice sequence downgraded). Then the number of 1s before the second to last occurrence of 2 starts to grow one by one until the choice sequences considered are of length $3 \cdot 2^{k+1} + 1$, and then the pattern repeats. The observed rule suggests that nonreducible choice sequences of $W(\tau)$ of lengths $2^k + 2$ to $3 \cdot 2^k + 1$ are related to nonreducible choice sequences of lengths $2^{k+1} + 2$ to $3 \cdot 2^{k+1} + 1$. Indeed, these choice sequences look identical: the latter ones are just "blown up" by a factor of 2. Since the morphism τ also "blows up" words by a factor of 2, we proceed to look at τ-images of the strategy trees of short choice sequences.

Consider the strategy tree for the choice sequence 2212 depicted in Figure 1. Substitute all letters of this tree with τ while preserving the branch structure to obtain the right tree of Figure 1. The obtained strategy tree gives a winning strategy for Alice in a word game with choice sequence 21211121. Let us next give an intuitive explanation for the strategy

from Alice's point of view. Alice can beat Bob in the word game with choice sequence 21211121 by imagining that she plays the word game with choice sequence 2212, for which she has a winning strategy. On her first turn, Alice lets Bob choose between 0 and 1. Since Alice wins this game of length 1, Alice can also win the game of length 2 with choice sequence 21 played on the τ-images $\tau(0)$ and $\tau(1)$ (choice sequence 11 is also possible but less interesting). Continuing, Alice lets Bob again choose between 0 and 1. The win on this play of length 2 ensures Alice winning the game of length 4 with choice sequence 2121 played on the τ-images $\tau(00)$, $\tau(01)$, $\tau(10)$, $\tau(11)$. Next, Alice gives Bob only one choice to ensure a win, so Bob, having no options, loses in the game of length 6 with choice sequence 212111 played on the respective τ-images. Overall, we see that the short winning strategy for the choice sequence 2212 enables Alice to always win the game with choice sequence 21211121. This longer choice sequence is constructed in such a way that all occasions of Bob having a real choice (branches of the strategy tree) correspond to Bob having a choice of two letters in the shorter game with choice sequence 2212; Alice just imagines playing a short game with choice sequence 2212 filling the suffixes of the τ-images by not letting Bob choose. Alice's method can indeed be viewed as a branch-preserving substitution of the strategy tree.

The method described above does not explain if it is possible for Alice to obtain a winning strategy for, e.g., the choice sequence 2211121 from some shorter winning strategy. Let us see how she could do this. Alice again imagines playing the winning strategy of the word game with choice sequence 2212 using her winning strategy of Figure 1. Now, however, during the first turn Alice lets Bob pick a suffix of length 1 of the τ-images of the letters 0 and 1 (which Bob is allowed to play on the first turn of the shorter game). Continuing as above, the played word will be a suffix of a word played in the word game with choice sequence 21211121 and a suffix of a τ-image of a word played in the word game with choice sequence 2212. Therefore also $2211121 \in W(\tau)$. Similarly the play on the τ-images does not have to complete the final image, the play can be restricted to a proper prefix of the τ-images. In this particular case of the Thue-Morse substitution, it is easy to be convinced that all long enough winning strategies are obtainable by substitution by working out some example desubstitutions on strategy trees.

Not only are Alice's winning strategies substitutible and desubstitutible, but the language of the winning shift itself has a substitutive structure. Let σ be a substitution defined by $\sigma(1) = 11$ and $\sigma(2) = 21$, and let $\diamond w2$ be a nonreducible choice sequence in $W(\tau)$ for a letter $\diamond$. The result is that the words $\diamond\sigma(w)2$ and $\sigma(\diamond w)2$ are in $W(\tau)$ and that all nonreducible choice sequences of length at least 5 are obtained in this manner. Thus in our particular example it is sufficient to know all nonreducible choice sequences of $W(\tau)$ of length at most 4 to completely describe $W(\tau)$.

Next, we define the generalized Thue-Morse substitutions and describe their winning shifts, which are similar to that of the Thue-Morse substitution. Moreover, we state how the structure of the winning shifts can be used to derive known formulas for their factor complexity functions. For more on generalized Thue-Morse words, see e.g. [1].

Let $s_b(n)$ denote the sum of digits in the base-b representation of the integer n. For $b \geq 2$ and $m \geq 1$, the generalized Thue-Morse word $\mathbf{t}_{b,m}$ is defined as the infinite word whose letter at position n equals $s_b(n) \bmod m$. It is straightforward to prove that $\mathbf{t}_{b,m}$ is the fixed point, beginning with the letter 0, of the primitive substitution $\varphi_{b,m}$ defined by

$$\varphi_{b,m}(k) = k(k+1)(k+2)\cdots(k+(b-1)),$$

for $k \in \{0, 1, \ldots, m-1\}$, where the letters are intepreted modulo m. The word $\mathbf{t}_{b,m}$ is ultimately periodic if and only if $b \equiv 1 \pmod{m}$ [1]. We make the assumption that $\mathbf{t}_{b,m}$ is aperiodic.

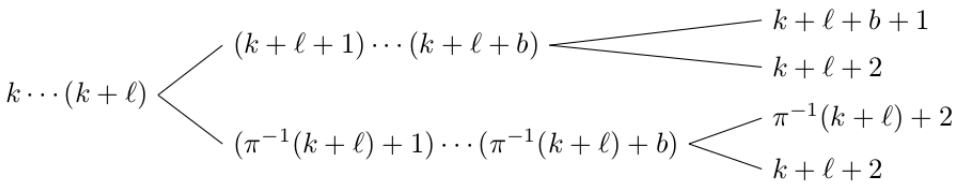

Figure 2: The subtree of Alice's winning strategy after Bob has chosen k in the game with choice sequence $\diamond 1^{\ell} 2 1^{b-1} 2$.

To clarify the notation, from now on we assume that letters are elements of the group $\mathbb{Z}_m$, so that we can naturally add letters. Moreover, we keep b and m fixed and simply write φ for $\varphi_{b,m}$.

Let $\pi \colon \mathbb{Z}_m \to \mathbb{Z}_m$ denote the permutation defined by setting $\pi(k) = k + b - 1$. In other words, the permutation π maps k to the final letter of the word $\varphi(k)$. We set q to be the order of π, that is, the least positive integer such that $q(b-1) \equiv 0 \pmod{m}$.

To describe the winning shift $W(\varphi)$ of φ, it is crucial to know words of $\mathcal{L}(\varphi)$ of length 2 and 3. The next two lemmas are respectively proven in [7] and [3].

Lemma 4. *We have*

- $\mathcal{L}_{\varphi}(2) = \{\pi^i(k-1)k \colon k \in \mathbb{Z}_m, 0 \leq i < q\}$ *and*
- $\mathcal{L}_{\varphi}(3) = \{\pi^i(k-1)k(k+1) \colon k \in \mathbb{Z}_m, 0 \leq i < q\} \cup \{(k-1)k\pi^{-i}(k+1) \colon k \in \mathbb{Z}_m, 0 \leq i < q\}$.

Lemma 5. *The substitution φ has synchronization delay $2b$.*

Let $\sigma_i \colon \{1, 2, \ldots, |S|\}^* \to \{1, 2, \ldots, |S|\}^*$ be the substitution defined by $\sigma_i(k) = k1^{i-1}$ for $k \in \{1, 2, \ldots, |S|\}$. It can be proved that every strategy tree of a winning strategy for a choice sequence in $W(\varphi)$ that has length at least $2b + 1$ (is greater than the synchronization delay) is desubstitutible. It follows, just as in the example case of the Thue-Morse substitution, that all long enough choice sequences in $W(\varphi)$ are of the form $\sigma_i(\diamond)\sigma_b(w)a$, where $\diamond wa \in W(\varphi)$ for letters $\diamond$ and a and $1 \leq i \leq b$.

Next we describe the choice sequences of length at most $2b$.

Proposition 6. *Let α in $W(\varphi)$ be a nonreducible choice sequence of length n.*

(i) If $2 \leq n \leq b+1$, then $\alpha = \diamond 1^{n-2} a$ with $\diamond \in \{1, \ldots, m\}$ and $a \in \{2, \ldots, q\}$.

(ii) If $b+2 \leq n \leq 2b$, then $\alpha = \diamond 1^{n-2} a$ or $\alpha = \diamond 1^{\ell} 2 1^{b-1} 2$ with $\diamond \in \{1, \ldots, m\}$ and $a \in \{2, \ldots, q\}$.

Proof. Consider first the case $2 \leq n \leq b+1$. Write $\alpha = \diamond ur$ with letters $\diamond$ and r, and let w be a winning play in the game with choice sequence α. First we argue that the prefix of w of length $n-1$ is of the form $k(k+1)\cdots(k+n-2)$ for some $k \in \mathbb{Z}_m$, that is, it equals $\varphi_{n-1,m}(k)$. If this were not the case, then this prefix equals $xijy$ for some words x and y and letters i and j such that $j \neq i+1$. Thus w has decomposition $|xi| \bmod b$. Since w is a winning play, Bob cannot choose inside a φ-image, and it must thus be that $|jy|$ is a positive multiple of b. This is impossible as now $|\alpha| > |xijy| \geq b+1$. Due to the restricted form of the prefix of w of length $n-1$, we see that Bob cannot make any choices between his first and last turns, so $\alpha = \diamond 1^{n-2} r$. Suppose for a contradiction that $r > q$. Now Bob can pick a letter c such that $c \notin \{\pi^i(k+n-1) \colon 0 \leq i < q\}$. It follows

that $\pi^{-1}(k+n-2)c$ is an ancestor of the played word $\varphi_{n-1,m}(k)c$. This is however a contradiction with Lemma 4. Therefore $r \leq q$. It is now clear that any word of the form $\diamond 1^{n-2}r$ with $\diamond \in \{1,\dots,m\}$ and $r \in \{2,\dots,q\}$ is in $W(\varphi)$: after Bob has chosen k, Alice forces him to play $\varphi_{n-1,m}(k)$ after which she lets him choose among the q letters c such that $\pi^{-1}(k+n-2)c$ is in $\mathcal{L}_\varphi(2)$.

Suppose then that $b+2 \leq n \leq 2b$. If α contains exactly two letters that are greater than 1, one at the beginning and one at the end, then α must again be of the form $\diamond 1^{n-2}a$ with $\diamond \in \{1,\dots,m\}$ and $a \in \{2,\dots,q\}$ (after Bob has chosen k, Alice forces him to play $\varphi_{n-b-1}(k)\varphi(\pi^{-1}(k+n-b-2)+1)$ after which she lets him choose among the q letters c such that $\pi^{-1}(k+n-b-2)(\pi^{-1}(k+n-b-2)+1)c \in \mathcal{L}(\varphi)$; see Lemma 4). Otherwise write $\alpha = \diamond urvs$ with letters $\diamond$, r, and s such that $r,s>1$, and let w again be a winning play in the game with choice sequence α. Analogous to the arguments of the preceding paragraph, we see that $|\alpha| > |\diamond urv| \geq 2b+1$ unless the prefix of w of length $|u|+1$ is of the form $\varphi_{|u|+1,m}(k)$ for some $k \in \mathbb{Z}_m$. Again, we have $u = 1^{|u|}$ and, further, $v = 1^{b-1}$. Assume for a contradiction that $r \geq 3$. After $|u|+1$ rounds Bob can choose a letter c such that $c \notin \{k+|u|+1, \pi^{-1}(k+|u|)+1\}$. Clearly the word played so far has decomposition $|u|+1 \bmod b$, so during her next $b-1$ turns Alice must let Bob complete the φ-image beginning with c. During his final turn Bob can pick a letter d such that $d \neq c+1$. It follows that the played word has the word $\pi^{-1}(k+|u|)cd$ as an ancestor. This contradicts Lemma 4, so $r=2$. The preceding arguments also show that w must have $\varphi_{|u|+1,m}(k)(k+|u|+1)$ or $\varphi_{|u|+1,m}(k)(\pi^{-1}(k+|u|)+1)$ as a prefix. Let us consider the former case. Since Bob wins if he can choose inside a φ-image, Alice must now force Bob to play $\varphi_{n-1,m}(k)$ to ensure that the word played so far has multiple ancestors. If $s \geq 3$, then as his ultimate move Bob can pick a letter c such that $c \notin \{k+n-1, \pi^{-1}(k+n-2)+1\}$. Then w has unique ancestor $\pi^{-1}(k+n-2-b)\pi^{-1}(k+n-2)c$. Our assumption that $b \neq 1$ implies by Lemma 4 that $\pi^{-1}(k+n-2)+1 = c$, which is impossible by the choice of c. Thus $s=2$, that is, $\alpha = \diamond 1^{|u|}21^{b-1}2$. It is now straightforward to derive a winning strategy for Alice for any $\diamond \in \{1,\dots,m\}$. The subtree of length $n-1$ of such a strategy is depicted in Figure 2; it is readily verified that the corresponding strategy is winning for Alice using Lemma 4. The claim follows. □

By substituting the nonreducible choice sequences of Proposition 6, we have a complete description of $W(\varphi)$.

Corollary 7. *Consider nonreducible choice sequences α of $W(\varphi)$ of length n.*

(i) If $2 \leq n \leq b+1$, then $\alpha = \diamond 1^{n-2}a$ with $\diamond \in \{1,\dots,m\}$ and $a \in \{2,\dots,q\}$.

(ii) If $n = b^{k+1}+\ell+1$ with $k \geq 0, 1 \leq \ell \leq b^{k+1}-b^k$, then $\alpha = \diamond 1^{n-2}a$ or $\alpha = \diamond 1^{\ell-1}21^{b^{k+1}-\ell-1}2$ with $\diamond \in \{1,\dots,m\}$ and $a \in \{2,\dots,q\}$.

(iii) If $n = 2b^{k+1}-b^k+\ell+1$ with $k \geq 0, 1 \leq \ell \leq b^{k+2}-2b^{k+1}+b^k$, then $\alpha = \diamond 1^{n-2}a$ with $\diamond \in \{1,\dots,m\}$ and $a \in \{2,\dots,q\}$.

Corollary 7 implies that for $n \geq 2$ the first difference function $\Delta(n)$ for $\mathbf{t}_{b,m}$ takes only two values: $(q-1)m$ and qm. Using induction, we can derive the values of $\Delta(n)$ and $C(n)$ (the factor complexity function of $\mathbf{t}_{b,m}$) for any $n \geq 1$; see Table 2. These functions have been derived by Š. Starosta with other methods [7].

4 Concluding Remarks

In this paper, we presented a description of the winning shift of a generalized Thue-Morse substitution. Results similar to those presented hold in a more general setting. If we

n	$\Delta(n)$	$C(n)$
1	$m-1$	m
$2 \leq n \leq b+1$	$(q-1)m$	$qm(n-1)-m(n-2)$
$b^{k+1}+\ell+1$ $k \geq 0, 1 \leq \ell \leq b^{k+1}-b^k$	qm	$qm(n-1)-m(b^{k+1}-b^k)$
$2b^{k+1}-b^k+\ell+1$ $k \geq 0, 1 \leq \ell \leq b^{k+2}-2b^{k+1}+b^k$	$(q-1)m$	$qm(n-1)-m(b^{k+1}-b^k+\ell)$

Table 2: The values of the first difference function $\Delta(n)$ and the factor complexity function $C(n)$ of the generalized Thue-Morse word $\mathbf{t}_{b,m}$.

consider a uniform substitution that is marked[2], then desubstituting long enough winning strategies is still possible. In this more general setting, long enough nonreducible choice sequences are essentially obtained by substitution, even though their precise form seems to elude precise description.

References

[1] Jean-Paul Allouche and Jeffrey Shallit. Sums of digits, overlaps, and palindromes. 4(1):1–10.

[2] R. P. Anstee, Lajos Rónyai, and Attila Sali. Shattering news. 18(1):59–73.

[3] Ľubomíra Balková. Factor frequencies in generalized Thue-Morse words. 48(3):371–385.

[4] David Gale and Frank M. Stewart. Infinite games with perfect information. In *Contributions to the Theory of Games*, volume 2 of *Annals of Mathematics Studies, no 28*, pages 245–266. Princeton University Press.

[5] M. Lothaire. *Algebraic Combinatorics on Words*. Number 90 in Encyclopedia of Mathematics and Its Applications. Cambridge University Press.

[6] M. Lothaire. *Combinatorics on Words*. Number 17 in Encyclopedia of Mathematics and Its Applications. Addison-Wesley.

[7] Štěpán Starosta. Generalized Thue-Morse words and palindromic richness. 48(3):361–370.

[8] Ilkka Törmä and Ville Salo. Playing with subshifts. 132:131–152.

[2] A marked substitution is a substitution such that all images of letters begin with distinct letters and end with distinct letters.

Testing isomorphism of central Cayley graphs over almost simple groups in polynomial time

Ilia Ponomarenko
(based on joint work with Andrey Vasil'ev)

St.Petersburg Department of the Steklov Mathematical Institute, St.Petersburg, Russia,
inp@pdmi.ras.ru

A Cayley graph over a group G is said to be central if its connection set is a normal subset of G. It is proved that for any two central Cayley graphs over explicitly given almost simple groups of order n, the set of all isomorphisms from the first graph onto the second can be found in time poly(n).

Cooperation in Dynamic Multicriteria Games with Random Horizons

*Anna Rettieva**

Institute of Applied Mathematical Research
Karelian Research Center of RAS
Pushkinskaya str. 11, Petrozavodsk, 185910, Russia
annaret@krc.karelia.ru

Abstract

We consider a dynamic, discrete-time, game model where the players use a common resource and have different criteria to optimize. Furthermore, the situation with incomplete information, namely when the players' planning horizons are different and random, is considered. A new approach to construct cooperative equlibrium in dynamic multicriteria games is constructed.
Keywords: dynamic games, multicriteria games, Nash bargaining solution, random planning horizons.

1 Introduction

Mathematical models involving more than one objective seem more adherent to real problems. Often players have more than one goal which are usually not comparable. These situations are typical for game-theoretic models in economic and ecology. Models with random planning horizons in exploitation processes are most appropriate for describing reality: external random factors can cause a game breach and the participants know nothing about them a priori.

In this paper we consider a dynamic, discrete-time, game model where the players use a common resource and have different criteria to optimize. First, we construct a multicriteria Nash equilibrium using the approach presented in [3]. Then, we find a multicriteria cooperative equilibrium as a solution of a Nash bargaining scheme with the multicriteria Nash equilibrium playing the role of status quo points.

2 Dynamic Multicriteria Game with Random Horizons

Consider a bicriteria dynamic game with two participants in discrete time. The players exploit a common resource and both wish to optimize two different criteria. The state dynamics is in the form

$$x_{t+1} = \varepsilon x_t - u_{1t} - u_{2t}\,, \;\; x_0 = x\,, \tag{1}$$

where $x_t \geq 0$ is the population size at time $t \geq 0$, $\varepsilon \geq 1$ denotes the natural birth rate, and $u_{it} \geq 0$ gives the catch of player i at time t, $i = 1, 2$.

Each player has two goals to optimize, they wish to maximize their profit from selling fish and minimize their catching cost. Suppose that the market price of the resource differs for both players, but their costs are identical and depend on both of players catches.

*This work was supported by the Russian Science Foundation, project no. 17-11-0107 and the Russian Foundation for Basic Research, projects no. 16-01-00183_a and 16-41-100062 p_a.

We explore a model in which the players possess heterogeneous planning horizons. By assumption, the players stop joint exploitation at random time steps: external stochastic processes can cause a game breach.

Suppose that players 1 and 2 harvest the fish stock during n_1 and n_2 steps, respectively. Here n_1 represents a discrete random variable taking values $\{1,\dots,n\}$ with the corresponding probabilities $\{\theta_1,\dots,\theta_n\}$. Similarly, n_2 is a discrete random variable with the value set and the probabilities $\{\omega_1,\dots,\omega_n\}$. We assume that the planning horizons are independent. Therefore, during the time period $[0,n_1]$ or $[0,n_2]$ the players harvest the same stock, and the problem consists in evaluating their optimal strategies.

The payoffs of the players are determined via the expectation operator:

$$J_1^1 = E\Big\{\sum_{t=1}^{n_1}\delta^t p_1 u_{1t} I_{\{n_1\le n_2\}} + \Big(\sum_{t=1}^{n_2}\delta^t p_1 u_{1t} + \sum_{t=n_2+1}^{n_1}\delta^t p_1 u_{1t}^a\Big) I_{\{n_1>n_2\}}\Big\} =$$
$$= \sum_{n_1=1}^{n}\theta_{n_1}\Big[\sum_{n_2=n_1}^{n}\omega_{n_2}\sum_{t=1}^{n_1}\delta^t p_1 u_{1t} + \sum_{n_2=1}^{n_1-1}\omega_{n_2}\Big(\sum_{t=1}^{n_2}\delta^t p_1 u_{1t} + \sum_{t=n_2+1}^{n_1}\delta^t p_1 u_{1t}^a)\Big)\Big], \tag{2}$$

$$J_2^1 = E\Big\{\sum_{t=1}^{n_2}\delta^t p_2 u_{2t} I_{\{n_2\le n_1\}} + \Big(\sum_{t=1}^{n_1}\delta^t p_2 u_{2t} + \sum_{t=n_1+1}^{n_2}\delta^t p_2 u_{2t}^a\Big) I_{\{n_2>n_1\}}\Big\} =$$
$$= \sum_{n_2=1}^{n}\omega_{n_2}\Big[\sum_{n_1=n_2}^{n}\theta_{n_1}\sum_{t=1}^{n_2}\delta^t p_2 u_{2t} + \sum_{n_1=1}^{n_2-1}\theta_{n_1}\Big(\sum_{t=1}^{n_1}\delta^t p_2 u_{2t} + \sum_{t=n_1+1}^{n_2}\delta^t p_2 u_{2t}^a)\Big)\Big], \tag{3}$$

where, for $i=1,2$, $p_i\ge 0$ is the market price of the resource for player i, $\delta\in(0,1)$ denotes the discount, u_{it}^a specifies the strategy of player i when its partner leaves the game.

And for the second criteria we suppose that if the player harvests the stock alone, then there is no cost, hence

$$J_1^2 = E\Big\{-\sum_{t=1}^{n_1}\delta^t m u_{1t}u_{2t} I_{\{n_1\le n_2\}} - \sum_{t=1}^{n_2}\delta^t m u_{1t}u_{2t} I_{\{n_1>n_2\}}\Big\} =$$
$$= -\sum_{n_1=1}^{n}\theta_{n_1}\Big[\sum_{n_2=n_1}^{n}\omega_{n_2}\sum_{t=1}^{n_1}\delta^t m u_{1t}u_{2t} + \sum_{n_2=1}^{n_1-1}\omega_{n_2}\sum_{t=1}^{n_2}\delta^t m u_{1t}u_{2t}\Big], \tag{4}$$

$$J_2^2 = E\Big\{-\sum_{t=1}^{n_2}\delta^t m u_{1t}u_{2t} I_{\{n_2\le n_1\}} - \sum_{t=1}^{n_1}\delta^t m u_{1t}u_{2t} I_{\{n_2>n_1\}}\Big\} =$$
$$= -\sum_{n_2=1}^{n}\omega_{n_2}\Big[\sum_{n_1=n_2}^{n}\theta_{n_1}\sum_{t=1}^{n_2}\delta^t m u_{1t}u_{2t} + \sum_{n_1=1}^{n_2-1}\theta_{n_1}\sum_{t=1}^{n_1}\delta^t m u_{1t}u_{2t}\Big], \tag{5}$$

where $m\ge 0$ indicates the catching cost and $\delta\in(0,1)$ denotes the discount factor.

First, we construct a multicriteria Nash equilibrium $V_i^{jN}(\tau,x)$ as step τ occurs, $i,j=1,2$, using the approach presented in [3].

