VOCAL 2018
8th VOCAL Optimization Conference: Advanced Algorithms

Esztergom, Hungary, December 10-12, 2018

Short Papers

Published by Pázmány Péter Catholic University
Szentkirályi utca 28., 1088 Budapest, Hungary
Telephone: +36 1 429-7200

Legally responsible publisher: Dr. Ferenc Friedler, professor, chair of Program Committee
Printed in format B5 by Pázmány Péter Catholic University Press
Responsible director: Margit Hesz

ISBN 978-963-308-346-8
VOCAL 2018. 8th VOCAL Optimization Conference: Advanced Algorithms. Esztergom, Hungary, December 10-12, 2018. Short Papers

Contents

3D reconstruction with depth prior using graph cut
Hichem Abdellal and Zoltan Kato 5
Integer programming formulations for college admissions with ties
Kolos Csaba Ágoston, Péter Biró, Endre Kováts and Zsuzsanna Jankó 11
IP solutions for international kidney exchange programmes
Péter Biró, Márton Gyetva, Radu-Stefan Mincu, Alexandru Popa and Utkarsh Verma 17
Markov decision processes with total effective payoff
Endre Boros, Khaled Elbassioni, Vladimir Gurvich, Kazuhisa Makino 23
New integer programming formulations for the stable exchange problem
Virginia Costa, Xenia Klimentova, Péter Biró, Ana Viana and João Pedro Pedroso 26
Some practical issues related to the implementation of type III sensitivity analysis of LP models
Imre Dimény and Tamás Koltai 32
An efficient heuristic for a complex scheduling problem
György Dósa, Tibor Dulai and Ágnes Werner-Stark 38
An exponential lifetime model: Stochastic order relations and a copula approach for modelling systemic risk
Sándor Guzmics 44
Interlacing in cyclic scheduling
Máté Hegyháti and Olivér Ősz 50
Generating sufficient matrices
Tibor Illés and Sunil Morapitiye 56
A new interior point algorithm for a class of market equilibrium problems
Tibor Illés and Anita Varga 62
A comparison of matching algorithms for kidney exchange programs
Tiago Monteiro, Xenia Klimentova, João Pedro Pedroso and Ana Viana 68
Mathematical model for power plant scheduling and its properties
Péter Naszvadi 74

Review and comparison of MILP approaches for cyclic scheduling of robotic cells
Ádám Papp, Olivér Ősz and Máté Hegyháti 79
New trends in interior-point algorithms
Petra Renáta Rigó 85
Optimizing data collection: a data-driven approach for sea exploration
Davi Pereira dos Santos and João Pedro Pedroso 91
The problem of using remnants of fabrics in upholstered furniture factories
Bogdan Staruch and Bozena Staruch 97
Task assignment to workers on the basis of their competencies
Bozena Staruch 102
A framework for defining scheduling problems
Attila Tóth and Miklós Krész 108
A heuristic approach for kidney exchange program
Utkarsh Verma and NarayanRangaraj 115

3D Reconstruction with Depth Prior Using Graph-Cut *

Hichem Abdellali and Zoltan Kato

Institute of Informatics, University of Szeged, H-6701 Szeged, PO. BOX 652., Hungary
Email: {hichem, kato}@inf.u-szeged.hu

Abstract. In this paper we propose a novel graph-cut based 3D reconstruction method which is able to take into account partially available depth data as a prior. We explored the possibility of using a prior information to achieve an efficient 3D scene reconstruction using MRF Modelling and graph-cut, which represent the disparity as an energy function. We formulate the energy in two representations: 1) assignment-based, which yields a standard binary energy; as well as 2) a multi-label one which yields a non-binary energy. Both representations have its advantages and disadvantages, which are analysed in detail through various experiments on the Middlebury stereo data set. Results show, that the use of depth prior information from different sources produces better 3D reconstructions.

Keywords: 3D Reconstruction · Graph-Cut · MRF Modelling.

1 Introduction

By using a pair of rectified binocular images, it is possible to reconstruct the 3D scene by finding dense correspondences between the images and building a disparity map. Depth information is useful for many application like modeling, monitoring, urban mapping, and autonomous navigation. Nowadays, various depth sensors are available to capture a 3D scene, like Time-of-flight devices, or Lidar. however, these are sensitive to lighting conditions and require a special setup, while stereo camera systems are more flexible, cheaper and suitable for disparity estimation. In this paper, we propose a new graph representable energy function based on the previous work of [4, 5, 3], with a new additional term which takes into account a prior disparity map collected from other sources. Introducing this depth prior provides a soft way to improve the disparity. Recently,

* This work was partially supported by the NKFI-6 fund through project K120366; "Integrated program for training new generation of scientists in the fields of computer science", EFOP-3.6.3-VEKOP-16-2017-0002; the Research & Development Operational Programme for the project "Modernization and Improvement of Technical Infrastructure for Research and Development of J. Selye University in the Fields of Nanotechnology and Intelligent Space", ITMS 26210120042, co-funded by the European Regional Development Fund.

other types of disparity estimation approaches have been proposed, such as mesh alignment regularization as well as convolutional neural networks. While these methods are quite powerful, it is not always possible to include meaningful prior information about the 3D scene. The energy function assumes rectified image pairs, hence the disparity estimation is reduced to one dimension (along with the horizontal scan lines). Consequently, disparities between the images are in the x-direction only. As disparity is inversely proportional to depth, having a disparity map provides a 3D reconstruction of the pixels up to scale. Starting from this point, we used two representations for disparity values. One is by assigning a pixel from the left frame to a corresponding pixel on the right frame, which yields a binary labeling problem [4]. The other way is to assign a multi-valued label to each pixel in the left frame directly representing the disparity value [3]. Experiments confirm that using a depth prior improves disparity estimates in both type of representation.

2 Energy Function and Graph-Cut

In this section, We will present two different representation: binary representation based on assignments [4] and multi-labeled pixel representation [3], including the proposed prior term.

Binary Label Representation: The representation is based on [4], where each pixel correspondence between the left and right images is represented as an assignment $a = (p, q)$ where a is a possible pixel pair in a limited disparity range, considering that p and q lies on the same horizontal scanline and q (right) is a possible corresponding pixel of p (left), A is the set of all assignments included in L (left frame) and R (right frame). A (binary) configuration is any map $f : A \to 0, 1$. Then an active assignment is when $f(a) = 1$ meaning that p and q correspond under the configuration f. If $f(a) = 0$, then a is inactive. According to [6] the n binary variables function is graph-representable only if each term of it satisfies the essentially sub-modularity condition, thus it can be minimized using graph-cut, see [6, 4]. In this representation, node corresponds to pixel pairs (assignments) rather than a single pixel, thus it handles occlusion and uniqueness naturally. The final energy function that we minimize including our prior term consists of five terms:

$$E(f) = \sum_{a, f(a)=1} D(a) + \sum_{a_1 \sim a_2} V_{a_1,a_2} S + \sum_{a, f(a)=1} P(a) + E_{Occ}(f) + E_{Uni}(f), \quad (1)$$

$S = T(f(a_1) \neq f(a_2))$ and T$(\cdot)$ equals 1 when its argument is true. The first term is the data cost $D(a) = D(p, q)$ which measures the dissimilarity between active assignment element p and q, we used the symmetric version of the Birchfield and Tomasi's Dissimilarity Measure [1]. $E_{Occ}(f)$ is the occlusion term, $E_{Uni}(f)$ is the uniqueness term. The smoothness term V_{a_1,a_2} is penalizing disparity jumps where there are no jumps in the intensity:

$$V_{a_1,a_a} = \begin{cases} 3\lambda \text{ If } \mathbf{max}(|I_1(p_1) - I_1(p_2)|, \\ \qquad |I_2(q_1) - I_2(q_2)|) < 8 \\ \lambda \qquad\qquad \text{Otherwise} \end{cases} \tag{2}$$

The uniqueness term E_{Uni} is enforcing only one active assignment per pixel by overflowing the energy when a pixel has more than one active assignment *i.e.* if the configuration is non-unique, null otherwise, $C_1 = T(f(a_1) = f(a_2) = 1)$:

$$E_{Uni}(f) = \sum_{\substack{a_1=(p,q_1) \\ a_2=(p,q_2) \\ q_1 \neq q_2}} \infty C_1 + \sum_{\substack{a_1=(p_1,q) \\ a_2=(p_2,q) \\ p_1 \neq p_2}} \infty C_1 \tag{3}$$

The occlusion term is penalizing inactive assignments by a penalty K. Then, the fewer the occluded pixels, the smaller the occlusion term:

$$E_{Occ}(f) = \sum_{a} KT(f(a) = 0) \tag{4}$$

The prior term $P(a) = P(p, q)$ is the difference between the given disparity and the prior disparity $P_r^{'}$ at pixel p with ω being the unit penalty for disparity difference:

$$P(p, q) = \omega \left| (p_x - q_x) - P_r^{'} \right| \tag{5}$$

Since the data term $D(p, q)$, prior term $P(p, q)$ and the occlusion term are all unary, it is possible to add them as in [6]. The sum is then trimmed for a homogenous cost: $T_r(D^{'}(p, q)) = min(\tau, D(p, q) + P(p, q))$. The final unary term, will be: $D_r^{'}(p, q) = T_r(D^{'}(p, q)) - K$

Multi-label Representation: This representation is based on [3], where disparity is represented as multi-valued labels and each label is equal to a discretized disparity value which is the difference between the quantized pixel horizontal coordinates from left and right frames. We consider here the widely known $\alpha - expansion$ move, it produces a local minimum within a known factor of the global minimum [3]. Note that this move works only in case of a metric pairwise interaction penalty. Our interest is about the vital discontinuity preserving function given by the Potts model $V(\alpha, \beta) = KT(\alpha \neq \beta)$ which is metric. This approach does not treat the images symmetrically and thus may yield inconsistent disparities. Occlusions are also ignored and adding a special label for occlusion would not use both images symmetrically [3, 5]. The energy function including the prior has three terms:

$$E(f) = \sum_{p \in P} D_p(f_p) + \sum_{\{p,q\} \in N} V_{p,q}(f_p, f_q) + \sum_{p \in P} P(f_p, {f_p}') \tag{6}$$

$D_p(f_p)$ is the data term which is a penalty for assigning a label d to a pixel p, depending on the intensity of pixels p and $q = p + d$, with the same prior

function defined previously. The final unary term, including the prior, will be: $min(D_p(f_p) + |(x_p - x_{p+f_p}) - f_p^{'}| * w, \tau)$, where x_p is the x-coordinate of pixel p in one frame, x_{p+f_p} is the x-coordinate of pixel p plus a disparity on the second frame, w is a constant weight, and $f_p^{'}$ is the prior disparity at p. The smoothness term $V_{p,q}(f_p, f_q)$ guarantees that the overall labeling f is smooth. It is based on the Potts model [3], without considering the second image. *i.e.* is more likely that two neighboring pixels have the same labeling if they have a similar intensity.

$$\sum_{\{p,q\} \in N} V_{p,q}(f_p, f_q) = \sum_{\{p,q\} \in N} u_{\{p,q\}} T(f_p \neq f_p) \tag{7}$$

where $u_{\{p,q\}}$ is a penalty for assigning different disparities to neighboring pixels p and q, μ is the Potts model parameter.

$$u_{\{p,q\}} = \begin{cases} 2\mu \; if \; |I_p - I_q| \leq 5 \\ \mu \quad \text{Otherwise} \end{cases} \tag{8}$$

3 Experiments

We used the provided code written in C++ from [4] for the binary representation, and the Matlab wrapper - GCoptimization - software for energy minimization with graph-cuts from [8, 6, 2] for the multi-label representation. For simplicity, we refer to the Binary label representation as BL and to the Multi-label as ML

Middlebury Stereo Datasets: in our experiments, we used the Middlebury Stereo benchmark. It has five sets, out of which we used the first four sets [7], in total 37 rectified stereo image pairs. A ground truth disparity map is available for each pair, which is the basis for quantitative evaluation of our disparity estimates. As the Multi-label representation handle weakly the occlusion, only by adding an extra label due to the nature of the model, the quantitative evaluation is limited to the non-occluded pixels, while occlusion accuracy is subjectively evaluated. In our experiments, the error rate is based on the number of bad matches with the available ground truth disparity. For the depth prior we extracted randomly a *partial* region from the ground truth, *i.e.* the prior is available in every image pixel, except an arbitrary masked region, which corresponds to high-resolution 3D data with some occlusion/missing data. We run the binary-labeling test with default parameters provided with the source code of [4], while the occlusion cost and the data fidelity are tuned automatically. For the Multi-label, we used the best parameter setting that we could achieve experimentally. The performance of the algorithms has been quantitatively evaluated over regions where prior depth data is available as well as over regions where prior information was not available. In this way, we can separately characterize the efficiency of the model where prior data is directly available and over regions where such information is not directly available, but - due to the pairwise interactions- the prior has an indirect effect. A first impression from the results is that our method performs

much better than the classical one, the introduction of the prior term improves the quality of the final result. Fig. 1 shows that the details come out and the occlusion appears almost correctly. Fig. 2 shows an overall view of the results, in particular, it shows how the bad matches were reduced over the Middlebury dataset. Apparently, the prior is making the final disparity map accurate over the whole dataset with an average error being below 8%, the experiments show that it is also possible to obtain a perfect reconstruction. For some cases with the higher error rate, the bad matches are due to the expansion of the size and the disparity range. We can notice that with a right occlusion penalty for the Multi-labeling representation, it is giving better results even if it is not perfect. Note also, that using the prior is also improving the image regions where the prior is not available, because of the pairwise interactions will propagate the right disparity value over such regions too. A large disparity range may yield more bad matches – this is also due to the fact that the optimization is global, *i.e.* when a large number of matches are wrong, they can perturb the others and end up with slightly worst matches.

