IS 2019

Zbornik 22. mednarodne multikonference

INFORMACIJSKA DRUŽBA

Zvezek I

Proceedings of the 22nd International Multiconference

INFORMATION SOCIETY

Volume I

Srednje-ervropska konferenca o uporabnem teoretičnem računalništvu

Middle-European Conference on Applied Theoretical Computer Science

Uredili / Edited by

Andrej Brodnik, Gábor Galambos, Branko Kavšek

http://is.ijs.si

10.–11. oktober 2019 / 10–11 October 2019

Ljubljana, Slovenia

Zbornik 22. mednarodne multikonference

INFORMACIJSKA DRUŽBA – IS 2019

Zvezek I

Proceedings of the 22nd International Multiconference

INFORMATION SOCIETY – IS 2019

Volume I

Srednje-ervropska konferenca o uporabnem teoretičnem računalništvu
Middle-European Conference on Applied Theoretical Computer Science

Uredili / Edited by

Andrej Brodnik, Gábor Galambos, Branko Kavšek

http://is.ijs.si

10.–11. oktober 2019 / 10–11 October 2019
Koper, Slovenia

Uredniki:

Andrej Brodnik
Univerza na Primorskem in Univerza v Ljubljani

Gábor Galambos
Univerza v Szegedu

Branko Kavšek
Univerza na Primorskem

Založnik: Institut »Jožef Stefan«, Ljubljana
Priprava zbornika: Mitja Lasič, Vesna Lasič, Lana Zemljak
Oblikovanje naslovnice: Vesna Lasič
Na naslovnici je uporabljena slika robota podjetja

Dostop do e-publikacije:
http://library.ijs.si/Stacks/Proceedings/InformationSociety

Ljubljana, oktober 2019

Informacijska družba
ISSN 2630-371X

```
Kataložni zapis o publikaciji (CIP) pripravili v Narodni in univerzitetni
knjižnici v Ljubljani
COBISS.SI-ID=302478080
ISBN 978-961-264-171-9 (epub)
ISBN 978-961-264-172-6 (pdf)
```

PREDGOVOR MULTIKONFERENCI INFORMACIJSKA DRUŽBA 2019

Multikonferenca Informaci družba (http://is.ijs.si) je z dvaindvajseto zaporedno prireditvijo tradicionalni osrednji srednjeevropski dogodek na področju informacijske družbe, računalništva in informatike. Informacijska družba, znanje in umetna inteligenca so - in to čedalje bolj – nosilci razvoja človeške civilizacije. Se bo neverjetna rast nadaljevala in nas ponesla v novo civilizacijsko obdobje? Bosta IKT in zlasti umetna inteligenca omogočila nadaljnji razcvet civilizacije ali pa bodo demografske, družbene, medčloveške in okoljske težave povzročile zadušitev rasti? Čedalje več pokazateljev kaže v oba ekstrema – da prehajamo v naslednje civilizacijsko obdobje, hkrati pa so notranji in zunanji konflikti sodobne družbe čedalje težje obvladljivi.

Letos smo v multikonferenco povezali 12 odličnih neodvisnih konferenc. Zajema okoli 300 predstavitev, povzetkov in referatov v okviru samostojnih konferenc in delavnic in 500 obiskovalcev. Prireditev bodo spremljale okrogle mize in razprave ter posebni dogodki, kot je svečana podelitev nagrad. Izbrani prispevki bodo izšli tudi v posebni številki revije Informatica (http://www.informatica.si/), ki se ponaša z 42-letno tradicijo odlične znanstvene revije.

Multikonferenco Informacijska družba 2019 sestavljajo naslednje samostojne konference:

- 6. študentska računalniška konferenca
- Etika in stroka
- Interakcija človek računalnik v informacijski družbi
- Izkopavanje znanja in podatkovna skladišča
- Kognitivna znanost
- Kognitonika
- Ljudje in okolje
- Mednarodna konferenca o prenosu tehnologij
- Robotika
- Slovenska konferenca o umetni inteligenci
- Srednje-evropska konferenca o uporabnih in teoretičnih računalniških znanostih
- Vzgoja in izobraževanje v informacijski družbi

Soorganizatorji in podporniki konference so različne raziskovalne institucije in združenja, med njimi tudi ACM Slovenija, SLAIS, DKZ in druga slovenska nacionalna akademija, Inženirska akademija Slovenije (IAS). V imenu organizatorjev konference se zahvaljujemo združenjem in institucijam, še posebej pa udeležencem za njihove dragocene prispevke in priložnost, da z nami delijo svoje izkušnje o informacijski družbi. Zahvaljujemo se tudi recenzentom za njihovo pomoč pri recenziranju.

V 2019 bomo sedmič podelili nagrado za življenjske dosežke v čast Donalda Michieja in Alana Turinga. Nagrado Michie-Turing za izjemen življenjski prispevek k razvoju in promociji informacijske družbe je prejel prof. dr. Marjan Mernik. Priznanje za dosežek leta pripada sodelavcem Odseka za inteligentne sisteme Instituta »Jožef Stefan«. Podeljujemo tudi nagradi »informacijska limona« in »informacijska jagoda« za najbolj (ne)uspešne poteze v zvezi z informacijsko družbo. Limono je dobil sistem »E-zdravje«, jagodo pa mobilna aplikacija »Veš, kaj ješ?!«. Čestitke nagrajencem!

Mojca Ciglarič, predsednica programskega odbora
Matjaž Gams, predsednik organizacijskega odbora

FOREWORD - INFORMATION SOCIETY 2019

The Information Society Multiconference (http://is.ijs.si) is the traditional Central European event in the field of information society, computer science and informatics for the twenty-second consecutive year. Information society, knowledge and artificial intelligence are - and increasingly so - the central pillars of human civilization. Will the incredible growth continue and take us into a new civilization period? Will ICT, and in particular artificial intelligence, allow civilization to flourish or will demographic, social, and environmental problems stifle growth? More and more indicators point to both extremes - that we are moving into the next civilization period, and at the same time the internal and external conflicts of modern society are becoming increasingly difficult to manage.

The Multiconference is running parallel sessions with 300 presentations of scientific papers at twelve conferences, many round tables, workshops and award ceremonies, and 500 attendees. Selected papers will be published in the Informatica journal with its 42-years tradition of excellent research publishing.

The Information Society 2019 Multiconference consists of the following conferences:

- 6. Student Computer Science Research Conference
- Professional Ethics
- Human – Computer Interaction in Information Society
- Data Mining and Data Warehouses
- Cognitive Science
- International Conference on Cognitonics
- People and Environment
- International Conference of Transfer of Technologies – ITTC
- Robotics
- Slovenian Conference on Artificial Intelligence
- Middle-European Conference on Applied Theoretical Computer Science
- Education in Information Society

The Multiconference is co-organized and supported by several major research institutions and societies, among them ACM Slovenia, i.e. the Slovenian chapter of the ACM, SLAIS, DKZ and the second national engineering academy, the Slovenian Engineering Academy. In the name of the conference organizers, we thank all the societies and institutions, and particularly all the participants for their valuable contribution and their interest in this event, and the reviewers for their thorough reviews.

For the fifteenth year, the award for life-long outstanding contributions will be presented in memory of Donald Michie and Alan Turing. The Michie-Turing award was given to Prof. Marjan Mernik for his life-long outstanding contribution to the development and promotion of information society in our country. In addition, a recognition for current achievements was awarded to members of Department of Intelligent Systems of Jožef Stefan Institute. The information lemon goes to the “E-Health” system, and the information strawberry to the mobile application “Veš, kaj ješ?!” (Do you know what you eat?!). Congratulations!

Mojca Ciglarič, Programme Committee Chair
Matjaž Gams, Organizing Committee Chair

KONFERENČNI ODBORI
CONFERENCE COMMITTEES

International Programme Committee

Vladimir Bajic, Južna Afrika
Heiner Benking, Nemčija
Se Woo Cheon, Južna Koreja
Howie Firth, Škotska
Olga Fomichova, Rusija
Vladimir Fomichov, Rusija
Vesna Hljuz Dobric, Hrvaška
Alfred Inselberg, Izrael
Jay Liebowitz, ZDA
Huan Liu, Singapur
Henz Martin, Nemčija
Marcin Paprzycki, ZDA
Claude Sammut, Avstralija
Jiri Wiedermann, Češka
Xindong Wu, ZDA
Yiming Ye, ZDA
Ning Zhong, ZDA
Wray Buntine, Avstralija
Bezalel Gavish, ZDA
Gal A. Kaminka, Izrael
Mike Bain, Avstralija
Michela Milano, Italija
Derong Liu, Chicago, ZDA
Toby Walsh, Avstralija

Organizing Committee

Matjaž Gams, chair
Mitja Luštrek
Lana Zemljak
Vesna Koricki
Marjetka Šprah
Mitja Lasič
Blaž Mahnič
Jani Bizjak
Tine Kolenik

Programme Committee

Mojca Ciglarič, chair
Bojan Orel, co-chair
Franc Solina
Viljan Mahnič
Cene Bavec
Tomaž Kalin
Jozsef Györkös
Tadej Bajd
Jaroslav Berce
Mojca Bernik
Marko Bohanec
Ivan Bratko
Andrej Brodnik
Dušan Caf
Saša Divjak
Tomaž Erjavec
Bogdan Filipič
Andrej Gams
Matjaž Gams
Mitja Luštrek
Marko Grobelnik
Vladislav Rajkovič
Grega Repovš
Nikola Guid
Marjan Heričko
Borka Jerman Blažič Džonova
Gorazd Kandus
Urban Kordeš
Marjan Krisper
Andrej Kuščer
Jadran Lenarčič
Borut Likar
Janez Malačič
Olga Markič
Dunja Mladenič
Franc Novak
Ivan Rozman
Niko Schlamberger
Stanko Strmčnik
Jurij Šilc
Jurij Tasič
Denis Trček
Andrej Ule
Tanja Urbančič
Boštjan Vilfan
Baldomir Zajc
Blaž Zupan
Boris Žemva
Leon Žlajpah

KAZALO / TABLE OF CONTENTS

Zbornik 22. mednarodne multikonference

INFORMACIJSKA DRUŽBA – IS 2019

Zvezek I

Proceedings of the 22nd International Multiconference

INFORMATION SOCIETY – IS 2019

Volume I

Srednje-ervropska konferenca o uporabnem teoretičnem računalništvu
Middle-European Conference on Applied Theoretical Computer Science

Uredili / Edited by

Andrej Brodnik, Gábor Galambos, Branko Kavšek

https://matcos.iam.upr.si/

10.–11. oktober 2019 / 10–11 October 2019
Koper, Slovenia

PREDGOVOR

Spet so minila tri leta in pred nami je nova konferenca MATCOS (Middle European Conference on Applied Theoretical Computer Science). Če se odločite, da boste prinesli konferenco v novo mesto in upate, da bo vaš napor obrodil sadove, poskušate pridobiti čim boljše strokovnjake v programski odbor. Da jih pridobite, pregledate vse svoje povezave – vse samo zato, da bi bila konferenca čim boljša. Ko konferenco organizirate v drugo, že bolje razumete težave, ki vas čakajo in se jim poskušate izogniti. Na koncu, če je konferenca uspešna, pričnete verjeti, da bo prav tako uspešna tudi naslednja. Tako lahko tretji MATCOS smatramo kot začetek nove tradicije. Poleg tega smo v paleto konference poskusili dodati nove »barve«.

V programskem in organizacijskem odboru so ostali vsi aktivni člani, vrata pa smo odprli nekaterim novim članom. Sicer je področje konference ostalo nespremenjeno, vendar so teme posameznih prispevkov širše. Ostala je tudi navezava na študentsko konferenco StuCosRec (Student Computer Science Research Conference). Novost je bila vključitev kratkih prispevkov poleg rednih. Po recenzijah bomo na konferenci tako poslušali 21 rednih in 16 kratkih prispevkov.

Thomas Pock z Unverze za tehnologijo v Gradcu je »zagotovilo«, da se bo kakovost vabljenih predavanj nadaljevala. Žarišče njegovega raziskovanja predstavlja po eni strani razvoj matematičnih modelov računalniškega vida in obdelave slik ter po drugi strani razvoj učinkovitih konveksnih ne-gladkih optimizacijskih algoritmov. Upamo, da bo predavanje uporabno tako za strokovnjake kot za širšo publiko.

Člani programskega in organizacijskega so v zadnjih nekaj mesecih pred konferenco opravili veliko delo. Zato vsem, ki ste pomagali pripraviti in izvesti »tradicionalni« tretji MATCOS en velik hvala.

Upamo, da boste ta dneva v Kopru resnično uživali ter vzpostavili nove strokovne stike med konferenco MATCOS-19.

V imenu organizatorjev
Andrej Brodnik and Gábor Galambos
sopredsedujoča

FOREWORD

Three years is over again, and we organise the new MATCOS (Middle European Conference on Applied Theoretical Computer Science) conference. If you decide to bring a conference in a new city you hope that your efforts will be successful, therefore you try to bring together good people in the PC and you leave no stone unturned within your connection to make the first conference on the best level. Arriving to the second instance you start to understand the difficulties of a conference organisation, but if this second attempt becomes more successful then you begin to trust to the next one. We can consider the third MATCOS as the beginning of a new tradition, so during the organisation process we tried to bring new "colours" to the palette of the conference.

As you can see the active members in the Organising Committee and the Program Committee remained, and we opened the door for new members. The scope is the same as earlier but the subject of the accepted papers are wider. We kept the StuCosRec (Student Computer Science Research Conference) adjoint. This time we accepted besides the regular papers also short papers giving place to more new subjects. After the review process we accepted 21 regular talks and 16 short talks to be presented.

Thomas Pock from the Graz University of Technology is the "assurance" that the high level invited talks will be continued this year too. The focus of his research is the development of mathematical models for computer vision and image processing as well as the development of efficient convex and non-smooth optimization algorithms. We hope that his talk will be useful for the expert and enquirers, equally.

The members of PC and OC did an excellent job during the last few months. Thanks to everybody who helped to organised this "traditional" 3rd MATCOS conference.

We hope you will enjoy these two days in Koper and you can establish new professional contacts during the MATCOS-19 conference.

On behalf of the organisers
Andrej Brodnik and Gábor Galambos
co-chairs

PROGRAMSKI ODBOR / PROGRAMME COMMITTEE

On the notion of duals of certain AB functions

Amar Bapić
UP FAMNIT
amar.bapic@famnit.upr.si

Enes Pasalic
UP FAMNIT
enes.pasalic@upr.si

Samir Hodžić
UP FAMNIT
samir.hodzic@famnit.upr.si

ABSTRACT

In this paper we employ two different notions of duals of certain classes of Boolean functions which are used for the purpose of deriving other interesting combinatorial objects from suitable mappings from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$. A class of particular interest in this context is almost bent (AB) functions having the property that their (Walsh) spectral characterization possess a desired structure. We give a general result regarding the Gold AB functions, state one conjecture regarding the Welch AB functions and some computational results for the Kasami AB functions. Applying another definition of dual, introduced by Hodžić *et al.* [7] we provide computational evidence that the duals of Gold AB functions may build a space of bent functions (vectorial bent) though a more rigour theoretical analysis is needed.

Keywords

Vectorial bent functions, AB functions, Dual functions, Walsh spectrum

1. INTRODUCTION

Mappings from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ are called vectorial Boolean or (n,m)-functions. Any such function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ can be represented in the form

$$F(\mathbf{x}) = (f_0(\mathbf{x}), f_1(\mathbf{x}), \ldots, f_{m-1}(\mathbf{x}))\,,\ \mathbf{x} \in \mathbb{F}_2^n$$

where $f_i : \mathbb{F}_2^n \to \mathbb{F}_2$, $i = 0, \ldots, m-1$, are called *coordinate (Boolean) functions* of F and the 2^m-1 non-zero linear combinations of its coordinates are termed as *component functions*. When n is odd, (n,n)-functions that offer optimal resistance against both linear and differential cryptanalysis [6, 9] are called *almost bent* (AB) functions. There are a few known infinite families of these functions though their complete classification seems to be elusive. Another combinatorial objects of particular importance in cryptography, coding and design theory, is a class of vectorial bent functions having the property that all the component functions are *bent* which are characterized by a unique feature of having (uniform) flat Walsh spectrum. Nevertheless, it was shown by Nyberg [10] that vectorial bent functions $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ may only exist for even n and then necessarily $m \leq n/2$. Even though there is an extensive research on both these classes of functions so far there has been no explicit connection between them. The main purpose of this article is to establish some (partial) connections and indicate the possibility of relating these structures through so called duals of Boolean functions. The concept of dual was originally defined for bent functions but it can be generalized to so-called plateaued Boolean functions. Employing two different definitions of duals, we will provide some theoretical results that indicate certain regularity in the dual space of AB functions. Furthermore, we provide computational evidence that a different definition of a dual function, introduced originally in [7], can identify vectorial bent functions in the dual space of Gold AB functions. Even though we do not completely understand the mechanisms behind this phenomena this connection is of great interest.

1.1 Definitions and terminology

The vector space $\mathbb{F}_2^n$ is the space of all n-tuples $\mathbf{x} = (x_0, \ldots, x_{n-1})$, where $x_i \in \mathbb{F}_2$. For $\mathbf{x} = (x_0, \ldots, x_{n-1})$ and $\mathbf{y} = (y_0, \ldots, y_{n-1})$ in $\mathbb{F}_2^n$, the usual dot product over $\mathbb{F}_2$ is defined as $\mathbf{x} \cdot \mathbf{y} = x_0y_0 \oplus \cdots \oplus x_{n-1}y_{n-1}$. The *weight* $wt(\mathbf{x})$ of $\mathbf{x} = (x_0, \ldots, x_{n-1}) \in \mathbb{F}_2^n$ is computed as $wt(\mathbf{x}) = \sum_{i=0}^{n-1} x_i$. By "$\sum$" we denote the integer sum (without modulo evaluation), whereas "$\bigoplus$" denotes the sum evaluated modulo two. The *Walsh-Hadamard transform* (WHT) of $f \in \mathcal{B}_n$, and its inverse WHT, at any point $\mathbf{u} \in \mathbb{F}_2^n$ are defined, respectively, by

$$W_f(\mathbf{u}) = \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus \mathbf{u} \cdot \mathbf{x}}, \tag{1}$$

$$(-1)^{f(\mathbf{x})} = 2^{-n} \sum_{\mathbf{u} \in \mathbb{F}_2^n} W_f(\mathbf{u})(-1)^{\mathbf{u} \cdot \mathbf{x}}. \tag{2}$$

The sequence of the 2^n *Walsh coefficients* given by (1), as $\mathbf{u}$ goes through $\mathbb{F}_2^n$ is called the *Walsh spectrum* of f, denoted by

$$\mathcal{W}_f = (W_f(\mathbf{u}_0), \ldots, W_f(\mathbf{u}_{2^n-1})),$$

where $\mathbf{u}_0, \ldots, \mathbf{u}_{2^n-1} \in \mathbb{F}_2^n$ are ordered lexicographically.
With $\mathcal{B}_n$ we denote the class of all *bent* Boolean functions defined on $\mathbb{F}_2^n$, i.e., all functions whose Walsh spectrum takes the values $\pm 2^{\frac{n}{2}}$. A class of Boolean functions on $\mathbb{F}_2^n$ characterised by the property that their Walsh spectra is three-valued (more precisely taking values in $\{0, \pm 2^{\frac{n+s}{2}}\}$ for a positive integer $s < n$) are called *s-plateaued* functions. [3] In case $s = 1$ ($s = 2$) for n odd (n even), the functions are called *semi-bent.* For a bent Boolean function f defined on $\mathbb{F}_2^n$, its ***dual*** $\tilde{f}$ is defined as a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2$, for which it holds that

$$(-1)^{\tilde{f}(\mathbf{u})} = 2^{-\frac{n}{2}} W_f(\mathbf{u}), \ \ \mathbf{u} \in \mathbb{F}_2^n.$$

A standard way of defining the dual $\tilde{f} : \mathbb{F}_2^n \to \mathbb{F}_2$ of an s-plateaued Boolean function f on $\mathbb{F}_2^n$ is as follows:

$$\tilde{f}(\mathbf{x}) = 2^{-\frac{n+s}{2}} |W_f(\mathbf{x})|, \ \ \mathbf{x} \in \mathbb{F}_2^n.$$

If $F = (f_0, \dots, f_{n-1})$ is an (n,n)-function, we define its dual $\tilde{F} = (\tilde{f}_0, \dots, \tilde{f}_{n-1})$ as an (n,n)-function whose coordinates correspond to the duals of the coordinates of F.

2. DUALS OF AB FUNCTIONS

In this section we analyze several classes of power AB functions with respect to their duals defined in the standard way. Table 1 gives a list of certain exponents d for which the function $F(x) = x^d$, where $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, is an AB function.

Function	d	Conditions	Degree
Gold	$2^i + 1$	$\gcd(i,n) = 1,\ 1 \le i \le \frac{n-1}{2}$	2
Welch	$2^t + 3$	$n = 2t + 1$	3
Kasami	$2^{2i} - 2^i + 1$	$\gcd(i,n) = 1,\ 1 \le i \le \frac{n-1}{2}$	$i+1$

Table 1: Some known AB power functions x^d defined on $\mathbb{F}_{2^n}$ [1]

2.1 Gold case

To clarify our approach we consider as an example the Gold function $F(x) = x^3$ defined on $\mathbb{F}_{2^3}$. By identifying $\mathbb{F}_{2^3}$ with $\mathbb{F}_2^3$, one can consider it as a function over the vector space $\mathbb{F}_2^3$. With $F_i(\mathbf{x}) = \mathbf{v}_i \cdot F(\mathbf{x})$, where $\mathbf{v}_1, \dots, \mathbf{v}_{2^3-1} \in \mathbb{F}_2^3 \setminus \{\mathbf{0}\}$ are ordered lexicographically, we denote the component functions of F. In Table 2, we list the truth tables of the component functions F_i of F as well as their dual functions $\tilde{F}_i$ on $\mathbb{F}_2^3$ and their corresponding Walsh spectra. In [5], it was proved that for any n, the Walsh support of any quadratic function on $\mathbb{F}_2^n$ is a flat on $\mathbb{F}_2^n$ of even dimension. Since all Gold functions are quadratic, the following proposition summarizes these observations and gives a more general result that includes the Gold case.

PROPOSITION 2.1. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an AB function. Suppose that the Walsh supports S_i of the component functions F_i of F are affine subspaces of dimension $n-1$. Then the component functions of the dual $\tilde{F}$ are linear functions defined on $\mathbb{F}_2^n$.*

PROOF. First we consider the Walsh-Haddamard transform of an arbitrary component function of F^*.
Case I: Suppose that $\mathbf{u} \neq \mathbf{0}$.

$$\begin{aligned} W_{\tilde{F}_i}(\mathbf{u}) &= \sum_{\mathbf{x}\in\mathbb{F}_2^n} (-1)^{\tilde{F}_i(\mathbf{x})\oplus \mathbf{u}\cdot\mathbf{x}} = \sum_{\mathbf{x}\in\mathbb{F}_2^n} (-1)^{\tilde{F}_i(\mathbf{x})}(-1)^{\mathbf{u}\cdot\mathbf{x}} \\ &= \sum_{\mathbf{x}\notin S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}} - \sum_{\mathbf{x}\in S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}} = \sum_{\mathbf{x}\notin S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}} \\ &+ \sum_{\mathbf{x}\in S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}} - \sum_{\mathbf{x}\in S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}} - \sum_{\mathbf{x}\in S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}} \\ &= \sum_{\mathbf{x}\in\mathbb{F}_2^n} (-1)^{\mathbf{u}\cdot\mathbf{x}} - 2\sum_{\mathbf{x}\in S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}} = -2\sum_{\mathbf{x}\in S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}}, \end{aligned}$$

Since every AB function is a permutation (see e.g. [1, 4]), then $\mathbf{0} \notin S_i$. Now, if we represent S_i as $S_i = \mathbf{a} + V$, where $\mathbf{a} \notin V$ and V is a linear subspace in $\mathbb{F}_2^n$ of dimension $n-1$, then $S_i^C = V$. Thus, denoting by $G = -2\sum_{\mathbf{x}\in S_i}(-1)^{\mathbf{u}\cdot\mathbf{x}}$ we have:

$$\begin{aligned} G &= 2\sum_{\mathbf{x}\notin S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}} - 2\sum_{\mathbf{x}\notin S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}} - 2\sum_{\mathbf{x}\in S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}} \\ &= 2\sum_{\mathbf{x}\notin S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}} - 2\sum_{\mathbf{x}\in\mathbb{F}_2^n} (-1)^{\mathbf{u}\cdot\mathbf{x}} = 2\sum_{\mathbf{x}\notin S_i} (-1)^{\mathbf{u}\cdot\mathbf{x}} \\ &= 2\sum_{\mathbf{x}\in S_i^C} (-1)^{\mathbf{u}\cdot\mathbf{x}} = 2\sum_{\mathbf{x}\in V} (-1)^{\mathbf{u}\cdot\mathbf{x}} \\ &= \begin{cases} 0, & \mathbf{u}\notin V^\perp \\ 2\cdot 2^{\dim V}, & otherwise \end{cases} = \begin{cases} 0, & \mathbf{u}\notin V^\perp \\ 2^n, & otherwise \end{cases} \end{aligned}$$

where $V^\perp = \{\mathbf{x} \in \mathbb{F}_2^n : \mathbf{x}\cdot\mathbf{v} = 0, \forall \mathbf{v} \in V\}$.
Case II: Suppose that $\mathbf{u} = \mathbf{0}$.

$$W_{\tilde{F}_i}(\mathbf{0}) = \sum_{\mathbf{x}\in\mathbb{F}_2^n} (-1)^{\tilde{F}_i(\mathbf{x})} = \sum_{\mathbf{x}\in S_i^C} 1 - \sum_{\mathbf{x}\in S_i} 1 = |S_i^C| - |S_i| = 0.$$

So, for every $\mathbf{u} \in \mathbb{F}_2^n$ we have

$$W_{\tilde{F}_i}(\mathbf{u}) = \begin{cases} 0, & \mathbf{u}\notin V^\perp \vee \mathbf{u} = \mathbf{0} \\ 2^n, & otherwise \end{cases}$$

Since V is of dimension $n-1$, $V^\perp$ is of dimension 1, i.e., $W_{\tilde{F}_i}$ is non-zero at only one vector. □

REMARK 1. *If the Walsh support of a semi-bent function f is not necessarily a flat on $\mathbb{F}_2^n$, the Walsh coefficients of its dual $\tilde{f}$ on $\mathbb{F}_2^n$ equal*

$$W_{\tilde{f}}(\mathbf{u}) = \begin{cases} 0, & \mathbf{u} = \mathbf{0} \\ -2\sum_{\mathbf{x}\in S_f} (-1)^{\mathbf{u}\cdot\mathbf{x}}, & otherwise \end{cases}$$

2.2 Welch and Kasami case

Let us now consider the Welch power function $F(x) = x^{2^t+3}$ on $\mathbb{F}_2^n$, where $t = \frac{n-1}{2}$, for which the duals of the component functions are given in Table 3. This leads us to the following conjecture.

n	d	Walsh spectra of $\tilde{F}$	Comment
3	5	$\{0, 8\}$	linear
5	7	$\{0, \pm 8\}$	AB
7	11	$\{0, \pm 16\}$	AB
9	19	$\{0, \pm 2^5, \pm 2^6\}$	5-valued Walsh spectra
11	35	$\{0, \pm 2^6, \pm 2^7\}$	5-valued Walsh spectra
13	67	$\{0, \pm 2^7, \pm 2^8\}$	5-valued Walsh spectra
15	131	$\{0, \pm 2^8, \pm 2^9\}$	5-valued Walsh spectra
17	259	$\{0, \pm 2^9, \pm 2^{10}\}$	5-valued Walsh spectra

Table 3: Walsh coefficients of duals of the Welch functions

CONJECTURE 2.1. *Let $F(x) = x^{2^{\frac{n-1}{2}}+3}$ be the Welch function defined on $\mathbb{F}_{2^n}$, $n \ge 9$ odd. Then the Walsh coefficients of the duals of the component functions $\tilde{F}_i$ are $0, \pm 2^{\frac{n+1}{2}}$ or $\pm 2^{\frac{n+3}{2}}$.*

$\mathbf{v}_i$	T_{F_i}	S_{F_i}	Duals $\tilde{F}_i$ on $\mathbb{F}_2^3$	$S_{\tilde{F}_i}$
$(0,0,1)$	$(0,1,1,0,1,0,1,0)$	$(0,-4,0,4,0,4,0,4)$	$(0,1,0,1,0,1,0,1)$	$(0,8,0,0,0,0,0,0)$
$(0,1,0)$	$(0,0,1,0,0,1,1,1)$	$(0,0,4,4,4,-4,0,0)$	$(0,0,1,1,1,1,0,0)$	$(0,0,0,0,0,0,8,0)$
$(0,1,1)$	$(0,1,0,0,1,1,0,1)$	$(0,4,-4,0,4,0,0,4)$	$(0,1,1,0,1,0,0,1)$	$(0,0,0,0,0,0,0,8)$
$(1,0,0)$	$(0,0,0,1,1,1,1,0)$	$(0,0,0,0,4,4,4,-4)$	$(0,0,0,0,1,1,1,1)$	$(0,0,0,0,8,0,0,0)$
$(1,0,1)$	$(0,1,1,1,0,1,0,0)$	$(0,4,0,4,-4,0,4,0)$	$(0,1,0,1,1,0,1,0)$	$(0,0,0,0,0,8,0,0)$
$(1,1,0)$	$(0,0,1,1,1,0,0,1)$	$(0,0,4,-4,0,0,4,4)$	$(0,0,1,1,0,0,1,1)$	$(0,0,8,0,0,0,0,0)$
$(1,1,1)$	$(0,1,0,1,0,0,1,1)$	$(0,4,4,0,0,4,-4,0)$	$(0,1,1,0,0,1,1,0)$	$(0,0,0,8,0,0,0,0)$

Table 2: Component functions and their duals for the Gold function x^3 on $\mathbb{F}_{2^3}$.

On the other hand, for the Kasami case, there seems not to be any regularity about the spectra of dual components. In Table 4 we give some observations for certain n.

n	d	Walsh coefficients of $\tilde{F}$	Comment	Degree
5	13	$\{0, \pm 8\}$	AB	3
7	13	$\{0, \pm 16\}$	AB	3
7	57	$\{0, \pm 16\}$	AB	4
9	13	$\{0, \pm 2^5, -2^6\}$	4-val	3
9	241	$\{0, \pm 2^5, \pm 2^6\}$	5-val	5
11	13	$\{0, \pm 2^6, \pm 2^7\}$	5-val	3
11	57	$\{0, \pm 2^6\}$	AB	4
11	241	$\{0, \pm 2^6\}$	AB	5
11	993	$\{0, \pm 2^6, \pm 2^7\}$	5-val	6
13	13	$\{0, \pm 2^7, \pm 2^8\}$	5-val	3
13	57	$\{0, \pm 2^7\}$	AB	4
13	241	$\{0, \pm 2^7\}$	AB	5
13	993	$\{0, \pm 2^7, \pm 2^8\}$	5-val	6
13	4033	$\{0, \pm 2^7, \pm 2^8\}$	5-val	7

Table 4: Walsh spectra of the duals - Kasami case

2.3 Vectorial bent functions from AB

The classical definition of a dual that we used previously does not take into account the signs of Walsh coefficients and in general such a dual is not balanced. This was the main reason for introducing another definition of a dual of an s-plateaued function f as follows [7].
With $S_f = \{\mathbf{x} \in \mathbb{F}_2^n : W_f(\mathbf{x}) \neq 0\}$ we denote the Walsh support of the function f. Its dual function f^* on S_f of cardinality 2^{n-s} is defined as $f^* : S_f \to \mathbb{F}_2$ by

$$W_f(\omega) = 2^{\frac{n+s}{2}}(-1)^{f^*(\omega)},$$

for $\omega \in S_f$. To specify the dual function as $f^* : \mathbb{F}_2^{n-s} \to \mathbb{F}_2$ we use the concept of *lexicographic ordering.* That is, a subset $E = \{\mathbf{e}_0, \ldots, \mathbf{e}_{2^{n-s}-1}\} \subset \mathbb{F}_2^n$ is ordered lexicographically if $|\mathbf{e}_i| < |\mathbf{e}_{i+1}|$ for any $i \in [0, 2^{n-s}-2]$, where $|\mathbf{e}_i|$ denotes the integer representation of $\mathbf{e}_i \in \mathbb{F}_2^n$. More precisely, for $\mathbf{e}_i = (e_{i,0}, \ldots, e_{i,n-1})$ we have $|\mathbf{e}_i| = \sum_{j=0}^{n-1} e_{i,n-1-j} 2^j$, thus having the most significant bit of $\mathbf{e}_i$ on the left-hand side. Since S_f is not ordered in general, *we will always represent it* as $S_f = \mathbf{v} \oplus E$, where E is lexicographically ordered for some fixed $\mathbf{v} \in S_f$ and $\mathbf{e}_0 = \mathbf{0}_n$. For instance, if

$$S_f = \{(0,1,0), (0,1,1), (1,0,0), (1,0,1)\},$$

by fixing $\mathbf{v} = (0,1,1) \in S_f$, then

$$E = \{\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\} = \{(0,0,0), (0,0,1), (1,1,0), (1,1,1)\}$$

is ordered lexicographically and consequently S_f is "ordered" as $S_f = \{\omega_0, \omega_1, \omega_2, \omega_3\} = \{(0,1,1), (0,1,0), (1,0,1), (1,0,0)\}$. This way we can make a direct correspondence between $\mathbb{F}_2^{n-s}$ and S_f through E so that for $\mathbb{F}_2^{n-s} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{2^{n-s}-1}\}$, where $\mathbb{F}_2^{n-s}$ is lexicographically ordered, we have

$$f^*(\omega_j) \leftrightsquigarrow f^*(\mathbf{e}_j) \leftrightsquigarrow f^*(\mathbf{x}_j), \quad \mathbf{x}_j \in \mathbb{F}_2^{n-s}, \tag{3}$$

where $\mathbf{x}_j \in E$, $j \in [0, 2^{n-s}-1]$, i.e., we set that $S_f = \{\omega_0, \ldots, \omega_{2^{n-s}-1}\}$ is ordered so that $S_i = \mathbf{v} \oplus \mathbf{e}_i$, and $E = \{\mathbf{e}_0, \ldots, \mathbf{e}_{2^{n-s}-1}\}$ is ordered lexicographically.

Following the result of Hodžić *et al.* it is known that if $S_f = \mathbf{v} \oplus E$, where E is a linear subspace of $\mathbb{F}_2^n$, with ordering described in Section 1, then f is a semi-bent function on $\mathbb{F}_2^n$ if and only if the dual f^* is bent on $\mathbb{F}_2^{n-1}$. In this way, since all coordinate functions f_i of Gold AB functions are semi-bent on $\mathbb{F}_2^n$ and all Walsh supports S_{f_i} are even dimensional flats, one could construct bent functions $f_i^*, i \in \{0, 1, \ldots, n-1\}$, on $\mathbb{F}_2^{n-1}$ and check if (f_i^*, f_j^*) form a bent vectorial Boolean function $\mathbb{F}_2^{n-1} \to \mathbb{F}_2^2$. More generally, if we consider $k \leq \frac{n-1}{2}$ bent Boolean functions $f_{i_1}^*, f_{i_2}^*, \ldots, f_{i_k}^*$, where $i_j \in \{0, \ldots, n-1\}$, can they form a bent vectorial function. With β_k we will denote the number of bent vectorial Boolean functions $(f_{i_1}^*, \ldots, f_{i_k}^*)$ composed of the duals f_i^*. In Table 5 we give the computational results obtained in `MAGMA`. (DNE=Does not exist; NC=Not computed)

n	d	β_2	β_3	β_4	β_5	β_6
5	3	5	DNE	DNE	DNE	DNE
5	5	3	DNE	DNE	DNE	DNE
7	3	13	4	DNE	DNE	DNE
7	5	7	1	DNE	DNE	DNE
7	9	13	6	DNE	DNE	DNE
9	3	14	1	0	DNE	DNE
9	5	19	6	0	DNE	DNE
9	17	15	1	0	DNE	DNE
11	3	25	4	NC	NC	DNE
11	5	24	4	NC	NC	DNE
11	9	36	20	NC	NC	DNE
11	17	29	14	NC	NC	DNE
11	33	30	13	NC	NC	DNE

Table 5: Number of bent VBF from Gold AB

Notice that the supports of the Welch and Kasami component functions are in general not flats in $\mathbb{F}_2^n$ and therefore the same approach cannot be easily applied to these classes.

Let us consider the Gold function $F(x) = x^5$ on $\mathbb{F}_{2^5}$ whose coordinate functions are $f_1, \dots, f_5$. Let $\mathcal{W}_i^*$ be the Walsh spectra of the duals f_i^*, for $i \in \{1, \dots, n\}$. Since the duals are bent, we have that $W_i^*(\mathbf{u}) = \pm 2^2$, for all $\mathbf{u} \in \mathbb{F}_2^4$. With $\mathcal{W}_{ij}^*$ we denote the product of the Walsh spectra $\mathcal{W}_i^*$ and $\mathcal{W}_j^*$, defined as

$$\mathcal{W}_{ij}^* = 2^{-\frac{n-1}{2}} \cdot (\mathcal{W}_i^* \odot \mathcal{W}_j^*),$$

where "$\odot$" denotes the component-wise multiplication of the Walsh spectra given as integer-valued vectors of length 2^{n-1}. In Table 6 we provide the spectra of $\mathcal{W}_{ij}^*$ for $1 \le i < j \le 5$. The inverse Walsh transform of a bent Boolean function

i	j	$\mathcal{W}_{ij}^*$
1	2	$(4, 4, 4, -4, 4, 4, -4, 4, -4, 4, -4, -4, -4, 4, 4, 4)$
1	3	$(4, 4, 4, -4, 4, -4, 4, 4, -4, -4, -4, 4, -4, 4, -4, -4)$
1	4	$(4, 4, 4, -4, 4, 4, -4, 4, 4, -4, -4, -4, -4, 4, -4, -4)$
1	5	$(4, -4, -4, 4, 4, -4, -4, 4, 4, 4, -4, -4, 4, 4, -4, -4)$
2	3	$(4, 4, 4, 4, 4, -4, -4, 4, 4, -4, 4, -4, 4, 4, -4, -4)$
2	4	$(4, 4, 4, 4, 4, 4, 4, 4, -4, -4, 4, 4, 4, 4, -4, -4)$
2	5	$(4, -4, -4, -4, 4, -4, 4, 4, -4, 4, 4, 4, -4, 4, -4, -4)$
3	4	$(4, 4, 4, 4, 4, -4, -4, 4, -4, 4, 4, -4, 4, 4, 4, 4)$
3	5	$(4, -4, -4, -4, 4, 4, -4, 4, -4, -4, 4, -4, -4, 4, 4, 4)$
4	5	$(4, -4, -4, -4, 4, -4, 4, 4, 4, -4, 4, 4, -4, 4, 4, 4)$

Table 6: Products $\mathcal{W}_{ij}^*$ of the Walsh spectra $\mathcal{W}_i^*$ and $\mathcal{W}_j^*$

(2) can be generalized, that is, for a given Walsh spectra $\mathcal{W} = (w_0, \dots, w_{2^n-1})$ and fixed $\mathbf{u} \in \mathbb{F}_2^n$ one can define an integer valued function $W^{-1} : \mathbb{F}_2^n \to \mathbb{Z}$ as

$$W^{-1}(\mathbf{u}) = 2^{-n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} w_{\mathbf{x}_{int}} (-1)^{\mathbf{x} \cdot \mathbf{u}}, \tag{4}$$

where $\mathbf{x}_{int}$ represents the integer representation of $\mathbf{x}$ and $w_{\mathbf{x}_{int}} \in \mathcal{W}$. With $\mathcal{W}^{-1} = (W^{-1}(\mathbf{u}_0), \dots, W^{-1}(\mathbf{u}_{2^n-1}))$, $\mathbf{u}_k \in \mathbb{F}_2^n$, we denote the inverse Walsh spectra (in short IWS) of $\mathcal{W} = (w_0, \dots, w_{2^n-1})$. In Table 7 we give the IWS of the spectra $\mathcal{W}_{ij}^*$.

i	j	$\mathcal{W}_{ij}^{*,-1}$
1	2	$(1, -1, 1, -1, -1, 1, 1, -1, 1, 1, 1, 1, 1, 1, -1, -1)$
1	3	$(0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 2, -2)$
1	4	$(0, 0, 2, 0, 0, 2, 0, 0, 2, 0, 0, 0, 0, 0, 0, -2)$
1	5	$(0, 0, 2, 2, 0, 0, 0, 0, 0, 0, -2, 2, 0, 0, 0, 0)$
2	3	$(1, 1, 1, 1, 1, 1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1)$
2	4	$(2, 0, 0, 0, 0, 0, -2, 0, 2, 0, 0, 0, 0, 0, 2, 0)$
2	5	$(0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 2, -2, 0, 2, 0)$
3	4	$(2, 0, 0, 0, 0, 0, 0, -2, 0, 0, 0, 2, 2, 0, 0, 0)$
3	5	$(0, 0, 0, 0, -2, 2, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0)$
4	5	$(1, 1, -1, 1, -1, 1, 1, 1, -1, 1, 1, 1, -1, -1, 1, -1)$

Table 7: The IWS of $\mathcal{W}_{ij}^*$

We notice three IWS that have values ± 1. From Table 5 we know that for $(n, d) = (5, 5)$ there are exactly three pairs (f_i^*, f_j^*) of bent vectorial functions mapping from $\mathbb{F}_2^4 \to \mathbb{F}_2^2$. The coordinates ij of $f_{ij}^* = f_i^* \oplus f_j^*$ correspond exactly to the coordinates ij of the three Walsh spectra $\mathcal{W}_{ij}^*$ for which the IWS have values ± 1. We obtained the same results for $(n, d) = (5, 3)$ and $(n, d) = (7, 9)$ leading us to the following conjecture.

Conjecture 2.2. *Let $F(x) = x^d$, $d = 2^i + 1$, $\gcd(i, n) = 1$, $1 \le i \le \frac{n-1}{2}$, be the Gold function defined on $\mathbb{F}_{2^n}$. With $f_1, \dots, f_n$ we denote the truth tables of its coordinate functions and with $f_1^*, \dots, f_n^*$ their corresponding duals defined on $\mathbb{F}_2^{n-1}$. Let $\mathcal{W}_{i,j}^{*,-1}$ denote the IWS, as described previously. Then, $f_{ij}^* = f_i^* \oplus f_j^*$, $1 \le i < j \le n$, is bent if and only if $|\mathcal{W}_{ij}^{*,-1}| = \mathbf{1}$.*

3. CONCLUSION

In the last few decades a lot of research has been done in the field of vectorial Boolean functions and understanding their structure and properties. However, many problems still remain open. In our future research we wish to get a better understanding of these duals and their connection with the original function from which they were derived.

4. REFERENCES

[1] A. Bapić. *On certain properties of APN and AB functions: master's thesis.* PhD thesis, UP FAMNIT, 2019. `http://www.famnit.upr.si/sl/izobrazevanje/zakljucna_dela/view/834`.

[2] L. Budaghyan. *Construction and Analysis of Cryptographic Functions.* Springer, 2014.

[3] C. Carlet. *Boolean Functions for Cryptography and Error-Correcting Codes*, page 257–397. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2010.

[4] C. Carlet. *Vectorial Boolean Functions for Cryptography*, pages 398–470. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2010.

[5] C. Carlet and S. Mesnager. On the supports of the walsh transforms of boolean functions. *IACR Cryptology ePrint Archive*, 2004:256, 2004.

[6] F. Chabaud and S. Vaudenay. Links between differential and linear cryptanalysis. In A. De Santis, editor, *Advances in Cryptology — EUROCRYPT'94*, pages 356–365, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg.

[7] S. Hodzic, E. Pasalic, and Y. Wei. A general framework for secondary constructions of bent and plateaued functions. *CoRR*, abs/1809.07390, 2018.

[8] S. Hodzic, E. Pasalic, Y. Wei, and F. Zhang. Designing plateaued boolean functions in spectral domain and their classification. *IEEE Trans. Information Theory*, 65(9):5865–5879, 2019.

[9] M. Matsui. Linear cryptanalysis method for DES cipher. In *Advances in Cryptology - EUROCRYPT '93, Proceedings*, pages 386–397, 1993.

[10] K. Nyberg. S-boxes and round functions with controllable linearity and differential uniformity. In *Fast Software Encryption: Second International Workshop. Leuven, Belgium, 14-16 December 1994, Proceedings*, pages 111–130, 1994.

A new graph decomposition method for bipartite graphs

[Extended Abstract]

Béla Csaba[*]
Bolyai Institute, Interdisciplinary Excellence Centre
University of Szeged
6720 Szeged, Hungary, Aradi vértanúk tere 1.
bcsaba@math.u-szeged.hu

ABSTRACT

We introduce a new graph decomposition method, which works for relatively small or sparse graphs, and can be used to substitute the Regularity lemma of Szemerédi in some graph embedding problems.

Categories and Subject Descriptors

G.2.2 [**Graph theory**]: Extremal graph theory; Regularity; Trees

General Terms

Graph theory

Keywords

regularity, graph decomposition, embedding

1. INTRODUCTION

All graphs considered in this paper are simple. The Szemerédi Regularity lemma [10] is one of the most powerful tools of graph[1] theory. It is also used in many areas outside graph theory, for example in number theory and algorithms.

Theorem 1 (Szemerédi). *For every $\varepsilon > 0$ there exists a $n_0 = n_0(\varepsilon) > 0$ such that if G is a simple graph on $n \geq n_0$ vertices then G admits an ε-regular equipartition of its vertex set.*

We will give a short introduction to the necessary notions in the next section. Here we only mention that ε-regularity is a notion of quasirandomness, and equipartition means, roughly, the partition of the vast majority of the vertex set of G into equal sized subsets so that all, but an ε proportion of the pairs of subsets span an ε-regular bipartite subgraph of G.

[*]Partially supported by the Ministry of Human Capacities, Hungary, Grant 20391- 3/2018/FEKUSTRAT, the NKFIH Fund No. KH 129597 and SNN 117879.

[1]There are also hypergraph versions that play crucial role in extremal hypergraph theory and combinatorial number theory, see eg., [6] or [9].

The dependence of n_0 on ε in Theorem 1 is determined by a *tower function* T evaluated at $1/\varepsilon^5$, where T can be defined inductively as follows: $T(1) = 2$, and for $i > 1$ we have $T(i) = 2^{T(i-1)}$. Hence, the value of n_0 makes the Regularity lemma essentially impractical. It is also well-known that we cannot hope for a much better bound, since as was proven by Gowers [4], there are graphs for which the number of clusters in the Regularity lemma is necessarily a tower function of $1/\varepsilon$. Note also that the lemma is only meaningful for so called dense graphs, that is, graphs that contain a constant proportion of the possible edges.

In this paper we present a new graph decomposition method for bipartite graphs, which can be applied for graphs of practical size and for graphs having vanishing density. While the Regularity lemma is useful in many areas of mathematics and computer science, our contribution may not be so widely applicable. Still, it can be used for finding certain subgraphs in a host graph. As an illustration, we will give the details of a tree embedding algorithm that uses this graph decomposition method.

Let us mention that Gowers in [5] presented a decomposition for bipartite graphs that is somewhat similar to the one discussed here, and used it for a problem in number theory. That decomposition has different parameters and a much longer and harder proof. Due to the importance of the Regularity lemma, other researchers also found weakened versions (eg. [1], [3]) in which the dependence of ε and n_0 is not determined by a tower function. These are important developments with several applications, still, none of them seems to be so widely applicable as the original one. One can find more details in [2]. The so called absorption method [11] is also a choice for avoiding the use of the Regularity lemma in some embedding problems.

The outline of the paper is as follows. First, we provide the necessary notions for the decomposition and then describe the decomposition method in the next section. In the subsequent section we provide an application, namely, we show that we can find a large subtree in a graph on n vertices having $\Omega(n^2 \log\log n/\log n)$ edges.

2. DEFINITIONS, MAIN RESULT

Given a graph G with vertex set V and edge set E, we let $deg_G(v)$ denote the degree of $v \in V$. If it is clear from the context, the subscription may be omitted. The neighborhood of v is denoted by $N(v)$, so $deg(v) = |N(v)|$. The minimum degree of G is denoted by $\delta(G)$. If $S \subset V$, then $deg(v;S) = |N(v) \cap S|$. The set of edges between two disjoint sets $S, T \subset V$ is denoted by $E(S,T)$, and we let $e(S,T) = |E(S,T)|$. We also let $e(G) = |E(G)|$.

Let $G = G(A,B)$ be a bipartite graph. The *density* $d_G(A,B)$ or, if G is clear from the context, $d(A,B)$, is defined as follows:

$$d(A,B) = d_G(A,B) = \frac{e(G)}{|A| \cdot |B|}.$$

Given a number $\varepsilon \in (0,1)$ we say that $G = G(A,B)$ is an *ε-regular pair* if the following holds for every $A' \subset A$, $|A'| \geq \varepsilon|A|$ and $B' \subset B$, $|B'| \geq \varepsilon|B|$:

$$|d_G(A,B) - d_G(A',B')| \leq \varepsilon.$$

The ε-regular equipartition of a graph G on n vertices means that $V(G) = V_0 \cup V_1 \cup \ldots \cup V_k$ such that $V_i \cap V_j = \emptyset$ for $i \neq j$, $|V_0| \leq \varepsilon n$, $||V_i| - |V_j|| \leq 1$ for every $1 \leq i,j \leq k$ and all but at most εk^2 pairs V_iV_j are ε-regular for $1 \leq i,j$. The V_i sets are called clusters, and V_0 is the exceptional cluster.

Roughly speaking, the Regularity lemma asserts that every graph can be well approximated by a collection of quasirandom graphs that are defined between the non-exceptional clusters. Unfortunately, by the result of Gowers [4], in general the number of non-exceptional clusters is a tower function of $1/\varepsilon$.

While our goal is to provide an alternative for the Regularity lemma, we will also make use of the regularity concept. Our definition is slightly more permissive than the usual one above, this enables us to give a very short proof of our decomposition, and it is still powerful enough to be applicable in several embedding problems. It is called *lower regularity*, and is used by other researchers as well.

Definition 2. *Given a bipartite graph $G = G(A,B)$ we say that G is a lower $(\varepsilon, \eta, \gamma)$-regular pair, if for any $A' \subset A, B' \subset B$ with $|A'| \geq \varepsilon|A|$, $|B'| \geq \eta|B|$ we have $e(A',B') \geq \gamma \cdot |A'| \cdot |B'|$.*

Note that in the usual definition of an ε-regular pair one has $\varepsilon = \eta$, and the edge density between two sufficiently large subsets is between $d_G - \varepsilon$ and $d_G + \varepsilon$. We want to have flexibility in this notion, and allow sub-pairs with relatively low density, and the $\varepsilon \neq \eta$ case, too.

We are ready to state our main result, the precise formulation is as follows.

Theorem 3. *Let $G = G(A,B)$ be a bipartite graph with vertex classes A and B such that $|A| = n$ and $|B| = m$, and every vertex of A has at least δm neighbors in B. Let $0 < \varepsilon, \eta, \gamma < 1$ be numbers so that $\eta \leq 1/6$ and $\gamma \leq \min\{\eta/4, \delta/20\}$. Then there exists a partition $A = A_0 \cup A_1 \cup \ldots \cup A_k$, and k not necessarily disjoint subsets $B_1, \ldots, B_k$ of B, such that $|A_i| \geq \varepsilon \cdot \exp\left(-2\log(\frac{1}{\varepsilon})\log(\frac{2}{\delta})/\eta\right) n$ for $i \geq 1$, $|A_0| \leq \epsilon n$, the subgraphs $G[A_i, B_i]$ for $1 \leq i \leq k$ are all lower $(\varepsilon, \eta, \gamma)$-regular, and*

$$\sum_{i=1}^{k} e(G[A_i, B_i]) \geq e(G) - (\varepsilon + 2\gamma)nm.$$

Moreover,

$$k \leq \frac{2}{\varepsilon\delta} e^{2\log(\frac{1}{\varepsilon})\log(\frac{2}{\delta})/\eta}.$$

3. PROOF OF THEOREM 3

Let us remark that we will not be concerned with floor signs, divisibility, and so on in the proof. This makes the notation simpler, easier to follow.

As we have seen, edge density plays an important role in regularity. We need a simple fact which is called *convexity of density* (see eg. in [7]), the proof is left for the reader.

Claim 4. *Let $F = F(A,B)$ be a bipartite graph, and let $1 \leq k \leq |A|$ and $1 \leq m \leq |B|$. Then*

$$d_F(A,B) = \frac{1}{\binom{|A|}{k}\binom{|B|}{m}} \sum_{X \in \binom{A}{k}, Y \in \binom{B}{m}} d(X,Y).$$

In order to prove Theorem 3 we need a lemma that is the basic building block of our decomposition method.

Lemma 5. *Let $F = F(A,B)$ be a bipartite graph with vertex classes A and B such that $|A| = a$ and $|B| = b$, and every vertex of A has at least δb neighbors in B. Let $0 < \varepsilon, \eta, \gamma < 1$ be numbers so that $\eta \leq 1/6$ and $\gamma \leq \min\{\eta/4, \delta/20\}$. Then F contains a lower $(\varepsilon, \eta, \gamma)$-regular pair $F[X,Y]$ such that $|X| \geq \exp\left(-2\log(\frac{2}{\varepsilon})\log(\frac{2}{\delta})/\eta\right) a$ and $|Y| \geq (\delta(1-\eta) - 2\gamma)b$.*

Proof: We prove the lemma by finding two sequences of sets $X_0 = A, X_1, \ldots, X_l$ and $Y_0 = B, Y_1, \ldots, Y_l$ such that for every $1 \leq i \leq l$ we have $X_i \subset X_{i-1}$, $Y_i \subset Y_{i-1}$,

$$\varepsilon|X_{i-1}|/2 \leq |X_i| \leq \varepsilon|X_{i-1}|$$

and

$$|Y_i| = (1-\eta)|Y_{i-1}|,$$

moreover, the last pair $F[X_l, Y_l]$ is lower $(\varepsilon, \eta, \gamma)$-regular. Hence, we may choose $X = X_l$ and $Y = Y_l$.