Proposition 1. *The multicriteria Nash equilibrium payoffs in the problem (1)–(5) have*

the form

$$V_i^{1N}(n-k,x)=\delta^{n-k}p_i\gamma_{in-k}^N x+p_iP_{n-k}^{n-k+1}(\varepsilon-\gamma_{1n-k}^N-\gamma_{2n-k}^N)x\cdot$$
$$\cdot[\delta^{n-k+1}\gamma_{in-k+1}^N+\sum_{j=n-k+2}^{n-1}\delta^j\gamma_{ij}^N\prod_{l=n-k+2}^{j-2}P_l^{l+1}(\varepsilon-\gamma_{1l}^N-\gamma_{2l}^N)]+$$
$$+\sum_{l=1}^{k}P_{n-k}^{n-l}C_{in-l}V_i^l(n_i)\,,\ i=1,2, \tag{6}$$

where

$$V_1^l(n_1)=\sum_{n_1=n-l+1}^{n}\theta_{n_1}\delta^{n_1}p_1u_{1n_1}^a\,,\ V_2^l(n_2)=\sum_{n_2=n-l+1}^{n}\omega_{n_2}\delta^{n_2}p_2u_{2n_2}^a\,,$$
$$P_\tau^{\tau+1}=\frac{\sum\limits_{l=\tau+1}^{n}\omega_l}{\sum\limits_{l=\tau}^{n}\omega_l}\frac{\sum\limits_{l=\tau+1}^{n}\theta_l}{\sum\limits_{l=\tau}^{n}\theta_l}\,,C_{1\tau}=\frac{\omega_\tau}{\sum\limits_{l=\tau}^{n}\omega_l}\frac{1}{\sum\limits_{l=\tau}^{n}\theta_l}\,,\ C_{2\tau}=\frac{\theta_\tau}{\sum\limits_{l=\tau}^{n}\theta_l}\frac{1}{\sum\limits_{l=\tau}^{n}\omega_l}\,.$$

$$V_i^{2N}(n-k,x)=-\delta^{n-k}m\gamma_{1n-k}^N\gamma_{2n-k}^Nx^2+P_{n-k}^{n-k+1}(\varepsilon-\gamma_{1n-k}^N-\gamma_{2n-k}^N)^2x^2\cdot$$
$$\cdot[-\delta^{n-k+1}m\gamma_{1n-k+1}^N\gamma_{2n-k+1}^N-\sum_{j=n-k+1}^{n-2}\delta^{j+1}m\gamma_{1j+1}^N\gamma_{2j+1}^N(\varepsilon-\gamma_{1j}^N-\gamma_{2j}^N)^2\cdot$$
$$\cdot\prod_{l=n-k+1}^{j}P_l^{l+1}+A\prod_{l=n-2}^{j}P_l^{l+1}(\varepsilon-\gamma_{1l}^N-\gamma_{2j}^N)^2P_{n-1}^n(\varepsilon-\gamma_{1n_1}^N-\gamma_{2n_1}^N)],\ i=1,2. \tag{7}$$

The multicriteria Nash equilibrium strategies are related by

$$\gamma_{2n-k}^N=\frac{2P_{n-k}^{n-k+1}\delta^{n-k+1}m\gamma_{1n-k+1}^N\gamma_{2n-k+1}^N(p_1-p_2)+\varepsilon(K_{n-k}^1-K_{n-k}^2)}{-\delta^{n-k}mp_2-K_{n-k}^1-K_{n-k}^2}+$$
$$+\gamma_{1n-k}^N\frac{-\delta^{n-k}mp_1-K_{n-k}^1-K_{n-k}^2}{-\delta^{n-k}mp_2-K_{n-k}^1-K_{n-k}^2}, \tag{8}$$

where

$$K_{n-k}^i=2Ap_i\prod_{l=k}^{n}P_{n-l}^l(-\delta^{n-k}m\gamma_{in-k+1}^N-\sum_{l=k+1}^{n-1}\prod_{j=n-l}^{l}P_{j-1}^jm\delta^{n-l}\gamma_{il+1}^N(\varepsilon-\gamma_{1l}^N-\gamma_{2l}^N)-$$
$$-\prod_{j=n-k+1}^{n-1}P_j^{j+1}A(\varepsilon-\gamma_{1j}^N-\gamma_{2j}^N),\ i=1,2,\ A=-\frac{m\delta^n(\varepsilon^2-1)}{4}\,.$$

The strategy of player 1 at the last step (the quantity γ_{1n-1}^N) is evaluated through one of the first-order optimality conditions.

Then, we find a multicriteria cooperative equilibrium as a solution of a Nash bargaining scheme with the multicriteria Nash equilibrium playing the role of status quo points [1]. So, to construct the cooperative strategies it is required to solve the problem

$$(V_1^{1c}(1,x)+V_2^{1c}(1,x)-V_1^{1N}(1,x)-V_2^{1N}(1,x))\cdot$$
$$(V_1^{2c}(1,x)+V_2^{2c}(1,x)-V_1^{2N}(1,x)-V_2^{2N}(1,x))\to\max_{u_{1t}^c,u_{2t}^c}\,, \tag{9}$$

where $V_i^{jc}(\tau, x)$ are the cooperative payoffs as step τ occurs and $V_i^{jN}(\tau, x)$ are the noncooperative payoffs (6), (7) $(i, j = 1, 2)$.

References

[1] Mazalov, V.V., Rettieva, A.N.: Asymmetry in a cooperative bioresource management problem. In: Petrosjan, L., Mazalov V. (eds.) Game Theory and Applications, vol. 17, pp. 113–152. Nova Science Publishers (2015)

[2] Pieri, G., Pusillo, L.: Multicriteria Partial Cooperative Games. App. Math. 6(12), 2125–2131 (2015)

[3] Rettieva, A.N.: Equilibria in dynamic multicriteria games. Int. Game Theory Rev. 19(1), 1750002 (2017)

[4] Shapley, L.S.: Equilibrium points in games with vector payoffs. Naval Res. Log. Quart. 6, 57–61 (1959)

Subset Synchronization in Monotonic Automata

Andrew Ryzhikov[1,2] *Anton Shemyakov*[3]

[1] Université Grenoble Alpes, Laboratoire G-SCOP, 38031 Grenoble, France
[2] United Institute of Informatics Problems of NASB, 220012 Minsk, Belarus
[3] Belarusian State University, 220030 Minsk, Belarus
ryzhikov.andrew@gmail.com, shemyanton@gmail.com

Abstract

We study extremal and algorithmic questions of subset synchronization in monotonic automata. We show that several synchronization problems that are hard in general automata can be solved in polynomial time in monotonic automata, even without knowing a linear order of the states preserved by the transitions. We provide asymptotically tight lower and upper bounds on the maximum length of a shortest word synchronizing a subset of states in monotonic automata. We prove that the Finite Automata Intersection problem is NP-hard for monotonic weakly acyclic automata over a three-letter alphabet, and the problem of computing the rank of a subset of states is NP-hard to approximate within a factor of $\frac{9}{8} - \epsilon$ for any $\epsilon > 0$ for the same class of automata. Finally, we give a simple necessary and sufficient condition when a strongly connected digraph with a selected subset of vertices can be transformed into a deterministic automaton where the corresponding subset of states is synchronizing.

1 Introduction

Let $A = (Q, \Sigma, \delta)$ be a deterministic finite automaton (which we further simply call an *automaton*), where Q is the set of its states, Σ is a finite alphabet and $\delta : Q \times \Sigma \to Q$ is a transition function. Note that our definition of automata does not include initial and accepting states. An automaton is called *synchronizing* if there exists a word that maps every its state to some fixed state $q \in Q$. Synchronizing automata play an important role in manufacturing, coding theory and biocomputing and model systems that can be controlled without knowing their actual state [Vol08].

A set $S \subseteq Q$ of states of an automaton $A = (Q, \Sigma, \delta)$ is called *synchronizing* if there exists a word $w \in \Sigma^*$ and a state $q \in Q$ such that the word w maps each state $s \in S$ to the state q. The word w is said to *synchronize* the set S. It follows from the definition that an automaton is synchronizing if and only if the set Q of all its states is synchronizing.

An automaton $A = (Q, \Sigma, \delta)$ is *monotonic* if there is a linear order $\leq$ of its states such that for each $x \in \Sigma$ if $q_1 \leq q_2$ then $\delta(q_1, x) \leq \delta(q_2, x)$. In this case we say that the transitions of the automaton *preserve*, or *respect* this order. Monotonic automata play an important role in the part-orienting process in manufacturing [AV04]. Once an order $q_1, \ldots, q_n$ of the states is fixed, we denote $[q_i, q_j] = \{q_\ell \mid i \leq \ell \leq j\}$, and $\min S$, $\max S$ as the minimum and maximum states of $S \subseteq Q$ with respect to the order.

An automaton $A = (Q, \Sigma, \delta)$ is called *weakly acyclic* if there exists an order of its states $q_1, \ldots, q_n$ such that if $\delta(q_i, x) = q_j$ for some $x \in \Sigma$, then $i \leq j$. Note that a monotonic automaton does not have to be weakly acyclic, and vice versa. Both weakly acyclic and monotonic automata present proper subclasses of a widely studied class of aperiodic automata [Vol08]. An automaton is called *strongly connected* if any its state can be mapped to any other state by some word. An automaton is called *orientable*, if there exists a cyclic order of its states that is preserved by all transitions of the automaton

(see [Vol08] for the discussion of this definition). Each monotonic automaton is obviously orientable.

Synchronizing automata model devices that can be reset to some particular state without having any information about their current state. Automata with a synchronizing set of states model devices that can be reset to a particular state with some partial information about the current state, namely when it is known that the current state belongs to a synchronizing subset of states. Checking whether an automaton is synchronizing can be performed in polynomial time [Vol08], but checking whether a given subset of states in an automaton is synchronizing (the Sync Set problem) is a PSPACE-complete problem in binary strongly connected automata [Vor16], and a NP-complete problem in binary weakly acyclic automata [Ryz17].

Eppstein [Epp90] provides a polynomial algorithm for the Sync Set problem, as well as for some other problems, in orientable automata. However, proposed algorithms assume that a cyclic order of the states preserved by transitions is known. Since the problem of recognizing monotonic (respectively, orientable) automata is NP-complete [Szy15], a linear (respectively, cyclic) order preserved by the transitions of an automaton cannot be computed in polynomial time unless P = NP. Thus, we should avoid using this order explicitly in algorithms, so we have to investigate other structural properties of monotonic automata. As shown in this paper, several synchronization problems are still solvable in polynomial time in monotonic automata without knowing an order of states preserved by the transitions.

If an automaton (or a subset of its states) is synchronizing, it is reasonable to find a shortest word synchronizing it. A lot of effort was put into the investigation of this problem, both from extremal and algorithmic point of view. It is known that any synchronizing n-state automaton can be synchronized by a word of length $\frac{n^3-n}{6}$ [Pin83], and the famous Černý conjecture states that the length of such word is at most $(n-1)^2$ [Vol08]. Approximating the length of a shortest word synchronizing a n-state automaton within a factor of $O(n^{1-\epsilon})$ for any $\epsilon > 0$ in polynomial time is impossible unless P = NP [GS15].

For words synchronizing a subset of states, the situation is quite different. It is known that the length of a shortest word synchronizing a subset of states in a binary strongly connected automaton can be exponential in the number of states of the automaton [Vor16]. In weakly acyclic automata, there is a quadratic upper bound on the length of such words, but approximating this length is still a hard problem [Ryz17]. For orientable n-state automata, a tight $n^2 - 2n + 1$ bound on the length of a shortest word synchronizing a subset of states is known [Epp90].

Given an automaton $A = (Q, \Sigma, \delta)$, the *rank* of a word $w \in \Sigma^*$ with respect to a set $S \subseteq Q$ is the size of the image of S under the mapping defined by w in A, i.e., the number $|\{\delta(s, w) \mid s \in S\}|$. The *rank* of an automaton (respectively, of a subset of states) is the minimum among the ranks of all words $w \in \Sigma^*$ with respect to the whole set Q of states of the automaton (respectively, to the subset of states). It follows from the definition that a set of states has rank 1 if and only if it is synchronizing. A state in an automaton is a *sink state* if all letters map this state to itself.

For n-state monotonic automata of rank at most r, Ananichev and Volkov [AV04] show an upper bound of $n-r$ on the length of a shortest word of rank at most r, and also provide bounds on the length of a shortest word of interval rank at most r. Shcherbak [Shc06] continues the investigation of words of bounded interval rank in monotonic automata. Ananichev [Ana05] provides bounds on the length of a shortest word of rank 0 in partial monotonic automata of rank 0.

In this paper, we study both extremal and algorithmic questions of subset synchronization in monotonic automata. In Section 2, we provide structural results about synchronizing

sets of states in monotonic automata and give algorithmic consequences of this results. In Section 3, we provide lower and upper bounds on the maximum length of shortest words synchronizing a subset of states in monotonic automata. In Section 4, we provide NP-hardness and inapproximability of several problems related to subset synchronization in weakly acyclic monotonic automata over a three-letter alphabet. In Section 5 we give necessary and sufficient conditions when a strongly connected digraph can be colored resulting in an automaton with a pre-defined synchronizing set.

2 Structure of Synchronizing Sets

Let A be an automaton, and S be a subset of its states. In general, if any two states in S can be synchronized (i.e., form a synchronizing set), S does not necessarily have to be synchronizing, as it is shown by the following theorem.

Theorem 1. *For any positive integer k_0, there exists a binary weakly acyclic automaton A and a subset S of its states such that $|S| \geq k_0$, each pair of states in S in synchronizing, but the rank of S equals $|S| - 1$.*

Proof. Consider the following automaton $A = (Q, \{0, 1\}, \delta)$. Let $S = \{s_1, \ldots, s_k\}$. Let $k = 2^\ell$ for an integer number ℓ, and let $\mathrm{bin}(i)$ be a word which is equal to the binary representation of i of length ℓ (possibly with zeros at the beginning). We introduce new states t_i, p_i for $1 \leq i \leq k$, a state f, and new intermediate states in Q as following. For each s_i, $1 \leq i \leq k$, consider a construction sending s_i to f for a word $\mathrm{bin}(i)$, and to t_i otherwise (such a construction has $k+1$ state). For each t_i, consider the same construction sending t_i to f for a word $\mathrm{bin}(i)$, and to p_i otherwise. For each i, define both transitions from p_i as self-loops. Define both transitions from f as self-loops.

In this construction, each word applied after a word of length 2ℓ obviously has no effect. Consider a word w of length 2ℓ, $w = w_1 w_2$, where both w_1 and w_2 have length ℓ. If $w_1 = w_2$, then the image of S under the mapping defined by w has size k. Otherwise, w synchronizes two states s_i and s_j with $\mathrm{bin}(i) = w_1$ and $\mathrm{bin}(j) = w_2$ and maps all other states to different states. Thus, the rank of S equals $k - 1$. □

The size of the whole automaton is $O(|S|(\log |S|)^2)$, thus S can be large comparing to the size of the whole set of states in the automaton.

Since the Sync Set problem is PSPACE-complete in strongly connected automata, pairwise synchronization of states in a subset does not imply that this subset is synchronizing for this class of automata unless P = PSPACE. Thus, it is reasonable to ask the following question.

Question 2. *How large can be the rank of a subset of states in a strongly connected automaton such that each pair of states in this subset can be synchronized?*

For the rest of section, fix a monotonic automaton $A = (Q, \Sigma, \delta)$, and let $q_1, \ldots, q_n$ be an order of its states preserved by all transitions.

Theorem 3. *Let $S \subseteq Q$ be a subset of states of A. Then S is synchronizing if and only if any two states in S can be synchronized.*

Proof. Obviously, any subset of a synchronizing set is synchronizing.

In the other direction, if any two states in S can be synchronized, then the minimal state $q_\ell = \min S$ and the maximal state $q_r = \max S$ in S can be synchronized by a word $w \in \Sigma^*$. Let $q = \delta(q_\ell, w) = \delta(q_r, w)$. Then the interval $[q_\ell, q_r] = \{q_\ell, \ldots, q_r\}$ is synchronized by w, because each state of $[q_\ell, q_r]$ is mapped to the interval $[\delta(q_\ell, w), \delta(q_r, w)] = \{q\}$, since A is monotonic. Thus, $S \subseteq [q_\ell, q_r]$ is synchronizing. □

Theorem 4. *The problem of checking whether a given set S is synchronizing can be solved in polynomial time for monotonic automata.*

Proof. By Theorem 3 it is enough to check that each pair of states in S can be synchronized, which can be done by solving the reachability problem in the subautomaton of the power automaton, build on all 2-element and 1-element subsets of Q [Vol08]. □

Theorem 5. *A shortest word synchronizing a given subset S of states can be found in polynomial time for monotonic automata.*

Proof. Consider the following algorithm. For each pair of states, find a shortest word synchronizing this pair. This can be done by solving the shortest path problem in the subautomaton of the power automaton, build on all 2-element and 1-element subsets of Q [Vol08]. Let W be the set of all such words that synchronize S. Output the shortest word in W.

By an argument similar to the proof of Theorem 3, any shortest word synchronizing $\{\min S, \max S\}$ is a shortest word synchronizing S, thus the algorithm is optimal. Since there are $\binom{|S|}{2}$ pairs of states, the algorithm is polynomial. □

Define the *synchronizing graph* $G(A)$ of an automaton $A = (Q, \Sigma, \delta)$ as following. The set of vertices of $G(A)$ is Q. Two vertices are adjacent in G if and only if the two corresponding states can be synchronized.

The following fact follows from the proof of Theorem 3 in [Ryz16] and shows that in general the structure of synchronization graphs can be very complicated.

Theorem 6. *Each graph is an induced subgraph of a synchronization graph of some binary weakly acyclic automaton.*

However, synchronizing graphs of monotonic automata are very special. An *interval graph* is the intersection graph of a set of intervals of a line.

Theorem 7. *Synchronization graphs of monotonic automata are interval graphs.*

Proof. For each state $q \in Q$ in A, consider the set S_q of all such states q' that q and q' can be synchronized. It follows from the proof of Theorem 3 that each maximal (by inclusion) synchronizing set is an interval $[q_i, q_j]$ for some q_i, q_j. Thus, each set S_q form an interval, which is the union of all intervals that are maximal synchronizing sets containing q.

Observe that two states can be synchronized if and only if the corresponding intervals intersect. Thus, we obtain that the synchronization graph of A is the intersection graph of the constructed set of intervals. □

Theorem 7 implies the following.

Theorem 8. *A synchronizing subset of states of maximum size can be found in polynomial time in monotonic automata.*

Proof. By Theorem 3, synchronizing sets in monotonic automata correspond to cliques (complete subgraphs) in their synchronization graph. By Theorem 7, synchronization graphs of monotonic automata are interval. A clique of maximum size can be found in polynomial time in interval graphs [Gol04]. □

The problem of finding a synchronizing subset of states of maximum size in general automata is PSPACE-complete [Ryz16]. Türker and Yenigün [TY15] study a variation of this problem, which is to find a set of states of maximum size that can be mapped by some word to a subset of a given set of states in a given monotonic automaton. They reduce the

N-Queens Puzzle problem [BS09] to this problem to prove its NP-hardness. However, their proof is unclear, since in the presented reduction the input has size $O(\log N)$, and the output size is polynomial in N.

3 Shortest Words Synchronizing a Subset of States

The length of a shortest word synchronizing a n-state monotonic automaton is at most $n-1$ [AV04]. In this section we investigate a more general question of bounding the length of a shortest word synchronizing a subset of states in a n-state synchronizing automaton.

Theorem 9. *Let S be a synchronizing set of states in a monotonic n-state automaton A. Then for $n \geq 3$ the length of a shortest word synchronizing S is at most $\frac{(n-1)^2}{4}$.*

Proof. Let $A = (Q, \Sigma, \delta)$, and $\{q_1, \ldots, q_n\}$ be an order of the states preserved by all transitions of A. Define $q_\ell = \min S$, $q_r = \max S$. We can assume that S can be mapped only to states in $[q_\ell, q_r]$. Indeed, assume without loss of generality that q_i, $i < \ell$, is a state such that S can be mapped to q_i. Note that for the states $q_i \leq q_j \leq q_k \leq q_t$, if q_k can be mapped to q_i, and q_t can be mapped to q_j, then q_t can be mapped to a state $q' \leq q_i$, because A is monotonic. Thus, S can be synchronized by applying only letters that map consecutive images of its states to states with smaller indexes and a shortest word synchronizing S has length at most $n-1$.

Now we can assume that S can be mapped only to states in $[q_\ell, q_r]$. This means that S can be mapped to a subinterval $[q_i, q_j]$ of $[q_\ell, q_r]$, and no state outside $[q_i, q_j]$ is reachable from any state of $[q_i, q_j]$. If both q_ℓ and q_r are mapped to states inside $[q_i, q_j]$, they can be synchronized by applying a word of length at most $j-i$, for example by applying only letters mapping the consecutive images of q_r to states with smaller indexes.

Suppose now that $w = w_1 \ldots w_m$ is a shortest word synchronizing S. Consider the sequence of pairs $(t_i, s_i) = (\delta(q_\ell, w_1 \ldots w_i), \delta(q_r, w_1 \ldots w_i))$, $i = 1, 2, \ldots, m$.

As w is a shortest word synchronizing S, and synchronization of S is equivalent to synchronization of $\{q_\ell, q_r\}$, no pair appears in this sequence twice, and the only pair with equal components is (s_m, t_m). Further, for each k, $1 \leq k \leq m$, $\delta(q_\ell, w_1 \ldots w_k) \leq q_i \leq q_j \leq \delta(q_r, w_1 \ldots w_k)$. Thus, the maximum length of w is reached when $q_i = q_j$ (since after both images are in $[q_i, q_j]$ the remaining length of a synchronizing word is at most $j-i$) and is at most $(i-1)(n-i) \leq \frac{(n-1)^2}{4}$. □

The bound is almost tight for monotonic automata over a three-letter alphabet as shown by the following example.

Theorem 10. *For each $m \geq 1$, there exist a $(2m+3)$-state monotonic automaton A over a three-letter alphabet, which has a subset S of states, such that the length of a shortest word synchronizing S is m^2+1.*

Proof. Consider the following monotonic automaton $A = (Q, \Sigma, \delta)$, $Q = \{q_1, \ldots, q_{2m+3}\}$. Let $\Sigma = \{0, 1, 2\}$. Let states q_1, q_{m+2} and q_{2m+3} be sink states. For every state q_i, $2 \leq i \leq m+1$, we set $\delta(q_i, 0) = q_{i+1}$, $\delta(q_i, 1) = q_i$, $\delta(q_i, 2) = q_1$. For every state q_i, $m+4 \leq i \leq 2m+2$, we set $\delta(q_i, 0) = q_{2m+3}$, $\delta(q_i, 1) = q_{i-1}$, $\delta(q_i, 2) = q_i$. Finally we define $\delta(q_{m+3}, 0) = q_{2m+2}$, $\delta(q_{m+3}, 1) = q_{m+3}$, $\delta(q_{m+3}, 2) = q_{m+2}$. See Figure 1 for an illustration of the construction.

All transitions of A respect the order $q_1, \ldots, q_{2m+3}$, so A is monotonic. Let us show that the shortest word synchronizing the set $S = \{q_2, q_{m+3}\}$ is $w = (01^{m-1})^m 2$. Let S' be a set of states such that $q_i, q_j \in S'$, $2 \leq i \leq m+1$, $m+3 \leq j \leq 2m+2$. The set S' can

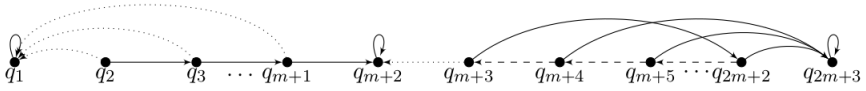

Figure 1: The automaton providing a lower bound for subset synchronization in monotonic automata over a three-letter alphabet. Solid arrows represent transitions for the letter 0, dashed – for the letter 1, dotted – for the letter 2. The states q_1, q_{m+2}, q_{2m+3} are sink states, self-loops are omitted.

be mapped only to q_{m+2}, because A is monotonic. Hence if any state of S' is mapped by a word to q_1 or to q_{2m+3}, then this word cannot synchronize S'.

We start with the set $S = \{q_2, q_{m+3}\}$. There is only one letter 0 that does not map the state q_2 to q_1 or the state q_{m+3} to q_{2m+3}, and maps S not to itself. Indeed, 1 maps q_2 to q_2 and q_{m+3} to q_{m+3}, and 2 maps q_2 to q_1. Thus, any shortest synchronizing word can start only with 0. Consider now the set $\{\delta(q_2, 0), \delta(q_{m+3}, 0)\} = \{q_3, q_{2m+2}\}$. There is only letter 1 that does not map the state q_3 to q_1, or q_{2m+2} to q_{2m+3} and maps this set not to itself. Indeed, 0 maps q_{2m+2} to q_{2m+3} and 2 maps q_3 to q_1. So the second letter of the shortest synchronizing word can only be 1. By a similar reasoning (at each step there is exactly one letter that maps a pair of states not to itself and does not map the states to the sink states q_1 and q_{2m+3}), we deduce that any shortest synchronizing word has to begin with $(01^{m-1})^m$ and it is easy too see that $(01^{m-1})^m 2$ synchronizes S. Thus, w is a shortest word synchronizing S, and its length is $m^2 + 1$. □

For a n-state automaton, the lower bound on the length of a shortest word in this theorem is $\frac{(n-3)^2}{4} + 1$, which is very close to the lower bound $\frac{(n-1)^2}{4}$ from Theorem 9.

By taking q_2 and q_{m+3} as initial states in two equal copies of the automaton in the proof of Theorem 10, and taking q_{m+2} as the only accepting state in both copies, we obtain the following result.

Corollary 11. *A shortest word accepted by two $(2m+3)$-state monotonic automata can have length $m^2 + 1$.*

For binary monotonic automata, our lower bound is slightly smaller, but still quadratic.

Theorem 12. *For each $m \geq 1$, there exist a $(4m+3)$-state binary monotonic automaton A, which has a subset S of states such that the length of a shortest word synchronizing S is at least m^2.*

Proof. Consider the following automaton $A = (Q, \Sigma, \delta)$ with $Q = \{q_1, \ldots, q_{4m+3}\}$, $\Sigma = \{0, 1\}$. Define δ as following. Set q_1, q_{2m+2}, q_{4m+3} to be sink states. Define $\delta(q_i, 1) = q_{i-1}$ for all $i \neq 1, 2m+2, 4m+3$. For each i, $2 \leq i \leq m+1$, define $\delta(q_i, 0) = q_{i+m}$, and for each i, $m+2 \leq i \leq 2m+1$, define $\delta(q_i, 0) = q_{2m+2}$. For each i, $2m+3 \leq i \leq 3m+3$, define $\delta(q_i, 0) = q_{m+i-1}$, and for each i, $3m+4 \leq i \leq 4m+2$, define $\delta(q_i, 0) = q_{4m+3}$. The defined binary automaton is monotonic, since all its transitions respect the order $q_1, \ldots, q_{4m+3}$. See Figure 2 for an example of the construction.

Define $S = \{q_{m+2}, q_{4m+2}\}$. Let us prove that a shortest word synchronizing S has length at least m^2.