4 Conclusion

We have proposed a novel energy function which handles prior depth information in two different representations: one leads to a standard binary problem, while the other one results in a multi-label energy. Experiments clearly show that our new term can be used easily and the reconstruction result improves considerably.

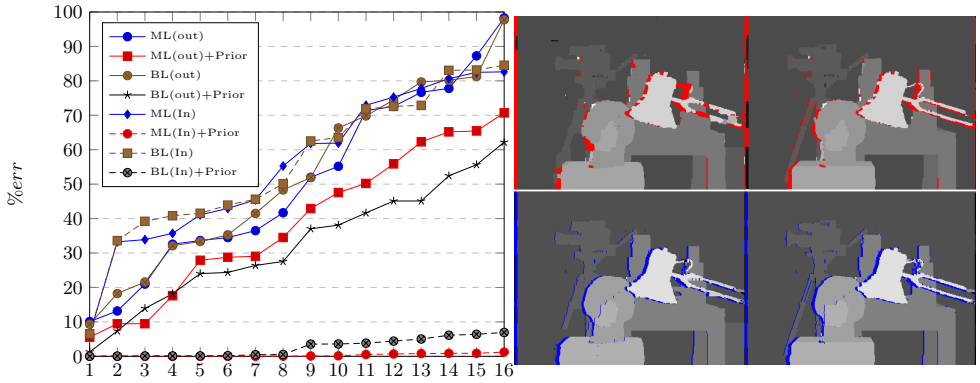


Fig. 1: Left: Error rate for the 2001, 2003 and 2005 Middleburry sets using both representations Inside(In) and outside(out) the region where the Prior depth is available. Right: Multi-label (red) & Binary Representation (blue) results on Tsukuba without the prior (Left column) and after using the *partial prior* (Right column). Colored pixels represent occluded pixels.

References

1. Stanley T. Birchfield and Carlo Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Trans. Pattern Anal. Mach. Intell.*
2. Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
3. Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*
4. Vladimir Kolmogorov, Pascal Monasse, and Pauline Tan. Kolmogorov and Zabih's Graph Cuts Stereo Matching Algorithm. *Image Processing On Line*.
5. Vladimir Kolmogorov and Ramin Zabih. Computing visual correspondence with occlusions via graph cuts.
6. Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
7. Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput. Vision*.
8. R.Zabih Y. Boykov, O. Veksler. Efficient approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

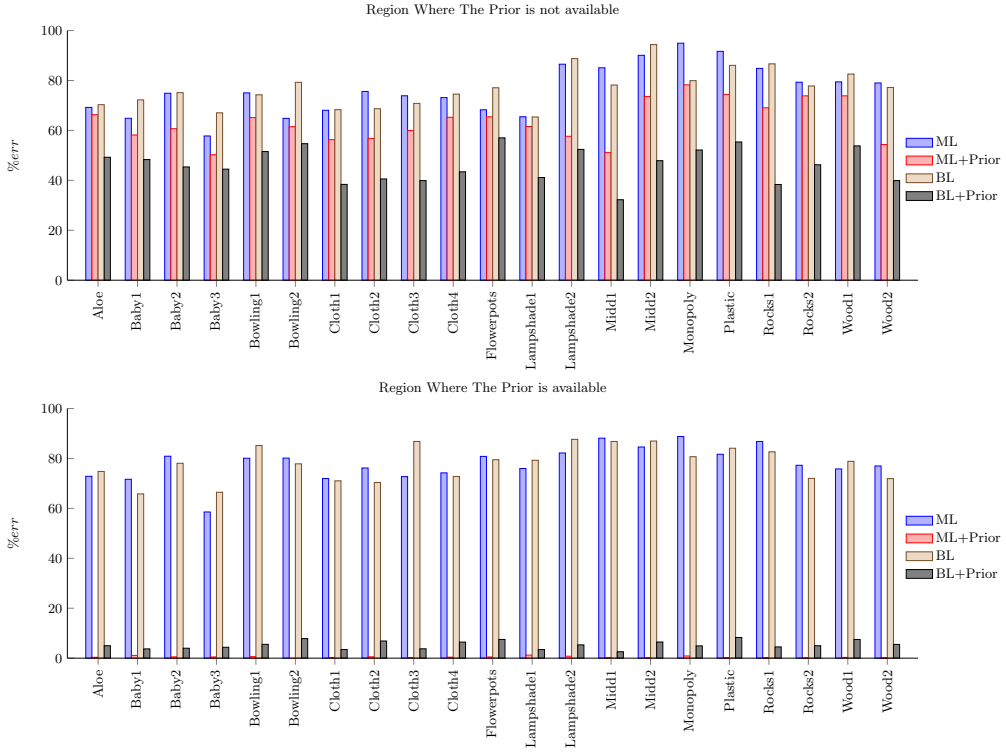


Fig. 2: The error rate obtained on the 2006 Middleburry stereo dataset including the prior using both representations.

Integer programming formulations for college admissions with ties

Kolos Csaba Ágoston[1,2], Péter Biró[1,2], Endre Kováts[3], and Zsuzsanna Jankó[4]

[1] Institute of Economics, Hungarian Academy of Sciences, Hungary
[2] Department of Operations Research and Actuarial Sciences, Corvinus University of Budapest, Hungary
[3] Budapest University of Technology and Economics
[4] University of Hamburg, Germany
peter.biro@krtk.mta.hu

Abstract. When two students with the same score are competing for the last slot at a university programme in a central admission scheme then different policies may apply across countries. In Ireland only one of these students is admitted by a lottery. In Chile both students are admitted by slightly violating the quota of the programme. Finally, in Hungary none of them is admitted, leaving one slot empty. We describe the solution by the Hungarian policy with various integer programing formulations and test them on a real data from 2008 with around 100,000 students. The simulations show that the usage of binary cutoff-score variables is the most efficient way to solve this problem when using IP technique. We also compare the solutions obtained on this problem instance by different admission policies. Although these solutions are possible to compute efficiently with simpler methods based on the Gale-Shapley algorithm, our result becomes relevant when additional constraints are implied or more complex goals are aimed, as it happens in Hungary where at least three other special features are present: lower quotas for the programmes, common quotas and paired applications for teachers studies.

Keywords: integer programming · college admissions · stable matching.

1 Introduction

Gale and Shapley gave a standard model for college admissions [15], where stable matching is was the solution concept suggested. Intuitively speaking a matching is stable if the rejection of an application at a college is explained by the saturation of that college with higher ranked students. Gale and Shapley showed that a stable matching can always be found by their so-called deferred-acceptance algorithm, which runs in linear time in the number of applications, see e.g. [16]. Moreover, the student-oriented variant results in the student-optimal stable matching, which means that no student could get a better assignment in any other stable matching. The theory of stable matchings have been intensively studied since 1962 by mathematicians/computer scientists (see e.g. [16])

and economists/game theorists (see e.g. [20]). The Gale-Shapley algorithm has also been used in practice all around the world [8], first in 1952 in the US resident allocation programme, called NRMP [18], then also in school choice, e.g. in Boston [1] and New York [2]. In Hungary, the national admission scheme for secondary schools follows the original Gale-Shapley model and algorithm [9], and the higher education admission scheme also uses a heuristic based on the Gale-Shapley algorithm [10].

The Hungarian higher education admission scheme have at least four important special features: the presence of ties, the lower and common quotas, and the paired applications. Each of the latter three special features makes the problem NP-hard [11], only the case of ties is resolvable efficiently [12]. In a recent paper [4] we studied the usage of integer programming techniques for finding stable solutions with regard to each of these four special features separately, and we managed to solve the case of lower quotas for the real instance of 2008. In this follow-up work we develop and test new IP formulations for the case of ties. The ultimate goal of this line of work is to suggest a solution concept for the college admission problem where ties and common quotas are also present, together with providing integer programming formulations that are suitable to compute this solution for large scale applications, such as the Hungarian university admission scheme with over 100,000 students.

First we start by investigating the basic Gale-Shapley model and then we consider the case of ties. **Due to the space limit we defer the description of IPs to the full version of the paper, here we present only the results of the simulations.**

2 Model descriptions

In this section first we present the classical Gale-Shapley college admission problem and then the case of ties.

2.1 The Gale-Shapley model

In the classical college admissions problem by Gale and Shapley [15] the students are matched to colleges.[5] In our paper we will refer the two sets as *applicants* $A = \{a_1, \ldots, a_n\}$ and *colleges* $C = \{c_1, \ldots c_m\}$. Let u_j denote the upper quota of college c_j. Regarding the preferences, we assume that the applicants provide strict rankings over the colleges, where r_{ij} denotes the ranking of the application (a_i, c_j) in applicant a_i's preference list. We suppose that the students are ranked according to their scores at the colleges, so college c_j ranks applicant a_i according to her score s_{ij}, where higher score is better. Let $E \subseteq A \times C$ denote the set of applications. A *matching* is a set of applications, where each student is admitted to at most one college and each college has at most as many assignees

[5] In the computer science literature this problem setting is typically called Hospital / Residents problem (HR), due to the National Resident Matching Program (NRMP) and other related applications.

and its quota, u_j. For a matching M let $M(a_i)$ denote the college where a_i is admitted (or $\emptyset$ if a_i is not allocated to any college) and let $M(c_j)$ denote the set of applicants admitted to c_j in M. A matching $M \subset E$ is *stable* if for any application (a_i, c_j) outside M either a_i prefers $M(a_i)$ to c_j or c_j filled its seats with u_j applicants who all have higher scores than a_i has. The deferred-acceptance algorithm of Gale and Shapley provides a student-optimal stable matching in linear time [15].

The notion of *cutoff scores* is important for both the classical Gale-Shapley model and its generalisations with ties and common quotas. Let t_j denote the cutoff score of college c_j and let $\mathbf{t}$ denote a set of cutoff scores. We say that matching M is implied by cutoff scores $\mathbf{t}$ if every student is admitted to the most preferred college in her list, where she achieved the cutoff score. We say that a set of cutoff scores $\mathbf{t}$ *corresponds* to a matching M if $\mathbf{t}$ implies M. For a matching M an applicant a_i has *justified envy* towards another applicant a_k at college c_j if $M(a_k) = c_j$, a_i prefers c_j to $M(a_i)$ and a_i is ranked higher than a_k at c_j (i.e. $s_{ij} > s_{kj}$). A matching with no justified envy is called *envy-free* (see [22] and [21]).