We find the set sequences $\{X_i\}_{i \geq 1}$ and $\{Y_i\}_{i \geq 1}$ by the help of an iterative procedure. This procedure stops in the lth step if $F[X_l, Y_l]$ is lower $(\varepsilon, \eta, \gamma)$-regular. We have another stopping rule: if $|Y_l| \leq (\delta(1 + \eta/2) - 2\gamma)b$ for some l, we stop. Later we will see that in this case we have found what is desired, $F[X_l, Y_l]$ must be a lower $(\varepsilon, \eta, \gamma)$-regular pair.

In the beginning we check, if $F[X_0, Y_0]$ is a lower $(\varepsilon, \eta, \gamma)$-regular pair. If it is, we stop. If not then X_0 has a subset X_1' precisely of size $\varepsilon|X_0|$ and Y_0 has a subset Y_1' precisely of size $\eta|Y_0|$ such that $e(F[X_1', Y_1']) < \gamma|X_1'| \cdot |Y_1'|$, here we used Claim 4 in order to obtain the sizes of X_1' and Y_1'.

Let X_1'' be the set of those vertices of X_1' that have more than $2\gamma|Y_1'|$ neighbors in $|Y_1'|$. Simple counting shows that $|X_1''| \leq |X_1'|/2$. Let $X_1 = X_1' - X_1''$, those vertices of X_1' that have less than $2\gamma|Y_1'|$ neighbors in $|Y_1'|$. By the above we have $|X_1'|/2 \leq |X_1| \leq |X_1'|$. Set $Y_1 = Y_0 - Y_1'$.

For $i \geq 2$ the above is generalized. If $F[X_{i-1}, Y_{i-1}]$ is not a lower $(\varepsilon, \eta, \gamma)$-regular pair then we do the following. First find $X_i' \subset X_{i-1}$ and $Y_i' \subset Y_{i-1}$ such that $|X_i'| = \varepsilon|X_{i-1}|$ and $|Y_i'| = \eta|Y_{i-1}|$ and $e(F[X_i', Y_i']) < \gamma|X_i'| \cdot |Y_i'|$. Similarly to the above we define $X_i \subset X_i'$ to be the set of those vertices of X_i' that have less than $2\gamma|Y_i'|$ neighbors in Y_i'. As before, we have $|X_i'|/2 \le |X_i| \le |X_i'|$. Finally, we let $Y_i = Y_{i-1} - Y_i'$.

Using induction one can easily verify that the claimed bounds for $|X_i|$ and $|Y_i|$ hold for every i. It might not be so clear that this process stops in a relatively few iteration steps. For that we first find an upper bound for the number of edges that connect the vertices of X_i with $B - Y_i$. If $u \in X_i$ then u have at most $2\gamma(|Y_1'| + \ldots + |Y_i'|) \le 2\gamma b$ neighbors in $B - Y_i$ using that $Y_s' \cap Y_t' = \emptyset$ for every $s \neq t$.

Next we show that if $(\delta(1 + \eta/2) - 2\gamma)(1 - \eta)b < |Y_l| \le (\delta(1 + \eta/2) - 2\gamma)b$ then $F[X_l, Y_l]$ must be lower regular. Assume that $u \in X_l$. Then $deg(u; Y_l) \ge (\delta - 2\gamma)b$, using our argument above, hence, the number of *non-neighbors* of u in Y_l is at most $(\delta(1 + \eta/2) - 2\gamma)b - (\delta - 2\gamma)b = \delta\eta b/2$. Let $Y' \subset Y_l$ be arbitrary with $|Y'| = \eta|Y_l|$. Then

$$\frac{5}{6}\eta(\delta(1 + \eta/2) - 2\gamma)b \le |Y'| \le \eta(\delta(1 + \eta/2) - 2\gamma)b,$$

using that $\eta \le 1/6$. We have

$$deg(u; Y') \ge |Y'| - \delta\eta b/2 \ge \frac{5}{6}\eta(\delta(1 + \eta/2) - 2\gamma)b - \delta\eta b/2.$$

Using the upper bounds we imposed on η and γ, one easily obtains that

$$deg(u; Y') \ge (\delta\eta/3 + 5\delta\eta^2/12 - 5/3\gamma\eta)b \ge \gamma|Y'|.$$

Hence, for every $X' \subset X_l$ and $Y' \subset Y_l$ with $|Y'| = \eta|Y_l|$ we have

$$e(X', Y') \ge \gamma|X'| \cdot |Y'|,$$

that is, if the procedure stopped because we applied the stopping rule, then the resulting pair must always be lower $(\varepsilon, \eta, \gamma)$-regular. Of course, this means that no matter how the procedure stops, it finds a lower regular pair.

Next we upper bound the number of iteration steps. In every step the Y-side shrinks by a factor of $(1 - \eta)$. We also have that $|Y_l| > (\delta(1 + \eta/2) - 2\gamma)(1 - \eta)b$. Putting these together we get that

$$(1 - \eta)^l > (\delta(1 + \eta/2) - 2\gamma)(1 - \eta) > \delta/2.$$

Hence,

$$l < \frac{\log(2/\delta)}{\log(1/(1 - \eta))} < 2\frac{\log(2/\delta)}{\eta},$$

here we used elemantary calculus (in particular, the Taylor series expansion of $\log(1 + x)$) and our condition that η is less than $1/6$.

What is left is to show the lower bound for $|X_l|$. Note, that $|X_i|/|X_{i-1}| \ge \varepsilon/2$ for every $i \ge 1$. Hence,

$$|X_l| \ge \left(\frac{\varepsilon}{2}\right)^l a = e^{-2\log(2/\varepsilon)\log(2/\delta)/\eta}a.$$

□

We are ready to prove the main result of the paper.

Proof (of Theorem 3): The proof is based on iteratively applying Lemma 5. First we apply Lemma 5 for G and find a lower $(\varepsilon, \eta, \gamma)$-regular pair $G[X_l, Y_l]$, where $X_l \subset A$ and $Y_l \subset B$. Let $A_1 = X_l$ and $B_1 = Y_l$. Next we repeat this procedure for the graph $G[A - A_1, B]$. Similarly to the above we define the A_2 and B_2 sets, where $A_2 \subset A - A_1$, $B_2 \subset B$, and $G[A_2, B_2]$ is a lower $(\varepsilon, \eta, \gamma)$-regular pair.

Continue this way, finding the lower regular pairs $G[A_i, B_i]$ using Lemma 3 such that $A_i \subset A - (A_1 \cup \ldots \cup A_{i-1})$, $B_i \subset B$, and $G[A_i, B_i]$ is a lower $(\varepsilon, \eta, \gamma)$-regular pair. We stop when

$$|A - (A_1 \cup \ldots \cup A_i)| < \varepsilon|A|.$$

At this point set $A_0 = A - (A_1 \cup \ldots \cup A_i)$.

Let us now prove the upper bound for the number of pairs in the decomposition. As we have shown earlier $|A_i| \ge \exp\left(-2\log(\frac{2}{\varepsilon})\log(\frac{2}{\delta})/\eta\right) n$ for $i \ge 1$. The number of edges in an A_iB_i pair is at least $|A_i|(\delta - 2\gamma)m > |A_i|\delta m/2$. For any $1 \le i \neq j \le k$ the edge sets of the pairs A_iB_i and A_jB_j are disjoint, and the total number of edges in lower regular pairs is at most nm. Hence, we have

$$k \le \frac{2nme^{2\log(\frac{1}{\varepsilon})\log(\frac{2}{\delta})/\eta}}{\varepsilon\delta nm} = \frac{2}{\varepsilon\delta}e^{2\log(\frac{1}{\varepsilon})\log(\frac{2}{\delta})/\eta}.$$

There is only one question left, bounding the total number of edges that belong to the lower regular pairs. Assume first that $u \in A - A_0$. We saw earlier in Lemma 3 that u lost at most $2\gamma|B|$ edges. This explains the $2\gamma mn$ term in the theorem. If $u \in A_0$, none of the edges incident to it belongs to any of the lower regular pairs, however, $|A_0| \le \varepsilon n$, therefore, the total number of edges incident to vertices of A_0 is at most εnm. With this we found the decomposition of G what was desired. □

Let us finish this section with a remark. Without the lower bound for the sizes of the A_i sets, the Theorem 3 would be trivial: every vertex $v \in A$ could be a "subset" A_v (a singleton), and its neighborhood $N(v)$ is the corresponding B_v. The result is interesting only when the A_i sets are large. For example, let G be the following. It is a sparse bipartite graph with vertex classes A and B such that $|A| = |B| = n$. Set $\varepsilon = \eta = 1/10$, $\delta = \log\log n/\log n$, and $\gamma = \delta/20$. Then G has $O(n^2 \log\log n/\log n)$ edges, and the A_i sets for $i \ge 1$ have size $\Omega(n/(\log n)^c)$, where $c < 60$, and every (A_i, B_i) pair is a lower $(0.1, 0.1, \log\log n/(20\log n))$-regular pair.

4. AN APPLICATION

The main advantage of Theorem 3 is that, as the above example shows, it can be applied for graphs having "real-life" size, or foe relatively sparse graphs, unlike the Szemerédi Regularity lemma. Therefore, it may extend the scope when usual methods for graph embedding (eg. counting lemma or the Blow-up lemma [8]) can be applied.

Below in Proposition 6 we show how to embed an almost spanning tree into *one* lower regular pair. This can be used to approximately tile the edge set of a sufficiently dense graph G by large edge-disjoint subtrees. The rough sketch of this approximate decomposition is as follows. Apply Theorem 3 for the graph G, and then using Proposition 6 find one-one almost spanning subtree in the lower regular pairs. Delete the edges used for the subtrees. If the resulting graph has sufficiently many edges then one can use Theorem 3 again, and then Proposition 6 for every lower regular pair. The process stops when the remaining vacant subgraph of G is too sparse, and therefore one cannot find many large degree vertices in it. Hence, with this method one can tile the vast majority of edges of a graph having sufficiently large density. Due to the length of the proof we do not give every detail in this extended abstract.

Given a tree T rooted at r its *level sets* are defined as follows: $L_1 = r$, $L_2 = N(r)$, in general, $L_{i+1} = N^i(r)$, etc., where $N^i(r)$ denotes those vertices of T that are exactly at distance i from r in T.

Proposition 6. *Let $0 < \varepsilon, \eta, \gamma < 1/10$ such that $\eta = 4\gamma$ and $\varepsilon = \gamma^2/10$. Assume $G[A, B]$ is a lower $(\varepsilon, \eta, \gamma)$-regular pair. Let T be a tree rooted at r, having color classes X and Y such that $r \in X$, $|X| \le (1 - 10\gamma)|A|$ and $|Y| \le (1 - 10\gamma)|B|$. Assume further that for every $i \ge 1$ we have $|L_{2i}| \le \varepsilon|A|$ and $|L_{2i+1}| \le \eta|B|$. Then $T \subset G[A, B]$.*

Let us remark that T does not have to have bounded degree, unlike in many tree embedding results. In fact, it can have vertices with linearly large degrees, if δ and the other parameters are constants. The statement holds for every G for which Lemma 5 can be applied, hence, G can have $o(n^2)$ edges.

We need the following simple claim, the proof is left for the reader.

Claim 7. *Let $F = F(U, V)$ be a lower $(\varepsilon, \eta, \gamma)$-regular pair. Let $U' \subset U$ and $V' \subset V$ such that $|U'| \ge \varepsilon|U|$ and $|V'| \ge \eta|V|$. Then U' can have at most $\varepsilon|U|$ vertices that have less than $\gamma|V'|$ neighbors in V'. Similarly, V' can have at most $\eta|V|$ vertices that have less than $\gamma|U'|$ neighbors in U'.*

Proof of the theorem: We prove the theorem via an embedding algorithm. Let $X = \{x_1, \dots, x_k\}$ and $Y = \{y_1, \dots, y_m\}$, where $r = x_1$. We will find the images of the vertices of T so that we embed height-2 subtrees of T in every step, having vertices from Y in the middle level.

Denote $\varphi : V(T) \longrightarrow A \cup B$ the edge-preserving mapping that we construct. Let A^f, respectively, B^f denote the *free* (ie. vacant) vertices of A, respectively, B. These sets are shrinking as the embedding of T proceeds, but due to the conditions of Proposition 6 we always have that $|A^f| \ge 10\gamma|A|$ and $|B^f| \ge 10\gamma|B|$. Divide A^f randomly into three disjoint, approximately equal-sized subsets A_1^f, A_2^f and A_3^f. Let $B_1' \subset B^f$ be the set of those vertices that have less than $\gamma|A|$ neighbors in A_1^f, the sets B_2' and B_3' are defined analogously. Then $|B_1'|, |B_2'|, |B_3'| \le \eta|B|$.

Let v be an arbitrary vertex of, say, A_1^f that has at least $\gamma|B^f - B_1' - B_2' - B_3'|$ neighbors in $B^f - B_1' - B_2' - B_3'$. By Claim 7 we know that A_1^f has many such vertices. By the definition of the B_i' sets we have that every vertex in $N(v)$ has at least $\gamma|A_i^f|/4$ neighbors in A_i^f for $i = 1, 2, 3$. Pick the largest of the A_i^f sets, say, it is A_2^f. Then the height-2 subtree originating at r will be embedded so that the neighbors of r will be mapped onto $N(v)$ arbitrarily ($|L_2|$ is smaller, than $|N(v)|$), and by construction every vertex of $N(v)$ will have many neighbors in A_2^f. Now we redetermine the subsets B_1', B_2', B_3', as some vertices have become covered in A and in B. For the third level of the height-2 subtree originating at r we take those vertices of A_2^f that are neighboring with at least a γ proportion of $B^f - B_1' - B_2' - B_3'$. Note that for every $\varphi(y)$ where y is in the middle level we have many choices: except at most $\varepsilon|A|$ vertices of A_2^f the neighborhood $N(\varphi(y))$ contains vertices with large degrees into $B^f - B_1' - B_2' - B_3'$. This means that we are able to map the third level. Next we continue this process so that we embed the height-2 subtrees originating at the vertices of the third level one-by-one.

There is only one missing detail here, the reason why we divided A^f randomly in the beginning: if we have three A_i^f sets, then the *active level* belongs to one of them, say, it is A_i^f. Then we map the vertices of T that are exactly two levels below them into the larger A_j^f-set, where $j \in \{1, 2, 3\} - i$. This way we never eat up any of the A_i^f sets at any point in time. Since the color classes of T are sufficiently small, this procedure never gets stuck. □

5. REFERENCES

[1] N. Alon, R. A. Duke, H. Lefmann, V. Rödl and R. Yuster, The algorithmic aspects of the Regularity Lemma, *Journal of Algorithms* **16**, (1994) 80–109.

[2] j. Fox, L. M. Lovász, Y. Zhao, On Regularity Lemmas and their Algorithmic Applications, *Combinatorics, Probaility and Computing*, **26** (2017) 481–505.

[3] A. M. Frieze and R. Kannan, Quick approximations to matrices and applications, *Combinatorica*, **19** (1999) 175–220.

[4] T. Gowers, Lower bounds of tower type for Szemerédi's uniformity lemma, *Geometric and Functional Analysis* **7** (1997), 322–337.

[5] W. T. Gowers, Bipartite graphs of approximate rank one, preprint.

[6] W. T. Gowers, Hypergraph regularity and the multidimensional Szemerédi theorem, *Ann. of Math.* **166** (2007), 897–946.

[7] J. Komlós, M. Simonovits, Szemerédi's Regularity Lemma and its Applications in Graph Theory, Combinatorics, Paul Erdős is eighty, Vol. **2** (Keszthely, 1993), 295–352. J. Komlós, M. Simonovits, Szemerédi's Regularity lemma and its applications in graph theory

[8] J. Komlós, G.N. Sárközy and E. Szemerédi, Blow-up Lemma, *Combinatorica*, **17** (1997), 109–123.

[9] V. Rödl, B. Nagle, J. Skokan, M. Schacht and Y. Kohayakawa, The hypergraph regularity method and its applications, *P. Natl. Acad. Sci. USA* **102** (2005), 8109–8113.

[10] E. Szemerédi, Regular Partitions of Graphs, Colloques Internationaux C.N.R.S No **260** - Problèmes Combinatoires et Théorie des Graphes, Orsay, (1976) 399–401

[11] E. Szemerédi, Is laziness paying off? (Absorbing method) In: Zannier U. (eds) Colloquium De Giorgi 2010-2012. Publications of the Scuola Normale Superiore, vol 4. Edizioni della Normale, Pisa (2013) 17–34.

A polynomial-time algorithm for recognizing subgraph-symmetry-compressible graphs

Uroš Čibej
University of Ljubljana
Faculty of computer and information science
Večna pot 113, Ljubljana
uros.cibej@fri.uni-lj.si

Jurij Mihelič
University of Ljubljana
Faculty of computer and information science
Večna pot 113, Ljubljana
jurij.mihelic@fri.uni-lj.si

ABSTRACT

Symmetries in graphs represent a natural mechanism to reduce redundancies in graph representations. We present a class of graphs that can be compressed using symmetries and an extension of this class that encompasses many more graphs from practice. We call the extended class subgraph-symmetry-compressible graphs. For this class we demonstrate that it can be recognized in polynomial time.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems

General Terms

Theory

Keywords

graph theory, compression, symmetries

1. INTRODUCTION

Due to enormous quantities of data, data compression is a very important building block of systems that gather, transfer, and save the information. One type of data that is becoming more and more important are graphs. This paper is a part of the exploration of how to use symmetries as a mechanism that could reduce the amount of data in such structures.

Symmetries are mathematically defined as graph automorphisms, and we will use these two terms synonymously. Based on this notion we formally define how the symmetry can be used to uniquely represent the graph, and what it means that a graph representation is smaller than some other representation. Based on this concepts we define two classes of graphs: 1) symmetry-compressible graphs and 2) subgraph symmetry-compressible graphs. This two graph classes have already been defined in our previous work [3]. This paper focuses on the question of computational complexity of subgraph symmetry-compressible graphs. We establish that the problem of recognizing this class of graph can be solved in polynomial time and we describe a practical algorithm for it.

The paper is structured as follows. The next section gives some preliminaries and introduces the concept of symmetry-compressible and subgraph symmetry-compressible graphs. Section 3 states a few basic theorems about symmetry--compressible graphs which will be used in the main result of this paper. Section 4 constructs the main properties of $\mathcal{SSC}$ graphs, which are used in Section 5 to describe a polynomial-time algorithm.

2. DEFINITIONS

We represent an undirected graph as a set of edges, $G = \{(u, v)\}$, where u, v are members of the set of vertices. The set of all vertices is denoted as $V(G)$.

An automorphism of a graph is a bijective mapping (a permutation) of $V(G)$, which preserves the connectivity of the graph. All automorphisms of a graph G form a group, denoted by $Aut(G)$. We work with two related representations of the symmetries. The standard representation are permutations of the vertex set, i.e., bijective functions $\pi : V(G) \to V(G)$. But we also require an alternative representation of an automorphism π, the permutation of the edges. The edge permutation, induced by the vertex permutation π, is defined as $\overline{\pi}((u, v)) = (\pi(u), \pi(v))$.

Permutations are considered in the standard cycle notation, i.e., as a set of disjoint cycles (e.g. (123)(45)(6)). This set contains also the cycles with only one element (identities of the permutation). The notation $cyc(\pi)$ denotes the set of all the cycles of the permutation π. Same goes for the edge permutation $\overline{\pi}$. To obtain the cycle which contains a vertex v (or edge e), we use $cyc(\pi, v)$ ($cyc(\overline{\pi}, e)$ for edges).

Since graphs are represented as a set of pairs, in order to have a comparable representation, also permutations will be represented as a set of pairs. Trivially, since it is a function, a symmetry can be represented by $|V(G)|$ pairs. But many redundancies can be removed in such a representation. Namely, the identities can be omitted, and one pair from each cycle can also be left out. The size (of the representation) is expressed in terms of the cycle sizes as

$|\pi| = \sum_{c \in cyc(\pi)} (|c| - 1)$.

Now we can define the two classes of graph, the first one being the class of graphs that can be more efficiently represented by using one automorphism.

Definition 1 (Symmetry Compressible Graphs). *A graph G is symmetry compressible ($G \in \mathcal{SC}$) if $\exists \pi \in Aut(G)$: $|\pi| + |G^\pi| < |G|$.*

Where G^π is the *required part* of the graph (under the symmetry π) and is defined as $G^\pi = \{e \in G \mid e = \min_{f \in cyc(\overline{\pi}, e)} f\}$. In general, the required part of the graph is the set of edges that are extracted from the cycles of $\overline{\pi}$ so that there is exactly one edge from each cycle in G^π. This makes it possible to reconstruct the original graph if we know π.

The second class of graph is the class of graphs that contains a symmetry-compressible graph as a subgraph.

Definition 2 (Subgraph $\mathcal{SSC}$). *A graph is subgraph symmetry-compressible ($G \in \mathcal{SSC}$) if $\exists H \subseteq G$ and $H \in \mathcal{SC}$.*

3. BASIC RESULTS ON SC GRAPHS

This section provides a few basic results about symmetry-compressible graphs. We will provide the relevant theorems without proofs. The complete proofs are available in [3].

The first theorem demonstrates the relation between π and $\overline{\pi}$ (moved vertices and moved edges) for symmetry-compressible graphs. The next theorem shows the condition when symmetries between two different connected components compress the graph (it trivially generalizes to more components). The next theorem shows that trees (forests to be precise) are not symmetry-compressible. And finally, we explore the compressibility of cycle graphs.

Theorem 1. $G \in \mathcal{SC} \Longleftrightarrow \exists \pi \in Aut(G) : |\pi| < |\overline{\pi}|$

Theorem 2. *Let us assume G is composed of two connected components $G_1 \notin \mathcal{SC}$ and $G_2 \notin \mathcal{SC}$, and there is a symmetry π mapping the component G_1 onto G_2. Then $G \in \mathcal{SC} \iff |G_1| > |V(G_1)|$.*

Theorem 3. *If the graph T is a forest then $T \notin \mathcal{SC}$.*

Theorem 4. *Even cycles $C_k, (k = 2i)$ are symmetry compressible, whereas odd cycles $C_k, (k = 2i+1)$ are not.*

Using these basic results, we can now explore in more detail the properties of $\mathcal{SSC}$ graphs.

4. CHARACTERIZATION OF SSC GRAPHS

The main result of this section will establish that the $\overline{\mathcal{SSC}}$ graph class can be specified by a forbidden graph characterization. More precisely, a graph is not in $\mathcal{SSC}$ if and only if it does not contain any of the following subgraphs: even cycle, compressible symmetric-handcuff, and compressible handcuff pair.

Definition 3 (Handcuff graphs). *A graph is a $H^k_{c_1,c_2}$-handcuff if it consists of two cycles of lengths c_1 and c_2, connected by a path of length k. We will call the path connecting the two cycles also a* chain *- with an odd (even) length an odd (even) chain. If $c_1 = c_2$ we call it a* symmetric handcuff*).*

Lemma 1. *Handcuff graphs with at least one even cycle are symmetry-compressible.*

Proof. This follows from Theorem 4, since we can apply the reflective symmetry to only that cycle, leaving the rest of handcuff fixed and thus getting a smaller representation of the entire graph. □

Because of this result we will focus mostly on handcuff graphs which contain only odd cycles.

Lemma 2. *Symmetric handcuffs $H^{2l}_{2k+1,2k+1}$ (with even chain) are symmetry-compressible, whereas $H^{2l+1}_{2k+1,2k+1}$ are not.*

Proof. We consider the symmetry of $H^c_{2k+1,2k+1}$ which maps one cycle onto another. In the case where $c = 2l$ the pivot of the symmetry is a node, but when $c = 2l+1$ the pivot is an edge.

- ($c = 2l$) The number of vertices $|V(H^c_{2k+1,2k+1})| = (4k+2)+(2l-1)$, the size of the symmetry π is therefore $|\pi| = \frac{(4k+2)+(2l-1)-1}{2} = \frac{2(2k+l)}{2} = 2k+l$ and the size of the residual graph under π is $|(H^{2l}_{2k+1,2k+1})^\pi| = 2k+1+l$. We see that we obtain a smaller representation:
$$|\pi| + |(H^{2l}_{2k+1,2k+1})^\pi| = 2k+l+2k+1+l = \\ = 4k+2l+1 < 2(2k+1)+2l = |H^{2l}_{2k+1,2k+1}|$$
- ($c = 2l+1$) In this case the size of the symmetry is $|\pi| = 2k+1+l$, and the size of the residual graph is $|(H^{2l+1}_{2k+1,2k+1})^\pi| = 2k+l+2$. This representation is the same in size as the original graph:
$$|\pi| + |(H^{2l+1}_{2k+1,2k+1})^\pi| = 2k+l+1+2k+l+2 = \\ = 4k+2l+3 \not< H^{2l+1}_{2k+1,2k+1} = 4k+2l+3$$

□

Definition 4 (Pair of handcuffs). *A pair of handcuffs is the graph $(H^c_{k,l}, H^c_{k,l})$.*

Lemma 3. *A pair of handcuffs is symmetry-compressible.*

Proof. This follows directly from Theorem 2, since $|H^c_{k,l}| > |V(H^c_{k,l})|$. □

Definition 5 (Minimal $\mathcal{SC}$ graph). *A graph $G \in \mathcal{SC}$, π being its compressing symmetry, such that $\forall e \in G : \overline{\pi}(e) \neq e$, is called a* minimal $\mathcal{SC}$ graph.

In the following analysis of $\mathcal{SSC}$ graphs we will focus mostly on the subgraphs that are minimal $\mathcal{SC}$ graphs because of the following lemma.

LEMMA 4. *All $G \in \mathcal{SC}$ contain a minimal $\mathcal{SC}$ subgraph.*

PROOF. Let $G \in \mathcal{SC}$, π its compressing symmetry, and $F \subseteq G$ such that $\forall e \in F : \overline{\pi}(e) = e$ - the edges fixed by symmetry π. We can show that the minimal $\mathcal{SC}$ subgraph is simply $H = G \setminus F$. Notice that by removing edges that are fixed by the symmetry π the graph $G \setminus F$ remains symmetric under the symmetry π. Notice also that the edges not moved by π are also present in G^π. So since we know that

$$|\pi| + |G^\pi| < |G| \Longrightarrow |\pi| + |G^\pi \setminus F| < |G \setminus F|.$$

□

In the following lemma we will require to identify graphs without even cycles. The most useful property of these graphs (for our application) is the following lemma.

LEMMA 5. *A graph has no even cycles if and only if each block in its block tree decomposition is either an odd cycle or K_2.*

The detailed proof of this lemma can be found in [1].

LEMMA 6. *If a graph has no even cycles and no pairs of handcuffs then for any subgraph $G' = G_1 \cup G_2, G1, G_2 \subset G, G_1 \cap G_2 = \emptyset$ there is no compressing symmetry $\pi \in Aut(G')$ that maps G_1 onto G_2.*

PROOF. Let us assume the opposite, i.e., G has two disjoint subgraphs $G_1, G_2 \subset G$ with a compressing symmetry π mapping G_1 onto G_2. From Theorem 2 we know that for two components, the necessary condition to be in $\mathcal{SC}$ is that $|V(G_1)| < |G_1|$. For this inequality to hold, G_1 needs at least two more edges than a tree would have. When adding edges to a tree, cycles are created, in this case at least two. And these two cycles have to be disjoint, otherwise an even cycle is created (which we assumed is absent in this graph). But if there are two disjoint cycles in a connected component, they are connected by a chain, forming a handcuff subgraph. And an isomorphic handcuff is also present in G_2, together forming a handcuff pair (which we also assumed is absent from the graph). □

Finally, we join these lemmas to identify three graph types that characterize the class $\mathcal{SSC}$.

THEOREM 5. *If a graph G has no even cycles C_{2i}, no handcuff pairs $(H^c_{k,l}, H^c_{k,l})$, and no symmetric handcuffs with even chain $H^{2i}_{k,k}$, then $G \notin \mathcal{SSC}$.*

PROOF. From the previous lemma we already know that if there is no even cycles and no handcuff pairs, then any minimal $\mathcal{SC}$ subgraph has to be connected. We prove that no such connected subgraph can exists, again by contradiction. Let $H \subseteq G$ be a connected minimal $\mathcal{SC}$ subgraph. Remember that H can be viewed as a tree where some vertexes can be substituted by odd cycles (see Lemma 5). We also know H is not a tree, so it has at least one cycle. Let us examine two possibilities:

- (H has only one cycle) The symmetry π has to map this one cycle onto itself. And since this is a minimal $\mathcal{SC}$ graph, all edges are moved by π, therefore the only possible symmetry is the rotational symmetry (the reflective symmetry would leave one edge fixed). H must be an odd cycle C_k, each vertex of this cycle having a tree T attached to it. The entire size fo the graph is $k + k|T|$. The size of the symmetry $|\pi| = |T|(k-1)+k-1$ and the size of $|H^\pi| = |T|+1$. So $|\pi|+|H^\pi| = |T|(k-1)+k-1+|T|+1 = k|T|+k \nless |H|$.
- (H has at least two cycles) Because of the aforementioned structure of H, only one cycle can be mapped onto itself by π. So at least one cycle must be mapped into another cycle, i.e., there are at least two cycles with the same size (lets call them C_1, C_2). Since H is connected these two cycles form a symmetric handcuff. And now we can show that the chain in this handcuff is of even length. There are again two possibilities for the pivot of the symmetry π, either the pivot is a 1) single vertex, or 2) an odd cycle. In case of 1), the distance of C_1 and C_2 has to be equal, therefore the length of the entire chain is even. In case of 2), the path between C_1 and C_2 passes over an odd cycle. We can make the chain either of odd length or even length by choosing one part or the other part of the cycle. So a symmetric handcuff with even chain is certainly a subgraph, even though we assumed it is not.

□

5. ALGORITHM

In this section we provide further details of the algorithm that follows from the forbidden graph characterization that we proved in the previous section.

LEMMA 7. *Checking if a graph contains an even cycle can be done in polynomial time.*

PROOF. From [2], we know that a block decomposition can be done in linear time. When we have a block decomposition, we check every block. If the block is anything other than K_2 or C_{2k+1}, the graph contains an even cycle $\longrightarrow$ the graph is $\in \mathcal{SSC}$. It is trivial to check if the block is only one edge or C_{2k+1} - for the latter the number of vertices and edges has to be the same and odd. □

LEMMA 8. *Every connected graph G, having ≥ 3 edge disjoint k cycles, contains a symmetric handcuff subgraph, i.e., $G \in \mathcal{SSC}$.*

Proof. We will prove this for the case when we have 3 disjoint cycles, for more it trivially follows.

Lets call the three cycles C_1, C_2, C_3. Since they are in a connected graph, there is a path between each pair of these cycles. Now we need to show that there is at least one even chain between a pair of cycles. There are two cases:
1) C_2 is not on the path between C_1 and C_3.
2) C_2 is on the path between C_1 and C_3.
When 1), there are two options to traverse C_2 when going from C_1 to C_3, i.e., an odd path across the cycle and an even path across the cycle. Which means we can always construct an even path between C_1 and C_3. When 2), the three cycles have a mutual junction (let us call this vertex j) of their chains. Let us denote the distances from the cycles to this junction as p_1, p_2, and p_3 respectively. If p_1 is odd, then if either p_2 or p_3 are odd we had an even path between C_1 and C_2 or C_3. But if both p_2 and p_3 are even, then the path between C_2 and C_3 is even. A very similar argument holds if p_1 is even. □

Theorem 6. *In graphs containing no even cycles, checking if they contain symmetric handcuffs can be done in polynomial time.*

Proof. From the above lemma it follows that we need to focus only on the case where there are exactly two cycles with the same cardinality. In the block decomposition tree, there is a unique path between these two cycles. If this path passes a block containing a cycle, then we have a symmetric handcuff. This follows from the fact that we have two choices of how to pass the cycle. One is of even length and one is of odd length. As a consequence, there is always a path of even length between the two cycles, making the handcuff symmetric. If the unique path in the decomposition tree passes no cycle, then it is unique also in the original graph, and it is trivial to check the parity of its length. □

Now we define an auxiliary problem, which will be used for recognizing our last family of forbidden graphs, i.e. handcuff pairs.

Definition 6. *Pair choice problem We define a problem where a sequences of pairs is given:* $a = \binom{a_1^0}{a_1^1}\binom{a_2^0}{a_2^1}\cdots\binom{a_n^0}{a_n^1}$ *and an integer* K.

The decision problem is whether a binary vector exists: $x \in \{0,1\}^n$ *such that* $\sum_{i=1}^n a_i^{x_i} = K$?

We can show that the problem of finding handcuff pairs in a graph can be reduced to the pair choice problem in polynomial time. The Pair-choice problem is an NP-complete problem, since the subset-sum problem can easily be reduced to the pair choice problem. But we can show how the limited versions of the pair choice problem can be solved in polynomial time with dynamic programming, which can be succinctly expressed with the following Bellmann equations:

$$s(0,0) = 1,$$
$$s(i,k) = s(i-1, k-a_i^0) \vee s(i-1, k-a_i^1),$$

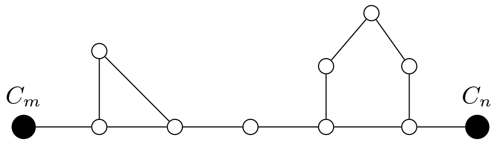


Figure 1: An example of a path between two cycles (the black vertices). To check if, e.g. there is a chain of size 7, we transform this subgraph into pairs $\binom{5}{4}, \binom{5}{2}$.

where $s(i,k)$ denotes the solution if only the first i pairs are taken into account, and the goal sum is k. This algorithm is polynomial ($O(nK)$), if the goal sum K is polynomial in n. We can demonstrate that the problem of identifying handcuff pairs can be reduced to an instance of the pair choice problem, which can be solved in polynomial-time using the above dynamic programming.

Theorem 7. *In graphs containing no even cycles, and no symmetric handcuffs checking if they contain handcuff pairs* $(H^c_{k,l}, H^c_{k,l})$ *can be done in polynomial time.*

Proof. In the tree of cycles, which we have when there are no even cycles in the graph, there can be only $O(n)$ cycles. For each pair of pairs of cycles, i.e. $(C_k, C_l), (C_k, C_l)$, we check all the possible distances between the cycles in a pair. This can be done by transforming every pair of cycles into a pair choice problem in the following way. First, notice there is a unique path between these cycles in the block decomposition. For each block that is also a cycle in the original graph, we create a pair, which represents the length of two possible paths across the cycle. An example of this reduction can be seen in Figure 1. In this way we can check all possible length paths between the start and end cycle. If the pars have a distance that coincides, than we have found a handcuff pair. □

6. CONCLUSIONS

This article answers an open question about recognizing a special kind of graphs, namely the subgraph symmetry compressible graphs. We showed that this problem is solvable in polynomial time, which is also practically useful information since it can be used in compression algorithms. A remaining open question is the time complexity of $\mathcal{SC}$ graphs. Since this problem is more closely related to the graph isomorphism problem, we believe it is at least GI-hard. In our future work we will focus on the above open question and on developing practical compression algorithms based on the described concepts of $\mathcal{SC}$ and $\mathcal{SSC}$ graphs.

7. REFERENCES

[1] J. G. Conlon. Even cycles in graphs. *Journal of Graph Theory*, 45(3):163–223, 2004.
[2] J. Hopcroft and R. Tarjan. Algorithm 447: Efficient algorithms for graph manipulation. *Commun. ACM*, 16(6):372–378, June 1973.
[3] U. Čibej and J. Mihelič. Two classes of graphs with symmetries allowing a compact representation. In *Informatics, 2019. Proceedings.* IEEE, 2019.

PSEUDORANDOM NUMBER GENERATORS BASED ON COMPOSITIONS OF AUTOMATA

Pál Dömösi
Faculty of Informatics
University of Debrecen
H-4028 Debrecen, Kassai út 26, Hungary
and Institute of Mathematics and Informatics
University of Nyíregyháza
H-4400 Nyíregyháza, Sóstói út 31/B, Hungary
domosi@unideb.hu

József Gáll
Faculty of Informatics
University of Debrecen
H-4028 Debrecen, Kassai út 26, Hungary
gall.jozsef@inf.unideb.hu

Géza Horváth
Faculty of Informatics
University of Debrecen
H-4028 Debrecen, Kassai út 26, Hungary
horvath.geza@inf.unideb.hu

Norbert Tihanyi
Faculty of Informatics
Eötvös Loránd University
H-1117 Budapest, Pázmány Péter sétány 1/C, Hungary
tihanyi.pgp@gmail.com

ABSTRACT

This paper is devoted to propose a novel PRNG based on compositions (temporal products of special Gluškov products) of abstract automata. Its utility shall be shown through a simple example. However, several questions are subject of future work, such as the analysis of further properties of the PRNG, as well as related statistical testing.

Keywords
pseudorandom number generator, automata network, products of automata

1. INTRODUCTION

Certain cryptographical applications of abstract automata compositions has already been considered in [1, 2, 3]. On the line of this research, we propose a new pseudorandom number generator based on compositions of abstract automata.

For all notions and notation not defined here we refer to the monographs [4, 5, 6, 9, 10].

2. PRELIMINARIES

Let us start with some standard concepts and notations. By an (abstract) *pseudorandom number generator* we mean a system $\mathcal{PRNG} = (K, S, s_0, f, U, g)$, where K, S, U are finite nonempty sets, the so-called *key space*, *state space*, and *output space*, respectively, $s_0 \in S$ is the *seed*, $f : S \to S$ is the *state transition function*, and $g : K \times S \to U$ is the *output function*. We will say that the *pseudorandom sequence* $u_1, u_2, \ldots, (u_1, u_2, \ldots \in U)$ generated by the $\mathcal{PRNG}$ if we have $u_n = g(k, s_n)$, where $s_n = f(s_{n-1})$ $(n \in \{1, 2 \ldots\})$.

An *automaton* will be meant to be a deterministic finite automaton without outputs. In more details, an automaton is an algebraic structure $\mathcal{A} = (A, \Sigma, \delta)$ consisting of the nonempty and finite *state set* A, the nonempty and finite *input set* Σ, and a *transition function* $\delta : A \times \Sigma \to A$. The elements of the state set are the *states*, whereas the elements of the input set are the *input signals*.

The transition matrix of an automaton is a matrix with rows corresponding to each input and columns corresponding to each state. Given an entry corresponding to a row indicated by an input $x \in \Sigma$ and a column indicated by a state $a \in A$ the state $\delta(a, x)$ is put into the entry. If all rows of the transition matrix are permutations of the state set then we call it a *permutation automaton*.

3. THEORETICAL BACKGROUND

Next we recall some definitions. (See also [1].)

Let $\mathcal{A}_i = (A_i, \Sigma_i, \delta_i)$ be automata where $i \in \{1, \ldots, n\}$, $n \geq 1$. Take a finite nonvoid set Σ and a *feedback function* $\varphi_i : A_1 \times \cdots \times A_n \times \Sigma \to \Sigma_i$ for every $i \in \{1, \ldots, n\}$. The *Gluškov-type product* [7] of the automata $\mathcal{A}_i$ with respect to the feedback functions φ_i $(i \in \{1, \ldots, n\})$ is defined to be the automaton $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n(\Sigma, (\varphi_1, \ldots, \varphi_n))$ with state set $A = A_1 \times \cdots \times A_n$, input set Σ, transition function δ, where the latter one is given by $\delta((a_1, \ldots, a_n), x) = (\delta_1(a_1, \varphi_1(a_1, \ldots, a_n, x)), \ldots, \delta_n(a_n, \varphi_n(a_1, \ldots, a_n, x)))$ for all

$(a_1, \ldots, a_n) \in A$ and $x \in \Sigma$. In particular, if $\mathcal{A}_1 = \ldots = \mathcal{A}_n$ then we say that $\mathcal{A}$ is a *Gluškov-type power.*

Let $\mathcal{A}_t = (A, \Sigma_t, \delta_t)$, $t = 1, 2$, be automata having a common state set A. Take a finite nonvoid set Σ and a mapping φ of Σ into $\Sigma_1 \times \Sigma_2$. Then the automaton $\mathcal{A} = (A, \Sigma, \delta)$ is a *temporal product* (t-product, see [8]) of $\mathcal{A}_1$ and $\mathcal{A}_2$ with respect to Σ and φ if for any $a \in A$ and $x \in \Sigma$, $\delta(a, x) = \delta_2(\delta_1(a, x_1), x_2)$, where $(x_1, x_2) = \varphi(x)$. The concept of temporal product is generalized in a natural way to an arbitrary finite family of $n > 0$ automata $\mathcal{A}_t$ $(t = 1, \ldots, n)$, all having the same state set A, such that for any mapping $\varphi : \Sigma \to \prod_{t=1}^{n} \Sigma_t$, one defines $\delta(a, x) = \delta_n(\cdots \delta_2(\delta_1(a, x_1), x_2), \cdots, x_n)$ when $\varphi(x) = (x_1, \ldots, x_n)$. In particular, a temporal product of automata with a single factor is just a (one-to-many) relabeling of the input letters of some input-subautomaton of its factor.

Given a function $f : X_1 \times \cdots \times X_n \to Y$, we say that f is *really independent of its i-th variable* if for every pair $(x_1, \ldots, x_n), (x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_n) \in X_1 \times \cdots \times X_n$ we have $f(x_1, \ldots, x_n) = f(x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_n)$. Otherwise we say that f *really depends on its i-th variable.*

A (finite) *directed graph* (or, in short, a *digraph*) $\mathcal{D} = (V, E)$ (of order $n > 0$) is a pair consisting of sets of *vertices* $V = \{v_1, \ldots, v_n\}$ and *edges* $E \subseteq V \times V$. Elements of V are sometimes called *nodes*. If $|V| = n$ then we also say that $\mathcal{D}$ is a digraph of order n. In what follows we will assume that V is an ordered set of integers $1, \ldots n$ for some positive integer n. Given a digraph $\mathcal{D} = (V, E)$ we say that the above defined Gluškov product is a $\mathcal{D}$-product if for every pair $i, j \in \{1, \ldots, n\}$ we have that $(i, j) \notin E$ implies that the feedback function φ_i is really independent of its j-th variable. Now assume that Σ is the set of all binary strings with a given length $\ell > 0$, n is a positive integer power of 2, $\mathcal{A}_1 = (\Sigma, \Sigma \times \Sigma, \delta_{\mathcal{A}_1})$ is a permutation automaton such that for every $a, x, x', y, y' \in \Sigma$, we have $\delta_{\mathcal{A}_1}(a, (x, y)) \neq \delta_{\mathcal{A}_1}(a(x', y))$, $\delta_{\mathcal{A}_1}(a, (x, y)) \neq \delta_{\mathcal{A}_1}(a(x, y'))$, and let $\mathcal{A}_i = (\Sigma, \Sigma \times \Sigma, \delta_{\mathcal{A}_i}), i = 2, \ldots, n$ be copies of $\mathcal{A}_1$. Consider the following simple bipartite digraphs:

$\mathcal{D}_1 = (\{1, \ldots, n\}, \{(n/2+1, 1), (n/2+2, 2), \ldots, (n, n/2)\})$,
$\mathcal{D}_2 = (\{1, \ldots, n\}, \{(n/4+1, 1), (n/4+2, 2), \ldots, (n/2, n/4)$, $(3n/4+1, n/2+1), (3n/4+2, n/2+2), \ldots, (n, 3n/4)\}), \ldots$,
$\mathcal{D}_{log_2 n-1} = (\{1, \ldots, n\}, \{(3, 1), (4, 2), (7, 5), (8, 6), \ldots,$ $(n-1, n-3), (n, n-2)\})$,
$\mathcal{D}_{log_2 n} = (\{1, \ldots, n\}, \{(2, 1), (4, 3), \ldots, (n, n-1)\})$,
$\mathcal{D}_{log_2 n+1} = \mathcal{D}_1, \ldots, \mathcal{D}_{2log_2 n} = \mathcal{D}_{log_2 n}$.

For every digraph $\mathcal{D} = (V, E)$ with $\mathcal{D} \in \{\mathcal{D}_1, \ldots, \mathcal{D}_{2log_2 n}\}$ let V_1 be the set of all incoming edges and let V_2 be the set of all outgoing edges, and define furthermore the Gluškov-type product (called *two-phase $\mathcal{D}$-product*)
$\mathcal{A}_{\mathcal{D}} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n(\Sigma^n, (\varphi_1, \ldots, \varphi_n))$ of $\mathcal{A}_1, \ldots, \mathcal{A}_n$ so that for every $(a_1, \ldots, a_n), (x_1, \ldots, x_n) \in \Sigma^n, i \in \{1, \ldots, n\}$ we have $\varphi_i(a_1, \ldots, a_n, (x_1, \ldots, x_n)) = (a_j \oplus x_j, x_i)$ if $(j, i) \in V_1$ and $a_j \oplus x_j$ is the bitwise addition modulo 2 of a_j and x_j, $\varphi_i(a_1, \ldots, a_n, (x_1, \ldots, x_n)) = (a'_j \oplus x_j, x_i)$ if $(j, i) \in V_2$, where a'_j denotes the state into which $\varphi_j(a_1, \ldots, a_n, (x_1, \ldots, x_n))$ takes the automaton from its state a_j, and $a'_j \oplus x_j$ is the bitwise addition modulo 2 of a'_j and x_j. [1]

Let $\mathcal{B} = (\Sigma^n, (\Sigma^n)^{2log_2 n}, \delta_{\mathcal{B}})$ be the temporal product of $\mathcal{A}_{\mathcal{D}_1}, \ldots, \mathcal{A}_{\mathcal{D}_{2log_2 n}}$ with respect to $(\Sigma^n)^{2log_2 n}$ and the identity map $\varphi : (\Sigma^n)^{2log_2 n} \to (\Sigma^n)^{2log_2 n}$. We say that $\mathcal{B}$ is a *key-automaton* with respect to $\mathcal{A}_1, \ldots, \mathcal{A}_n$.[2] Obviously, $\mathcal{B}$ is unambigously defined by the transition matrix of $\mathcal{A}_1$.

Theorem. *Every key automaton transition function can be applied as an output function of a pseudorandom number generator.*

Proof. Let $\mathcal{B} = (\Sigma^n, (\Sigma^n)^{2log_2 n}, \delta_{\mathcal{B}})$ be an above defined key automaton. Moreover, let $f : \Sigma^n \to \Sigma^n$ be a bijective function. Consider two random words $u_0, v_0 \in \Sigma^n$. For every positive integer k let $u_k = f(u_{k-1})$. Define the output function g such that for every positive integer k, we have $g(v_0, u_k) = \delta_{\mathcal{B}}(u_k, v_0^{2log_2 n})$. Then we can get the system $\mathcal{PRNG} = (K, S, s_0, f, U, g)$, where $K(= \{v_0\})$ is a singleton set, $S = U = \Sigma^n$ and $s_0 = u_0$.

This completes the proof.

Remarks. It is shown in [1] that every key automaton is a permutation automaton. In other words, for every triplet $u_1, u_2, x \in \Sigma^n$, $u_1 \neq u_2$ implies $\delta_{\mathcal{B}}(u_1, x^{2log_2 n}) \neq \delta_{\mathcal{B}}(u_2, x^{2log_2 n})$. Therefore, for every seed $u_0 \in \Sigma^n$ and key $v_0 \in \Sigma^n$ the length of the period of $\mathcal{PRNG}$ (i.e. the minimal nonnegative integer k for which u_0 and $g(u_k, v_0)$ coincide) is equal to the minimal number m for which $u_0 = f(u_m)$. Consequently, if f generates a full cycle –i.e. $\{u_k \mid u_k = f(u_{k-1}, k \in \{1, 2, \ldots\}\} = \Sigma^n$– then, of course, the length of the period of $\mathcal{PRNG}$ does not depend on u_0 or v_0.

In [1] it is also shown that a small change in either the state blocks or the input blocks results in a significant change in the next state of the state transitions. In other words, for every triplet $u'_0, u''_0, v_0 \in \Sigma^n$, having $u'_0 \neq u''_0$ results significant change in $\delta_{\mathcal{B}}(u'_0, v_0^{2log_2 n})$ and $\delta(u''_0, v_0^{2log_2 n})$. On the other hand, if f generates a full cycle and Σ^n is not a singleton then for every pair $u'_0, u''_0 \in \Sigma^n$ we have that $u'_0 \neq u''_0$ obviously implies $u'_k \neq u''_k$ whenever $u'_k = f(u'_{k-1}), u''_k = f(u''_{k-1}), k \in \{1, 2, \ldots\}$ But then the pairs $\delta_{\mathcal{B}}(u'_k, v_0^{2log_2 n})$ and $\delta_{\mathcal{B}}(u''_k, v_0^{2log_2 n})$ with $u'_k = f(u'_{k-1}), u''_k = f(u''_{k-1}), k \in \{1, 2, \ldots\}$ should also have significant differences.

4. EXAMPLE

Consider the following transition table of an automaton $\mathcal{A} = (\{0, 1\}, \{0, 1\}^2, \delta)$:

δ	0	1
00	0	1
01	1	0
10	1	0
11	0	1

[1]We note that for every $j \in V_2$ there exists a unique $i \in V_1$ with $(j, i) \in E$, and conversely, for every $i \in V_1$ there exists a unique $j \in V_2$ with $(j, i) \in E$. Therefore, all of $\varphi_1, \ldots, \varphi_n$ are well-defined.

[2]Recall that n should be a power of 2.

Let $n = 4$ and assume that all of $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4$ coincide with $\mathcal{A}$. Then $log_2 n)(= log_2 4) = 2$ and thus

$\mathcal{D}_1 = (\{1, \ldots, 4\}, \{(3, 1), (4, 2)\}),$

$\mathcal{D}_2 = (\{1, \ldots, 4\}, \{(2, 1), (4, 3)\}.$

Let $v_0 = (1, 0, 1, 0)$ be a fixed input signal of the key automaton $\mathcal{B}$ which is the temporal product of $\mathcal{A}_{\mathcal{D}_1}$ and $\mathcal{A}_{\mathcal{D}_2}$. Assume that $\mathcal{B}$ is in the state $u_k = (0, 1, 1, 0)$. Thus we have $(a_1, a_2, a_3, a_4) = (0, 1, 1, 0)$, $(x_1, x_2, x_3, x_4) = (1, 0, 1, 0)$.

Denote by $\varphi_i, a_i, a'_i, x_i, i \in \{1, 2, 3, 4\}$ the feedback functions, the state components, the next state components, and the input components of $\mathcal{A}_{\mathcal{D}_1}$, respectively. Then $\varphi_1((0, 1, 1, 0), 1, 0, 1, 0) = (a_3 \oplus x_3, x_1) = (1 \oplus 1, 1) = (1, 1)$, $\varphi_2((0, 1, 1, 0), 1, 0, 1, 0) = (a_4 \oplus x_4, x_2) = (0 \oplus 0, 1) = (0, 1)$, moreover $\delta(0, (1, 1)) = 0(= a'_1)$ and $\delta(1, (0, 1)) = 0(= a'_2))$ and thus $\varphi_3((0, 1, 1, 0), 1, 0, 1, 0) = (a'_1 \oplus x_1, x_3) = (0 \oplus 1, 1) = (1, 1)$, $\varphi_4((0, 1, 1, 0), 1, 0, 1, 0) = (a'_2 \oplus x_2, x_4) = (0 \oplus 0, 0) = (0, 0)$. Hence, $\delta(1, (1, 1)) = 1(= a'_3)$ and $\delta(0, (0, 0)) = 0(= a'_4)$.

Next we denote by $\varphi_i, a_i, a'_i, x_i, i \in \{1, 2, 3, 4\}$ the feedback functions, the state components, the next state components, and the input components of $\mathcal{A}_{\mathcal{D}_2}$, respectively. Recall that (a_1, a_2, a_3, a_4) coincides with the new state of $\mathcal{A}_{\mathcal{D}_1}$. Thus $(a'_1, a'_2, a'_3, a'_4) = (0, 1, 1, 0)$, and again $(x_1, x_2, x_3, x_4) = (1, 0, 1, 0)$.

Then $\varphi_1((0, 0, 1, 0), 1, 0, 1, 0) = (a_2 \oplus x_2, x_1) = (0 \oplus 0, 1) = (0, 1)$, $\varphi_3((0, 0, 1, 0), 1, 0, 1, 0) = (a_4 \oplus x_4, x_3) = (0 \oplus 0, 1) = (0, 1)$, moreover $\delta(0, (0, 1)) = 1(= a'_1)$ and $\delta(0, (0, 1)) = 0(= a'_3))$, and thus $\varphi_2((0, 0, 1, 0), 1, 0, 1, 0) = (a'_1 \oplus x_1, x_2) = (1 \oplus 1, 0) = (1, 0)$, $\varphi_4((0, 0, 1, 0)1, 0, 1, 0) = (a'_3 \oplus x_3, x_4) = (0 \oplus 1, 0) = (1, 0)$. Hence $\delta(0, (1, 0)) = 1(= a'_2)$ and $\delta(0, (1, 0)) = 1(= a'_4)$.

Now let $(a_1, a_2, a_3, a_4) = (1, 1, 0, 1)$ and again $(x_1, x_2, x_3, x_4) = (1, 0, 1, 0)$. Repeating the above procedure we get the output $(a'_1, a'_2, a'_3, a'_4) = g((a_1, a_2, a_3, a_4), (a_1, a_2, a_3, a_4))$ in the following way.