The set S can only be mapped to q_{2m+2}, since it is a sink state between $\min S$ and $\max S$. Thus, no synchronizing word maps any state of S to q_1 or q_{4m+3}. Consider now an interval $[q_i, q_j]$ for $2 \leq i \leq m+1$, $2m+3 \leq j \leq 4m+2$ and note that applying 0 reduces its length by 1 (or maps its right end to q_{4m+3}), and applying 1 maps its ends to the ends

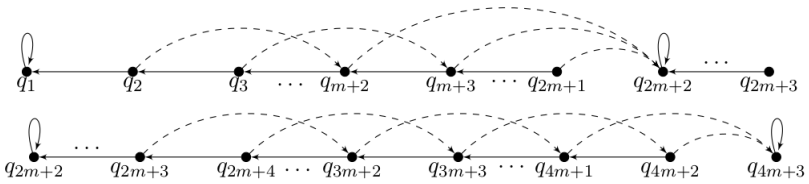

Figure 2: The automaton providing a lower bound for subset synchronization in binary monotonic automata. Dashed arrows represent transitions for the letter 0, solid – for the letter 1. The states q_1, q_{2m+2}, q_{4m+3} are sink states. The picture is divided into two parts because of its width.

of another interval of this form with the same length (or maps its left end to q_1). The maximal length of a segment of this form that allows its left end to be mapped to q_{2m+2} is $2m+1$, so before any end of the interval is mapped to q_{2m+2}, the letter 0 has to be applied at least m times. Each application of 0 moves the right end of the intervals $m-1$ states to the right, so each application of 0 requires $m-1$ applications of 1 so that 0 can be applied one more time. Thus, the word mapping S to q_{2m+2} has length at least m^2. Note that S can be synchronized by a word $w = (1^{m-1}0)^m 1^{2m}$ of length $|w| = m^2 + 2m$. □

For a n-state binary monotonic automaton we get a lower bound of $\frac{(n-3)^2}{16}$ from this theorem.

4 Complexity Results

In this section, we obtain complexity results for several problems related to subset synchronization in monotonic automata. We improve Eppstein's construction [Epp90] to make it suitable for monotonic automata. We shall need the following NP-complete SAT problem [Sip12].

SAT
Input: A set X of n boolean variables and a set C of m clauses;
Output: Yes if there exists an assignment of values to the variables in X such that all clauses in C are satisfied, No otherwise.

Provided a set X of boolean variables $x_1, \ldots, x_n$ and a clause c_j, construct the following automaton $A_j = (Q, \Sigma, \delta)$. Take

$$Q = \{q_1, \ldots, q_{n+1}\} \cup \{q'_2, \ldots, q'_n\} \cup \{s, t\}.$$

Let $\Sigma = \{0, 1, r\}$. Define the transition function δ as following. For each i, $1 \le i \le n$, map a state q_i to q'_{i+1} (or to t if $i = n$) by a letter $x \in \{0, 1\}$ if the assignment $x_i = x$ satisfies c_j, and to q_{i+1} otherwise. For each i, $2 \le i \le n-1$, set $\delta(q'_i, x) = \delta(q'_{i+1}, x)$ for $x \in \{0, 1\}$. Set $\delta(q'_n, x) = t$ for $x \in \{0, 1\}$. Define transitions from t for letters $0, 1$ as self-loops. Finally, define $\delta(q, r) = s$ for $q \in Q \setminus \{t\}$, $\delta(t, r) = t$. See Figure 3 for an example.

Note that A_j is monotonic, since it respects the order

$$s, q_1, q_2, q'_2, q_3, q'_3, \ldots, q_n, q'_n, q_{n+1}, t.$$

It is also weakly acyclic, since its underlying digraph has no simple cycles of length at least 2.

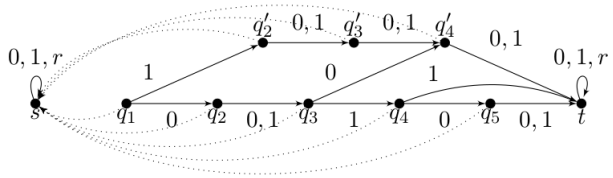

Figure 3: The automaton A_j for a clause $c_j = (x_1 \vee \overline{x}_3 \vee x_4)$. Dotted arrows represent transitions for the letter r.

Also, provided the number of variables n, construct an automaton $T = (Q_T, \Sigma, \delta_T)$ as following. Take $Q_T = \{a, p_1, \ldots, p_{n+1}, b\}$, $\Sigma = \{0, 1, r\}$. Define $\delta(p_i, x) = p_{i+1}$ for each i, $1 \leq i \leq n$, and $x \in \{0, 1\}$, and $\delta(p_{n+1}, x) = b$ for $x \in \{0, 1\}$. Define also $\delta(a, x) = a$ and $\delta(b, x) = b$ for each $x \in \Sigma$, and $\delta(p_i, r) = a$ for $1 \leq i \leq n + 1$. See Figure 4 for an example. This automaton is monotonic, since it respects the order $a, p_1, \ldots, p_{n+1}, b$, and it is obviously weakly acyclic.

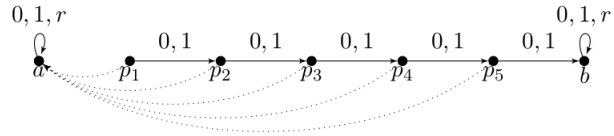

Figure 4: The automaton T for $n = 4$ variables. Dotted arrows represent transitions for the letter r.

First, we prove NP-completeness of the following problem.

FINITE AUTOMATA INTERSECTION
Input: Automata $A_1, \ldots, A_k$ (with input and accepting states);
Output: Yes if there is a word which is accepted by all automata, No otherwise.

This problem is PSPACE-complete for general automata [Koz77], and NP-complete for binary weakly acyclic automata [Ryz17].

Theorem 13. *The* FINITE AUTOMATA INTERSECTION *problem is NP-complete for monotonic weakly acyclic automata over a three-letter alphabet.*

Proof. The fact that the problem is in NP follows from the fact that FINITE AUTOMATA INTERSECTION for weakly acyclic automata is in NP [Ryz17].

To prove hardness, we reduce the SAT problem. For each clause $c_j \in C$, construct an automaton A_j, and set q_1 as its initial state and t as its only accepting state. Construct also the automaton T with initial state p_1 and accepting state a.

We claim that C is satisfiable if and only if all automata $\{A_j \mid c_j \in C\} \cup \{T\}$ accept a common word w. Indeed, assume that there is a common word accepted by all these automata. Then none of the first n letters of this word can be r, otherwise all automata A_j are mapped to s, which is a non-accepting sink state. The next letter has to be r, otherwise T is mapped to b, which is a non-accepting sink state. But that means that in each A_j, the set q_1 is mapped by a n-letter word $z_1 \ldots z_n$ to the accepting state t. Thus, by construction, the assignment $x_i = z_i$ satisfies all clauses in C.

By the same reasoning, if the assignment $x_i = z_i$, $1 \leq i \leq n$, satisfies all clauses in C, then $z_1 \ldots z_n r$ is a word accepted by all automata. □

Now we switch to a related Set Rank problem.

> Set Rank
> *Input*: An automaton A and a set S of its states;
> *Output*: The rank of S.

This problem is hard to approximate for binary weakly acyclic automata [Ryz17]. To get inapproximability results for monotonic automata, we use the following problem.

> Max-3SAT
> *Input*: A set X of n boolean variables and a set C of m 3-term clauses;
> *Output*: The maximum number of clauses that can be simultaneously satisfied by some assignment of values to the variables.

This problem cannot be approximated in polynomial time within a factor of $\frac{8}{7} - \epsilon$ for any $\epsilon > 0$ unless P = NP [Hås01].

Theorem 14. *The* Set Rank *problem cannot be approximated in polynomial time within a factor of $\frac{9}{8} - \epsilon$ for any $\epsilon > 0$ in monotonic weakly acyclic automata over a three-letter alphabet unless $P = NP$.*

Proof. We reduce the Max-3SAT problem. For each clause $c_j \in C$, construct an automaton A_j. Construct also m copies of the automaton T, denoted T_j, $1 \leq j \leq m$. Define an automaton A with the set of states which is the union of all sets of states of $\{A_j, T_j \mid 1 \leq j \leq m\}$, alphabet Σ and transition functions defined in all constructed automata. For each j, identify the state t in A_j with the state a in T_j. Take S to be the set of states q_1 from each automaton A_j. The constructed automaton is monotonic and weakly acyclic.

If h is the minimum number of clauses in C that are not satisfied by an assignment, the set S has rank $m+h$. Indeed, consider an assignment $x_i = z_i$, $1 \leq i \leq n$, not satisfying exactly h clauses in C. Then the word $z_1 \dots z_n r$ has rank $m + h$ with respect to S.

In the other direction, let w be a word of minimum rank with respect to the set S. If any of the first n letters of w is r, then q_1 in each A_i is mapped to s in the corresponding automaton, and thus w has rank $2m$ with respect to S. The same is true if $(n+1)$st letter of w is not r, because then p_1 in each T_i is mapped to b in the corresponding automaton. If first n letters $z_1, \dots, z_n$ of w are not r, and the next letter is r, then the assignment $x_i = z_i$ does not satisfy exactly h' clauses, where $m + h'$ is the rank of the word w with respect to S. For the word of minimum rank, we get the required equality.

It is NP-hard to decide between (i) all clauses in C are satisfiable and (ii) at most $\frac{7}{8}m$ clauses in C can be satisfied by an assignment [Hås01]. In the case (i), the rank of S is m, in the case (ii) it is at least $m + \frac{1}{8}m$. Since it is NP-hard to decide between this two options, we get $(\frac{9}{8} - \epsilon)$-inapproximability for any $\epsilon > 0$. □

By using an argument similar to the proof of Theorem 14, we can show inapproximability of the maximization version of Finite Automata Intersection (where we are asked to find a maximum number of automata accepting a common word). Indeed, take m copies of T together with the set $\{A_j \mid c_j \in C\}$ as the input of Finite Automata Intersection and reduce Max-3SAT to it (input and accepting states are assigned according to the construction in Theorem 13). Then the maximum number of automata accepting a common word is $m + g$, where g is the maximum number of simultaneously satisfied clauses in C, since all copies of T have to accept this word. Thus it is NP-hard to decide between (i) all $2m$ automata accept a common word and (ii) at most $m + \frac{7}{8}m$ automata accept a common word, and we get the following result.

Theorem 15. *The maximization version of the* Finite Automata Intersection *problem cannot be approximated in polynomial time within a factor of $\frac{16}{15} - \epsilon$ for any $\epsilon > 0$ in monotonic weakly acyclic automata over a three-letter alphabet unless $P = NP$.*

Question 16. *What is the complexity of the mentioned problems for binary monotonic automata? Are the problems discussed in this section in NP for monotonic automata?*

We note that it does not matter in the provided reductions whether a linear order preserved by all transitions is known or not.

5 Subset Road Coloring

The famous Road Coloring problem is formulated as following. Given a strongly connected digraph with all vertices of equal out-degree k, is it possible to find a coloring of its arcs with letters of alphabet Σ, $|\Sigma| = k$, resulting in a synchronizing deterministic automaton. This problem was stated in 1977 by Adler, Goodwyn and Weiss [AGW77] and solved in 2007 by Trahtman [Tra09]. A natural generalization of this problem is to find a coloring of a strongly connected digraph turning it into a deterministic automaton where a given subset of states is synchronizing. We introduce the problem formally and show that its solution is a consequence of [BP14]. In particular, the problem of deciding whether such a coloring exists is solvable in polynomial time.

Let $G = (V, E)$ be a strongly connected digraph such that each its vertex has out-degree k. A *coloring* of G with letters from alphabet Σ, $|\Sigma| = k$, is a function assigning each arc of G a letter from Σ, such that for each vertex, each pair of arcs outgoing from it achieves different letters. We say that a coloring *synchronizes* $S \subseteq V$ in G if S is a synchronizing set in the resulting automaton.

If the greatest common divisor of the lengths of all cycles of G is ℓ, the set V can be partitioned into sets $V_1, \ldots, V_\ell$ in such a way that if (v, u) is an arc of G, then $v \in V_i, u \in V_{i+1}$ or $v \in V_\ell, u \in V_1$ [Fri90]. Moreover, such partition is unique.

Theorem 17. *An strongly connected digraph G with vertices of equal out-degree has a coloring synchronizing a set $S \subseteq V$ if and only if $S \subseteq V_i$ for some i.*

Proof. Obviously, if two vertices of S belong to distinct sets V_i and V_j, S can not be synchronized. Assume that $S \subseteq V_i$ for some i. As proved in [BP14], there exists a coloring of G such that the resulting automaton A has rank ℓ. In this coloring each V_j, $1 \leq j \leq \ell$, is a synchronizing set, since no two states from two different sets V_p, V_t, $p \neq t$, can be synchronized and A has rank ℓ. Hence, $S \subseteq V_i$ is also a synchronizing set. □

According to this theorem, checking whether there exists such a coloring can be performed in polynomial time. Construction of this coloring can be done in polynomial time using the algorithm from [BP14].

Acknowledgments

We thank Ilia Fridman for useful comments on the presentation of this paper.

References

[AGW77] Roy L. Adler, L. Wayne Goodwyn, and Benjamin Weiss. Equivalence of topological markov shifts. *Israel J. Math*, 27(1):49–63, 1977.

[Ana05] Dmitry S. Ananichev. The mortality threshold for partially monotonic automata. In Clelia De Felice and Antonio Restivo, editors, *DLT 2005. LNCS, vol. 3572*, pages 112–121. Springer, Berlin, Heidelberg, 2005.

[AV04] Dimitry S. Ananichev and Mikhail V. Volkov. Synchronizing monotonic automata. *Theor. Comput. Sci.*, 327(3):225–239, 2004.

[BP14] Marie-Pierre Béal and Dominique Perrin. A quadratic algorithm for road coloring. *Discrete Appl. Math*, 169:15–29, 2014.

[BS09] Jordan Bell and Brett Stevens. A survey of known results and research areas for N-queens. *Discrete Math.*, 309(1):1 – 31, 2009.

[Epp90] David Eppstein. Reset sequences for monotonic automata. *SIAM J. Comput*, 19(3):500–510, 1990.

[Fri90] Joel Friedman. On the road coloring problem. *Proc. Amer. Math. Soc*, 110:1133–1135, 1990.

[Gol04] Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs (Annals of Discrete Mathematics, Vol 57)*. North-Holland Publishing Co., 2004.

[GS15] Paweł Gawrychowski and Damian Straszak. Strong inapproximability of the shortest reset word. In F. Giuseppe Italiano, Giovanni Pighizzini, and T. Donald Sannella, editors, *MFCS 2015. LNCS, vol. 9234*, pages 243–255. Springer, Heidelberg, 2015.

[Hås01] Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.

[Koz77] Dexter Kozen. Lower bounds for natural proof systems. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 254–266. IEEE Computer Society, 1977.

[Pin83] Jean-Éric Pin. On two combinatorial problems arising from automata theory. *Ann. Discrete Math.*, 17:535–548, 1983.

[Ryz16] Andrew Ryzhikov. Approximating the maximum number of synchronizing states in automata. *CoRR*, abs/1608.00889, 2016.

[Ryz17] Andrew Ryzhikov. Synchronization problems in automata without non-trivial cycles. *CoRR*, abs/1702.08144, 2017.

[Shc06] Tamara Shcherbak. The interval rank of monotonic automata. In Jacques Farré, Igor Litovsky, and Sylvain Schmitz, editors, *CIAA 2005. LNCS, vol. 3845*, pages 273–281. Springer, Berlin, Heidelberg, 2006.

[Sip12] Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, 3rd edition, 2012.

[Szy15] Marek Szykuła. Checking whether an automaton is monotonic is NP-complete. In Frank Drewes, editor, *CIAA 2015. LNCS, vol. 9223*, pages 279–291. Springer, Cham, 2015.

[Tra09] Avraham N. Trahtman. The road coloring problem. *Israel J. Math*, 172(1):51–60, 2009.

[TY15] Uraz Cengiz Türker and Hüsnü Yenigün. Complexities of some problems related to synchronizing, non-synchronizing and monotonic automata. *Int. J. Found. Comput. Sci.*, 26(01):99–121, 2015.

[Vol08] Mikhail V. Volkov. Synchronizing automata and the Černý conjecture. In Carlos Martín-Vide, Friedrich Otto, and Henning Fernau, editors, *LATA 2008. LNCS, vol. 5196*, pages 11–27. Springer, Heidelberg, 2008.

[Vor16] Vojtěch Vorel. Subset synchronization and careful synchronization of binary finite automata. *Int. J. Found. Comput. Sci.*, 27(5):557–578, 2016.

A New Proof for Undecidability of the Bi-Infinite Post Correspondence Problem (Extended Abstract)

Esa Sahla

Department of Mathematics and Statistics, University of Turku, Finland

1 Introduction

In this paper we study a variant of the *Post's Correspondence Problem* (PCP) called the *bi-infinite Post's Correspondence Problem* ($\mathbb{Z}$PCP). It is well known that the PCP can be stated using morphisms over free monoids. Therefore, we give the following formal definition for the $\mathbb{Z}$PCP:

Problem. *Given two morphisms $h, g\colon A^* \to B^*$, determine whether or not there exists a bi-infinite word $\omega : \mathbb{Z} \to A$ such that $h(\omega) = g(\omega)$.*

Equality of the images of bi-infinite words is defined in the following way: $h(\omega) = g(\omega)$ if and only if there is a constant $k \in \mathbb{Z}$ such that $h(\omega)(i) = g(\omega)(i + k)$ for all positions $i \in \mathbb{Z}$.

Undecidability of $\mathbb{Z}$PCP was originally proved by Ruohonen in [4] using undecidability in linearly bounded automata (LBA). Here we give an outline of a new simpler proof of undecidability. Our proof is simpler because 1) it uses the word problem of the special type of semi-Thue systems in the reduction, and, 2) we have a fixed word in the beginning which is reduced to the instance of the PCP.

In the proof of our main result we reduce $\mathbb{Z}$PCP to the *circular word problem* for deterministic semi-Thue systems. The undecidability of the circular word problem was proved in [3] by constructing a reversible semi-Thue system where all steps of the derivation were remembered in the words of the derivation. We will give a more straightforward construction for the circular word problem (for B-deterministic semi-Thue systems) using the halting problem of deterministic Turing machines.

2 Semi-Thue System

In the proof of our main result we are using a special type of semi-Thue systems that are called B-deterministic. The construction of our system $T_{\mathcal{M}}$ is made to replicate the computation of a deterministic Turing machine $\mathcal{M}$ by adding re-writing rules that correspond to the transition function of the machine.

A computation in a semi-Thue system is called cyclic if started from some word w it will return to w after some finite number of derivation steps. From our construction it follows that the system has a *cyclic* computation started from the configuration corresponding to the empty input if and only if the machine halts when started on the blank tape. Thus it is undecidable whether or not $T_{\mathcal{M}}$ has a cyclic computation started from the initial configuration, and we have the following lemma:

Lemma. *The circular word problem is undecidable for deterministic semi-Thue systems.*

3 The Main Result

The proof of the following theorem uses an idea of Claus [1], where the basic reduction form word problem to PCP is given, and the modification of [1] by Halava and Harju [2] for the undecidability of the (one-way) infinite PCP.

Theorem. *The $\mathbb{Z}PCP$ is undecidable.*

Proof. We give here only the idea of the proof:

To prove the undecidability of ZPCP we construct an instance (h, g) of $\mathbb{Z}$PCP and claim that the instance has a solution if and only if $T_{\mathcal{M}}$ has a cyclic computation. We construct our instance (h, g) of $\mathbb{Z}$PCP using the rules of $T_{\mathcal{M}}$. The idea behind the construction is to have morphisms that simulate the derivation (see [1]): if there is a derivation $xuy \rightarrow_{T_{\mathcal{M}}} xvy$ with a rule $t = (u, v)$, then $g(xty) = xuy$ and $h(xty) = xvy$. Furthermore, we will have the images of h and g *desynchronized* with different symbols.

Our morphisms will be $h, g : (\{a_1, a_2, b_1, b_2, \overline{a_1}, \overline{a_2}, \overline{b_1}, \overline{b_2}, \#, \overline{\#}, \#_1, \#_2\} \cup R_{\mathcal{M}})^* \rightarrow \{a, b, d, e, f, \#\}^*$ given by the table

	h	g	
x_1	dxd	xee,	$x \in \{a, b\}$
x_2	ddx	xee,	$x \in \{a, b\}$
t_i	$d^{-1}l_{d^2}(v_i)$	$r_{e^2}(u_i)$,	$t_i \notin \{t_0, t_h\}$
t_0	$d^{-1}l_{d^2}(v_0)$	$r_{e^2}(u_0)e^{-2}f^3$	
t_h	$dr_{e^2}(v_h)e^{-2}ff$	$r_{e^2}(u_h)$	
$\#$	$dd\#d$	$\#ee$	
$\overline{x_1}$	xee	xdd,	$\overline{x} \in \{\overline{a}, \overline{b}\}$
$\overline{x_2}$	exe	xdd,	$\overline{x} \in \{\overline{a}, \overline{b}\}$
$\overline{t_i}$	$e^{-2}l_{e^2}(v_i)e$	$r_{d^2}(u_i)$,	$\overline{t_i} \notin \{\overline{t_0}, \overline{t_h}\}$
$\overline{t_0}$	$e^{-2}l_{e^2}(v_0)e$	$r_{d^2}(u_0)d^{-2}f^3$	
$\overline{t_h}$	$r_{d^2}(v_h)d^{-2}f$	$r_{d^2}(u_h)$	
$\overline{\#}$	$e\#ee$	$\#dd$	
$\#_1$	$f\#ee$	$\#ee$	
$\#_2$	$ff\#d$	$\#dd$	

To eventually obtain a matching situation we introduce overlined copies of the configurations whose images under the morphisms are still desynchronized with different symbols but the symbols are now inverted. This "trick" leaves out the possible trivial solutions which appear in the construction for infinite PCP in [2]. Because of the nature of $\mathbb{Z}$PCP we need not worry about the local matching of the images. By having the same configurations overlined and unoverlined has the effect that these configurations will be desynchronized with the same symbols. The matching will be acquired by shifting either image by a constant k such that the images of overlined words are matched with the images of corresponding words that are not overlined. The size of k depends on the length of the (cyclic) derivation in $T_{\mathcal{M}}$ as well as the lengths of the words in the derivation.

The main point in the proof is that the two different desynchronizing symbols force the solutions to alternate between symbols that are overlined and ones that are not. The morphisms are constructed in a way that transitioning from unoverlined symbols to overlined symbols forces the configuration corresponding to the initial configuration (again corresponding to a Turing machine's empty tape) of our system $T_{\mathcal{M}}$ to appear in the solution.

By the forms of h and g we will then have a solution ω to our instance if there is a cyclic computation in $T_{\mathcal{M}}$. Conversely from the fact that the initial configuration appears

in the solution ω we can deduce that again by the forms of h and g all of the configurations reached from the initial configuration using a finite number of derivation steps must appear in ω and that the computation must be cyclic.

Our result then follows from the fact that it is undecidable whether or not there is a cyclic computation in $T_{\mathcal{M}}$ started on the initial configuration. □

References

[1] V. Claus, Some remarks on PCP(k) and related problems, *Bull. EATCS* **12** (1980), 54 – 61.

[2] V. Halava, T. Harju. *Undecidability of Infinite Post Correspondence Problem for Instances of Size 9*. RAIRO–Theor. Inf. Appl. **40**, 551 – 557, 2006.

[3] V. Halava and T. Harju, New proof for the undecidability of the circular PCP, Acta Inf. **50** (5-6), 331 – 341, 2013.

[4] K. Ruohonen: *On Some Variants of Post's Correspondence Problem*, Acta Informatica **19**, 357 – 367, 1983.

Order-theoretic characteristics and dynamic programming for Precedence Constrained Traveling Salesman Problem

Yaroslav Salii

Krasovskii Insitute of Mathematics and Mechanics UB RAS, Ural Federal University
yvs314@gmail.com

Abstract

This report details the reasons for and our experience of applying the relation- and order-theoretical methods to quantifying the feasibility of solving precedence-constrained traveling salesman problem instances (TSP-PC; also known as SOP, sequential ordering problem) through dynamic programming (DP) or a DP-derived heuristic called "restricted DP." Its primary contribution is (a) a computation of width, density, and transitive reduction for the orders corresponding to the TSP-PC instances from the well-known TSPLIB benchmark instances library and (b) time and space complexity estimates for exact DP and restricted DP solutions of the general TSP-PC. In view of width-derived complexity estimates for TSP-PC, we have determined which open TSPLIB problem would lend itself better to being closed by means of DP: we have closed the long-standing `ry48p.3.sop`.

Introduction

The well-known Traveling Salesman Problem (TSP) consists of visiting several points, typically called "cities," once and once only, in a way that minimizes the total cost, the cost function being defined on pairs of cities and mostly interpreted as the distance between them; for a relatively recent general reference, see [21]. There are several minor variations in its statement, mostly concerning whether it is mandated to return to the starting point ("closed," or rather, "ordinary" TSP) or not (open TSP), or if there is a mandatory exit point in addition to the starting point (Shortest Hamiltonian Chain Problem, [12]); we consider the latter variety, with fixed starting point and end point, and we refer to all these varieties as "TSP" indiscriminately.

We consider TSP on a complete graph, or rather, eschew the graph-theoretic issues altogether. Then, the problem reduces to finding an optimal route—a permutation of the cities' indices. In precedence-constrained TSP (TSP-PC)[1], these permutations are further restricted by a certain partial order P on the cities—only the permutations that express *linear extensions* of P are accepted as feasible solutions, which diminishes the DP state space cardinality, making DP solutions feasible for certain "highly constrained" problems even if they have relatively many cities (compared with the ordinary TSP), and there is a problem of determining whether a given problem instance is "highly constrained" or not.

A hands-on approach would be to run exact DP on a problem and then see whether it will exhaust the available memory; however, the perk of DP is the ability to determine at least some estimate of memory usage a priory. Admittedly, memory usage is also a curse of plain exact DP; however, in present state of the relation between processing power and memory size in computers, it is our experience that if a given problem instance will "fit," it can be solved in reasonable time. This experience is substantiated by the fact that the time complexity of DP is polynomially related to its space complexity, which we will show

[1] Also known as Sequential Ordering Problem (SOP), see the terminological note in [31].

below. We are not aware of any direct studies of time or space complexity for TSP-PC except [19, 31], however, the results for a precedence-constrained scheduling problem from the seminal paper [37] are broadly applicable to the case, the only difference being in the definition of a DP state; we build upon them.