It is not hard to see that a matching is envy-free if and only if it is implied by some cutoff scores [3]. Note that an envy-free matching might not be stable because of blocking with empty seats, i.e. when a student a_i prefers c_j to $M(a_i)$ and c_j is not saturated (i.e. $|M(c_j)| < u_j$). In this case a matching is called *wasteful*. Again, by definition it follows that a matching is stable if and only if it is envy-free and non-wasteful (see also [6]). To achieve non-wastefulness we can require the cutoff of any unsaturated college to be minimum (zero in our case). Alternatively we may require that no cutoff score may be decreased without violating the quota of that college, while keeping the other cutoff scores. Furthermore, we may also satisfy the latter condition by ensuring that we select the student-optimal envy-free matching, which is the same as the student-optimal stable matching [22]. To return this solution we only need to use an appropriate objective function. We will use the above described connections when developing our IPs.

2.2 Case of ties

In many nationwide college admission programmes the students are ranked based on their scores, and ties may appear. In Hungary, for instance, the students can obtain integer points between 0 and 500 (the maximum was 144 until 2007), so ties do occur. When ties are present then one way to resolve this issue is to break ties by lotteries, as done in Ireland (so a lucky student with 480 point may be admitted to law studies, whilst an unlucky student with the same score may be rejected). However, the usage of lotteries can be seen unfair, so in some countries, such as Hungary [12] and Chile [17] equal treatment policies are used, meaning that students with the same score are either all accepted or all rejected. In case of such a policy, there are two reasonable variants when deciding about the last group of students without whom the quota is unfilled and with whom the quota is violated. In the restrictive policy, used in Hungary, the quotas are

never violated, so this last group of students is always rejected, whilst in Chile they use a permissive policy and they always admit this last group of students. For instance, if there are three students, a_1, a_2 and a_3, applying to a programme of quota 2 with scores 450, 443, and 443, respectively then in Hungary only a_1 is admitted, whilst in Chile all three students are admitted. In Ireland, a_1 is admitted and they use a lottery to decide whether a_2 or a_3 will get the last seat.

Stable matchings for the case of ties were defined through the cutoff scores in [12]. The usage of cutoff scores in case of ties make the solution *envy-free*, meaning that no student a_i may be rejected from college c_j if this college admitted another student with score equal to or lower than the score of student a_i. This allocation concept is called also equal treatment policy, as the admission of a student to a programme implies the admission offer to all other students with the same score. A matching is envy-free for college admission problem with ties if and only if it is induced by cutoff scores [3]

For the restrictive policy used in Hungary, the stability of the matching can be defined by adding a non-wastefulness condition to envy-freeness. Namely, a matching induced by cutoff scores is *stable* if no college can decrease its cutoff score without violating its quota, assuming that the other cutoff scores remain the same. In the more permissive Chilean policy a matching is stable if by decreasing the cutoff score of any college there would be empty seats left there. (We note that the stability of a matching can be equivalently defined by the lack of a set of blocking applications involving one college and a set of applicants such that this set of applications would be accepted by all parties when compared to the applications of the matching considered. See more about this connection in [14].)

Biró and Kiselgof [12] showed two main theorems about stable matchings for college admissions with ties. In their first theorem they showed that a student-optimal and a student-pessimal stable matchings exist for both the restrictive policy (Hungary) and the permissive policy (Chile), where the cutoff scores are minimal / maximal, respectively. Furthermore, they also proved the intuitive results that if we compare the student-optimal cutoff scores (or the student-pessimal ones) with respect to the three reasonable policies, namely the Hungarian (restrictive), the Irish (lottery), and the Chilean (permissive), then the Hungarian cutoff scores are always as high for each college than the Chilean cutoff scores and the Irish cutoff scores are in between these. When considering the student-optimal stable matching, it turns out to be also the student-optimal envy-free matching, as described in [3].

3 Simulations

In this section we present the main simulation results.

3.1 Gale-Shapley model

We took the 2008 data after breaking the ties randomly, by considering only the faculty quotas and keeping only the highest ranked application of each student

for every programme (i.e. the application for either the state funded or the privately funded seat). We used AMPL with Gurobi for solving the IPs.

IP formulations	#variables	#constraints	#non-0 elem.	size(Kb)	run time(s)
SO-BB	287,035	381,115	73,989,595	1,319,663	1,139
SO-NW-CUT	291,935	673,050	2,463,808	69,464	81
MIN-CUT	289,485	668,150	2,169,423	64,254	5,062
MSMR-CUT	289,485	668,150	2,169,423	69,846	2,318
SO-NW-BIN-CUT	574,070	955,185	3,028,078	75,810	107
MIN-BIN-CUT	574,070	952,735	2,738,593	65,657	871
MSMR-BIN-CUT	574,070	952,735	2,738,593	66,467	4,325
MSMR-EF	n.a.	n.a.	n.a.	8,667,403	n.a.

3.2 Case of ties

We used the 2008 data with the original ties by considering again the faculty quotas and keeping only the highest ranked application of each student for every programme.

IP formulations	#variables	#constraints	#non-0 elem.	size(Mb)	run time(s)
MIN-CUT	289,485	668,150	2,169,423	59,694	5,247
MSMR-CUT	289,485	668,150	2,169,423	65,286	1,460
MIN-BIN-CUT	428,513	807,178	2,447,479	53,548	982
MSMR-BIN-CUT	428,513	807,178	2,447,479	57,106	1,362
SO-H-NW-CUT	578,970	1,694,333	4,793,409	114,882	1,310
SO-H-NW-BIN-CUT	861,105	1,813,840	5,352,772	118,828	165

Finally, we conducted the simulation on the same 2008 data, where we compared the results for the Hungarian, Irish and Chilean policies. The results indeed follow the theorems of [12] regarding the cutoff scores for the three different policies. The most interesting fact of the simulation is that for the Hungarian and Irish policies the difference between the student-optimal and student-pessimal solutions is minor, as demonstrated also in earlier paper for large markets, such as [19]. However, for the Chilean policy this difference was more significant.

	size of matching		average rank		average cutoffs		# rejections	
policies	A-opt.	C-opt.	A-opt.	C-opt.	A-opt.	C-opt.	A-opt.	C-opt.
Hungarian	86,195	86,195	1.2979	1.2979	58.3931	58.3931	37,698	37,698
Irish	86,410	86,410	1.2916	1.2916	58.2090	58.2106	36,802	36,804
Chilean	86,650	86,614	1.2824	1.2844	57.2502	57.5200	35,668	35,901

Acknowledgements

The authors were supported by the Hungarian Academy of Sciences under its Momentum Programme (LP2016-3/2018) and Cooperation of Excellences Grant (KEP-6/2018).

References

1. Abdulkadiroğlu A., Pathak A., and Roth A.E. The New York City high school match. *American Economic Review, P&P*, 95(2):364–367, 2005.
2. Abdulkadiroğlu A., Pathak P.A., Roth A.E., and Sönmez T. The Boston public school match. *American Economic Review, P&P*, 95(2):368–371, 2005.
3. Ágoston K.Cs., and Biró P. Modelling Preference Ties and Equal Treatment Policy. *In Proceedings of ECMS 2017: 31st European Conference on Modelling and Simulation*, pages 516-522, 2017.
4. Ágoston K.Cs., Biró P., and McBride I. Integer programming methods for special college admissions problems. *Journal of Combinatorial Optimization*, 32(4):1371–1399, 2016.
5. Ágoston K.Cs., Biró P., and Szántó R. Stable project allocation under distributional constraints. *Operations Research Perspectives*, 5:59–68, 2018.
6. Azevedo, E.M., and Leshno, J.D. A supply and demand framework for two-sided matching markets. *Journal of Political Economy*, 124(5):1235–1268, 2016.
7. Baïou M., and Balinski M. The stable admissions polytope. *Mathematical Programming*, 87(3), Ser. A:427–439, 2000.
8. Biró, P. Applications of matching models under preferences. In Endriss, U., editor, *Trends in Computational Social Choice*, chapter 18, pages 345–373. AI Access. 2017.
9. Biró P. Matching Practices for Secondary Schools – Hungary. `matching-in-practice.eu`, accessed on 23 August 2014.
10. Biró P. University admission practices - Hungary. `matching-in-practice.eu`, accessed on 23 August 2014.
11. Biró, P., Fleiner, T., Irving, R.W, and Manlove, D.F. The College admissions problem with lower and common quotas. *Theoretical Computer Science*, 411:3136–3153, 2010.
12. Biró, P., and Kiselgof, S. College admissions with stable score-limit. *Central European Journal of Operations Research*, 23(4):727–741, 2015.
13. Biró P., McBride I., and Manlove D.F. The Hospitals / Residents problem with Couples: Complexity and Integer Programming models. *In Proceedings of SEA 2014: the 13th International Symposium on Experimental Algorithms*, vol 8504 of LNCS, pp 10-21, Springer, 2014.
14. Fleiner T., and Jankó Zs. Choice Function-Based Two-Sided Markets: Stability, Lattice Property, Path Independence and Algorithms. *Algorithms*, 7(1):32–59, 2014.
15. Gale D., and Shapley L.S. College Admissions and the Stability of Marriage. *American Mathematical Monthly*, 69(1):9–15, 1962.
16. Manlove D.F. Algorithms of Matching Under Preferences. *World Scientific Publishing*, 2013.
17. Rios I., Larroucau T., Parra G., and Cominetti R. College admissions problem with ties and flexible quotas. *Working paper*, 2014.
18. Roth A.E. The evolution of the labor market for medical interns and residents: a case study in game theory. *Journal of Political Economy*, 6(4):991–1016, 1984.
19. Roth A.E., and Peranson E. The Redesign of the Matching Market for American Physicians: Some Engineering Aspects of Economic Design. *American Economic Review*, 89:748–780, 1999.
20. Roth A.E., and Sotomayor M.A.O. Two-sided matching: a study in game-theoretic modeling and analysis. *Cambridge: Econometric Society monographs*, 1990.
21. Yokoi Y. Envy-Free Matchings with Lower Quotas. arXiv preprint, arXiv:1704.04888, 2017.
22. Wu Q., and Roth A.E. The lattice of envy-free matchings. mimeo, 2016.

IP Solutions for International Kidney Exchange Programmes

Péter Biró[1,2], Márton Gyetvai[1,2], Radu-Stefan Mincu[3], Alexandru Popa[3], and Utkarsh Verma[4]

[1] Institute of Economics, Hungarian Academy of Sciences, Hungary
[2] Department of Operations Research and Actuarial Sciences, Corvinus University of Budapest, Hungary
[3] Department of Computer Science, University of Bucharest, Romania
[4] Department of Industrial Engineering and Operations Research, IIT Bombay, India
peter.biro@krtk.mta.hu

Abstract. In kidney exchange programmes patients with end-stage renal failure may exchange their willing, but incompatible living donors among each other. National kidney exchange programmes are in operation in ten European countries, and some of them have already conducted international exchanges through regulated collaborations. The exchanges are selected by conducting regular matching runs (typically every three months) according to well-defined constraints and optimisation criteria, which may differ across countries. In this work we give integer programming formulations for solving international kidney exchange problems, where the optimisation goals and constraints may be different in the participating countries and various feasibility criteria may apply for the international cycles and chains. We also conduct simulations showing the long-run effects of international collaborations for different pools and under various national restrictions and objectives.

Keywords: integer programming · kidney exchanges · simulations.

1 Introduction

When an end-stage kidney patient has a willing, but incompatible living donor, then in many countries this patient can exchange his/her donor for a compatible one in a so-called kidney exchange programme (KEP). The first national kidney exchange programme was established in 2004 in the Netherlands in Europe [9]. Currently there are ten countries with operating programmes in Europe [6], the largest being the UK programme [11].

Typically the matching runs are conducted in every three months on pools with around 50-300 patient-donor pairs. The so-called virtual compatibility graph represents the patient-donor pairs with nodes and an arc represents a possible donation between the corresponding donor and patient, that is found compatible in a virtual crossmatch test. The exchange cycles are selected by well-defined optimisation rules, that can vary across countries. The most important constraints are the upper limits on the length of exchange cycles, for examples, two

in France, three in the UK and Spain, and four in the Netherlands [6]. The main goal of the optimisation in Europe is to facilitate as many transplants as possible, i.e. to maximise the number of nodes covered in the compatibility graph by independent cycles. The corresponding computational problem for cycle-length limits three or more is NP-hard, and the standard solution technique used is integer programming [1].