Then $\varphi_1((1, 1, 0, 1), 1, 0, 1, 0) = (a_3 \oplus x_3, x_1) = (0 \oplus 1, 1) = (1, 1)$, $\varphi_2((1, 1, 0, 1), 1, 0, 1, 0) = (a_4 \oplus x_4, x_2) = (1 \oplus 0, 1) = (1, 1)$, moreover $\delta(1, (1, 1)) = 1(= a'_1)$ and $\delta(1, (1, 1)) = 1(= a'_2))$, and thus $\varphi_3((1, 1, 0, 1), 1, 0, 1, 0) = (a'_1 \oplus x_1, x_3) = (1 \oplus 1, 1) = (0, 1)$, $\varphi_4((1, 1, 0, 1), 1, 0, 1, 0) = (a'_2 \oplus x_2, x_4) = (1 \oplus 0, 0) = (1, 0)$. Hence $\delta(0, (0, 1)) = 1(= a'_3)$ and $\delta(1, (1, 0)) = 0(= a'_4)$.

Now we have $(a'_1, a'_2, a'_3, a'_4) = (1, 1, 1, 0)$, and again $(x_1, x_2, x_3, x_4) = (1, 0, 1, 0)$. Then $\varphi_1((1, 1, 1, 0), 1, 0, 1, 0) = (a_2 \oplus x_2, x_1) = (1 \oplus 0, 1) = (1, 1)$, $\varphi_3((1, 1, 1, 0), 1, 0, 1, 0) = (a_4 \oplus x_4, x_3) = (0 \oplus 0, 1) = (0, 1)$, moreover $\delta(1, (1, 1)) = 1(= a'_1)$ and $\delta(1, (0, 1)) = 0(= a'_3))$, and thus $\varphi_2((1, 1, 1, 1, 0), 1, 0, 1, 0) = (a'_1 \oplus x_1, x_2) = (1 \oplus 1, 0) = (0, 0)$, $\varphi_4((1, 1, 1, 0)1, 0, 1, 0) = (a'_3 \oplus x_3, x_4) = (0 \oplus 1, 0) = (1, 0)$. Thus $\delta(1, (0, 0)) = 1(= a'_2)$ and $\delta(0, (1, 0)) = 1(= a'_4)$.

Hence the actual pseudorandom output is $g((a_1, a_2, a_3, a_4), (x_1, x_2, x_3, x_4)) = g((0, 1, 1, 0), (1, 0, 1, 0)) = (1, 1, 0, 1)$.

5. SOME TECHNICAL COMMENTS

Using the above mentioned parameters with 256 possible states (1 byte long states) we need 16 automata having a transition matrix $2^{16} = 65536$ lines and $2^8 = 256$ columns. Each cell of the automaton contains 1 byte long data (one state). The size of the matrix is 16 megabytes and the number of the possible matrixes are $256!^{65536}$, where the exclamation mark means the factorial operation.

6. CONCLUSION AND FUTURE WORK

This paper is devoted to propose a novel PRNG based on the compositions (temporal products of special Gluškov products) of abstract automata. Through a simple example we have shown its utility. However, several questions are still open for future work: a serious security analysis, evaluations regarding the randomness, a rigorous machine-independent investigation and discussion over how the PRNG proposed in this article compares to the ones in the literature. Of course, the interesting case is the one where the state transition function has a low computational complexity. (For example, when it is a linear congruential generator.)

7. REFERENCES

[1] Dömösi P., Gáll, J., Horváth, G., Tihanyi, N. Some Remarks and Tests on the DH1 Cryptosystem Based on Automata Compositions. *Informatica* (Slovenia), vol 43, 2 (2019), 199-207.

[2] Dömösi, P. and Horváth, G. A novel cryptosystem based on abstract automata and latin cubes. *Studia Scientiarum Mathematicarum Hungarica*, 52(2):221–232, 2015.

[3] Dömösi, P. and Horváth, G. A novel cryptosystem based on Gluškov product of automata. *Acta Cybernetica*, 22:359–371, 2015.

[4] Dömösi, P. and Nehaniv, C. L. Algebraic theory of automata networks. An introduction. *SIAM Monographs on Discrete Mathematics and Applications*, 11. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2005.

[5] Gécseg F. Products of Automata. *EATCS Monogr. Theoret. Comput. Sci.* 7, Springer-Verlag, Berlin, , New York, Tokyo, 1986.

[6] Gécseg, F. and Peák, I. Algebraic theory of automata. *Disquisitiones Mathematicae Hungaricae*, 2, Akadémiai Kiadó Budapest (Hungary), 1972.

[7] Gluskov, V. M. Abstract theory of automata (in Russian). *Uspekhi Mat. Nauk*, 16 (101) (1961), 3-62; correction, ibid., 17 (104) (1962), 270.

[8] Ivanov, G. I. Temporal representation of controlled automata. (in Russian) *IZV AN SSSR. Technicheskaia Kibernetika*, 6 (1973), 106-113.

[9] Hopcroft, J. E., Motwani, R., and Ullman, J D. Introduction to Automata Theory (second edition). *Addison-Wesley Series in Computer Science*, Addison-Wesley Co., Reading, MA, 2001.

[10] Menezes, A. J., Oorschot, P. C., Vanstone, S. A. Handbook of Applied Cryptography. *CRC Press Series on Discrete Mathematics and Its Applications*, CRC Press LLC, Boca Raton, FL, USA, 1996, 2001, 2008.

Simulation Framework for Evaluating Production Networks

Péter Egri József Váncza Ádám Szaller Judit Monostori
Centre of Excellence in Production Informatics and Control
Institute for Computer Science and Control
{egri, vancza, szaller.adam, mesterne.monostori.judit}@sztaki.hu

ABSTRACT

The manuscript presents the ongoing development of an agent-based production network simulation framework. The simulation is intended to analyze the high level (strategic and tactical) planning problems decomposed into simple subproblems, similarly to the practical approach applied by the Enterprise Resource Planning (ERP) systems. The background, the goals and the design of the framework are described, and some preliminary experiments with the current phase of the development are shown.

Keywords

Logistics optimization, agent-based simulation, robustness

1. INTRODUCTION

In recent years the shortening product life cycles and the increasing product variety have led to complex production networks with dynamic structure, fluctuating demand, embedded in volatile environments. Handling such complex and uncertain problems with exact mathematical optimization models is time-consuming and usually impractical.

There are two main difficulties in creating practical planning models. On the one hand, some parameters or dynamics are unknown or uncertain, therefore they are only estimated or approximated. For example, the demand curve usually assumes a simple relationship between the price and the demand, and completely disregards other important factors that influence the market, such as sudden changes in customers' preferences. On the other hand, if a model contains too many details, it can result in an *overfitted* solution. In this case the plan might be optimal considering fixed parameters, however, any change in the environment—e.g., a late supply or inappropriate quality—can cause a major change in the execution. Due to these difficulties, the realization usually diverges from the plan or the forecast.

In the industrial practice, commonly the basic planning algorithms that are built into the ERP systems are used. These general algorithms neglect several details of the problem, but usually result in plans that have more room for adaptation and are more flexible to changes. Furthermore, they are readily available, do not require additional software and interface development, and frequently provide comparable results to specialized optimization algorithms [5]. For example, the scheduling algorithm of SAP APO computes the order finish date simply by adding the production time to the start time, where the production time is a sum of the setup time, of the processing time multiplied by the quantity and of the interoperation time [9]. This approach disregards the capacity and the load of the resources, as well as the possibilities of unexpected disturbances.

The goal of our current research is to develop a testbed for studying production networks in a simulated volatile environment. The desired characteristics of the simulation framework are to be general, modular and flexible. It should allow modeling dynamic production networks in uncertain environments, with a wide range of products, both mass produced and customized. The decision problems considered are focused on the strategic and tactical levels. Each node can apply different planning algorithms that are available in ERP systems. The performance of the network and the nodes should be evaluated according to multiple criteria, thus we are going to model various network footprints and strategies (see [6]). The first application of the developed framework is to study the fields of *resilience*, *pricing* and *trust* in production networks.

Resilience corresponds to balancing *robustness* and *agility* in supply chains [1]. Agility is the capability to react to changes, while robustness is resulted by a proactive strategy enabling to cope with turbulences without taking further actions. Monostori [7] introduced measures of structural and operational robustness of supply chains, and described a framework for evaluating robustness, complexity and efficiency. A supply chain simulation for evaluating robustness and coordination is presented in [2].

Two types of uncertainty are especially relevant in supply chains: *stochastic events* and *low probability high impact disruptions* [10]. The former ones can be forecasted based on historic data and/or expert knowledge. These include factors such as demand fluctuation, production and transportation times, as well as raw material and transportation prices. The disruptions, however, are rare, thus traditional forecast-

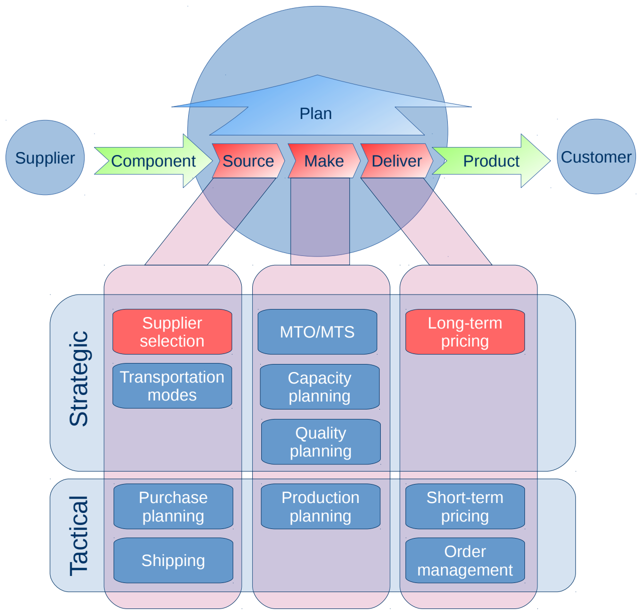


Figure 1: Decision problems at each node of the network.

ing techniques are inappropriate to handle them. These include events such as sudden disturbances at supply chain members, unexpected damages, and natural disasters.

Pricing is also important in influencing the market demand, and eventually, the profitability of the companies [11].

There are several approaches for collaboration in production networks decreasing the undesirable effects such as double marginalization or the bullwhip effect, see e.g., [12]. However, identifying the benefits of collaboration is still a challenge in supply chain management, and particularly in supply chain simulation [8]. Trust is a precondition of successful collaboration, but it is rarely considered formally in decision models, because it consists of a complex belief of dependability, competence and integrity. One of the few exceptions can be found in [4], where trust in a supplier is measured as its average order fill rate, i.e., the number of supplied goods divided by the number of ordered goods. The authors have observed that using trust based supplier selection, the robustness of the supply chain network increases.

2. THE SIMULATION FRAMEWORK

The simulation model is intended to be as general as possible. We consider a network consisting of nodes that are similar in the sense that each of them creates products, consumes components and has the same decision problems illustrated in Fig. 1. However, the specific products, components and applied decision algorithms can be different. This characterization of the nodes is based on the high level model of the Supply Chain Operations Reference (SCOR) and the supply chain planning matrix, see [3]. The dynamic nature of the network is also taken into consideration, i.e., nodes can enter and exit, choose different sources of materials, therefore changing the network structure.

As Fig. 1 shows, our model considers the higher strategic and tactical planning levels and does not include operational problems such as shop floor control. These long- or medium-term plans are more exposed to the uncertainties that are in the focus of our study. The strategic problems usually consider a one period planning horizon, oftentimes a year. This is then divided into shorter periods for the tactical decisions, where the horizon usually consists of multiple shorter periods, e.g., weeks. The main decisions considered in our model include capacity investment, supplier selection (including single and dual sourcing), transportation modes (e.g., air, water, land), Make-to-Stock (MTS) or Make-to-Order (MTO) production, pricing, quality control, order management, inventory control and procurement decisions. It is assumed that these decisions are made sequentially and not simultaneously, which is often the case in the practice. This is also true along the supply chains, where it is common to assume Stackelberg-games, i.e., when the leader decides first, then the follower reacts. The two strategic tasks indicated in the figure with red color are the ones we have started to implement and study first.

Most decision problems have the minimal cost or the maximal profit as their objective. But besides cost, there are usually multiple important criteria that are considered in practice, such as resource utilization. We consider three types of Key Performance Indicators (KPIs) that cover the most important aspects of the performance. The first type includes *financial* indicators, such as profit and total cost, which describe the economic sustainability. The second type is related to the *manufacturing efficiency*, e.g., the Overall Equipment Effectiveness (OEE). The last type measures *supply chain related* indicators, including service level, item fill rate, inventory turnover and lead time between order placement and delivery.

The description of the network is based on data generally available in Enterprise Information Systems (EIS). The first type of the data consists of information about the resources, i.e., the network nodes. These include for example the location of the nodes, their capabilities, costs and available transportation modes. The second type is related to the materials, including bills-of-materials (BOMs), demand forecasts, inventories and prices. The third type describes the process, such as the production times and costs. Finally, the fourth type is related to the operations, e.g., the realized demand or the purchase orders.

The simulation includes uncertainties in form of stochastic variables such as demand, component quality, production and transportation times, material and transportation prices. Besides, it also allows to generate sudden disturbances like perished shipments, resource outage and other unexpected events.

Fig. 2 shows an overview of the system architecture. The network model is given in an SQLite database which represents the different information systems containing the available data. The simulation model is automatically built, which then provides the run-time behavior of the network, including disturbances. The decision making functions are implemented separately, in a modular way. This enables the customization of the simulated system and also facilitates

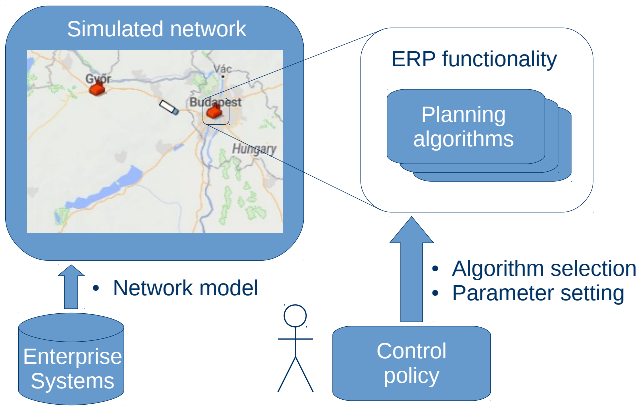


Figure 2: Overview of the architecture.

changing, analyzing and comparing different planning and optimization rules. This way it can be used to find trade-offs between different KPIs, such as cost and service level.

The simulations help to visualize and evaluate the consequences of the decisions in the network, and to analyze typical scenarios, such as a new product introduction. The simulation system is being developed with the AnyLogic tool [13], which provides the possibility of applying any optimization algorithms implemented in the Java language.

3. SUPPLIER SELECTION AND PRICING PROBLEMS

In the following the notations for the formal decision problems are introduced. The models are assumed to be deterministic, but in the simulation most of the parameters can be randomized in order to investigate the impact of the different kinds of uncertainties. Furthermore, since this study focuses on two strategic level decision problems, we omit the parts of the model that are not required for these tasks, such as the time-dependent prices and the transportation modes. We also simplify our trust model for this study.

Let N_i $(i = 1..n)$ denote the nodes in the network and ρ_{ij} is the distance between N_i and N_j. The transportation cost between N_i and N_j is $\rho_{ij}C^{(t)}$. The transportation mode and quality level (Q_i) are considered to be already given, since these optimization problems are ignored here.

The materials are denoted by M_k $(k = 1..m)$. The production portfolio is described by Y_{ik}, which equals 1 if N_i produces M_k, otherwise 0. The relationship between the materials is described by the BOMs: B_{kl} is the number of M_l directly required for producing one unit of M_k. The same material can be viewed as a product and as a component by different nodes of the supply network (see Fig. 1). Unit price of M_k at N_i (as the supplier) is P_{ik}. The $C^{(p)}_{ik}$ is the unit production cost of M_k at N_i. In this study we assume MTO production throughout the network, therefore we omit the input (components) and output (products) inventories from this description. The time required for the production of one unit of M_k is $T^{(p)}_k$.

For each required component one or more supplier(s) should be selected. Let Z_{ijk} denote the ratio of the component demand for M_k that N_j orders from N_i. The total demand for a component should be divided among its selected suppliers, i.e., $\forall j, k : \sum_{i=1}^{n} Z_{ijk} = 1$. This way a node can decide that a component should be supplied by only one supplier (single sourcing), two suppliers with 50%-50% share, or any other possibility. The set of the selected suppliers is called the *supplier basis* of the node. For each supplier in the basis a $C^{(b)}$ one time cost occurs that can represent the cost for building the connection between the nodes, e.g., sharing product designs or connecting data interfaces.

The demand of M_k at N_i at time t is modeled with the isoelastic function $D_{ikt} = D_k P_{ik}^{-r_k}$, where $r_k > 1$ is the price elasticity and D_k is the maximum demand of M_k.

The supplier selection is based on the cost of the purchase and the trust towards the suppliers. The cost consists of the distance-based transportation cost and the price paid for the components[1]. This latter assumes already known unit prices of the components, i.e., the suppliers should decide about the prices first. However, the demand for the components can only be estimated without the knowledge of any downstream pricing or supplier selection decisions. The trust is considered in a simplified way for this study: if the node does not trust in the suppliers, it chooses the dual sourcing strategy instead of the single one.

The pricing decision depends on whether the product has a market demand or used as a component for another product. In case of a market product, the profit—disregarding the constant transportation costs—is $D_{ikt}(P_{ik} - C^{(p)}_{ik} - C^{(a)})$, where $C^{(a)}$ denotes the total value of the consumed components determined by the previous supplier selection. Using the isoelastic demand function, the optimal price can be derived and is given by $P^*_{ik} = r_k(C^{(p)}_{ik} + C^{(a)})/(r_k - 1)$. In case of pricing a component, the demand should be estimated in the same way as for the supplier selection problem. Then the price is determined that provides a desired percent of profit rate considering the estimated demand, the production price, the total value of the components and the total transportation cost.

4. PRELIMINARY EXPERIMENTS

In the preliminary experimental study a simple network has been analyzed in order to evaluate the simulation framework. Only the supplier selection and the pricing decisions are included in the study, thus the other decisions are not implemented or only simple rules are applied, such as the lot-for-lot ordering policy. Five nodes are considered: one end product manufacturer and four component suppliers—two suppliers for both of the two components required for the product. Each material is produced only to orders, i.e., no inventories are included. The quality of the production in a node is considered to influence the production time: with probability Q_i the produced goods have acceptable quality, otherwise additional rework is needed increasing the production time.

[1]Note that in practice sometimes an even simpler rule is applied considering only the component prices.

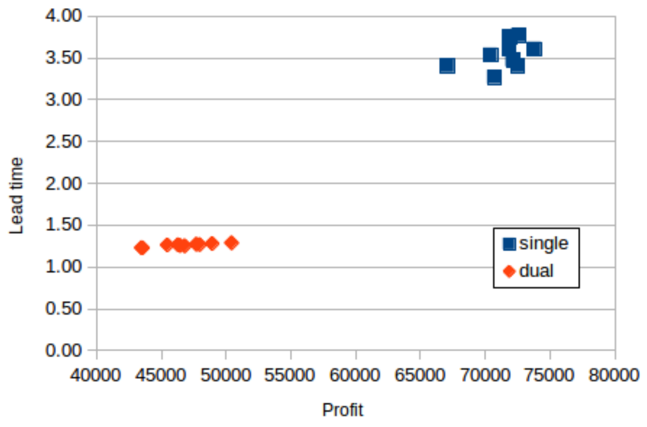


Figure 3: Performance using different sourcing strategies.

The trust is included in a straightforward way: the end product manufacturer either trusts the suppliers and has a single supplier for each component, or does not trust them and applies dual sourcing. In both cases the decision about the supplier basis depends on the estimated transportation and purchasing costs described in the previous section.

The KPIs considered are the average lead time—i.e., the time between receiving a customer order and satisfying it—and the total profit. Both indicators are computed during simulation runs over a one year horizon.

Fig. 3 illustrates the KPIs of the end product manufacturer during 20 runs, half of them using single, the other half dual sourcing strategies. The analysis shows the inversely proportional relationship between the costs and the lead times. Purchasing only from the most inexpensive suppliers results in lower costs, which leads to a lower product price, higher demand and eventually, higher profit. However, dual sourcing performs better regarding to the lead time: the lower component demand is further divided between the suppliers who work in parallel, thus the components are available more quickly reducing the lead time. The simulations support human decision makers to estimate the effects of their decisions on the KPIs, which is even more important when multiple complex decision problems are considered and the performance of the network is hard to be analyzed exactly.

5. CONCLUSION AND FUTURE WORK

The paper reports an ongoing work of developing a simulation framework for analyzing the robustness of production networks. The simulation model considers the common strategic and tactical decision problems at each node. Preliminary experiments are also demonstrated focusing on the supplier selection and the product pricing problems.

The next step of the development is to implement several basic decision making algorithms for each problem. The framework then will be used for evaluating these algorithms in different scenarios, e.g., new product introduction. Furthermore, by implementing simple contract types such as buyback or quantity discount, the effects of supply chain collaboration can be analyzed.

The simulation system will be also deployed at our experimental smart factory. That highly digitized production environment allows us to run simulations based on real data available from the Manufacturing Execution System (MES). The demonstration use case will enable analyzing the resilience and efficiency of the network consisting of the factory and its component suppliers.

6. ACKNOWLEDGMENTS

This research has been supported by the GINOP-2.3.2-15-2016-00002 and the H2020 project EPIC No. 739592 grants. The authors would like to thank to the anonymous reviewers for their comments.

7. REFERENCES

[1] C. F. Durach, A. Wieland, and J. A. Machuca. Antecedents and dimensions of supply chain robustness: a systematic literature review. *International Journal of Physical Distribution & Logistics Management*, 45(1/2):118–137, 2015.

[2] P. Egri, B. Kádár, and J. Váncza. Towards coordination in robust supply networks. *8th IFAC Conference on Manufacturing Modelling, Management and Control MIM*, 49(12):41–46, 2016.

[3] B. Fleischmann and H. Meyr. Planning hierarchy, modeling and advanced planning systems. *Handbooks in Operations Research and Management Science*, 11:457–523, 2003.

[4] Y. Hou, X. Wang, Y. J. Wu, and P. He. How does the trust affect the topology of supply chain network and its resilience? An agent-based approach. *Transportation Research Part E: Logistics and Transportation Review*, 116:229–241, 2018.

[5] T. Kreiter and U. Pferschy. Integer programming models versus advanced planning business software for a multi-level mixed-model assembly line problem. *Central European Journal of Operations Research*, Aug 2019. 10.1007/s10100-019-00642-z.

[6] G. Lanza, K. Ferdows, S. Kara, D. Mourtzis, G. Schuh, J. Váncza, L. Wang, and H.-P. Wiendahl. Global production networks: Design and operation [in press]. *CIRP Annals, Manufacturing Technology*, 2019.

[7] J. Monostori. Supply chains' robustness: Challenges and opportunities. *Procedia CIRP*, 67:110–115, 2018.

[8] D. Mourtzis, M. Doukas, and D. Bernidaki. Simulation in manufacturing: Review and challenges. *Procedia CIRP*, 25:213–229, 2014.

[9] SAP. SAP help portal. https://help.sap.com/, Accessed: 2019-08-23.

[10] D. Simchi-Levi. *Operations Rules: Delivering Customer Value through Flexible Operations*. MIT Press, 2010.

[11] D. Simchi-Levi. The new frontier of price optimization. *Sloan Management Review*, Fall:22–26, 2017.

[12] D. Simchi-levi, P. Kaminsky, and E. Simchi-Levi. *Designing and Managing the Supply Chain: Concepts, Strategies, and Case Studies*. 01 2003.

[13] The AnyLogic Company. AnyLogic simulation tool. https://www.anylogic.com/, Accessed: 2019-08-23.

Process network solution of an extended multi-mode resource-constrained project scheduling problem with alternatives

Zsolt Ercsey
Department of System and Software Technology, Faculty of Engineering and Information Technology, University of Pécs, Boszorkány u. 2, 7624 Pécs, Hungary,
ercsey@mik.pte.hu

Nándor Vincze
Department of Applied Informatics, Faculty of Education, University of Szeged, Boldogasszony u. 6, 6725 Szeged, Hungary,
vincze@jgypk.u-szeged.hu

Zoltán Kovács
Optin Ltd. Oroszlán u. 4. 6720 Szeged, Hungary,
zoltan.kovacs@optin.hu

When dealing with project scheduling problems, generally there are four basic parameters considered in connection with the activities, i.e. duration, resource constraints, logical connections to the other activities within the project including presence of precedence, and the mode of execution. As a special subfield, the multimode resource-constrained project scheduling problem (MRCPSP) has emerged and has gained significant attention recently. Here, the term multimode usually refers to the fact that each activity within the project can be executed within multiple duration time considering multiple resource allocations. When considering the resources that activities require, there are renewable resources, for example manpower, and nonrenewable resources, for example an overall budget. Obviously, these resources are always limited to some extent. In principle, when considering this kind of problems, during the project the resource requirements of the activities do not change over the time. Moreover, each activity must be performed in only one of the possible modes, and mode switching is not allowed during execution.

A well known example of the above mentioned problem class is a large scale hydropower construction project, that was published by Xu and Feng (2014). They model the hydropower construction as three parallel subprojects, where uncertainties, fuzzy random environment and hybrid particle swarm optimization algorithms were considered, generating a single fixed real value as the duration of the activities, helping to minimize time and cost for the overall project. In the current paper the MRCPSP is extended, namely MRCPSP problems are considered where each activity may be executed in parallel in two different modes. First it is shown how a directed bipartite process network should be generated that represents the original MRCPSP issue. Second, the corresponding mathematical programming model is formulated. It is explained and illustrated, how multimode activities, called alternatives, may be executed in parallel to each other and yet be considered together. Time optimal and cost optimal mathematical programming models are given. Finally, the aforementioned hydropower construction project is presented as illustration.

(1) Jiuping Xu and Cuiying Feng. Multimode Resource-Constrained Multiple Project Scheduling Problem under Fuzzy Random Environment and Its Application to a Large Scale Hydropower Construction Project. The Scientific World Journal. Volume 2014, Article ID 463692, http://dx.doi.org/10.1155/2014/463692.

On the Combination of Finite State Transducers and Finite Automata with Translucent Letter

[Extended Abstract]

Madeeha Fatima
Department of Mathematics
Eastern Mediterranean University
Famagusta, North Cyprus, via Mersin 10, Turkey
madeeha.fatimaz@gmail.com

Benedek Nagy
Department of Mathematics
Eastern Mediterranean University
Famagusta, North Cyprus, via Mersin 10, Turkey
nbenedek.inf@gmail.com

ABSTRACT

In this paper, we study Transduced-Input Non-deterministic Finite Automata with Translucent Letters, i.e., T-input-NFAwtl and some closure properties of the class of languages accepted by this model. Finite automata with translucent letters are extensions of the usual finite state automata allowing to proceed the input not strictly left to right manner. T-inputFAwtl is a further extension of finite automata with translucent letters. The input is preprocessed by a finite state transducer and given to finite automata with translucent letters, i.e., FAwtl. However, T-inputFAwtl has more expressive power than FAwtl, because this new model accepts some linguistically important not context-free languages which are not accepted by FAwtl. In this paper, we show that the language class accepted by T-inputNFAwtl is closed under union (if the same transducer is used), and it is closed under intersection with regular languages.

Categories and Subject Descriptors

F.4.3 [**Formal Languages**]

General Terms

Theory, Automata, Languages

Keywords

t-input automata, automata with translucent letters, closure properties, Mealy automata, formal languages, finite state machines

1. INTRODUCTION

One of the main and most powerful branches of theoretical computer science includes automata theory and formal languages. With ongoing development in this field, the main purpose of studying automata theory is to develop such computational models which are simple and have more expressive power than the ones which are already developed. Instead of developing a really new model, we combine two relatively simple finite-state machines, i.e., Mealy machines and finite state automata, to obtain a still finite state model with a relatively large accepting power.

Finite automata are used in text processing, formal linguistics, and hardware design etc. The language class accepted by finite automata is the class of regular languages. Finite automata with translucent letter (FAwtl) is a more powerful model [15]. In each state there are some input letters which are translucent. The automaton reads and erases the first visible input letter on the tape. To understand this in detail, one can read a closely related model: cooperative distributed systems of restarting automata [16, 17]. The accepted language is closed under regular operations: union, concatenation and Kleene star and contains all rational trace languages [14, 16]. Its relation to linguistics is studied in [12, 13]. The language class accepted by FAwtl is a superset of the class of regular languages and includes some non-context-free languages.

Here we study T-inputFAwtl, it is the extension of the finite automata with translucent letters. Mealy machines are finite-state machines transforming the input to an output. In our model, first, the input is transduced and, then, it is given to a deterministic or non-deterministic FAwtl for deciding the acceptance. We have proved in [11] that the important mildly context-sensitive, not context-free languages, the multiple agreement $\{a^nb^nc^n\}$, the cross dependencies $\{a^nb^mc^nd^m\}$ and the marked copy $\{wcw|w \in \{a,b\}^*\}$ are accepted by T-inputDFAwtl. These three languages are belonging to the linguistically important mildly context-sensitive language classes [4, 10]. There are many other models, which have generating/accepting power including these languages [1, 5, 18], but our model is finite state, and therefore, with a moderate complexity.

In this paper, we are concentrating on some closure properties. In the next section we present some formal definitions and an example. In Section 3 the main results are presented, while Section 4 concludes the paper.

2. NOTATIONS AND DEFINITIONS

In this section we will recall the definitions and fix our notations. We assume that the reader already knows the basic concepts of finite automata and formal languages. For any further details, one can read standard textbooks, e.g., [2, 3, 8, 19].

A non-deterministic finite automaton (or finite state machine), an NFA, is a 5-tuple $A = (Q, \Sigma, I, F, \delta)$, where Q is the finite set of internal states, Σ is the finite alphabet of input letters, $I \subseteq Q$ is the set of initial states, $F \subseteq Q$ is the set of final (or accepting) states and δ is the transition relation of the form $Q \times \Sigma \to 2^Q$. If $|I| = 1$ and $|\delta(q, a)| \leq 1$ holds for all $q \in Q$, $a \in \Sigma$, then A is a deterministic finite automaton, a DFA. The transition relation can be extended to words $w \in \Sigma^*$, as usual. A word w is accepted by A if $q_f \in \delta(q_0, w)$, where $q_f \in F, q_0 \in I$, i.e., $\delta(q_0, w)$ includes an accepting state. The set of all accepted words form the language $L(A)$ accepted by A. Finite automata are usually represented by their graphs [2, 3, 8, 19].

A deterministic *finite state transducer* (FST), a Mealy automaton [9], is deterministic finite automaton which is transforming the input to output (and does not accept a language). Formally, it is a system $T = (Q, \Sigma, \Delta, q_0, \gamma)$, where Q and Σ are the same as at DFA, $q_0 \in Q$ is the initial state. Δ is the finite set of output symbols and γ is the transition function of the form $Q \times \Sigma$ to $Q \times \Delta$. Originally T is in its initial state, the input tape contains an input word $w \in \Sigma^*$ and the output tape is empty. When T reads a letter from the input tape, it writes an output letter to the output tape (concatenating it to the previously written letters if any) and changes its state accordingly. By $T(w) \in \Delta^*$ we denote the output produced by T on input w. FSTs can also be represented by their graphs.

Now we recall the concept of non-deterministic finite automaton with translucent letters (NFAwtl) from [12, 13, 14, 15, 16, 17]. Formally, it is a septuple $A = (Q, \Sigma, \$, \tau, I, F, \delta)$, where Q, Σ, I and F are the same as at NFA, $\$ \notin \Sigma$ is a special symbol that is used as an end marker of the input, τ is the translucency mapping of the form $Q \to 2^\Sigma$ and $\delta : Q \times \Sigma \to 2^Q$ is the transition relation that satisfies the following condition: $\forall q \in Q, \forall a \in \tau(q) : \delta(q, a) = \emptyset$. For each state $q \in Q$, the letters from the set $\tau(q)$ are translucent for q. A is called DFAwtl (deterministic FAwtl), if $|I| = 1$ and $|\delta(q, a)| \leq 1$ holds for all $q \in Q$, $a \in \Sigma$. The automaton A starts the process from an initial state and the whole input w with end marker, i.e., $w\$$ is on the input tape. If A is in a state q and its tape content is of the form $uav\$$ such that $u \in (\tau(q))^*$, $a \notin \tau(q)$, $v \in \Sigma^*$, A erases the first occurrence of the non-translucent letter a, obtaining the tape content $uv\$$ and changing the state to a state in $\delta(q, a)$. Whenever, there is no transition is defined on letter a, A could not continue the computation and rejects. Otherwise, if the tape content is $u\$$ such that $u \in (\tau(q))^*$ in a state q, the input w is accepted if $q \in F$ and rejected if $q \notin F$. The set of accepted words w is the accepted language $L(A)$. An NFAwtl may not process the input strictly from left to right and may accept a word without reading/erasing all of its letters due to the translucency mapping. Deterministic and non-deterministic FAwtl can also be given by their graphs [12, 13, 15].

Further, the model FAwtl is extended in such a way that the input is preprocessed by an FST. This new model is introduced recently in [11] and it is motivated by [6, 7] where various types of pushdown automata had such a preprocessed input. The formal definition of our new cascade/combo automata is as follows.

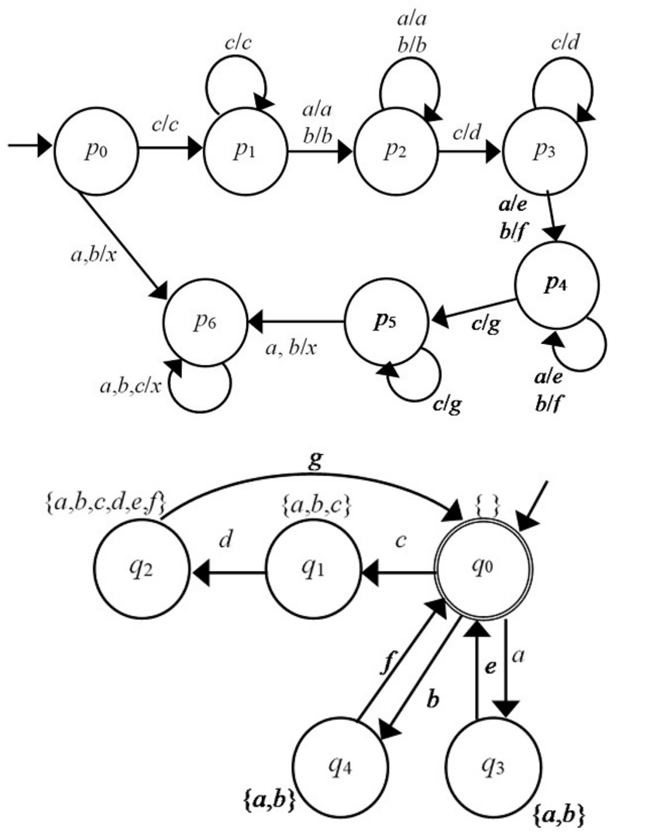


Figure 1: The T-inputDFAwtl that accepts the language L_c.

Definition 1. *Let A be an NFAwtl and $T = (Q, \Sigma, \Delta, q_0, \gamma)$ be a Mealy machine such that the output alphabet Δ of T is the same as the input alphabet of A. Then, the pair (T, A) is called a transduced-input non-deterministic finite automaton with translucent letters, i.e. T-inputNFAwtl shorty. The language accepted by the combined automata (T, A) is defined as*

$$L(T, A) = \{w \in \Sigma^* | T(w) \in L(A)\}.$$

It is shown in [11] that this type of combo machines accept three important non-context-free languages where both the Mealy machine T and the FAwtl A are deterministic. Now we give another example for a T-inputNFAwtl.

Example 1. The language $L_c = \{c^n wc^n wc^n | w \in \{a, b\}^+$ and $n > 0\}$ is accepted by the T-inputNFAwtl presented graphically in Figure 1. The figure shows the graphical representation of the transducer T (up) and the NFAwtl A (bottom). The Mealy automaton T has the following roles:

- It checks if the input is of the form $c^+(a + b)^+c^+(a +$

$b)^+c^+$; particularly if there are three factors of letter c in the input word, moreover, they are separated by non-empty words over $\{a,b\}$. Whenever the form of the input does not match, T puts at least one x to the output tape noticing this fact.

- T rewrites the second and third blocks of c's to d's and g's, respectively.
- It also rewrites the second block over $\{a,b\}$ by the alphabetic morphism $h(a)=e$, $h(b)=f$, in this way the original letters a and b are mapped to e and f, respectively, on the output tape of T in this block.

At the NFAwtl A, the set of translucent letters is shown at each state. Observe that in fact, A is also deterministic, it is a DFAwtl. It works as follows. In its initial state, which is also the only final state, there is no translucent letter, thus it must read the first letter of the word T passes to it. By the transitions of its first three states, it erases a letter c, a letter d and a letter g, thus in this cycle it checks if the number of c's and d's and g's are the same. If in the original input the format was appropriate, and the number of the c's in each of the three blocks were the same, then and only then, all c's, d's and g's are erased by A. Otherwise, either it gets stuck (if there were more c's in the first block than in any of the other blocks) or some d's and/or g's are left (if the first block of c's was shorter than the other blocks). In the second phase of the computation states q_0, q_3 and q_4 are used (in an accepting run). A erases the first letter of the remaining input, and depending on if it is an a or a b, q_3 or q_4 is reached. From this state the original letters a and b are both translucent, and the first letter of the other block, an e or an f is read such that it must fits to the previously read original letter. Observe that A cannot read any letter x. Therefore, it follows that (T,A) accepts the language L_c.

For instance, the input word *cabcabc* is in the language L_c. T preprocesses it as follows. The preprocessing starts at state p_0. The first c is kept in the transduced input as c, and state p_1 is reached. Then the first a is kept in the transduced input as a, then b has also been kept, and state p_2 is reached. After that c is rewritten to d, and state p_3 is reached. Here, the next letter, a, is rewritten to e, similarly b is transformed to f, and state p_4 is reached. Then c is transduced to g, and p_5 is reached. Thus, the word $cabdefg$ is obtained and passed to A with the end marker \$. In its initial state q_0 nothing is translucent, therefore first letter c is read and state q_1 is reached with remaining input $abdefg\$$. Here a, b and c are translucent, thus A reads d (which is the image of the second c) by changing its state to q_2 with remaining input $abefg\$$. Here again a, b, c, and also d, e, f are translucent, therefore A reads g (which in fact refers for the third block of c's) by changing its state back to q_0 with remaining input $abef\$$. Here nothing is translucent, and now the first letter is a. A reads it and state q_3 is reached with remaining input $bef\$$. Here a and b are translucent, thus, e is read from the remaining input and A moves into the state q_0 with remaining input $bf\$$. Here, there is no translucent letter, b is read and state q_4 is reached. Here a and b are translucent, the last letter f is read and A moves into its accepting state q_0 with a fully processed input. Thus, the string *cabcabc* is accepted by (T,A).

On the other hand, for example the input word *abcabc* is preprocessed by the Mealy automaton T to *xxxxxx*. It is clearly not accepted by A. The word *abcabc* is not in the language L_c. Observe that A cannot read any letter x because no transition is defined with letter x. It is used as a kind of failure symbol.

To present some closure property results in Section 3 we need to recall that all NFAwtl can be converted into normal form. In [16] (Theorem 6.5) it is proven that every NFAwtl A has an equivalent NFAwtl A' accepting the same language with special properties.

Definition 2. *An NFAwtl A is in normal form if the following conditions hold*
– In each state there is exactly 1 letter for which transitions are allowed.
– The last occurrence of each letter $a \in \Sigma$ of the input word is erased in a transition (from a state) such that the translucency mapping is empty.
– Every input letter is processed in an accepting computation.
– The automaton has exactly 1 accepting state.

A kind of extension of the normal form is helpful to prove some of the closure properties shown in the next section.

3. RESULTS

In this section we present two of the closure properties of the language class accepted by T-inputNFAwtl. First, the regular operation union is studied, we show that the language class accepted by NFAwtl is closed under union if the same transducer is used.

Theorem 1. *Let (T,A_1) and (T,A_2) be T-inputNFAwtl. The union of the languages accepted by (T,A_1) and (T,A_2) is also accepted by a transduced-input non-deterministic finite automaton with translucent letters with transducer T.*

Proof. Given the T-inputNFAwtl (T,A_1) and (T,A_2), where $T=(Q,\Sigma,\Delta,q_0,\gamma)$ is a Mealy machine and $A_1=(Q_1,\Delta,\$,\tau_1,I_1,F_1,\delta_1)$, $A_2=(Q_2,\Delta,\$,\tau_2,I_2,F_2,\delta_2)$ are two NFAwtl, we will construct the combined automaton (T,B), where B is an NFAwtl such that $L(T,A_1)\cup L(T,A_2) = L(T,B)$.

Without loss of generality, we may assume that $Q_1 \cap Q_2 = \emptyset$.

Then, let $B=(Q_1\cup Q_2,\Delta,\$,\tau,I_1\cup I_2,F_1\cup F_2,\delta)$, where $\delta(q)=\begin{cases}\delta_1(q) & \text{if } q\in Q_1\\ \delta_2(q) & \text{if } q\in Q_2\end{cases}$ and $\tau(q)=\begin{cases}\tau_1(q) & \text{if } q\in Q_1\\ \tau_2(q) & \text{if } q\in Q_2.\end{cases}$

Since there is no interference between the computations done by A_1 and A_2 encoded in B, each of the accepting (and non-accepting) computations of A_1 and A_2 has a one-to-one correspondence with an accepting (non-accepting) computation of B, respectively. Thus, $L(B)=L(A_1)\cup L(A_2)$, and therefore, $L(T,A_1)\cup L(T,A_2)=L(T,B)$. $\square$

Now we turn to another interesting closure property, namely we study intersection by regular languages.

Theorem 2. *The language class accepted by T-input NFAwtl is closed under intersection with regular languages.*

Proof. We present the idea of the proof. Let T be a deterministic transducer, A is an NFAwtl, and B is a DFA. We want the intersection of the language accepted by the T-inputNFAwtl (T, A) and by B. To do this, the "intersection" of T and B is constructed, i.e., the transducer T' and then, its output is forwarded to A' (which is based on the normal form of the NFAwtl A).

The intersection of the two finite state machines T and B is done by the usual cross-product method, however, here one of the automata is an accepting machine while the other, and as well as the resulted automata, are transducers. In what follows, the accepting states of B must be encoded in the output allowing the NFAwtl A' to check also this condition. Thus, the output alphabet of T is doubled, and whenever, B is in accepting state (which is clearly identified since B is a DFA) in its process, a marked output letter (such as $\overline{a}$) is written in the output tape (instead the original output letter a) allowing to distinguish the positions where the prefix of the input is also in the regular language defined by B or not.

However, since NFAwtl may proceed the input in a not usual left-to-right way, we need to be careful how to know that the input is in both of the languages of (T, A) and B. To do this, we can built A' to fulfill some additional properties.

Without loss of generality, we assume that A is given in normal form as we have recalled in the end of Section 2. The NFAwtl A' should also be in normal form, moreover, it is modified in such a way that it also fulfils the following properties:
– the states of A are doubled, there is a state for transitions for an original input letter and there is also a copy with transitions of its marked version.
– the translucency mapping for each state contains both the original and the marked version of the given letters.
– there is only one accepting state.
– empty translucency mapping when reading the last letter.
– the last input letter must be a marked one.

That is, in the new normal form, in the NFAwtl, all accepting computations erase the entire input word and one can also be sure when the last letter of the input is processed. In this way the condition to be in the language $L(B)$ can also be checked by A'. In this way, (T', A') is a T-inputNFAwtl and $L(T, A) \cap L(B) = L(T', A')$. $\square$

4. CONCLUSIONS

In this paper some closure properties of a relatively new class of languages are studied. Particularly, we have shown that the language class accepted by T-inputNFAwtl is closed under union with same signatures (where same signature means that the same transducer is used for preprocessing). Also, it was shown that the language class of T-inputNFAwtl is closed under intersection with regular languages. Future work includes the investigation of other closure properties with other cases of deterministic/non-deterministic versions of T-inputFAwtl with and without assuming the same signatures.

5. REFERENCES

[1] G. Gazdar. Natural languages and context-free languages. *Linguist. Philos.*, 4:469–473, 1982.

[2] J. E. Hopcroft, R. Motwani, and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* 3rd Edition. Addison-Wesley, Boston, MA, USA, 2006.

[3] G. Horváth and B. Nagy. *Formal Languages and Automata Theory.* Typotex, Budapest, Hungary, 2014.

[4] G. Jäger and J. Rogers. Formal language theory: refining the Chomsky hierarchy. *Philos Trans R Soc Lond B Biol Sci.*, 367(1598):1956–1970, 2012.

[5] A. Kornai. Natural languages and the Chomsky hierarchy. In *Proceedings of EACL'85*, pages 1–7, 1985.

[6] M. Kutrib, A. Malcher, and M. Wendlandt. Tinput-driven pushdown automata. In *MCU 2015: 7th Int. Conf. Machines, Computations, and Universality, LNCS 9288*, pages 94–112, 2015.

[7] M. Kutrib, A. Malcher, and M. Wendlandt. Tinput-driven pushdown, counter, and stack automata. *Fundam. Inform.*, 155(1-2):59–88, 2017.

[8] P. Linz. *An Introduction to Formal Languages and Automata.* Jones & Bartlett Learning, USA, 2011.

[9] G. H. Mealy. A method for synthesizing sequential circuits. *The Bell System Technical Journal*, 34(5):1045–1079, 1955.

[10] F. Morawietz. *Two-Step Approaches to Natural Language Formalism.* Studies in Generative Grammar 64. De Gruyter, Berlin, 2003.

[11] B. Nagy and M. Fatima. Transduced-input automata with translucent letters. *submitted to publication*, 2019.

[12] B. Nagy and L. Kovács. Linguistic applications of finite automata with translucent letters. In *ICAART 2013: 5th Int. Conf. Agents and Artif. Intell., Barcelona, vol. 1*, pages 461–469, 2013.

[13] B. Nagy and L. Kovács. Finite automata with translucent letters applied in natural and formal language theory. *LNCS-TCCI, Trans. Comp. Collective Intell.*, 17, *LNCS* 8790:107–127, 2014.

[14] B. Nagy and F. Otto. CD-systems of stateless deterministic R(1)-automata accept all rational trace languages. In *LATA 2010: 4th Int. Conf. Language and Automata Theory and Applications, LNCS 6031*, pages 463–474, 2010.

[15] B. Nagy and F. Otto. Finite-state acceptors with translucent letters. In *BILC: 1st Int. Workshop on AI Methods for Interdiscipl. Res. in Language and Biology*, pages 3–13, 2011.

[16] B. Nagy and F. Otto. On CD-systems of stateless deterministic R-automata with window size one. *J. Comp. Syst. Sci.*, 78:780–806, 2012.

[17] B. Nagy and F. Otto. On globally deterministic CD-systems of stateless R-automata with window size one. *Int. J. Comp. Math.*, 90:1254–1277, 2013.

[18] S. Wintner. Formal language theory for natural language processing. In *Proceedings of ACL'02*, pages 71–76, 2002.

[19] S. Yu. *Regular languages (Chaper 2).* In: Rozenberg, G., Salomaa, A. (eds.) *Handbook of Formal Languages. vol. 1.* Springer, Berlin, 1997.

On a possible use of optimality conditions in interval Branch and Bound methods[*]

Boglárka G.-Tóth

Department of Computational Optimization
University of Szeged
Szeged, Hungary
boglarka@inf.szte.hu

ABSTRACT

Interval Branch and Bound methods (IBB) are used when rigorous solutions are needed for Nonlinear Programming (NLP) problems. Nowadays, various implementations of IBB exist, although many of them do not use the Karush-Kuhn-Tucker (KKT) or Fritz-John (FJ) optimality conditions for eliminating non-optimal boxes. When it is used, it is used only in the general form, where an interval linear system of equations needs to be solved. This is rather time-consuming, and in many cases it has a negative outcome: the tested box cannot be removed because with the overestimation on the inclusion of the gradients one can find that the optimality conditions may fulfill. In order to save unnecessary computations, the common rule is to apply such tests only when the box is "small enough". However, depending on the problem at hand "small enough" might be difficult to predict.

The idea in this research is to investigate the use of the optimality conditions from a geometrical point of view and to minimize the computational effort when the optimality conditions cannot be used to discard the given box. In this way, there is no need to predict when to apply the test on optimality conditions and so it may become more efficient. In this paper, we describe a method that checks if the conic hull of the enclosure of the gradients of the active constraints is not full, so the test can have a positive outcome.

Categories and Subject Descriptors

G.1.6 [**Optimization**]: Constrained optimization, Global optimization; G.4 [**Mathematical Software**]: Algorithm design and analysis; G.1.0 [**Mathematics of Computing**]: Numerical Analysis—*Interval arithmetic*

[*]The project was supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002) and by the Hungarian National Research, Development and Innovation Office - NKFIH (OTKA grant PD115554).

General Terms

Theory, algorithm, application

Keywords

Interval Branch and Bound, Nonlinear Programming, Optimality conditions

1. INTRODUCTION

There are many applications in physics, chemistry, and even engineering fields, where a rigorous solution of a mathematical program is sought. A Nonlinear Programming problem with difficult constraints can be solved in reasonable time only for low-dimensional instances. We consider the following problem,

$$\begin{aligned} \min_{x\in[a,b]} \quad & f(x) \\ \text{s.t.} \quad & g_i(x) \le 0, \quad i = 1,\ldots,m, \end{aligned} \tag{1}$$

where $[a,b] \subseteq \mathbb{R}^n$ denotes a general bound constraint, f and $g_i, i = 1,\ldots,m$ are continuously differentiable nonlinear functions. In order to find the global optimum, spatial Branch and Bound methods are usually used, where the search space is divided into smaller regions, thus the original problem is replaced with smaller sub-problems. For the sub-problems, lower and upper bounds are computed and their feasibility is checked, so suboptimal or unfeasible regions can be removed from the sub-problems. For rigorous computations of the bounds, interval arithmetics is one of the best choices. It guarantees that rounding errors are taken into account automatically, and even approximated parameters can be included with their validated enclosures such that all errors are included. In the literature, these methods are called Interval Branch and Bound methods (IBB).

2. INTERVAL BRANCH AND BOUND METHOD

First, we briefly summarize the fundamental concepts of interval analysis and introduce the prototype IBB algorithm. For more details, the interested reader is referred to [2, 4].

2.1 Interval Arithmetics

Following the usual notation in literature, real numbers and vectors are denoted by $x, y, \ldots$, intervals and interval vectors are denoted by $\mathbf{x} = [\underline{x}, \overline{x}], \mathbf{y} = [\underline{y}, \overline{y}], \ldots$, where components of vectors are distinguished from the vectors by use of subscripts, i.e. $x = (x_1,\ldots,x_n), \overline{x} = (\overline{x}_1,\ldots,\overline{x}_n)$; while matrices are denoted by $A, B, \ldots$. Brackets "$[\cdot]$" delimit intervals, while parentheses "$(\cdot)$" vectors and matrices. Underlines will denote lower bounds of intervals and overlines give upper bounds of intervals. The *midpoint* of an interval $\mathbf{x}$ is denoted by $m(\mathbf{x}) = \frac{\underline{x}+\overline{x}}{2}$, its *width* by $w(\mathbf{x}) = \overline{x} - \underline{x}$. The midpoint of an interval vector $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^T$ is given by $m(\mathbf{x}) = (m(\mathbf{x}_1), \ldots, m(\mathbf{x}_n))^T$, whereas its width is to be understood as $w(\mathbf{x}) = \max\{w(\mathbf{x}_i) : i = 1, \ldots, n\}$. The set of intervals will be denoted by $\mathbb{I}$, and the set of n-dimensional interval vectors, also called *boxes*, by $\mathbb{I}^n$.

The *interval arithmetic operations* are defined by

$$\mathbf{x} * \mathbf{y} = \{x * y : x \in \mathbf{x}, y \in \mathbf{y}\} \text{ for } \mathbf{x}, \mathbf{y} \in \mathbb{I}, \qquad (2)$$

where the symbol $*$ stands for $+, -, \cdot$ and $/$, where $\mathbf{x}/\mathbf{y}$ is only defined if $0 \notin \mathbf{y}$. Definition (2) is equivalent to simple constructive rules (see [2, 4]).

Definition 1. A function $\mathbf{f} : \mathbb{I}^n \to \mathbb{I}$ is called an *inclusion function* of $f : \mathbb{R}^n \to \mathbb{R}$ if it fulfills that

$$\{f(x) : x \in \mathbf{x}\} \subseteq \mathbf{f}(\mathbf{x})$$

for all boxes $\mathbf{x} \subset \mathbb{I}^n$ within the domain of f.

Observe that if $\mathbf{f}$ is an inclusion function for f then we can directly obtain lower bounds and upper bounds of f over any box $\mathbf{x}$ within the domain of f just by taking $\underline{\mathbf{f}}(\mathbf{x})$ and $\overline{\mathbf{f}}(\mathbf{x})$, respectively.