A DP solution's feasibility is effectively dictated by the given problem instance's state space cardinality—all states, along with their values, must be stored to obtain the optimal route, hence the desire to estimate it without explicitly generating all the states, which, since the quantity of states is bounded by the number of ideals of P multiplied by the width $w(P)$, hinges on determining the width and, preferably, obtaining a partition of P into chains, which is known [37] to yield an upper bound on the number of ideals of P.

For all TSP-PC (also known as SOP) instances of the well-known TSPLIB library, we determined the width and the transitive reduction of the corresponding order relation, and—successfully—used it to determine which open problem would lend itself better to being closed by means of DP: we closed the long-standing `ry48p.3.sop`—it took about 125GB of RAM and $1\frac{3}{4}$ hours of computation time to verify that the upper bound 19894 reached by [17] and [28] is in fact the optimal value, attained by the route $(0, 21, 10, 14, 16, 42, 29, 36, 5, 43, 30, 39, 15, 40, 4, 41, 47, 38, 20, 46, 32, 45, 35, 26, 18, 27, 22, 2, 13, 24, 12, 31, 23, 9, 44, 34, 3, 25, 1, 28, 33, 11, 19, 6, 17, 37, 8, 7, 48)$, which may or may not be the only optimal one.

DP is not, in general, a typical method in solving TSP-PC and its generalizations: to the best of the author's knowledge, it has not been attempted since 1992 [5] with exceptions in [3], which dealt with a particular case, TSP-PC on a line, and the works by A.G. Chentsov and his students and co-authors (precedence constrained problems were first considered circa 2004 [8]), including the author. The most recent research in solving benchmark TSP-PC instances was based on a branch-and-bound scheme that used multivalued decision diagrams [13], a custom branch-and-bound scheme [36, 28], and—state of the art for TSPLIB instances—a branch-and-cut scheme [17].

1 TSP-PC Statement

1.1 Preliminaries

We denote *equality by definition* by the symbol $\triangleq$, write *unordered pairs* of objects in curly braces $\{a, b\}$ and *ordered pairs* in parentheses (a, b). Longer ordered tuples are written either in parentheses, e.g., $(a_1, \ldots, a_k)$, or in "sequence notation," $(a_i)_{i=1}^{k}$.

The $\subset$ symbol is understood in the non-strict sense—it does not necessarily denote a *proper* subset. All subsets of a set K are denoted by $\mathcal{P}(K)$; all *nonempty* subsets of K are denoted by $\mathcal{P}'(K)$. Set-valued functions are referred to as "operators."

As usual, $\mathbb{N}$ denotes the set of all natural numbers, and $\overline{p,q} \triangleq \left\{ i \in \mathbb{N} \,\middle|\, (p \leq i) \wedge (i \leq q) \right\}$ for all natural p and q; note that this is still well-defined in the case $q > p$, and the resulting set is *empty*.

The set of all permutations of elements of a set K is denoted by (bi)$[K]$; permutations themselves are generally denoted by lowercase Greek letters and, for a permutation $\alpha \in$ (bi)$[K]$, the element of K that is indexed by i in the permutation is denoted by α_i, while α_k^{-1} denotes the index of the element $k \in K$. In the TSP context, permutations will be referred to as "routes," interpreted as the sequence the cities are visited in between the special *depot* and the *terminal* (defined below).

1.2 "Geometry"

We consider a Hamiltonian Chain-like problem (see [12]), where both the starting point and the terminal point are explicitly specified.

The *starting point* (also *depot* or *base*) is denoted by 0. Let $n \in \mathbb{N}$ denote the number of cities to be visited, then, $\overline{1,n}$ denotes the set of all "proper" cities. The *end point* (also *terminal*) is denoted by $\mathbb{t} \triangleq n+1$; designations $\mathbb{t}$ and $n+1$ are interchangeable and one or another will be used where appropriate. We generally do not distinguish between *cities* and their *indices*.

The *cost* of moving between the cities is expressed through the *cost function* $\mathfrak{c}$, $\mathfrak{c}\colon \overline{0,n} \times \overline{1,n+1} \to \mathbb{R}$. The cost $\mathfrak{c}(a,b)$ measures the expense incurred in traveling from a to b, and the lower the total cost the better.

The *total*, or *aggregated* cost, is defined over a route,

$$\mathfrak{C}[\alpha] \triangleq \mathfrak{c}(0,\alpha_1) \oplus \mathfrak{c}(\alpha_1,\alpha_2) \oplus \ldots \oplus \mathfrak{c}(\alpha_{n-1},\alpha_n) \oplus (\alpha_n,\mathbb{t}); \tag{1}$$

it is the cost as aggregated over the route α by a suitable aggregation operation $\oplus\colon \mathbb{R}\times\mathbb{R} \to \mathbb{R}$.

To prove the forward DP recurrence relation (e.g [22, 37]), we impose on the aggregation operation the properties of being *non-decreasing* in its *first* argument,

$$\forall x,y,z \in \mathbb{R}\ \big(x \geq y\big) \to \big((x \oplus z) \geq (y \oplus z)\big),$$

and being *left-associative*,

$$\forall x,y,z \in \mathbb{R}\ \big(x \oplus y \oplus z\big) = \big((x \oplus y) \oplus z\big).$$

For the backward DP recurrence relation (e.g. [2, 7]) to give valid results, the aggregation operation should be non-decreasing in its second argument and right-associative.

These properties are satisfied by the common arithmetic + aggregation, as in the ordinary TSP, but also by the max or "bottleneck" aggregation characteristic of Bottleneck TSP (see [21, Ch. 15]). For a proof of (backwards) DP recurrence relation for a generalized[2] TSP-PC "with abstract aggregation," refer to [35].

1.3 Precedence Constraints

Precedence constraints are specified by means of a strict *partial order* $P \triangleq \big(\overline{1,n}, <_P\big)$. We will not generally distinguish between the order P and its order relation $<_P$ and use the term "order" for both. If $a <_P b$, then it is assumed that a must be visited *before* b. The *incomparability* relation of P is denoted by $\|$; if $a \,\|\, b$, then, any ordering thereof is feasible.

Along with the order [37], precedence constraints are known to be specified by a *directed acyclic graph* [16] or well-founded binary relation [7]; in case of finite number of cities, all three representations are equivalent since a covering relation determines the ordering [18, p. 6, Lemma 1]; see also the digraph interpretation [1].

The set of feasible routes is defined as follows:

$$\mathbb{A} \triangleq \Big\{\alpha \in (\mathrm{bi})[\overline{1,n}] \;\Big|\; \forall a,b \in \overline{1,n}\ \big(a <_P b\big) \to \big(\alpha^{-1}(a) < \alpha^{-1}(b)\big)\Big\}. \tag{2}$$

And the problem is to find the least-cost feasible route:

$$\mathfrak{C}[\alpha] \xrightarrow[\alpha\in\mathbb{A}]{} \min. \tag{TSP-PC}$$

[2] cities are grouped into *clusters* or *megalopolises*, see [21, Ch. 13]; sometimes this generalization is also called *international* TSP

We will also need the notions of order-theoretic minimal and maximal elements in a set $K \in \mathcal{P}(\overline{1,n})$ in view of the partial order P. The definitions of the operators selecting them follow [34, Def. 2.1.5, 2.3.2],

$$\mathrm{Max}[K] \triangleq \{m \in K \mid \forall k \in K (m \geq_P k) \vee (m \parallel k)\}; \tag{3}$$

$$\mathrm{Min}[K] \triangleq \{m \in K \mid \forall k \in K (m \leq_P k) \vee (m \parallel k)\}. \tag{4}$$

2 Dynamic Programming for TSP-PC

Dynamic programming is essentially about decomposing the complete problem into a set of smaller subproblems, the DP states, to the point of the downright trivial requiring to optimally walk from the base through a single city, which serve as the initial conditions. However, in view of precedence constraints, not every subset of $\overline{1,n}$ can form a feasible "task set."

2.1 Feasible task sets

A set $I \subset \overline{1,n}$ is considered *feasible* if it may be ordered "in accordance" with $<_P$, that is, if it is an *order-theoretic ideal* (down-set) [14, §1.27]; hereinafter we omit the designation "order-theoretic" and call such sets *ideals*[3] or feasible (task) sets. Let us gather all ideals into the family

$$\mathcal{I} \triangleq \left\{ I \in \mathcal{P}\big(\overline{1,n}\big) \;\middle|\; \forall i \in I \; \forall j \in \overline{1,n} \; \big(j <_P i\big) \to \big(j \in I\big) \right\} \tag{5}$$

and stratify it by the ideals' cardinality, $\forall k \in \overline{0,n} \; \mathcal{I}_k \triangleq \{I \in \mathcal{I} \mid |I| = k\}$. Note that the *empty set* is (trivially) an ideal as well.

To define the state space, we first establish the *bottom-up* connection between the ideals of neighboring cardinality by means of a *feasible expansion* operator $\mathbf{E}\colon \mathcal{I} \setminus \overline{1,n} \to \mathcal{P}'(\overline{1,n})$,

$$\mathbf{E}[I] \triangleq \{m \in \overline{1,n} \setminus I \mid I \cup \{m\} \in \mathcal{I}\}; \tag{6}$$

note that the operator $\mathbf{E}$, in fact, selects the *minimal* elements of $\overline{1,n} \setminus I$ (see (4)). Let us prove it in the form of the following *characterization* theorem.

Theorem 1. *For all $I \in \mathcal{I} \setminus \overline{1,n}$ and $m \in \overline{1,n} \setminus I$, the following conditions are equivalent:*

(i) $I \cup \{m\} \in \mathcal{I}$;

(ii) $m \in \mathrm{Min}\Big[\overline{1,n} \setminus I\Big]$.

Proof. Let us start with the implication (i)→(ii). Fix $I \in \mathcal{I} \setminus \overline{1,n}$ and $m \in \overline{1,n} \setminus I$ such that $I \cup \{m\} \in \mathcal{I}$. Suppose $m \notin \mathrm{Min}\Big[\overline{1,n} \setminus I\Big]$. Then, there must be another element that is less than it, $\exists m' \in \overline{1,n} \setminus I \colon (m' \neq m) \wedge (m' \leq_P m)$. Since $m' \notin I \cup \{m\}$ and $m' \leq_P m$, the set $I \cup \{m\}$ is not an ideal, which *contradicts* the assumption that it is, which proves (i)→(ii).

Let us now prove the converse, (ii)→(i). Fix an ideal $I \in \mathcal{I} \setminus \overline{1,n}$ and $m \in \mathrm{Min}\Big[\overline{1,n} \setminus I\Big]$; suppose $I \cup \{m\}$ is not an ideal. Since I is an ideal by definition, there must exist $m' \in \overline{1,n} \setminus I$ such that $(m' \neq m) \wedge (m' \leq_P m)$. However, the second conjunct *contradicts* the assumption that $m \in \mathrm{Min}\Big[\overline{1,n} \setminus I\Big]$; this proves (ii)→(i), and completes the theorem. □

[3]Note that it is equally possible to use the dual "filter" (coideal, up-set, etc.) formulation and the backwards DP procedure of [2, 7].

The *top-down* connection is better expressed through a different operator, $\mathbf{I}\colon \mathcal{I}\setminus\{\varnothing\} \to \mathcal{P}(\overline{1,n})$

$$\mathbf{I}[I] \triangleq \left\{ m \in I \;\middle|\; I \setminus \{m\} \in \mathcal{I} \right\};$$

It has a similar characterization to $\mathbf{E}$,

Theorem 2. *For all $I \in \mathcal{I} \setminus \{\varnothing\}$, the following conditions are equivalent:*

(i) $I \setminus \{m\} \in \mathcal{I}$;

(ii) $m \in \operatorname{Max}[I]$.

Proof. The proof is very similar to that of Theorem 1; we omit it.

Compared with [7], our usage of $\mathbf{I}$ is similar to its definition [7, Eq. 2.2.28]; however, the definition and the characterization are effectively reversed; also, we use the order-theoretic language in contrast with the one based on well-founded relations, and primarily deal with order ideals instead of order filters.

We should also note that the characterization of the feasible expansion operator $\mathbf{E}$ provides a means of generating all the order ideals starting with the trivial empty one. This generation procedure is not unlike that of [24] and is said [23] to have, in the general case, the complexity of $O(n^2)\cdot|\mathcal{I}|$—*quadratic* in the number of ideals—however, it was developed as a dual to the *top-down* ideal (or rather, filter) generation specified in e.g. [9].

2.2 Dynamic programming states and recurrence procedure

A dynamic programming *state* is an *ordered pair* of the form (K, x), where K is the set of the cities already visited (the *task set*) after exiting from the depot 0, while $x \in \overline{1,n+1}\setminus I$ denotes the city the agent is currently in. The set of all (feasible—only the feasible ones are considered below) states is defined in the following way:

$$\begin{aligned}\mathcal{S} \triangleq &\left\{ (\varnothing, x) \in \{\varnothing\} \times \overline{1,n} \;\middle|\; \forall a \in \overline{1,n}\setminus\{x\}\ \big(x <_P a\big) \vee \big(x \parallel a\big) \right\} \cup \\ &\left\{ (I, x) \in \mathcal{I} \times \overline{1,n} \;\middle|\; \big(I \in \mathcal{I}\big) \wedge \big(x \in \mathbf{E}[I]\big) \right\} \cup \left\{ \big(\overline{1,n}, \mathbb{t}\big) \right\};\end{aligned} \tag{7}$$

let us also stratify them by task set cardinality, $\mathcal{S}_k \triangleq \{(K,x) \in \mathcal{S} \mid |K| = k\}$. Each such state set is called a state space *layer*, and the layers with neighboring cardinality of task sets can be said to "inherit" both the top-down and bottom-up connections that the task sets possess. In view of the mentioned connections, let us say that a state $s' = (K', x')$ *covers* a state $s = (K, x)$ if $\big(K \cup \{x\} = K'\big)$.

Note that there are two special layers, $\mathcal{S}_0$, the states in which exist to specify the initial conditions, and $\mathcal{S}_n$, which denotes the "complete" problem (all cities visited).

The states' *value*—the optimal cost—is defined as follows:

$$v(K,x) \triangleq \begin{cases} \mathfrak{c}(0,x), & K = \varnothing; \\ \min\limits_{\alpha\in\mathbb{A}_K} \left\{ \mathfrak{c}(0,\alpha_1) \oplus \ldots \oplus \mathfrak{c}(\alpha_{|K|-1},\alpha_{|K|}) \oplus \mathfrak{c}(\alpha_{|K|},x) \right\}, & \text{otherwise;} \end{cases}$$

here, $\mathbb{A}_K$ denotes the feasible routes over K, which are defined similarly to (2), and can be viewed as *prefixes* of certain routes from $\mathbb{A}$.

The Bellman function (the recurrence relation) describes the cost of a state through the *values* of the states it covers, $\mathrm{BF}(I,x) \triangleq \min_{e\in\mathbf{I}[I]} \left\{ v(I\setminus\{e\}, e) \oplus \mathfrak{c}(e,x) \right\}$; and the *validity of the dynamic programming* is nothing but the fact that a state's *Bellman function* actually **matches** its *value*—except for those in $\mathcal{S}_0$, for which the former makes no sense.

Theorem 3. *For all* $(I, x) \in \mathcal{S} \setminus \mathcal{S}_0$*, we have*

$$v(I,x) = \mathrm{BF}(I,x) = \min_{e \in \mathbf{I}[I]} \big\{ v(I \setminus \{e\}, e) \oplus \mathfrak{c}(e,x) \big\}. \tag{BF}$$

We omit the proof of this theorem in view of space constraints; it is similar to that of [35] except for (a) notation, (b) direction, (c) generality.

An optimal route—the solution—may then be recovered by means of a top-down search: first find the city α_n that achieves the minimum in (BF) for $v(\overline{1,n}, \mathbb{t})$. Then, α_{n-1} will be the city that achieves the minimum in (BF) for $v(\overline{1,n} \setminus \{\alpha_n\})$, and so on.

Theorem 3 guarantees that this search will be successful, and it is bound to take much less time than the whole procedure of generating the ideals and states and calculating the Bellman function.

Note that there may be multiple optimal routes, and it is possible to list them by "branching" on each α_k where more than one city achieves the minimum in (BF). However, unless $\oplus$ is strictly increasing in its first argument, some (but never all) of the optimal routes may be overlooked since they would not satisfy the principle of optimality as expressed in (BF), see [26, Ch. 9].

This procedure can be made a little less time-consuming by "remembering" the best (feasible) state as the state space is traversed in course of calculating the Bellman function (bottom-up) and memoizing the routing decisions (the cities that achieve the minimum in (BF)) for each state; such memoization was proposed in, e.g., [24, 26]. However, in dynamic programming for TSP-like problems, it is mostly *space*, not *time* that is the limiting factor; thus, memoization of routing decisions may actually limit the maximum dimensions of problems solvable on a given computer.

2.3 Restricted DP

The Restricted DP heuristic was first proposed in [25] for Time-Dependent TSP (the travel cost function depends on the number of cities already visited, etc.), then used as a "heuristic framework" for a family of vehicle routing problems (VRP) [20]. It was also applied to Sequence-Dependent Precedence Constrained Bottleneck Generalized TSP [31] and TSP-PC by the author.

It is effectively a generalization of the nearest neighbor algorithm: instead of retaining the whole layer $\mathcal{S}_i$, it only retains $H, H \in \mathbb{N}$, best, and generates the next layer based on those retained. Thus, as H increases, it approaches exact DP. It always produces an *upper bound* on the optimal value of the problem—clearly, if not all (covered) states are taken into account, the value of minimum in (BF) does not decrease.

The main features of this heuristic are the following:

- it always produces a feasible solution
- its time and space complexities are polynomial with respect to H and n
- its easy to implement based on an existing exact dynamic programming solution code base

3 Time and Space Complexity of DP for TSP-PC

3.1 Exact DP

Let us start with space complexity—the number of DP states.

Proposition 4. *$|\mathcal{S}| \leq w|\mathcal{I}|$, where w is the width of P—the cardinality of the maximal (by inclusion) antichain in P.*

Proof. Recall (7) that the states are composed of ideals and "interfaces," the latter depending on the ideal through (6). This operator picks the *minimal elements* of the corresponding set, and the minimal elements of any set clearly form an antichain. □

Note that it is generally hard to enumerate all the order ideals: there is a bijection between the set of all order ideals and all antichains: an antichain A uniquely determines its *principal ideal*

$$\downarrow A = \left\{ i \in \overline{1,n} \,\middle|\, \exists a \in A \colon i <_P a \right\};$$

the inverse direction is covered by order-theoretic maximum, the operator **I**. The problem of enumerating all antichains is known[29] to be $\#P$-hard, hence the need to estimate it (see [37] for estimates). However, in many specific classes of orders, it is much easier; [37] is a very good reference. For example, the precedence constraints that arise in the problem of routing the tool in CNC sheet cutting machines (see e.g. [11]) form a forest (indegree is at most 1), the ideals of which can be enumerated in at most square of the number of vertices (cities) [32].

And now for the time complexity:

Proposition 5. *The time complexity of an exact DP solution for TSP-PC is at most $\mathcal{O}(t_{\mathrm{BF}} w n^2 |\mathcal{S}|)$, where t_{BF} denotes the time it takes to compute $v(I \setminus \{e\}, e) \oplus \mathfrak{c}(e, x)$ in* (BF).

Proof. To solve the problem, each state $s \in \mathcal{S}$ must first be generated. For each ideal it takes at most $\mathcal{O}(n^2)$ to generate both its "successors" (ideals that contain the elements of this ideal and some other city) and the states that correspond to it—it is the complexity of (6) [23][4], a simple set-theoretic method; note that there exist better (e.g. *logarithmic* in the number of ideals) algorithms for generating the ideals, see [6, Appendix A.2.2]; to the best of the author's knowledge, there were no attempts to use them in DP for TSP-like problems.

After the state is generated, (BF) must be calculated for it. Since the image of the operator **I** is an antichain (the set of maximal elements of the ideal), one tests at most w elements to determine the value of the Bellman function, and each such "test" takes t_{BF} by definition.

Having constructed the Bellman function, we must now "recover" the optimal solution through it. The recovery consists of repeatedly testing whether it was indeed $(K, x) \in \mathcal{S}_{|K|}$ that achieved the minimum in $v(K \cup \{x\}, y)$, $(K \cup \{x\}, y) \in \mathcal{S}_{|K|+1}$—in the worst case, the same as recomputing (BF) without generating either the states or the ideals, hence, it does not modify the order of time complexity. □

The time complexity is evidently polynomially dependent on the number of states.

Note that the inclusion of t_{BF} is important for practical considerations—depending on the cost functions and the aggregation operation, it may make a difference. Very involved and laborious (and requiring a floating-point-valued computer implementation) cost functions and aggregation operations may be present e.g. in CNC tool routing problems [11] and in problems of decommissioning a nuclear power generation unit [9]. The time t_{BF} also depends on the kind of data structure used to store the values of the Bellman function.

[4] see "The Lawler Approach" in [23]

3.2 Restricted DP

Proposition 6. *The number of states in Restricted DP heuristic solution is less than or equal to* $B(n+1)$.

Proof. Evident in view of definition: there are $(n+1)$ state space layers, each has at most B states. □

Proposition 7. *The time complexity of Restricted DP heuristic is* $\mathcal{O}(t_{\mathrm{BF}} w n^3 B)$.

Proof. Similar to the exact case. □

3.3 Estimating the number of ideals

The papers by G. Steiner [37, 38] appear to still be the best reference on the matter, and the best general and "analytical" (in the same papers, a statistical study was also made) estimates are as follows:

The lower bound on the number of ideals is $2^w + n - w$ [38, p. 286].

The upper bound, obtained by applying the arithmetic mean–geometric mean inequality to the number of ideals in the parallel composition of a chain partition of P, where the latter, by Dilworth's theorem [15], has to contain at least w chains, is $\left(\frac{n+w}{w}\right)^w$ [38, p. 286].

Naturally, one would like to tighten the given bounds. The mentioned upper bound is clearly made less tight by the use of the inequality. By abstaining from the use of the inequality, we come to the bound based on a chain partition of the given order. Since for most orders the chain partition is not unique, there arises a natural question of *optimal chain partition*:

Problem 8. *Given a partial order* $P = (\overline{1,n}, <_P)$*, find its chain partition* $C^* = \bigsqcup_{i=1}^{k} C_i^*$ *that minimizes* $\prod_{i=1}^{k} |C_i^* + 1|$*, where* $k \leq n$ *and* $\bigsqcup$ *denotes the disjoint set union.*

3.4 Determining order-theoretic characteristics

The transitive reduction was determined by relation-theoretic means [33] through a composition of the initial relation with its transitive closure[4, p. 121]; the latter was obtained through the Roy–Warshall algorithm [33, § 3.2.8] in its functional programming form, adapted from [4]. Note that in TSPLIB the relations describing precedence constraints are required to be transitively closed, however, it is not always the case—some authors prefer the "assembly language" of well-founded relations, e.g., [9].

The width was obtained through maximal matching in an auxiliary bipartite poset (see [37, p. 107] for a brief description or [6, § 4.5] for a complete one, including proof), which was computed by means of the Haskell `Data.Graph.MaxBipartiteMatching` library.

One also has to mention the partial order *density*. It is intended as a measure of the order's "restrictivity," and is defined as follows (see e.g. [38]):

$$d(P) = \frac{|\leq_P|}{n(n-1)/2},$$

that is, it is the cardinality of the given order relation divided by the cardinality of the corresponding linear order. It is still used in the papers dealing with TSP-PC, for example, in [27, 28], although in [37, 38] it was proved that there exist high-density (e.g. 0.75) classes of partial orders with an exponential in n number of ideals and low-density (less than

0.25) classes with only a polynomial in n number of ideals. Thus, if a simple normalized "restrictivity" index is required, it is arguably better to take the relation between the number of cities and the width, $\frac{w}{n}$.

Table 3.4 presents the results. The still unsolved instance names are written in **boldface**. The recently closed `ry48p.3.sop` is *italicized*. The columns are as follows:

- n is the number of "proper" cities (not counting the base and the terminal)
- "RED." is the cardinality of the partial order's transitive reduction
- "CL." is the cardinality of the partial order relation ("transitve closure")
- "LOG_S-LB" is the base 2 logarithm of the "lower"[5] bound on the number of states, $w(2^w - n + w)$
- "LOG_S-UB" is the base 2 logarithm of the upper bound on the number of states, $w\binom{n+w}{w}^w$.

The "LOG_S-LB" and "LOG_S-UB" columns offer a practical means of estimating the feasibility of solving a given problem through exact DP: recall that 1 Gigabyte is 2^{30} bytes. If the "lower" bound surpasses 2^{40} ("1 Terabyte"), it is clearly impractical to attempt the solution while if the upper bound is about the same, it should be reasonable to try.

Everything was encoded in Haskell. Sets were rendered as sorted lists of the built-in `Int` type. Computation time was not rigorously measured; for most instances, computation time could be characterized as "instant"; bigger instances took something on the order of few minutes. The base 2 logarithms for the upper and "lower" bounds on the number of states were calculated in MS Excel, then rounded up to one digit.

4 Concluding remarks

We were able to compute the transitive reduction, width, and width-based upper and "lower" bounds on the number of DP states for the well-known TSP-PC benchmark instances (TSPLIB SOP [30]), which lead to a successful identification of the unsolved instance that should lend itself to a solution by exact DP—`ry48p.3.sop`, and solve it we did; the known upper bound 19894 was proved to be precise. The computed parameter values are presented in Table. 3.4. Haskell proved to be adequate means of expression for this sort of algorithms.