International kidney exchanges have already started in Europe between Austria and Czech Republic [7] since 2016, between Portugal, Spain and Italy since summer 2018, and between Sweden, Norway and Denmark in the Scandiatransplant programme (STEP), built on the Swedish initiative [2]. The above mentioned first two collaborations are organised in a sequential fashion, first the national runs are conducted and then the international exchanges are sought for the remaining patient-donor pairs. A related game-theoretical model has been studied in [8]. In the Scandinavian programme, however, the protocol proposed is to find an overall optimum for the joint pool. In the latter situation, the fairness of the solution for the countries involved can be seen as an important requirement, which was studied in [10] with extensive long-term simulations by proposing the usage of a compensation scheme among the countries. [5]

In this study we will compare the sequential and the joint pool scenarios in our simulations. We will not consider compensations, or any strategic issues, but we will allow the countries to have different constraints and goals with regard to the cycles and chains they may be involved in. In particular, we will compare the benefits of the countries from international collaborations when they have different upper bounds on their national cycles, and thus also possible different constraints on the segments of the international cycles they are participating in. As an example, we mention the Austro-Czech cooperation, where Austria requires on having all exchanges simultaneously, so they allow short national cycles and short segments only, whilst in Czech Republic the longer non-simultaneous cycles and chains are also allowed. We formulate novel IP models for dealing with potentially diverse constraints and goals in international kidney exchange programmes and we test two-country cooperations under different assumptions over their constraints, the possibility of having chains triggered by altruistic donors, and the sizes and compositions of their pools.

2 Model of international kidney exchanges

In a standard kidney exchange problem, we are given a directed *compatibility graph* $D(V, A)$, where the nodes $V = \{1, 2, \dots n\}$ correspond to patient-donor pairs and there is an arc (i, j) if the donor of pair i is compatible with the patient of pair j. Furthermore we have a non-negative weight function w on the arcs, where $w_{i,j}$ denotes the weight of arc (i, j), representing the value of the transplantation. (In most applications the primary concern is to save as many patients as possible, so the value is simply equal to one.)

[5] Similar situation arises in the US kidney exchange problem, where the transplant centres are the strategic agents [4, 5, 3, 13].

Let $\mathcal{C}$ denote the set of cycles allowed in D, which are typically to be of length at most K. The solution of a classical kidney exchange problem is a set of disjoint cycles of $\mathcal{C}$, i.e. a cycle-packing in D. For cycle $c \in \mathcal{C}$, let $A(c)$ denote the set of arcs in c and $V(c)$ denote the set of nodes covered by c.[6]

In an international kidney exchange programme multiple countries (N) are involved in the exchange, so the set of nodes is partitioned into $V = V^1 \cup V^2 \cup \cdots \cup V^N$, where V^k is the set of patient-donor pairs in country k. We have the following modifications of the classical problem. Let A^k denote the arcs pointing to V^k (so the donations to patients in country k). Note that $A = A^1 \cup A^2 \cup \cdots \cup A^N$. The weights of the arcs in A^k should reflect the preferences of country k. (We may assume that these are scaled, e.g. by having the same average score in order not to bias the overall optimal solution towards some countries.) Finally, let $A^{\mathcal{N}}$ and $A^{\mathcal{I}}$ denote the national and international donations, i.e. $A = A^{\mathcal{N}} \cup A^{\mathcal{I}}$.

In a global optimal solution, small cycles within the countries can have different requirement than international cycles. Therefore, we separate the two sets of cycles into $\mathcal{C} = \mathcal{C}^{\mathcal{N}} \cup \mathcal{C}^{\mathcal{I}}$, where $\mathcal{C}^{\mathcal{N}}$ is the set of national cycles and $\mathcal{C}^{\mathcal{I}}$ is the set of international cycles. We call the national parts of an international cycle *segments*, where a segment is a path within a country, and the segments are linked by international arcs. A l-segment is a path of length $l-1$, with all the l nodes belonging to the same country. Let $\mathcal{S}$ denote the set of all possible segments, and let $\mathcal{S}^k$ denote the set of segments allowed in country k. For $s \in \mathcal{S}$, let $A(s)$ denote the set of (national) arcs and let $V(s)$ denote the set of nodes covered (in the same country). Note that a segment may also consist of a single node, which corresponds to the case when an international donation is immediately followed by another international donation. We can have the following natural restrictions on the national and international cycles:[7]

1. different limits on the length of national cycles for each country;
2. different limits on the length of segments in international cycles for each country;
3. limit on the total length of an international cycle;
4. limit on the number of countries involved in one cycle;
5. limit on the number of patient-donor pairs from a country in one cycle;
6. limit on the number of segments in a country within one cycle.

[6] In addition, we can also consider *altruistic donors*, in which case we separate the node set into patient-donor pairs V_p and altruistic donors V_a, so $V = V_p \cup V_a$. The solution may contain exchange cycles and chains triggered by altruistic donors. The latter can be conducted non-simultaneously, so different restrictions may apply for them. In this paper we focus on cycles, but we note that one can reduce the problem of finding chains to the problem of finding cycles by adding artificial patients to the altruistic donors, who are compatible with all donors.

[7] We can also have different constraints for altruistic chains, and we may require that an international chain may have to end in the same country where it started.

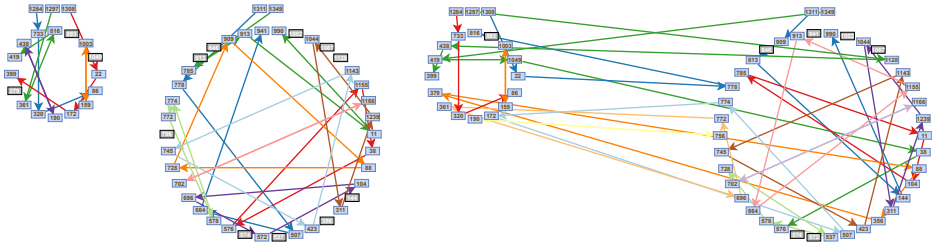
Integer programming formulations and simulation plan: We propose new integer programming formulations, where besides the standard edge- and cycle-formulations [1], we introduce new variables for country segments. We defer this part, together with the description of the simulation plan, to the journal version of the paper. Below we only present one simulation.

3 A simulation example

To determine the benefits of international kidney exchange programmes (KEPs) we conducted a case study involving two countries which aim to develop a joint KEP and are concerned about the advantages and disadvantages of cooperation between their KEPs. We compare the individual benefits from the no cooperation case to the sequential matchings and merged pool scenarios.

The simulation involves 20 instances each containing the compatibility information for 1000 patient-donor pairs. We assume that an extra 10% of this amount are altruist donors. The length of the considered time-frame in the simulated kidney exchange schemes is 5 years with matching runs occurring every 3 months for each instance, as in [12]. Each agent is assigned an uniformly distributed arrival time, and the patient-donor pairs stay in the KEP for a maximum of 1 year (or 4 matching runs) after which they leave the programme (which means that they opt for an alternative solution, such as having a direct transplant after desensitisation or getting an organ from a deceased donor).

Fig. 1. Graphic representation of the first KEP stage in one of the instances: altruist donors are at the top, patient-donor pairs form circles for each country and arcs represent transplants. Left side, individual KEPs: 13/16 patients receive transplants in the small country, 28/38 patients in the large country are transplanted. Right side, merged KEP: the numbers are 15/16 for the small country, and 32/38 for the large one.



In the illustrative simulation we have two countries. Country 1 is twice the size of Country 2, meaning that it will have roughly twice as many patient-donor pairs, as well as altruist donors. On average, each month will mean the arrival of 33.33 patients to Country 1 and 16.66 to Country 2. Country 2 runs a smaller programme and allows only 2-cycles and 3-chains, while country 1 allows for 3-cycles and 4-chains. When they collaborate, they allow international 2-cycles, international 3-cycles where there is only one patient-donor pair involved from Country 2 and chains that must end in the same country where the altruist

donor is coming from. The objective is simply to maximise the number of transplants. There are three settings for collaboration: no cooperation (i.e. separate KEPs, baseline scenario), sequential matchings (each country runs a local KEP optimisation and then the remaining patient pools enter a joint KEP) and full collaboration (a single KEP for both countries). We present our findings in Fig. 1, Fig. 2 and Table 1.

Fig. 2. Average across 20 instances of transplants and dropouts from the KEP recorded after each of the 20 stages of the KEP.

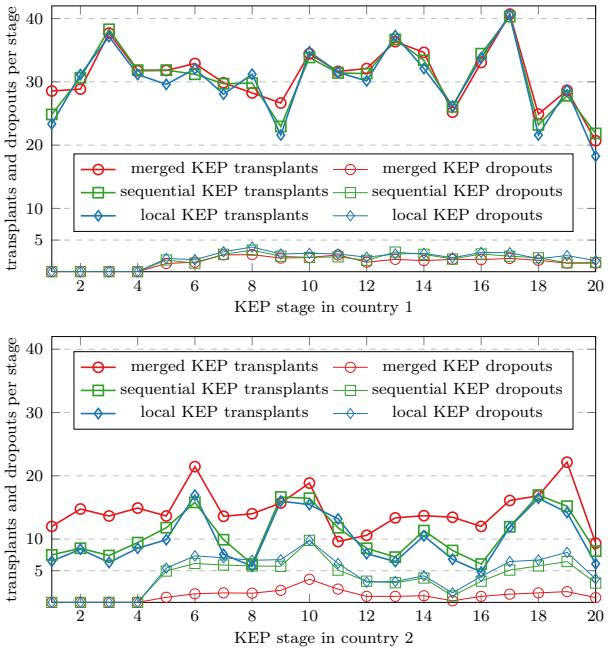


Table 1. Average total transplants (tr.) and total patients who drop out (d.o.) of the programme across all 20 instances after 5 years.

	tr.	d.o.
C1 local	600.2	42.15
C2 local	199.45	90.2
C1 seq.	611.45	36.35
C2 seq.	214.85	78.35
C1 joint	618.7	30.85
C2 joint	289.7	22.15

We observe firstly that the size of the KEP donor pool is important to increase the number of compatible transplants: the smaller country struggles with a lower rate of transplants than the larger one, and has a significant 31% drop out rate. While the larger country does not see much benefit from entering a joining KEP with the smaller country, we can observe that its number of transplants does not decrease when international collaboration increases. However, the smaller country benefits greatly, especially in the case of merged KEPs, where the drop out rate is significantly lowered to about 7%, a value similar to the drop out rate of the single larger country's individual KEP scenario. This information suggests that newly developing and smaller KEPs have much to gain from full collaboration with a larger KEP. On the other hand, the improvement in the sequential KEP scenario is much less than that of the fully joint KEP for the smaller country.

Acknowledgements: We acknowledge the support of the ENCKEP COST Action for short term scientific missions for Biró, Gyetvai, Mincu and Popa. Biró was also supported by the Hungarian Academy of Sciences under its Momentum Programme (LP2016-3/2018) and Cooperation of Excellences Grant (KEP-6/2018), and by the Hungarian Scientific Research Fund – OTKA (no. K129086).