For a function h predeclared in some programming language (like sin, exp, etc.), it is not too difficult to obtain a *predeclared* inclusion function $\mathbf{h}$ since the monotonicity intervals of predeclared functions are well known and then we can take $\mathbf{h}(\mathbf{x}) = \{h(x) : x \in \mathbf{x}\}$ for any $\mathbf{x} \in \mathbb{I}$ in the domain of h. For a general function $f(x)$, $x \in \mathbb{R}^n$, the easiest method to obtain an inclusion function is the *natural interval extension*, which is obtained by replacing each occurrence of the variable x with a box $\mathbf{x}$ including it, each occurrence of a predeclared function h with its corresponding inclusion function $\mathbf{h}$, and the real arithmetic operators with the corresponding interval operators. Other inclusion functions have been proposed in the literature, from those the most widely used is the centered form

$$\mathbf{f}_c(\mathbf{x}) = \mathbf{f}(c) + \nabla\mathbf{f}(\mathbf{x})(\mathbf{x} - c),$$

where $c \in \mathbf{x}$ is usually the midpoint, and $\nabla\mathbf{f}(\mathbf{x})$ is the inclusion of the gradient of f over $\mathbf{x}$. It is generally computed by the use of Automatic Differentiation (AD), see [1]. In short, AD use operator overloading to compute the gradient along with the function evaluation, where also higher-order derivatives can be evaluated.

2.2 The prototype Interval Branch and Bound algorithm

In an IBB algorithm, there are five main steps: selection, bounding, discarding, division, and termination (see Algorithm 1). These steps have to be specified for a given implementation and their choices can have a huge effect on the efficiency of the method. Since we are going to focus only on one discarding test, the remaining steps are done as general. The usual selection rule consists of choosing a box with the smallest lower bound on the objective function $\underline{\mathbf{f}}(\mathbf{x})$ from the list of generated and non-rejected boxes (working list $\mathcal{L}_W$).

For the bounding rule, we have used both the natural interval extension and the centered form, as these are the most widely used inclusion functions.

The branching rule applied here is bisecting the two widest dimensions of the actual box in one step, generating four boxes at once.

The termination rule is either based on the width of the box, $w(\mathbf{x})$ or on the width of the inclusion of the objective, $w(\mathbf{f}(\mathbf{x}))$ (sometimes both). In this study, $w(\mathbf{x}) < \epsilon$ was used since this is the most general rule for termination.

The general discarding tests are the midpoint and cut-off tests, these are always included in an IBB method. Namely, if the current upper bound of the minimum, $\tilde{f}$ is smaller then the lower bound $\underline{\mathbf{f}}(\mathbf{x})$, the box can be discarded as it cannot contain the global minimum. Monotonicity test, non-convexity test, and the interval Newton method can be used when no constraints are present, in the opposite case feasibility tests and the KKT or FJ optimality conditions can be applied.

The feasibility test works as follows. A box $\mathbf{x}$ is feasible if it satisfies all the constraints, i.e. $\overline{\mathbf{g}}_i(\mathbf{x}) \leq 0$, $\forall i = 1, \ldots, m$, infeasible if it does not satisfy at least one of the constraints, $\exists j, \underline{\mathbf{g}}_j(\mathbf{x}) > 0$, and *undetermined* otherwise. The feasibility test discards boxes that are infeasible, and marks those boxes as feasible which satisfy all the constraints. It can also provide information about the active constraints, $A = \{i \mid 0 \in \mathbf{g}_i(\mathbf{x}), i = 1, \ldots, m\}$. Notice that to apply this test we just need an inclusion function $\mathbf{g}_i$ for each constraints.

In the next section, we will discuss in more detail the KKT and FJ optimality conditions, what we will study later on from a geometric perspective.

3. OPTIMALITY CONDITIONS

Necessary optimality conditions can be used to discard regions of the search space which fail to fulfill them. In unconstrained optimization, it is enough to check the monotonicity of the objective function. The box with the objective being monotonous can either be discarded or shrank to the facet with its minimizer that belongs to the boundary of the search space. For the constrained case it becomes much more complicated.

For problem (1) the Fritz-John optimality conditions state that for any local optimizer point x there exist multipliers $\mu_i \geq 0, i = 0, \ldots, m$, such that

$$\mu_0 \nabla f(x) + \sum_{i=1}^{m} \mu_i \nabla g_i(x) = 0 \qquad (3)$$

$$\mu_i g_i(x) = 0 \qquad i = 1, \ldots, m \qquad (4)$$

The case when $\mu_0 > 0$ is equivalent to the Karush-Kuhn-Tucker conditions when constraint qualifications hold, while $\mu_0 = 0$ means that the Mangasarian–Fromovitz constraint qualification (MFCQ) does not hold.

Algorithm 1: Prototype Interval Branch and Bound

```
1  $\mathcal{L}_\mathcal{W} \leftarrow [a,b]$, $\mathcal{L}_\mathcal{S} \leftarrow \emptyset$
2  while ( $\mathcal{L}_\mathcal{W} \neq \emptyset$ ) do
3      Select a box $\mathbf{x}$ from $\mathcal{L}_\mathcal{W}$                        // Selection Rule
4      Compute bounds for $\mathbf{f}(\mathbf{x}), \mathbf{g_j}(\mathbf{x}), j = 1, \ldots, m$   // Bounding Rule
5      if ($\mathbf{x}$ cannot be discarded) then                                       // Discarding Tests
6          Divide $\mathbf{x}$ into subboxes $\mathbf{x}_1, \ldots, \mathbf{x}_s$        // Division Rule
7      for $i = 1$ to $s$ do
8          if ($\mathbf{x}_i$ satisfies the termination criterion) then                 // Termination Rule
9              Store $\mathbf{x}_i$ in $\mathcal{L}_\mathcal{S}$
10         else
11             Store $\mathbf{x}_i$ in $\mathcal{L}_\mathcal{W}$
12 return $\mathcal{L}_\mathcal{S}$
```

Conditions (4) can be omitted if in (3) only active constraints are considered (constraint g_i is active at x if $g_i(x) = 0$).

The straightforward extension of these optimality conditions is to solve the interval valued system of equations

$$\boldsymbol{\mu}_0 \nabla \mathbf{f}(\mathbf{x}) + \sum_{i=1}^{m} \boldsymbol{\mu}_i \nabla \mathbf{g}_i(\mathbf{x}) = 0 \quad (5)$$

$$\boldsymbol{\mu}_i \mathbf{g}_i(\mathbf{x}) = 0 \qquad i = 1, \ldots, m \quad (6)$$

for a given box $\mathbf{x}$ with $\boldsymbol{\mu}_i = [0, M]$ (using a big enough M [2]). By solving such an interval valued system of equations, we either discard the box, if no solution exists, or narrow it to the solution box otherwise. However, if the enclosures of the gradients are too wide we cannot remove any part of the box.

3.1 Geometrical interpretation of the optimality conditions

Many textbooks give a nice figure to show the graphical meaning of the optimality conditions such as Figure 1 taken from [3].

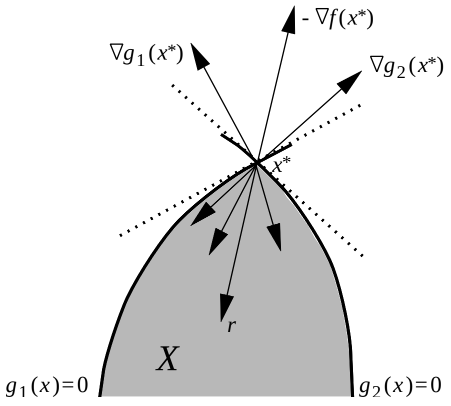
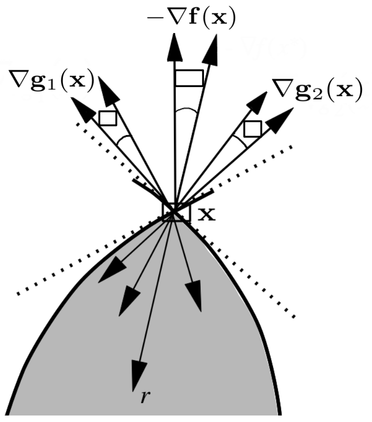


Figure 1: Graphical meaning of the optimality conditions, r is a feasible direction.

We can see the feasible directions, like r, in the gray feasible area, and the gradient vectors at the optimizer point x^*. Graphically, the necessary condition is that $-\nabla f(x)$ has to be in the conic hull of the gradients $\nabla g_i(x)$ of the active constraints.

In the interval world, instead of point x^*, we have a box $\mathbf{x}$, and instead of the gradients, enclosures of the gradients are given. The graphical interpretation can be seen in Figure 2. Let us note that in this case if the enclosure of a gradient contains 0 in its interior, it contains all directions. It follows that even if one constraint i fulfills that $0 \in \text{int}(\nabla \mathbf{g}_i(\mathbf{x}))$ the conic hull is full, so it will contain all directions of $\nabla \mathbf{f}(\mathbf{x})$ (int refers to the interior of a set).



$-\nabla \mathbf{f}(\mathbf{x})$
$\nabla \mathbf{g}_1(\mathbf{x})$
$\nabla \mathbf{g}_2(\mathbf{x})$
$\mathbf{x}$
r

Figure 2: Graphical interpretation of the optimality conditions with enclosures of the gradients.

If $\text{int}(\nabla \mathbf{f}(\mathbf{x}))$ contains 0, so its conic hull is full, there are always directions which are contained in the conic hull C of the active gradients. Thus, the box cannot be removed, although it may be narrowed by propagating the gradients in $\nabla \mathbf{f}(\mathbf{x})$ which are not in C.

4. GEOMETRIC OPTIMALITY TEST

Instead of solving directly the interval-valued linear system (3)-(4), we want to build a method which returns as soon as we know that the test cannot succeed.

The final procedure will build the conic hull of the gradients of the active constraints and will check the intersection of $\nabla \mathbf{f}(\mathbf{x})$ with this conic hull. If the intersection is empty, the box can be deleted as no point exists in it which could fulfill the optimality conditions.

To achieve this, we first seek the normal vector of the hyperplane which separates the gradient boxes of the active constraints from the origin. Taking the conical projection of the boxes on this hyperplane, one can compute the convex hull of the projected boxes to get the conic hull of the original boxes. If no separating hyperplane exists, the test cannot remove any part of the box, so we stop. Otherwise, the normal vector of the hyperplane, v, will be computed. Here, we only discuss this phase, because it already can speed up the IBB algorithm by skipping the calculation of the solution of an interval-valued system of equations.

The separating hyperplane search method is described in Algorithm 2. As a first step in line 3, we check and store the active constraints into the index set A. In fact, we only need to check if the box $\mathbf{x}$ is undetermined for a given constraint. Then, in line 4, if either one of the gradients of the active constraints or the gradient of the objective function contains 0, we stop. We do this, because in the first case no discarding is possible, while in the second case chances to reduce the box are too small. In steps 6-9 we collect for each direction j the monotonous constraints into sets M_j^+ and M_j^- depending on the sign of the gradient in dimension j. In lines 10-12, if for a direction j all active constraints are either monotone increasing or monotone decreasing, then the separating hyperplane exists and v_j is either the minimal lower bound or the maximal upper bound of the gradient's enclosures.

Algorithm 2: Separating hyperplane search $\mathbf{x}$

```
1  SeparatingHyperplane($\mathbf{x}, \nabla\mathbf{g}_i(\mathbf{x})$)
2      $v = 0$
3      $A = \{i = 1, \ldots, m \mid 0 \in \mathbf{g}_i(\mathbf{x})\}$
               // set of active constraints
4      if $\exists i \in A, 0 \in \mathrm{int}(\nabla\mathbf{g}_i(\mathbf{x}))$ or $0 \in \mathrm{int}(\nabla\mathbf{f}(\mathbf{x}))$
        then
5          return false
6      for $\forall j \in D := \{1, \ldots, n\}$ do
7          $M_j^+ = \{i \in A \mid \nabla_j\mathbf{g}_i(\mathbf{x}) > 0\}$
8          $M_j^- = \{i \in A \mid \nabla_j\mathbf{g}_i(\mathbf{x}) < 0\}$
9          $M_j = M_j^+ \cup M_j^-$
10     if $\exists j : M_j^+ = A$ (or $\exists j : M_j^- = A$) then
11         $v_j = \min_{i \in M_j^+} \underline{\nabla_j\mathbf{g}_i}(\mathbf{x})$
            (or $\max_{i \in M_j^-} \overline{\nabla_j\mathbf{g}_i}(\mathbf{x})$)
12         return $v$
13     $Conf = \{j : |M_j^+| > 0 \text{ and } |M_j^-| > 0\}$
14     for $j \in Conf$ do
15         if $\nexists(l \in D, *, \times \in \{+,-\}) : M_j^* \subseteq M_l^\times$ then
16             return false   // conic hull is full
17         if $M_j \subseteq M_l^+$ then
18             $v_l = \min_{i,k}\{\underline{\nabla_l\mathbf{g}_i}(\mathbf{x}), \underline{\nabla_l\mathbf{g}_k}(\mathbf{x})\}$
19         if $M_j \subseteq M_l^-$ then
20             $v_l = \max_{i,k}\{\overline{\nabla_l\mathbf{g}_i}(\mathbf{x}), \overline{\nabla_l\mathbf{g}_k}(\mathbf{x})\}$
21         if $\exists(l \in D, *, \times \in \{+,-\}) : M_j^* \subseteq M_l^\times$ then
                       // Translate the case to the
               positive quadrant
22             $v_j = 1$
23             $\mathbf{G}_{ij} = *\nabla_j\mathbf{g}_i(\mathbf{x})\ \forall i \in M_j$  // $* \in \{+,-\}$
24             $\mathbf{G}_{il} = \times\nabla_l\mathbf{g}_i(\mathbf{x})\ \forall i \in M_l$  // $\times \in \{+,-\}$
25             if $v_l = \min_{i \in M_j^*, k \in M_l^\times}\left\{\frac{\overline{\mathbf{G}_{kj}}}{\underline{\mathbf{G}_{kl}}} - \frac{\overline{\mathbf{G}_{ij}}}{\underline{\mathbf{G}_{il}}}\right\} > 0$
                then return $v$ else return false
```

In line 13 the conflicting directions are collected to $Conf$, that is, a coordinate direction where there are both monotone increasing and monotone decreasing constraints. From line 14 to 25 we try to resolve these conflicting cases if possible and return if not as follows.

First of all, in lines 15-16, a conflicting direction j can only be resolved if a monotonous direction l exists where all M_j^+ or M_j^- constraints are included in M_l^+ or M_l^-. If no such monotonous direction exists, there is no separating hyperplane, thus the conic hull is full and the test cannot delete any part of the box $\mathbf{x}$. On the positive side, if there is a coordinate direction j where all conflicting constraints are all monotone increasing (resp. decreasing), then v_l can be set to be the maximal upper bound (resp. the minimal lower bound) of the involved constraints (see lines 17-20).

In the last part, lines 21-25, we deal with the case where all the monotone increasing (or decreasing) constraints are also monotone increasing (or decreasing) in another direction, that is, $M_j^+ \subseteq M_l^+$, or $M_j^- \subseteq M_l^+$, or $M_j^- \subseteq M_l^+$, or $M_j^- \subseteq M_l^-$. Here it is easier to handle everything in one case, so we translate it to the positive quadrant, and check if $v_l > 0$. If this is the case, the conflict can be solved.

5. SUMMARY

We have built a procedure which tests if the conic hull of the gradients of the active constraints can be generated such that it is not full. If the procedure returns v, we can take further steps trying to discard parts of the examined box, while if the procedure returns false, we already know that the optimality test could not discard any part of the box. We expect that this method can be efficiently used as a filter before any optimality test.

6. REFERENCES

[1] A. Griewank and G. Corliss, editors. *Automatic differentiation of algorithms: theory, implementation and application.* SIAM, Philadelphia, 1991.

[2] E. Hansen. *Global optimization using interval analysis.* Marcel Dekker, New York, 1992.

[3] E. Hendrix and B. G.-Tóth. *Introduction to nonlinear and global optimization.* Springer, New York, 2010.

[4] R. Kearfott. *Rigorous global search: continuous problems.* Kluwer, Dordrecht, 1996.

On the membership problem for some classes of random context grammars

[Extended Abstract]

Zsolt Gazdag
Department of Foundations of Computer Science
Institute of Informatics
University of Szeged
gazdag@inf.u-szeged.hu

ABSTRACT

In this paper we show that the (uniform) membership problem for permitting random context grammars is NP-hard. A similar result is shown for a restricted class of forbidding random context grammars.

Keywords

Random context grammars, membership problem, complexity, NP-hardness

1. INTRODUCTION

Given a class $\mathcal{F}$ of formal language representations, one of the most important questions concerning $\mathcal{F}$ is the complexity of its (uniform) membership problem which sounds as follows. Given a formal language representation $F \in \mathcal{F}$ and a string w, decide if w is in the language represented by F. Clearly, for practical applications a polynomial-time solvable membership problem is desirable.

In this work we investigate the complexity of the membership problem for some classes of *random context grammars* (RCGs, for short) [9]. These grammars are such extensions of context-free grammars, where two sets of nonterminals, a permitting set P and a forbidding one Q, are associated to every context-free rule. Then a rule is applicable if it is applicable in the context-free sense, and nonterminals in Q do not occur while every nonterminal in P does occur in the current sentential form. If in an RCG each rule is associated with an empty forbidding set (resp. permitting set), then the grammar is called a *permitting* (resp. *forbidding*) RCG. These grammars were widely investigated as they are simple yet powerful extensions of context-free grammars (see e.g. [1, 2, 3, 4, 5, 6, 10] and the references therein).

It is known that the membership problem for forbidding RCGs is NP-hard even if the use of erasing rules are not allowed (see e.g. Chapter 3 in [7] and the references therein). On the other hand, to the best of our knowledge, it is not known if there is a polynomial-time algorithm to solve the membership problem for permitting RCGs. In this work we show that there is no such an algorithm unless P = NP. In fact, we are going to show that there is an efficient reduction of the 3-PARTITION problem, a well known NP-complete problem, to the membership problem of permitting RCGs with no erasing rules.

There is a restriction of RCGs where different rules with the same nonterminal on the left-hand side should be associated with the same permitting and forbidding sets. Moreover, one of these sets is always a singleton and the other one is empty [6]. We refer to this variant in this paper as *restricted random context grammars*. It is shown in [1] that for every permitting RCG G an equivalent restricted permitting RCG G' can be efficiently constructed such that G employs erasing rules if and only if G' does. Using this and the NP-hardness of the membership problem for permitting RCGs we can easily conclude that the membership problem for restricted permitting RCGs with no erasing rules is NP-hard, too.

It is known that restricted forbidding RCGs are equivalent to forbidding RCGs if erasing rules are allowed to use [5]. Moreover, the construction presented in [5] is an efficient one. Thus, in this case the NP-hardness of the membership problem for the restricted variant follows from the known NP-hardness of this problem for the unrestricted variant. On the other hand, it is open whether the mentioned equivalence holds also when erasing rules are not allowed to use. Nevertheless, we can show that the membership problem for the restricted variant is NP-hard even in this case. More precisely, we show that the membership problem for restricted fRCGs without erasing rules is NP-hard by giving an efficient reduction of the 3-PARTITION problem to this membership problem.

2. PRELIMINARIES

We assume that the reader is familiar with the basic concepts of the theory of formal languages. For a comprehensive guide we refer to [8].

$\mathbb{N}$ denotes the set of natural numbers and, for a number $i \geq 1$, $[i]$ denotes the set $\{1, 2, \dots, i\}$. For a word $w \in \Sigma^*$, where Σ is an alphabet, $|w|$ denotes the number of symbols in

w. Let $a \in \Sigma$. Then $|w|_a$ denotes the number of occurrences of a in w.

The 3-PARTITION problem is defined as follows. Given a multiset $H = \{n_1, \ldots, n_{3m}\}$ of positive integers, for some $m \geq 1$. Decide if there is a partition of H into m triplets $H_1, \ldots, H_m$ such that the sum of the numbers in each H_i ($i \in [m]$) is equal.

A *random context grammar* (*RCG* for short) is a 4-tuple $G = (V, \Sigma, R, S)$, where V and Σ are disjoint alphabets of the *nonterminal* and *terminal symbols*, respectively, $S \in V$ is the *start symbol*, and R is a finite set of *production rules*. The rules in R are of the form $(A \to \alpha, P, Q)$, where $A \in V, \alpha \in (V \cup \Sigma^*)$ (that is, $A \to \alpha$ is a usual context-free rule) and $P, Q \subseteq V$. P and Q are called the *permitting* and *forbidding sets* of the corresponding rule, respectively. For a rule $r : (A \to \alpha, P, Q)$, A and α are called the *left-* and *right-hand side* of r, respectively. Moreover, if $\alpha = \varepsilon$, then r is an *erasing rule*. If, for every rule in R, the permitting (resp. forbidding) set is empty, then G is a *forbidding RCG*, or *fRCG* for short (resp. *permitting RCG*, a *pRCG* for short). For the sake of readability, in pRCGs (resp. in fRCGs) we will drop the (empty) forbidding (resp. permitting) sets from the rules.

The *derivation relation* of G is defined as follows. For every word $u_1, u_2, \alpha \in (V \cup \Sigma)^*$ and $A \in V$, $u_1 A u_2 \Rightarrow_G u_1 \alpha u_2$ if and only if there is a rule $(A \to \alpha, P, Q) \in R$ such that

1) for every nonterminal $B \in P$, $|u_1 A u_2|_B \geq 1$, and
2) for every $B \in Q$, $|u_1 A u_2|_B = 0$.

We will drop G from $\Rightarrow_G$ if G is clear from the context. The *reflexive, transitive closure of* $\Rightarrow$ is denoted by $\Rightarrow^*$ and the *language generated by* G is $L(G) := \{u \in \Sigma^* \mid S \Rightarrow^* u\}$. A word $\alpha \in (\Sigma \cup V)^*$ is called a *sentential form* if $S \Rightarrow^* \alpha$. Any derivation of the form $S \Rightarrow^* u$, where $u \in L(G)$ is called a *successful* derivation. G is called ε-free, if the rule set of G contains no erasing rules.

Next we recall a restriction on RCGs introduced in [6]. A random context grammar $G = (V, \Sigma, R, S)$ is called *restricted* (G is an *rRCG* for short) if

1) for every rule $(A \to \alpha, P, Q)$ in R, $|P| + |Q| = 1$ and,
2) for any two rules $r_1 : (A \to \alpha_1, P_1, Q_1)$ and $r_2 : (A \to \alpha_2, P_2, Q_2)$ in R, we have $P_1 = P_2$ and $Q_1 = Q_2$.

Throughout the paper we will often combine the above introduced abbreviations concerning random context grammars. For example, rfRCG abbreviates that "restricted forbidding random context grammar".

3. THE MAIN RESULTS

Here we show that the membership problem for ε-free permitting RCGs as well as for ε-free restricted forbidding RCGs is NP-hard. We do this by giving efficient reductions of the 3-PARTITION problem. This problem is a well known strongly NP-complete problem, which means that it is NP-complete even if the numbers of the input instances are encoded in unary. The usefulness of this is that in this case we can implement the sum of two numbers simply with the concatenation of the words representing these numbers.

THEOREM 1. *The membership problem for ε-free pRCGs is NP-hard.*

PROOF. Consider an instance $\mathcal{I}$ of 3-PARTITION, where $H = \{n_1, \ldots, n_{3m}\}$. We construct a permitting random context grammar $G_{\mathcal{I}}$ as follows. Let $G_{\mathcal{I}} = (V, \Sigma, R, S_0)$, where

- $V = \{A_1, \ldots, A_{3m}, S_0, S_1, \ldots, S_{3m}\}$
- $\Sigma = \{a, \bullet, \lhd\}$
- $R = R_1 \cup R_2 \cup R_3 \cup R_4$, where
  - $R_1 = \{(S_{i-1} \to A_j S_i, \emptyset) \mid i, j \in [3m], i \not\equiv 0 \pmod 3\}$,
  - $R_2 = \{(S_{i-1} \to A_j \bullet S_i, \emptyset) \mid i, j \in [3m], i \equiv 0 \pmod 3\}$,
  - $R_3 = \{(S_{3m} \to \lhd, \{A_1, \ldots, A_{3m}\})\}$, and
  - $R_4 = \{(A_i \to a^{n_i}, \emptyset\}) \mid i \in [3m]\}$.

Consider now a successful derivation $\mathcal{C}$ of $G_{\mathcal{I}}$. We claim that $\mathcal{C}$ can be divided into the following three parts.

- In the first part, $G_{\mathcal{I}}$ applies rules from $R_1 \cup R_2$,
- in the next part the only rule in R_3 is used,
- in the final part rules from R_4 are applied.

Next we prove this claim. Denote r the only rule in R_3. It is clear that $G_{\mathcal{I}}$ applies exactly $3m$ rules from $R_1 \cup R_2$ until S_{3m} appears and no rules from $R_1 \cup R_2$ are applied after that. Thus there are exactly $3m$ nonterminals from $\{A_1, \ldots, A_{3m}\}$ in the sentential form when S_{3m} appears. Moreover, to apply r all the nonterminals in $\{A_1, \ldots, A_{3m}\}$ must be present in the current sentential form. Thus, there is exactly one copy of each nonterminal form $\{A_1, \ldots, A_{3m}\}$ in the sentential form when r is applied. Therefore no rules from R_4 can be applied before the rule r is applied which finishes the proof of the claim.

Consequently, at the end of the first part of $\mathcal{C}$ the sentential form has the form

$$A_{i_1} A_{i_2} A_{i_3} \bullet A_{i_4} A_{i_5} A_{i_6} \bullet \ldots \bullet A_{i_{3m-2}} A_{i_{3m-1}} A_{i_{3m}} \bullet S_{3m},$$

where $i_1, \ldots, i_{3m}$ is a permutation of $[3m]$. Then $G_{\mathcal{I}}$ replaces S_{3m} by $\lhd$ during the second part of $\mathcal{C}$ and then replaces the nonterminals A_i ($i \in [3m]$) with a^{n_i} during the last part. It follows that $G_{\mathcal{I}}$ generates the following language:

$$L(G_{\mathcal{I}}) = \{a^{n_{i_1}+n_{i_2}+n_{i_3}} \bullet a^{n_{i_4}+n_{i_5}+n_{i_6}} \bullet \ldots$$
$$\ldots \bullet a^{n_{i_{3m-2}}+n_{i_{3m-1}}+n_{i_{3m}}} \bullet \lhd$$
$$\mid i_1, \ldots, i_{3m} \text{ is a permutation of } [3m]\}.$$

In this way the words in $L(G_{\mathcal{I}})$ encode exactly those sequences of numbers which correspond to the sums of the numbers occurring in the triplets in the possible partitions of H. Therefore, $\mathcal{I}$ is a positive instance of 3-PARTITION if and only if $L(G_{\mathcal{I}})$ contains the word

$$w = a^k \bullet a^k \bullet \ldots \bullet a^k \bullet \lhd,$$

where a^k occurs $3m$ times in w and $k = \frac{\sum_{i=1}^{3m} n_i}{m}$. That is, we reduced the decision of whether $\mathcal{I}$ is a positive instance or not to the decision of whether w belongs to $L(G_{\mathcal{I}})$ or not. To finish the proof it is enough to note that the construction of $G_{\mathcal{I}}$ and w can be carried out by a polynomial-time deterministic Turing machine. □

In [1] it was shown that for every ε-free pRCG an equivalent ε-free restricted pRCG can be constructed. Looking at that construction one can see that it is in fact a polynomial-time construction. Using this, Theorem 1, and the fact that polynomial-time reductions are closed under composition, we get that 3-PARTITION can be reduced efficiently to the membership problem for ε-free restricted permitting RCGs. This yields the following result.

COROLLARY 1. *The membership problem for ε-free rpRCGs is NP-hard.*

Next we show a similar result concerning restricted forbidding RCGs.

THEOREM 2. *The membership problem for ε-free rfRCGs is NP-hard.*

PROOF. Consider an instance $\mathcal{I}$ of 3-PARTITION, where $H = \{n_1, \ldots, n_{3m}\}$. We construct an rfRCG $G_{\mathcal{I}}$ as follows. The basic concept is similar to the one used in the proof of Theorem 1. However, here we can use in the rules only one forbidding nonterminal instead of a set of permitting nonterminals. Therefore we will also use the concept of *complementary pairs of nonterminals* introduced in [5]. Roughly, these are such pairs of nonterminals which cannot occur together in a sentential form since otherwise the derivation cannot be successful. According to this, some rules of $G_{\mathcal{I}}$ will introduce not only those nonterminals that were already used in the proof of Theorem 1 but also the complementary pairs of certain nonterminals. We will distinguish these complementary pairs from the original nonterminals using the $\bar{}$ sign.

Let $G_{\mathcal{I}} = (V, \Sigma, R, S_0)$, where

- $V = \{A_1, \ldots, A_{3m}, \bar{A}_1, \ldots, \bar{A}_{3m}, S_0, \ldots, S_{3m}, \bar{S}_0, \ldots, \bar{S}_{3m}\}$
- $\Sigma = \{a, \bullet, \lhd\}$
- $R = R_1 \cup R_2 \cup \ldots \cup R_5$, where
  - $R_1 = \{(S_{i-1} \to A_j S_i, \{\bar{S}_{i-1}\}) \mid i, j \in [3m], i \not\equiv 0 \pmod 3\}$,
  - $R_2 = \{(S_{i-1} \to A_j \bullet S_i, \{\bar{S}_{i-1}\}) \mid i, j \in [3m], i \equiv 0 \pmod 3\}$,
  - $R_3 = \{(S_{3m} \to \lhd, \{\bar{S}_{3m}\})\}$,
  - $R_4 = \{(A_i \to a^{n_i} \bar{A}_i \bar{S}_1 \ldots \bar{S}_{3m}, \{\bar{A}_i\}) \mid i \in [3m]\}$,
  - $R_5 = \{(\bar{S}_i \to a, \{S_i\}) \mid i \in [3m]\} \cup (\bar{A}_i \to a, \{A_i\}) \mid i \in [3m]\}$.

Let us consider a successful derivation $\mathcal{C}$ of $G_{\mathcal{I}}$. We claim that $\mathcal{C}$ can be divided into the following two parts:

- in the first part, only rules from $R_1 \cup R_2 \cup R_3$ are used, while
- in the second part, only rules from $R_4 \cup R_5$ are used.

We prove this claim as follows. The rules in R_4 cannot be used until an S_j ($j \in [3m]$) is in the sentential form. Indeed, if a rule $(A_i \to a^{n_i} \bar{A}_i \bar{S}_1 \ldots \bar{S}_{3m}, \{\bar{A}_i\})$ was applied while an S_j was in the sentential form, then S_j and $\bar{S}_j$ would occur at the same time and none of them could be removed any more (notice that $\bar{S}_j$ forbids the application of rules with S_j on the left-hand side and vice versa). Furthermore, rules in R_5 cannot be used before the rules in R_4 since only rules in R_4 introduce nonterminals $\bar{A}_i$ or $\bar{S}_i$. Consequently, the rules in $R_4 \cup R_5$ cannot be used before the rules in $R_1 \cup R_2 \cup R_3$ which proves our claim.

One can see that at the end of the first part of $\mathcal{C}$ the sentential form has the form

$$w = A_{i_1} A_{i_2} A_{i_3} \bullet A_{i_4} A_{i_5} A_{i_6} \bullet \ldots \bullet A_{i_{3m-2}} A_{i_{3m-1}} A_{i_{3m}} \bullet \lhd,$$

where $i_1, \ldots, i_{3m} \in [3m]$. Assume now that there is an $i \in [3m]$ such that w contains more than one occurrence of A_i. In this case, when one A_i is rewritten by the corresponding rule in R_4 the sentential form contains both $\bar{A}_i$ and A_i. However then the derivation cannot be successful as neither of these two nonterminals can be rewritten by any rule. Consequently, $i_1, \ldots, i_{3m}$ must be a permutation of $[3m]$.

During the second part of $\mathcal{C}$, $G_{\mathcal{I}}$ rewrites a nonterminal A_i ($i \in [3m]$) to $a^{n_i} a^{3m+1}$ using rules both from R_4 and R_5. Thus $G_{\mathcal{I}}$ generates the following language:

$$L(G_{\mathcal{I}}) = \{a^{n_{i_1}} a^{3m+1} a^{n_{i_2}} a^{3m+1} a^{n_{i_3}} a^{3m+1} \bullet \ldots \bullet a^{n_{i_{3m-2}}} a^{3m+1} a^{n_{i_{3m-1}}} a^{3m+1} a^{n_{i_{3m}}} a^{3m+1} \bullet \lhd \mid i_1, \ldots, i_{3m} \text{ is a permutation of } [3m]\}.$$

Clearly, $L(G_{\mathcal{I}})$ can be written in the following form, where we underlined certain parts of the words:

$$L(G_{\mathcal{I}}) = \{a^{n_{i_1}+n_{i_2}+n_{i_3}} \underline{a^{3(3m+1)}} \bullet \ldots \bullet a^{n_{i_{3m-2}}+n_{i_{3m-1}}+n_{i_{3m}}} \underline{a^{3(3m+1)}} \mid i_1, \ldots, i_{3m} \text{ is a permutation of } [3m]\}.$$

If we remove the underlined subwords from the words above, then we get the same language that was already considered in the proof of Theorem 1. Thus we can conclude that the words in $L(G_{\mathcal{I}})$ encode all the possible 3-partitions of H.

Now consider the word

$$w = a^{k+3(3m+1)} \bullet \dots \bullet a^{k+3(3m+1)} \bullet \triangleleft$$

containing the sub-word $a^{k+3(3m+1)}\bullet$ $3m$ times, where $k = \frac{\sum_{i=1}^{3m} n_i}{m}$. One can see that $\mathcal{I}$ is a positive instance of 3-PARTITION if and only if $w \in L(G_{\mathcal{I}})$. Since $G_{\mathcal{I}}$ and w can be constructed by a polynomial-time Turing machine, we could give an efficient reduction of the 3-PARTITION problem to the membership problem for ε-free rfRCGs. Thus this latter problem is NP-hard which we wanted to prove. $\square$

4. CONCLUSIONS

We have shown that the membership problem for ε-free permitting random context grammars and for ε-free restricted forbidding random context grammars is NP-hard. On the other hand, as far as we know, it remains an open question whether the complexity of this problem for these classes of RCGs is in NP or not.

To give an NP upper bound on the complexity of this problem seems to be not trivial as it is discussed below. Consider an ε-free RCG $G = (V, \Sigma, R, S)$. Since G is ε-free, the lengths of the sentential forms in a derivation of G are monotonically increasing. Consider a derivation *der* of G and those steps in *der*, where the length of the sentential form grows. Let us call these steps *growing steps*. Assume that *der* is a derivation of a word $w \in \Sigma^*$ with length n. Clearly *der* can contain at most n growing steps. Let us call those parts of *der* which are between two consecutive growing steps *nonincreasing parts*. Notice that in a nonincreasing part, G can apply only rules of the form $(A \to u, P, Q)$, where $u \in V \cup \Sigma$.

Assume now that *der* is one of the shortest derivations of w. Clearly, to give a polynomial upper bound on the length of *der* it is enough to give such an upper bound on the lengths of its nonincreasing parts. If G is a context-free grammar (that is, every rule of G has empty permitting and forbidding sets), then giving such an upper bound is not difficult. Indeed, in this case we can rearrange the order of the rules applied in a nonincreasing part of *der* such that those rules which are applied on the same position of the sentential form are applied right after each other. Let us rearrange the rules in this way in all the nonincreasing parts of *der* and denote the yielded derivation by der'. Clearly, der' is a valid derivation of w. Let us consider a nonincreasing part der'' of der' such that all the rules in der'' are applied on the same position of the sentential form. Clearly the length of der'' is upper-bounded by $|V|$ since otherwise there would be two different rewriting steps in der'' where the same nonterminal is rewritten and thus der'' could be shortened contradicting the fact that der' is one of the shortest derivations of w. Using this we can easily conclude that the length of der' is polynomially bounded by $n \cdot |V|$.

However, if G is an RCG, then the order of the rules in *der* cannot be freely rearranged since the application of the rules depends also on context conditions. Therefore it is not clear how can one shorten a nonincreasing part der' of *der* even if der' is longer than $n \cdot |V|$. Thus, giving a polynomial upper bound on the length of the derivations of G is not possible using the method described above even if we consider only pRCGs or rfRCGs. A further investigation of this question is a topic for our future work.

5. ACKNOWLEDGEMENTS

This research was supported by grant TUDFO/47138-1/2019-ITM of the Ministry for Innovation and Technology, Hungary.

6. REFERENCES

[1] J. Dassow and T. Masopust. On restricted context-free grammars. *Journal of Computer and System Sciences*, 78(1):293–304, 2012.

[2] S. Ewert and A. van der Walt. A shrinking lemma for random forbidding context languages. *Theoretical Computer Science*, 237(1-2):149–158, 2000.

[3] S. Ewert and A. van der Walt. A pumping lemma for random permitting context languages. *Theoretical Computer Science*, 270(1-2):959–967, 2002.

[4] Z. Gazdag. A note on context-free grammars with rewriting restrictions. In A. Brodnik and G. Galambos, editors, *Proceedings of the 2010 Mini-Conference on Applied Theoretical Computer Science*. University of Primorska Press, Koper, 2011.

[5] Z. Gazdag. Remarks on some simple variants of random context grammars. *Journal of Automata, Languages and Combinatorics*, 19(1-4):81–92, 2014.

[6] T. Masopust. Simple restriction in context-free rewriting. *Journal of Computer and System Sciences*, 76(8):837 – 846, 2010.

[7] G. Rozenberg and A. Salomaa. *Handbook of Formal Languages: Volume 2. Linear Modeling: Background and Application*. Springer-Verlag Berlin Heidelberg, 1997.

[8] A. Salomaa. *Formal Languages*. Academic Press, New York, London, 1973.

[9] A. van der Walt. Random context languages. *Inform. Process.*, 71:66–68, 1972.

[10] G. Zetzsche. On erasing productions in random context grammars. In S. Abramsky, C. Gavoille, C. Kirchner, F. auf der Heide, and P. G. Spirakis, editors, *Automata, Languages and Programming, 37th International Colloquium, ICALP (2). LNCS 6199*, pages 175–186. Springer, 2010.

Portfolio selection based on a configuration model and hierarchical clustering for asset graphs

Imre Gera
University of Szeged Institute of Informatics
P.O. Box 652
H-6701 Szeged, Hungary
gerai@inf.u-szeged.hu

András London
University of Szeged Institute of Informatics,
Poznan University of Economics, Department of Operations Research
london@inf.u-szeged.hu

ABSTRACT

In this paper we present a null model based clustering method for asset graphs constructed of correlation matrices of financial asset time series. Firstly, we utilize a standard configuration model of the correlation matrix that provides the null model for comparison with the original one. Based on this comparison we define a distance matrix – called asset graph – on which we perform hierarchical clustering procedures. We apply this method to find clusters of similar assets in correlation based graphs obtained form various stock market data sets. We evaluate the performance of the procedure through the Markowitz portfolio selection problem by providing a simple asset allocation strategy based on the obtained cluster structure.

Categories and Subject Descriptors

I.6 [**Simulation and Modeling**]: Applications
; G.1.6 [**Optimization**]: Nonlinear programming

Keywords

Correlation matrix, Complex networks, Clustering, Portfolio selection

1. INTRODUCTION

Correlation matrices are of central importance in financial economics, especially in portfolio theory. Correlations among various assets' returns is used to determine the relative amount of capital should be invested in different assets in order to minimize the investor's risk [4]. Graphs can be easily constructed from correlation matrices in different ways. In asset graphs a node represents a company and a weighted edge between two nodes indicates, for instance, the equal-time Pearson correlation coefficient between their corresponding stock prices [3, 9, 11]. Considering correlation matrices as graphs, a wide range of tools in network analysis, like centrality measures, frequent sub-graph search or community detection, becomes available [1]. Nevertheless, the direct conversion to graphs is not evident, since the problem of information content of correlation matrices plays a key role in applications, especially in risk management. The estimation of the correlation matrix is associated with a significant level of a statistical uncertainty (sometimes called noise) due to the finite length of the asset return time series [15]. Recently, several approaches, that appeared especially in the 'Econophysics' and 'Complex Networks Analysis' literature, have been developed to handle this issue, e.g. [2, 6, 13, 14]. The idea is to filter the 'information core' of the correlation matrix that is robust against statistical uncertainty. One approach is based on random matrix theory and the idea is to compare the empirical correlation matrix with a null model matrix. The null model matrix is defined as the correlation matrix of the same number of random time series of the same length as the empirical one. A barely different approach, preferably used in the finance literature, is the principal component analysis [5]. Other filtering methods perform hierarchical clustering procedures such as single-linkage clustering [9] or average-linage clustering [13].

In this work we follow a standard null model approach for correlation matrices, but consider the information filtering problem as a graph based data mining task. We should emphasize, that in [8] the authors showed that treating the original correlation matrix as a weighted graph directly and apply modularity maximization for clustering using a standard null model approach may lead to biased results. This is due to the fact that the configuration null model doesn't necessarily give enough importance to node pairs with stronger correlations, however this is often desired in clustering algorithms. They also provided several versions of the modularity function for correlation matrices. We choose a much simpler way: we filter the original correlation matrix using a null model matrix and transform the filtered matrix to a distance matrix in a proper way. Then a hierarchical clustering procedure on the distance matrix is performed, regarded as a heuristic to maximize a modularity-like function.

The paper is organized as follows. In Section 2 we briefly describe some ways to construct asset graphs from correlation matrices, and present a heuristic for community detection (i.e. clustering) for these graphs. In Section 3 we present our experiments in various stock market data sets through the Markowitz portfolio selection problem by providing an asset allocation strategy based on the obtained cluster structure. Finally, we summarize in Section 4.

2. METHODS

Let $X_i \equiv \{x_i(t) : t = 0, 1, \ldots, T\}$ be a time series represents the value of some unit i ($i = 1, 2, \ldots n$) at time t. Particularly, in financial markets i is an asset and $x_i(t)$ is the logarithmic return of it, i.e. $x_i(t) = \log P_i(t)/P_i(t-1)$, where $P_i(t)$ is the price of asset i at time t. The system of n assets is often investigated via the correlation matrix $\mathbf{C}$ that statistically measures the pairwise dependencies, where C_{ij} is the Pearson correlation coefficient of assets i and j. It is calculated as

$$C_{ij} = \frac{\mathrm{Cov}(X_i, X_j)}{\sqrt{\mathrm{Var}(X_i) \cdot \mathrm{Var}(X_j)}},$$

where

$$\mathrm{Cov}(X_i, X_j) = \overline{X_i \cdot X_j} - \overline{X_i} \cdot \overline{X_j}$$

is the covariance of X_i and X_j, $\mathrm{Var}(X_i) = \mathrm{Cov}(X_i, X_i) = \sigma_i^2$ is the auto-covariance of X_i and $\overline{X_i}$ denotes the temporal average of the observations of X_i, i.e.

$$\overline{X_i} = \frac{1}{T} \sum_{t=0}^{T} x_i(t),$$

$$\overline{X_i X_j} = \frac{1}{T} \sum_{t=0}^{T} x_i(t) x_j(t).$$

We assume that X_i is standardized as $(X_i - \overline{X_i})/\sigma_i$.

2.1 Asset graphs

Since the correlation matrix $\mathbf{C}$ is a symmetric $n \times n$ matrix, it can be viewed as the adjacency matrix of a weighted graph. In this graph, nodes represent the assets and edges represent correlation coefficient of asset pairs. In the literature, $\mathbf{C}$ is often transformed into a distance matrix $\mathbf{D}$ with entries $D_{ij} = \sqrt{2(1 - C_{ij})}$. This is motivated by the hypothesis that ultrametric spaces[1] are meaningful in economic perspective [10].

A simple filtering technique is to threshold the values of $\mathbf{C}$ (or $\mathbf{D}$), leaving only those edges that are greater than an arbitrarily chosen value. Although the method effectively discards the weakest correlations, that are likely to caused by random fluctuations in the time series, using an inappropriate threshold value may hide important structural features of the asset graph.

A different technique, that does not require to choose a global threshold value is the minimal spanning tree approach. It reduces the number of edges of the graph from $n \cdot (n-1)/2$ to $n - 1$. The procedure is closely related to agglomerative hierarchical clustering performed with the single-linkage distance definition [9]. The approach assumes that the original correlations are approximated well by the filtered ones, and similarly to the threshold based filtering it discards all the weaker correlations. To discard less information, one can use the planar maximally filtered graph approach [14]. The method retains both the correlations used to create the minimal spanning tree and additional information as well, provided that the result is a planar graph.

[1]Ultrametric spaces are defined by an ultrametric distance that satisfy the axioms (i) $D_{ij} = 0 \Leftrightarrow i = j$, (ii) $D_{ij} = D_{ji}$ and (iii) $D_{ij} \le \max\{D_{ik}, D_{kj}\}, \forall(i, j, k)$.

An important technique, based on fundamental results of random matrix theory, decomposes the correlation matrix $\mathbf{C}$ into a 'structured' and a 'random' part [7]. This is done by comparing eigenvalues of the empirical correlation matrix with the correlation matrix of the same number of random time series of the same length. The latter is known to be given by the Marchenko-Pastur distribution [12]. We use a similar technique in this work, but we choose a so-called configuration model to construct the null model matrix that will be compared with the original one.

2.2 Configuration model and community detection in graphs

A null model correlation matrix $\mathbf{C}^0$ is an $n \times n$ matrix, where C^0_{ij} is the mean value of the correlation between assets i and j under some null model benchmark. For example, under the assumption that every asset is uncorrelated then $\mathbf{C}^0$ would be the $n \times n$ identity matrix. Here, we use a configuration model as null model to generate C^0_{ij} by replacing edges (of the correlation graph) independently at random. The assumption is that the generated $\mathbf{C}^0$ correlation matrix preserves the *strength* of each asset i, i.e. $C_i = \sum_j C_{ij}$ is fixed as much as possible, while randomizing the 'correlation structure'.

We consider $\mathbf{C}' = |\mathbf{C} - \mathbf{C}^0|$ as the filtered (i.e. 'cleaned') correlation matrix. Then we define the re-scaled $\mathbf{D}_c = -\mathbf{C}' + |\min \mathbf{C}'| + |\max \mathbf{C}'|$ distance matrix, that may be interpreted as a weighted graph related to the correlation matrix. Here smaller distance between two nodes refers larger correlation between the corresponding assets. We then apply hierarchical clustering to $\mathbf{D}_c$. This method can be regarded as a heuristic to maximize a modularity-type function, used for clustering, given as $\sum_{i,j}[C_{ij} - C^0_{ij}]\delta_{ij}$, where $\delta_{ij} = 1$ if i and j assigned to the same cluster, and $\delta_{ij} = 0$ otherwise. Hierarchical clustering results in a *dendrogram* that can we cut at an arbitrary level h from the root to get h clusters of stocks.

3. EXPERIMENTS

Correlation (covariance) matrices often used in portfolio optimization (a widely-used model will be described). The performance of the different noise filtering procedures is generally measured via various performance metrics of composed portfolios using filtered correlation matrices.

3.1 Data sets

For our experiments we have relied on the daily closure price time series of three different stock data sets available at *Yahoo! Finance.* The selection of the stocks was based on global indices in two cases (FTSE100 and DOW30), and we also chose the 30 stocks that were active for the longest period among the available time series data. For the sake of simplicity, we refer to these data sets as "FTSE" ($n = 32$ stocks, 1183 records from 16-05-2011 to 27-01-2016), "DOW" ($n = 29$ stocks, 2849 records from 19-03-2008 to 12-07-2019) and "Active30" ($n = 30$ stocks, 5398 records from 19-01-1995 to 27-06-2016).

3.2 Markowitz portfolio selection

The Markowitz portfolio selection problem is an optimization problem where the investor would like to create an opti-

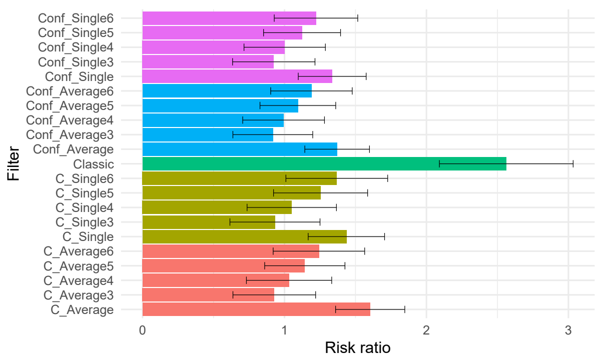


Figure 1: Risk ratios on the 'FTSE' dataset. The lower, the better.

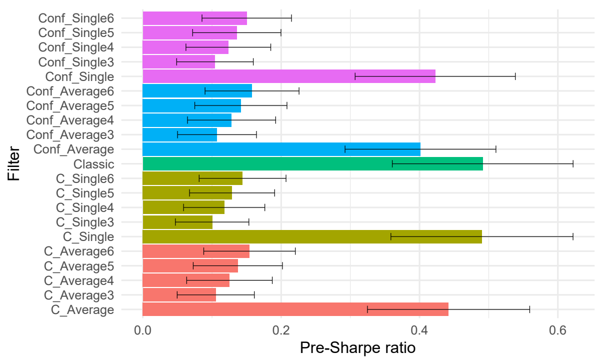


Figure 2: Sharpe ratios on the 'FTSE' dataset. The greater, the better.

mal portfolio of assets with minimum risk, given an expected return in advance. The portfolio is represented as a vector $\mathbf{p}$ that consists of the fraction of wealth to be invested in each asset. We also assume that $\sum_i p_i = 1$, i.e. 100% of wealth is invested. For example $\mathbf{p} = (0.2, 0.8)$ means investing 20% of our wealth in stock #1 and 80% in stock #2. To reach the optimum, the portfolio has to satisfy two conditions. Firstly, it has to achieve an expected return $\overline{r}_p = \sum_i p_i \overline{X}_i$, where $\overline{X}_i$ is the mean log-return of stock i, greater than a specified value R (this is an arbitrary choice). Secondly, it has to provide minimal risk, measured as $\sigma_p^2 = \mathbf{p}\mathbf{\Sigma}\mathbf{p}^T$, where $\mathbf{\Sigma}$ is the covariance (i.e. not normalized correlation) matrix of the assets considered. Negative p_i weights, also referred to as *short-selling*, are allowed.

3.3 Methodology

We used the following rolling window approach to calculate the correlation (and covariance) matrices from the time series data and perform the optimizations described previously. In each dataset we calculated the correlation matrix on the time range $[t_0, t_0 + \Delta T]$, performed a filtering procedure, in a similar way as in [13], and the optimization which gave us a portfolio $\mathbf{p}$. This meant four main optimizations per each t_0 starting day: optimization without filtering ("Classic"), filtering using hierarchical clustering on (i) asset graph $\mathbf{D}$ ("C_Single", "C_Average") and (ii) on configuration model based asset graph $\mathbf{D}_c$ ("Conf_Single", "Conf_Average"). In case of clustering procedures, our portfolio selection strategy was choosing only one asset from each cluster at random and performed portfolio optimization considering only the pre-selected assets. We then evaluated the performance of the portfolios on the interval $[t_0 + \Delta T, t_0 + 2 \cdot \Delta T]$, where $t_0 \in \{0, 30, 60, \dots\}$ and $\Delta T = 100$.

For each portfolio $\mathbf{p} = (p_1, p_2, \dots)$ we calculated the realized return as

$$\sum_{i=1}^{n} p_i \frac{P_i(t_0 + 2\Delta T) - P_i(t_0 + \Delta T)}{P_i(t_0 + \Delta T)},$$

the Pre-Sharpe ratio ($\overline{r}_p / \hat{\sigma}_p^2$), and the risk ratio ($\sigma_p^2 / \hat{\sigma}_p^2$), that is the fraction of the 'realized' and estimated risk. We calculated the mean of each metric but trimmed the data by 20% (10% on the lower and 10% on the upper end) to remove possible outlier values.

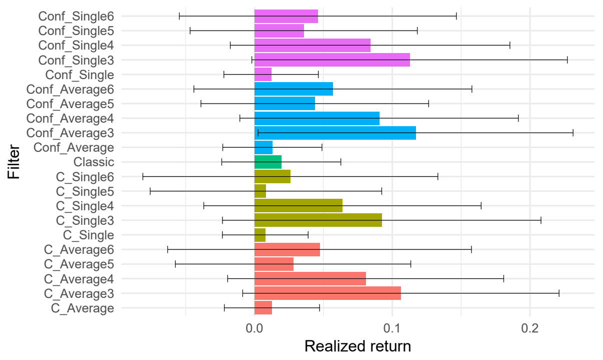


Figure 3: Realized returns on the 'FTSE' dataset. The greater, the better.

3.4 Results

Our experiments show that the resulting portfolios in general had significant improvements in all metrics when filtering methods were applied to the correlation (and hence covariance) matrix. The configuration model based approach provided lower realized risk and lower difference between estimated and realized risks than the other filtering methods (Fig. 1). The risk estimation was even better when we only used one stock per cluster (using 3 or 4 clusters provided the best risk ratios), but the estimated risk increased (the increase of the estimated risk brought it closer to the realized one). In these cases we chose a random element of the cluster, hence it was not guaranteed that we chose the assets with the lowest risk overall. Regarding Sharpe ratios (Fig. 2), it can be noted that the single-linkage clustering was the closest one to the original Markowitz-model, although when using only one stock per cluster, the value significantly decreased (due to the fact that the estimated return did not grow, but the risk increased). The configuration model performed similarly, albeit a bit worse than the other methods.

Regarding realized returns, as Fig. 3 shows, the cluster-based asset selection improved performance. When looking at the results of all the clustering-based approaches, the configuration model provided the highest realized returns with 3 clusters (and thus 3 stocks). The worst performer was the single-linkage clustering. Filtering procedures show a similar shape over time and outperform the classic method

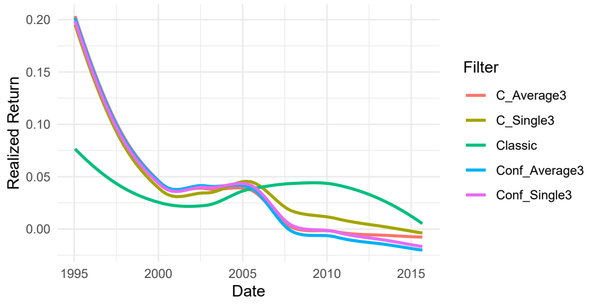


Figure 4: Realized returns of three filters on the 'Active30' dataset from 19-01-1995 to 31-08-2015.