4.1 Future work directions

There are four main directions. The *first* one is to try and use the height or other order-theoretic statistics to get better and tighter bounds. One could also try to find the optimal method of chain partitioning, or, at least, see how does any chain partition compare with the pure width-based estimate. Finally, harnessing the best ideal generators (see the note on "Enumeration of $\mathcal{D}(P)$" in [6, Appendix A.2]), it may be possible to obtain the ideal and state quantities through sheer brute force.

The *second* is to increase the domain of research by processing other notable TSP-PC instance libraries (e.g. SOPLIB [27]).

[5] it is not, strictly speaking, a lower bound on the number of *states* since there may be less than w states with the same ideal; however, it is was deemed more informative of the true lower bound than the one that would assume that there is only one state per each ideal

Table 1: Analysis of TSP-PC instances from TSPLIB

NAME	n	RED.	CL.	DENSITY	WIDTH	LOG_S-LB	LOG_S-UB
br17.10.sop	16	10	15	0.12	10	13.4	17.2
br17.12.sop	16	12	22	0.18	9	12.2	16.5
ESC07.sop	7	6	7	0.33	5	7.5	8.7
ESC11.sop	11	3	5	0.09	9	12.2	13.6
ESC12.sop	12	7	11	0.17	10	13.4	14.7
ESC25.sop	25	9	11	0.04	19	23.3	27.3
ESC47.sop	47	10	32	0.03	41	46.4	50.6
ESC63.sop	63	95	233	0.12	53	58.8	65.7
ESC78.sop	78	77	283	0.09	32	37.1	62.1
ft53.1.sop	52	12	12	0.01	42	47.4	54.3
ft53.2.sop	52	25	30	0.02	34	39.1	50.7
ft53.3.sop	52	48	217	0.16	24	28.6	44.5
ft53.4.sop	52	63	759	0.57	13	16.8	33.9
ft70.1.sop	69	17	17	0.01	55	60.8	70.3
ft70.2.sop	69	35	48	0.02	44	49.5	65.4
ft70.3.sop	69	68	215	0.09	35	40.2	60.2
ft70.4.sop	69	86	1325	0.56	16	20.1	42.6
kro124p.1.sop	99	25	33	0.01	78	84.3	98.5
kro124p.2.sop	99	49	68	0.01	65	71.1	92.9
kro124p.3.sop	99	97	266	0.05	43	48.5	79.6
kro124p.4.sop	99	131	2305	0.48	22	26.5	58.6
p43.1.sop	42	9	11	0.01	36	41.2	45.4
p43.2.sop	42	20	34	0.04	26	30.8	40.8
p43.3.sop	42	37	96	0.11	21	25.4	37.7
p43.4.sop	42	50	496	0.58	13	16.8	30.8
prob.100.sop	98	41	41	0.01	57	62.9	88.1
prob42.sop	40	10	19	0.02	34	39.1	43.3
rbg048a.sop	48	192	447	0.40	32	37.1	47.4
rbg050c.sop	50	256	508	0.41	31	36	48
rbg109a.sop	109	622	5329	0.91	12	15.7	43.6
rbg150a.sop	150	952	10334	0.92	13	16.8	51.2
rbg174b.sop	174	1113	13955	0.93	22	26.5	73.9
rbg253a.sop	253	1721	30181	0.95	22	26.5	84.7
rbg323a.sop	323	2412	48202	0.93	47	52.6	145.5
rbg341a.sop	341	2542	54303	0.94	33	38.1	120.7
rbg358a.sop	358	3239	56536	0.88	55	60.8	165.8
rbg378a.sop	378	3069	63585	0.89	55	60.8	169.6
ry48p.1.sop	47	11	12	0.01	37	42.3	49
ry48p.2.sop	47	23	26	0.02	29	33.9	45.2
ry48p.3.sop	47	42	132	0.12	19	23.3	38.4
ry48p.4.sop	47	58	596	0.55	12	15.6	31.2

The *third* is to attempt to estimate the number of ideals of fixed cardinality; to the best of the author's knowledge, for general posets, no nontrivial estimates are known, there is only Wild's enumeration algorithm [39]. The latter estimates will allow to account for swapping from RAM to hard disks, which could allow to resolve slightly more complicated problems—the limiting factor will be not the whole number of states but only the number of states of two most populous neighboring state space layers. Another possible improvement from this knowledge might be a redistribution of "degrees of freedom" in restricted DP: for $H > w^2$, some "allocated states" are wasted on e.g. layer $\mathcal{S}_1$, which could never have more than w^2 states; the remainder of its quota could be better spent on more populous layers. This leads to two possible approaches: (a) append the unused quota to that of the next layer (b) redistribute it a priori, giving more to the more populous layers.

The *fourth* is to quantify the maximum possible performance gain from the distributed parallel computations scheme for dynamic programming [10]. Although this scheme is formulated in the optimal control theory terms, it effectively boils down to an independent computation of (BF) for the states of some layer $\mathcal{S}_k$, $k < n$ (in [10], $k = n - 1$), the results of which are used by the master node to complete the computation all the way to $\mathcal{S}_n$. Evidently, the greater k, the more benefit of dynamic programming versus exhaustive search is lost (for some states, (BF) is recomputed many times), however, small k will not provide the desired distribution of effort. The degree of "intersection" between the states covered by members of $\mathcal{S}_k$ is clearly dependent on the partial order that defines the precedence constraints.

References

[1] Alfred V. Aho, Michael R Garey, and Jeffrey D. Ullman. The transitive reduction of a directed graph. *SIAM Journal on Computing*, 1(2):131–137, 1972.

[2] Richard Bellman. Dynamic programming treatment of the travelling salesman problem. *Journal of the ACM (JACM)*, 9(1):61–63, 1962.

[3] Thierry Benoist, Antoine Jeanjean, and Vincent Jost. Call-based dynamic programming for the precedence constrained line traveling salesman. In *International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems*, pages 1–14. Springer, 2014.

[4] Rudolf Berghammer. A functional, successor list based version of Warshall's algorithm with applications. In *International Conference on Relational and Algebraic Methods in Computer Science*, pages 109–124. Springer, 2011.

[5] Lucio Bianco, Aristide Mingozzi, Salvatore Ricciardelli, and Massimo Spadoni. The traveling salesman problem with precedence constraints. In *Papers of the 19th Annual Meeting/Vorträge der 19. Jahrestagung*, pages 299–306. Springer, 1992.

[6] Nathalie Caspard, Bruno Leclerc, and Bernard Monjardet. *Finite ordered sets: concepts, results and uses.* Number 144 in Encyclopedia of Mathematics and Its Applications. Cambridge University Press, 2012.

[7] Aleksandr G Chentsov. *Extremal problems of routing and scheduling: a theoretical aproach [in Russian].* Izhevsk:Regular and Chaotic Dynamics, 2008.

[8] Aleksandr G Chentsov and Pavel A Chentsov. A precedence constrained routing problem (precedence-constrained TSP): dynamic programming [in Russian]. *Bull. USTU*, (15):148–151, 2004.

[9] Aleksei A Chentsov, Aleksandr G Chentsov, and Pavel A Chentsov. Elements of dynamic programming in extremal routing problems. *Automation and Remote Control*, 75(3):537–550, 2014.

[10] Alexander G Chentsov and Alexey M Grigoryev. A scheme of independent calculations in a precedence constrained routing problem. In *International Conference on Discrete Optimization and Operations Research*, pages 121–135. Springer, 2016.

[11] Pavel A Chentsov and Alexander A Petunin. Tool routing problem for CNC plate cutting machines. *IFAC-PapersOnLine*, 49(12):645–650, 2016.

[12] Nicos Christofides. The shortest hamiltonian chain of a graph. *SIAM Journal on Applied Mathematics*, 19(4):689–696, 1970.

[13] Andre A Cire and Willem-Jan van Hoeve. Multivalued decision diagrams for sequencing problems. *Operations Research*, 61(6):1411–1428, 2013.

[14] Brian A Davey and Hilary A Priestley. *Introduction to lattices and order*. Cambridge University Press, 2002.

[15] Robert P Dilworth. A decomposition theorem for partially ordered sets. *Annals of Mathematics*, pages 161–166, 1950.

[16] Laureano F Escudero. An inexact algorithm for the sequential ordering problem. *European Journal of Operational Research*, 37(2):236–249, 1988.

[17] Luis Gouveia and Mario Ruthmair. Load-dependent and precedence-based models for pickup and delivery problems. *Computers & Operations Research*, 63:56–71, 2015.

[18] George Grätzer. *General Lattice Theory*, volume 52. Birkhäuser, 2012.

[19] Alexey M Grigoriev, Evgeny E Ivanko, and Alexander G Chentsov. Dynamic programming in a generalized courier problem with inner tasks: elements of a parallel structure. *Model. Anal. Inform. Sist.*, 18(3):101–124, 2011.

[20] Joaquim Gromicho, Jelke J van Hoorn, AL Kok, and JMJ Schutten. Restricted dynamic programming: a flexible framework for solving realistic VRPs. *Computers & operations research*, 39(5):902–909, 2012.

[21] G. Gutin and Abraham P. Punnen, editors. *The traveling salesman problem and its variations*, volume 12 of *Combinatorial optimization*. Springer Science & Business Media, 2002.

[22] Michael Held and Richard M Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial & Applied Mathematics*, 10(1):196–210, 1962.

[23] Edward PC Kao and Maurice Queyranne. On dynamic programming methods for assembly line balancing. *Operations Research*, 30(2):375–390, 1982.

[24] Eugene L Lawler. Efficient implementation of dynamic programming algorithms for sequencing problems. *Stichting mathematisch centrum preprint*, 1979.

[25] Chryssi Malandraki and Robert B Dial. A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem. *European Journal of Operational Research*, 90(1):45–55, 1996.

[26] Michel Minoux. *Programmation mathèmatique. Théorie et algorithmes*. Lavoisier, 2^e edition, 2008.

[27] Roberto Montemanni, Derek H Smith, and Luca Maria Gambardella. Ant colony systems for large sequential ordering problems. In *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE*, pages 60–67. IEEE, 2007.

[28] Vassilis Papapanagiotou, J Jamal, Roberto Montemanni, Ghassan Shobaki, and Luca Maria Gambardella. A comparison of two exact algorithms for the sequential ordering problem. In *Systems, Process and Control (ICSPC), 2015 IEEE Conference on*, pages 73–78. IEEE, 2015.

[29] J Scott Provan and Michael O Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal on Computing*, 12(4):777–788, 1983.

[30] G Reinelt. {TSPLIB}: a library of sample instances for the TSP (and related problems) from various sources and of various types. URL: http://comopt. ifi. uniheidelberg. de/software/TSPLIB95. Accessed: 2017-03-24.

[31] Yaroslav Salii. Restricted dynamic programming heuristic for Precedence Constrained Bottleneck Generalized TSP. In Andrey Sozykin, Elena Akimova, and Dmitry Ustalov, editors, *Proceedings of the 1st Ural Workshop on Parallel, Distributed, and Cloud Computing for Young Scientists*, volume 1513, pages 85–108. CEUR Workshop Proceedings, 2015.

[32] Yaroslav V Salii. On the effect of precedence constraints on computational complexity of dynamic programming method for routing problems [in Russian]. *Vestnik Udmurtskogo Universiteta. Matematika. Mekhanika. Komp'yuternye Nauki*, (1):76–86, 2014.

[33] Gunther Schmidt and Thomas Ströhlein. *Relations and graphs: discrete mathematics for computer scientists*. EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1993.

[34] Bernd S W Schröder. *Ordered sets*. Springer, 2003.

[35] Aleksandr N. Sesekin, Aleksandr G. Chentsov, and Aleksei A. Chentsov. Routing with an abstract function of travel cost aggregation [in Russian]. *Trudy Inst. Mat. i Mekh. UrO RAN*, 16(3):240–264, 2010.

[36] Ghassan Shobaki and Jafar Jamal. An exact algorithm for the sequential ordering problem and its application to switching energy minimization in compilers. *Computational Optimization and Applications*, 61(2):343–372, 2015.

[37] George Steiner. On the complexity of dynamic programming for sequencing problems with precedence constraints. *Annals of Operations Research*, 26(1):103–123, 1990.

[38] George Steiner. On estimating the number of order ideals in partial orders, with some applications. *Journal of statistical planning and inference*, 34(2):281–290, 1993.

[39] Marcel Wild. Output-polynomial enumeration of all fixed-cardinality ideals of a poset, respectively all fixed-cardinality subtrees of a tree. *Order*, 31(1):121–135, 2014.

Regularity of the maximal distance minimizers

Yana Teplitskaya

Abstract

We study the properties of sets Σ which are the solutions of the maximal distance minimizer problem, id est of sets having the minimal length (one-dimensional Hausdorff measure) over the class of closed connected sets $\Sigma \subset \mathbb{R}^2$ satisfying the inequality $\max_{y\in M} \operatorname{dist}(y,\Sigma) \le r$ for a given compact set $M \subset \mathbb{R}^2$ and some given $r > 0$. Such sets play the role of the shortest possible pipelines arriving at a distance at most r to every point of M, where M is the set of customers of the pipeline.

In this work it is proved that each maximal distance minimizer is a union of finite number of curves, having one-sided tangent lines at each point. Moreover the angle between these lines at each point of a maximal distance minimizer is greater or equal to $2\pi/3$. It shows that a maximal distance minimizer is isotopic to a finite Steiner tree even for a "bad" compact M, which differs it from a solution of the Steiner problem (an example of a Steiner tree with an infinite number of branching points can be find in [9]). Also we classify the behavior of a minimizer in a neighbourhood of any point of Σ.

In fact, all the results are proved for more general class of local minimizer, i. e. sets which are optimal in any neighbourhood of its arbitrary point.

For a given compact set $M \subset \mathbb{R}^2$ consider the functional

$$F_M(\Sigma) := \sup_{y\in M} \operatorname{dist}(y,\Sigma),$$

where Σ is a subset of $\mathbb{R}^2$ and $\operatorname{dist}(y,\Sigma)$ stands for the Euclidean distance between y and Σ (naturally, $F_M(\emptyset) := +\infty$). The quantity $F_M(\Sigma)$ will be called the *energy* of Σ. Consider the class of closed connected sets $\Sigma \subset \mathbb{R}^2$ satisfying $F_M(\Sigma) \le r$ for some $r > 0$. We are interested in the properties of sets of the minimal length (one-dimensional Hausdorff measure) $\mathcal{H}^1(\Sigma)$ over the mentioned class. Such sets will be further called *minimizers*. They can be viewed as the shortest possible pipelines arriving at a distance at most r to every point of M which in this case is considered as the set of customers of the pipeline.

It is proven (in fact, even in the general n-dimensional case $M \subset \mathbb{R}^n$; see [6] for the rigorous statement and details) that the set $OPT^*_\infty(M)$ of minimizers (for all $r > 0$) is nonempty and coincides with the set $OPT_\infty(M)$ of solutions of the dual problem: minimize F_M over all closed connected sets $\Sigma \subset \mathbb{R}^2$ with prescribed bound on the total length $\mathcal{H}^1(\Sigma) \le l$, thus the solutions are called minimizers of maximal distance. It is known also that for each minimizer of positive length its energy is equal to r. If one minimizes maximum or average distance functional over discrete sets with an a priori restriction on the number of connected components (rather than over connected one-dimensional sets) one gets another class of closely related problems known as k-center problem and k-median problem (see e.g. [4, 10, 11] as well as [1, 2] and references therein).

Note that Σ is compact because $\Sigma \subset \overline{B_r(\operatorname{conv} M)}$, where $\operatorname{conv} M$ is the convex hull of M. Some basic properties of minimizers for the problem mentioned above in n-dimensional case (like the absence of loops and Ahlfors regularity) have been proven in [7]. Further, in [6] the following characterization of minimizers has been studied. Let $B_\rho(x)$ be the open ball of radius ρ centered at a point x, and let $B_\rho(M)$ be the open ρ-neighborhood of M i.e.

$$B_\rho(M) := \bigcup_{x\in M} B_\rho(x).$$

Further, we introduce

Definition 1. A point $x \in \Sigma$ is called *energetic*, if for all $\rho > 0$ one has

$$F_M(\Sigma \setminus B_\rho(x)) > F_M(\Sigma).$$

Denote the set of all energetic points of Σ by G_Σ.

Let us consider a minimizer Σ with energy $r = F_M(\Sigma)$ (the subset of $OPT^*_\infty(M)$ of minimizers with energy r will be further denoted by $OPT^*_\infty(M, r)$). Then the set Σ can be split into three disjoint subsets:

$$\Sigma = E_\Sigma \sqcup \mathrm{X}_\Sigma \sqcup \mathrm{S}_\Sigma,$$

where $X_\Sigma \subset G_\Sigma$ is the set of isolated energetic points (i.e. every $x \in X_\Sigma$ is energetic and there is a $\rho > 0$ possibly depending on x such that $B_\rho(x) \cap G_\Sigma = \{x\}$), $E_\Sigma := G_\Sigma \setminus X_\Sigma$ is the set of non isolated energetic points and $S_\Sigma := \Sigma \setminus G_\Sigma$ is the set of non energetic points also called Steiner part. In [6] the following assertions have been proven:

(a) For every point $x \in G_\Sigma$ there exists a point $Q_x \in M$ (may be not unique) such that $\operatorname{dist}(x, Q_x) = r$ and $B_r(Q_x) \cap \Sigma = \emptyset$. If X_Σ is not finite, the limit points of X_Σ belong to E_Σ.

(b) For all $x \in S_\Sigma$ there exists an $\varepsilon > 0$ such that $S_\Sigma \cap B_\varepsilon(x)$ is either a segment or a regular tripod, i.e. the union of three segments with an endpoint in x and relative angles of $2\pi/3$. If a point $x \in S_\Sigma$ is a center of a regular tripod, then it called a *Steiner point* (or a *branching point*) of Σ.

Also, as Σ is connected closed set of finite length, at almost every point of Σ there exists a tangent line.

The main result is the following.

Theorem 2. *Let Σ be a maximal distance minimizer for a compact set $M \subset \mathbb{R}^2$. Then*

- *Σ is a union of a finite number of curves;*
- *the angle between each tangent rays at every point of Σ is greater or equal to $2\pi/3$.*

The Theorem implies that set with countable many branching points can not be a maximal distance minimizer. Note that it is not true for the Steiner problem; one can find the example in the work [9].

1 Notation

Definition 3. We say that an order of a point x in the set Σ is equal to n (and write $\operatorname{ord}_x \Sigma = n$), if there exists such $\varepsilon_0 = \varepsilon_0(x) > 0$, that for every open set U containing x and with $\operatorname{diam} U < \varepsilon_0$, the capacity of the intersection $\#(\partial U \cap \Sigma)$ is greater or equal to n, and for each $\varepsilon < \varepsilon_0$ there exists such open set $V \ni x$ that $\operatorname{diam} V < \varepsilon$ and $\#(\partial V \cap \Sigma) = n$.

Note that connectivity of Σ in nontrivial case implies a positive order $\operatorname{ord}_x \Sigma$ for each point $x \in \Sigma$.

Definition 4. We will call by an arc $\breve{]ax]} \subset \Sigma$ a continuous injective image of a segment $]0, 1]$.

Definition 5. Let $x \in \Sigma$. We say that a ray $(ax]$ is a tangent ray of the set Σ at the point (vertex) x, if there exists a consequence of points $x_k \in \Sigma$ such that $x_k \to x$ and $\angle x_k x a \to 0$.

Definition 6. Let $x \in \Sigma$. We say that a ray $(ax]$ is an *energetic edge* of a set Σ at the point (vertex) x, if there exists a consequence of energetic points $x_k \in G_\Sigma$ such that $x_k \to x$ and $\angle x_k x a \to 0$.

Definition 7. We say that a line (ax) is a tangent line to a set Σ at the point x, if for each consequence of points $x_k \in \Sigma$ such that $x_k \to x$ holds $\angle((x_k x)(ax)) \to 0$.

Definition 8. We say that a ray $(ax]$ is one-sided tangent line to a set Σ at the point x, if for each consequence of points $x_k \in \Sigma$ such that $x_k \to x$ holds $\angle x_k x a \to 0$.

1.1 Steiner problem

The Steiner problem which has several different but more or less equivalent formulations, is that of finding a set S with minimal length (one-dimensional Hausdorff measure $\mathcal{H}^1(S)$) such that $S \cup A$ is connected, where A is a given compact subset of a given complete metric space X.

Namely, defined

$$\mathrm{Ntw}\,(A) := \{S \subset X : S \cup A \text{ is connected}\}$$

one has to find an element of $\mathrm{Ntw}\,(A)$ with minimal $\mathcal{H}^1$-length.

If S is a solution (in the case when X is proper and connected a solution exists [8]) to the Steiner problem for a given set A, then $\bar{S}$ is called a Steiner tree for the set A (or a Steiner tree connecting the set A, or just a Steiner set). It has been proven in [8] that in the case $\mathcal{H}^1(S) < +\infty$ the following properties hold:

1. $\overline{S}$ contains no loops (homeomorphic images of S^1);
2. $S \setminus A$ has at most countably many connected components, and each of the latter has strictly positive length;
3. the closure of every connected component of $S \setminus A$ is a topological tree with endpoints on A and with at most one endpoint belonging to each connected component of A.

From now on we will consider the Steiner problem in the case when the ambient space X is the Euclidean plane $\mathbb{R}^2$. Then

4. $\overline{S} \setminus A$ consists of line segments (this follows from the result of [8] stating that away from the data Steiner tree is an embedded graph consisting of geodesic segments);
5. the angle between two segments adjacent to the same vertex is greater or equal to $2\pi/3$ [5].
6. Let us call *a Steiner (or branching) point* such a point of $\bar{S}$ that does not belong to A and which is not an interior point of a segment of $\overline{S}$.

   The degree (in the graph theoretic sense) of a Steiner point x is equal to 3. In this case the angle between any pair of segments adjacent to x is equal to $2\pi/3$ (see [5]). Such figure is called *regular tripod*.
7. It is well-known that for $A = \{x_1, x_2, x_3\} \subset \mathbb{R}^2$ there is unique solution of the Steiner problem. We denote it by $\mathcal{S}t(x_1, x_2, x_3)$.

We will say that a set $S \in \mathrm{Ntw}\,(A)$ is a *locally minimal network* for the given set A if for an arbitrary point $x \in S \setminus A$ there exists a neighbourhood $U \ni x$ such that $\bar{S} \cap \overline{U}$ is a Steiner tree for $\bar{S} \cap \partial U$. If a neighbourhood of a point $x \in \bar{S}$ is a regular tripod then it is called *Steiner point*.

A locally minimal network satisfies all the properties of a Steiner tree mentioned above except the first one (see [8, 5]).

2 Classification of local behavior of Σ. Examples

In this section we will show some examples of maximal distance minimizers.

We have proved the

Theorem 9 (Classification theorem). *1. For $x \in S_\Sigma$, there exists such $\varepsilon > 0$ that $\Sigma \cap B_\varepsilon(x)$ coincides with one of two sets:*

- *(a) a segment (in this case an order of the point x is equal to 2);*
- *(b) a regular tripod (in this case an orger of the point x is equal to 3).*

2. For $x \in X_\Sigma$, there exists such $\varepsilon > 0$ that $\Sigma \cap B_\varepsilon(x)$ coincides with one of two sets:

- *(a) a segment with an end at x (in this case an order of the point x is equal to 1);*
- *(b) a union of two segments with common end at x and an angle greater or equal to $2\pi/3$ between them (in this case an order of the point x is equal to 2).*

3. For $x \in E_\Sigma$ there exists such $\varepsilon > 0$ that $\Sigma \cap B_\varepsilon(x)$ coincides with one of two sets:

- *(a) an arc with an end at x having in this point a one-sided tangent line (in this case an order of the point x is equal to 1). Herewith x can*
  - *i. be a limit of points from X_Σ;*
  - *ii. not be a limit of points from X_Σ.*
- *(b) a union of two arcs with a common end at the point x, having in this point the one-sided tangent lines with angle greater or equal to $2\pi/3$ between them (in this case an order of the point x is equal to 2). Herewith the point x can have*
  - *i. one energetic edge;*
  - *ii. two energetic edges.*

Usually it is not easy to proof that a certain set is a minimizer. At the picture 6 the set M is a circle of radius $R > r$. The set Σ, depicted at this picture is a union of an arc of the circumference of radius $R - r$ and two segments tangent to this circumference. The proof of the fact that set Σ is a minimizer with $R > 5r$ one can find in the work [3].

Note that if M is a finite set of points there is an easy criterion: a connected set Σ is a minimizer for a set M and a number $r > 0$, if the inequality $\mathcal{H}^1(\Sigma) \le \mathcal{H}^1(St(M)) - r\sharp M$ holds, where $\sharp M$ is a cardinality of elements of set M, and $St(M)$ is an arbitrary Steiner set for M. It is easy to show that strong inequality can not be true.

We will show the examples for each option. At the pictures given sets M are drawn by green color, energetic points of minimizer Σ by red color, remaining points of Σ by violet.

1. $x \in S_\Sigma$
   - (a) $\Sigma \cap B_\varepsilon(x)$ is a segment. Point S at the Pic. 6, 1, 2, 3, 4.
   - (b) $\Sigma \cap B_\varepsilon(x)$ is a regular tripod. Point F at the Pic. 3.

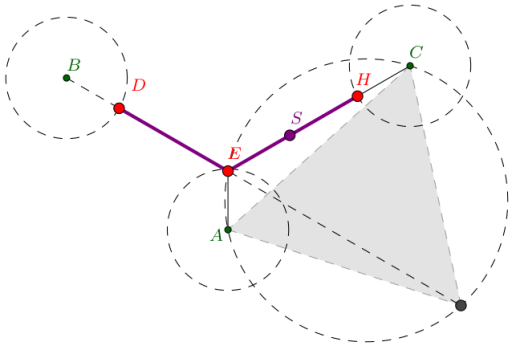

Figure 1: An example when $M := \{A, B, C\}$, $\Sigma = [DE] \cup [EH]$. Point E belongs to X_Σ and has an order 2. The angle $\angle DEH$ is equal to $2\pi/3$.