References

1. D.J. Abraham, A. Blum, and T. Sandholm. Clearing algorithms for barter exchange markets: enabling nationwide kidney exchanges. In *Proc. EC '07: the 8th ACM Conference on Electronic Commerce*, pages 295–304. ACM, 2007.
2. T. Andersson and J. Kratz. Pairwise Kidney Exchange over the Blood Group Barrier. *Lund University Department of Economics Working Paper 2016:11*, 2016.
3. I. Ashlagi, F. Fischer, I.A. Kash, and A.D. Procaccia. Mix and match: A strategyproof mechanism for multi-hospital kidney exchange. *Games and Economic Behavior*, 91(Supplement C):284 – 296, 2015.
4. I. Ashlagi and A.E. Roth. New challenges in multihospital kidney exchange. *American Economic Review*, 102(3):354–359, 2012.
5. I. Ashlagi and A.E. Roth. Free riding and participation in large scale, multi-hospital kidney exchange. *Theoretical Economics*, 9(3), 2014.
6. P. Biró, B. Haase, and et al.˙ Building kidney exchange programmes in Europe – An overview of exchange practice and activities. *Transplantation (to appear)*, 2018.
7. G.A. Böhmig, J. Fronek, A. Slavcev, G.F. Fischer, G. Berlakovich, and O. Viklicky. Czech–Austrian kidney paired donation: first European cross-border living donor kidney exchange. *Transplant International*, 30(6):638–639, 2017.
8. M. Carvalho, A. Lodi, J.P. Pedroso, and A. Viana. Nash equilibria in the two-player kidney exchange game. *Mathematical Programming*, 161(1-2):389–417, 2017.
9. M. De Klerk, K.M. Keizer, F.H.J. Claas, M. Witvliet, B. Haase-Kromwijk, and W. Weimar. The Dutch national living donor kidney exchange program. *American Journal of Transplantation*, 5(9):2302–2305, 2005.
10. X. Klimentova, N. Santos, J.P. Pedroso, and A. Viana. Fairness models for multi-agent kidney exchange programs. *Working paper*.
11. D.F. Manlove and G. O'Malley. Paired and altruistic kidney donation in the UK: Algorithms and experimentation. *ACM Journal of Experimental Algorithmics*, 19(2), article 2.6, 21pp, 2014.
12. N. Santos, P. Tubertini, A. Viana, and J.P. Pedroso. Kidney exchange simulation and optimization. *Journal of the Operational Research Society*, pages 1–12, 12 2017.
13. P. Toulis and D.C. Parkes. Design and analysis of multi-hospital kidney exchange mechanisms using random graphs. *Games and Economic Behavior*, 91(Supplement C):360 – 382, 2015.

Markov decision processes with total effective payoff

Endre Boros[1], Khaled Elbassioni[2], Vladimir Gurvich[1], and Kazuhisa Makino[3]

[1] Rutgers University, NJ, USA
[2] Masdar Institute, Abu Dhabi, UAE
[3] RIMS, Kyoto, Japan
eboros@business.rutgers.edu

Abstract. We consider finite state Markov decision processes with undiscounted total effective payoff. We show that there exist uniformly optimal pure and stationary strategies that can be computed by solving a polynomial number of linear programs. This implies that in a two-player zero-sum stochastic game with perfect information and with total effective payoff there exists a stationary best response to any stationary strategy of the opponent. From this, we derive the existence of a uniformly optimal pure and stationary saddle point. Finally we show that the traditional mean payoff can be viewed as a special case of total payoff.

Keywords: Markov processes · Stationary strategies.

We consider finite state, finite action Markov decision processes with *undiscounted total effective payoff*. We denote by V the set of states, and by $v_t \in V$ the state at which the system is at time t. The controller (MAX) chooses an action, that results in a transition to state v_{t+1}. Note that this transition is stochastic, and thus v_t, $t = 0, 1, ...$ are random variables. Every transition $v_t \to v_{t+1}$ results in a *local reward* $r(v_t, v_{t+1})$ that is known in advance and depends only on the pair of states.

A policy (strategy) of MAX is a mapping that to any time moment t and state v_t assigns a choice of actions. Such a policy maybe stochastic, may depend on the history of previous choices, etc. We say that a policy is *positional* if this choice depends only on the current state. We say that a policy is *deterministic* if actions are chosen with $0/1$ probabilities. Finally we say that a policy is *uniformly optimal*, if it is an optimal policy for all possible initial states.

Once an initial state $v_0 \in V$ is fixed, and MAX chooses a strategy $\mathfrak{s} \in \mathfrak{S}$, the above process produces a series of states $v_t(\mathfrak{s}) \in V$, $t = 0, 1, \ldots$, which generally are random variables. We associate to such a process the sequence of expected local rewards and consider two payoff functions that measure the quality of such an infinite process:

$$\phi_{\mathfrak{s}}(v_0) \;=\; \liminf_{T\to\infty} \frac{1}{T+1} \sum_{t=0}^{T} \mathbb{E}_{\mathfrak{s}}[r(v_t, v_{t+1})], \tag{1}$$

$$\psi_{\mathfrak{s}}(v_0) \;=\; \liminf_{T\to\infty} \frac{1}{T+1} \sum_{t=0}^{T} \sum_{j=0}^{t} \mathbb{E}_{\mathfrak{s}}[r(v_j, v_{j+1})]. \tag{2}$$

The first one, called *mean payoff*, is classic [4], [1]. The second one, called *total payoff* or *total reward*, was introduced by [8], as a "refinement" of the mean payoff. For instance, if local rewards represent rate of return on our investment, than maximizing $\phi_{\mathfrak{s}}(v_0)$ provides us with an optimal investment policy. If however local rewards are transactions in and out of our account, then $\Psi_{\mathfrak{s}}(v_0)$ is related to the current account balance, and maximizing it makes perfect sense.

We also consider the two person game version, in which MIN is an adversary of MAX and tries to minimize the same objective. In this version it is assumed that some states are controlled by MAX, while the other states are controlled by MIN.

Our first result counters the intuitive heuristic view of [8] cited above:

Theorem 1. *Mean payoff is a special case of total payoff, in the sense that to every system with payoff function ϕ one can associate another system (roughly twice as many states) with payoff function ψ such that there is a one-to-one correspondence between policies and $\psi_{\mathfrak{s}}(v_0) = \phi_{\mathfrak{s}'}(v'_0)$ holds for corresponding policies $\mathfrak{s}$ and $\mathfrak{s}'$.*

Our main result is about the existence and efficient computability of optimal policies:

Theorem 2. *In every MDP with total effective payoff ψ,* MAX *possesses a uniformly optimal deterministic positional strategy. Moreover, such a strategy, together with the optimal value can be found in polynomial time.*

For mean payoff MDPs, the analogous result is well-known, see, e.g. [5], [1], [3], [7]. In fact there are several known approaches to construct the optimal stationary strategies. For instance, a polynomial-time algorithm to solve mean payoff MDPs is based on solving two associated linear programs, see, e.g., [3].

Our approach for proving Theorem 2 is inspired by a result of [9]. We extend their result to characterize the existence of pure and stationary optima within *all* possible strategies by the feasibility of an associated system of equations and inequalities. Next, we show that this system is always feasible and a solution can be obtained by solving a polynomial number of linear programming problems.

Theorem 3. *Every two-person game with total effective payoff ψ has a value and a uniformly optimal deterministic positional equilibrium.*

For the mean payoff games with perfect information the analogous result is well-known [4], [6].

The full version of our paper on these and additional results can be found at [2].

References

1. Blackwell, D.: Discrete dynamic programming. Ann. Math. Statist., **33** 719–726 (1962)
2. Boros, E., Elbassioni, K., Gurvich V., Makino, K.: Markov decision processes and stochastic games with total effective payoff. Annals of Operations Research, May 2018. ISSN 1572-9338. doi: 10.1007/s10479-018-2898-8. https://doi.org/10.1007/s10479-018-2898-8
3. Derman, C.: Finite State Markov decision processes. Academic Press, New York and London (1970)
4. Gillette, D.: Stochastic games with zero stop probabilities. In: Dresher, M., Tucker, A. W. and Wolfe, P. (eds) Contribution to the Theory of Games III, volume 39 of Annals of Mathematics Studies, pp. 179–187. Princeton University Press (1957)
5. Howard, R. A.: Dynamic programming and Markov processes. Technology press and Willey, New York (1960)
6. Liggett, T. M., Lippman, S. A.: Stochastic games with perfect information and time-average payoff. SIAM Review **4** 604–607 (1969)
7. Mine, V., Osaki, S.: Markovian decision process. American Elsevier Publishing Co., New York (1970)
8. Thuijsman, F., Vrieze, O. J.: The bad match, a total reward stochastic game. Operations Research Spektrum, **9**, 93–99 (1987)
9. Thuijsman, F., Vrieze, O. J.: Total reward stochastic games and sensitive average reward strategies. Journal of Optimization Theory and Application **98** 175–196 (1998) ISSN 0022-3239. http://dx.doi.org/10.1023/A:1022697100194

New Integer Programming Formulations for the Stable Exchange Problem*

Virginia Costa[1], Xenia Klimentova[1], Péter Biró[2], Ana Viana[1,3], and João Pedro Pedroso[1,4]

[1] INESC TEC, Porto, Portugal
[2] Institute of Economics, Hungarian Academy of Sciences, Budapest, Hungary
[3] ISEP - School of Engineering, Polytechnic of Porto, Porto, Portugal
[4] Faculdade de Ciências, Universidade do Porto, Porto, Portugal
xenia.klimentova@inesctec.pt

Abstract. In the stable exchange problem the agents are endowed with a single good, e.g. a house or a kidney donor, and they have preferences over the others' endowments. The problem is to find an exchange of goods such that no group of agents can block the solution in an exchange cycle. An exchange is called stable if there is no blocking cycle where all the agents involved strictly prefer the new solution. An exchange is strongly stable if no weakly blocking cycle exists, where at least one agent improves and neither of them gets a worse allocation. When the lengths of the exchange cycles is not limited then a stable solution always exists and can be found efficiently by Gale's Top Trading Cycle algorithm. However, when the length of the exchange cycles is limited then a (strongly) stable solution may not exist and the problem of deciding the existence is NP-hard. This setting is particularly relevant in kidney exchange programs, where the length of exchange cycles is limited due to the simultaneity of the transplantations, e.g. the maximum length of the cycles is 3 in the UK and 4 in the Netherlands. In this work we develop several integer programming formulations to solve the (strongly) stable exchange problem, which is a novel approach for this solution concept. We compare the effectiveness of these models by conducting computational experiments on generated kidney exchange data.

Keywords: stable matching · integer programming · k-way exchange.

* This work is financed by the ERDF European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme, and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within project "mKEP - Models and optimisation algorithms for multicountry kidney exchange programs" (POCI-01-0145-FEDER-016677), by FCT project SFRH/BPD/101134/2014 and by COST Action CA15210 ENCKEP, supported by COST (European Cooperation in Science and Technology) – http://www.cost.eu/. Biró is supported by the Hungarian Academy of Sciences under its Momentum Programme (LP2016-3/2018) and Cooperation of Excellences Grant (KEP-6/2018), and by the Hungarian Scientific Research Fund – OTKA (no. K129086).

1 Introduction

Barter exchange markets – such as kidney exchange programs – can be represented as directed graphs where agents are vertices and arcs indicate exchange opportunities. A solution consists of a set of disjoint cycles. In this paper we consider the case where agents have preferences, represented by ranks on outgoing arcs. An exchange that contains no cycle with length more than k is a *k-way exchange*. A *k-way stable exchange* is a k-way exchange such that there is no cycle where all the vertices would be better off, according to their preferences, than in the current solution. When strict preference in the blocking cycle is required only for one vertex then we speak about *strongly stable exchanges*. The problem of deciding existence is NP-hard for both problems [2, 5]. In this work, we present three novel integer programming formulations for these problems, which is a novel approach in the literature. Preliminary computational results highlight the efficiency of one formulation over the others.

1.1 Notation and definitions

Consider a digraph $G = (V, A)$, where V is the set of vertices and A is the set of arcs. Define also the *preference list* of $i \in V$ as the set $\delta(i) = \{j \mid (i, j) \in A\} \subseteq V$ where there is a strict preference order on its elements. Each $j \in \delta(i)$ is ranked with value $r \in \{1, \ldots, |\delta(i)|\}$. For $j, j' \in \delta(i)$ ranked with r, r', respectively, we say that vertex i prefers j to j', and denote by $j <_i j'$, if $r' > r$.