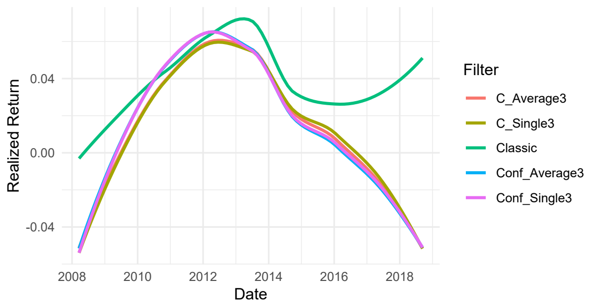


Figure 5: Realized returns of three filters on the 'DOW' dataset from 19-03-2008 to 12-09-2018.

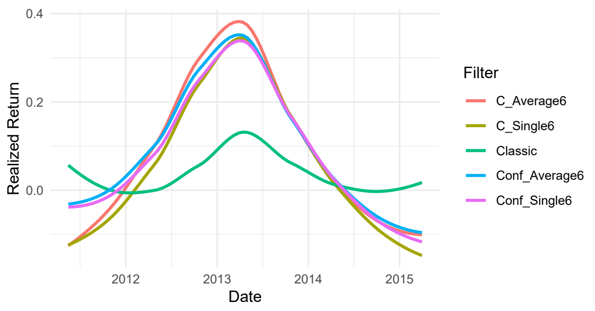


Figure 6: Realized returns of three filters on the 'FTSE' dataset from 16-05-2011 to 01-04-2015.

in certain intervals (Figs. 4-6). However, understanding the shape of the curves and the underlying causes are worth further investigation.

4. SUMMARY

In this work, by combining techniques used to investigate correlation matrices and used in graph based data mining, we performed clustering procedures for asset graphs constructed of filtered correlation matrices of financial asset time series. We provided an asset allocation strategy based on the obtained cluster structure and using Markowitz' portfolio optimization. The above discussion of our findings shows that the utilized methodology is able to provide reliable portfolios in terms of risk estimation and is competitive with classical methods in terms of return realization as well. Defining asset graphs based on different filtering procedures and cluster based asset selection strategies leave open many questions for further investigations.

Acknowledgments

The project has been supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002). IG was supported by the UNKP-19-2 New National Excellence Program of The Ministry of Human Capacities. AL was partially supported by the National Research, Development and Innovation Office - NKFIH, SNN-117879.

5. REFERENCES

[1] A.-L. Barabási et al. *Network science*. Cambridge University Press, 2016.

[2] J. Bun, J.-P. Bouchaud, and M. Potters. Cleaning large correlation matrices: tools from random matrix theory. *Physics Reports*, 666:1–109, 2017.

[3] K. T. Chi, J. Liu, and F. C. Lau. A network perspective of the stock market. *Journal of Empirical Finance*, 17(4):659–667, 2010.

[4] E. J. Elton, M. J. Gruber, S. J. Brown, and W. N. Goetzmann. *Modern portfolio theory and investment analysis*. John Wiley & Sons, 2009.

[5] R. F. Engle, V. K. Ng, and M. Rothschild. Asset pricing with a factor-arch covariance structure: Empirical estimates for treasury bills. *Journal of Econometrics*, 45(1-2):213–237, 1990.

[6] L. Laloux, P. Cizeau, J.-P. Bouchaud, and M. Potters. Noise dressing of financial correlation matrices. *Physical Review Letters*, 83(7):1467, 1999.

[7] L. Laloux, P. Cizeau, M. Potters, and J.-P. Bouchaud. Random matrix theory and financial correlations. *International Journal of Theoretical and Applied Finance*, 3(03):391–397, 2000.

[8] M. MacMahon and D. Garlaschelli. Community detection for correlation matrices. *Physical Review E*, 5:021006, 2013.

[9] R. N. Mantegna. Hierarchical structure in financial markets. *The European Physical Journal B*, 11(1):193–197, 1999.

[10] R. N. Mantegna and H. E. Stanley. *Introduction to econophysics: correlations and complexity in finance*. Cambridge University Press, 1999.

[11] J.-P. Onnela, K. Kaski, and J. Kertész. Clustering and information in correlation based financial networks. *The European Physical Journal B*, 38(2):353–362, 2004.

[12] A. M. Sengupta and P. P. Mitra. Distributions of singular values for some random matrices. *Physical Review E*, 60(3):3389, 1999.

[13] V. Tola, F. Lillo, M. Gallegati, and R. N. Mantegna. Cluster analysis for portfolio optimization. *Journal of Economic Dynamics and Control*, 32(1):235–258, 2008.

[14] M. Tumminello, T. Aste, T. Di Matteo, and R. N. Mantegna. A tool for filtering information in complex systems. *PNAS*, 102(30):10421–10426, 2005.

[15] M. Verleysen and D. François. The curse of dimensionality in data mining and time series prediction. In *International Work-Conference on Artificial Neural Networks*, pages 758–770. Springer, 2005.

Empirical Study of S-graph Approaches for Limited-Wait Storage Policy

Máté Hegyháti
Department of Information Technology
Széchenyi István University
9026 Egyetem tér 1
Győr, Hungary
hegyhati@sze.hu

Olivér Ősz
Department of Information Technology
Széchenyi István University
9026 Egyetem tér 1
Győr, Hungary
osz.oliver@sze.hu

Tibor Holczinger
Department of Applied Informatics
University of Pannonia
8800 Zrínyi Miklós u. 18.
Nagykanizsa, Hungary
holczinger.tibor@uni-pen.hu

ABSTRACT

Storage limitations of intermediate materials add an additional layer of complexity to the scheduling of batch chemical processes. Not only the storage capacities have to be taken into account, the properties of the intermediates in question often pose limitations on the storage time as well. In this paper, different techniques are discussed, which allow proper modeling of such timing limitations within the S-graph framework. The introduced techniques were implemented and tested on literature examples and case studies, to identify the most efficient one.

Keywords

scheduling, limited-wait storage policy, S-graph

1. INTRODUCTION

In case of multi-stage production recipes, storing the intermediates is an inherent burden of batch plants, while it is not an issue for their continuous counterparts. Storage operations cause more complexity during both planning and operation, however, they can have a huge impact on the optimal schedule. The storage policy may vary for different intermediates, and usually it has two dimensions: the facility's capacity for storing the material in question, and the time limitations on the storage operation. This paper focuses on the second issue.

Storage time of an intermediate is sometimes limited by some physical or chemical property, which is important for a subsequent step, but fades over time, e.g., temperature, homogeneity. This is referred to as Limited-Wait (LW) case, and Zero-Wait (ZW) in its extreme, when the intermediate has to be processed immediately. In the case of absence of any such limitations, the intermediate is considered to have Unlimited-Wait (UW) policy.

The way to address LW policy depends on the method used for scheduling. In this paper we focus on the S-graph framework, and compare the various techniques that can tackle these kinds of timing limitations. The S-graph is a directed graph model of the scheduling problem, and the framework applies branch-and-bound algorithms to find the optimal schedule.

The paper is structured as follows: in Section 2, a small motivational example is shown. Section 3 provides the problem definition, and a short overview of related publications. In Section 4, the different approaches of tackling LW policy in the S-graph framework are briefly introduced. Section 5 shows empirical test results of the approaches of Section 4.

2. MOTIVATIONAL EXAMPLE

To illustrate the influence of storage time limitations on the quality of the optimal schedule, a small motivational example is presented. The example entails 3 products, A, B, and C, which are produced via 3 units: $U1$, $U2$, and $U3$. The details of the recipe are shown in Figure 1.

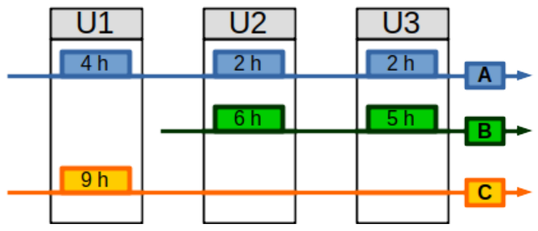


Figure 1: Motivational example

For this example, and for the rest of the paper, the objective function to be considered is the minimization of makespan, and it is also assumed, that the facility has enough storage space for any intermediate. The optimal makespan of the motivational example depends on the time limitations posed on the 3 intermediate materials. For the sake of simplicity, all of the materials are assumed to have the same time limit for this example.

The optimal makespan in the UW case is 13 hours, as shown in Figure 2. This schedule, however, needs to store the intermediates of product A. The first storage operation requires

2 hours, and the second 3 hours. If the limit on storage time is set to 2 hours, the makespan is increased to 14 hours, as shown in the second part of Figure 2.

Naturally, as the limits get lower, the makespan may increase. In the extreme case, i.e., in the case of ZW policy, the makespan is 17 hours, as shown in the third part of Figure 2.

3. PROBLEM DEFINITION AND LITERATURE OVERVIEW

Although the approaches presented in Section 4 may address a wider range of scheduling problems, for the sake of simplicity, the considered problem class has the following features:

- multipurpose recipe, i.e., recipes for products are sequential, but the order of the units are not necessarily the same
- a task may have several suitable non-identical units with different processing times
- the facility is equipped with enough capacity to store all of the intermediates
- cleaning-, transfer-, and changeover times are negligible
- all of the raw materials are available from the start
- preemption is not allowed
- the goal is to minimize the overall production time, i.e., the makespan
- for each intermediate, a non-negative upper time limit is given for its storage time, that may be 0 or infinity

The first papers to investigate zero-wait policies date back to the 70's (see e.g., [4, 16]). Zero-wait constraints are also thoroughly investigated for shop problems with more than 400 papers [1]. Batch process scheduling gained the attention of both engineers and optimization experts in the 90's and numerous methods have been presented since then to solve these industrial scheduling problems [9, 2]. The variety of the proposed approaches stretch from Mixed Integer Linear Programming (MILP) formulations [15] through state space exploration techniques (Timed Automata, Timed Petri Nets) to directed graph based approaches [17, 6].

LW constraints have been addressed in many different ways in the literature. Some MILP models can easily express them via linear constraints [14, 8], other approaches apply heuristics to solve such problem classes [5, 18], or rely on a separate branch-and-bound technique [3].

The S-graph framework was originally introduced to address problems with UW policy [17]. Since then the framework has been extended to tackle many different problem classes, some of which required LW or ZW constraints on intermediates [12]. The goal of this paper is to empirically compare different options, developed previously [10] or new, and find the most efficient one.

4. APPROACHES FOR LW POLICY WITH THE S-GRAPH FRAMEWORK

The S-graph framework uses weighted directed arcs to express timing constraints within the events represented by the vertices, that are either the starting time of tasks or the shipping of products. An arc leading from node n_i to $n_{i'}$ with the weight of $w_{i,i'}$ encodes, that the starting time of task i' must be at least $w_{i,i'}$ later than the starting of task i.

These type of arcs are used to express both the production precedences and the scheduling decisions made by the algorithms. Unlike all of these, limited-wait constraints enforce not a lower but upper bound on the starting of a task, thus, they can not immediately be expressed with the available tools of the framework. The following two subsections briefly introduce two possible extensions.

4.1 Combinatorial approach

The LW constraints enforce an upper bound on the starting time of a subsequent task in the form

$$ST_{i+1} \leq ST_i + pt_i + LW_i$$

where ST_i, ST_{i+1} are the starting times of two subsequent tasks in the production of some products, pt_i is the processing time of task i, and LW_i is the maximal time, the intermediate produced by task i can be stored. This constraint can easily be converted to the form

$$ST_i \geq ST_{i+1} + (-pt_i - LW_i)$$

This way, the constraint can be expressed by a regular S-graph arc leading from n_{i+1} to n_i, which have the negative weight of $-pt_i - LW_i$. The introduction of negative-weighted arcs, however, require slight modifications on the longest path method that is used for providing lower bound on the makespan, and to report infeasible schedules when finding cycles in the graph.

After introducing the negative "backward arcs", cycles now naturally occur in the graph, and only those with positive total weight are the sign of infeasibility. Some of the 0 weighted cycles also represent an infeasible schedule, which phenomenon is called the cross-transfer [11].

The advantage of this approach is to have only a minor overhead compared to the original algorithm. Moreover, the longest paths in the graph can be cached in a difference-bound-matrix[7], which allows quick updates and constant time lookup in the implementation. On the other hand, the bounds can be weak farther from the leafs.

4.2 Bounding LP approach

The S-graph model shows a lot of similarity with the general precedence MILP models found in the literature. Both of these approaches address the scheduling problems as assignment and sequencing decisions, thus there is a one-to-one relation between different components of the two sides. It is not the goal of this paper to detail such precedence based MILP models, however, to ease further explanations, three

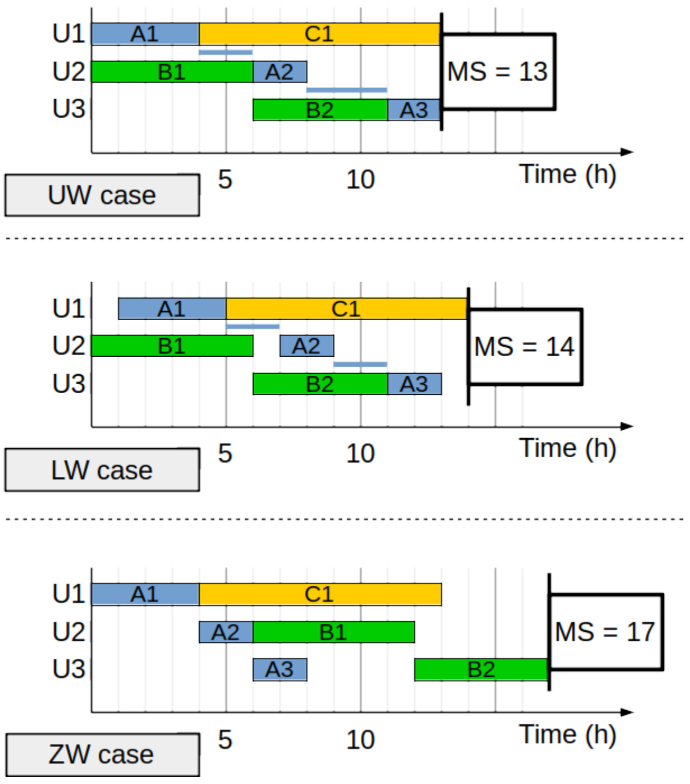


Figure 2: Optimal schedule of the motivational example for UW, LW and ZW policies

typical timing constraints are shown here.

$$ST_{i+1} \geq ST_i + \sum_{j \in J_i} pt_{i,j} \cdot Y_{i,j} \tag{1}$$

This equation expresses that the starting time of task $i+1$, that is the subsequent task of i, must start at least as much later, as the processing time of task i in the selected unit j. ST_i and ST_{i+1} are continuous variables for the starting times, and $Y_{i,j}$ is a binary assignment variable. The sequencing of tasks assigned to the same unit are expressed by:

$$ST_i \geq ST_{i'} + pt_{i',j} - M \cdot (3 - X_{i',i} - Y_{i',j} - Y_{i,j}) \tag{2}$$

where $X_{i',i}$ is the binary sequencing variable, that takes the value of 1 if i' precedes i in any unit. In such a model, the LW constraints can be easily expressed in the form:

$$ST_{i+1} \leq ST_i + \sum_{j \in J_i} pt_{i,j} \cdot Y_{i,j} + LW_i \tag{3}$$

This MILP model can be integrated into the S-graph solution algorithm the following way:

- At the root of the branch-and-bound tree, the precedence based model of the problem is generated, and the sequencing, assignment variables are relaxed to be continuous variables from the interval $[0, 1]$.
- When scheduling decisions are made by the S-graph algorithm to create child subproblems, the LP problem is copied, and the relaxed binary variables related to those decisions get fixed in the LP model of the child.
- Instead of calling the longest path algorithm to provide bound, the relaxed LP model is solved.

The advantage of this approach is better bounds close to the top of the tree, where the assignment decisions are not yet made. However, the bounding step requires much more computation, not to mention the overhead for allocating, copying, destroying the LP models in the memory. To increase the efficiency of this approach, instead of copying the LP model for each node, only one instance could be stored (per thread), and the bounding step only modifies the intervals of the decided binary variables. Moreover, a child may use the solution of the parent node with the dual-simplex algorithm.

5. EMPIRICAL RESULTS

All of the approaches mentioned in Section 4 have been implemented, and thoroughly tested. Here, we show the results for a literature example [13], which reflect our overall experience. 13 test cases were considered, and each approach ran with a 1 hour time limit. The computational results are shown in Table 1.

Although the results vary, it is obvious that the combinatorial approach with the negatively weighted arcs far outperform the LP bound based approaches. The latter two sometimes exceeded the time limit as indicated in the table. Surprisingly, the basic LP bound approach with copying

Table 1: Test results

		CPU times (s)		
Test case	Makespan (h)	Negative arcs	LP bound basic	LP bound advanced
1	118	1223.88	-	-
2	84	2.32	224.21	188.87
3	33	8.76	472.53	415.92
4	133	0.22	10.78	13.32
5	148	1.31	97.61	86.34
6	72	0.19	2.67	49.73
7	97	9.20	743.96	694.80
8	89	27.64	3009.87	2454.94
9	33	1520.99	-	-
10	153	131.23	630.68	-
11	154	497.82	942.72	634.81
12	76	2.27	160.23	441.31
13	162	75.28	-	-

the LP model, and not using the parents solution actually proved to be faster than the more sophisticated one.

6. CONCLUSIONS

Three different methods were presented for addressing LW constraints in the S-graph framework: a combinatorial technique with negatively weighted arcs, and an updated longest path algorithm, and two approaches using the relaxed LP model of a precedence based MILP formulation as bounding function. The empirical tests clearly showed, that the combinatorial approach outperforms the LP bound based techniques in all of the test cases. Thus, future extensions of the S-graph framework should rely on this technique to tackle LW or ZW constraints.

7. ACKNOWLEDGMENTS

This work was supported by the National Research, Development and Innovation Office (NKFIH), Grant no. SNN 129178.

8. REFERENCES

[1] A. Allahverdi. A survey of scheduling problems with no-wait in process. *European Journal of Operational Research*, 255(3):665–686, 2016.

[2] A. Allahverdi, E. Pesch, M. Pinedo, and F. Werner. Scheduling in manufacturing systems: new trends and perspectives. *International Journal of Production Research*, 56(19):6333–6335, 2018.

[3] Y.-J. An, Y.-D. Kim, and S.-W. Choi. Minimizing makespan in a two-machine flowshop with a limited waiting time constraint and sequence-dependent setup times. *Computers & Operations Research*, 71:127–136, 2016.

[4] J. R. Callahan. *The Nothing Hot Delay Problem int the Production of Steel*. PhD thesis, Department of Industrial Engineering, University of Toronto, Canada, 1971.

[5] A. Condotta and N. Shakhlevich. Scheduling coupled-operation jobs with exact time-lags. *Discrete Applied Mathematics*, 160(16):2370–2388, 2012.

[6] A. D'Ariano, D. Pacciarelli, and M. Pranzo. A branch and bound algorithm for scheduling trains in a railway network. *European Journal of Operational Research*, 183(2):643–657, 2007.

[7] D. L. Dill. Timing assumptions and verification of finite-state concurrent systems. In J. Sifakis, editor, *Automatic Verification Methods for Finite State Systems*, pages 197–212, Berlin, Heidelberg, 1990. Springer Berlin Heidelberg.

[8] C. Gicquel, L. Hege, M. Minoux, and W. van Canneyt. A discrete time exact solution approach for a complex hybrid flow-shop scheduling problem with limited-wait constraints. *Computers & Operations Research*, 39(3):629 – 636, 2012.

[9] M. Hegyháti and F. Friedler. Overview of industrial batch process scheduling. *Chemical Engineering Transactions*, 21:895–900, 2010.

[10] M. Hegyháti, T. Holczinger, A. A. Szoldatics, and F. Friedler. Combinatorial Approach to Address Batch Scheduling Problems with Limited Storage Time. *Chemical Engineering Transactions*, 25:495–500, 2011.

[11] M. Hegyháti, T. Majozi, T. Holczinger, and F. Friedler. Practical infeasibility of cross-transfer in batch plants with complex recipes: S-graph vs MILP methods. *Chemical Engineering Science*, 64(3):605–610, 2009.

[12] M. Hegyháti, O. Ösz, B. Kovács, and F. Friedler. Scheduling of Automated Wet-Etch Stations. *Chemical Engineering Transactions*, 39:433–438, 2014.

[13] Y. Liu and I. Karimi. Scheduling multistage, multiproduct batch plants with nonidentical parallel units and unlimited intermediate storage, 2007.

[14] Y. Liu and I. Karimi. Scheduling multistage batch plants with parallel units and no interstage storage. *Computers & Chemical Engineering*, 32(4):671–693, 2008.

[15] C. A. Méndez, J. Cerdá, I. E. Grossmann, I. Harjunkoski, and M. Fahl. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers & Chemical Engineering*, 30(6-7):913–946, 2006.

[16] M. S. Salvador. A solution to a special class of flow shop scheduling problems. In S. E. Elmaghraby, editor, *Symposium on the Theory of Scheduling and Its Applications*, pages 83–91, Berlin, Heidelberg, 1973. Springer Berlin Heidelberg.

[17] E. Sanmartí, L. Puigjaner, T. Holczinger, and F. Friedler. Combinatorial framework for effective scheduling of multipurpose batch plants. *AIChE Journal*, 48(11):2557–2570, 2002.

[18] N. Zhou, M. Wu, and J. Zhou. Research on power battery formation production scheduling problem with limited waiting time constraints. In *2018 10th International Conference on Communication Software and Networks (ICCSN)*, pages 497–501, July 2018.

Visualization of 3D Earth using GIS services

Aljaž Jeromel
Laboratory for Geometric Modelling and Multimedia Algorithms
Koroška cesta 46
Maribor, Slovenia
aljaz.jeromel@um.si

Mitja Žalik
Laboratory for Geometric Modelling and Multimedia Algorithms
Koroška cesta 46
Maribor, Slovenia
mitja.zalik@student.um.si

Matej Brumen
Laboratory for Geometric Modelling and Multimedia Algorithms
Koroška cesta 46
Maribor, Slovenia
matej.brumen@um.si

Niko Lukač
Laboratory for Geometric Modelling and Multimedia Algorithms
Koroška cesta 46
Maribor, Slovenia
niko.lukac@um.si

ABSTRACT

In this paper, a method is presented for rendering a 3D digital Earth. The method works in three steps. Firstly, the world map is partitioned into square tiles and multiple levels of detail. Then, the transformation from 3D to 2D is reversed, vertices are calculated, and connected into triangles. In the final step, the vertices are offset to model the Earth's terrain. The RAM usage and FPS of the proposed method were measured in regard to the number of GIS map layers. Experimental results have shown that the proposed method is suitable for real-time visualization of the Earth, even when multiple map layers are used.

Categories and Subject Descriptors

I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modelling;
I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism

General Terms

Computer Graphics

Keywords

3D Computer Graphics, Visualization, Digital Earth

1. INTRODUCTION

A 3D computer model of the Earth has a lot of practical applications. The motivation behind it and the technology needed for its realization were discussed by the US Senator Al Gore as early as 1998 [1]. In his speech, he also presented some practical use cases for such an application. The model could be used for political purposes, crime-fighting purposes, biodiversity preservation, climate change prediction, and educational purposes. At the time, however, the technology to realise such an application was very limited. The sub-meter precision that Gore envisioned requires a Central Processing Unit (CPU) that is capable of processing a lot of data to generate the triangles for visualization. A powerful Graphics Processing Unit (GPU) is also needed to store these triangles and visualize them in real time, while storing maps for texturing the triangles requires huge amounts of storage space (for example, if a 1m $\times$ 1m area was represented by a single pixel, 120 MB of data would be needed just to cover the equatorial line).

In 2011, Goodchild published an overview of the existing technologies related to Gore's 1998 speech [2]. He compared the functionalities of geobrowsers (for example, Google Earth) and GIS (Geographic information systems), and provided the reasons why the former were used more widely. He viewed powerful visualization as the main advantage of the geobrowsers over GIS, as GIS contained mainly information without aerial images of the Earth. The second, and also very important advantage of geobrowsers, is the ease of use, the "child-of-ten standard", which means that a child of ten years can learn to use a geobrowser within ten minutes without any previous knowledge. The GIS, on the other hand, required advanced geographical knowledge to really understand the presented data.

For detailed real-time Earth visualization a lot of data have to be processed, which is a problem, even for modern hardware. The resolution of distant data can be lowered to decrease the number of rendered vertices. Since faraway objects are displayed smaller on the screen, they appear the same to users, even if most of the details are not displayed. This concept is called the Level Of Detail (LOD). Many different approaches have been used to implement LOD for real-time rendering. Pajarola [3] proposed adaptive triangulation based on a restricted quadtree. In [4] the domain is tiled, and a discrete set of LODs is generated for each tile. However, the whole terrain should be preprocessed to avoid mesh re-triangulation at run-time. The approach with tiling is also used in [5], where adaptive rendering is used (i.e. on slower hardware less triangles are being drawn). This results in less detailed terrain, but the frame rate is not affected by the capability of the device. Another adaptive rendering method was proposed by Cai, Li and Su [6], where the terrain is divided into several blocks, which are preprocessed for faster visualization. An interesting approach to tiled visualization is also described in [7], where fractal noise displacement is used to synthesise a more detailed view than that stored. Larsen and Christensen proposed a method [8] for efficient visualization of polygonal surface data. It is based on compact regular grid representation, and requires minimal preprocessing. However, none of the previously mentioned approaches visualise the terrain planet wide. Cignoni et al. addressed planet wide visualization in the P-BDAM algorithm (together with similar algorithms described in [9, 10, 11]).

In this paper, a method is presented for rendering a 3D model of the Earth by using a Digital Terrain Model (DTM) data. The main contribution of the method is the approach to rendering the 3D digital Earth using GIS layers. The method consists of three steps. In the first step, the world map is partitioned into square tiles, which are obtained from the GIS web services. The vertices are generated and connected into triangles according to the tiles in the second step. In the final step, the vertices are offset to model the terrain. The results of the method are considered in terms of RAM (Random-Sccess Memory) usage and FPS (Frames Per Second). According to the results, the proposed method is suitable for real-time visualization of a 3D digital Earth.

This paper is structured as follows. The proposed method for visualization is presented in the next Section. The results of the proposed method are given in Section 3. The paper is summarised in Section 4.

2. PROPOSED METHOD

The description of the proposed method is divided into two parts. The tiling system of the world map in 3D is presented in the first part. The second part discusses the modelling of the 3D terrain.

2.1 Tile System

A transformation is needed to map rectangular surfaces of textures to the surface of the Earth, because its shape is close to a spheroid. The most commonly used is Mercator projection, which projects the surface of the Earth onto a cylinder with base radius of the same length as the equatorial radius of the Earth. This, of course, introduces distortions to the map, which stretch the objects that are close to the poles.

In this paper, a square map is used, partitioned into 20 layers of increasing quality. The first layer contains one texture or tile. That tile is then partitioned in a quad-tree-like fashion. That way, the second layer contains 4 tiles, the third layer contains 16 tiles, etc. Textures for the tiles are obtained from the web using the GIS Web Map Service (WMS). The textures are partitioned and prepared for tiling by the server. The vertices can then be generated to be spaced equally across a tile or the 3D world. In both cases, a transformation is applied to either map vertices to tiles, or to map tiles to vertices. The transformation is done only in the direction of the Y-axis, since the Mercator projection retains the length ratios across the same line of latitude. The transformation that maps tiles to vertices is given in Eq. 1, where v is the Y-coordinate on the texture, Y is the Y-coordinate of the vertex and $M = 2\,\mathrm{atan}\,(\sinh \pi)$.

$$v = 0.5 - \frac{\mathrm{asinh}\left(\tan\left(\frac{M}{\pi}\,\mathrm{asin}\,Y\right)\right)}{\pi} \tag{1}$$

When mapping vertices to tiles, Eq. 2 is used to calculate the Y-coordinate of the vertex, $V.Y$, where Y is the Y-coordinate on the world map in the range [-0.5, 0.5]. The X and Z-coordinates of the vertex can then be calculated from its Y-coordinate and X-coordinate on the world map using trigonometric functions.

$$V.Y = \sin\left(\frac{\pi}{M}\,\mathrm{atan}\,(\sinh(2\pi Y))\right) \tag{2}$$

All data necessary for visualization can be calculated after estimating all vertices. However, as the number of tiles grows exponentially with each level of detail, it is sensible to render just a few of them. Calculation of which tiles are needed for rendering can be done by the following procedure. Firstly, the position of the camera is normalised to the surface of the Earth. Then, the viewing direction is projected onto a plane perpendicular to the Earth at the previously normalised point according to Eq. 3, where $\vec{r}$ is the projected viewing direction, $\vec{p}$ is the vector from centre of the sphere to the position of the normalised point, and $\vec{d}$ is the viewing direction vector. The projected vector is then split into its vertical ($\vec{v}$) and horizontal ($\vec{h}$) components using Eq. 4 and 5, respectively, where $\vec{u} = (0, 1, 0)$.

$$\vec{r} = \vec{p} \times \left(\vec{d} \times \vec{p}\right) \tag{3}$$

$$\vec{v} = \vec{r}.y \tag{4}$$

$$\vec{h} = \mathrm{sgn}\,(\vec{r} \cdot (-\vec{p} \times \vec{u}))\,(-\vec{p} \times \vec{u}) \tag{5}$$

Finally, the index is calculated of the farthest visible tile. The farthest visible point on the sphere can be obtained by rotating the directional vector of the camera around an axis perpendicular to the vectors of the camera's direction and the vector between the position of the camera and centre of the sphere, then calculating the tangent point on the sphere. The angle of rotation is calculated from Eq. 6, where $\vec{l}$ is the vector of the viewing direction, $\vec{c}$ is the vector from the centre of the sphere to the camera, and r is the radius of the sphere. The tangent point $\vec{t}$ can then be calculated using Eq. 7, where $\vec{p}$ is the position of the camera, $\vec{d}$ is the direction of the camera, and r is the radius of the sphere. The Equation assumes that the centre of the sphere is located at the point $(0, 0, 0)$. If the sphere is displaced, its centre should be subtracted from $\vec{p}$ and then added to $\vec{t}$.

$$\phi = \mathrm{acos} - \frac{\vec{l} \cdot \vec{c}}{\|\vec{l}\|\|\vec{c}\|} - \mathrm{asin}\,\frac{r}{\|\vec{c}\|} \tag{6}$$

$$\vec{t} = \vec{p} + \vec{d}\left(\vec{d} \cdot \vec{p} - \sqrt{\left(\vec{d} \cdot \vec{p}\right)^2 + r^2 - \vec{p} \cdot \vec{p}}\right) \tag{7}$$

Assuming the sphere is positioned in the centre of the coordinate system with north pole oriented towards positive Y axis and Prime Meridian towards positive Z axis, the indices of the tile covering any position of the sphere can be calculated using Eq. 8 and 9, where l is the level of detail, $\vec{p}$ is the position we are calculating the tile indices for, and M is the same constant as used in the Eq. 1.

$$tx = \left\lfloor 2^{l-1}\left(1 + \frac{1}{\pi}\,\mathrm{atan}\,\frac{\vec{p}.x}{\vec{p}.z}\right)\right\rfloor \tag{8}$$

$$ty = 2^l - 1 - \left\lfloor 2^{l-1}\left(1 + \frac{1}{\pi}\,\mathrm{asinh}\left(\tan\left(\frac{M}{\pi}\,\mathrm{asin}\,\frac{\vec{p}.y}{\|\vec{p}\|}\right)\right)\right)\right\rfloor \tag{9}$$

2.2 3D Terrain

Two types of textures are used to render a 3D surface:

- Colour textures, which affect the colour of pixels, and
- Elevation textures, which are needed to displace vertices and, thus, form the 3D terrain.

Both texture types are retrieved from the GIS server (e.g. GeoServer) through WMS and the Web Coverage Service (WCS) respectively. WMS and WCS are specifications created by the Open Geospatial Consortium for requesting georeferenced data. While WMS is the most widely used Standard for retrieving map products from the GIS server, WCS can provide more information (e.g. terrain heights). Which textures are needed for a particular camera position is determined by the Tile System (see subsection 2.1). Colour textures are used in the fragment shader of the graphics pipeline (Fig. 1) to display a map, satellite image, or any geographically related data on the surface of the Earth. Elevation textures contain only one channel - elevation (E) of some point on the Earth).

In the vertex buffer of the graphics pipeline, the distance

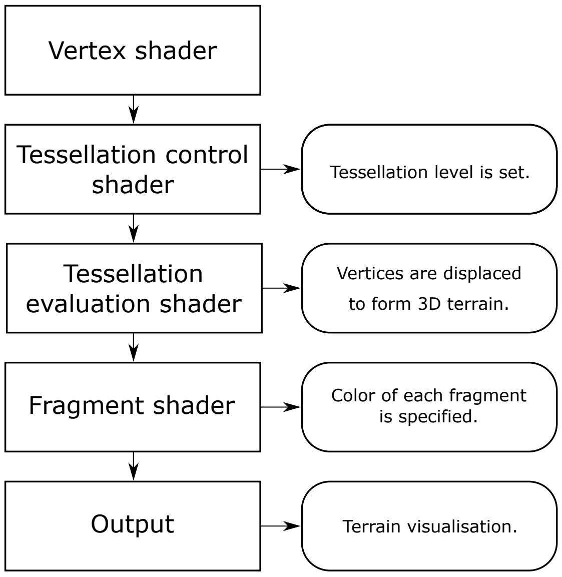

Vertex shader
Tessellation control shader
Tessellation level is set.
Tessellation evaluation shader
Vertices are displaced to form 3D terrain.
Fragment shader
Color of each fragment is specified.
Output
Terrain visualisation.

Figure 1: The outline of the graphic pipeline. On the left side, the stages of used pipeline are shown while on the right the proposed implementation is shown.

between each vertex and the centre of the digital Earth is 1. Thus, without elevation textures, the Earth is represented as a perfect sphere. On the GPU, vertices are displaced based on Eq. 10 to get new position $\vec{v'}$. The original vertex position is denoted by $\vec{v}$, and the radius of the sphere (3D Earth) is denoted by R.

$$\vec{v'} = \vec{v}\left(1 + \frac{E}{R}\right) \tag{10}$$

Displacement of vertices is realised in the tessellation stage of the graphics pipeline. Consequently, elevation textures can be received independently of colour textures, since no local preprocessing is needed for displacing vertices. Thus, the source of elevation textures can be added or removed during the programme execution. When the elevation texture is retrieved, the terrain will be elevated immediately. If the source of the elevation textures does not cover the entire Earth, only the provided area will be elevated. Furthermore, the tessellation level can be changed during visualization to change the resolution of visualized data dynamically without the need of vertex buffer recalculation on the CPU (see Fig. 2).

Since geometry is created on the GPU global memory, it

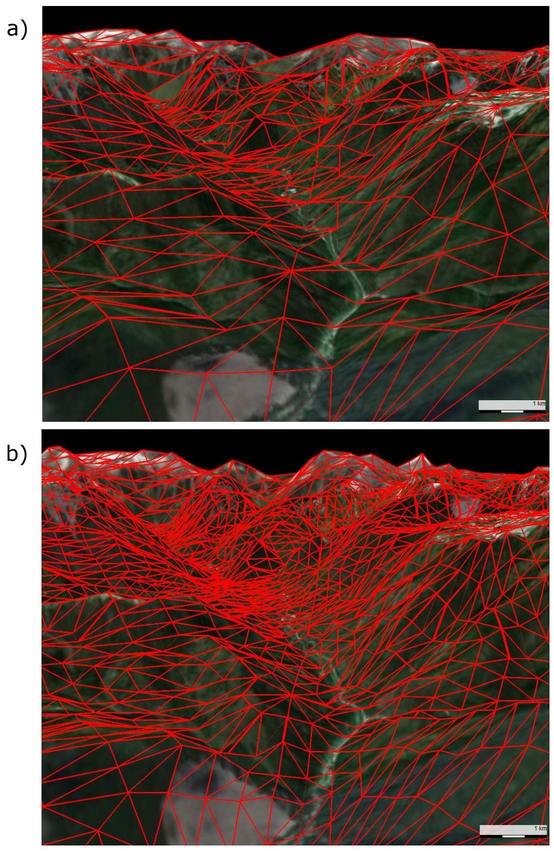


Figure 2: Display of resulting terrain with two different resolutions. In a) the middle resolution of terrain is shown (tessellation level 4), while b) shows rendered terrain in high resolution (tessellation level 8). For clarity's sake, the triangles boundaries are displayed in red.

does not exist on the host memory. Nevertheless, the position and height of a point on the surface can be retrieved from the GPU for each pixel on the screen. After obtaining the depth component of the corresponding pixel, unprojecting can be done to obtain fragment coordinates in the world space ($\vec{p'}$). After that, Eq. 11 is used to get elevation in metres:

$$E = (\|\vec{p'}\| - 1)R \tag{11}$$

Eq. 12 and Eq. 13 are used to retrieve longitude (lon), latitude (lat).

$$lon = \frac{180}{\pi} \operatorname{atan} \frac{\vec{p'}.x}{\vec{p'}.z} \tag{12}$$

$$lat = \frac{360}{\pi^2} \operatorname{asin}\left(\frac{\vec{p'}.y}{\|\vec{p'}\|}\right) \operatorname{atan}(\sinh \pi) \tag{13}$$

3. RESULTS

The results of the proposed method are presented in this section. The RAM (Random Access Memory) usage was measured and FPS (Frames Per Second) was calculated when rendering the Earth with different numbers of GIS map layers. For calculating the FPS, the average time needed to process and draw each frame on the screen was measured, and then inverted to obtain the FPS. Measurements were done on a computer with the following configuration: Intel Core i5-3570K 3.40 GHz CPU, GeForce GTX 1060 6GB, 32 GB of RAM, and Windows 10 Education operating system. The graph of average results is shown in Fig. 3. As seen,

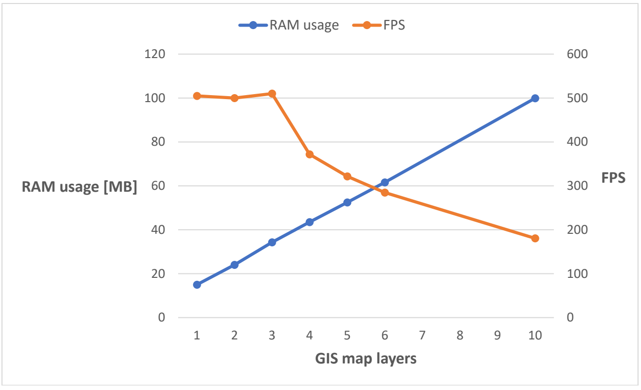


Figure 3: RAM usage and FPS when rendering different number of GIS map layers with the proposed method

the RAM usage increases linearly with the number of GIS map layers, with each layer increasing the usage by approximately 9 MB. On the other hand, FPS tends to decrease when more than three layers are rendered. This happens because FPS is inversely proportional to rendering time. The rendering time increases by approximately half a millisecond each time a layer is added, which results in lower and lower FPS drops when more layers are added.
The results have shown that the proposed method is suitable for real-time rendering of a 3D Earth on modern personal computers. Therefore, the proposed method could be used as a standalone application (i.e. a geobrowser), or as a visualization option for larger applications.

4. CONCLUSION

In this paper, a new method is proposed for tile-based, planet wide terrain visualization. At first, the position and visual quality of needed tiles are calculated, based on camera position. Then, two types of tiles are retrieved from the GIS server in the form of textures. Colour textures define the colour of each pixel in the viewing frustum. Elevation textures affect the elevation of each vertex. Since terrain is formed on the GPU, the proposed method needs almost no preprocessing.
The performed experiments proved that adding additional layers does not decrease performance significantly, and that modern GPUs can visualise several layers real-time. Therefore, the proposed method could be used in multiple applications, such as visualization of maps with multiple layers (e.g. visualization of land usage, annual weather analysis, etc.). For future work, the method could be improved to support separate elevation sources for each colour source. With this feature, several completely different terrains could be visualised at the same time.

5. ACKNOWLEDGEMENTS

This research was supported by Research Programme P2-0041 from the Slovenian Research Agency and the Dewesoft d.o.o. company.

6. REFERENCES

[1] A. Gore. The digital earth: Understanding our planet in the 21st century. *Australian Surveyor*, 43(2):89–91, June 1998.
[2] M. F. Goodchild. The use cases of digital earth. *International Journal of Digital Earth*, 1(1):31–42, 2008.
[3] R. Pajarola. Large scale terrain visualization using the restricted quadtree triangulation. In *Proceedings of the IEEE Visualization 98*, pages 19–26,515. IEEE, October 1998.
[4] J. Schneider and R. Westermann. Gpu-friendly high-quality terrain rendering. *Journal of WSCG*, 14(1-3):49–56, February 2006.
[5] J. Pouderoux and J.-E. Marvie. Adaptive streaming and rendering of large terrains using strip masks. In *Proceedings of the 3rd International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*, GRAPHITE '05, pages 299–306, New York, NY, USA, 2005. ACM.
[6] X. Cai, J. Li, and Z. Su. Large-scale terrain rendering using strip masks of terrain blocks. In *2008 7th World Congress on Intelligent Control and Automation*, pages 1768–1772, June 2008.
[7] F. Losasso and H. Hoppe. Geometry clipmaps: terrain rendering using nested regular grids. *ACM Transactions on Graphics (TOG)*, 23(3):769–776, August 2004.
[8] B. D. Larsen and N. J. Christensen. Real-time terrain rendering using smooth hardware optimized level of detail. *Journal of WSCG*, 11(2):282–289, August 2004.
[9] P. Cignoni, F. Ganovelli, E. Gobetti, F. Marton, F. Ponchio, and R. Scopigno. Bdam - batched dynamic adaptive meshes for high performance terrain visualization. *Computer Graphics Forum*, 22(2):505–514, November 2003.
[10] P. Cignoni, F. Ganovelli, E. Gobbetti, F. Marton, F. Ponchio, and R. Scopigno. Planet-sized batched dynamic adaptive meshes (p-bdam). In *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, VIS '03, pages 20–, Washington, DC, USA, 2003. IEEE Computer Society.
[11] E. Gobbetti, F. Marton, P. Cignoni, M. di Benedetto, and F. Ganovelli. C-bdam - compressed batched dynamic adaptive meshes for terrain rendering. *Computer Graphics Forum*, 25(3):333–342, December 2006.

A simulator to study the stability of network centrality measures

Orsolya Kardos
University of Szeged Institute of Informatics
P.O. Box 652
H-6701 Szeged, Hungary
kardoso@inf.u-szeged.hu

András London[*]
University of Szeged Institute of Informatics
P.O. Box 652
H-6701 Szeged, Hungary
Poznań University of Economics, Department of Operations Research
london@inf.u-szeged.hu

Tamás Vinkó
University of Szeged Institute of Informatics
P.O. Box 652
H-6701 Szeged, Hungary
tvinko@inf.u-szeged.hu

ABSTRACT

Measuring nodes' importance in a network and ranking them accordingly is a relevant task regarding many applications. Generally, this measurement is done by a real-valued function that evaluates the nodes, called node centrality measure. Nodes with the largest values by a centrality measure usually give the highest contribution in explaining some structural and functional behavior of the network. The stability of centrality measures against perturbations in the network is of high practical importance, especially in the analysis of real network data that often contains some amount of noise. In this paper, by utilizing a simulator we implemented in R, a formal definition of stability introduced in [13] and various perturbation methods are used to experimentally analyze the stability of some commonly used node centrality measures.

Keywords

Network science, Centrality measures, Stability, R language

1. INTRODUCTION

In a complex network, being social (e.g. Facebook friendship), economical (e.g. international trade), biological (e.g. protein-protein interaction) or technological (e.g. transportation) network, the position of the nodes in the topology of the underlying graph is of central importance. Central nodes in this graph topology often have major impact, whereas peripheral nodes usually have limited effect on the structure and functioning of the network. Thus, identifying the central and most important nodes helps in better understanding the networks from many different perspectives. Node centrality measures are metrics designed to identify these important nodes. However, the importance of a node can be interpreted in many different ways, therefore, depending on the applications, many centrality measures have been developed and effectively applied in various domains [7]. The most commonly used centrality measures are degree [11, 14], closeness [1, 12], eigenvector [2], betweenness [8], PageRank [4] and HITS [10]. Degree centrality measures the importance of a node simply by the number of its neighbors. Closeness centrality shows the average shortest path length from the node to every other node in the network. Eigenvector centrality, and similarly PageRank, of a node is computed (iteratively) as a function of the importance of its neighbors. Betweenness centrality measures the relative number of shortest paths in the network that go through a node.

The stability of centrality measures has often been investigated in an empirical way by comparing the network with one obtained by modifying the original one according to some randomisation procedure [3, 6, 15]. Recently Segarra and Ribeiro gave a formal definition for the stability of centrality measures and proved that degree, closeness and eigenvector centrality are stable whereas betweenness centrality is not [13]. In this work we experimentally investigate the stability of degree and eigenvector centrality measures on various data sets, and under two different perturbation processes. By doing so we introduce our simulation environment which is implemented in R and available online as an interactive tool.

This paper is organized as follows. In Section 2 we will briefly discuss the definition of stability for centrality measures and introduce the main notations used in the paper. In Section 3 we will present a simulation environment written in R and describe the data sets used in our experiments. In Section 4 we will describe the two perturbation processes, discuss our results and draw some succinct conclusions.

2. NODE CENTRALITY AND STABILITY

Let us consider a network represented by a graph $G = (V, E)$, where V is the set of nodes and E is the set of edges (i.e. pairs of nodes) of the network. Centrality measure is a real-valued function $C^G : V_G \to \mathbb{R}_{\geq 0}$, that assigns a non-negative number to each node of network G. Here we will

[*]The author was partially supported by the National Research, Development and Innovation Office - NKFIH, SNN-117879

not give the formal definitions of the investigated centrality measures that can be found e.g. in [7]. We use the definition of stability introduced in [13] as follows. A node centrality measure C is said to be stable if

$$|C^G(x) - C^H(x)| \leq K_G \cdot d(G, H) \qquad (1)$$

holds for every node $x \in V$, where G and H are two graphs over the same node set V, K_G is a constant, and $d(\cdot, \cdot)$ is a distance function between two graphs.

The definition says that a centrality measure is stable if the maximum change in node centrality is bounded by a constant times the distance of the two graphs. This constant value must be universal to any perturbed version of the initial graph. Furthermore, the constant value does not depend on the presence of normalization of centrality values. Note that the definition is similar to the definition of Lipschitz-continuity, applied in a discrete space. In order to make the above inequality meaningful a graph distance $d : G \times H \to \mathbb{R}_{\geq 0}$ should be specified. Here, the distance of two graphs with identical node set V is defined as

$$d(G, H) = \sum_{i,j} |A^G_{ij} - A^H_{ij}|,$$

where A denotes the (weighted) adjacency matrix of the network.

It is of empirical interest to study how graph H occurs from a given graph G and how it affects the constant K_G in formula (1). In Section 3 different graph perturbation methods using various input graphs and data sets in order to examine the ranges of K_G are discussed.

2.1 Theoretical values in stability concepts

Segarra and Ribeiro showed that using the stability concept (1) the degree, closeness and eigenvector centrality measures are stable, whereas betweenness centrality is not [13]. The theoretical K_G values for the three stable measures were determined. Given a directed and weighted graph G, $K_G = 1$ for degree centrality. This is because the distance of the two adjacency matrices will be at least the maximum difference of the degree centrality value. Furthermore this theoretical value for undirected weighted graphs can be reduced to 1/2 due to the symmetry of the adjacency matrices. For closeness centrality it was proved that the theoretical bound K_G is equal to the number of nodes, hence it is not a universal constant. The eigenvector centrality is stable and the constant K_G can be computed as $4/(\lambda_1 - \lambda_2)$, where λ_1 and λ_2 are the greatest and second greatest eigenvalue of the adjacency matrix of graph G, respectively.

Although there exist some theoretical results for the constant K_G, it could still be interesting to analyze its actual value in real networks under natural perturbation scenarios. In the next section we describe our simulation environment and data sets used for experimental analysis.

3. SIMULATION ENVIRONMENT

R is an open-source programming language developed by the R Foundation and can be widely used for statistical computations and representations. The functions which are mainly used in our project for graph manipulation and related computations, generating synthetic graphs and graph visualization are part of the `igraph` package. We also use the `plotly` library which is an online analytical and data visualization tool. It can be easily integrated in various developer environments, thus combined with R can be widely used for data visualization.

With the help of these tools we designed and implemented a versatile simulation environment that we use to perform our experiments. The simulator can handle various network data structures, while the output of a simulation can be various plots, data tables, statistics depending on the user defined parameters. A version of the simulator with limited functionality that uses the data as input as discussed below is available online at:

`https://kardosorsi.shinyapps.io/stability`

The interested readers are cordially invited to visit our website and try out different experiments. The full version of the simulator is available upon request.

3.1 Data sets

We have performed a wide-range of experiments on various synthetic and real data sets using different types of perturbations. In the following two experiments are elaborated in more detail.

S&P 500

Firstly, a correlation based financial graph was used. The main motivation behind using stock data was to obtain the perturbation method directly from real-life processes. The experiments were performed using the daily closing prices of stocks of the S&P 500 in the period of 01/01/1995 – 31/12/2018, including the assets of 330 leading U.S. companies[1]. We used a time-window of 200 days to construct correlation matrices from stock return time series on that interval with starting points $T_0 = 01/01/1995$, $T_k = T_0 + k\Delta T$ with $\Delta T = 50$, $k = 1, 2, \ldots$. This way we obtained 116 consecutive networks, with the fixed set of 330 nodes and weighted edges represent the correlation coefficient of each pair of assets on the corresponding time interval. Here, the changes in edge weights between each consecutive network pairs simulates the perturbation process.

Cooper-Frieze graph process

Secondly, we implemented the Cooper-Frieze graph evolution process based on a general model of web graphs proposed in [5]. That is a general model of a random graph process to generate a graph of power-law degree distribution as follows. Starting from an initial graph G_0 at time $t = 0$, the process evolves randomly by the addition of new nodes and/or edges at each time step $t = 1, 2, \ldots$. The following six parameters of the process provide a high-level of freedom in graph generation. With probability $\alpha \in [0, 1]$ and $1 - \alpha$ a new node is created or an existing node generates edges, respectively. With probability $p = (p_i : i \geq 1)$ a new node generates i edges. For new nodes, with probability $\beta \in [0, 1]$ the terminal node of a new edge is made uniformly at random and with $1 - \beta$ according to degree (i.e. new edges are

[1]We selected those assets from the $S\&P$ 500 list that were complete in the considered time period.

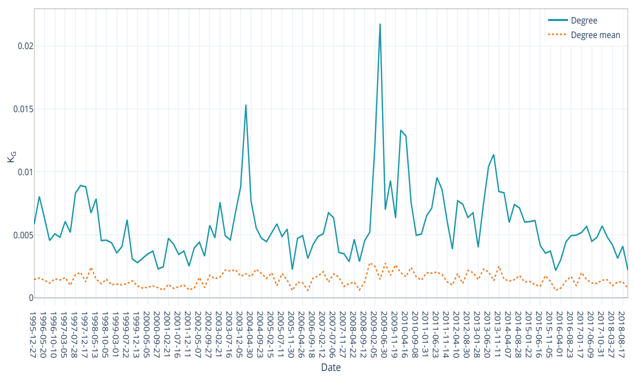


Figure 1: K_G constant values for degree centrality measure during the perturbation simulation.

preferentially attached). If an already existing node generates an edge, where the number of edges given by probability $q = (q_i : i \geq 1)$, the initial node is selected uniformly with probability δ and according to the degree with $1 - \delta$. The parameter γ has similar role for existing nodes as β had in the case of new nodes. Using this process we are able to simulate a graph perturbation process. The initial graph (at time $t = 0$) can be set as an input parameter and then in every time step $t = 1, 2, \ldots$ a new (perturbed) graph is created by the evolutionary graph process.

4. RESULTS AND DISCUSSION

Two main perturbation categories are examined. The first category is the *graph structure perturbation* that can be raised from real-life data (like stock correlations) or synthetic perturbation obtained by rewiring edges selected uniformly at random. The other group is raised from *graph evolution*. Here we will present our experiments on the *S&P* 500 data set for the structure perturbation and on Cooper-Frieze networks for the graph evolution. Results are shown on consecutive graphs as discussed in Section 3. During our experiments reported here the degree and eigenvector centrality measures were considered[2].

Graph structure perturbation. At the global level, an interesting result is provided by the behavior of the K_G constant value regarding both degree and eigenvector centrality measures over time, see Figure 1 and Figure 2, respectively. The mean values for centrality C are calculated as

$$\frac{1}{|V|} \sum_{x \in V} |C^G(x) - C^H(x)|. \tag{2}$$

We can observe that for both centrality measures are very stable, only very slight changes in their values are observed. Interestingly, these changes happen in periods of crisis. The increases around 2004, 2007-2008 and 2010-2011 can be noticed. The 2007-2008 period can be associated with the Lehman Brothers failure, whereas the 2010-2011 may reflect the Sovereign debt crisis. It is a well-known stylized fact in finance that assets correlation increases in times of financial distress. Note that these actual K_G values are way lower than their theoretical bounds.

[2]Note that a more detailed presentation of our results will be part of a paper planned to be published later.