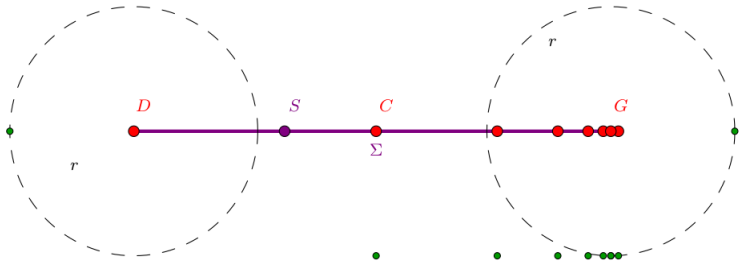

Figure 2: An example when M is a set of two points of the line (DG) and a countable many points of a line parallel to (DG); $\Sigma = [DG]$. Point G is a limit of points X_Σ and belongs to E_Σ.

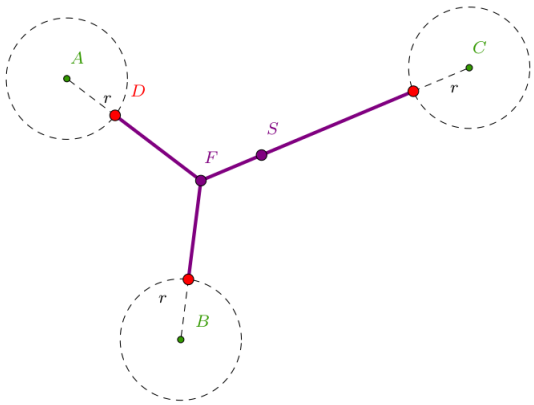

Figure 3: An example where $M := \{A, B, C\}$, Σ is a tripod.

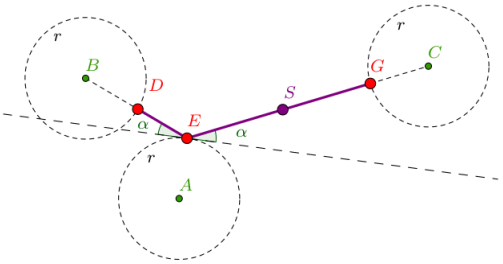

Figure 4: An example where $M := \{A, B, C\}$, $\Sigma = [DE] \cup [EG]$. Point E belongs to X_Σ and has an order 2. An angle $\angle DEH$ is greater than $2\pi/3$.

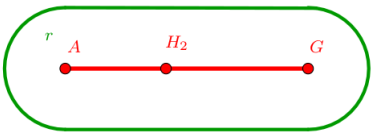

Figure 5: An example when $M := \partial B_r([AG])$, $\Sigma = [AG]$. Point $G \in E_\Sigma$ and $\operatorname{ord}_G \Sigma = 1$.

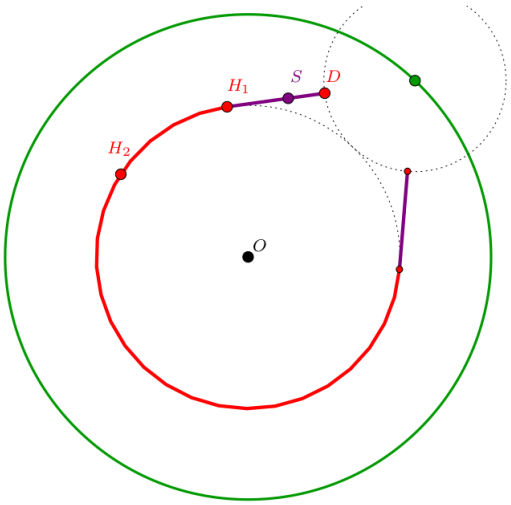

Figure 6: An example when $M = \partial B_R(O)$, where $R > 4.98r$.

2. $x \in X_\Sigma$
   (a) $\Sigma \cap B_\varepsilon(x)$ is a segment with an end at x. Point D at the Pic. 6, 1, 2, 3, 4.
   (b) $\Sigma \cap B_\varepsilon(x)$ is a union of two segments with common end at x and an angle greater between them (point E at the picture Pic. 4) or equal (point E at the picture Pic. 1) to $2\pi/3$ between them.
3. $x \in E_\Sigma$
   (a) $\Sigma \cap B_\varepsilon(x)$ is an arc with an end at x (a point G at the picture Pic. 2, 5). Herewith x can
      i. be a limit of points from X_Σ (point G at the Pic. 2);
      ii. not be a limit of points from X_Σ (point G at the Pic. 5).
   (b) $\Sigma \cap B_\varepsilon(x)$ is a union of two arcs with a common end at the point x, having in this point the one-sided tangent lines with angle greater or equal to $2\pi/3$ between them (points H_1 at Pic. 6 and H_2 at Pic. 6, 5). Herewith the point x can have
      i. one energetic edge (point H_1 at Pic. 6);
      ii. two energetic edges (point H_2 at Pic. 6, 5).

References

[1] G. Bouchitté, C. Jimenez, and R. Mahadevan. Asymptotic analysis of a class of optimal location problems. *J. Math. Pures Appl. (9)*, 95(4):382–419, 2011.

[2] A. Brancolini, G. Buttazzo, F. Santambrogio, and E. Stepanov. Long-term planning versus short-term planning in the asymptotical location problem. *ESAIM Control Optim. Calc. Var.*, 15(3):509–524, 2009.

[3] Danila Cherkashin and Yana Teplitskaya. On the horseshoe conjecture for maximal distance minimizers. *arXiv preprint arXiv:1511.01026*, 2015.

[4] S. Graf and H. Luschgy. *Foundations of quantization for probability distributions*, volume 1730 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 2000.

[5] Alexander O Ivanov and Alexei A Tuzhilin. *Minimal Networks: The Steiner Problem and Its Generalizations*. CRC press, 1994.

[6] M. Miranda, Jr., E. Paolini, and E. Stepanov. On one-dimensional continua uniformly approximating planar sets. *Calc. Var. Partial Differential Equations*, 27(3):287–309, 2006.

[7] E. Paolini and E. Stepanov. Qualitative properties of maximum distance minimizers and average distance minimizers in $\mathbb{R}^n$. *J. Math. Sci. (N. Y.)*, 122(3):3290–3309, 2004. Problems in mathematical analysis.

[8] E. Paolini and E. Stepanov. Existence and regularity results for the Steiner problem. *Calc. Var. Partial Differential Equations*, 46(3-4):837–860, 2013.

[9] Emanuele Paolini, Eugene Stepanov, and Yana Teplitskaya. An example of an infinite Steiner tree connecting an uncountable set. *Advances in Calculus of Variations*, 8(3):267–290, 2015.

[10] A. Suzuki and Z. Drezner. The p-center location problem in an area. *Location science*, 4(1):69 – 82, 1996.

[11] A. Suzuki and A. Okabe. Using Voronoi diagrams. *Drezner, Z. (ed.): Facility location: A survey of applications and methods, Springer series in operations research*, pages 103 – 118, 1995.

The Logics Taught and Used at High Schools Are Not the Same

Antti Valmari[1] *Lauri Hella*[2]

[1] Tampere University of Technology, Mathematics
[2] University of Tampere, Mathematics

Abstract

Typical treatises on propositional and predicate logic do not tell how to deal with undefined expressions, such as division by zero. However, there seems to be a sound (albeit inexplicit) reasoning system that addresses undefined expressions, because equations and inequations involving them are routinely solved in schools and universities without running into fundamental inconsistencies. In this study we discover this *school logic* and formalize its semantics. The need to do so arose when developing software that gives students feedback on every reasoning step of their solution, instead of just telling whether the roots that they finally report are the correct roots. The problem of undefined expressions has been addressed in computer science. However, school logic proves different from those approaches. School logic is based on a Kleene-style third "undefined" truth value and the treatment of "⇒" and "⇔" not as propositional operators but as reasoning operators.

1 Introduction

To discuss the motivation of this study, it is useful to first consider an example. It illustrates the use of the symbols "⇒" and "⇔" when solving equations and inequations in Finnish schools in the 1970s. They denote logical implication and logical equivalence, respectively. The same kind of reasoning is used also today, but without writing these symbols. We assume that the set $\mathbb{R}$ of real numbers is used.

$$\begin{array}{lrcll}
 & \sqrt{x}+\sqrt{x+1} & = & \sqrt{2x+1} & \text{compute square of both sides} \\
\Rightarrow & x+2\sqrt{x}\sqrt{x+1}+x+1 & = & 2x+1 & \text{move } x \text{ and } x+1 \text{ to the right} \\
\Leftrightarrow & 2\sqrt{x}\sqrt{x+1} & = & 0 & \text{divide by 2, compute square} \\
\Rightarrow & x(x+1) & = & 0 & \text{a product is 0 iff a factor is 0} \\
\Leftrightarrow & x=0 & \vee & x+1=0 & \text{move 1 to the right} \\
\Leftrightarrow & x=0 & \vee & x=-1 &
\end{array}$$

Because some reasoning steps used implication instead of equivalence, it is necessary to check the obtained values of x. When assigned in the place of x, 0 makes the original equation hold, so $x=0$ is a valid root to the equation. On the other hand, $x=-1$ makes both sides of the original equation undefined, so it is not a valid root. Therefore, as was taught in Finnish schools, we conclude

$$\sqrt{x}+\sqrt{x+1}=\sqrt{2x+1} \quad \Leftrightarrow \quad x=0\ .$$

When this conclusion is instantiated with $x=-1$, we get

$$\sqrt{-1}+\sqrt{-1+1}=\sqrt{2\cdot(-1)+1} \quad \Leftrightarrow \quad -1=0\ ,$$

that is,

$$\sqrt{-1}+0=\sqrt{-1} \quad \Leftrightarrow \quad -1=0\ .$$

(We refrain from simplifying $\sqrt{-1}+0$ to $\sqrt{-1}$ because $\sqrt{-1}$ is undefined, and until Section 2 we do not know whether $f+0$ may be simplified to f also when f is undefined.) We have concluded that a particular claim involving undefined expressions (that is, $\sqrt{-1}+0=\sqrt{-1}$) is logically equivalent to a false claim ($-1=0$).

This conclusion cannot be expressed in traditional binary propositional and predicate logic that are taught in courses on logic in schools and universities, because they do not have machinery for talking about undefined expressions. What is more, in traditional logic we may reason that if $\varphi \Leftrightarrow \psi$, then $\neg\varphi \Leftrightarrow \neg\psi$. When applied to the above conclusion it yields

$$\neg(\sqrt{-1}+0=\sqrt{-1}) \quad \Leftrightarrow \quad \neg(-1=0)\ .$$

Let T denote "true". If we assume further that $\neg(f=g) \Leftrightarrow f \neq g$ also with undefined expressions, we get

$$\sqrt{-1}+0 \neq \sqrt{-1} \quad \Leftrightarrow \quad \mathsf{T}\ ,$$

which seems counterintuitive indeed.

Fortunately, this is not how logic is used in Finnish schools. Therefore, the logic that is used in Finnish schools — which we will call *school logic* from now on — is not precisely the same as the traditional binary propositional and predicate logic. School logic is implicit in the sense that while it is the logic that is actually used, it is not explicitly taught.

The difference arises with undefined expressions. They are frequent in high school and elementary university mathematics. Just to mention a few examples: $\sqrt[n]{x}$ when n is even and x is negative, $\frac{x}{y}$ when $y=0$, $\log x$ when $x \leq 0$, $\lim_{x\to 0} \sin\frac{1}{x}$, $\min(\emptyset)$, $\max(\mathbb{R})$, and the direction of the null vector.

School logic is apparently sound and consistent, because, if properly used, it seems to never lead to an incorrect conclusion. When a wrong conclusion is obtained, it can be traced to an error in reasoning. For instance, assume that the above example is modified as follows:

$$\begin{array}{rrcll}
 & \sqrt{x}+\sqrt{x+1} & = & \sqrt{2x+1} & \text{both sides are non-negative} \\
\Leftrightarrow & (\sqrt{x}+\sqrt{x+1})^2 & = & (\sqrt{2x+1})^2 & \text{like before} \\
\Leftrightarrow & 2\sqrt{x}\sqrt{x+1} & = & 0 & \text{a product is 0 iff a factor is 0} \\
\Leftrightarrow & 2=0 \ \vee \ \sqrt{x}=0 & \vee & \sqrt{x+1}=0 & \text{solve } x \\
\Leftrightarrow & x=0 & \vee & x=-1 &
\end{array}$$

Now -1 was incorrectly obtained as a root to the equation. This is because the rule "a product is 0 iff a factor is 0" is incorrect. The correct rule is "a product is 0 iff a factor is 0 *and all factors are defined*". It yields correctly

$$\begin{array}{rrcl}
 & 2\sqrt{x}\sqrt{x+1} & = & 0 \\
\Leftrightarrow & (2=0 \vee \sqrt{x}=0 \vee \sqrt{x+1}=0) & \wedge & \mathsf{T} \wedge x \geq 0 \wedge x \geq -1 \\
\Leftrightarrow & (x=0 \vee x=-1) & \wedge & x \geq 0 \\
\Leftrightarrow & x & = & 0
\end{array}$$

In this study we try to recognize the semantics and some rules of school logic. Because school logic is not the same as the traditional logic, at least some traditional law must be rejected either in the domain of discourse (e.g., we might reject $\forall x : x+0=x$), in the logic (e.g., we might reject $\varphi \vee \neg\varphi \Leftrightarrow \mathsf{T}$), or both. We want this to only happen with undefined expressions.

By the nature of our endeavour, many (but not all!) of our arguments are "soft": instead of formal proofs we present examples that try to reveal how students and teachers think. This method suffers from the problem that while an example yielding a clearly unacceptable result provides evidence that school logic does not work like that, there

is the risk that the opposite design choice also leads to unacceptable consequences that we did not see. Our faith on that we have recognized the right semantics is based on extended experience with the use of school logic and on the fact that our semantics have been successfully implemented in the MathCheck tool mentioned below, where they have worked well so far.

The problem with undefined expressions is ubiquitous in computer science, both in programming (e.g., indexing an array out of bounds), formal specification (like programming), and theoretical computer science (e.g., the result of a computation that does not terminate). As a consequence, many computer scientists have made it clear how they deal with them. We will discuss some approaches in this study. They will prove different from school logic in one or more important aspects.

The need to formalize school logic arose in the development of the MathCheck tool [7].[1] Its purpose is to give students feedback on their solutions to elementary mathematics problems. For instance, if the student computes $\sqrt{x^2-6x+9} = \sqrt{(x-3)^2} = x-3$, MathCheck accepts the first "=" but refutes the second with a counter-example such as $x = 0$ where the left hand side yields a positive and the right hand side a negative value. The problem with undefined expressions first arose with such examples as $\frac{x}{x} = 1$ and $\frac{1}{x^2} > 0$. Later on, MathCheck was made able to check solutions to equations using "$\Leftrightarrow$" and "$\Rightarrow$" similarly to towards the beginning of this section (provided that the teacher has given the correct roots to MathCheck). To do that, it was necessary to make MathCheck obey the semantics of school logic, making it necessary to find out the semantics.

In Section 2, we reject the idea of treating undefined as a value in the domain of discourse. The next section tells how MathCheck deals with undefined expressions in relation chains such as $\frac{x\sqrt{x}}{\sqrt{x}} = x > 0$. Section 4 discusses two computer science approaches and introduces the undefined truth value. Based on Kleene's ideas [4, 1], we prove in Section 5 that the language of school logic disallows writing a unary predicate that yields T if and only if its argument is undefined. The syntax and semantics of "$\Leftrightarrow$" and "$\Rightarrow$" are given in Section 6. The topic of Section 7 is reasoning in school logic. Among other things, the result in Section 5 proves important regarding why school logic works well. It also turns out that in many cases, the issue of undefined expressions does not affect reasoning. Section 8 briefly summarizes how and why school logic works.

2 Undefined Value in the Domain

A rather widely used approach is to make all expressions denote total functions by adding an extra element to the domain of discourse, and use the traditional logic. This element is often called *bottom* or *bot* and denoted with $\perp$. When $\perp$ appears as an argument of a function that normally yields a value in the domain of discourse, the function yields $\perp$. Functions that map to truth values (that is, predicates) always yield F (that is, false) or T (that is, true).

This approach yields, among other things, the intuitively acceptable result $\sqrt{-1} \neq 1$. It also yields $\sqrt{-1} = \frac{1}{0}$, because $\sqrt{-1} = \perp = \frac{1}{0}$. In the case of our running example, it yields $\sqrt{-1} + \sqrt{-1+1} = \perp + 0 = \perp$ and $\sqrt{2\cdot(-1)+1} = \sqrt{-1} = \perp$. So this approach deems -1 and even -2 as roots to $\sqrt{x} + \sqrt{x+1} = \sqrt{2x+1}$, which is against our goal.

More generally, let $x_0 \in \mathbb{R}$ and f and g be expressions on x. We now justify that if any subexpression of $f(x_0)$ is undefined, then $f(x_0)$ must be undefined. This is because otherwise $f(x_0)$ would yield some real number a, so $x = x_0$ would be an undesired root to the equation $f(x) = a$. Furthermore, if $f(x_0)$ or $g(x_0)$ is undefined, then $f(x_0) = g(x_0)$,

[1] http://math.tut.fi/mathcheck/ (This talk: . . . /mathcheck/logic.html)

$f(x_0) < g(x_0)$, $f(x_0) \geq g(x_0)$, and so on must not hold, because otherwise $x = x_0$ would be a root to the (in)equation $f(x) = g(x)$, $f(x) < g(x)$, $f(x) \geq g(x)$, or so on.

Because neither $f(x_0) < g(x_0)$ nor $f(x_0) \geq g(x_0)$ holds, we must either reject the familiar law $x < y \Leftrightarrow \neg(x \geq y)$, or make comparisons involving an undefined value yield neither the F nor the T of traditional logic. The default behaviour of the "not-a-number" value of floating point arithmetic [2] (almost) conforms to the first option. Most (but not all!) arithmetic operations involving a not-a-number yield a not-a-number. Each comparison involving at least one not-a-number yields F. Indeed, the recommended way to test whether x holds a not-a-number is to check whether $x = x$ yields F. This implies that both $\sqrt{-1} = 0$ and $\sqrt{-1} \neq 0$ yield false.

With the first option, unary predicates would be partitioned to those that yield F when the argument is undefined, and those that yield T. If $\varphi(x)$ belongs to one class, then $\neg\varphi(x)$ belongs to the opposite class. We find this distinction artificial. It is also clumsy, because one would have to be careful with whether a comparison is within the scope of an odd or even number of negations. Therefore, we choose to keep the law and reject binary logic. We also keep the law $x \neq y \Leftrightarrow \neg(x = y)$.

In the remainder of this study we obey the convention that if an expression maps to the domain of discourse and any of its subexpressions is undefined, then the expression as a whole is undefined. We denote the undefined result with the symbol $\perp$ but do not treat $\perp$ as a value in the domain. In particular, we have neither $\perp = \perp$ nor $\perp \neq \perp$.

That is, each expression f on k variables is interpreted as denoting a total function $(\mathbb{D} \cup \{\perp\})^k \mapsto \mathbb{D} \cup \{\perp\}$ such that $f(\ldots, \perp, \ldots)$ yields $\perp$. However, the interpretation of $f = g$ and $f \neq g$ when f or g yields $\perp$ is non-standard. It will be described in Section 4.

3 Arithmetic Relation Chains in MathCheck

The first version of MathCheck was only capable of checking equality and inequality chains, such as the following:

$$\frac{1}{x^2 + a(a - 2x)} \;=\; \frac{1}{(x - a)^2} \;>\; 0$$

MathCheck checks such chains by trying a number of different *instantiations* (also known as *bindings*), that is, combinations of values of the variables that occur in the chain. In our example, $x = a = 0$ is among them. So MathCheck reports that the second relation ">" fails, because $x = a = 0$ makes its left hand side undefined and right hand side 0. Although $x = a = 0$ makes both sides of the first relation "=" undefined, MathCheck does not report an error there, for the reason discussed very soon. If "assume $x \neq a$;" is added to the front of the chain, MathCheck reports no errors.

In school mathematics, it is common to not specify when an equality or inequality is defined, if both of its sides are undefined with the same instantiations. For example, we usually write "$\tan x = \frac{\sin x}{\cos x}$" instead of "if x is not of the form $\frac{\pi}{2} + n\pi$ where $n \in \mathbb{Z}$, then $\tan x = \frac{\sin x}{\cos x}$". Finding out when an expression is undefined may require non-trivial work, as is the case with $\frac{1}{x^2+x+1}$. When solving equations, it is usually unnecessary, because it is sufficient and easier to find a superset of the roots without worrying about undefined expressions, and finally check which ones among them are valid roots to the original equation, similarly to our example in Section 1. (It is necessary with typical inequations and, for instance, $(\sqrt{x})^2 = x$. Some teachers tell the students to always do it, instead of only doing it when the superset is too big to check.)

Therefore, only if the following hold, MathCheck reports an error saying that one side of an equality or inequality is undefined: the opposite side is defined, the instantiation

in question has not been ruled out by an assume clause, and MathCheck happens to detect the problem. We mentioned the latter because MathCheck only tries to find most errors instead of all errors. Finding all errors is often difficult or even uncomputable. MathCheck is currently good in finding errors with an interval of counter-examples such as $\sqrt{(3x-100)(3x-101)} \geq 0$, to which it gives the counter-example $x = \frac{167}{5} \approx 33.4$. However, it is not good in finding errors with a finite or countably infinite number of counter-examples, such as $\frac{x+3}{x+3} = 1$.

Before attempting to refute an (in)equality by finding a counter-example, MathCheck tries to prove it correct using simple rules such as $0f = 0$. However, MathCheck must not (and does not) reason $0\sqrt{x} = 0$ without an assume clause that rules out the negative values of x. This is implemented by treating each expression as a total function from a subset of $\mathbb{R}^k$ to $\mathbb{R}$, where k is the number of variables that occur in the (in)equality chain and its optional assume clause. (For simplicity, we ignore here the fact that MathCheck also has integer variables.) The domain of the function is where the expression is defined, intersected with the contents of the assume clause. The assume clause must be always defined, in the sense discussed in Section 4.

In other words, $f < g$ in the scope of the assume clause φ is interpreted as

$$(\mathrm{dom}_f \wedge \mathrm{dom}_g \wedge \varphi \wedge f < g) \ \vee \ \neg\varphi \ \vee \ (\neg\mathrm{dom}_f \wedge \neg\mathrm{dom}_g) \ ,$$

where undefined propositional values are treated as will be explained in Section 4, and $\mathrm{dom}_{\ldots}$ is a predicate that tells when $\ldots$ is defined. It is obtained recursively via such rules as $\mathrm{dom}_{f/g} \Leftrightarrow \mathrm{dom}_f \wedge \mathrm{dom}_g \wedge g \neq 0$. A similar claim also holds for the other five relations "$\leq$", "$=$", "$\neq$", "$\geq$", and "$>$".

Application of $0f = 0$ as a rewrite rule from left to right is treated as universally legal. However, when it is applied to $0\sqrt{x}$, the resulting 0 has the domain $x \geq 0$ (unless the assume clause restricts it further). We denote it with $0 : \{x \in \mathbb{R} \mid x \geq 0\} \mapsto \mathbb{R}$. MathCheck does not treat it as the same as the function $\mathbb{R} \mapsto \mathbb{R}$ whose value is the constant 0, because of different domains. We denote the latter with $0 : \{x \in \mathbb{R} \mid \mathsf{T}\} \mapsto \mathbb{R}$ or $0 : \mathbb{R} \mapsto \mathbb{R}$.

This means that in the case of $0\sqrt{x} = 0$ without any assume clause, the proof stage of MathCheck transforms the left hand side to $0 : \{x \in \mathbb{R} \mid x \geq 0\} \mapsto \mathbb{R}$. It is not the same as the right hand side, which is $0 : \mathbb{R} \mapsto \mathbb{R}$. This makes MathCheck enter the counter-example seeking stage, reporting an error after trying a negative value for x. In the case of "assume $x > 0$; $0\sqrt{x} = 0$", the proof stage of MathCheck transforms the left hand side to $0 : \{x \in \mathbb{R} \mid x > 0 \wedge x \geq 0\} \mapsto \mathbb{R}$ and optimizes it to $0 : \{x \in \mathbb{R} \mid x > 0\} \mapsto \mathbb{R}$. Because also the right hand side is $0 : \{x \in \mathbb{R} \mid x > 0\} \mapsto \mathbb{R}$, MathCheck recognizes them as the same function and considers the equality proven. (A potential topic for future development is to make MathCheck to try to find an instantiation that is in the domain of one but not of the other side of an (in)equality, when it failed to prove that the domains are the same but also failed to find a counter-example to the (in)equation. This is not trivial, because the domains are represented as logical expressions, such as $x^2 + x + 1 \neq 0$.)

This treatment of undefined expressions has been working well. However, when features involving similar reasoning as in Section 1 were added to MathCheck, the problem of undefined expressions emerged again. Consider $11\sqrt{|x|+1} = 25 - x$. A natural step towards solving it is $(x < 0 \wedge 11\sqrt{1-x} = 25 - x) \vee (x \geq 0 \wedge 11\sqrt{x+1} = 25 - x)$. This rules out the idea of using a domain where everything is defined, because $x = 3$ is a root to the original equation but not in the domain of $11\sqrt{1-x}$.