Within this context, a *matching* $\mathcal{M} \subset A$ is a set of pairs (i, j) where $i \in V$ and $j \in \delta(i)$. In addition, a vertex always prefers to be matched to any of the elements in its preference list, rather than be unmatched. A vertex i is *unmatched* if there is no vertex j such that $(i, j) \in \mathcal{M}$. Let $\mathcal{C}$ be a set of cycles in G of length at most k. We denote by $V(c)$ and $A(c)$ the set of vertices and arcs, respectively, that are involved in a cycle $c \in \mathcal{C}$. We say that $c \in \mathcal{M}$ if, and only if, $A(c) \subseteq \mathcal{M}$. Let $|c|$ denote the length of cycle c, i.e., $|c| = |V(c)| = |A(c)|$. Let $\mathcal{C}(i) \subseteq \mathcal{C}$ be the set of cycles that contain vertex i. We say that vertex i *prefers* cycle $c \in \mathcal{C}(i)$ over cycle $c' \in \mathcal{C}(i)$, and denote by $c \prec_i c'$, if for $(i, j) \in A(c)$ and $(i, j') \in A(c')$, $j <_i j'$.Vertex i is *indifferent* between cycles c and c' if there exists a vertex j such that $(i, j) \in A(c) \cap A(c')$, i.e., (i, j) is both in c and c'. Finally, i *weakly prefers* c to c' if it prefers c to c' or it is indifferent between them. We define the *Stable (Strongly Stable) Exchange Problem* as the problem of finding in G a vertex-disjoint packing of directed cycles with length at most k that corresponds to a *stable (strongly stable) matching*. The definitions of stable and strongly stable matchings [2, 5] are provided below.

Definition 1. A *blocking cycle* $c \notin \mathcal{M}$ is a cycle such that every vertex i in $V(c)$ is either unmatched in $\mathcal{M}$ or prefers c to c', where $c' \in \mathcal{C}(i) \cap \mathcal{M}$. A matching $\mathcal{M}$ is called *stable* if there is no blocking cycle $c \notin \mathcal{M}$.

Definition 2. A *weakly blocking cycle* is a cycle $c \notin \mathcal{M}$ such that for every $i \in V(c)$, i is either unmatched in $\mathcal{M}$ or weakly prefers c to c', where $c' \in \mathcal{C}(i) \cap \mathcal{M}$, with strict preference for at least one vertex. A matching $\mathcal{M}$ is called *strongly stable* if there is no weakly blocking cycle $c \notin \mathcal{M}$.

2 Integer Programming Formulations

The Stable Exchange Problem can be seen as a optimization problem. In what follows we propose three integer programming formulations for it.

2.1 Stable Cycle Formulation

For each pair (i, c), $i \in V$, $c \in \mathcal{C}(i)$ we define two sets of cycles: $B_{i,c} = \{\bar{c} \in \mathcal{C}(i), \bar{c} \neq c : \bar{c} \preceq_i c\}$, which is the set of cycles that are different from c and better or equally preferable for i than c, and $S_{i,c} = \{\bar{c} \in \mathcal{C}(i) : \bar{c} \prec_i c\}$, which is the set of cycles that are strictly better for vertex i than cycle c. Consider vector $x = (x_1, \ldots, x_{|\mathcal{C}|})$ of variables such that $x_c = 1$ if all arcs in $A(c)$ are in $\mathcal{M}$, 0 otherwise. The following set of constraints will define a *stable* matching $\mathcal{M}$:

$$\sum_{c:i \in V(c)} x_c \leq 1 \qquad \forall i \in V \tag{1}$$

$$x_c + \sum_{s \in \bigcup_{i \in V(c)} B(i,c)} x_s \geq 1, \qquad \forall c \in \mathcal{C}, \tag{2}$$

$$x_c \in \{0, 1\} \qquad \forall c \in \mathcal{C}, \tag{3}$$

Constraints (1) guarantee that $\mathcal{M}$ is a set of disjoint cycles. Constraints (2) mean that either $c \in \mathcal{M}$, or, for some vertex $i \in V(c)$, there exists a cycle $c' \in B(i, c)$ such that $i \in V(c')$ and $c' \preceq_i c$. For a *strongly stable* matching, constraints (2) are replaced by:

$$x_c + \sum_{s \in \bigcup_{i \in V(c)} S(i,c)} x_s \geq 1, \forall c \in \mathcal{C}, \tag{4}$$

Constraints (4) guarantee that either c is in the matching, or otherwise one of its vertices is matched in a cycle strictly better than c.

The objective function considered maximizes the maximum number of cycles in $\mathcal{M}$ and is described as follows:

$$F(x) = \sum_{c:c \in \mathcal{C}} |c| \cdot x_c. \tag{5}$$

2.2 Stable Edge Formulation

To define the stable edge formulation, we depart from the edge formulation in [1], where $y_{i,j}$ is a binary variable denoting whether arc (i, j) is included in the solution, or not. A feasible solution with cycles of length at most k can be formalized as follows:

$$\sum_{j:(j,i) \in A} y_{j,i} - \sum_{j:(i,j) \in A} y_{i,j} = 0 \qquad \forall i \in V \tag{6}$$

$$\sum_{j:(i,j) \in A} y_{i,j} \leq 1 \qquad \forall i \in V \tag{7}$$

$$\sum_{(i,j) \in A(p)} y_{i,j} \leq k - 1 \qquad \forall p \in \mathcal{P}. \tag{8}$$

where $\mathcal{P}$ is a set of all non-cyclic paths p in G with k arcs, and $A(p)$ is the set of arcs of G in p. Note that sub-cycles with more than k arcs are removed from the set of feasible solutions by constraints (8). To achieve stability, according to definition 1, we introduce the following set of constraints:

$$\sum_{(i,j)\in A(c)} \left[y_{i,j} + \sum_{r:r<_i j} y_{i,r} \right] \geq 1, \qquad \forall c \in \mathcal{C}. \tag{9}$$

Strong stability can be achieved by replacing inequalities (9) by the following set of constraints:

$$|c| \cdot \left[\sum_{(i,j)\in A(c)} \sum_{r:r<_i j} y_{i,r} \right] + \sum_{(i,j)\in A(c)} y_{i,j} \geq |c|, \qquad \forall c \in \mathcal{C}. \tag{10}$$

The inequality is satisfied for cycle c by the first term if there is an agent strictly preferring her matching in the solution to what she would receive in c. The second term ensures that a cycle already in the solution cannot be a blocking cycle.

Since the sum of all binary variables $y_{i,j}$ is equal to $|\mathcal{M}|$, the objective function can be written as:

$$F(y) = \sum_{(i,j)\in A} y_{i,j}. \tag{11}$$

Note that, if the feasibility constraints from (6) to (8) and the stability constraints (9) or strong stability constrains (10) are satisfied, we obtain the maximum number of cycles in $\mathcal{M}$ by maximizing $F(y)$ in (11).

2.3 Stable Cycle-Edge Formulation

In the stable (strongly stable) cycle-edge formulation, we use the integer variables of the two formulations above in a consistent way. That is, for every cycle $c \in \mathcal{C}$, we require that $x_c = 1$ if and only if $y_{i,j} = 1$ for every $(i,j) \in A(c)$. This correspondence can be achieved by the basic feasibility cycle-constraints (1) and edge-constraints (6), and by adding the following three sets of inequalities:

$$|c| \cdot x_c \leq \sum_{(i,j)\in A(c)} y_{i,j}, \forall c \in \mathcal{C}, \tag{12}$$

$$\sum_{(i,j)\in A(c)} y_{i,j} - |c| + 1 \leq x_c, \forall c \in \mathcal{C}, \tag{13}$$

$$\sum_{j:(i,j)\in A} y_{i,j} \leq \sum_{c:i\in V(c)} x_c, \forall i \in V \tag{14}$$

Stability and strong stability are assured by constraints (9) and (10), respectively. Both (5) and (11) can be used as objective functions.

Table 1. Stable exchange problem formulations: stable cycle formulation (SCF), stable edge formulation (SEF) and stable cycle-edge formulation (SCEF).

Instances					Formulations														
					SCF					SEF					SCEF				
n	|A|	$\|\mathcal{C}\|$	$\|\mathcal{P}\|$	k	Rows	Columns	Non-zeros	Loading time (s)	Solver time (s)	Rows	Columns	Non-zeros	Loading time (s)	Solver time (s)	Rows	Columns	Non-zeros	Loading time (s)	Solver time (s)
30	165	37	3,584	3	57	37	550	0.00	0.00	3,681	165	11,617	0.0274	0.03	189	202	1,295	0.00	0.00
		153	17,477	4	177	153	14,016	0.01	0.02	17,690	165	72,772	0.1509	0.15	541	318	5,724	0.01	0.01
		269	73,636	5	294	269	51,515	0.04	0.07	73,965	165	369,782	0.7135	0.69	890	434	10,913	0.01	0.02
50	617	584	82,009	3	632	584	88,616	0.05	0.14	82,693	617	265,292	0.60	1.16	1,900	1,201	27,089	0.03	0.06
		5,236	951,322	4	5,284	5,236	10,188,648	5.80	126.70	956,658	617	4,028,087	7.25	49.64	15,856	5,853	317,803	0.23	1.56
		38,591	11,004,062	5	38,639	38,591	794,566,412	525.10	**n.m.**	11,042,753	617	56,920,039	89.02	926.14	115,921	39,208	2,852,329	1.81	24.27
70	1135	611	174,480	3	662	611	80,809	0.04	0.19	175,231	1,135	548,667	1.31	5.16	2,019	1,746	33,321	0.05	0.13
		6,700	2,135,151	4	6,753	6,700	14,035,100	7.81	150.48	2,141,991	1,135	8,876,487	15.88	191.83	20,288	7,835	458,502	0.36	5.70
		48,762	26,135,720	5	48,815	48,762	1,092,827,519	721.96	**n.m.**	26,184,622	1,135	133,510,623	229.04	2061.98	146,474	49,897	4,081,818	2.82	60.31
90	2063	3,214	884,802	3	3,298	3,214	1,846,921	1.04	13.92	888,196	2,063	2,829,076	5.91	133.75	9,904	5,277	218,618	0.21	0.94
		49,386	18,407,917	4	49,471	49,386	687,653,906	406.07	**n.m.**	18,457,483	2,063	77,174,437	141.18	1414.87	148,421	51,449	4,440,627	3.46	51.14
		710,726	382,999,769	5	**n.m.**	**n.m.**	**n.m.**	**n.m.**	**n.m.**	**n.m.**	**n.m.**	**n.m.**	**n.m.**	**n.m.**	2,132,441	712,789	78,912,742	52.86	1061.07

Table 2. Strongly stable exchange problem formulations: strongly stable cycle formulation (SSCF), strongly stable edge formulation (SSEF) and strongly stable cycle-edge formulation (SSCEF).

Instances					Formulations														
					SSCF					SSEF					SSCEF				
n	|A|	$\|\mathcal{C}\|$	$\|\mathcal{P}\|$	k	Rows	Columns	Non-zeros	Loading time (s)	Solver time (s)	Rows	Columns	Non-zeros	Loading time (s)	Solver time (s)	Rows	Columns	Non-zeros	Loading time (s)	Solver time (s)
30	165	37	3,584	3	57	37	490	0.00	0.00	3,681	165	11,617	0.02	0.00	189	202	1,295	0.00	0.00
		153	17,477	4	177	153	11,181	0.01	0.00	17,690	165	72,772	0.09	0.02	541	318	5,724	0.01	0.00
		269	73,636	5	294	269	40,684	0.02	0.01	73,965	165	369,782	0.41	0.10	890	434	10,913	0.01	0.00
50	617	584	82,009	3	632	584	81,497	0.05	0.02	82,693	617	265,292	0.40	0.09	1,900	1,201	27,089	0.03	0.01
		5,236	951,322	4	5,284	5,236	9,293,007	5.20	3.60	956,658	617	4,028,087	4.58	1.87	15,856	5,853	317,803	0.23	0.12
		38,591	11,004,062	5	38,639	38,591	725,505,674	437.41	385.29	11,042,753	617	56,920,039	56.87	28.57	115,921	39,208	2,852,329	1.85	1.38
70	1135	611	174,480	3	662	611	74,205	0.04	0.02	175,231	1,135	548,667	0.84	0.23	2,019	1,746	33,321	0.05	0.02
		6,700	2,135,151	4	6,753	6,700	12,928,785	7.01	5.09	2,141,991	1,135	8,876,487	10.49	4.45	20,288	7,835	458,502	0.37	0.36
		48,762	26,135,720	5	48,815	48,762	1,001,482,550	610.08	**n.m.**	26,184,622	1,135	133,510,623	134.24	67.56	146,474	49,897	4,081,818	2.81	3.56
90	2063	3,214	884,802	3	3,298	3,214	1,765,893	0.96	0.61	888,196	2,063	2,829,076	3.95	1.61	9,904	5,277	218,618	0.23	0.25
		49,386	18,407,917	4	49,471	49,386	659,470,242	389.51	341.85	18,457,483	2,063	77,174,437	91.75	44.11	148,421	51,449	4,440,627	3.35	5.97
		710,726	382,999,769	5	**n.m.**	**n.m.**	**n.m.**	**n.m.**	**n.m.**	**n.m.**	**n.m.**	**n.m.**	**n.m.**	**n.m.**	2,132,441	712,789	78,912,742	53.14	104.28

3 Computational Experiments

In this section, we compare the proposed formulations in terms of time needed to find a solution, time needed to load the coefficient matrix associated with each formulation (loading time) and the length of that matrix (number of rows, columns and non-zeros elements). We consider four instances from the literature [3], with 30, 50, 70 and 90 vertices (n), and consider that the maximum length of cycles (k) allowed ranges from 3 to 5. We used C++ language and GUROBI library [4], with default options, as integer programming solver. Tests were executed in a computer with 12 cores Intel(R) Xeon(R) CPU X5675/3.07GHz, 50GB of RAM memory, Ubuntu 16.04.3 LTS operation system and g++ version 5.4.0. Preliminary tests on the (Strongly) Stable Cycle–Edge Formulation (SCEF and SSCEF), showed that by using (11) as objective function, the model was more efficient. Therefore, for the two formulations above, we only report results obtained when this objective was considered.