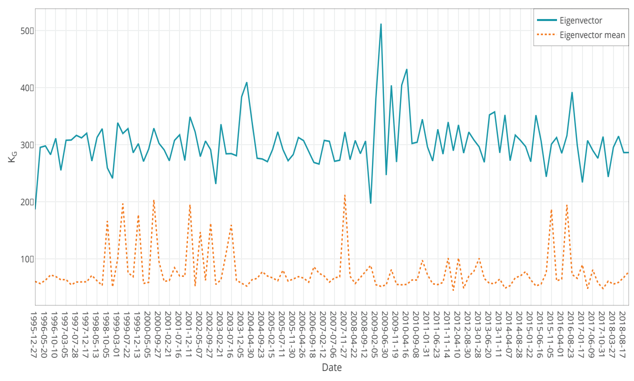


Figure 2: K_G constant values for eigenvector centrality measure during the perturbation simulation.

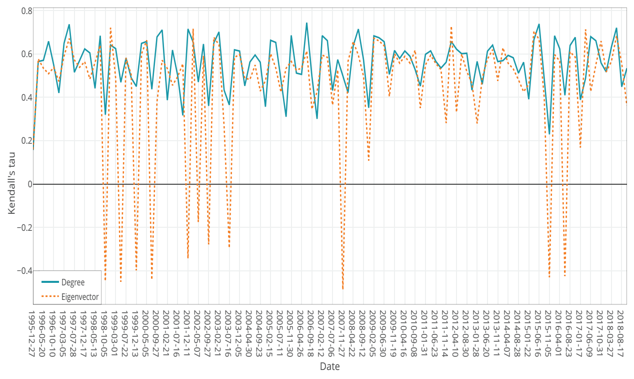


Figure 3: Kendall's tau coefficient between rank by different centrality measures during perturbation

The other interesting aspect in analyzing the stability of the different network centrality measures is the order or ranking provided by the metrics. The Kendall rank correlation coefficient [9] is used to measure the ordinal association between two measured quantities. The coefficient results in high value when observations have a similar rank (i.e. relative position label of the observations within the variable: 1st, 2nd, 3rd, etc.) between the two variables. The simulator can be parametrized in order to visualize the correlation between the order by centrality measures for the different measures respectively. Therefore it is possible to analyze the correlation between the two rank vectors during the graph perturbation procedure. On Figure 3 the Kendall correlation coefficients are reported. The degree centrality stays quite stable in the range of $0.35 - 0.7$, whereas the eigenvector centrality shows some seemingly radical changes over time. We can observe that these extreme changes in ranking shown on Figure 3 are related to the higher K_G constant values regarding the average change in centrality measures presented on Figure 2.

Graph evolution. The other aspect that we wanted to study in our experiments was the graph perturbation caused by some evolutionary process. The concept behind this was that studying the maximum of the difference in centrality measures during graph evolution can be an interesting ap-

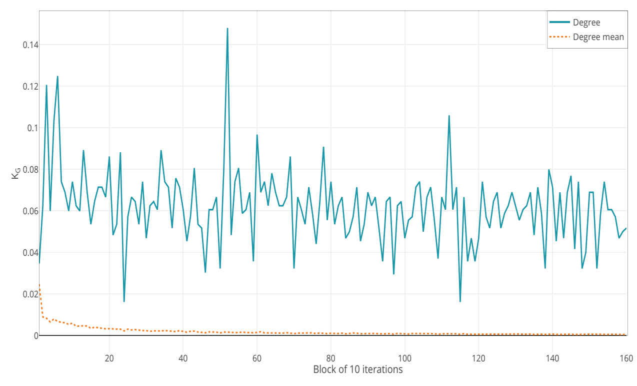


Figure 4: K_G constant values for degree centrality measure during the graph evolution process

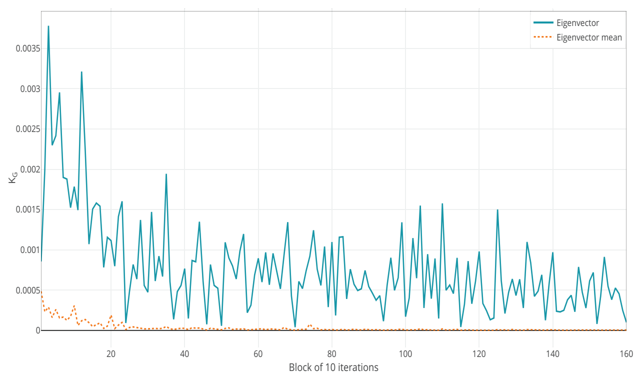


Figure 5: K_G constant values for eigenvector centrality measure during the graph evolution process

proach regarding many real-life applications. The perturbation behind evolution relies on the fact that in these networks new vertices can connect to the initial graph with one or more edges, also new edges can appear between existing nodes in the network.

For these experiments the perturbed versions of the initial graph were provided by the Cooper–Frieze graph process. In the reported results a graph of two nodes connected with an edge as initial input graph was used. We started to measure the centrality stability values after the 100th iteration by blocks of ten iterations. Thus, at the end of an iteration block consisting of 10 time steps t, a perturbed graph is produced with new nodes and edges compared to the graph from the previous block.

As it can be seen on Figures 4 and 5 the empirical values of K_G bound are about one order magnitude higher than those for the $S\&P$ dataset. Note that they are still much lower than their theoretical values and they show only slight fluctuation. Moreover, the mean values (calculated as (2) converges to zero by the growth of the size of the network. Similar convergence can be noticed on Figure 6 regarding the Kendall's correlation which shows the evidence that even the nodes ranking remain practically unchanged during the graph evolution process.

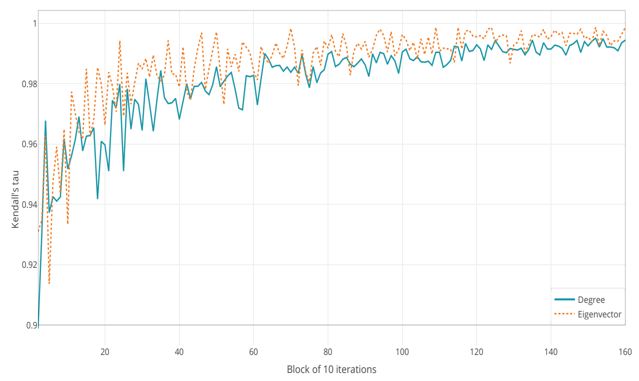


Figure 6: Kendall's tau coefficient between rank by different centrality measures during the graph evolution process

Acknowledgments

The project has been supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002).

5. REFERENCES

[1] M. A. Beauchamp. An improved index of centrality. *Behavioral Science*, 10(2):161–163, 1965.
[2] P. Bonacich. Factoring and weighting approaches to status scores and clique identification. *Journal of Mathematical Sociology*, 2(1):113–120, 1972.
[3] S. P. Borgatti, K. M. Carley, and D. Krackhardt. On the robustness of centrality measures under conditions of imperfect data. *Social Networks*, 28(2):124–136, 2006.
[4] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.
[5] C. Cooper and A. Frieze. A general model of web graphs. *Random Structures & Algorithms*, 22(3):311–335, 2003.
[6] E. Costenbader and T. W. Valente. The stability of centrality measures when networks are sampled. *Social Networks*, 25(4):283–307, 2003.
[7] K. Das, S. Samanta, and M. Pal. Study on centrality measures in social networks: a survey. *Social Network Analysis and Mining*, 8(1):13, 2018.
[8] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.
[9] M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.
[10] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
[11] U. Nieminen. On the centrality in a directed graph. *Social Science Research*, 2(4):371–378, 1973.
[12] G. Sabidussi. The centrality index of a graph. *Psychometrika*, 31(4):581–603, 1966.
[13] S. Segarra and A. Ribeiro. Stability and continuity of centrality measures in weighted graphs. *IEEE Transactions on Signal Processing*, 64(3):543–555, 2015.
[14] M. E. Shaw. Group structure and the behavior of individuals in small groups. *The Journal of Psychology*, 38(1):139–149, 1954.
[15] B. Zemljič and V. Hlebec. Reliability of measures of centrality and prominence. *Social Networks*, 27(1):73–88, 2005.

An ILP Formulation for a Variant of the University Timetabling Problem [*]

Martin Milanič
University of Primorska
Koper, Slovenia
martin.milanic@upr.si

Nevena Pivač[†]
University of Primorska
Koper, Slovenia
nevena.pivac@iam.upr.si

Jernej Vičič
University of Primorska
Koper, Slovenia
jernej.vicic@upr.si

ABSTRACT

We consider a variant the university timetabling problem, the problem of assigning courses to time intervals with respect to certain conditions. We define a natural generalization of the actual timetabling problem for the Faculty of Mathematics, Natural Sciences and Information Technologies at the University of Primorska. We develop a mathematical model based on integer linear programming for solving this NP-complete problem. The model is implemented using programming language Zimpl and evaluated using Gurobi. A timetable representing the result of the implementation is compared with the one made by hand.

Keywords

university timetabling, integer linear programming, mathematical modelling

1. INTRODUCTION

Research considering the timetabling problems started during the 1950s and until now there are many papers considering various timetabling problems (see, e.g., [6, 9, 16, 17, 18]). One of the most basic variants of the problem known as TIMETABLE DESIGN is defined in monograph by Garey and Johnson [10, SS19]. The problem is known to be NP-complete.

In this work we consider a variant of the university timetabling problem, that is, the problem of scheduling a sequence of teaching sessions involving lecturers and students in a predetermined period of time, normally within a week, while satisfying a set of constraints [17]. The problem can have additional constraints, which are desired to be satisfied, but do not necessarily have to hold. For that reason, constraints are divided into two groups: hard constraints and soft constraints. Hard constraints must be satisfied by a feasible timetable, while soft constraints represent requirements that are desirable to be satisfied, but their violation has no influence to the feasibility of the solution. Every institution has its own set of constraints that should be satisfied when constructing a timetable, so for that reason a general solution for university timetabling problems does not exist. There are some commercial solutions for the university timetabling problem (see, for example, [2, 3]). However, since there is no general solution for the problem, models developed in commercial solutions often have to be adapted in order to satisfy the conditions of a specific institution. Differences may be large, and their modeling and implementation represent a difficult and time-consuming process.

An overview of the computational complexity of a number of university timetabling problems can be found in [13]. For example, timetabling problems concerning just time slots and courses, involving lecturers, students, and an unbounded number of classrooms of unlimited capacity can be solved in polynomial time. Furthermore, there are examples of problems concerning lectures of the same length for which the number of steps needed for solving the problem is significantly reduced in comparison with the same problems having the lectures of distinct length [7].

One of the first ideas for modelling a timetabling problem using integer linear programming was developed for a school timetable during the 1960s by Lawrie [12]. After that the number of papers presenting similar models grew rapidly. Nowadays, powerful software is available for solving integer linear programs (ILPs), so ILP is again one of the main approaches for solving combinatorial optimization problems, including timetabling problems. There are various models in literature, depending on constraints and preferences of corresponding institutions. One of them is available in papers by Daskalaki et al. [8] and Daskalaki and Birbas [7], where a huge set of constraints is represented using linear inequalities. A nice summary of types of constraints used in the literature is given by Aizam and Caccetta (see [5]), while a good description of elements of objective function is available in a paper by Pereira and Costa (see [15]). One of the difficulties when using this approach is a big number of variables and constraints for larger instances of problem. Also, it is often not trivial to model some very specific constraint, which is not part of requirements at other institutions, and sometimes at all to understand the timetable from the obtained solution, which is typically a binary vector of large dimension.

Our contribution. Although the university timetabling problem is solved for many institutions (see e.g. [5, 7, 8]), it seems that these solutions cannot be easily adapted for

[*]Supported in part by the Slovenian Research Agency (I0-0035, research program P1-0285 and research projects J1-9110, N1-0102, and a Young Researchers Grant), the European Commission (InnoRenew CoE project, Grant Agreement #739574, under the Horizon2020 Widespread-Teaming program) and the Republic of Slovenia (Investment funding of the Republic of Slovenia and the European Union of the European regional Development Fund).

[†]Corresponding author.

an arbitrary new institution. Based on the description of the teaching process at the proposed institution, we introduce the Famnit Timetable Design problem, that is, the timetabling problem for the Faculty of Mathematics, Natural Sciences and Information Technologies at the University of Primorska (UP FAMNIT). This problem is NP-complete, see [14]. We develop in Section 3 an integer linear programming model for the problem. The model is implemented with real data input from the Spring semester of academic year 2016/17 using the programming software Zimpl (see [11]) and Gurobi Optimizer (see [1]). Section 4 contains results of implementation.

2. THE PROBLEM

In order to model a timetabling problem for some concrete institution, we have to describe rules and requirements of the institution that are relevant for the timetable. In this work we construct a timetable for five working days. We introduce a number of soft constraints. They are a measure for the quality of timetable and represent a part of the objective function, but have no influence to the existence of a feasible solution. We describe them in Section 3.3. The Famnit Timetable Design problem is formulated as a decision problem that checks if there exists a feasible solution of the system, i.e., a timetable that satisfies all the hard constraints.

Famnit Timetable Design
Instance: a finite set D of *days*; a finite set T of *time slots* (d, h) (day, hour), linearly ordered with respect to the time line within a 5-day week; we define addition in T so that given a time slot t and a number $i \in \mathbb{N}$, time slot $t' = t + i$ is defined as a time slot being the $(t + i)$-th element of the linear order of set T; for each $d \in D$; a finite set $T_d \subseteq T$ of time slots at day $d \in D$; a finite set M of *meetings*; a finite set S of *student groups*; a finite set L of *lecturers*; a finite set R of *rooms*; a finite set K of *locations*; subsets $T_\ell \subseteq T$ and $T_m \subseteq T$ of available hours for each lecturer $\ell \in L$ and meeting $m \in M$, respectively (for every meeting the lecturer is known, so T_m depends on lecturer's availability); a subset $T_r \subseteq T$ of available hours for each room $r \in R$; a subset $T_{AM} \subseteq T$ of morning time slots; subsets $M_s \subseteq M$ and $M_\ell \subseteq M$ of meetings incident with each student subgroup $s \in S$ and lecturer $\ell \in L$, respectively; a subset $R_m \subseteq R$ of available rooms for each meeting $m \in M$; a subset $M_k \subseteq M$ of meetings that take place at location $k \in K$ (in our case $k \in \{1, 2\}$); for each lecturer $\ell \in L$, the maximum number $\rho_\ell \in \mathbb{N}$ of hours ℓ can teach per day; a finite set N of *parts* of a day (e.g., morning, noon, evening,...); the set $T_n \subseteq T$ of all time slots at n-th part of day d, over all $d \in D$; a set $S' \subseteq S$ of student subgroups that can only have lectures within a single part of a day; a set $S_{\mathrm{Ext}} \subseteq S$ of student groups consisting of students of external programs; a set $G \subseteq M \times T \times R$ of pre-scheduled triples; a set $F \subseteq M \times T \times R$ of unacceptable triples; for any meeting m there is a vector p_m, with element $p_m(i) = k$, if a block of duration i of meeting m has to be repeated k times per week; for any meeting m, the set H_m of all block lengths appearing in the division of meeting m, that is, $H_m = \{i \mid p_m(i) \neq 0\}$.

Question:
Is there a timetable that schedules all meetings, that is, a function $f : M \times T \times R \to \{0, 1\}$ (where $f(m, t, r) = 1$ means that meeting m is assigned to time slot t and room r) that schedules the desired number of hours of all meetings and satisfies certain constraints. Due to space limitation, constraints are presented in Section 3 (see also [14, Sec. 4.2]).

3. THE ILP FORMULATION

In this section we describe an integer linear programming model for the problem.

3.1 Variables of the ILP

There are three sets of binary variables. For every triple of a meeting $m \in M$, a time slot $t \in T$, and a room $r \in R_m$ that is acceptable for that meeting, there is one corresponding variable $x_{m,t,r}$. This variable will take value 1 if meeting m is scheduled at time slot t in classroom r, and 0 otherwise. For every triple of a meeting $m \in M$, a time slot $t \in T$ and a predefined length $i \in H_m$ of individual blocks of meeting m we define a variable $y_{m,t,i}$.

The variable will take value 1 if time slot t is the first appearance of i consecutive hours of m, and 0 otherwise. In the last set of variables we have the so called z-variables, auxiliary variables for modeling some hard and soft constraints. For each constraint type p and the corresponding index set I_p, we define a variable $z_{p,i}$ for every $i \in I_p$. These variables appear in the modeling of hard constraints of type F (Section 3.2) and soft constraints of types S_2 and S_3 (Section 3.3).

3.2 Constraints of the ILP

There are six types of constraints.
A) Every meeting has to be assigned to available resources. In this group we have three types of constraints: lecturers cannot have lectures at unacceptable time slots, classrooms can only be used at specified time slots, every meeting has to take place in an acceptable classroom. The last constraint is satisfied by variable definition. The remaining two yield the following linear equations:

$$\begin{aligned} \textstyle\sum_{m\in M_\ell} \sum_{t\in T\setminus T_m} \sum_{r\in R_m} x_{m,t,r} &= 0, \quad \forall \ell \in L, \\ \textstyle\sum_{m\in M_r} \sum_{t\in T\setminus T_r} x_{m,t,r} &= 0, \quad \forall r \in R. \end{aligned}$$

B) Overlapping is not permitted. In this group we have the following constraints: for every student group at most one meeting and one classroom can be assigned to every teaching period, every member of the teaching staff shall be assigned at most one meeting and one classroom at a time, every classroom can be assigned to at most one meeting at a time. We model them with the following linear inequalities:

$$\begin{aligned} \textstyle\sum_{m\in M_s} \sum_{r\in R_m} x_{m,t,r} &\leq 1, \quad \forall s \in S, \forall t \in T, \\ \textstyle\sum_{m\in M_\ell} \sum_{r\in R_m} x_{m,t,r} &\leq 1, \quad \forall \ell \in L, \forall t \in T, \\ \textstyle\sum_{m\in M} x_{m,t,r} &\leq 1, \quad \forall r \in R, \forall t \in T. \end{aligned}$$

C) Timetable has to be complete.
C_1) All meetings in the curriculum of each student subgroup should be in the timetable and in the right amount of teaching periods, with respect to weekly duration:

$$\sum_{t\in T} \sum_{r\in R_m} x_{m,t,r} = \sum_i p_m(i) \cdot i \quad \forall m \in M.$$

C_2) A meeting m of duration $i \in H_m$ has to start and finish at the same day, so some variables $y_{m,t,i}$ are defined to have

2

value 0:

$$y_{m,t,i} = 0\, \forall m \in M,\, \forall i \in H_m,\, \forall t = (d,h) \in T : h > \tau - i + 1\,,$$

where τ represents the number of time slots in a day.

C_3) Given a meeting m, at most one time slot can be the first appearance of m in a single day:

$$\sum_{i \in H_m} \sum_{t \in T_d} y_{m,t,i} \leq 1, \quad \forall m \in M,\, \forall d \in D.$$

C_4) A given meeting m of duration i (where i is the index of a nonzero element of vector p_m) has to appear exactly $p_m(i)$ times per week (i.e., in $p_m(i)$ days). All such indices i are contained in H_m, so we have:

$$\sum_{t \in T} y_{m,t,i} = p_m(i), \quad \forall m \in M,\, \forall i \in H_m.$$

C_5) If a course m of duration i is assigned at day d, it has to be assigned to exactly i hours:

$$i \cdot \sum_{t \in T_d} y_{m,t,i} \leq \sum_{r \in R_m} \sum_{t \in T_d} x_{m,t,r}, \quad \forall m \in M, \forall d \in D, \forall i \in H_m.$$

C_6) Appearances of meeting m of duration i in a single day should be consecutive:

$$y_{m,t,i} \leq \textstyle\sum_{r \in R_m} x_{m,t+j,r}$$
$$\forall t = (d,h) \in T, \forall m \in M, \forall i \in H_m, \forall j \in \{0, \ldots, i-1\}.$$

C_7) All consecutive hours of one meeting should take place in the same classroom:

$$x_{m,t,r} + x_{m,t+1,r'} \leq 1\, \forall m \in M, \forall t \in T,\, \forall r, r' \in R_m \text{ s.t. } r \neq r'.$$

D) Pre-scheduled meetings:

$$x_{m,t,r} = 1, \quad \forall (m,t,r) \in G.$$

E) Every lecturer ℓ can have at most ρ_ℓ time slots of teaching obligations per day:

$$\sum_{t \in T_d} \sum_{m \in M_\ell} \sum_{r \in R_m} x_{m,t,r} \leq \rho_\ell, \quad \forall \ell \in L, \quad \forall d \in D.$$

F) Student requirements. Students of external interdisciplinary programs should have lectures just in the morning, or in the evening, but not both. We define the variable $z_{F,s,d}$ to have value 1 if condition F is violated for parameters $s \in S, d \in D$; and 0 otherwise.

$$\sum_{m \in M_s} \sum_{t \in T_d \setminus T_{AM}} \sum_{i \in H_m} y_{m,t,i} \leq 2 \cdot z_{F,s,d}, \forall d \in D, \forall s \in S_{\text{Ext}},$$

$$\sum_{m \in M_s} \sum_{t \in T_d \setminus T_{AM}} \sum_{i \in H_m} y_{m,t,i} \leq 2 - 2z_{F,s,d}, \forall d \in D, \forall s \in S_{\text{Ext}}.$$

Moreover, it is desired for every student subgroup not to have meetings at two distinct locations in a day:

$$\textstyle\sum_{i \in H_m} y_{m,t,i} + \sum_{i \in H_{m'}} y_{m',t',i} \leq 1, \forall d \in D, \forall s \in S,$$
$$\forall \{t, t'\} \in T_d, \forall m \in M_{k_1} \cap M_s, \forall m' \in M_{k_2} \cap M_s.$$

3.3 Soft constraints

Among all the implicitly generated feasible solutions, we would like to find one that satisfies as many soft constraints as possible. For that reason we define an objective function with a penalty $w_p > 0$ for violation of constraint p. Some of the soft constraints are modeled using auxiliary variables, namely $z_{p,i}$, for a constraint of type p and for every $i \in I_p$, where I_p is the index set relevant for constraints of type p. The variable $z_{p,i}$ has value 1 if the constraint of type p is not satisfied for element i of the index set I_p, and 0 otherwise. Here we briefly describe a set of soft constraint in order to define the objective function.

S_1) **Minimize use of payable classrooms.** Some classrooms are available for lecturing, but for an additional payment, so we want to minimize the use of these classrooms.

S_2) **Compact timetable.** Teaching obligations of teaching staff should be reasonably grouped during the day: it is not desirable for one teacher to have some teaching hours in the morning and then again at the evening, with a long break in between. For simplicity we denote by L_+ the set of teachers teaching more than one session.

S_3) **Requirements related to students.** For some Master's study programs it is desirable to offer lectures only within the afternoons time slots $T_{PM} \subset T$. We denote by M_{PM} the set of meetings relevant to these programs. Another constraint related to students' preferences concerns minimization of lectures scheduled at Friday afternoon. A third constraint in this group of constraints concerns upper bound on number of teaching hours related to one student group in a day.

S_4) **Requirements related to lecturers.** Every lecturer ℓ can have some preferences among the time slots in T_ℓ and it is desired to take these preferences into account when constructing timetable.

3.4 The objective function

Putting together the above constraints, we formulate the objective function of the ILP model as follows:

$$\sum_{t \in T_r} \sum_{m \in M} \sum_{r \in R_m} w_{S_1,r,t} \cdot x_{m,t,r} + \sum_{\ell \in L_+} \sum_{d \in D} w_{S_2,\ell,d} \cdot z_{S_2,\ell,d} +$$
$$\sum_{m \in M_{PM}} \sum_{t \in T \setminus T_{PM}} \sum_{i \in H_m} w_{S_3,m} \cdot y_{m,t,i} +$$
$$\sum_{m \in M} \sum_{t \in T_5 \cap T_{PM}} \sum_{r \in R_m} w_{S_3,m,t} \cdot x_{m,t,r} +$$
$$\sum_{d \in D} \sum_{s \in S} w_{S_3,s,d} \cdot z_{S_3,s,d} + \sum_{\ell \in L} \sum_{m \in M_\ell} \sum_{t \in T} \sum_{r \in R_m} w_{S_4,\ell,t} \cdot x_{m,t,r}.$$

4. RESULTS

The ILP model was implemented using the open source programming software Zimpl [4], and evaluated using the Gurobi Optimization software [1]. The specifications of computer used for the computations are: RAM `32GB DDR3 1800Mhz` and CPU: `Intel i7-3820 3.60GHz`. In order to find an optimal solution of the proposed model, we used input data for the Spring Semester of the academic year 2016/17 at UP FAMNIT: 17 distinct study programs that

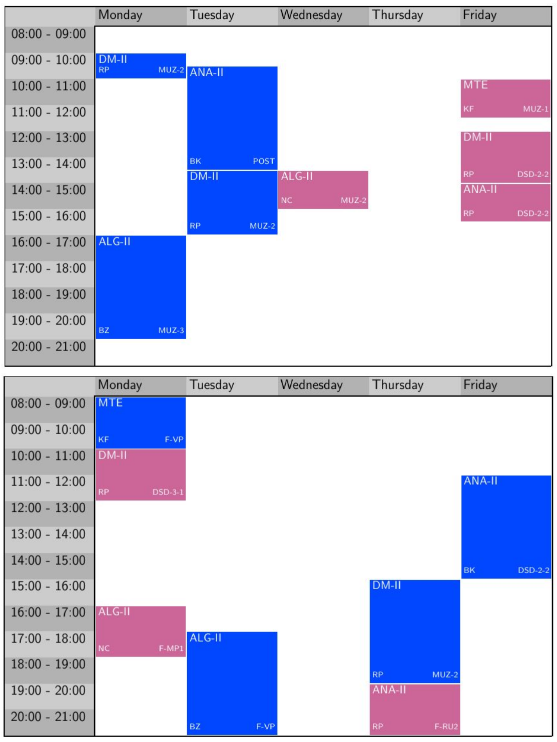


Figure 1: Timetable for one of the student groups prepared manually (top) and by solver (bottom).

in total define 48 student groups, 185 meetings, 26 classrooms, and 65 time slots, and 118 lecturers. The timetable is prepared for 5 working days and $\tau = 13$ sixty-minutes time slots within a day.

Using Zimpl we generated an `.lp` file representing the proposed model for real data in ILP standard form, containing $171,455$ variables and $2,752,376$ constraints, where $7,780,635$ entries of corresponding matrix are nonzero. The resulting `.lp` file represents the input for the solver. As expected, since all variables of the ILP are constrained to be binary, the complexity of the problem is very large. For that reason finding an optimal solution of the problem was a time-consuming process; within 48 hours no feasible solution was found. We then simplified the objective function so that just the first term of the objective function remained in the model. For this simplified objective function, we got an optimal solution in about $20,000s$.

In order to give the reader some feeling about the quality of results obtained by the implementation, in Figure 1 we display timetables produced by hand for the academic year 2016/17 and by solver, respectively, for one of the student groups.

5. CONCLUSION AND FURTHER WORK

The solution obtained here can be far from the optimum in general. Nevertheless, the automated approach produced a timetable that could be used for the desired application. It can be computed faster than the manually prepared one and it seems to have good compactness properties. In summary, this work is a good first step in the process of automating the approach to the timetabling problem at UP FAMNIT. Furthermore, as the problem is rather general, its ILP formulation or parts of it may be applicable to other institutions as well. Tasks for future research include simplifying the model to reduce the number of variables, automating the data preparation for the model, and improving the formulation of the objective function so that a reasonably good solution for the whole model can be obtained more efficiently.

6. REFERENCES

[1] Gurobi Optimizer Reference Manual, 2014. `http://www.gurobi.com`. Accessed: 2017-08-10.

[2] MathPlan. `http://www.mathplan.de`. Accessed: 2019-09-13.

[3] University Timetabling: Comprehensive Academic Scheduling Solutions. `http://www.unitime.org`. Accessed: 2019-09-13.

[4] ZIMPL: Zuse Institut Mathematical Programming Language. `http://zimpl.zib.de/`. Accessed: 2017-08-10.

[5] N. A.H. Aizam and L. Caccetta. Computational models for timetabling problems. *Numerical Algebra, Control and Optimization*, 4(1):269–285, 2014.

[6] E. Burke, K. Jackson, J. H. Kingston, and R. Weare. Automated university timetabling: The state of the art. *The Computer Journal*, 40(9):565–571, 1997.

[7] S. Daskalaki and T. Birbas. Efficient solutions for a university timetabling problem through integer programming. *European Journal of Operational Research*, 160(1):106–120, 2005.

[8] S. Daskalaki, T. Birbas, and E. Housos. An integer programming formulation for a case study in university timetabling. *European Journal of Operational Research*, 153(1):117–135, 2004.

[9] D. de Werra. An introduction to timetabling. *European Journal of Operational Research*, 19(2):151–162, 1985.

[10] M. R. Garey and D. S. Johnson. *Computers and Intractability*. WH Freeman, New York, 2002.

[11] T. Koch. *Rapid Mathematical Programming*. PhD thesis, Berlin Institute of Technology Germany, 2004.

[12] N. L. Lawrie. An integer linear programming model of a school timetabling problem. *The Computer Journal*, 12(4):307–316, 1969.

[13] A. L. Lovelace. On the complexity of scheduling university courses. Master's thesis.

[14] N. Mitrović. The UP FAMNIT Timetabling Problem – Complexity and an ILP Formulation. Master's thesis, University of Primorska, Koper, 2017.

[15] V. Pereira and H. Gomes Costa. Linear integer model for the course timetabling problem of a faculty in Rio de Janeiro. *Advances in Operations Research*, 2016.

[16] S. Petrovic and E. K. Burke. University timetabling. *Handbook of Scheduling: Algorithms, Models and Performance Analysis*, 2004.

[17] A. Schaerf. A survey of automated timetabling. *Artificial Intelligence Review*, 13(2):87–127, 1999.

[18] A. Wren. Scheduling, timetabling and rostering - a special relationship? In *International Conference on the Practice and Theory of Automated Timetabling*, pages 46–75. Springer, 1995.

Energy usage minimization with the S-graph framework

Olivér Ősz
Department of Information Technology
Széchenyi István University
9026 Egyetem tér 1
Győr, Hungary
osz.oliver@sze.hu

Máté Hegyháti
Department of Information Technology
Széchenyi István University
9026 Egyetem tér 1
Győr, Hungary
hegyhati@sze.hu

ABSTRACT

This paper proposes an extension of the S-graph framework for minimizing energy usage. The S-graph framework is a methodology for solving batch process scheduling problems. It was originally developed for makespan minimization of chemical batch processes. Since then, the framework has been extended to various scheduling problems arising in different applications.

The hereby presented extension aims to incorporate sustainability metrics, as objectives and constraints, into the S-graph framework. The proposed approach can be used to minimize total energy consumption while satisfying demand within a given time horizon, waste limitations, and energy availability.

Keywords

scheduling, combinatorial optimization, sustainability

1. INTRODUCTION

Production scheduling is an optimization problem where timing and resource allocation decisions have to be made for a set of given production tasks, while satisfying certain feasibility constraints. The objective of optimization is, in most cases, to minimize makespan (total completion time), but other frequent objectives include throughput and profit maximization, and cost, tardiness, and cycle time minimization.

In this work, the objective is to minimize energy usage, which is a special case of cost minimization, where only the energy costs are considered. Apart from the objective of minimizing energy consumption, the aim to sustainable production is also reflected in a constraint limiting the total waste produced. Both energy usage and waste production is influenced by machine assignment and operation mode selection.

Considering energy usage and waste production has been an important aspect of planning and scheduling of chemical batch processing systems[2] for a long time. But optimizing sustainability and energy efficiency in various production systems has gained an increased attention in recent years[1, 5]. The aim of this work is to extend the already versatile S-graph framework to be able to solve such problems as well.

2. PROBLEM DEFINITION

The given inputs of the problem are the recipes of the products (P) that have to be produced, the available machines (M), the length of the time horizon (H), the upper limit of total waste production (W^U), and hourly energy availability (E^U). The recipe of product $p \in P$ defines the set of tasks (T^p) to be carried out, their precedence relations ($R \subset T^p \times T^p$), and their timing and resource parameters.

The set of machines that can execute task t is denoted by M_t. The possible operation modes of executing t by $m \in M_t$ is denoted by O_{tm}. The following task parameters are dependent on the assigned machine and operation mode ($o \in O_{tm}$):

- Duration: d_o
- Hourly energy usage: e_o
- Waste produced: w_o

A solution of the problem assigns a starting time (s_t), machine (m_t), and operation mode (o_t) to each task. A solution is feasible if the following constraints are satisfied:

- Execution periods of tasks assigned to the same machine do not overlap.
- Precedence relations are satisfied, i.e., if $(t, t') \in R$, t' cannot be started before t is finished.
- Every task is finished at time H or earlier.
- Total waste production is at most W^U.
- At any moment, the total energy usage of the tasks being executed is at most E^U.

The goal is to find a solution with minimal total energy usage among the feasible ones.

3. PREVIOUS WORK

This section explains the basics and the recent advancements of the S-graph framework that the proposed approach is based on.

3.1 The S-graph framework

The original S-graph framework[4] was developed for minimizing makespan. The approach uses directed graphs to model (partial) schedules. Task nodes (N^T) and product nodes (N^P) represent the events of starting a task and completing a product, respectively. Arcs denote the order of events, where arc weights are the lower bounds of the time difference between them.

Solution starts with the so-called recipe graph, which only contains recipe arcs (A_1), which denote the technological order (R). Then a branch-and-bound algorithm is used to make assignment and scheduling decisions. Scheduling decisions introduce new ordering between tasks which are modeled by adding schedule arcs (A_2) to the graph. The detailed method of adding these arcs to the graph based on different storage policies, can be found in [4].

An S-graph is denoted as $G(N, A_1, A_2)$, where $N = N^T \cup N^P$. If a cycle is present in the directed graph $(N, A_1 \cup A_2)$, the schedule is infeasible. Otherwise, the length of the longest path in G is a lower bound on the makespan on schedules reachable from it, since scheduling decisions can only add new arcs to G and increase arc weights.

Figure 1 shows an example recipe graph with 3 products, each consisting of 3 consecutive tasks. One fully scheduled S-graph of this example is shown in Figure 2. (Weights of 0 are omitted from schedule arcs.) In a complete schedule, the processing order of tasks assigned to the same machine must be decided by the directed arcs. In Figure 2 for example, the task order of machine E1 is 1-9-7.

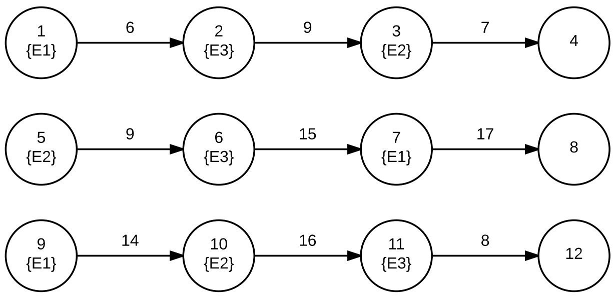


Figure 1: A recipe graph

3.2 MMRCPSP with S-graphs

The limited energy availability constraint is equivalent to the resource constraints used in the well-known Resource-Constrained Project Scheduling Problem (RCPSP), as energy can be regarded as a renewable resource. The S-graph framework has been previously extended[6] to the Multi-Mode RCPSP (MMRCPSP), where tasks can have multiple possible operation modes with varying duration and resource usage, just like in the currently investigated problem.

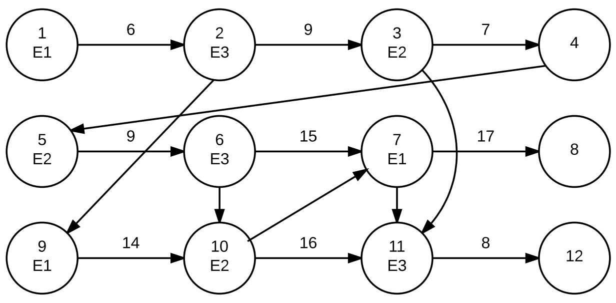


Figure 2: A fully scheduled S-graph

In a resource-feasible schedule, the total resource usage of tasks being executed at the same time cannot exceed the resource capacity. To avoid situations where the capacity may be exceeded, each minimal violating set of task-mode assignments is identified at the start. During the solution process, scheduling arcs are inserted in such a way that a violating set of tasks will not be executed at the same time.

3.3 Minimizing a cost function

Even though the original objective of the S-graph framework is makespan minimization, it can be used for cost minimization with a modified bounding method, while timing information can be used for feasibility constraints or cost calculation. This modification was shown[3] on a crew scheduling problem with routing and lateness penalties. Because of travel and penalty costs, that approach used a bounding method to approximate costs of future decisions. For the current problem, approximation is simpler, as only energy costs are considered, and cost is not affected by timing related decisions, only by unit assignments and operation mode selections.

4. PROPOSED APPROACH

The amount of consumed energy depends on the chosen machines and operation modes only, while timing decisions affect the feasibility of a solution. A two-step optimization method could be used to sort the solutions of the assignment problem by increasing energy usage, then try to find a feasible schedule for each, in order. The first feasible schedule found would be the optimal solution. This approach is good if the problem is not very tightly constrained, and a feasible schedule can be found among the first subproblems. However, there is a lot of redundancy in finding a feasible schedule for subproblems with very similar task-machine-mode assignments.

In the following, an integrated approach is presented, where assignment and scheduling decisions are made at the same level. This approach consists of 3 major modifications to the original S-graph solution method. They are detailed in the following subsections.

1. Minimal Resource Incompatible Sets (MRISs) are generated at the start of the solution process, and their status is maintained in the search according to the decisions made by branching.

2. The branch procedure makes assignment and scheduling decisions until the machine and operation mode is determined for each task, and every remaining MRIS is resolved by additional scheduling arcs to prevent parallel executions of their members.
3. Lower bound of a partial schedule is calculated by summing up the energy usages of previously assigned tasks in the chosen operation modes, and considering the minimum energy usage of unassigned tasks among their possible machine and mode assignments.

4.1 Generating incompatible sets

A Resource Incompatible Set is a set of task-machine-mode triplets, whose total resource (energy) usage exceeds available resource capacity, and the tasks are unique. It is an MRIS if no proper subset of it has the same property. Therefore, choosing any 2 tasks from an MRIS and prohibiting their parallel execution avoids resource violation by that particular MRIS (and possibly by other ones as well).

For efficient generation of the MRISs, a constraint programming (CP) model is used. Finding all feasible solutions of the following CP model provides all MRISs:

$$\sum_{m\in M_t}\sum_{o\in O_{tm}} x_{tmo} \le 1 \quad \forall t \in T \tag{1}$$

$$\mathit{waste} = \sum_{t\in T}\sum_{m\in M_t}\sum_{o\in O_{tm}} x_{tmo}\cdot w_o \tag{2}$$

$$\mathit{usage} = \sum_{t\in T}\sum_{m\in M_t}\sum_{o\in O_{tm}} x_{tmo}\cdot e_o \tag{3}$$

$$\mathit{waste} > W^U \Rightarrow \mathit{infeasible} \tag{4}$$

$$\mathit{usage} > E^U \Rightarrow \mathit{incompatible} \tag{5}$$

$$\mathit{infeasible} \vee \mathit{incompatible} \tag{6}$$

$$\mathit{incompatible} \Rightarrow \sum_{t\in T: m\in M_t}\sum_{o\in O_{tm}} x_{tmo} \le 1 \quad \forall m \in M \tag{7}$$

$$x_{tmo}\cdot(\mathit{usage} - e_o) \le E^U \quad \forall t\in T, m\in M_t, o\in O_{tm} \tag{8}$$

$$x_{tmo}\cdot(\mathit{waste} - w_o) \le W^U \quad \forall t\in T, m\in M_t, o\in O_{tm} \tag{9}$$

In the model, x_{tmo} is the binary membership variable. Constraint (1) ensures that each task appears with at most 1 machine and operation mode assignment in a set. Total waste production and hourly energy usage of the set are denoted by *waste* and *usage* respectively.

Constraints (4-6) state that an MRIS is *infeasible* due to exceeding the waste limit, or *incompatible* due to their hourly energy usage. The mode assignment of tasks in an *infeasible* MRIS is forbidden, at least one task must be executed in a different operation mode. The mode assignments of an *incompatible* MRIS are prohibited, but the tasks cannot be executed in parallel in the given operation modes. As a machine cannot execute multiple tasks in parallel, only those *incompatible* MRISs are considered, where each task is assigned to a different machine, as stated in Constraint (7). Constraints (8-9) ensure that the set is minimal, i.e., removing any member would resolve the incompatibility.

If an MRIS is a singleton, the assignment it contains is forbidden. If a task has no permitted assignments, the problem is infeasible.

Note that machines could be modeled as renewable resources (just like the hourly energy capacity) with 1 unit as capacity and usage for each task. However, that would result in many MRISs with only 2 members, as a machine cannot process more than 1 task simultaneously. Also, the same 2 tasks would be present in multiple MRISs, which only differ in the operation modes. Therefore, machine assignment is handled separately from other resources such as energy and waste.

4.2 Branch-and-bound algorithm

The main solution procedure of the original S-graph approach is only slightly modified, the major differences are in the branching method and bound calculation.

```
1: procedure SOLVE
2:     $\mathcal{I} :=$ generateMRISs()
3:     $S := \{(G(N, A_1, \emptyset), N^T, \mathcal{I}, \emptyset, \emptyset)\}$
4:     $opt := \infty$
5:     repeat
6:         $(G(N, A_1, A_2), N^U, \mathcal{I}', L, X) :=$ takeOne($S$)
7:         $b :=$ BOUND$(G(N, A_1, A_2), N^U, L, X)$
8:         if $b < opt \wedge$ maxPath$(G(N, A_1, A_2)) \le H$ then
9:             if $N^U = \mathcal{I}' = \emptyset$ then
10:                $opt := b$
11:                $optimal\_solution := (G(N, A_1, A_2), X)$
12:            else
13:                $S := S \cup$
    BRANCH$(G(N, A_1, A_2), N^U, \mathcal{I}', L, X)$
14:            end if
15:        end if
16:    until $S = \emptyset$
17:    if $opt \neq \infty$ then
18:        return $optimal\_solution$
19:    end if
20: end procedure
```

Remaining decisions consist of both unassigned tasks (N^U) and unresolved MRISs ($\mathcal{I}'$). The longest path is used to determine feasibility, not the objective bound. L contains the most recent (m, t) task assignment for each machine, or $(m, \emptyset)$ if m will not be assigned to any more tasks. X stores the (t, m, o) task-machine-mode assignments.

```
1: procedure BOUND$(G(N, A_1, A_2), N^U, L, X)$
2:     $bound := 0$
3:     for all $(t, m, o) \in X$ do
4:         $bound := bound + e_o$
5:     end for
6:     for all $t \in N^U$ do
7:         $bound := bound + \min_{\forall m\in M_t, (m,\emptyset)\notin L, o\in O_{tm}}(e_o)$
8:     end for
9:     return $bound$
10: end procedure
```

The bounding function simply sums up the energy usage of the assignments previously made, and increases it with the minimum possible usage of unassigned tasks.

Initially, the BRANCH procedure works in a similar way to the method proposed by Sanmartí et al. [4]: A machine is selected, and new branches are created where a suitable task is added to the end of the machine's production queue (L). In the investigated problem, operation modes introduce new assignment decisions. In the new approach, when the task is assigned to the machine, the operation mode is decided as well, and the resulting (t, m, o) triplet is added to X. This method creates $|O_{tm}|$ new branches for each task $t \in N^U$ that can be executed by the selected machine m, instead of only 1 per task.

Based on the assignment decisions made during branching, the set of MRISs (I) is updated by removing those that are resolved. An MRIS is resolved when either of these conditions are met:

1. It contains the currently scheduled task with a different machine or mode assignment.
2. A new path is created between 2 task nodes that are contained in the *incompatible* MRIS.

Furthermore, *infeasible* MRISs are checked to detect infeasible assignments.

After every task is assigned to a machine and operation mode, the only remaining MRISs are the *incompatible* ones, that contain the same assignments that have been made at the branching steps leading to the current node. In such a node, the Branch procedure selects one of the remaining MRISs, and introduces precedence relations between its members. A new branch is created for every ordered pair of the member tasks, where a schedule arc is added between the respective task nodes.

5. CONCLUSIONS

Previous extensions of the S-graph framework have been combined and further extended to provide a theoretical approach for solving energy minimization problems. The presented method can be generalized further to handle limits for separate waste types, additional utilities and resources, and multiple cost coefficients in the objective function.

6. ACKNOWLEDGMENTS

This work was supported by the National Research, Development and Innovation Office (NKFIH), Grant no. SNN 129178.

7. REFERENCES

[1] P. M. Castro, I. Harjunkoski, and I. E. Grossmann. Optimal scheduling of continuous plants with energy constraints. *Computers & Chemical Engineering*, 35:372–387, 2011.

[2] R. Grau, M. Graells, J. Corominas, A. Espuña, and L. Puigjaner. Global strategy for energy and waste analysis in scheduling and planning of multiproduct batch chemical processes. *Computers & Chemical Engineering*, 20(6-7):853–868, 1996.

[3] T. Holczinger, O. Ősz, and M. Hegyháti. Scheduling approach for on-site jobs of service providers. *Flexible Services and Manufacturing Journal*, May 2019.

[4] E. Sanmartí, L. Puigjaner, T. Holczinger, and F. Friedler. Combinatorial framework for effective scheduling of multipurpose batch plants. *AIChE Journal*, 48(11):2557–2570, 2002.

[5] E. Seid and T. Majozi. Optimization of energy and water use in multipurpose batch plants using an improved mathematical formulation. *Chemical Engineering Science*, 111:335–349, may 2014.

[6] O. Ősz and M. Hegyháti. An S-graph based approach for multi-mode resource-constrained project scheduling with time-varying resource capacities. *Chemical Engineering Transactions*, 70(2017):1165–1170, 2018.

Statistics-based chain code compression with decreased sensitivity to shape artefacts

David Podgorelec
University of Maribor
Faculty of Electrical Engineering and Computer Science
Maribor, Slovenia
david.podgorelec@um.si

Andrej Nerat
University of Maribor
Faculty of Electrical Engineering and Computer Science
Maribor, Slovenia
andrej.nerat@um.si

Borut Žalik
University of Maribor
Faculty of Electrical Engineering and Computer Science
Maribor, Slovenia
borut.zalik@um.si

ABSTRACT

Chain codes compactly represent raster curves. To further improve the compression, several statistics-based techniques assign shorter extra codes to frequent pairs of consecutive symbols. We systematically extend this concept to sequences of up to six symbols. A curve may thus be described by exponentially many overlapped chains, and the dynamic programming is proposed to determine the optimal one. We also propose utilization of multiple averaged hard coded pseudo-statistical models, since the exact statistical models of individual curves are huge and may also significantly differ from each other. A competitive compression efficiency is assured in this way and, as a pleasant side effect, it is less affected by the shape, rasterization algorithm, noise, and resolution, than in other contemporary methods.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity; I.4.2 [**Computing Methodologies**]: Image Processing and Computer Vision—*Compresssion (Coding)*

General Terms

Algorithms, Performance, Theory

Keywords

Chain code, dynamic programming, pseudo-statistical model

1. INTRODUCTION

Chain codes compactly describe raster curves. More than half a century ago, Freeman [3] used symbols $s_i \in [0 .. 7]$ to represent each curve pixel p_i with the azimuth direction from its predecessor p_{i-1}, measured anticlockwise from the positive x-axis. Several alternative chain codes were later introduced, but the concept remains the same as in the pioneering Freeman chain codes in eight or four directions (F8 and F4): symbols from a small alphabet are assigned to subsequent primitives along a curve. A primitive may refer to a curve pixel (as in F8), a vertex between the considered curve pixel and adjacent pixels (Vertex Chain Code – VCC [2], or Three-Orthogonal chain code – 3OT [10]), an edge separating the curve pixel from a background pixel (Differential Chain Code – DCC [9], or a rectangular cell of pixels (in quasi-lossless representation from [9]). Meanwhile, a symbol models some local geometric relation e.g. relative position of the observed primitive with respect to the previous one.

All these basic representations are efficient, as they use only 2 or 3 bits per primitive instead of coding grid coordinates with, e.g., $2 \cdot 16$ bits per pixel. Nevertheless, numerous methods have been proposed to additionally compress raster curves. Statistical coding is often utilized when the symbols' probability distribution is significantly non-uniform. Further advances in statistics-based approaches were achieved by introducing extra symbols for frequent pairs of primitives [8, 7], or by utilization of multiple statistical models in the context-based approaches [1]. On the other hand, non-statistical methods perform various string transformations [11, 12], e.g. Burrows-Wheeler Transform (BWT) or Move-To-Front Transform (MTFT) to increase the number of zeros and prepare the data for run-length encoding (RLE) and/or binary arithmetic coding (BAC).

In this paper, we introduce a new statistics-based approach where symbols may represent sequences of up to six primitives. Our aim was to achieve a competitive compression efficiency, but an interesting pleasant side effect was brought into focus during the method development. Namely, influences of the curve shape, rasterization, noise, geometric transformations, and image resolution on the compression ratio are significantly reduced in comparison to other contemporary methods. This problem has been so far addressed indirectly within the context-based approaches, while it was completely neglected in other related works. Section 2 illustrates the overall idea of the proposed approach, while Section 3 describes the preparation and utilisation of multiple averaged hard coded pseudo-statistical models, crucial for the minimization of the aforementioned influences. Section 4 experimentally confirms the compression efficiency and the reduced dependence on artefacts of the input curve. Finally, Section 5 briefly summarizes the presented work, and discusses some challenges for the future research.

2. NEW CHAIN CODE METHOD

Some years after F8 and F4, Freeman proposed the chain-difference coding (CDC) [4]. A pixel p_i is coded with the angle difference $\angle(p_i - p_{i-1}, p_{i-1} - p_{i-2})$. Unlike F8 where all symbols have practically the same probabilities, the 0° angle difference is usually much more frequent than other 7 symbols, providing a good basis for statistical coding. However, some tens of bits have to be spent to store the best-fitted statistical model (BFSM) for an individual curve, which is, particularly with shorter curves, not negligible. Liu and Žalik [6] presented the directional difference chain coding (DDCC), where CDC BFSMs of over 1000 training curves are averaged into a suboptimal hard-coded statistical model (HCSM), which is then used for compression in non-training use cases. Some years later, the compressed DDCC (C_DDCC) [7] was introduced, where three extra symbols for usually frequent pairs (±45°, ∓45°) and for sequences of 12 to 27 zeros were added to HCSM. We take a step forward by systematically extending the DDCC coding scheme with extra codes for sequences of up to six symbols. The proposed method consists of two separate phases.

1. The training phase provides a representative repertoire of training curves, extracts the BFSM for each curve, classifies BFSMs with respect to some measurable curve artefacts, and then derives HCSMs by separately averaging BFSMs within the classes. The detailed description follows in Section 3.
2. The exploitation phase analyses the input curve in order to heuristically select the most appropriate of the stored HCSMs. The chosen HCSM is then utilized to compress the curve. The main challenge in designing this phase is the strategy for determination of the optimal sequence (chain) of codes, which is emphasized in the following paragraphs.

The existing chain code techniques construct the chain of codes by a greedy algorithm. A raster curve is parsed primitive by primitive, and each of them is immediately coded either alone or as a member of some longer pattern. If different possibilities for coding a primitive exist, the predefined priority is decisive. In C_DDCC, for example, extra codes for (±45°, ∓45°) pairs have higher priority than the corresponding single-pixel codes. However, such priority-based greedy algorithms cannot be simply adjusted to efficiently handle higher number of extra codes for longer patterns of symbols. In the proposed approach, each pixel can be coded with its own code or, theoretically, with one of 20 codes of longer sequences. These include two pairs, three triplets and so on till six sextets. A longer context of patterns before and behind the considered symbol determines which of these possibilities shall be used to code the pixel. We therefore have a combinatorial optimization problem where we look for an optimal chain from a large set of multiply overlapped chains. Unlike greedy algorithms, we found dynamic programming capable to provide an optimal choice. Its utilization also facilitates the so-called context dilution problem [1, 7]. Namely, introduction of extra codes for longer patterns of symbols usually extends the codes of several symbols and other patterns.

The proposed dynamic programming approach is adaptation of the so-called exon chaining algorithm from the field of bioinformatics, the simplest of the so-called similarity-based gene prediction approaches [5]. The vertices $v_1, ..., v_n$ of the weighted dynamic programming graph represent pixels along the curve, and edges correspond to chain codes. An edge from v_i to v_{i+1} represents a single-pixel code of v_i, and an edge from v_i to $v_j, i < j - 1$, represents a joint code of pixels $v_i, ..., v_{j-1}$. Weights $w_{i,j}$ represent bit lengths of the corresponding Huffman codes. Let $IN(i)$ represent the set of left ends of all graph edges with the right end v_i. The dynamic programming algorithm computes the total weight $s_i = \min_{v_j \in IN(i)}(s_j + w_{j,i})$ of the shortest path from v_1 to $v_i, 1 < i \leq n$. The vertex $pred_i \in IN(i)$, participating to the minimum, is also memorized. The s_n represents the total bit length of the optimal chain, and the chain itself is then reconstructed by following the vertices $pred_i$ from v_n backwards to v_1. Bold edges in Fig. 1 represent the optimal chain for the given example. Patterns $v_1v_2, v_3..v_5$, and $v_6..v_9$ are coded with $4 + 6 + 8 = 18$ bits. Although the number of multiply overlapped sequences is exponential (Theorem 1), the optimization runs in linear time (Theorem 2).

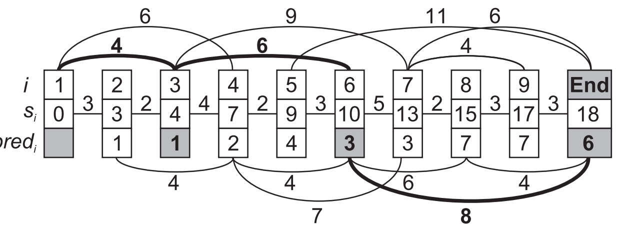


Figure 1: The dynamic programming graph with the optimal chain in bold style.