4 The Undefined Truth Value

In the remainder of this study, "$\lessgtr$" denotes any of "$<$", "$\leq$", "$=$", "$\neq$", "$\geq$", and "$>$". Consider the axiom that says that every non-zero real number has an inverse:

$$\forall x \neq 0 : x \cdot \frac{1}{x} = 1 \ .$$

This formulation uses a so-called *bounded quantifier* of the form $\forall x \lessgtr f : \varphi$, where x is a variable, f is an expression (also called *term* in logic parlance), and φ is a formula. By definition, it is an abbreviation of $\forall x : x \lessgtr f \rightarrow \varphi$, where "$\rightarrow$" is the implication in traditional logic. Therefore, written out in full, the axiom is

$$\forall x : x \neq 0 \rightarrow x \cdot \frac{1}{x} = 1 \ .$$

By usual laws of logic, it is equivalent to

$$\forall x : x \cdot \frac{1}{x} = 1 \vee x = 0 \ .$$

By the definition of the universal quantifier, it means that $x \cdot \frac{1}{x} = 1 \vee x = 0$ holds for every $x \in \mathbb{R}$. For each $x \in \mathbb{R} \setminus \{0\}$ it yields $\mathsf{T} \vee \mathsf{F}$, which is unproblematic. On the other hand, for $x = 0$ it yields $0 \cdot \frac{1}{0} = 1 \vee \mathsf{T}$, which is an undefined logical expression. Intuitively, the original formulation says that the part after the ":" must not be evaluated when $x = 0$. However, unwinding the definition of bounded quantifiers reveals that it is evaluated.

Vienna Development Method (VDM) [3] is a formal specification method that uses logic and set theory concepts for the specification, design, and implementation of software. According to [3, Sect. 3.3], undefined expressions are interpreted as never yielding a value. This is natural to computer scientists, because a program may fail to terminate. For instance, division of a natural number by a natural number can be implemented as repeated subtraction. Then division by zero yields an infinite loop.

In VDM, binary Boolean operators are treated as if both arguments were evaluated simultaneously. If the result that comes first suffices to determine the result of the operation, then the evaluation of the other argument is aborted. For instance, $x \neq 0 \rightarrow x \cdot \frac{1}{x} = 1$ yields T when $x = 0$, because $0 \neq 0$ yields eventually F and both $\mathsf{F} \rightarrow \mathsf{F}$ and $\mathsf{F} \rightarrow \mathsf{T}$ yield T. So the never-ending evaluation of $0 \cdot \frac{1}{0} = 1$ is aborted and T is returned as the result. On the other hand, $\frac{1}{0} = \frac{2}{0} \vee \frac{1}{0} \neq \frac{2}{0}$ does not yield any truth value, because neither argument of the disjunction terminates.

On [3, p. 70–71], the following truth tables are given and called the *logic of partial functions*. Because the usage of "$\Rightarrow$" and "$\Leftrightarrow$" in VDM is the same as the usage of "$\rightarrow$" and "$\leftrightarrow$" in MathCheck and different from the usage of "$\Rightarrow$" and "$\Leftrightarrow$" in MathCheck and school logic, we have replaced "$\rightarrow$" for "$\Rightarrow$" and "$\leftrightarrow$" for "$\Leftrightarrow$". We have also replaced F for false, T for true, and U for $*$.

$\wedge$	F	U	T
F	F	F	F
U	F	U	U
T	F	U	T

$\vee$	F	U	T
F	F	U	T
U	U	U	T
T	T	T	T

$\neg$	
F	T
U	U
T	F

$\rightarrow$	F	U	T
F	T	T	T
U	U	U	T
T	F	U	T

$\leftrightarrow$	F	U	T
F	T	U	F
U	U	U	U
T	F	U	T

According to [3], "In these tables, the absence of a value is marked by $*$; but there is no sense in which this is a new value — it is just a reminder that no value is available." The present authors disagree with not calling it a value, but do appreciate the intuition behind this statement. Indeed, every entry in the tables that corresponds to one or two U

arguments can be obtained by trying both F and T in the place of the U. If the result can be both F and T, then U is written in the table, otherwise the unique result is written in the table. If both arguments are U, then all four combinations (F,F), (F,T), (T,F), and (T,T) are tried. Entries that correspond to only non-U arguments have the traditional value. The tables match Kleene's ternary logic K_3^S [4, 1].

It is easy to check from the above truth tables that $\varphi \leftrightarrow \psi$ denotes the same function as $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$. Similarly, $\varphi \rightarrow \psi$ denotes the same as $\neg\varphi \vee \psi$ and $\varphi \vee \psi$ denotes the same as $\neg(\neg\varphi \wedge \neg\psi)$. So, as in traditional binary logic, "$\wedge$" and "$\neg$" suffice as elementary operators. However, unlike traditional logic, not all functions from truth values to truth values can be constructed. We will see in the next section that the function notU cannot be constructed such that $\mathsf{notU}(\mathsf{U})$ yields F and $\mathsf{notU}(\mathsf{F})$ and $\mathsf{notU}(\mathsf{T})$ yield T.

On the other hand, for any expression f on the real number variables $x_1, \ldots, x_k$, it is possible to write a predicate dom_f on the same variables that yields T when f is defined and F otherwise. It is obtained recursively via such rules as $\mathrm{dom}_{f/g} \Leftrightarrow \mathrm{dom}_f \wedge \mathrm{dom}_g \wedge g \neq 0$. When g is undefined, this yields F thanks to dom_g, although $g \neq 0$ yields U. For each variable symbol x_i we have $\mathrm{dom}_{x_i} \Leftrightarrow \mathsf{T}$. This has been implemented in MathCheck, as a part of the approach discussed in Section 3.

This is extended to predicates by defining $\mathrm{dom}_{f \lesseqgtr g}$ as $\mathrm{dom}_f \wedge \mathrm{dom}_g$, $\mathrm{dom}_{\neg\varphi}$ as dom_φ, and $\mathrm{dom}_{\varphi\wedge\psi}$ as $(\mathrm{dom}_\varphi \wedge \mathrm{dom}_\psi) \vee (\mathrm{dom}_\varphi \wedge \neg\varphi) \vee (\mathrm{dom}_\psi \wedge \neg\psi)$. This does not contradict the unconstructibility of notU, because dom_φ is not $\{\mathsf{F},\mathsf{U},\mathsf{T}\} \mapsto \{\mathsf{F},\mathsf{U},\mathsf{T}\}$ but $\mathbb{R}^k \mapsto \{\mathsf{F},\mathsf{U},\mathsf{T}\}$ for some $k \in \mathbb{N}$. That is, although $\mathrm{dom}_\varphi(x_1,\ldots,x_k)$ can be constructed, it usually cannot be represented in the form $\xi(\varphi(x_1,\ldots,x_k))$ for some ξ.

This approach facilitates formulae that say "if a claim is defined, return its truth value, otherwise return F" and "if a claim is defined, return its truth value, otherwise return T":

$$\mathrm{dom}_f \wedge \mathrm{dom}_g \wedge f \lesseqgtr g \qquad\qquad \neg\mathrm{dom}_f \vee \neg\mathrm{dom}_g \vee f \lesseqgtr g$$

The latter can also be written as $\mathrm{dom}_f \wedge \mathrm{dom}_g \rightarrow f \lesseqgtr g$. Unlike in Section 2, the truth value of $\varphi(\bot)$ is not determined by the parity of the number of negation operators, but on the user's choice to use one, the other, or neither of the above two options.

We will use the above truth tables and dom *in the rest of this study. When we say that a formula* holds*, we mean that it yields* T*. An arithmetic relation* $f \lesseqgtr g$ *yields* U *if and only if* f *or* g *is undefined.*

The Z formal specification method has adopted a subtly different convention [6, p. 40–41]: "If one or both of E_1 and E_2 are undefined, then we say that the predicates $E_1 = E_2$ and $E_1 \in E_2$ are *undetermined*: we do not know whether they are true or false. This does not mean that the predicates have some intermediate status in which they are 'neither true nor false', simply that we have chosen not to say whether they are true or not." The text continues by presenting and discussing the formulae $x \in \mathrm{dom}f \wedge f(x) = y$ and $x \in \mathrm{dom}f \Rightarrow f(x) = y$, which are essentially the same as what we gave above.

The subtle difference is that while an undefined predicate invocation such as $\frac{1}{0} = \frac{2}{0}$ yields neither F nor T in VDM, in Z it yields one of them but it is not known which one. VDM uses a ternary logic, but Z uses the traditional binary logic [6, p. 69]. So the law of the excluded middle holds in Z, implying that $\frac{1}{0} = \frac{2}{0} \vee \neg(\frac{1}{0} = \frac{2}{0}) \Leftrightarrow \mathsf{T}$ in Z but not in VDM. However, the intention is clearly that this kind of formulae should not be written, where a potentially undefined expression is not guarded by "$x \in \mathrm{dom}f \wedge$" or "$x \in \mathrm{dom}f \Rightarrow$". So from the point of view of the intended use of Z and VDM, this difference does not matter.

In Section 1 we concluded that in school logic,

$$\sqrt{x} + \sqrt{x+1} \;=\; \sqrt{2x+1} \quad\Leftrightarrow\quad x = 0\ .$$

In VDM, this logical equivalence is not true. Indeed, it yields no truth value, because its left hand side and thus the equivalence as a whole yields no truth value with negative values of x. In Z, the truth value of this logical equivalence is not known, because the truth value of its left hand side is not known for negative values of x. Both in VDM and Z, the intention is that the claim is written as

$$x \geq 0 \ \wedge \ \sqrt{x} + \sqrt{x+1} = \sqrt{2x+1} \quad \Leftrightarrow \quad x = 0 \ .$$

This formulation is true in both.

However, from the point of view of school logic, this formulation suffers from the problem that it looks like $x \geq 0$ was part of the original question, although it was not. If the original question were "find the positive values of x such that $\sqrt{x}+\sqrt{x+1} = \sqrt{2x+1}$", then it would be natural to express the (in this case non-existent) roots to the equation as

$$x > 0 \ \wedge \ \sqrt{x} + \sqrt{x+1} = \sqrt{2x+1} \quad \Leftrightarrow \quad \mathsf{F} \ .$$

Similarly, $x \geq 0 \wedge \ldots$ makes it look like the original question were "find the non-negative ...".

Furthermore, as was mentioned in Section 3, finding out the domain in an explicit form is usually unnecessary work, because it suffices to solve the equation using "$\Rightarrow$" at places and finally check which of the obtained tentative roots indeed are roots. The tentative roots and the results of their checking could be represented in VDM and Z as

$$x = 0 \ \wedge \ x \neq -1 \ \wedge \ \sqrt{x} + \sqrt{x+1} = \sqrt{2x+1} \quad \Leftrightarrow \quad x = 0 \ .$$

However, without knowing why this notation is used, it would seem odd. Definitely it is not what is done in Finnish schools.

The following expresses the intention clearly:

$$\mathrm{dom}_{\sqrt{x}+\sqrt{x+1}} \ \wedge \ \mathrm{dom}_{\sqrt{2x+1}} \ \wedge \ \sqrt{x} + \sqrt{x+1} = \sqrt{2x+1} \quad \Leftrightarrow \quad x = 0 \ .$$

Unfortunately, it is clumsy, as it requires writing each side of the equation twice. It would be better if the restriction to the domains were inherent in the notation, without having to be explicitly stated. That is, we want to interpret

$$\sqrt{x} + \sqrt{x+1} \ = \ \sqrt{2x+1} \quad \Leftrightarrow \quad x = 0$$

as saying that among those values of x that make both sides of the equation defined, 0 is the only root.

MathCheck uses the "undefined" truth value U and the truth tables in this section, but, as will be explained in Section 6, treats "$\Rightarrow$" and "$\Leftrightarrow$" differently from VDM and Z. So in MathCheck, $0 \cdot \frac{1}{0} = 1 \vee 0 = 0$ yields $\mathsf{U} \vee \mathsf{T}$ yields T. Furthermore, $\frac{1}{0} = \frac{2}{0}$ and even $\frac{1}{0} = \frac{1}{0} \vee \neg(\frac{1}{0} = \frac{1}{0})$ yield U in MathCheck.

MathCheck uses precise rational number arithmetic when it can, and otherwise a representation consisting of two double-precision floating point values such that the precise value is between them. The representation also allows expressing the possibility that the value is undefined. For instance, on some computer the lower bound to the value of $\sin^2 1 + \cos^2 1 - 1$ is approximately -10^{-15} and the upper bound approximately $9 \cdot 10^{-16}$. Consequently, the value of $\sqrt{\sin^2 1 + \cos^2 1 - 1}$ is undefined or between 0 and approximately $3 \cdot 10^{-8}$. So the comparison $\sqrt{\sin^2 1 + \cos^2 1 - 1} > 0$ may yield any truth value. For this reason, the truth value data type of MathCheck can represent any non-empty combination of F, U, and T. In this sense, MathCheck has both the unknown truth value (the combination of all three) and three partially known truth values (the combinations of any two) as distinct from the undefined truth value.

5 Regularity

In this section we present an extension of a result by Kleene [1, Lemma 1.32].

We first define $\forall x : \varphi$ in the obvious way: if any $x \in \mathbb{R}$ makes φ yield F, then $\forall x : \varphi$ yields F; otherwise, if any $x \in \mathbb{R}$ makes φ yield U, then $\forall x : \varphi$ yields U; otherwise $\forall x : \varphi$ yields T. Then $\exists x : \varphi$ is defined as $\neg\forall x : \neg\varphi$.

When an argument to a predicate is undefined, we denote it with $\perp$.

Definition 1. The 0-ary predicates F, U, and T are *regular*. A function $\varphi : \{\mathsf{F}, \mathsf{U}, \mathsf{T}\} \mapsto \{\mathsf{F}, \mathsf{U}, \mathsf{T}\}$ is *regular* if and only if either $\varphi(\mathsf{F})$, $\varphi(\mathsf{U})$, and $\varphi(\mathsf{T})$ yield the same result (that is, $\varphi(x)$ is a constant) or $\varphi(\mathsf{U})$ yields U. Let $\mathbb{D}$ denote the domain of discourse. A predicate $\varphi : \mathbb{D} \cup \{\perp\} \mapsto \{\mathsf{F}, \mathsf{U}, \mathsf{T}\}$ is *regular* if and only if either $\varphi(x)$ yields the same result for all $x \in \mathbb{D} \cup \{\perp\}$ or $\varphi(\perp)$ yields U. A k-ary predicate with $k > 1$ is *regular* if and only if every unary predicate obtainable from it by fixing all but one arguments is regular.

The predicate notU in Section 4 is not regular. The following theorem implies its unconstructibility.

Theorem 2. *If φ maps to $\{\mathsf{F}, \mathsf{U}, \mathsf{T}\}$ and has been constructed only using arithmetic operators, "$<$", "$\leq$", "$=$", "$\neq$", "$\geq$", "$>$", F, U, T, "$\wedge$", "$\vee$", "$\neg$", "$\rightarrow$", "$\leftrightarrow$", "$\forall$", and "$\exists$", then it is regular.*

Proof. An expression with no free variables denotes either F, U, or T, so the denoted predicate is regular. The case of expressions with two or more free variables follows immediately from the case with one free variable. From now on we assume that the expression has precisely one free variable x. It is either real-valued extended with $\perp$, or truth-valued.

Arithmetic operators can only occur within subexpressions of the form $f(x) \lesseqgtr g(x)$. Assume that x occurs in $f(x)$. If x is $\perp$, then $f(x)$ is $\perp$ and $f(x) \lesseqgtr g(x)$ yields U. Similar reasoning applies if x occurs in $g(x)$. So $f(x) \lesseqgtr g(x)$ is regular.

If $\varphi(x)$ is regular, then also $\neg\varphi(x)$ is, because $\neg\mathsf{U}$ yields U and the negation of a constant predicate is a constant predicate.

Assume that also $\psi(x)$ is regular. If $\varphi(x)$ yields F for every x or $\psi(x)$ does so, then $\varphi(x) \wedge \psi(x)$ yields F for every x. If both $\varphi(x)$ and $\psi(x)$ yield T for every x, then $\varphi(x) \wedge \psi(x)$ yields T for every x. In the remaining case, when x is $\perp$ or U, one of them yields U and the other yields U or T, so also $\varphi(x) \wedge \psi(x)$ yields U. We conclude that $\varphi(x) \wedge \psi(x)$ is regular.

Assume that $\varphi(y, x)$ is regular, where y is a real-valued variable. If there is $y \in \mathbb{R}$ such that $\varphi(y, x)$ yields F for every x, then $\forall y : \varphi(y, x)$ yields F for every x. If $\varphi(y, x)$ yields T for every $y \in \mathbb{R}$ and x, then $\forall y : \varphi(y, x)$ yields T for every x. In the remaining case, when x is $\perp$ or U and $y \in \mathbb{R}$, $\varphi(y, x)$ yields U or T, and U is obtained with at least one $y \in \mathbb{R}$. So also $\forall y : \varphi(y, x)$ yields U. We conclude that $\forall y : \varphi(y, x)$ is regular.

Because the remaining propositional operators and "$\exists$" can be constructed from "$\neg$", "$\wedge$", and "$\forall$", also they only yield regular predicates. □

6 The Syntax and Semantics of "$\Rightarrow$" and "$\Leftrightarrow$"

In VDM and Z, $\varphi \Rightarrow \psi$ and $\varphi \Leftrightarrow \psi$ are logical operations that obey similar syntactic rules as $\varphi \wedge \psi$ and, given the truth values of φ and ψ, yield a truth value. The following table is given on page 4 in [3], and the following grammar on page 69 in [6]. The latter remarks "These connectives are shown in decreasing order of binding power; the connective $\Rightarrow$ associates to the right, and the other binary ones associate to the left." We failed to find

out how "⇒" associates in VDM, but examples such as $E_1 \Rightarrow (E_2 \Rightarrow E_3)$ on page 298 suggest that not to the right. For the other binary operators the direction of associativity does not matter, because they are associative.

operator	read as	priority
¬	not	highest
∧	and	
∨	or	
⇒	implies	
⇔	is equivalent to	lowest

```
Predicate ::= ¬ Predicate
            | Predicate ∧ Predicate
            | Predicate ∨ Predicate
            | Predicate ⇒ Predicate
            | Predicate ⇔ Predicate
```

In school logic and MathCheck (and also often in universities), $\varphi \Rightarrow \psi$ and $\varphi \Leftrightarrow \psi$ are not like that. Instead, they are reasoning steps that obey similar syntactic rules as $x \leq y$ and are either valid or invalid. VDM-like implication and equivalence are written as "→" and "↔" in MathCheck. Their precedences are like in VDM and Z, but also "→" associates to the left. As a grammar,

```
Reasoning ::=
      Predicate ⇒ Predicate
    | Predicate ⇔ Predicate
    | Reasoning ⇒ Predicate
    | Reasoning ⇔ Predicate
```

```
Predicate ::= Atom
            | ¬ Predicate
            | Predicate ∧ Atom
            | Predicate ∨ Atom
            | Predicate → Atom
            | Predicate ↔ Atom
```

That "⇔" is different from "↔" is clearly seen by considering

$$2x - 6 = 0 \;\Leftrightarrow\; 2x = 6 \quad \text{and} \quad 2x - 6 = 0 \;\Leftrightarrow\; 2x = 6 \;\Leftrightarrow\; x = 3\ .$$

Both are valid reasonings. Furthermore, using the truth table in Section 4, $2x - 6 = 0 \leftrightarrow 2x = 6$ yields T for every $x \in \mathbb{R}$. On the other hand, applying similar syntactic rules to $2x - 6 = 0 \leftrightarrow 2x = 6 \leftrightarrow x = 3$ as with "∧", when $x \neq 3$ we get $\mathsf{F} \leftrightarrow \mathsf{F} \leftrightarrow \mathsf{F}$, that is, $(\mathsf{F} \leftrightarrow \mathsf{F}) \leftrightarrow \mathsf{F}$ yielding $\mathsf{T} \leftrightarrow \mathsf{F}$ yielding F. So it does not yield T for every $x \in \mathbb{R}$.

There is also the difference that "⇒" and "⇔" are often interpreted relative to a context. For instance, $2x + |x| = (x-7)^2 + 3$ may be split to two cases, $x < 0$ and $x \geq 0$. If "⇔" is interpreted relative to the context, the former can be solved by reasoning $2x - x = (x-7)^2 + 3 \Leftrightarrow x = (x-7)^2 + 3 \Leftrightarrow \mathsf{F}$, because $x < 0$ and $(x-7)^2 + 3 > 0$. If the interpretation were not relative to the context, $x = (x-7)^2 + 3 \Rightarrow \mathsf{F}$ would be incorrect because $\frac{15 \pm \sqrt{17}}{2}$ are counter-examples, so we would lack handy notation.

When "⇔" and "↔" are used as distinct symbols in binary logic, the usual convention is that $\varphi_1 \Leftrightarrow \varphi_2 \Leftrightarrow \ldots \Leftrightarrow \varphi_n$ is valid if and only if for each $1 \leq i < n$ and for each instantiation allowed by the context, $\varphi_i \leftrightarrow \varphi_{i+1}$ holds. Also "⇒" may appear in the chain, then $\varphi_i \rightarrow \varphi_{i+1}$ must hold. In the case of "⇒" and $n = 2$, this can be thought of as the Modus Ponens rule.

Unfortunately, things are more complicated with our ternary logic. For instance, $\frac{1}{x} + 1 > 1 \Rightarrow \frac{1}{x} > 0$ should be valid, because both sides are defined for the same values of x, the claim holds for those values of x, and our goal is, to the extent possible, to liberate the user from the obligation to explicitly write the domains. However, $\frac{1}{x} + 1 > 1 \rightarrow \frac{1}{x} > 0$ yields U, not T, when $x = 0$. This cannot be sorted out by considering $\varphi \Rightarrow \psi$ valid if and only if for every instantiation allowed by the context, $\varphi \rightarrow \psi$ yields U or T, because then both $x < 0 \Rightarrow \sqrt{x} < 0$ and $\sqrt{x} < 0 \Rightarrow x \geq 0$ would be treated as valid, yielding $x < 0 \Rightarrow x \geq 0$ as valid by the transitivity of "⇒".

The semantics of "⇒" and "⇔" in MathCheck and school logic can be derived from the following four easily acceptable principles from traditional logic (with the adaptation that

the third principle mentions U and instantiation with undefined expressions is allowed), together with one example of what we want to achieve:

1. $\varphi \Leftrightarrow \psi$ if and only if $\varphi \Rightarrow \psi$ and $\psi \Rightarrow \varphi$.
2. If $\varphi \Rightarrow \psi$ and $\psi \Rightarrow \xi$, then $\varphi \Rightarrow \xi$.
3. $\varphi \Rightarrow \psi$ if and only if for every instantiation $\bar{x}$ of the variables that occur in φ or ψ that is allowed by the context, we have $\varphi(\bar{x}) \Rightarrow \psi(\bar{x})$, where $\varphi(\bar{x}) \in \{\mathsf{F}, \mathsf{U}, \mathsf{T}\}$ denotes the truth value of φ with the instantiation $\bar{x}$, and similarly with ψ.
4. $\mathsf{F} \Rightarrow \mathsf{F}$, $\mathsf{F} \Rightarrow \mathsf{T}$, and $\mathsf{T} \Rightarrow \mathsf{T}$, but not $\mathsf{T} \Rightarrow \mathsf{F}$.
5. $\frac{1}{x} = \frac{1}{x} \Leftrightarrow x \neq 0$.

Theorem 3. *The five principles above lead to the following validity tables.*

$\Rightarrow$	F	U	T
F	√	√	√
U	√	√	√
T	–	–	√

$\Leftrightarrow$	F	U	T
F	√	√	–
U	√	√	–
T	–	–	√

Proof. By (3), it suffices to investigate the cases where φ and ψ are among F, U, and T. The corners of the table for "$\Rightarrow$" are given by (4). Assigning $x = 0$ in (5) yields $\mathsf{U} \Leftrightarrow \mathsf{F}$. From it (1) yields $\mathsf{U} \Rightarrow \mathsf{F}$ and $\mathsf{F} \Rightarrow \mathsf{U}$. From them (2) implies $\mathsf{U} \Rightarrow \mathsf{U}$. Furthermore, $\mathsf{U} \Rightarrow \mathsf{F}$, $\mathsf{F} \Rightarrow \mathsf{T}$, and (2) yield $\mathsf{U} \Rightarrow \mathsf{T}$. Finally, $\mathsf{T} \Rightarrow \mathsf{U}$ is not valid, because if we assume that it is valid, then we get $\mathsf{T} \Rightarrow \mathsf{U} \Rightarrow \mathsf{F}$, contradicting (4). The table for "$\Leftrightarrow$" follows by (1). □

That is, an implication is invalid if and only if there is an instantiation that makes its left hand side true and its right hand side either false or undefined. This approach is similar to the not-a-number in Section 2 in that undefined is eventually treated as equivalent to false. However, it is converted to false at the level of reasoning steps, not at the level of predicates. At the level of predicates it is represented via the undefined truth value. This protects it from being transformed to true, if the undefined relation is in the scope of negation.

Consider $x = 0 \Rightarrow \neg(\frac{1}{x} \neq 0) \Rightarrow \frac{1}{x} = 0$, for instance. Both in the logic of the not-a-number and in the logic of MathCheck, it yields $\mathsf{F} \Rightarrow \neg\mathsf{T} \Rightarrow \mathsf{F}$ when $x \neq 0$ (where $\neg\mathsf{T}$ is F). When $x = 0$ it yields $\mathsf{T} \Rightarrow \neg\mathsf{F} \Rightarrow \mathsf{F}$ with the not-a-number (where $\neg\mathsf{F}$ is T) and $\mathsf{T} \Rightarrow \neg\mathsf{U} \Rightarrow \mathsf{U}$ with MathCheck (where $\neg\mathsf{U}$ is U). So the first implication is valid and the second is invalid with the not-a-number, and the other way round with MathCheck. We find the MathCheck convention more natural, because it keeps the law $f \neq g \Leftrightarrow \neg(f = g)$ universally valid, and does not treat $\neg(\frac{1}{0} \neq 0)$ as true.