In Tables 1 and 2, $|\mathcal{C}|$ and $|\mathcal{P}|$ are the number of cycles of length at most k and the number of non-cyclic paths with k arcs, respectively. Entries "n.m." indicate that execution was halted due to insufficient memory.

Table 1 shows the experiments results for stable formulations. Notice that for $k = 3$, SCF presents better times then SEF. This fact can be explained by the number of rows and non-zero elements in the coefficient matrix. SEF has more rows because of constraints (8), that are written for all paths in $\mathcal{P}$. However, for $k = 4$ and $k = 5$, the number of non-zero elements in SCF matrices considerably increased, as well as loading times and solver times. This is due to the number of elements in sets $B_{i,c}$ that increases according to k and to the number of arcs and vertices which are common to cycles in $\mathcal{C}$. Table 1 also shows that, for all k, there is a reduction in the number of rows, columns and non-zero elements in SCEF. This happens because, in this formulation, 1) the path constraints (8) are no longer required; 2) since the stability constraints are written in terms of y_{ij}, the number of columns and non-zero elements are reduced. Table 2 shows the corresponding results for strongly stable formulations. The observations made for Table 1 also hold here.

4 Conclusion

In this work, we presented three new integer formulations for modeling k-way stable exchange problems. Computational tests were done with small instances selected from [3]. Results show that the number of rows, columns and non-zero elements of the coefficient matrix associated with each formulation increases the loading time, the solver time and the memory usage with increasing values of k. Furthermore, SCEF and SSCEF outperform the other formulations for all instances, independently of k. These formulations do also request for less memory.

References

1. David J. Abraham, Avrim Blum, and Tuomas Sandholm. Clearing algorithms for barter exchange markets: Enabling nationwide kidney exchanges. In *Proceedings of the 8th ACM Conference on Electronic Commerce*, EC '07, pages 295–304, New York, NY, USA, 2007. ACM.
2. Péter Biró and Eric McDermid. Three-sided stable matchings with cyclic preferences. *Algorithmica*, 58(1):5–18, Sep 2010.
3. Miguel Constantino, Xenia Klimentova, Ana Viana, and Abdur Rais. New insights on integer-programming models for the kidney exchange problem. *European Journal of Operational Research*, 231(1):57 – 68, 2013.
4. LLC Gurobi Optimization. Gurobi optimizer reference manual, 2018.
5. Chien-Chung Huang. Circular stable matching and 3-way kidney transplant. *Algorithmica*, 58(1):137–150, Sep 2010.

Some practical issues related to the implementation of type III sensitivity analysis of LP models

Imre Dimény[1], Tamás Koltai[1]

[1] Budapest University of Technology and Economics, Department of Management and Business Economics, Budapest, Hungary
dimeny.imre@gmail.com

Abstract. Allocation of scarce resources is a typical problem often encountered by managers, and linear programming (LP) is a widely used tool for supporting decision making in this area. Since many of the parameters involved in the models are generally approximations, expectations or forecasts based on statistically available data, managers must deal with the uncertainty of the available data.
Although LP sensitivity analysis provides valuable information to support management decisions, many papers demonstrate erroneous management decisions based on the misinterpretation of sensitivity analysis results. Koltai and Terlaky classified three types of sensitivity information. Most of the commercial LP solvers provide only Type I sensitivity information but from a management standpoint Type III sensitivity information are far more important. Type III sensitivity provides information about the invariance of the rate of change of the objective value function and thus is independent of the optimal solution found and depends only on the problem data.
This paper discusses some practical problems related to the implementation of Type III sensitivity information in practice.

Keywords: Decision support, LP Sensitivity analysis, Type III sensitivity

1 Introduction

Organizations all over the world use business analytics (BA) to gain insight in order to drive business strategy and planning. With the increasing amount of available data larger models are created to support decision making, but managers also must deal with the uncertainty of the input parameters. In this perspective LP models have two valuable properties: the required computation time allows large models to be solved and further valuable insight can be gained about the problem using sensitivity analysis.

Every linear programming problem, referred to as a primal problem, can be converted into a dual problem, which provides an upper bound to the optimal value of the primal problem. The optimal solution of an LP problem provides the optimal allocation of limited resources, while the optimal solution of the dual problem provides information about the marginal change of the objective function of the primal problem (shadow price), if a right-hand-side parameter changes.

Sensitivity analysis provides information about the validity range of the primal and dual optimum. The validity range of the objective function coefficients (OFC) provides a range for each coefficient, within which the primal optimal solution will not change. Validity range of the right-hand-side (RHS) elements provides a range for each right-hand-side element. Within this range the dual optimum will not change.
If the optimal solution of the primal problem (dual degeneracy) or the dual problem (primal degeneracy) is not unique the resulting sensitivity information can be misleading for managers.

There is a wide range of available tools to solve LP problems. Many of these tools use an implementation of the simplex method and provides an optimal solution related sensitivity information. The sensitivity information generated by such solvers are often used by managers to support decisions.

Evans and Baker [3] provided examples to show that under degeneracy the interpretation of sensitivity information calculated by commercial LP solvers can be erroneous and have significant managerial implications. Problems and possible solutions related to LP Sensitivity analysis has an extensive literature since then.
Aucamp and Steinberg [2] demonstrated that shadow prices are not necessarily equal to dual variables except in the case when the primal problem is nondegenerate and suggested an alternative definition for the shadow price. Akgül [2] differentiated between the positive and negative shadow prices. Gal [4] made an extensive survey on the managerial interpretation problem of shadow prices. Many papers demonstrate erroneous management decisions based on the misinterpretation of sensitivity analysis results [6] [10].
Koltai and Terlaky [7] classified three types of sensitivity information. In non-degenerate cases the three types of sensitivities are identical, but in degenerate cases different sensitivity information could be provided by LP solvers. Type I sensitivity determines those values of some model parameters for which a given optimal basis remains optimal. Most of the commercial LP solvers provide only Type I sensitivity information but from a management standpoint Type III sensitivity information are far more important. Type III sensitivity provides information about the invariance of the rate of change of the objective value function. This information is independent of the optimal solution found and depends only on the problem data.
A practical approach to calculate Type III sensitivity information was presented by Koltai and Tatay [7]. The suggested approach uses additional LP's to calculate the related sensitivity ranges.

To create a tool which makes easily accessible this valuable information for management decision making some further practical issues need to be addressed. The first column of Table 1 contains the standard form of a primal linear programming problem, while the last column contains the standard form of the dual linear programming problem [5]. The second column contains a perturbed primal problem, where δ can take both positive and negative values.

Table 1. Primal, perturbed primal and dual linear programing problems

primal problem		perturbed primal problem		dual problem	
$\mathbf{Ax} \leq \mathbf{b}$ $\mathbf{x} \geq \mathbf{0}$ $\mathbf{max}(\mathbf{c^T x})$	(1)	$\mathbf{Ax} \leq \mathbf{b} + \delta \boldsymbol{e_j}$ $\mathbf{x} \geq \mathbf{0}$ $\mathbf{max}(\mathbf{c^T x})$	(2)	$\mathbf{A^T y} \geq \mathbf{c}$ $\mathbf{y} \geq \mathbf{0}$ $\mathbf{min}(\mathbf{b^T y})$	(3)

Table 2 contains the additional LP's required to calculate Type III sensitivity intervals. If λ=1 maximal increase will be calculated, while calculating the maximal decrease requires the λ parameter to be set to -1. γ_i and ξ_j are the decision variables used to calculate maximal decrease/increase allowed for the c_i OFC and b_j RHS parameter.

Table 2. Additional LP problems for sensitivity analysis

Sensitivity analysis of OFC parameters		Sensitivity analysis of RHS parameters	
$\mathbf{A^T y} \geq \mathbf{c} + \lambda \gamma_i \boldsymbol{e_i}$ $\mathbf{b^T y} = OF^* + \lambda\, \gamma_i \boldsymbol{x_i^*}$ $\gamma_i \geq \mathbf{0}$ $\mathbf{max}(\gamma_i)$	(4)	$\mathbf{Ax} \leq \mathbf{b} + \delta \boldsymbol{e_j} + \lambda\, \xi_j \boldsymbol{e_j}$ $\mathbf{c^T x} = OF^* + \lambda\, \xi_j y_i^*$ $\xi_j \geq \mathbf{0}$ $\mathbf{max}(\xi_j)$	(5)

The remainder of this paper is organized as follows. First the question related to the size of the perturbation used to calculate Type III sensitivity ranges of RHS elements is discussed. Next, a possibility to decrease the calculation time of the validity ranges is proposed. Finally, a practical tool for implementation is presented.

2 Perturbation size

Under degeneracy the effect of increase and the effect of decrease of the RHS elements can be different. Consequently, information about the marginal increase and the marginal decrease of each RHS parameter are necessary. To calculate the linearity intervals related to the increase of an RHS element a δ >0 perturbation is used while a δ<0 perturbation is used to calculate sensitivity range related to the decrease of an RHS parameter in LP problem (5). If the value of δ is set overly small numerical error could occur, while setting δ overly large could result an erroneous validity range.

Figure 1 shows the objective value function related to an RHS parameter. If the perturbation (δ_2) is larger than the validity range related to the shadow price at the original value of the RHS parameter (value of the RHS parameter set to b_2), the calculated Type III interval is erroneous. The root cause of the problem is that the original RHS parameter value (b) is outside the validity range of the shadow price related to the perturbed primal LP problem.

This problem could be considered just theoretical since in practice, for a given LP problem, decision makers could set perturbation sizes which are small enough to consider smaller changes irrelevant.

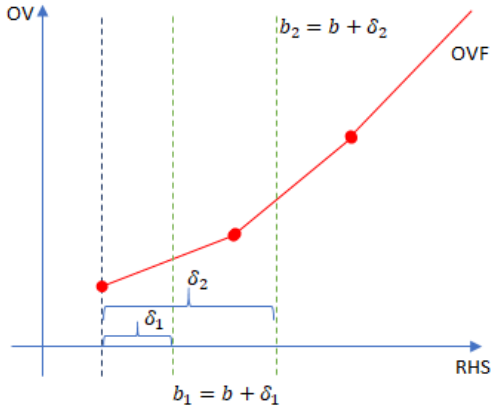


Fig. 1. Objective value function related to the *b* right-hand-side element.