THEOREM 1. *There are exponentially many multiply overlapped chain code sequences if extra codes for patterns of up to six pixels are used.*

PROOF. Let c_i represent the number of different sequences to code $v_1, ..., v_i$. Each sequence ends with a code for k vertices, $1 \leq k \leq min(i, 6)$, preceded with one of $c_i - k$ possible sequences coding $v_1, ..., v_{i-k}$. Obviously, $c_1 = c_0 = 1, c_2 = c_0 + c_1 = 2, c_3 = c_0 + c_1 + c_2 = 4$, etc. In comparison to the well-known Fibonacci sequence, we have $c_0 = Fib_0, c_1 = Fib_1, c_2 = Fib_2$, and $c_i = Fib_i + f(i), i > 2, f(i) > 0$. As the Fibonacci sequence has the proven exponential growth, the sequence c_i also grows (at least) exponentially. □

THEOREM 2. *Optimal chain detection, based on the dynamic programming, runs in $O(n)$ time.*

PROOF. The cardinality $|IN(i)|, 1 < i \leq n$, cannot exceed 6, as each v_i may only represent the end of patterns of length from 1 to 6. Computation of $s_i, 1 < i \leq n$ thus requires $O(6n) = O(n)$ time. □

3. TRAINING PHASE

To reduce the size of the statistical model and the context dilution effect, we limit the length of patterns with attached codes to 6 pixels. Even in this way, the statistical model

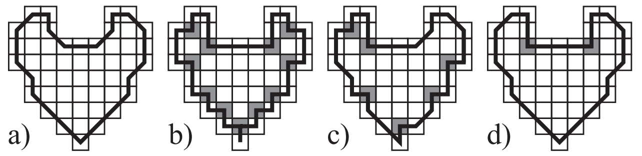


Figure 2: Different levels of forcing the 4-connectivity: a) 0%, b) 100%, c) 50%, and d) special scenario where 4-connectivity requires adjacent edges at least 2 pixels long.

derived from the basic DDCC scheme can theoretically contain $8 + 8^2 + \ldots + 8^6 = 299,592$ entries. Although many of these patterns never appear in practice, and even if we manage to further reduce the model size (to some tens entries in practice), there is the only possibility to use an averaged statistical model (or more of them). Its derivation requires a careful consideration of the following important issues.

3.1 Training set

Whatever averaged statistical model we construct, it cannot equivalently replace the BFSMs of any curve. Although the relative compression ratio to F8 only slightly varies in C_DDCC (between 0.46 and 0.55) and similarly in other existing methods, we must be aware that the training sets and testing use cases in their presentations usually followed some curve creation and rasterization methodology and, thus, they had some evident common artefacts. In C_DDCC tests, for example, there were a huge probability of shorter sequences of $0°$ symbols, relatively high probabilities of $(\pm 45°, \mp 45°)$ pairs, and rather low probabilities of $\pm 90°$ symbols. As the distributions of longer patterns from a bigger repertoire are much less predictable, we decided to use multiple averaged statistical models and, consequently, to cluster the training set and testing use cases with regard to some chosen measurable artefacts. In this manner, the method gains generality, as the compression efficiency becomes less dependent on the curve creation and rasterization methodology. To provide an adequate training set and a relevant mixture of testing use cases, we have implemented a tool with functionalities of image rotation and scaling, manual inversions of binary values of selected pixels, and extraction of the bounding contour of the presented binary object. In this last operation, the parameter "Force 4-connectivity" controls the amount of $\pm 90°$ symbols along the diagonal edges (Fig. 2) and, thus, simulates different rasterization methodologies. Basic shapes used in the training set and testing use cases are shown in Fig. 3, but we actually used a variety of instances of these shapes in different resolutions and orientations, and with different levels of forcing the 4-connectivity.

3.2 Statistical model reduction

To reduce huge amount of data in each BFSM and to mitigate the context dilution effect, a pattern $P = x_1..x_k$ is inserted in the statistical model only if its probability $p_{1..k}$ is higher than the product of probabilities (weighted with w_2) of any sequence of shorter patterns whose concatenation forms P. To prevent insertion of too low probabilities, we use additional threshold w_1. The following statement considers patterns of length $k = 3$.

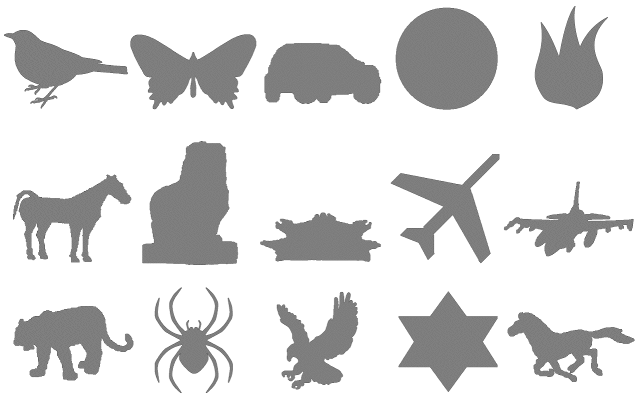


Figure 3: Examples of training and testing objects.

if $(p_{123} > max(w_1, w_2 * max(p_1p_2p_3, p_1p_{23}, p_{12}p_3)))$
then insert $(x_1x_2x_3, w_3 * 3 * p_{123})$ into BFSM.

For patterns of lengths 2, 4, 5 and 6, additional $1 + 7 + 15 + 31$ products have to be tested. Obviously, the method must first evaluate shorter patterns, as their probabilities are used in acceptance criteria for longer ones. Single-pixel symbols are unconditionally included in the BFSM to provide terminability of the dynamic programming optimization. In current tests, the weights w_1, w_2 and w_3 are set to 0.02, 1.0 and 1.0, respectively, while we intend to determine them heuristically in the future.

3.3 Statistical vs. pseudo-statistical model

We do not wish (and neither we are able) to distribute the probabilities of particular symbols and patterns among different longer patterns, as this would actually lead to the priority-based greedy approach, which we intentionally try to avoid. This means that each symbol participates to probabilities of all the patterns, which include it. We therefore do not deal with true statistical models, as we use weighted probabilities (multiplied with $w_3 * k$). Furthermore, the sum of weighted probabilities in such *pseudo-statistical model* may be theoretically as high as $(1 + 2 + 3 + 4 + 5 + 6) * w_3 = 21w_3$. Nevertheless, all weighted probabilities are involved in a single Huffman tree construction. The best fitted and averaged pseudo-statistical models will be labelled BFPSMs and APSMs in continuation.

3.4 Averaging the pseudo-statistical models

Averaging is a two-stage process. The BFPSMs of the training set curves are first classified with regard to some measured curve artefacts. In each class, the corresponding APSM is then constructed by using the same acceptance criteria as in the BFPSM reduction. During the determination of the BFPSM, we have computed several features of the considered curve that will be used in the future to experimentally select optimal classification criteria. Currently, we use three criteria listed below, each with a single threshold, partitioning the BFPSMs into two classes.

- *Average turn per pixel.* Each $\pm 45°$ symbol participates 1 to this value, $\pm 90°$ symbols participate 2, $\pm 135°$ symbols participate 3, and each $180°$ symbol partici-

Table 1: Compression rate in bits per pixel (*bpp*) of the state-of-the-art (SOTA) and the new method

Object	Transform	Pixels	bpp (SOTA)	bpp (new method)
Bird	100, 0, 0	4080	1.11	**1.03**
Butterfly	100, 0, 0	1122	1.45	**1.33**
Car	100, 0, 0	541	1.48	**1.25**
Circle	100, 0, 0	1831	1.13	**0.99**
Horse	100, 0, 0	2143	1.51	**1.39**
Bird	100, 50, 70	671	1.60	**1.31**
Butterfly	140, 45, 100	2681	1.68	**1.21**
Car	200, 33, 50	1472	1.84	**1.49**
Circle	20, 0, 0	308	1.39	**1.06**
Horse	50, 15, 20	1284	1.93	**1.51**

pates 4. The total sum is then divided with the curve length in pixels. This feature separates smooth curves from more winding and noisy ones. It is also strongly correlated with the (expectedly high) probabilities of 0° symbols and their longer runs.

- *Probability of* $(\pm 45^\circ, \mp 45^\circ)$ *pairs* is higher in curves with oblique segments than in those with mostly axis-aligned and/or ideally diagonal segments.
- *Probability of* $\pm 90^\circ$ *symbols* is closely correlated with the value "Force 4-connectivity", although the latter only controls the amount of concave right angles and does not affect the convex ones.

Three single-threshold criteria result in 8 classes. To mitigate impacts of suboptimal training set, classification criteria and thresholds selection, we use soft borders between the APSMs. Before averaging, each class is extended with the weighted probabilities from BFPSMs of all "adjacent" classes, distinct in one criterion from the considered class.

4. RESULTS

Table 1 shows typical results of our early testings of the proposed method on different instances of some objects from Fig 3. With the basic "user friendly" configurations (the top five lines), the new algorithm is for $10 - 15\%$ more efficient than the compared methods (3OT, VCC, C_DDCC, and best of MTFT+ARLE [11] variants). This difference increases to $25 - 40\%$, when more sophisticated configurations with the scaling factor, rotation angle, and/or amount of additional 4-connectivity pixels different from $100\%, 0^\circ, 100\%$, respectively (see column Transform), are considered.

5. CONCLUSIONS

In this paper, we introduce a new statistics-based cain code compression methodology by using multiple averaged pseudo-statistical models correlated with some measurable curve artefacts, and by heuristically selecting the most appropriate model prior to the compression. Furthermore, the introduced models contain systematically inserted extra codes for patterns of up to six symbols, and the dynamic programming approach replaces the common greedy method in order to determine the optimal chain of patterns and corresponding codes. The first results are promising, but there is a plenty of work left in order to ultimately affirm the proposed methodology. Our future goals include among others:

- direct comparison to modern non-statistical methods both, on "standard" and less "user-friendly" cases,
- adaptation of the introduced methodology to other basic chain code representations (VCC, 3OT, F4, F8, and NAD - normalised angle-difference chain code [11]),
- experimentation with varying the classification criteria, number and values of particular classification thresholds, weights in the pattern acceptance criteria, etc.,
- improving the training set and preparation of rich repertoire of benchmarks, and
- RLE codes for longer patterns of zeros.

6. ACKNOWLEDGMENTS

The authors acknowledge the financial support from the Slovenian Research Agency (Research Core Funding No. P2-0041, as well as Bilateral Research Project No. BI-BA/19-20-003). We are also grateful to our former colleague Denis Špelič Ph.D. who importantly assisted the implementation.

7. REFERENCES

[1] A. Akimov, A. Kolesnikov, and P. Fränti. Lossless compression of map contours by context tree modeling of chain codes. *Pattern Recogn.*, 40(3):944–952, 2007.

[2] E. Bribiesca. A new chain code. *Pattern Recognition*, 32(2):235–251, 1999.

[3] H. Freeman. On the encoding of arbitrary geometric configurations. *IRE Transactions on Electronic Computers*, EC10:260–268, 1961.

[4] H. Freeman. Computer processing of line drawing images. *ACM Computing Surveys*, 6(1):57–97, 1974.

[5] N. C. Jones and P. A. Pevzner. *An Introduction to Bioinformatics Algorithms.* The MIT Pres, 2004.

[6] Y. K. Liu and B. Žalik. An efficient chain code with huffman coding. *Pattern Recogn.*, 38(4):553–557, 2005.

[7] Y. K. Liu, B. Žalik, P.-J. Wang, and D. Podgorelec. Directional difference chain codes with quasi-lossless compression and run-length encoding. *Signal Processing: Image Commun.*, 27(9):973–984, 2012.

[8] Y. K. Liu, W. Wei, P.-J. Wang, and B. Žalik. Compressed vertex chain codes. *Pattern Recognition*, 40(11):2908–2913, 2007.

[9] P. Nunes, F. Marqués, F. Pereira, and A. Gasull. A contour based approach to binary shape coding using multiple grid chain code. *Signal Processing: Image Communication*, 15(7-8):585–599, 2000.

[10] H. Sánchez-Cruz, E. Bribiesca, and R. M. Rodríguez-Dagnino. Efficiency of chain codes to represent binary objects. *Pattern Recogniton*, 40(6):1660–1674, 2007.

[11] B. Žalik and N. Lukač. Chain code lossless compression using Move-To-Front transform and adaptive Run-Length Encoding. *Signal Processing: Image Communication*, 29(1):96–106, 2014.

[12] B. Žalik, D. Mongus, N. Lukač, and K. R. Žalik. Efficient chain code compression with interpolative coding. *Information Sciences*, 439-440:39–49, 2018.

Conflict Resolving - A Maximum Independent Set Heuristics for Solving MaxSAT

Julian Reisch*
DB Netz AG
Rotfederring 3
60327 Frankfurt
julian.reisch
@deutschebahn.com

Peter Großmann
Synoptics GmbH
Chemnitzer Str. 48b
01187 Dresden
peter.grossmann
@synoptics.de

Natalia Kliewer
Freie Universität Berlin
Information Systems
Garystrasse 21
14195 Berlin
natalia.kliewer@fu-berlin.de

ABSTRACT
Many combinatorial optimization problems of practical relevance can be formulated as Maximum Satisfiability problems (MaxSAT). There is an easy polynomial reduction from SAT and hence MaxSAT to the Maximum Independent Set Problem (MIS). We propose a fast heuristic algorithm for the MIS called Conflict Resolving (CR) and apply it to transformed MaxSAT instances. The algorithm on the transformed instances performs equally well and sometimes even better than MaxSAT solvers directly applied to the MaxSAT instances do. We prove this with experimental results where we compare our approach to state-of-the-art MaxSAT solvers submitted for the MaxSAT challenges of 2018 and 2019 in the incomplete track.

Categories and Subject Descriptors
H.4 [**Information Systems Applications**]: Miscellaneous; G.2.3 [**Mathematics of Computing**]: Applications; J.6 [**Computer Applications**]: Computer-Aided Engineering

General Terms
Algorithm, Applications

Keywords
Heuristics, Optimization, SAT, Maximum Independent Set

1. INTRODUCTION
There is a wide interest in algorithms that solve SAT and MaxSAT because many practical combinatorial optimization problems can be encoded as such. For example, the Periodic Event Scheduling Problem can be encoded in SAT and solved with the help of fast SAT solvers[8]. Moreover, there are applications in data analysis [4], model checking [5], finding bounds on Ramsey numbers [10] and many more. In this paper, we propose a novel approach for solving MaxSAT instances by reducing them to the Maximum Independent Set Problem (MIS). The reduction to MIS has been studied before [9] but we are the first ones to benefit from the fast MIS heuristic CR. There are also heuristic algorithms for the SAT problems itself [11] but even though our reduction makes the problem instances bigger, with our approach we are compatible and on some instances even better than state-of-the-art SAT solvers.

In the very broad problem class of SAT, one deals with a Boolean formula and the task is to determine whether or not there exists an interpretation of the formula. That is an assignment of true or false to all literals such that the formula evaluates to true. A formula is in conjunctive normal form (CNF) if the literals are grouped in clauses where they are connected with *or* and the clauses are connected with *and*. All SAT instances can be formulated in CNF so we can assume a formula is in CNF. The task is then to fulfill all or as many clauses as possible. The latter case is called the MaxSAT problem. A clause is fulfilled if it evaluates to true. Note that we apply a heuristics so we cannot guarantee optimality for the MIS and hence, not all clauses might be fulfilled. This is why we focus on the MaxSAT problem.

The outline of this paper will be as follows. In the following section, we explain the reduction from SAT to MIS. Then, in Section 3, we sketch our heuristic algorithm. In Section 4, we give experimental results on recent MaxSAT challenges and finally, we give a conclusion and an outlook in Section 5.

2. REDUCTION FROM SAT TO MIS
Given an undirected graph, an independent set is a set of vertices such that no two of them share an edge. It is well known that the MIS is NP-complete [7].

The SAT problem can be reduced to MIS in the following way [9]. Each pair of one clause and one literal gives rise to a vertex. Two vertices are joined by an edge if their literals come from the same clause or if one literal is the negation of the other one. By maximality, an optimal solution of the MIS assigns one literal from each clause, if possible. Hence, a maximum independent set yields an interpretation of the formula where maximally many clauses are fulfilled. For the backwards transformation, each vertex from the independent set stands for some literal which is the value in the

*Corresponding author.

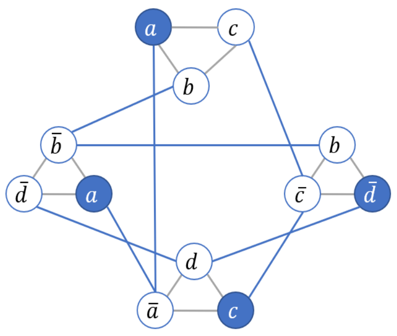


Figure 1: Example SAT to MIS

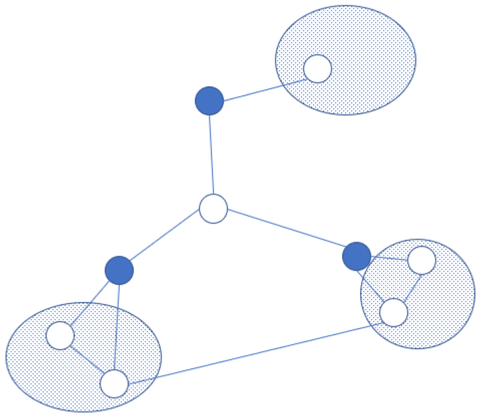


Figure 2: A root vertex to be improved (in the middle), its solution neighbors (blue), and their children, possibly replacing their parents.

SAT solution. Note that if a literal is chosen, then its negation cannot be in the independent set which ensures that the MIS solution yields a feasible SAT solution.

Example.

$$(a \vee b \vee c) \wedge (b \vee \bar{c} \vee \bar{d}) \wedge (\bar{a} \vee c \vee d) \wedge (a \vee \bar{b} \vee \bar{d})$$

In this example taken from [6], we are dealing with 4 clauses and 4 literals. In Figure 1, see the graph reduced from the example. In blue, we have marked an independent set of size 4. This implies that all 4 clauses are fulfilled. The SAT solution reads a and c is true, d is false and b can be either true or false, which does not change the objective value.

It is clear that this reduction can be done in polynomial time as there are number of literals times number of clauses many vertices in the reduced graph.

As in our case we cannot guarantee optimal solutions for the MIS, we can only find feasible solutions that fulfill many of the clauses. Hence, we cannot deal with hard clauses that necessarily have to be fulfilled. Therefore, we only consider instances with merely soft clauses which are called MaxSAT instances and are also widely studied.

3. CONFLICT RESOLVING ALGORITHM

We propose a fast heuristic algorithm for solving the MIS called Conflict Resolving (CR). The algorithm locally improves the solution by iteratively picking a non-solution vertex and trying to include this vertex in the solution. Therefore, its solution neighbors have to be replaced by their non-solution neighbors not incident to any other solution vertex. If this step can be performed, the solution size has grown by 1 without violating the property of an independent set.

In Figure 2, see how such an improvement can be performed. The root vertex in the middle is considered to be inserted into the independent set. Therefore, its blue neighbors need to leave the independent set, but must be replaced by one non-solution neighbor each, highlighted with a shallow blue. Of course, the replacement can only be performed if these vertices do not have a second neighbor in the solution set and there is no edge among them not to the root.

After iterating over all non-solution vertices, we have arrived in a local maximum. In order to be able to leave it for a better solution, we additionally deploy a perturbation step in the beginning of the algorithm where a random vertex is forced into the solution, leading to its solution neighbors to leave the solution. Since this might worsen the solution, we embed the algorithm in a simulated annealing framework and allow this worsening only with a probability that gets smaller the later the phase in the algorithm. Else, we restore the previously found best solution.

In Algorithm 1, see the overall procedure of perturbation, improvements and solution checking until some maximum of iterations is reached. This pattern is inspired by a state-of-the-art MIS heuristic proposed by Andrade et al. [3] but they use different solution checking and improvement procedures.

Algorithm 1 ConflictResolving()

```
while iteration limit has not been reached do
    Perturb solution by brute force insertion of one vertex
    Improve solution by replacing $k$ by $k+1$ vertices everywhere possible
    Check solution and allow worsening with decreasing probability
end
```

4. EXPERIMENTAL RESULTS

As testing instances, we use those instances of the MaxSAT Challenges 2018 [1] and 2019 [2] with merely soft clauses. All these instances are derived from various real-world applications such as timetabling or scheduling and general problems such as mincut, treewidth computation and many more.

We have applied the reduction from MaxSAT to MIS and subsequently applied the CR algorithm. The numbers in Tables 1 and 2 show the number of soft clauses violated per instance. Hence, the lowest value is the best result. All runs have a time limit of 300 seconds for the 2018 instances and 60 seconds in the 2019 case. We have performed all our computations on an Intel(R) Core(TM) i7-8700K. As the

Table 1: Results from 2018 with 300 seconds computation time

Benchmark (number of soft clauses)	RC2-B	RC2-A	maxino	MaxHS	Open-WBO-Gluc	Open-WBO-Riss	LMHS	QMax-SAT	CR
maxclique-brock800-2 (800)	782	780	780	782	780	781	781	781	**779**
maxclique-p-hat1000-1 (1000)	991	**990**	**990**	991	**990**	**990**	**990**	**990**	**990**
maxclique-p-hat1000-2 (1000)	958	**954**	955	959	959	960	960	960	**954**
set-covering-scpclr11 (1353)	**23**	**23**	26	**23**	31	30	34	32	33
set-covering-scpclr12 (2542)	**23**	26	26	**23**	33	33	35	35	35
set-covering-scpclr13 (4810)	29	**28**	29	29	35	34	34	34	35
set-covering-scpcyc06 (432)	**60**	**60**	**60**	**60**	72	71	71	74	61
set-covering-scpcyc07 (1120)	150	**144**	151	150	192	203	199	187	151
set-covering-scpcyc08 (2816)	357	**346**	389	356	448	552	573	488	360
set-covering-scpcyc09 (6912)	830	**805**	1285	828	1024	1343	1321	1150	843
set-covering-scpcyc10 (16640)	1916	**1890**	5540	1915	2304	11401	3685	2626	1926
set-covering-scpcyc11 (39424)	4320	**4282**	28160	4320	5120	27951	7495	8779	4371

CR is a randomized algorithm, we started 3 runs each and display the mean here.

See the results in Tables 1 and 2 for instances from 2018 and 2019, respectively. We see that our approach yields compatible results and often even the best solution. While in the 2018 instances, our results are the best for all Maximum Clique (MC) instances and not much behind the winning algorithm of the other ones, in the 2019 challenge we even achieve the best result in almost half of the cases. The only instances where our approach performs poorly are the instances for bounding a Ramsey number, such as ram-k4-n20. We firstly conclude that our approach performs best on MC instances which is a similar problem to MIS. Secondly, CR outperforms other solver when the time is limited as we compete better within 60 than 300 seconds.

5. CONCLUSION AND OUTLOOK

We have seen that reducing MaxSAT instances to MIS and then solving it with CR is similarly good and often even better in terms of good results in short time compared to state-of-the-art MaxSAT solvers.

As an outlook, it would be interesting if we can include hard clauses in our MIS heuristic. We think that this will not be a trivial task as for the MIS, one can always find a feasible solution, the empty set for instance. Finding a feasible solution for SAT, however, is NP-complete. In this respect, exact algorithms might be the better choice.

Finally, there are many problem specific instances from real-world applications that are encoded in SAT. We think that often, it would be even faster to find a direct encoding of the problem as a MIS instead of the proposed reduction. We plan to enable CR to identify instances that can be formulated in MIS directly. For example, in the trivial case, when an originally MIS instance is encoded in SAT, the algorithm should not reduce the SAT encoding as described above, but rather identify the MIS as such and solve it immediately.

6. REFERENCES

[1] MaxSAT Evaluation 2018 https://maxsat-evaluations.github.io/2018/index.html.

[2] MaxSAT Evaluation 2019 https://maxsat-evaluations.github.io/2019/index.html.

[3] D. V. Andrade, M. G. C. Resende, and R. F. F. Werneck. Fast local search for the maximum independent set problem. *J. Heuristics*, 18(4):525–547, 2012.

[4] J. Berg, A. Hyttinen, and M. Jarvisalo. Applications of maxsat in data analysis. In D. L. Berre and M. Jarvisalo, editors, *Proceedings of Pragmatics of SAT 2015 and 2018*, volume 59 of *EPiC Series in Computing*, pages 50–64. EasyChair, 2019.

[5] E. Clarke, A. Biere, R. Raimi, and Y. Zhu. Bounded model checking using satisfiability solving. *Formal Methods in System Design*, 19(1):7–34, Jul 2001.

[6] J. Erickson. Lecture notes on NP-hardness. *http://jeffe.cs.illinois.edu/teaching/algorithms/book/12-nphard.pdf*, 2019.

[7] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, 1979.

[8] P. Großmann, S. Hölldobler, N. Manthey, K. Nachtigall, J. Opitz, and P. Steinke. *Solving Periodic Event Scheduling Problems with SAT.* in: International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, pp. 166-175, Springer, 2012.

[9] P. Jégou. Decomposition of domains based on the micro-structure of finite constraint-satisfaction problems. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, AAAI'93, pages 731–736. AAAI Press, 1993.

[10] B. Molnár, M. Varga, Z. Toroczkai, and M. Ercsey-Ravasz. A high-performance analog max-sat solver and its application to ramsey numbers. *CoRR*, abs/1801.06620, 2018.

[11] O. Ohrimenko, P. J. Stuckey, and M. Codish. Propagation = lazy clause generation. In C. Bessière, editor, *Principles and Practice of Constraint Programming – CP 2007*, pages 544–558, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

Table 2: Results from 2019 with 60 seconds computation time

Benchmark (number of soft clauses)	Loandra	LinSBPS-2018	SATLike	Open-WBO-g	sls-mcs-lsu	sls-mcs	Open-WBO-ms	CR
brock200-1.clq (1132)	180	180	**179**	179	181	181	180	272.5
brock400-2.clq (1188)	258	257	**252**	261	**252**	**252**	256	287
brock400-3.clq (400)	378	377	375	378	378	378	378	**371**
brock800-1.clq (1022)	212	**205**	**205**	224	**205**	**205**	217	235
brock800-3.clq (800)	781	780	**779**	781	782	782	782	**779**
hamming10-4-1024 (1024)	**984**	992	**984**	986	992	992	989	**984**
MANN-a45-1035 (1035)	693	**690**	695	694	693	693	693	691
MANN-a81-3321 (3321)	2225	**2221**	2225	2235	2225	2225	2225	**2221**
p-hat1000-1.clq (1000)	**990**	**990**	**990**	**990**	**990**	991	991	**990**
p-hat1000-3.clq (1000)	935	935	**932**	945	938	938	947	**932**
p-hat500-2.clq (500)	**464**	465	**464**	466	470	470	468	**464**
p-hat700-1.clq (700)	690	**689**	**689**	**689**	691	691	690	**689**
p-hat700-3.clq (700)	642	641	**638**	645	644	644	645	**638**
ram-k3-n10.ra0 (300)	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**
ram-k3-n11.ra0 (495)	**7**	**7**	**7**	**7**	**7**	**7**	**7**	**7**
ram-k3-n12.ra0 (715)	**10**	**10**	**10**	12	**10**	**10**	11	12
ram-k3-n13.ra0 (1001	**16**	**16**	**16**	**16**	**16**	**16**	20)	17
ram-k3-n14.ra0 (1365)	**21**	23	**21**	27	**21**	**21**	27	23
ram-k3-n15.ra0 (1820)	**30**	31	**30**	31	**30**	**30**	35	33
ram-k3-n16.ra0 (2380)	40	40	**39**	41	**39**	**39**	47	43
ram-k3-n17.ra0 (3060)	58	**50**	**50**	51	**50**	**50**	61	65
ram-k3-n18.ra0 (3876)	63	**60**	**60**	92	**60**	**60**	98	80
ram-k3-n19.ra0 (4845)	84	76	**75**	83	**75**	**75**	126	95
ram-k3-n20.ra0 (5985)	102	95	**90**	97	**90**	**90**	124	153
ram-k4-n18.ra0 (6120)	14	11	**9**	31	**9**	**9**	29	22
ram-k4-n19.ra0 (7752)	25	18	**15**	80	**15**	**15**	42	46
ram-k4-n20.ra0 (9690))	37	33	**24**	116	**24**	**24**	67	85
sanr200-0.9.clq (200)	162	160	**158**	162	160	160	161	**158**
sanr400-0.5.clq (400)	388	388	**387**	388	389	389	388	**387**
sanr400-0.7.clq (400)	380	**379**	**379**	380	382	382	381	**379**
scpclr11-maxsat (1353)	29	26	24	36	**23**	**23**	35	32
scpclr12-maxsat (2542)	29	29	28	40	28	**23**	40	35
scpclr13-maxsat (4810)	**28**	29	30	45	30	29	45	35
scpcyc06-maxsat (432)	**60**	62	**60**	73	**60**	61	73	**60**
scpcyc07-maxsat (1120)	**149**	151	158	191	153	154	204	152
scpcyc08-maxsat (2816)	385	410	392	490	366	364	557	**357**
scpcyc09-maxsat (6912)	960	1387	838	1077	838	**831**	1316	839
scpcyc10-maxsat (16640)	2253	5540	1936	2497	1925	**1920**	7728	1930
scpcyc11-maxsat (39424)	5793	28160	4352	28160	4321	**4321**	28160	4379

Combining algorithms for vertex cover and clique search

Sándor Szabó
Institute of Mathematics and Informatics
University of Pecs
7624, Ifjúsag útja 6.
Pécs, Hungary
sszabo7@hotmail.com

Bogdán Zavalnij
Alfréd Rényi Institute of Mathematics
Hungarian Academy of Sciences
1053, Reáltanoda u. 13–15.
Budapest, Hungary
bogdan@renyi.hu

ABSTRACT
We look closely to two NP-hard problems, the minimum vertex cover and the maximum clique problem. Strictly from mathematical point of view they are absolutely the same problem. Interestingly some algorithms are better for the first one and other for the second one. Why is there such a difference? Can one make a better algorithm by combining the two approaches?

Categories and Subject Descriptors
H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms
Theory

Keywords
minimum vertex cover, maximum clique, kernelization

1. INTRODUCTION
Let $G = (V, E)$ be a finite simple graph. The graph does not contain any double edges and the graph does not contain any loops. Let $G' = (V, E')$ be the complement of G, that is iif $x, y \in V, x \neq y, \{x, y\} \notin E$ then $\{x, y\} \in E'$

The subgraph Δ of G is a clique in G if each two distinct nodes of Δ are adjacent in G. The number of nodes of Δ is called the size of the clique Δ. A clique Δ is a maximum clique in G if G does not contain any clique whose size is bigger then the size of Δ. A usual task is to find the size (and possibly show an example) of a maximum clique. It is an empirical fact that finding cliques in a given graph has many applications inside and outside of computer science [2, 8, 11, 13, 3, 15, 18].

A set of nodes C of G' is a vertex cover of G', that is each edge of the graph G' is incident to at least one node of the set. A vertex cover C is a minimum vertex cover if there is no smaller vertex cover. A usual task is to find the size (and possibly show an example) of a minimum vertex cover.

2. COMPARING VERTEX COVER AND CLIQUE SEARCH
As it is well known, if C is a vertex cover of G' then $V \setminus C$ is an independent set in G', and so a clique in G. Also, vice versa, if Δ is a clique in G, and so an independent set in G', then $V \setminus \Delta$ a vertex cover in G' [10]. Consequently, if C is a minimal vertex cover of G' then $V \setminus C$ is a maximum clique in G, and vice versa. This observation makes the two problems exactly the same from mathematical point of view.[1] Unsurprisingly the decision version of these problems are both listed among Karp's original 21 NP-complete problems [10].

The message of the mathematical equality says that one could freely use a maximum clique search program for finding minimum vertex covers or the other way around. Surprisingly T. Akiba and Y. Iwata in [1] found the opposite. They compared a Branch and Bound algorithm of their own, a known ILP formulation solved by CPLEX and a maximum clique search program MCS by Tomota et al. They used real large sparse network examples that often used in vertex cover comparisons and graphs from the second DIMACS challenge often used in maximum clique comparisons. They found the two approaches (their own and CPLEX being one and MCS the other) work better on "their own" instances. "We first observe that B&R and CPLEX clearly outperform MCS on real sparse networks. (...) In contrast, on DIMACS instances, as it is tailored to these instances, MCS generally works better." Note, that the differences are not small but extremely big, and almost without counterexamples. There were instances that one approach could solve in couple of seconds while the other approach could not solve in 24 hours and vice versa.

To resolve the contradiction one can consider three possibilities: 1) against the above facts there are major differences between the two problems; 2) there are no differences, but the usual tests are different in some ways; 3) there are no differences, but the algorithmic approaches are different. We believe that 2) was true from the beginning, and therefore, as a consequence 3) became true as well. Mainly,

[1]There are differences from point of view of parametrized complexity and approximability, but these differences are out of the scope of this paper.

the clique search programs usually deal with small hard instances, while the vertex cover community deals with huge but somehow easy instances. The vertex cover search is done with lots of preconditioning, nowadays called kernelization. This reduces the original instances to a smaller kernel, which is the really hard part of the problem. So this community is focused on and strong in regard of reducing the problems. The clique search community faces problems that are hard to precondition, so they are much stronger in dealing with the hard problems themselves. If our belief is true, then it explains the extreme difference in the T. Akiba and Y. Iwata paper. The vertex cover solvers far from perfect when dealing with really hard problems. Also, the clique search programs simply do not try to do any preconditioning and so fails to deal with huge graphs.

If our belief is true, then the combination of the two approaches could be good in both scenarios. As it turns out that is exactly the case, as it will be shown. We added to our maximum clique search program [17] – slightly upgraded – some kernelization steps. The resulting program turned out reasonably competitive in both scenarios.

The structure of the paper as follows. First, we detail the used kernelization methods, which should reduce the problem. Second, we shortly describe our maximum clique search program, which deals with the hard kernel. Third, we discuss the results of the combined program.

3. USED KERNELIZATION METHODS

3.1 Dominance

First, we used the method called dominance. That is two nodes v and x are not connected, but the neighborhood of x is included in the neighborhood of v, then v dominates x and x can be deleted.

3.2 The "struction" reduction

Second, we used the subset of the method "struction" [7], namely those transformations that would always reduce the size of the graph. Also, we restricted ourselves to transformations that could be implemented by transformations "in place". That is no new nodes needed to be added to the graph.

Note, that the present program tries to be a simple one, so lots of other methods has been left out: magnets, removing unconfined vertices, twins, funnels, desks, also if the graph consists of several components, etc. . .

In details, the above two approaches lead to these steps:

- Full $(n-1)$ degree node v: it is in the maximum clique (MC), so not in the minimum vertex cover (MVC).
- $n-2$ degree node v (1 non-neighbor): the non-neighboring node is dominated, so can be deleted (in MVC), node v in MC (not in MVC).
- The non-neighbors of a node v are not connected to each other – no edges in the non-neighborhood: the non-neighboring nodes are dominated, can be deleted (in MVC), node v in the MC (not in MVC).
- Node v has degree $n-3$ (2 non-neighbors), which are connected: these nodes can be folded. (The non connected case is a subcase from the previous.)
- Node v has degree $n-4$ (3 non-neighbors – x, y, z):
  1. x, y are connected, z is a singleton: z is dominated, can be deleted (in MVC), x, y can be folded.
  2. x, y and y, z are connected: the two edges both can be folded.
  3. all three nodes are connected: we can fold the x, y, z triangle according to [7].
- If in the subgraph spanned by the non-neighbors of a node all the nodes are degree 1 nodes – that is it has non intersecting edges: we can fold these edges at once.
- If in the subgraph spanned by the non-neighbors of a node there is a star, that is a central node to which all other nodes are connected: we can fold all edges at once.
- If in the subgraph spanned by the non-neighbors of a node v there are singleton nodes (not connected to any other nodes): these nodes dominated by v and thus can be deleted.

3.3 Chromatic degree

Third, we used the chromatic degree of a node. That is we color the graph by any valid coloring, the chromatic degree of a node is the number of colors appear in its neighborhood. If the chromatic degree is less than $k-1$ that means this node cannot be in a k-clique, and so can be deleted – thus it should be in the minimum vertex cover.

4. MAXIMUM CLIQUE SEARCH

As known from the literature (see [6]) one can often build a more efficient parameterized algorithm than a general one. So we followed this path and build our own program named $k$clique, which instead of solving the maximum clique optimization problem deals with the k-clique decision problem. Based on this program we could build a very simple and yet efficient maximum clique search program, which we call $k$clique-sequence. Here we summarize the main properties of this approach. For more detailed description see [17].

The main idea behind our program is the strong reduction of the size of the search tree. For both branching and bounding the choice of searching for k-clique helps us to reduce the search tree. Because we should decide if a k-clique exists we can always bound by the value of k. In details, for branching we can use the value of k as follows. It is well known, that the number of colors from any coloring gives an upper limit for the clique size. Thus if given the value of k and a coloring with c colors ($c \geq k$), then we can choose the smallest $c-(k-1)$ color classes, and use the nodes in them for branching – as a *branching rule*. As a terminology we will call these nodes a k-clique covering node set (KCCNS), as introduced in [16]. The importance of this comes from the nature of the Branch and Bound algorithms. These algorithms sort out the already examined nodes, meaning that they are not taken into account in the future search. Thus if all these nodes are eliminated, then the remaining nodes can be colored with $(k-1)$ colors, so there cannot be any k-clique present. Note, that without the value of k one cannot make this branching rule, and need to branch on all nodes.

Algorithm 1 summarizes our $k$clique algorithm based on this branching rule.

Algorithm 1 $k$clique

```
Require: $G = (V, E), P = V$
 1: function kCLIQUE($P, k$)
 2:     if $k = 1$ then return true
 3:     end if
 4:     KCCNS ← construct a $k$-clique covering node set
 5:     for all vertex $p \in$ KCCNS do
 6:         if kCLIQUE($P \cap N(p), k-1$) then return true
 7:         end if
 8:         $P \leftarrow P \setminus \{p\}$
 9:     end for
10:     return false
11: end function
```

We used coloring procedure named DSatur due to Brélaz [4] and also used in addition another technique, the Iterated Coloring presented by Culberson [5]. This technique uses reordering of the color classes and using a sequential coloring several times. The result cannot be worse than the previous coloring in terms of the number of colors, but it can be better. The experiments showed, that in fact the iterated recoloring reduces the number of colors quite well in most of the cases [12]. Thus we started from a Dsatur coloring and performed iterated coloring. Our stopping criteria was if the number of colors did not decreased after 1000 iterations, and we used this method on the top of the search tree.

During the Branch and Bound procedure, when there are less and less nodes as we go down on the search tree, we can reuse the coloring of the previous level, and use the repacking feature of the sequential greedy coloring. We sort the color classes by their size, and start a greedy sequential coloring from the biggest color class. As the k-clique covering node set is actually the $c-(k-1)$ sets of smallest color classes, the nodes from them moved ahead to the bigger color classes. This procedure directly reduces the size of the k-clique covering node set and so the branching factor. Our tests showed, that using this method the size of the search tree is comparable with that when we would use a DSatur coloring at each level while greatly reducing the running time.

From previous results [19] on parallel clique search algorithms we concluded, that the ordering of nodes is even more important than it was thought before. It seems that the sequence of the nodes by which we proceed in the branch has a big effect on the search tree size if pruning is present. This was shown for SAT problems [14]. This effect is used by our algorithm, and so it reduces the search space. We use a very basic reordering rule. We proceed with the nodes with the smallest degree in the remaining subgraph, that is, we used sequence of *increasing* node degrees. By doing this we solve first the easier problems and reduce the sizes of the later ones. Although simple and algorithmically cheap this approach had quite a good effect on the size of the search tree.

The structure of our maximum clique problem is extremely simple. First we find an lower bound for the size of the maximum clique. This is done by a simple greedy clique search algorithm. We set k equal to the obtained number plus one and run our $k$clique program with this parameter. If the result was that the graph do contain a k-clique we increased the value of k by one. Repeating this procedure as a sequence our program finally finds the smallest value of k for which there is no k-clique present. Thus $\omega(G) = k-1$. We call this program $k$clique-sequence, see Algorithm 2. Note, that the program calls Algorithm 1 several times.

```
Require: $G = (V, E)$
  function MAIN
      $k$ ← an lower bound by greedy clique search
      kCLIQUE-SEQ
      Print $k-1$ as the size of the maximum clique
  end function
```

Algorithm 2 $k$clique-sequence

```
 1: function kCLIQUE-SEQ
 2:     FOUND ← true
 3:     while FOUND do
 4:         FOUND ← kCLIQUE($V, k$)
 5:         if FOUND then
 6:             $k \leftarrow k+1$
 7:         end if
 8:     end while
 9:     return k
10: end function
```

5. RESULTS AND EVALUATION

Our new program uses a combined algorithm of preconditioning – kernelization –, and the branch and bound approach for the hard kernel. It was compiled for the 2019 PACE exact vertex cover challenge, but this article tries to answer the question if combining the two approaches would result in a solver good for both set of usual test problems.

First, we should check if it is good for minimum vertex cover. On the 2019 PACE competition, which was open for any contestants, the program resulted in third place (76 solved instances from 100), while the second place solved 77, and the first place 87. We also tried the program against some problems listed in the T. Akiba and Y. Iwata [1] paper. We looked at especially hard cases. Some of those we could not solve because of memory limitations – it is obvious for us, that some engineering refinement of our program could eliminate such problems. But one example showed the advantages of the program clearly. We could solve web-Standford graph in 40 minutes, while the best approach was 18 hours. These results clearly show that this approach is fruitful and maybe the desired one for minimum vertex cover.

Second, we also checked our program against the usual maximum clique finding problems. We found little difference from our previous version, but few due to the fact that we inverted the sequence in $k$clique-sequence. (For detailed comparison see [17].) Some tests run faster, a few slower. In fact the reduction played little to no role in solving these instances, explaining why such reductions are usually not implemented in maximum clique search programs. This result clearly says, that our modified program is still very competitive in maximum clique searching.

Summarizing the above, we could show, that the combination of strong reductions as kernelization and a sophisticated branch and bound program for hard kernels can be among best three minimum vertex cover *and* maximum clique search programs at the same time. We should also point out, that all three winners in the PACE 2019 exact vertex cover competition used the same approach and used reduction with a program that was originally designed for maximum clique search. See for details [9].

6. ACKNOWLEDGEMENTS

This research was supported by National Research, Development and Innovation Office - NKFIH Fund No. SNN-117879.

7. REFERENCES

[1] T. Akiba and Y. Iwata. Branch-and-reduce exponential/fpt algorithms in practice: A case study of vertex cover. *Theoretical Computer Science*, 609:211–225, 2016.

[2] I. M. Bomze, M. Budinich, P. M. Pardalos, and M. Pelillo. The maximum clique problem. In *Handbook of combinatorial optimization*, pages 1–74. Springer, 1999.

[3] A. Bóta and M. Krész. A high resolution clique-based overlapping community detection algorithm for small-world networkss. *Informatica*, 39(2), 2015.

[4] D. Brélaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22(4):251–256, 1979.

[5] J. Culberson. Iterated greedy graph coloring and the difficulty landscape. Technical report, University of Alberta, 1992.

[6] M. Cygan, F. V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized algorithms*. Springer, 2015.

[7] C. Ebenegger, P. Hammer, and D. De Werra. Pseudo-boolean functions and stability of graphs. In *North-Holland mathematics studies*, volume 95, pages 83–97. Elsevier, 1984.

[8] J. Hasselberg, P. M. Pardalos, and G. Vairaktarakis. Test case generators and computational results for the maximum clique problem. *Journal of Global Optimization*, 3(4):463–482, 1993.

[9] D. Hespe, S. Lamm, C. Schulz, and D. Strash. Wegotyoucovered: The winning solver from the PACE 2019 implementation challenge, vertex cover track, 2019.

[10] R. Karp. Reducibility among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[11] D. Kumlander. *Some Practical Algorithms to Solve the Maximal Clique problem PhD.* PhD thesis, Thesis, Tallin University of Technology, 2005., 2005.

[12] S. Margenov, T. Rauber, E. Atanassov, F. Almeida, V. Blanco, R. Ciegis, A. Cabrera, N. Frasheri, S. Harizanov, R. Kriauzien, G. Rünger, P. S. Segundo, A. Starikovicius, S. Szabo, and B. Zavalnij. Applications for ultrascale systems. In J. Carretero, E. Jeannot, and A. Y. Zomaya, editors, *Ultrascale Computing Systems*, Computing, pages 189–244. Institution of Engineering and Technology, 2019.

[13] C. Morgan. *A Combinatorial Search with Dancing Links*. PhD thesis, PhD. Thesis, Univ. of Warwick, 1999–2000.

[14] M. Ouyang. How good are branching rules in dpll? *Discrete Applied Mathematics*, 89(1-3):281–286, 1998.

[15] S. Szabó and B. Zaválnij. Reducing graph coloring to clique search. *Asia Pacific Journal of Mathematics*, 3(1):64–85, 2016.

[16] S. Szabó and B. Zaválnij. Decomposing clique search problems into smaller instances based on node and edge colorings. *Discrete Applied Mathematics*, 242:118–129, 2018.

[17] S. Szabó and B. Zaválnij. A different approach to maximum clique search. In *20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC2018)*, pages 310–316. IEEE, 2018.

[18] S. Szabo and B. Zavalnij. Reducing hypergraph coloring to clique search. *Discrete Applied Mathematics*, 264:196–207, 2019. Combinatorial Optimization: between Practice and Theory.

[19] B. Zavalnij. Speeding up parallel combinatorial optimization algorithms with las vegas method. In *10th International Conference on Large-Scale Scientific Computing*, pages 258–266. Lecture Notes in Computer Science, Springer, 2015.

Splitting partitions and clique search algorithms

Sándor Szabó
Institute of Mathematics and Informatics
University of Pécs
7624, Ifjúság útja 6.
Pécs, Hungary
sszabo7@hotmail.com

Bogdán Zavalnij
Alfréd Rényi Institute of Mathematics
Hungarian Academy of Sciences
1053, Reáltanoda u. 13–15.
Budapest, Hungary
bogdan@renyi.hu

ABSTRACT

Dividing a graph into two smaller ones in the course of a clique search algorithm is referred to as branching. In the most commonly used clique search procedures the sizes of the resulting subgraphs may widely differ. In an earlier work a novel branching method, the method of splitting partitions, was suggested to overcome this unbalance. The present paper revisits this branching idea. This time we will describe a practical technique to construct splitting partitions. In order to assess the performance of the procedure we carried out numerical experiments.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

Keywords

maximum clique, branch and bound, parallelization

1. INTRODUCTION

Let $G = (V, E)$ be a finite simple graph. Here V is the set of nodes and E is the set of edges of G. The set of edges E consists of unordered pairs of elements of V. The graph G does not have double edges or loops. Both sets V and E have finitely many elements.

A subgraph Δ of G is called a clique in G if two distinct nodes of Δ are always adjacent in G. If the clique Δ has k nodes we will say that Δ is a k-clique in G. A clique Δ is maximal if it cannot be extended to a larger clique in G by adding a node of G to Δ. A k-clique Δ in G is a maximum clique if G does not contain any $(k+1)$-clique. A graph G may contain maximal cliques of various sizes. But all the maximum cliques of G have the same size. This well defined number is called the clique number of G and it is denoted by $\omega(G)$.

Problem 1. *Given a finite simple graph $G = (V, E)$. Determine $\omega(G)$.*

Problem 2. *Given a finite simple graph $G = (V, E)$ and given a positive integer k. Decide if G contains a k-clique.*

Problem 1 is referred to as the maximum clique problem. It is an optimization problem and by the complexity theory of the algorithms it belongs to the NP-hard complexity class. Problem 2 is referred to as the k-clique problem. It is a decision problem and by the complexity theory of the algorithms it belongs to the NP-complete complexity class. (For further details see [3].) Both problems have many applications and considered to be important problems in theoretical and practical computer science.

Many clique search algorithms used in practice have the following structure. Using relatively inexpensive methods upper and lower bounds for the clique size of the given graph are established. If the lower and upper estimates are equal, then the clique size of the graph is computed. If there is a gap between the upper and lower estimates, then we divide the clique search instance into smaller instances. In short we carry out an optimality test and when this test is inconclusive a branching takes place.

Let $G = (V, E)$ be a finite simple graph. The ordered triplet (P, Q, R) of the subsets $P, Q, R \subseteq V$ is called a splitting partition of the graph G if the following conditions are satisfied.

(1) $P \cup Q \cup R = V$.

(2) $P \neq \emptyset$, $R \neq \emptyset$.

(3) $P \cap Q = P \cap R = Q \cap R = \emptyset$.

(4) $p \in P$, $r \in R$ implies that the unordered pair $\{p, r\}$ is not an edge of the graph G.

Let H be the subgraph of G induced by the set of nodes $P \cup Q$ and let K be the subgraph of G induced by the set of nodes $Q \cup R$. Let Δ be a clique in G. In [5] it was proved that either Δ is a clique in H or Δ is a clique in K. This

Table 1: The adjacency matrices of the graph in Example 1. In the second adjacency matrix we rearranged the rows and columns to make the splitting partition more visible.

	1	2	3	4	5	6
1	×	●		●	●	
2	●	×	●			
3		●	×	●		●
4	●		●	×	●	
5	●			●	×	
6			●			×

	1	5	2	4	3	6
1	×	●	●	●		
5	●	×		●		
2	●		×		●	
4	●	●		×	●	
3			●	●	×	●
6					●	×

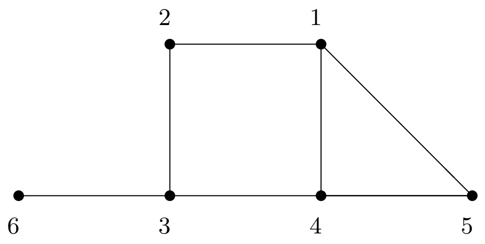


Figure 1: A graphical representation of the graph G in Example 1.

result has the following consequence. If we are looking for a maximum clique in the graph G, then we may restrict our attention to look for a maximum clique in the smaller graphs H and K. The larger are the sizes of the sets P and R the smaller are the subgraph H and K. Thus if we are able to locate a splitting partition in a computationally affordable manner, then this splitting partition maybe used as a branching rule in a maximum clique or in a k-clique search algorithm.

In this paper we will propose a method to locate splitting partitions in a given graph. There is no guarantee that the proposed procedure provides splitting sets with optimal P and R sets. We will carry out numerical experiments to demonstrate that the procedure works reasonably well.

2. THE EDGE AUXILIARY GRAPH

Given a finite simple graph $G = (V, E)$ we construct a new auxiliary graph $\Gamma = (W, F)$. The nodes of Γ are ordered pairs $w = (x, y)$ for which $x, y \in V$, $x \neq y$ and the unordered pair $\{x, y\}$ is not an edge of the graph G. Two distinct nodes $w_1 = (x_1, y_1)$ and $w_2 = (x_2, y_2)$ are adjacent in the graph Γ if any of the following two conditions is satisfied.

(1) $x_1 = x_2$ or $y_1 = y_2$.

(2) $x_1 \neq x_2$, $y_1 \neq y_2$ and the unordered pairs $\{x_1, y_2\}$, $\{x_2, y_1\}$ are not edges of the graph G.

We call the graph Γ the edge auxiliary graph associated with the graph G.

Lemma 1. *If the ordered triplet (P, Q, R) of the subsets $P, Q, R \subseteq V$ forms a splitting partition of the graph G, then the edge auxiliary graph $\Gamma = (W, F)$ contains a k-clique Δ, where $k = |P||R|$.*

Proof. Let us assume that the ordered triplet (P, Q, R) of the subsets $P, Q, R \subseteq V$ forms a splitting partition of the graph G and let us consider the following list

$$(p, r), \quad p \in P, \quad r \in R \tag{1}$$

of ordered pairs. Clearly, the ordered pairs on list (1) are pair-wise distinct and in addition $p \neq r$ holds for each ordered pair. The list (1) contains $k = |P||R|$ ordered pairs. Note that if (p_1, r_1) and (p_2, r_2) are two distinct ordered pairs from the list (1), then they are adjacent nodes of the edge auxiliary graph $\Gamma = (W, F)$. $\Box$

Next assume that we have located a k-clique Δ in the edge auxiliary graph $\Gamma = (W, F)$ and the ordered pairs

$$(x_1, y_1), \ldots, (x_k, y_k) \tag{2}$$

are all the nodes of Δ. Let $p_1, \ldots, p_\alpha$ be all the distinct elements among $x_1, \ldots, x_k$ and let $r_1, \ldots, r_\beta$ be all the distinct elements among $y_1, \ldots, y_k$. Set $P = \{p_1, \ldots, p_\alpha\}$, $R = \{r_1, \ldots, r_\beta\}$, $Q = V \setminus (P \cup R)$.

Lemma 2. *With the sets $P, Q, R \subseteq V$ defined above the triplet (P, Q, R) forms a splitting partition of the graph G.*

Proof. Clearly, the conditions (1), (2), (3) in the definition of a splitting partition are satisfied. We need to verify only that condition (4) is also satisfied. Let us consider the following list of ordered pairs

$$\begin{array}{ccc} (p_1, r_1) & \ldots & (p_1, r_\beta) \\ \vdots & \ddots & \vdots \\ (p_\alpha, r_1) & \ldots & (p_\alpha, r_\beta) \end{array} \tag{3}$$

arranged into α rows and β columns. The reader will notice that each element of the list (2) appears on list (3). If an ordered pair (p_i, r_j) on list (3) appears on list (2), then we underline the pair (p_i, r_j).