Perhaps the most surprising detail is that $\mathsf{U} \Rightarrow \mathsf{F}$ is valid in MathCheck and school logic. It cannot be avoided, because we want roots to an equation to be expressible as $f(x) = g(x) \Leftrightarrow x = x_1 \vee \cdots \vee x = x_n$ without having to explicitly restrict the left hand side to its domain. It is different from the behaviour of "$\rightarrow$" in both Kleene's K_3^S and Łukasiewicz's ternary logic L_3 [5, 1], where $\mathsf{U} \rightarrow \mathsf{F}$ yields U. It contradicts the intuitive idea that undefined is between false and true. Therefore, MathCheck and school logic reject that idea.

We have $\frac{1}{x} \neq 0 \Leftrightarrow x \neq 0$ and in particular $\frac{1}{0} \neq 0 \Leftrightarrow \mathsf{F}$, which almost — but only almost — says that $\frac{1}{0} = 0$, which cannot be accepted. According to the validity table of "$\Leftrightarrow$", when one of φ and ψ is U and the other is F, then $\varphi \Leftrightarrow \psi$ does not yield $\neg\varphi \Leftrightarrow \neg\psi$. So the incorrect conclusion $\frac{1}{0} = 0 \Leftrightarrow \mathsf{T}$ is not obtained. Each claim of the form $\frac{1}{0} = x$, $\frac{1}{0} \neq x$, $\frac{1}{0} < x$, and so on is equivalent to F, but so is also each of their negations.

The claim that φ yields T can be written both as φ and as $\varphi \Leftrightarrow \mathsf{T}$. Similarly, $\neg\varphi$ and $\neg\varphi \Leftrightarrow \mathsf{T}$ express that φ yields F. On the other hand, $\varphi \Leftrightarrow \mathsf{F}$ expresses that φ yields either F or U. That φ yields U can be expressed as $\varphi \vee \neg\varphi \Leftrightarrow \mathsf{U}$ and as $\varphi \vee \neg\varphi \Leftrightarrow \mathsf{F}$. We saw in Section 5 that it cannot be expressed as a function that inputs the truth value of φ and is only constructed using the five propositional operators.

7 Reasoning in School Logic

In Section 4 the predicates dom_f and dom_φ were introduced that yield T when the expression f and the truth-valued function φ are defined, and F otherwise. The only way to violate $\varphi \Rightarrow \psi$ is that, for some instantiation allowed by the context, φ yields T and ψ yields either U or F, that is, $\varphi \wedge (\neg\mathrm{dom}_\psi \vee \neg\psi)$ holds. Consequently, $\varphi \Rightarrow \psi$ is valid if and only if $\neg\mathrm{dom}_\varphi \vee \neg\varphi \vee \psi$ holds for every allowed instantiation. That is,

$$\varphi \Rightarrow \psi \text{ is valid if and only if } (\mathrm{dom}_\varphi \wedge \varphi) \to \psi \text{ holds}$$
$$\text{for every instantiation that is allowed by the context.}$$

The predicates $\varphi \wedge (\neg\mathrm{dom}_\psi \vee \neg\psi)$ and $\neg\mathrm{dom}_\varphi \vee \neg\varphi \vee \psi$ illustrate that the law of the excluded middle has been replaced by $\neg\varphi \vee \neg\mathrm{dom}_\varphi \vee \varphi$. Therefore, the opposite of φ is $\neg\mathrm{dom}_\varphi \vee \neg\varphi$, which is not necessarily $\neg\varphi$.

If a predicate never yields U, let us call it *Boolean*. Some examples are dom_φ, $\mathrm{dom}_\varphi \wedge \varphi$, and $\neg\mathrm{dom}_\varphi \vee \varphi$. Logically equivalent Boolean sub-formulae can be replaced by each other, that is, if φ and ψ are Boolean and $\varphi \Leftrightarrow \psi$, then $\zeta(\varphi) \Leftrightarrow \zeta(\psi)$. Replacement is not necessarily correct with non-Boolean sub-formulae, because $\mathsf{F} \Leftrightarrow \mathsf{U}$ is valid but $\neg\mathsf{F} \Leftrightarrow \neg\mathsf{U}$ is invalid.

The truth and validity tables in the previous sections also yield

$$\varphi \;\Leftrightarrow\; \mathrm{dom}_\varphi \wedge \varphi \quad \text{and} \quad \mathrm{dom}_\varphi \;\Leftrightarrow\; \varphi \vee \neg\varphi \;.$$

Because $\mathrm{dom}_\varphi \wedge \varphi$ is Boolean, the first law can often be used to reduce a reasoning task into binary logic. To illustrate this, we prove that the following three yield each other:

$$\begin{array}{rcl} \varphi & \Rightarrow & \psi \\ \mathrm{dom}_\varphi \wedge \varphi & \Rightarrow & \mathrm{dom}_\psi \wedge \psi \\ \neg\mathrm{dom}_\psi \vee \neg\psi & \Rightarrow & \neg\mathrm{dom}_\varphi \vee \neg\varphi \end{array}$$

The second can be derived from the first using the above-mentioned law twice: $\mathrm{dom}_\varphi \wedge \varphi \Leftrightarrow \varphi \Rightarrow \psi \Leftrightarrow \mathrm{dom}_\psi \wedge \psi$. The first can be derived from the second in a similar fashion. The law of contrapositive of binary logic says that $\varphi \Rightarrow \psi$ if and only if $\neg\psi \Rightarrow \neg\varphi$. Because both sides of the second are Boolean, this law can be applied to it. Together with De Morgan's law it yields the third. The second can be similarly derived from the third.

The above result presents a variant of the law of contrapositive that holds in school logic, but requires explicitly writing the domains. We now derive a result that often allows dropping the domains. Assume that $\varphi \Rightarrow \psi$ and $\mathrm{dom}_\psi \Rightarrow \mathrm{dom}_\varphi$. We have already proven $\neg\psi \Rightarrow \mathrm{dom}_\psi$, $\neg\psi \Rightarrow \neg\mathrm{dom}_\varphi \vee \neg\varphi$, and $\neg\varphi \Rightarrow \mathrm{dom}_\varphi$. These yield $\neg\psi \Rightarrow \mathrm{dom}_\varphi \wedge (\neg\mathrm{dom}_\varphi \vee \neg\varphi) \Leftrightarrow \mathrm{dom}_\varphi \wedge \neg\varphi \Leftrightarrow \neg\varphi$. We got the following.

$$\text{If } \varphi \Rightarrow \psi \text{ and } \mathrm{dom}_\psi \Rightarrow \mathrm{dom}_\varphi, \text{ then } \neg\psi \Rightarrow \neg\varphi.$$

This means that when φ and ψ have the same domain, the binary logic law of contrapositive can be used as such.

In traditional logic, universal quantification can be removed by replacing any expression for the instances of the quantified variable (under certain rules that protect against name

clashes). That is, if f is an expression that obeys the rules, then $(\forall x : \varphi(x)) \Rightarrow \varphi(f)$. In school logic, this is replaced by $(\forall x : \varphi(x)) \Rightarrow \mathrm{dom}_f \rightarrow \varphi(f)$, which is equivalent to $(\forall x : \varphi(x)) \Rightarrow \neg\mathrm{dom}_f \vee \varphi(f)$. For instance, if $f = \sqrt{x}$, the law $\forall y : y - y = 0$ yields $x \geq 0 \rightarrow \sqrt{x} - \sqrt{x} = 0$ and thus does not yield the incorrect result $\sqrt{-1} - \sqrt{-1} = 0$. The law $\forall y : y \neq 0 \rightarrow \frac{y}{y} = 1$ yields $x \geq 0 \rightarrow (\sqrt{x} \neq 0 \rightarrow \frac{\sqrt{x}}{\sqrt{x}} = 1)$, that is, $x > 0 \rightarrow \frac{\sqrt{x}}{\sqrt{x}} = 1$.

This means that the axioms of real numbers and many definitions (such as the definition of "$\leq$" in terms of "$<$" and "$=$") retain their traditional form, but their use in reasoning is changed. Going through all the cases is beyond the scope of this study, but let us investigate the case of applying an equational law. Before presenting the theorems, let us discuss an example.

Assume that when solving an equation on x, we have concluded that $\frac{\sqrt{x}}{\sqrt{x}} = 1 \wedge x = -3$. The law $\frac{y}{y} = 1$ holds for every non-zero real number y, that is, $y \neq 0 \rightarrow \frac{y}{y} = 1$ yields T for every real number y. Nevertheless, it would be incorrect to reason $\frac{\sqrt{x}}{\sqrt{x}} = 1 \wedge x = -3 \Leftrightarrow 1 = 1 \wedge x = -3 \Leftrightarrow x = -3$. We will see that using "$\Rightarrow$" in the place of the first "$\Leftrightarrow$" would be correct. Also $\frac{\sqrt{x}}{\sqrt{x}} = 1 \wedge x = -3 \Leftrightarrow x \geq 0 \wedge \sqrt{x} \neq 0 \wedge 1 = 1 \wedge x = -3$ is correct, where $x \geq 0$ expresses the domain of $\sqrt{x}$ and $\sqrt{x} \neq 0$ expresses the domain of the division.

The theorem considers the application of a law of the form $\xi(y) \rightarrow f(y) = g(y)$, where $f(y)$ and $g(y)$ are expressions and $\xi(y)$ is a predicate on variable y. Often $\xi(y)$ is just T. The law can be written also as $\neg\xi(y) \vee f(y) = g(y)$. In the example above, $f(y)$ is $\frac{y}{y}$, $g(y)$ is 1, and $\xi(y)$ is $y \neq 0$. The law is applied to a predicate of the form $\varphi(f(h))$, where h is an expression. In the example, h is $\sqrt{x}$, $f(h)$ is $\frac{\sqrt{x}}{\sqrt{x}}$, and $\varphi(y)$ is $y = 1 \wedge x = -3$.

Please notice that the variable y occurs in $y - y$ but not in 0, although both express the same function (the constant function $0 : \mathbb{R} \mapsto \mathbb{R}$).

Theorem 4. *Let y be a variable on $\mathbb{R} \cup \{\bot\}$, $f(y)$, $g(y)$, and $h(y)$ be real-valued (potentially undefined) expressions such that y occurs in $f(y)$, and $\xi(y)$ and $\varphi(y)$ be predicates. If $\xi(y) \rightarrow f(y) = g(y)$, $\varphi(\bot)$ yields F or U, and $\mathrm{dom}_{f(y)} \rightarrow \xi(y)$, then*

$$\varphi(f(h)) \quad \Leftrightarrow \quad \mathrm{dom}_{f(h)} \wedge \varphi(g(h)) \quad \Leftrightarrow \quad (\mathrm{dom}_{f(h)} \vee \neg\mathrm{dom}_{g(h)}) \wedge \varphi(g(h)) \quad .$$

Proof. If $\mathrm{dom}_{f(h)}$ yields F, then $f(h)$ yields $\bot$, $\varphi(f(h))$ yields F or U, the middle formula yields F, and the right hand side reduces to $\neg\mathrm{dom}_{g(h)} \wedge \varphi(g(h))$. The latter is F if $g(h)$ is defined and F or U otherwise. The equivalences hold, because none of the three yields T.

Otherwise $\mathrm{dom}_{f(h)}$ yields T, implying $\xi(h)$. Because $f(h)$ is defined and y occurs in $f(y)$, also h is defined, that is, dom_h yields T. The law $\xi(y) \rightarrow f(y) = g(y)$ yields $\mathrm{dom}_h \rightarrow (\xi(h) \rightarrow f(h) = g(h))$ which yields $f(h) = g(h)$. So $\mathrm{dom}_{f(h)}$ yields T and $\varphi(f(h))$ yields the same as $\varphi(g(h))$, giving the claim. □

Thanks to regularity (please see Theorem 2), the only way in which $\varphi(y)$ can fail the assumption in Theorem 4 is to yield T for every y. In that case it is legal and easier to replace T for $\varphi(f(h))$ instead of $g(h)$ for $f(h)$. The condition $\mathrm{dom}_{f(y)} \rightarrow \xi(y)$ tends to hold, because often the task of $\xi(y)$ is only to filter out the situations where $f(y)$ is undefined, so $\xi(y)$ is $\mathrm{dom}_{f(y)}$. This is the case with $y \neq 0 \rightarrow \frac{y}{y} = 1$ and $y \geq 0 \rightarrow (\sqrt{y})^2 = y$, for instance. This means that Theorem 4 is widely applicable.

If $f(y)$ and $g(y)$ have the same domain, then Theorem 4 lets to reason simply $\varphi(f(h)) \Leftrightarrow \varphi(g(h))$, analogously to Section 3. Otherwise it suffices to also require $\mathrm{dom}_{f(h)}$ (or any predicate that is between $\mathrm{dom}_{f(h)}$ and $\mathrm{dom}_{f(h)} \vee \neg\mathrm{dom}_{g(h)}$) on the right hand side. In our example, the theorem yields $\frac{\sqrt{x}}{\sqrt{x}} = 1 \wedge x = -3 \Leftrightarrow x \geq 0 \wedge \sqrt{x} \neq 0 \wedge 1 = 1 \wedge x = -3$.

Another option is to simply ignore the issue of undefined expressions while solving the equation, and finally check the obtained tentative roots. Often Theorem 4 gives the

permission to reason $\varphi(f(h)) \Rightarrow \varphi(g(h))$. In our example, it yields $1 = 1 \wedge x = -3$. Then -3 is a tentative root that fails the check.

The importance of the assumption on $\varphi(y)$ in Theorem 4 is seen by letting $f(y)$ be $y - y$, $g(y)$ be 0, h be $\frac{1}{0}$, $\xi(y)$ be T, and $\varphi(y)$ yield T when y is undefined and F otherwise. Then $\varphi(f(h))$ yields T but $\varphi(g(h))$ yields F. Without the requirement that y occurs in $f(y)$, the choice $f(y)$ is 0, $g(y)$ is $y - y$, $h(y)$ is $\frac{1}{0}$, $\xi(y)$ is T, and $\varphi(y)$ is $y = 0$ would yield $0 = 0 \Leftrightarrow \frac{1}{0} - \frac{1}{0} = 0$.

The next theorem gives more instances where "$\Rightarrow$" can be used.

Theorem 5. *Let y be a variable on $\mathbb{R} \cup \{\bot\}$, $f(y)$, $g(y)$, and $h(y)$ be real-valued (potentially undefined) expressions, and $\xi(y)$ and $\varphi(y)$ be predicates. Assume that $\xi(y) \to f(y) = g(y)$. Then*

$$\varphi(f(h)) \quad \Rightarrow \quad \varphi(g(h)) \vee \neg\xi(h) \vee \neg\mathrm{dom}_h \tag{1}$$

$$\mathrm{dom}_h \wedge (\xi(h) \vee \neg\mathrm{dom}_{\xi(h)}) \wedge \varphi(f(h)) \quad \Rightarrow \quad \varphi(g(h)) \tag{2}$$

Assume, furthermore, that y occurs in $f(y)$. If y occurs in $g(y)$ or $\varphi(y)$ is regular, then

$$\varphi(f(h)) \quad \Rightarrow \quad \varphi(g(h)) \vee \neg\xi(h) \tag{3}$$

$$(\xi(h) \vee \neg\mathrm{dom}_{\xi(h)}) \wedge \varphi(f(h)) \quad \Rightarrow \quad \varphi(g(h)) \tag{4}$$

Proof. If $\varphi(f(h))$ yields F or U, then the claims clearly hold. If $\xi(h)$ yields F, then $\neg\xi(h)$ yields T and $\neg\mathrm{dom}_{\xi(h)}$ yields F, thus the claims hold again. So from now on, let $\varphi(f(h))$ yield T and $\xi(h)$ yield U or T.

If h is defined, then the law $\neg\xi(y) \vee f(y) = g(y)$ can be applied with $y = h$. Because $\neg\xi(h)$ yields F or U and the law yields T, we have $f(h) = g(h)$ (which entails that $f(h)$ and $g(h)$ are both defined). So $\varphi(g(h))$ yields the same as $\varphi(f(h))$, that is, T, making the claims hold.

In the remaining case, $\varphi(f(h))$ yields T, $\xi(h)$ yields U or T, and h is undefined, that is, dom_h yields F. In (1) and (2), the claim follows immediately from the value of dom_h. The cases (3) and (4) remain.

If y occurs both in $f(y)$ and $g(y)$, then $f(h)$ and $g(h)$ are undefined because h is undefined. Therefore, $\varphi(g(h))$ yields the same as $\varphi(f(h))$, that is, T. So the claims hold.

If y occurs in $f(y)$ and $\varphi(y)$ is regular, then $f(h)$ is undefined and $\varphi(f(h))$ yields either U or the same as $\varphi(g(h))$. In both cases the claim is immediately obtained. □

In (3) and (4), the importance of the regularity assumption and the assumption that y occurs in $f(y)$, are seen by the counter-examples that were used after Theorem 4 for a similar purpose.

8 Conclusions

The roots to $x + \sqrt{x} = 2\sqrt{x}$ are $x = 0$ and $x = 1$. We explicated a logic where this can be expressed as $x + \sqrt{x} = 2\sqrt{x} \Leftrightarrow x = 0 \vee x = 1$. When $x = -1$, then $x = 0 \vee x = 1$ is F, but $x + \sqrt{x} = 2\sqrt{x}$ is a comparison of two undefined expressions. To make $x + \sqrt{x} = 2\sqrt{x} \Leftrightarrow x = 0 \vee x = 1$ valid also when $x = -1$, we adopted the principle that at the level of "$\Rightarrow$" and "$\Leftrightarrow$", undefined claims are equivalent to false claims.

This principle cannot be implemented by making each predicate yield F on an undefined argument, because if $\varphi(x)$ is such a predicate, then $\neg\varphi(x)$ is not although it is a predicate. This problem was solved by employing a third truth value U similarly to Kleene's K_3^S. It is its own negation, and it is equated to F only at the level of "$\Rightarrow$" and "$\Leftrightarrow$".

An aspect of this approach is that negation, conjunction, and so on cannot be applied to "$\Rightarrow$" and "$\Leftrightarrow$". Unlike "$\neg$", "$\wedge$", and so on, they are not treated as operators that yield a ternary truth value but as binary relations on ternary truth values. Their result could be thought of as a traditional binary truth value, but to avoid confusion, we did not use that parlance and instead said that $\varphi \Rightarrow \psi$ is or is not a valid reasoning step. There are related ternary truth value -yielding operators "$\rightarrow$" and "$\leftrightarrow$", and $\varphi \Rightarrow \psi$ is valid if and only if $(\mathrm{dom}_\varphi \wedge \varphi) \rightarrow \psi$ yields T for every instantiation that is allowed by the context, where dom_φ yields F when φ yields U and T otherwise. Similarly to $f < g \leq h$ in arithmetic, sequences such as $\varphi \Rightarrow \psi \Leftrightarrow \xi$ are interpreted as saying that both $\varphi \Rightarrow \psi$ and $\psi \Leftrightarrow \xi$ are valid. This is unlike $\varphi \rightarrow \psi \leftrightarrow \xi$, which is interpreted similarly to $f - g + h$ as $(\varphi \rightarrow \psi) \leftrightarrow \xi$.

Although developing a full-fledged reasoning system for school logic was beyond the scope of this study, we did analyse the application of (perhaps guarded) equational laws $f(y) = g(y)$. Among other things, it turned out that if $f(y)$ and $g(y)$ are undefined for the same arguments (which is often the case), then the law can be applied similarly to traditional logic. This result depends on the fact that not all functions on ternary truth values can be constructed only using "$\neg$", "$\wedge$", and so on. In particular, no $\varphi(x)$ can be constructed such that it yields T if and only if x is U.

We believe that this logic is what mathematicians implicitly use when reasoning in real number arithmetic.

Acknowledgements. The authors thank Kerkko Luosto for helpful discussions on the topic and the reviewers of RuFiDiM IV for useful comments.

References

[1] Bertram Fronhöfer. Introduction to many-valued logics. https://web.archive.org/web/20131225052706/ http://www.wv.inf.tu-dresden.de/Teaching/SS-2011/mvl/mval.HANDOUT2.pdf, 2011.

[2] ISO. *ISO/IEC/IEEE 60559:2011 Information technology — Microprocessor Systems — Floating-Point arithmetic.* International Organization for Standardization, Geneva, Switzerland, 2011.

[3] Clifford B. Jones. *Systematic software development using VDM (2. ed.).* Prentice Hall International Series in Computer Science. Prentice Hall, 1991.

[4] Stephen C. Kleene. *Introduction to metamathematics.* Bibliotheca mathematica. North-Holland Pub. Co., 1964.

[5] Jan Łukasiewicz. Philosophische Bemerkungen zu mehrwertigen Systemen des Aussagenkalküls. *Comptes rendus des séances de la Societé des Sciences et des Lettres de Varsovie*, 23(Cl. III):51–77, 1930.

[6] J. Michael Spivey. *Z Notation - a reference manual (2. ed.).* Prentice Hall International Series in Computer Science. Prentice Hall, 1992.

[7] Antti Valmari and Terhi Kaarakka. MathCheck: A tool for checking math solutions in detail. In *44th SEFI Conference, Engineering Education on Top of the World: Industry University Cooperation, 12-15 September 2016, Tampere, Finland.* European Society for Engineering Education SEFI, 2016.

Turku Centre for Computer Science
TUCS Lecture Notes

1. **Ralph-Johan Back och Joakim von Wright**, Matematik med lite logik: Strukturerade härledningar I gymnasiematematiken
2. **Ralph-Johan Back och Joakim von Wright**, Matematik med lite logik: En kort kurs i talteori
3. **Ralph-Johan Back och Joakim von Wright**, Matematik med lite logik: Studentexamen i lång matematik, våren 2003
4. **Ralph-Johan Back ja Joakim von Wright**, Matematiikkaa logiikan avulla: Rakenteiset päättelyketjut lukiomatematiikassa
5. **Ralph-Johan Back ja Joakim von Wright**, Matematiikkaa logiikan avulla: Lyhyt lukuteorian kurssi
6. **Ralph-Johan Back ja Joakim von Wright**, Matematiikkaa logiikan avulla: Pitkän matematiikan ylioppilaskoe, kevät 2003
7. **Ralph-Johan Back och Joakim von Wright**, Matematik med lite logik: Introduktion till strukturerade härledningar
8. **Ralph-Johan Back och Joakim von Wright**, Matematik med lite logik: Logik för strukturerade härledningar
9. **Ralph-Johan Back och Joakim von Wright**, Matematik med lite logik: Strukturerade härledningar som allmänt bevisformat
10. **Ralph-Johan Back ja Joakim von Wright**, Matematiikkaa logiikan avulla: Johdatus rakenteisiin päättelyketjuihin
11. **Ralph-Johan Back ja Joakim von Wright**, Matematiikkaa logiikan avulla: Logiikka ja rakenteiset päättelyketjut
12. **Ralph-Johan Back ja Joakim von Wright**, Matematiikkaa logiikan avulla: Rakenteiset päättelyketjut yleisenä todistusmuotona
13. **Jarkko Kari (Editor)**, Proceedings of JAC 2010 – journées Automates Cellulaires
14. **Mike Stannet, Danuta Makowiec, Anna T. Lawniczak and Bruno N. Di Stefano**, Proceedings of the Satellite Workshops of UC 2011
15. **Timo Leino (Editor)**, Proceedings of the IRIS 2011 Conference
16. **Hongxiu Li (Editor)**, Studies on Inequalities in Information Society – Proceedings of the Conference, Well-Being in the Information Society. WIS 2012
17. **Vesa Halava, Juhani Karhumäki and Yuri Matiyasevich (Editors)**, RuFiDiM II, Proceedings of the Second Russian Finnish Symposium on Discrete Mathematics 2012
18. **Michael Butler, Stefan Hallerstede and Marina Waldén (Editors)**, Proceedings of the 4th Rodin User and Development Workshop
19. **Hongxiu Li and Jonna Järveläinen (Editors)**, Effective, Agile and Trusted eServices Co-Creation – Proceedings of the 15th International Conference on Electronic Commerce ICEC 2013
20. **Juhani Karhumäki, Markus Whiteland and Luca Zamboni (Editors)**, Local Proceedings of WORDS 2013
21. **Juha-Pekka Soininen, Sergey Balandin, Johan Lilius, Petri Liuha and Tullio Salmon Cinotti (Editors)**, Proceedings of the Open International M3 Semantic Interoperability Workshop
22. **Lutz M. Wegner**, Sorting – The Turku Lectures
23. **Kai K. Kimppa, Diane Whitehouse, Tiina Kuusela and Jackie Phahlamohlaka (Editors)**, Human Choice and Computers – HCC11, Work-in-Progress Proceedings
24. **Jarkko Kari, Ilkka Törmä and Michal Szabados (Eds.)**, 21st International Workshop on Cellular Automata and Discrete Complex Systems – Exploratory Papers of AUTOMATA 2015
25. **Juhani Karhumäki and Aleksi Saarela (Editors)**, Proceedings of the Finnish Mathematical Days 2016
26. **Juhani Karhumäki, Aleksi Saarela and Yuri Matiyasevichin (Eds.)**, Proceedings of the Fourth Russian Finnish Symposium on Discrete Mathematics

Turku
Centre *for*
Computer
Science

http://www. tucs.fi

tucs@abo.fi

University of Turku
Faculty of Mathematics and Natural Sciences
- Department of Information Technology
- Department of Mathematics and Statistics

Turku School of Economics
- Institute of Information Systems Science

Åbo Akademi University
Faculty of Science and Engineering
- Computer Engineering
- Computer Science

Faculty of Social Sciences, Business and Economics
- Information Systems

ISBN 978-952-12-3547-4
ISSN 1797-8823

Karhumäki, Matiyasevich & Saarela (Eds.)

Proceedings of the Fourth Russian Finnish Symposium on Discrete Mathematics