Note that p_1 is equal to the first component of one of the pairs on the list (2). It means that at least one of the pairs of the first row of list (3) is underlined. In general each row of list (3) contains at least one underlined pair. A similar reasoning gives that each column of list (3) contains at least one underlined pair. If the ordered pair (p_i, r_j) on list (3) is underlined, then it is a node of the clique Δ and consequently the associated unordered pair $\{p_i, r_j\}$ is not an edge of the graph G.

We claim that for each ordered pair (p_i, r_j) on list (3) the associated unordered pair $\{p_i, r_j\}$ is not an edge of the graph G.

In order to verify the claim assume on the contrary that there is an ordered pair (p_i, r_j) on list (3) such that the associated unordered pair $\{p_i, r_j\}$ is an edge of the graph G. There is an index t such that the ordered pair (p_i, r_t)

Table 2: The adjacency matrix of the edge auxiliary graph Γ in Example 1.

	1	1	2	2	2	3	3	4	4	5	5	5	6	6	6	6
	3	6	4	5	6	1	5	2	6	2	3	6	1	2	4	5
1,3	×	●									●	●				
1,6	●	×			●				●		●	●				
2,4			×	●	●										●	●
2,5			●	×	●		●								●	●
2,6		●	●	●	×				●			●				
3,1						×	●						●			●
3,5				●		●	×						●			●
4,2								×	●	●		●		●		
4,6		●			●			●	×	●		●				
5,2								●	●	×	●			●		
5,3	●	●								●	×	●				
5,6	●	●			●			●	●		●	×				
6,1						●	●						×	●	●	●
6,2								●		●			●	×	●	●
6,4			●	●									●	●	×	●
6,5			●	●		●	●						●	●	●	×

on list (3) is underlined. Similarly, there is an index s such that the ordered pair (p_s, r_j) on list (3) is underlined. In this situation the ordered pairs (p_i, r_t) and (p_s, r_j) are nodes of the clique Δ in the edge auxiliary graph Γ. Since the edges (p_i, r_t), (p_s, r_j) are adjacent in Γ, it follows that the unordered pairs $\{p_s, r_t\}$, $\{p_i, r_j\}$ are not edges of the graph G. This contradicts to the fact that the unordered pair $\{p_i, r_j\}$ is an edge of the graph G. $\square$

3. LEGAL COLORING OF THE NODES

In section 2 we have reduced the problem of spotting a splitting partition to the problem of spotting a clique in the edge auxiliary graph. There is a large number of algorithms for locating a not necessarily maximum clique in a given graph. In the literature they came under the name of non-exact clique search algorithms. Coloring of the nodes can be used to locate suboptimal cliques in a straight-forward manner.

The problem of determining the chromatic number of a given graph is an NP-hard problem. (For further details see [3].) It is an empirical fact that legally coloring the nodes of G using not necessarily the optimal number of colors has practical utility. In this paper we will use two approximate coloring algorithms. The first one is is known as the simple greedy node coloring and the second one is the so-called dsatur algorithm. (For further details see [2], [1], [8].)

A legal coloring of the nodes of the finite simple graph $G = (V, E)$ can be conveniently described by a function $f : V \to \{1, \ldots, k\}$. Here the numbers $1, \ldots, k$ stand for the colors and the equation $f(v) = i$ expresses the fact the node v receives color i. The set of nodes $C_i = \{v : v \in V, f(v) = i\}$ is called the i-th color class. It is the set of nodes of G colored by color i.

It is plain that a color class is an independent set of the graph G. Therefore the elements of a color class form the nodes of a clique in $\overline{G}$ the complement graph of G. So when we are looking for a clique in the edge auxiliary graph to locate a splitting partition we may do this by legally coloring the nodes of the complement of the auxiliary graph. We may pick the elements of any color class as the nodes of a clique.

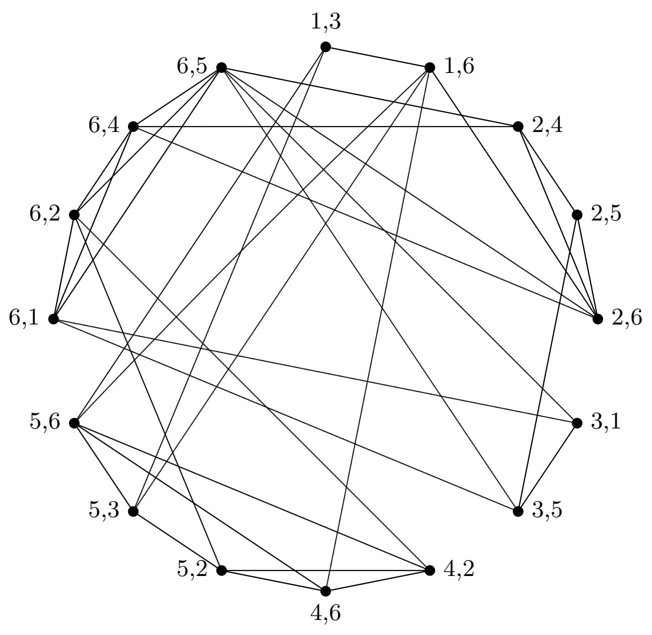


Figure 2: A geometric representation of the edge auxiliary graph Γ in Example 1

4. A SMALL SIZE TOY EXAMPLE

In order to illustrate the results presented so far we work out a small size example in details.

EXAMPLE 1. *Let us consider the graph $G = (V, E)$. Here $V = \{1, \ldots, 6\}$. The adjacency matrix of G is depicted in Table 1. Figure 1 shows a geometric representation of G.*

There reader can verify easily that the triplet (P, Q, R) of the subsets

$$P = \{1, 5\}, \quad Q = \{2, 4\}, \quad R = \{3, 6\} \tag{4}$$

is a splitting partition of the graph G. Note that upper right and the lower left two by two sub-matrices are unfilled in the second adjacency matrix in Table 1.

Using the graph G we constructed the edges auxiliary graph Γ. Table 2 displays the adjacency matrix of Γ. The rows and columns of this adjacency matrix are labeled with ordered pairs. In order to avoid an overly cluttered table we suppressed the parentheses when we recorded the ordered pairs. For instance instead of $(1, 3)$ we wrote simply $1, 3$. Figure 2 depicts a geometric representation of the edges auxiliary graph Γ.

Next we legally colored the nodes of Γ using the greedy sequential coloring procedure. The color classes are the following

$$\begin{aligned} C_1 &= \{(1,3),(1,6),(5,3),(5,6)\}, \\ C_2 &= \{(2,4),(2,5),(2,6)\}, \\ C_3 &= \{(3,1),(3,5),(6,1),(6,5)\}, \\ C_4 &= \{(4,2),(4,6),(5,2)\}, \\ C_5 &= \{(6,2),(6,4)\}. \end{aligned}$$

Applying Lemma 2 to the first color class gives the splitting partition based on the subsets (4). Using the fourth color class we get a splitting partition based on the subsets $P = \{4,5\}$, $Q = \{1,3\}$, $R = \{2,6\}$. Although the fourth color class has fewer elements than the first color class the two resulting splitting partitions have the same sizes.

As a last step we legally color the nodes of the edge auxiliary graph Γ. The nodes of Γ can be legally colored using 5 colors. The nodes of a clique must receive pair-wise distinct colors at a legal coloring of the nodes of a graph. This implies that the clique number is less than or equal to the chromatic number for each finite simple graph. Therefore, the clique number of the edge auxiliary graph Γ is at most 5. On the other hand we have located a 4-clique in Γ. The moral of this observation is that a legal coloring of the nodes of the edge auxiliary graph can be used to assess how far is the spotted suboptimal clique, we use to construct a splitting partition, from the optimal clique size.

5. NUMERICAL EXPERIMENTS

For testing purposes we have selected three infinite families of graphs that are connected to the existence and construction of certain error detecting and error correcting codes. The so-called monotonic matrices are in intimate connection with codes over the alphabet $\{1,\ldots,n\}$. Each code words has length three. The problem is to find a code whose inner distance is at least two. (See [6], [7].) The deletion error detecting codes are consisting of binary code words of length n. These words are sent over a noisy channel. Due to transmission error on the receiver side a shorter word may arrive. The task is devise a code that makes possible to detect a one bit deletion error. (For further details see [4].) The Johnson codes we are considering here are binary codes with word length n. Each code word consists of 4 1's and $n-4$ 0's. The Hamming distance of two distinct code words is at least 3.

The results of the numerical experiments are summarized in the Tables 3, 4. We describe the meaning of the entries using the last row of Table 3 as an illustration. A graph G is associated with a monotonic matrix of parameter $n = 6$. The graph has $|V| = 216$ vertices. The associated edge auxiliary graph Γ has $|W| = 11\ 340$ nodes. These numbers occupy the cells in the first three columns of the row. The splitting partition (P,Q,R) we have spotted has the parameters $|P| = \alpha = 9$, $|R| = \beta = 14$ and the last two columns contain these α, β values. This time we used the dsatur coloring procedure to get a legal coloring of the nodes of the edge auxiliary graph Γ.

At this stage we may conclude that the algorithm seems to work in connection with non-trivial size graphs in a reliable manner. Only after working with the algorithm for a longer period of time involving a much wider variety and range of graphs would enable us to assess the merits of the proposed procedure.

Table 3: Monotonic matrices.

n	$\|V\|$	$\|W\|$	α	β
3	27	324	3	4
4	64	1 440	5	6
5	125	4 500	7	10
6	216	11 340	9	14

Table 4: Johnson codes and Deletion error correcting codes.

n	$\|V\|$	$\|W\|$	α	β
6	15	120	2	4
7	35	420	2	5
8	70	1 120	2	6
9	126	2 520	1	20
10	210	5 040	2	8
11	330	9 240	4	4
12	495	15 840	4	5

n	$\|V\|$	$\|W\|$	α	β
3	8	38	3	3
4	16	126	3	4
5	32	382	3	5
6	64	1 086	4	4
7	128	2 942	4	7
8	256	7 678	5	6

6. ACKNOWLEDGMENTS

This research was supported by National Research, Development and Innovation Office - NKFIH Fund No. SNN-117879.

7. REFERENCES

[1] F. T. Leighton. A graph coloring algorithm for large scheduling problems. *Journal of Research of National Bureau of Standards*, 84:489–506, 1979.

[2] D. W. Matula, G. Marble, and J. Isaacson. Graph coloring algorithms. In R. Read, editor, *Graph Theory and Computing*, pages 109–122. Academic Press, 1972.

[3] C. H. Papadimitriou. *Computational Complexity.* Addison-Wesley Publishing Company, Inc., Reading, MA, 1994.

[4] N. Sloane. Challenge problems: Independent sets in graphs. *Information Sciences Research Center*, 2005. `http://neilsloane.com/doc/graphs.html`.

[5] S. Szabó. Parallel algorithms for finding cliques in a graph. *Journal of Physics, Conference Series*, 268, 2011.

[6] S. Szabó. Monoton matrices and finding cliques in a graph. *Annales Univ. Sci. Budapest., Sect. Computatorica*, 41:307–322, 2013.

[7] E. W. Weisstein. Monotonic matrix. `http://mathworld.wolfram.com/MonotonicMatrix.html`.

[8] Ümit V. Çatalyürek, J. Feo, A. H. Gebremedhin, M. Halappanavar, and A. Pothen. Graph coloring algorithms for multi-core and massively multithreaded architectures. *Parallel Computing*, 38(10):576 – 594, 2012.

Method for estimating tensiomyography parameters from motion capture data

Dino Vlahek[*]
dino.vlahek1@um.si

Tadej Stošić[*]
tadej.stosic1@um.si

Tamara Golob[*]
t.golob@um.si

Domen Mongus[*]
domen.mongus@um.si

Miloš Kalc[†]
milos.kalc@ism-mb.si

ABSTRACT
Tensiomyography is a muscle performance assessment technique that measures its mechanical responses. In this study, we explore a possibility to replace traditional tensiomyography measurement system with motion capture. The proposed method allows for measurement of multiple muscle's points simultaneously, while achieving measurements during a patient's movements. The results show that approximately $5mm$ error is achieved when estimating maximal muscle displacement, while time delay in muscle contraction and contraction time are assessed with upto $20ms$ error. As confirmed by physicians, the introduced errors are with the acceptable margin and, thus, the obtained results are medically valid.

Categories and Subject Descriptors
H.4 [**Information Systems Applications**]: Miscellaneous; I.3.8 [**Computer Graphics**]: Computational Geometry and Object Modeling; J.3 [**Life And Medical Sciences**]:

Keywords
Tensiomyography, marker-based motion capture, 3D points, geometric transformation

1. INTRODUCTION
Tensiomyography (TMG) is a non-invasive mechanomyography method that measures muscle mechanical response based on radial muscle belly displacement induced by electrical stimulus. Measurement unit usually includes an electrical stimulator, data acquisition subunit, a digital sensor, and muscle electrodes [2]. TMG output is a displacement-time curve evaluated with following parameters: Delay time (Td) is a time difference between the electrical impulse and 10% of the contraction, contraction time (Tc) is a time difference between 10% and 90% of the contraction, sustain time (Ts) is a time difference between 50% of the contraction and 50% of the relaxation, and relaxation time (Tr) is a time difference between 90% and 50% of the relaxation and maximal displacement of the muscle contraction (Dm). TMG is used in order to evaluate individual's maximal speed, explosiveness, endurance, and flexibility [13]. It is also applied in training optimization process in order to prevent negative effects of muscle asymmetry and asynchrony on individual's performance [16]. Additionally, after an injury, muscle functional capacity can be assessed using TMG, so that the most effective rehabilitation treatment is administered [17]. In medical research it is used in order to estimate muscle composition [18], for evaluation of muscle atrophy [9], measuring adaptation to different pathologies [10], and in order to determine muscle fiber type composition [6]. However, following TMG drawbacks can be identified: it is a fixed, static tool, which can perform single point measurements [2], [9], reliable measurement is highly dependent on an experienced measurer, since sensor and electrode placement could affect the reliability of the results [18], and measurements are generally performed in a static and relaxed position [2].
In order to address the above-mentioned issues, we propose a method that generates output similar to TMG using a marker-based motion capture. It allows a measurement of multiple points simultaneously, thus reducing the effort required in order to measure muscle contractions. The measurements can be achieved not only in relaxed positions, but also while moving, as control markers are used in order to stabilize limb natural movement in makers. The rest of the paper is organised as follows: a related work is present in Section 2. Section 3 introduces a new method that estimates TMG output from motion capture. The results of proposed method are presented in Section 4. Section 5 concludes the paper.

2. RELATED WORK
Motion capture allows for recording the movement of objects or people. Various marker-based motion capture types exist, such as acoustical systems, mechanical systems, magnetic, and optical systems. All mentioned technologies result in a time series of 3D positions of markers. Focus of this paper is on optical marker-based system. Using cameras, it records motions of special makers attached to an object. There are two types of markers: Passive markers that reflect light generated near cameras lens and active markers that emit their own light. Data captureed from image sensors is used in order to triangulate the 3D positions of a marker between two or more cameras calibrated in order to provide overlapping projections [11]. Motion capture has already been used in the sports, medicine, and entertainment industry for years. In the latter, method for capturing and modelling skin deformation in human motion [14] was proposed. It computes the motion of the skin by segmenting markers into the motion

[*]Author is with Institute of Computer Science, Faculty of Electrical Engineering and Computer Science at the University of Maribor, Maribor, Slovenia, 2000

[†]Author is with Institute of Sports Medicine, Faculty of Medicine at the University of Maribor, Maribor, Slovenia, 2000

of a set of rigid parts and residual deformations in order to animate the natural bending, bulging, and jiggling of the human form. A framework for high-fidelity facial performance acquisition, presented in [12], combines motion capture and 3D data scans in order to automatically select a minimum set of facial expressions by minimizing the reconstruction errors associated with correspondences between the motion capture markers and the face scans.
On the other hand, 3D motion capture Microsoft Kinect was proposed as a tool for gait analysis opposed to Vicon 3D motion capture, used in sport medicine, rehabilitation, and treatment for motor impairments. Captured 3D data was then processed in order to determine the accuracy of each system [15]. Integration of a real-time, interactive biofeedback stimulus from motion capture system was proposed as improvement of anterior cruciate ligament injury prevention and rehabilitation program in [3]. A methodology to preform functional simulations of the hip joint in extreme positions was presented in [4]. The authors of the above mentioned study have shown that active range of motion of the hip joint could be accurately determined with 3D reconstruction from motion capture data, which is difficult to achieve in everyday clinical practice. Hand movement reconstruction based on 3D point transformation of joints was shown as a good tool for clinical application [7]. Marker-based motion capture is frequently used for gait and skeleton analysis in sports medicine as well as for animating 3D objects in entertainment industry. Previously mentioned cases provide a solid foundation for technology used in our example.

3. METHOD

In this section, a method for estimating TMG parameters from 3D motion capture data is presented. The proposed method uses a set of markers in order to trace muscle contraction using motion capture, while TMG parameters are estimated during the following steps:
i) Point stabilization is achieved first in order to compensate for limb natural movements and preserve only those movements that result from muscle contractions.
ii) Construction of displacement-time curves is achieved next by estimating displacement distances from stabilized 3D marker positions. Extraction of TMG parameters is finally achieved based on the estimated displacement-time curve.
Following the description of the mathematical framework, these steps are described in detail.

3.1 Theoretical background

The implementation of the proposed mathematical framework is given in homogeneous coordinate system. This allows for implementing all the used geometric transformations, including translation, by matrix multiplication and, thus, enables efficient utilisation of graphic processing unit (GPU) [8].
Let a set of markers $M = \{{}^t m_i\}$, where ${}^t m_i^T = [{}^t x_i, {}^t y_i, {}^t z_i, 1]$, while i is a markers index and t is the time t of its capture. A vector between points ${}^t m_i$ and ${}^t m_j$ is denoted as ${}^t\vec{v}_{i,j} = {}^t m_i - {}^t m_j$, while its projections to $XY-$ and $XZ-$planes are denoted as ${}^t u_{i,j}^T = ({}^t x_{i,j},\ {}^t y_{i,j}, 1)$ and ${}^t w_i^T = ({}^t x_{i,j},\ {}^t z_{i,j}, 1)$, respectively. A translation for an arbitrary vector ${}^t\vec{v}_T = (x_T,\ y_T,\ z_T)$ is then given by a translation matrix M_T. In addition, rotation matrices $M_{R_y}(\Theta_y)$ and $M_{R_z}(\Theta_z)$ define rotation around $Y-$ and $Z-$axis for given angles Θ_y and Θ_z, respectively [19].

3.2 Point stabilization

In order to account for natural movement of limb, two control markers need to be placed on the limb joints in such a way that they are not affected by the movement of the measured muscles. Thus, they are used for point stabilization and are referred to as control markers defined by the indices $i = 1$ and $i = 2$. A stabilized set of markers can, therefore, be obtained by translating the corresponding control vector ${}^t\vec{v}_{1,2}$ to the origin of a given coordinate system and aligning it with the $X-$axis. Note that latter only requires rotation around $Y-$ and $Z-$axis, while the rotation round $X-$axis can be neglected due to the nature of measurement that limits such limb movements. Thus, a stabilization transformation can be defined by translation for a vector $\vec{v}_T = ({}^t x_1,\ {}^t y_1,\ {}^t z_1)$ and two rotations, defined by rotation angles Θ_y and Θ_z. Note that rotation angles ${}^t\Theta_z$ and ${}^t\Theta_y$ are defined as angles between projected vectors ${}^t\vec{u}_{1,2}$ and ${}^t\vec{w}_{1,2}$ and the $X-$axis, respectively [19]. Point stabilization transformation M_S can then be defined by

$$M_S = M_T({}^t m_1) * M_{R_y}(\Theta_y) * M_{R_z}(\Theta_z) = \begin{bmatrix} cos^t\Theta_y cos^t\Theta_z & -sin^t\Theta_z cos^t\Theta_y & sin^t\Theta_y & {}^t x_1 \\ sin^t\Theta_z & 0 & 0 & {}^t y_1 \\ -sin^t\Theta_y cos^t\Theta_z & sin^t\Theta_y sin^t\Theta_z & cos^t\Theta_y & {}^t z_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (1)$$

A stabilized set of markers ${}^T M' = \{{}^t m_i'\}$, where ${}^t m_i' = ({}^t x_i', {}^t y_i', {}^t z_i')$ is, thus, given as:

$${}^t m_i^{'} = M_S *\ {}^t m_i. \quad (2)$$

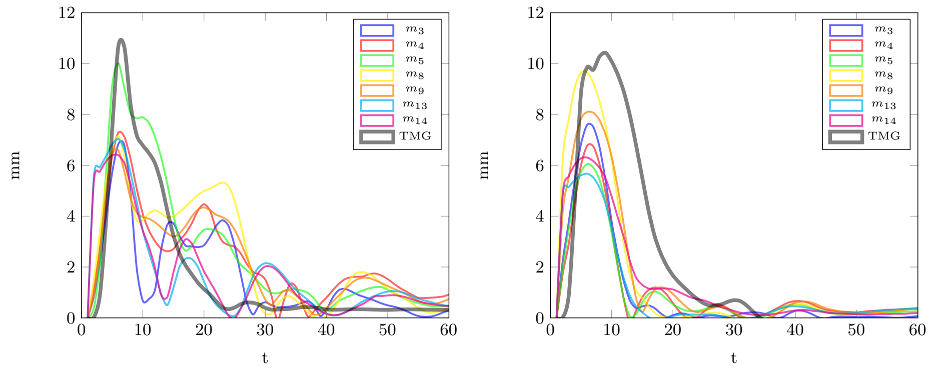
3.3 Construction of displacement-time curves and TMG parameters extraction

The objective of this step is to construct a displacement time curves from ${}^T M'$ and extract the required TMG parameters. As muscle contraction is captured by the movement of stabilised markers, a displacement curve for each marker ${}^t m_i' \in^T M'$ is generated by measuring its distance ${}^t d_i$ in time $t > 0$ from its starting point, given at $t = 0$. Formally, a displacement curve is given by a discrete mapping function $D : (t, i) \to \mathbb{R}$ defined by:

$$D(t, i) = \sqrt{({}^0 m_i' -\ {}^t m_i')^2}. \quad (3)$$

As Eq. 3 cannot produce negative values, it is critical that the initial measurement given at time $t = 0$ is measured in the relax (non-contracted) state of the muscle. $D(t, i)$, thus, provides a set of control points based on which a polynomial interpolation is achieved in order to increase the precision of the estimated TMG parameters. As polynomial interpolation is a well-know problem, it is not further discussed here. Its efficient implementation is described in [5]. Moreover, as explained in Section 1, there are five parameters that can be extracted from a displacement curve, where most of the medically relevant information is contained in maximal contraction Dm, delay time Td, and contraction time Tc. Given an interpolated displacement curve $d_i(t)$, definitions are as follows:

$$\begin{aligned} Dm(i) &= \max_t\ d_i(t), \\ Td(i) &= \arg\min_t(t; d_i(t) \geq 0.1 * Dm(i)), \\ Tc(i) &= Td(i) - \arg\min_t(t; d_i(t) \geq 0.9 * Dm(i)). \end{aligned} \quad (4)$$



(a) (b)

Figure 1: Displacement-time curves from traditional TMG and markers of muscles a) rectus femoris and b) vastus medialis.

Table 1: Results of parameters extraction from selected markers and parameters of traditional TMG.

	Rectus femoris								Vastus medialis							
	TMG	m_3	m_4	m_5	m_8	m_9	m_{13}	m_{14}	TMG	m_3	m_4	m_5	m_8	m_9	m_{13}	m_{14}
Dm (mm)	11.4	6.9	7.5	10.3	7.4	7.0	7.3	6.7	9.9	7.7	7.2	7.2	9.9	8.6	6.2	6.9
Td (ms)	26.1	16.1	7.4	21.7	9.9	17.6	2.2	1.7	25.1	3.9	2.6	7.3	1.0	1.5	2.0	3.1
Tc (ms)	42.9	39.4	44.3	27.8	37.6	27.6	41.4	38.7	30.6	45.7	47.6	38.7	32.5	40.7	38.9	36.5
Dm error		4.5(39%)	3.9(34%)	1.1(10%)	4(35%)	4.4(39%)	4.1(36%)	4.7(41%)		2.2(22%)	2.7(27%)	2.7(27%)	0(0%)	1.3(13%)	3.7(37%)	3(30%)
Td error		10(38%)	18.7(72%)	4.4(17%)	16.2(62%)	8.5(33%)	23.9(92%)	24.4(93%)		21.2(84%)	22.5(90%)	17.8(71%)	24.1(96%)	23.6(94%)	23.1(92%)	22(88%)
Tc error		3.5(8%)	-1.4(3%)	15.1(35%)	5.3(12%)	15.3(36%)	1.5(3%)	4.2(10%)		-15.1(49%)	-17(56%)	-8.1(26%)	-1.9(6%)	-10.1(33%)	-8.3(27%)	-5.9(19%)

4. RESULTS AND DISCUSSION

During the experiment, 3×5 matrix of markers was placed on the quadriceps femoris of the left leg of the tested subject, while two control markers were placed over the trochanter head and lateral condyle (see Fig. 2). Muscles contractions were captured with a Smart-D, BTS s.p.a. motion capture system [1] that consisted of 8 infra-red cameras with 800×600 spatial and 60Hz temporal resolution. Rectus femoris and vastus medialis muscles were stimulated with a single maximal electrical impulse, while control measurements were obtained using a traditional TMG sensor.

The implementation of the proposed method was done using C++ and all the execution times were measured on a workstation with Intel® CoreTM i5$^{\text{TM}}$-8400 CPU, Nvidia GeForce GTX 970 and 16 GB of main memory. As geometrical transformation were implemented on GPU, their computational complexity is $O(n)$, where n is the number of markers (in our case $n = 15$). On the other hand, the complexity of polynomial interpolation which is used in TMG parameter extraction is $O(m^2)$, where m is the number of points used for the interpolation. Thus, the theoretical computational complexity of the proposed method is equal to $O(n * m^2)$.

As shown in Fig. 1, the obtained displacement-time curves display different level of agreement with the control TMG measurement. Higher agreement was detected in cases of markers, placed near to the TMG sensor, namely m_5 in case of rectus femoris and m_8 in case of vastus medialis stimulation. In addition, m_3, m_4, m_9, m_{13}, and m_{14} showed statistically significant correlation (over 0.8) with the control measurements and, thus, TMG parameters extracted from this particular markers were further examined. The obtained results are show in Table 1. When considering Dm and Td of rectus femoris, the lowest error rates were observed in case of m_5 with $1.1mm$ and $4.4ms$ respectively, while error rates between $3.9-4.7mm$ in case of Dm and $8.5-24.4ms$ in case Td were observed in other cases. On the other hand, Tc related error rates were the lowest in case of m_4 ($-1.4ms$) and the highest in case of m_9 (15.3ms). As with m_5 induced error-rate equal to $5.3ms$. In case of vastus medialis, no Dm error was observed at m_8, while the error rates in other cases ranged between $1.3-3.7mm$. On the other hand, m_8 introduced the highest Td error of $24.1ms$. Tc error rates were in the range from $-1.9-17ms$, with the smallest related to m_8. According to the evaluation provided by the medical experts, the measured errors were, thus, within the acceptable ranges and can be considered as medically irrelevant. The error of Dm can be explained by the fact that the TMG sensor is slightly pressed into the soft tissue, resulting in small depression at baseline level causing higher value of Dm when a traditional TMG is measured. As expected, there were high errors in Td parameter, since the signals from motion capture and TMG were not properly synchronized. On the other hand, markers m_3 and m_4 registered significant movements, even though they were not placed in the anatomical regions, where contraction of Rectus femoris and Vastus medialis was expected. Such an outcome might have different explanations:

i) strong electrical stimulation can cause the propagation of the electrical stimuli in deeper tissues, causing muscle contraction of adjacent muscles,

ii) the passive mass, represented by inactivated muscles and adipose tissue near the stimulated region can vibrate, causing errors in measurements.

5. CONCLUSION

A new method for estimating TMG parameters from 3D motion capture, proposed in this paper, allows for measurement

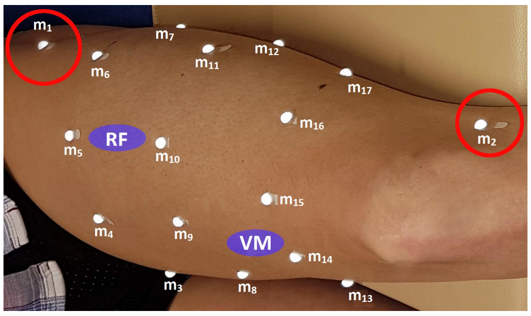


Figure 2: Placement of fifteen markers and two control markers on the subject's leg. Violet area represents placement of TMG sensor during measurement, while red circles indicate control markers.

of TMG parameters at multiple points simultaneously, while measurements can be obtained during the patient's movement. With the error rates of $5mm$ when estimating maximal muscle displacement and upto $20ms$ when estimating delay time and contraction time, the provided results proved to be medically relevant. Nevertheless, selection and a proper placement of markers is required.

One of the future tasks is synchronization of the TMG and motion capture signals that would allow for obtain the exact starting time of muscle contraction and, thus, further improved contraction and delay time assessment. In addition, improved point stabilization with compensating for rotations along X-axis will be considered. Finally, as the described study is only a proof of concept, additional test, together with statistical analysis of the extended results, are required in order to prove its value.

6. ACKNOWLEDGE

This work was supported by the Slovenian Research Agency under Grants J2-8176 and P2-0041, and the project "Wearable Integrated Smart Brace for Rehabilitation Monitoring and Diagnostic of Disorders in Muscular Functions - WIBRANT" that is co-financed by the Republic of Slovenia, Ministry of Education, Science and Sport, and the European Union under the European Regional Development Fund. More info: `www.eu-skladi.si/?set_language=en`.

7. REFERENCES

[1] Bts smart- d. Technical report, BTS Bioengineering, IT, 2019.

[2] T. Žagar and D. Križaj. Validation of an accelerometer for determination of muscle belly radial displacement. *Medical and Biological Engineering and Computing*, 43(1):78–84, Feb. 2005.

[3] S. Bonnette, C. Dicesare, A. Kiefer, M. A Riley, K. Barber Foss, S. Thomas, K. Kitchen, J. Diekfuss, and G. Myer. Injury risk factors integrated into self-guided real-time biofeedback improves high-risk biomechanics. *Journal of sport rehabilitation*, pages 1–28, 01 2019.

[4] C. Charbonnier, F. C. Kolo, V. B. Duthon, N. Magnenat-Thalmann, C. D. Becker, P. Hoffmeyer, and J. Menetrey. Assessment of congruence and impingement of the hip joint in professional ballet dancers: A motion capture study. *The American Journal of Sports Medicine*, 39(3):557–566, 2011.

[5] M. F. I. Chowdhury, C. Jeannerod, V. Neiger, . Schost, and G. Villard. Faster algorithms for multivariate interpolation with multiplicities and simultaneous polynomial approximations. *IEEE Transactions on Information Theory*, 61(5):2370–2387, May 2015.

[6] R. Dahmane, S. Djordjevic, B. Šimunič, and V. Valencic. Spatial fiber type distribution in normal human muscle Histochemical and tensiomyographical evaluation. *Journal of biomechanics*, 38(12):2451–2459, 2005.

[7] R. Degeorges, J. Parasie, D. Mitton, N. Imbert, J.-N. Goubier, and F. Lavaste. Three-dimensional rotations of human three-joint fingers: an optoelectronic measurement. preliminary results. *Surgical and Radiologic Anatomy*, 27(1):43–50, Mar 2005.

[8] P. Du, R. Weber, P. Luszczek, S. Tomov, G. Peterson, and J. Dongarra. From cuda to opencl: Towards a performance-portable solution for multi-platform gpu programming. *Parallel Computing*, 38(8):391 – 407, 2012.

[9] O. García-García, A. Cuba-Dorado, T. Álvarez Yates, J. Carballo-López, and M. Iglesias-Caamaño. Clinical utility of tensiomyography for muscle function analysis in athletes. *Open Access Journal of Sports Medicine*, 10:49–69, 2019.

[10] K. Grabljevec, H. Burger, K. Kersevan, V. Valencic, and C. Marincek. Strength and endurance of knee extensors in subjects after paralytic poliomyelitis. *Disability and Rehabilitation*, 27(14):791–799, July 2005.

[11] G. Guerra-Filho. Optical motion capture: Theory and implementation. *RITA*, 12:61–90, 2005.

[12] H. Huang, J. Chai, X. Tong, and H.-T. Wu. Leveraging motion capture and 3d scanning for high-fidelity facial performance acquisition. *ACM Trans. Graph.*, 30(4):74:1–74:10, July 2011.

[13] I. Loturco, S. Gil, C. Laurino, H. Roschel, R. Kobal, C. Abad, and F. Nakamura. Differences in muscle mechanical properties between elite power and endurance athletes: A comparative study. *The Journal of Strength and Conditioning Research*, 12 2014.

[14] S. I. Park and J. K. Hodgins. Capturing and animating skin deformation in human motion. *ACM Trans. Graph.*, 25(3):881–889, July 2006.

[15] A. Pfister, A. M. West, S. Bronner, and J. A. Noah. Comparative abilities of microsoft kinect and vicon 3d motion capture for gait analysis. *Journal of Medical Engineering & Technology*, 38(5):274–280, 2014.

[16] E. Rey, C. Lago-Peñas, and J. Lago-Ballesteros. Tensiomyography of selected lower-limb muscles in professional soccer players. *Journal of Electromyography and Kinesiology*, 22(6):866 – 872, 2012.

[17] P. S Dias, J. S Fort, D. A Marinho, A. Santos, and M. Marques. Tensiomyography in physical rehabilitation of high level athletes. *The Open Sports Sciences Journal*, 3, 03 2014.

[18] B. Simunic. Between-day reliability of a method for non-invasive estimation of muscle composition. *Journal of electromyography and kinesiology*, 22:527–30, 04 2012.

[19] J. Vince. *Mathematics for Computer Graphics*. 01 2010.

ParallelGlobal with Low Thread Interactions

Dániel Zombori
Department of Computational Optimization
University of Szeged
zomborid@inf.u-szeged.hu

Dr. Balázs Bánhelyi
Department of Computational Optimization
University of Szeged
banhelyi@inf.u-szeged.hu

ABSTRACT

Global is an optimization algorithm conceived in the '80s. Since then several papers discussed improvements of the algorithm, but adapting it to a multi-thread execution environment is only a recent branch of development [1]. Our previous work focused on parallel implementation on a single machine but sometimes the use of distributed systems is inevitable. In this paper we introduce a new version of Global which is the first step towards a fully distributed algorithm. While the proposed implementation still works on a single machine, it is easy to see how gossip based information sharing can be built into and be utilized by the algorithm. We show that ParallelGlobal is a feasible way to implement Global on a distributed system. However, further improvements must be made to solve real world problems with the algorithm.

Categories and Subject Descriptors

[**Computing methodologies**]: Optimization algorithms; [**Computing methodologies**]: Parallel algorithms

1. INTRODUCTION

Global is an optimization algorithm built from multiple modules working in an ensemble. While older implementations viewed the algorithm as a whole, the most recent GlobalJ framework handles algorithms as a collection of interlocking modules. GlobalJ has several implementations for local search algorithms and variants of Global. Main characteristics of the single threaded version were established in [4]. In recent years Global was further developed [6] and it has several applications [5, 10] where it aids mostly other research works. To speed up optimization processes we developed an algorithm [1] that is capable of utilizing multiple computational threads of a single machine. It cannot be directly implemented for distributed systems as the millisecond order of magnitude latency in communication would significantly slow down the synchronization of threads. To mitigate this problem we propose ParallelGlobal, a parallel implementation suitable for distributed systems with high latency or even with unreliable communication channels. In this paper we introduce an experimental version whose main purpose is to test the feasibility of the proposed solution. It provides an algorithm skeleton for a real distributed implementation.

2. GLOBAL

Global is a global optimizer designed to solve black box unconstrained optimization problems with low number of function evaluations and probabilistic guarantees [1, 2, 3, 4, 6, 7, 8, 11]. It uses local search algorithms to refine multiple sample points hence Global is a multi-start method. Global also utilizes the *Single Linkage Clustering* algorithm to make an estimation about the value of samples from the aspect of optimization.

2.1 Updated Global Algorithm

While the updated Global algorithm has only minor changes and in a lot of cases performs equally to the original, it is superior in execution order, therefore we consider it as the basis for improvements.

Global has an iterative framework where samples in an iteration compete with samples of previous iterations. The original version contains four phases in every iteration consisting of sampling, reduction, clustering and local search. In the updated algorithm the clustering and local search phases are merged by an implementation alternating between the two.

Algorithm 1 describes the updated Global in detail. In lines 2-5 the algorithm performs the sampling phase. Selection of sample points is stochastic, using uniform distribution in the search space. The generated samples are placed in container S which is a list structure. To find the most promising samples, S is sorted and a reduced set of samples is acquired with the lowest function values. R contains the reduced set, which is removed from S.

When samples are ready to be processed, in lines 6-24 the algorithm alternates between clustering and local searches while there are unprocessed samples left. At 7-15 samples in R are tried against the clustered samples. To determine if $r_i \in R$ is part of cluster C we need the distance threshold d_c. d_c depends on the dimension of the objective function, the number of samples currently known in the clustering process and the $\alpha \in [0, 1]$ parameter. The latter controls the decrease speed of d_c while more samples are added, in order

Algorithm 1 GLOBAL

```
 1: while $termination\text{-}criteria()$ is not $true$ do
 2:   $S \leftarrow S \cup \{n_i = uniform(lb, ub) : i \in [1, new\ samples]\}$
 3:   $S \leftarrow sort(F(s_i) < F(s_{i+1})), s_i \in S$
 4:   $R \leftarrow \{s_i : i \in [1, reduced\ set\ size]\}$
 5:   $S \leftarrow S \setminus R$
 6:   while $R$ is not $\emptyset$ do
 7:     for $C$ in $clusters$ do
 8:       $d_c \leftarrow \left(1 - \alpha^{\frac{1}{|clustered|+|R|-1}}\right)^{\frac{1}{dim(F)}}$
 9:       $N \leftarrow \{r_i : d_c > \|r_i - c_j\|_\infty \wedge F(r_i) > F(c_j)\}$
10:       if $N$ is not $\emptyset$ then
11:         $C \leftarrow C \cup N$
12:         $R \leftarrow R \setminus N$
13:         repeat iteration
14:       end if
15:     end for
16:     $l \leftarrow local\text{-}search(r_1 \in R)$
17:     $C_l, d_{min} \leftarrow \underset{C \in clusters}{argmin} \left\| l - \underset{c_i \in C,}{argmin}\ F(c_i) \right\|_\infty$
18:     if $d_{min} < d_c/10$ then
19:       $C_l \leftarrow C_l \cup \{l, r_1\}$
20:     else
21:       $clusters \leftarrow clusters \cup \{\{l, r_1\}\}$
22:     end if
23:     $R \leftarrow R \setminus \{r_1\}$
24:   end while
25: end while
```

to adapt to the expected decrease in distance between two random samples. With d_c set, sample pairs $(r_i \in R, c_j \in C)$ are evaluated to determine if r_i is part of C. The two criteria are having a clustered sample c_j with lower function value than r_i and it being closer with the infinity norm (Manhattan distance) than d_c. Samples in R satisfying both of them are moved to the current cluster C. When a sample is clustered, all samples in R can potentially be clustered too therefore $r_i \in R$ is rechecked against C. After the *for* cycle finished, samples in R cannot be the part of an existing cluster therefore performing a local search is inevitable.

Local searches are performed in lines 16-23, where l is the local optimum reached from r_1. To determine if l is a newly found local optimum a comparison with the cluster centers is needed. The center of a cluster is the sample in the cluster with the lowest function value. By finding the cluster with the closest center the algorithm can decide if the optimum is already found. If the distance d_{min} to the cluster C_l with the closest center is lower than a tenth of the d_c threshold, it is considered the same local optimum. In this case l and r_1 are added to C_l, otherwise they form a new cluster. Since r_1 is either in an already existing cluster or in a newly created one, we can remove it from R. Lines 6-24 are repeated until R becomes empty. With no unclustered samples left Global finished an iteration. The number of executed iterations is limited by the termination criteria.

3. PARALLEL GLOBAL

Our goal is to derive an implementation from the updated Global which is multi-threaded with low interactions between threads. The necessity for low thread interactions comes from the fact that on huge scale optimization tasks a single computer is not sufficient and in multi-computer environments the communication between machines is relatively slow compared to inter-thread communication. We address this problem by removing the synchronization of computational threads and replacing it with a message based information sharing scheme.

We can view ParallelGlobal as a naive parallelization of Global. The main idea lies in the parallel execution of Global iterations, while sharing information between computational threads. Consequently, inter-thread communication is necessary, however only a few selected data containers have to be shared. Also, the shared containers have independent data points and no deletions, therefore inconsistencies cannot arise from data insertions. These considerations make the algorithm for distributed systems viable.

3.1 ParallelGlobal Worker

Algorithm 2 describes the ParallelGlobal worker which is the implementation of a single computational thread. The worker might run on a machine by itself, or multiple workers can use the multi-threaded environment of a computer.

Algorithm 2 ParallelGlobal

```
 1: while $termination\text{-}criteria()$ is not $true$ do
 2:   $exchange\text{-}data()$
 3:   $s \leftarrow uniform(lb, ub)$
 4:   $R \leftarrow reduce(\{s\})$
 5:   $d_c \leftarrow \left(1 - \alpha^{\frac{1}{|clustered|+1-1}}\right)^{\frac{1}{dim(F)}}$
 6:   for $C$ in $clusters$ do
 7:     $N \leftarrow \{r_i : d_c > \|r_i - c_j\|_\infty \wedge F(r_i) > F(c_j)\}$
 8:     $C \leftarrow C \cup N$
 9:     $R \leftarrow R \setminus N$
10:   end for
11:   $l \leftarrow local\text{-}search(r_1 \in R)$
12:   $C_l, d_{min} \leftarrow \underset{C \in clusters}{argmin} \left\| l - \underset{c_i \in C}{argmin}\ F(c_i) \right\|_\infty$
13:   if $d_{min} < d_c/10$ then
14:     $C_l \leftarrow C_l \cup \{l, r_1\}$
15:   else
16:     $clusters \leftarrow clusters \cup \{\{l, r_1\}\}$
17:   end if
18: end while
```

Similarly to Global, ParallelGlobal also runs in a loop to complete iterations until a termination criterion is met. Unlike Global, the new algorithm needs a data exchange step (line 2). At the start of every iteration, received messages can be processed and new messages can be sent according to a suitable policy. The messages contain evaluated data points arranged into clusters. These clusters can be handled as if they were evaluated locally by clustering the center point (minimum) of the cluster. If the center point corresponds to an existing cluster, the two clusters should be merged while duplicate points are filtered out. Otherwise, the received cluster describes a previously unknown local optimum and it can be added to the existing clusters without modifications.

In lines 3 and 4 happens the sampling and reduction. In previous Global versions sampling and reduction was performed by taking a randomized sample set, then using a sorted sample pool and taking the best samples out. ParallelGlobal cannot utilize a common pool efficiently due to the distributed nature of the system. In this version, for simplicity we envisioned taking a single sample every iteration and using stochastic sample reduction, possibly aided with spatial measures on the samples information value. A more complex but possible solution would be a distributed sample pool. Samples could be transferred between local pools over reliable data connection. This would ensure that a sample is only evaluated by a single worker and would create a bigger variety of samples to choose from.

In lines 5-10 occurs the clustering. It is very similar to the original clustering algorithm. The only change is that we know that no more than one sample is in R. This is also true for the local search (lines 11-17) which is identical with the original local search part.

3.2 Current implementation

The current implementation of ParallelGlobal only simulates the described functionality with some simplification. First, it runs on a single machine with multiple threads as a single program. Second, messaging is simulated by synchronization on the given containers while they are written, but reading operations happen simultaneously. During clusterization, the cluster list is only read to a point determined before the process starts, hence new clusters will be excluded from already started searches. This also resembles the effects of messaging, like delays and losses in information spread. Because no real messaging is present, the *exchange-data*() function is only a placeholder for now. The *reduce*() function is also a placeholder and the subject of further development. Currently, every sample is evaluated by the clustering and local search steps.

4. RESULTS

The algorithm was examined from two aspects; comparison with the updated Global in the number of function evaluations and scaling of run time with additional threads. Numerical results were obtained on the following functions, definitions can be found in [9]. *Ackley*, *Discus*, *Easom*, *Griewank*, *Levy*, *Rastrigin*, *Schaffer*, *Schwefel*, *Shekel-5*, *Shekel-7*, *Shekel-10*, *Shubert*, *Spikes*[1] and *Zakharov*. For the evaluations we used two termination criteria, the maximum number of function evaluations is 10^5 which is a soft condition therefore overshoot is possible. To check whether an optimum point is reached we use the following expression

$$|F(x^*) - F(x)| < 10^{-8} + |F(x^*)| \cdot 10^{-6}$$

where x^* is a known global optimum point and x is the point in question. To emulate computationally more expensive functions we defined the hardness level. A hardness level of h means that the function will be evaluated 10^h times at the requested point. Global is a stochastic optimizer, moreover ParallelGlobal is also affected by the operating systems thread scheduling, consequently run times and the number of function evaluations can differ largely from one optimization process to the other. To reduce the noise induced by this, we obtained data points by averaging the results of 100 runs with every configuration. The algorithm parameterizations were identical except for the number of threads.

[1]Spikes function definition:

$$f(x) = \begin{cases} 1002 + \Pi_{x_i} sin(2\pi x_i), & \text{if } \|x - (15.25, 15.75)\|_2 > \frac{1}{4} \\ 1000, & \text{otherwise} \end{cases}$$

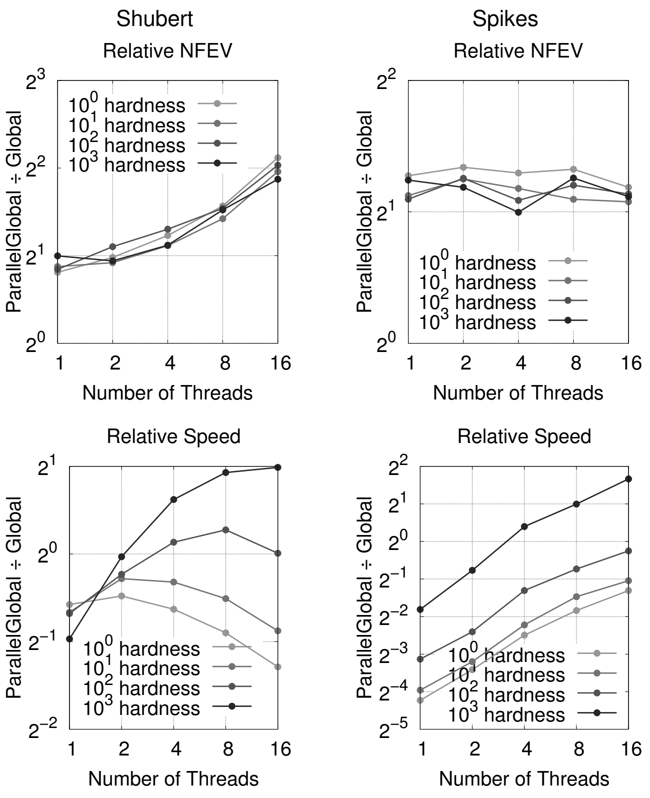


Figure 1: Numeric results on Shubert (left) and Spikes (right) test functions.

On the left side of Figure 1 we show results for the *Shubert* test function, namely the number of function evaluations (NFEV) and the speed of the optimization process, both relative to Global. On the horizontal axes we see the number of threads. The vertical axes show the number of function evaluations and optimization processes run in unit time respectively, both divided by the result of Global on a single core. *Shubert* is a function with many local optima and a flat global trend. In case of Global, NFEV is mostly in the $[500, 2000]$ range with an average of 900. On the top-left graph relative NFEV shows that we have an increase with a factor of two. On a single thread the multiplier of 2 shows that the algorithm is by itself inferior to Global. This static multiplier is explained by the lack of a sample pool which reduces the necessary number of local searches. They create the bulk of the NFEV and while Global uses 1.5 local searches on average ParallelGlobal needs much more. The dynamic growth is also explained by the local searches, combined with multi-threading. Finding the global optimum with local search takes several function evaluations in sequence. Since multiple threads start local searches independently, more evaluations can happen until one of them reaches the global optimum. Moreover when the optimum is found, the program does not terminate immediately, all local searches have to finish. This phenomenon increases the NFEV due to the intrinsic usage of multi-threading and local searches.

The bottom-left graph of Figure 1 shows the speedup with additional threads and different hardness values. While for hardness 0 and 1 the additional threads caused a slowdown due to synchronization time and increased NFEV, on computationally more demanding versions we achieved a significant speedup. The results are promising because for the hardness value of 3 on a single thread a function evaluation took only 650μs on average. With higher evaluation times, the addition of computational power would have more effect.

On the right side of Figure 1, we show the results for the *Spikes* test function which also has many local optima and a flat global trend. ParallelGlobal suffers from the lack of a sample pool on the *Spikes* function too. On the other hand, no dynamic change in NFEV is experienced. Without a sample pool, ParallelGlobal had a much harder time finding the global optimum, which would often exceed the 10^5 NFEV limit. This resulted in close to constant NFEV and no saturation of threads. Based on the relative speed graph, we gain speed linearly with additional CPU power in every hardness level. Since the function is very cheap to evaluate and ParallelGlobal has to do much more evaluations, only hardness 3 gives an advantage to the multi-threaded implementation.

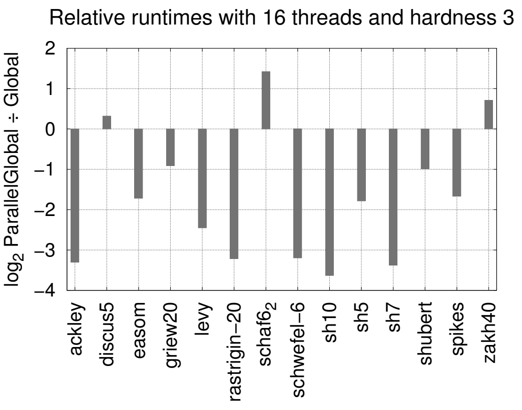


Figure 2: Relative runtimes on all test functions with 16 threads and hardness 3.

On Figure 2 we show relative runtimes for the configuration of 16 threads and hardness 3 on every test function. Since the plot is logarithmic, 0 and values below mean similar and better results compared to Global. On the functions which experienced slowdown either the lack of a sample pool or the intrinsic properties of ParallelGlobal prevented gains in speed. About 50% of the functions with speedup were solved successfully where the NFEV limit had no effects.

5. CONCLUSION

During our work we came to multiple important conclusions about the ParallelGlobal algorithm. The most needed change is the implementation of a distributed sample pool with sample sharing between threads. Having a set of probe points in the search space would ensure that local searches only start from promising regions. This change would probably move the algorithm much closer to the NFEV values of Global.

Many of our results show slowdown with ParallelGlobal, but huge improvements as hardness values increase, *Shubert* function is a good example. To keep our run times manageable we kept the hardness value relatively low. By going up from the current millisecond order to the second or 10 second order in function evaluations we would have a clearer image on how much speedup can we achieve. This would still undershoot the evaluation time of many practical problems, however it would be sufficient for proper testing on distributed systems.

To achieve these changes, first the addition of a distributed framework is needed. Both the sharing of probe samples and cluster information would rely on it. It is also a key for testing on computationally expensive problems.

6. ACKNOWLEDGMENTS

This work was supported by the Hungarian Government under the grant number EFOP-3.6.1-16-2016-00008. The project has been supported by the European Union, co-funded by the European Social Fund, and by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences.

7. REFERENCES

[1] B. Bánhelyi, T. Csendes, B. Lévai, L. Pál, and D. Zombori. *The GLOBAL Optimization Algorithm.* Springer, 2018.

[2] B. Betró and F. Schoen. Optimal and sub-optimal stopping rules for the multistart algorithm in global optimization. *Mathematical Programming*, 57:445–458, 1992.

[3] C. Boender and A. Rinnooy Kan. On when to stop sampling for the maximum. *Global Optimization*, 1:331–340, 1991.

[4] C. Boender, A. Rinnooy Kan, G. Timmer, and L. Stougie. A stochastic method for global optimization. *Mathematical Programming*, 22:125–140, 1982.

[5] T. Csendes, B. Garay, and B. Bánhelyi. A verified optimization technique to locate chaotic regions of hénon systems. *Journal of Global Optimization*, 35:145–160, 2006.

[6] T. Csendes, L. Pál, J. Sendin, and J. Banga. The global optimization method revisited. *Optimization Letters*, 2:445–454, 2008.

[7] I. Lagaris and I. Tsoulos. Stopping rules for box-constrained stochastic global optimization. *In Applied Mathematics and Computation*, 197:622–632, 2008.

[8] J. Sendín, J. Banga, and T. Csendes. Extensions of a multistart clustering algorithm for constrained global optimization problems. *Industrial & Engineering Chemistry Research*, 48:3014–3023, 2009.

[9] S. Surjanovic and D. Bingham. http://www.sfu.ca/%7essurjano/optimization.html.

[10] A. Szenes, B. Bánhelyi, L. Z. Szabó, G. Szabó, T. Csendes, and M. Csete. Improved emission of siv diamond color centers embedded into concave plasmonic core-shell nanoresonators. *Scientific Reports*, 7:an:13845, 2017.

[11] A. Törn. *A search clustering approach to global optimization*, pages 49–62. Elsevier, North-Holland, 1978.

Indeks avtorjev / Author index

IS
20
19

Konferenca / Conference

Uredili / Edited by

Srednje-ervropska konferenca o uporabnem teoretičnem računalništvu / Middle-European Conference on Applied Theoretical Computer Science

Andrej Brodnik, Gábor Galambos, Branko Kavšek