To create a general solution for calculating Type III sensitivity information the size of the perturbation must be set automatically and based only on the LP parameters. An initial δ value could be set by defining an arbitrary function on the parameters of the LP problem. Then, to prevent the problem of setting an overly large perturbation value, the Type I validity range of the perturbed LP (2) must be calculated. If the original value of the RHS parameter is inside the Type I interval of the perturbed LP problem, then no further steps are required. Figure 2 presents the situation when the original value of the RHS parameter is outside the Type I interval of the perturbed problem

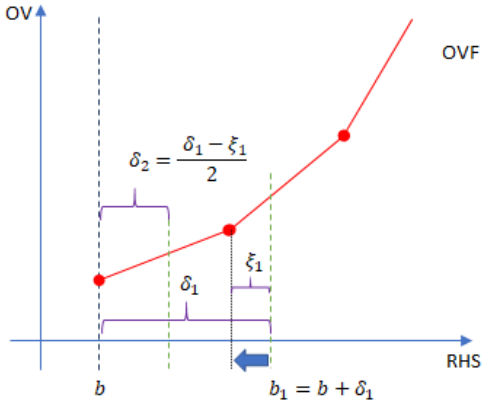


Fig. 2. Calculation of proper perturbation size.

In this case a new perturbation size must be calculated using the following formula:

$$\delta_2 = \frac{\delta_1 - \xi_1}{2}, \tag{6}$$

where δ_2 is the size of the new perturbation, δ_1 is the size of the previous perturbation and ξ_1 is the difference between the original perturbation and the edge of the left validity interval of the perturbed dual LP. This step is repeated until the original RHS parameter value is inside the validity interval of the shadow price of the perturbed LP.

3 Decreasing the calculation time

To calculate the sensitivity information for all RHS and OFC parameters in case of *I* variables and *J* constraint at least 2*I*+6*J* additional LP problems must be solved. The number of additional LP problems to be solved could increase if for some RHS parameter the initial perturbation level was set too high. One way to decrease the required computation time is to calculate Type III sensitivity information just for those parameters that are important from a managerial stand point.

Calculation time can be further decreased by taking advantage of the possibility to initialize solver runs. The vector $\mathbf{x}' = (\mathbf{x}^*, 0)$ is a feasible solution for the additional LP problems related to the calculation of the Type III sensitivity ranges of RHS parameters (5), where $\mathbf{x}^*$ is the optimal solution of the perturbed primal LP (2). The vector $\mathbf{y}' = (\mathbf{y}^*, 0)$ is a feasible solution for the additional LP problems related to the calculation of the Type III sensitivity ranges of OFC parameters (4), where $\mathbf{y}^*$ is the optimal solution of the dual LP problem (3). By instructing the solver to initialize the solver run using this information the calculation of the additional LP problems can be accelerated.

4 A practical tool

To support decision makers with Type III sensitivity information, a tool is required which can solve the numerous LP's presented in Table 1 and Table 2, and has good algorithmic capabilities to connect the models and collect the resulting information. Such a tool is provided by the AIMMS Prescriptive Analytics Platform, which is often used for solving commercial optimization problems in a wide range of industries including retail, consumer products, healthcare, oil and chemicals, steel production and agribusiness [8].

AIMMS Prescriptive Analytics Platform is a tool for those with an Operations Research or Analytics background and offers a straightforward mathematical modelling environment and a wide range of available solvers. AIMMS also features an advanced graphical user interface editor which allows the creation of optimization application to individuals without a technical or analytics background. AIMMS own structural language allows the creation of procedures to connect the multiple models required to calculate Type III ranges for all the parameters. The list of available solvers also includes simplex method-based solvers such as CPLEX which can be used to calculate Type I sensitivity

information required to check the perturbation size. CPLEX can also be instructed to use an initial solution indicated in section 3 and this way the calculation time can be significantly decreased.

With the use of the build-in user interface editor, a user-friendly interface can be created for decision makers to choose the relevant parameters of the model.

5 Conclusion

When sensitivity information of LP parameters is important for management decision making, then Type III sensitivity information must be calculated. In this case the erroneous conclusions triggered by degenerate solutions can be avoided. The mathematical models for generating Type III information are well known. The practical implementation of the calculation, however requires the solution of some numerical and computational problems.

In this paper the setting of proper perturbation size and the possibilities to decrease calculation time was discussed.

The findings presented in this paper are important steps to the development of a practical tool, which can be used by manager when LP models are applied for the allocation of scares resources.

References

1. Akgül, M.: A note on shadow prices in linear programming. Journal of the Operational Research Society 35(5), 425-431 (1984).
2. Aucamp, D.C., Steinberg, D.I.: The computation of shadow prices in linear programming. Journal of the Operational Research Society 33, 557-565 (1982).
3. Evans, J.R., Baker, N.R.: Degeneracy and the (mis)interpretation of sensitivity analysis in linear programming. Decision Science 13, 348-354 (1982).
4. Gal, T.: Shadow prices and sensitivity analysis in linear programming under degeneracy. OR spectrum 8(2), 59-71 (1986).
5. Hiller, F.S., Lieberman, G.J.: Introduction to Operation Research. McGraw-Hill, New York. (1995)
6. Jansen, B., De Jong, J.J., Roos, C., Terlaky, T.: Sensitivity analysis in linear programming: just be careful! European Journal of Operational Research 101(1), 15-28 (1997)
7. Koltai, T., Tatay, V.: A practical approach to sensitivity analysis in linear programming under degeneracy for management decision making. International Journal of Production Economics 131(1), 392-398 (2011).
8. Koltai, T., Terlaky, T.: The difference between the managerial and mathematical interpretation of sensitivity analysis results in linear programming. International Journal of Production Economics 65(3), 257-274 (2000).
9. Roelofs, M., Bisschop, J: AIMMS The User's Guide. AIMMS B.V (2018).
10. Wagner, H.M., Rubin, D.S.: Shadow prices: Tips and traps for managers and instructors. Interfaces 20, 150-157 (1990).

An efficient heuristic for a complex scheduling problem

György Dósa, Tibor Dulai, and Ágnes Werner-Stark

University of Pannonia, Veszprém, Hungary
dulai.tibor@virt.uni-pannon.hu

Abstract. In this paper a complex scheduling problem is dealt with. Several types of products should be produced with a heterogeneous resource set: some resources have the same operating capabilities, some of them are quite different, while others have similarities regarding their capabilities but they have different parameters (like operation time). Setup time is also considered in the model. The production of the products follows different workflows, allowing also assembly lines. The goal is to produce all the products in minimum time.
Because of the complexity of the problem exact solvers require too much time to solve the problem. A compound heuristic algorithm is introduced that finds a near-optimal solution very fast. The method follows the idea of "Divide et Impera": the basic problem is divided into smaller subproblems that are easier to solve. During the solution process simplifications are applied, and first a preemptive version of the simplified problem is solved. Then a rounding procedure results in a non-preemptive solution. Finally, for improving the solution, several kinds of local search are applied.
The efficiency of the heuristics is demonstrated on multiple problem classes. A multitude of instances were solved and analyzed according to different aspects.

Keywords: scheduling · heuristics · problem splitting.

1 Introduction

This paper introduces a special scheduling problem where predetermined job sequences have to be scheduled on unrelated machines for obtaining different products. Some papers with related topic are mentioned below. Precedence constraints are applied in [1], too, but on a single machine. [2] contains a review of various metaheuristic methods for solving different kinds of models with huge size. A classical work in this topic is [3]. Generally in these models there is only one resource, but the model can be extended by different constraints.
The considered problem can be approached from the theory of scheduling. In the three-field-notation, the problem can be denoted as $R_m|prec|C_{max}$, i.e. given m unrelated machines, there are precedence constraints between the jobs, and the makespan has to be minimized. In the model presented in this paper the length

of the chains in the precedence graph is short (contains only chains of length at most 3). Other related works dealing with unrelated machines are [4, 5], none of them considers the same model that is treated here. There are only a few papers considering unrelated machines in the presence of precedence constraints, except under special conditions, like the one where the precedence constraints are in fact chains. But it is worth to note that in our model some components are assembled, it means that there are points in the precedence graph with more than one predecessor. Among the few exceptions, [6] provides a heuristic algorithm for problem $R_m|prec|C_{max}$, and gives computational experiments.

2 Presentation of the problem

For the presentation of the complexity of the problem that is intended to solve a small number of jobs (also called tasks) and resources is enough, this way the presentation is also easily understandable. In this model there are two types of workflows, they differ only in one task as it can be seen in Figure 1. Both types of workflows have to be executed several ($n1$ and $n2$, respectably) times as the prescribed quantity of products defines. The model of the problem includes

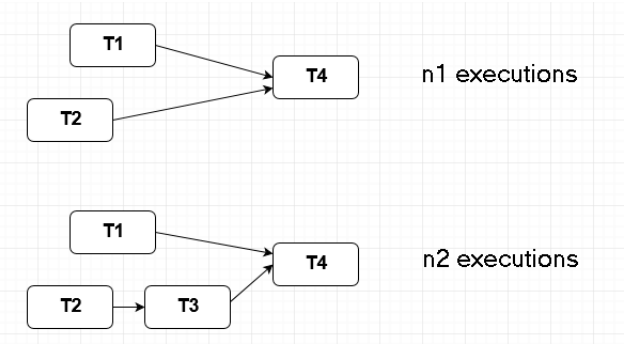


Fig. 1. Workflows of the applied model.

also the description of resources and their capabilities. Here 8 resources are applied, some of them are of similar types, some of them share some capabilities, while others differ totally from each other as table 1 shows. Important to note that $p(R_3, T_2) < p(R_6, T_2)$ and $p(R_3, T_3) > p(R_6, T_3)$. It means that resource R_6 is better (quicker) in performing tasks with type T_3 than resources R_3-R_5, but worse in executing tasks with type T_2. Moreover, there is a setup time for resources R_3, R_4, R_5 and R_6 if they switch between the task types they are able to carry out ($setup(T_2, T_3)$ from T_2 to T_3 and $setup(T_3, T_2)$ from T_3 to T_2).
The goal is to find the schedule with minimal makespan.

Table 1. Resource capabilities of the model /$p(R,T)$ denotes processing time of task T by resource R; '-' means that a resource is not able to perform the task/.

	T_1	T_2	T_3	T_4
R_1	$p(R_1,T_1)$	-	-	-
R_2	$p(R_1,T_1)$	-	-	-
R_3	-	$p(R_3,T_2)$	$p(R_3,T_3)$	-
R_4	-	$p(R_3,T_2)$	$p(R_3,T_3)$	-
R_5	-	$p(R_3,T_2)$	$p(R_3,T_3)$	-
R_6	-	$p(R_6,T_2)$	$p(R_6,T_3)$	-
R_7	-	-	-	$p(R_7,T_4)$
R_8	-	-	-	$p(R_7,T_4)$

3 The proposed solution method: "Divide et Impera"

The main idea was to separate the problem into sub-problems that can be solved one after one easier than the whole problem. During this separation parts have to be handled carefully that are not independent from each other in the resource allocation process. Since resources R_3-R_6 are able to substitute each other just like R_1-R_2 and R_7-R_8, regarding the jobs they can carry out; the problem was divided into 3 sub-problems /rooms/ as Figure 2 shows.

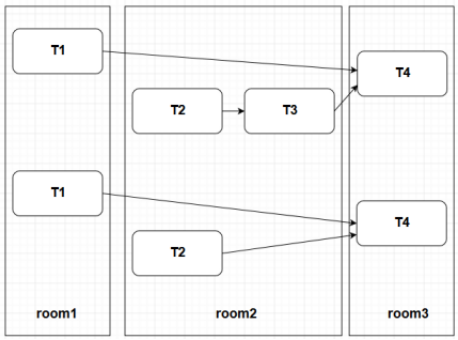


Fig. 2. The separation of the initial problem into sub-problems.

3.1 *Room*$_1$

To find the optimal schedule for $room_1$ is easy. There are 2 identical machines with the same capabilities. Since there is no previous sub-task, the execution of a task can begin as soon as there is a free machine. So, here a trivial greedy algorithm that can be applied to solve the first sub-task.

3.2 *Room*$_2$

To solve the second sub-problem is challenging. There are 4 machines, the capability of one of the four machines differs from that of the others, the remaining 3 machines have identical parameters. All the 4 machines are able to execute both task types of the sub-problem. A heuristic algorithm is applied for this room.
When a machine executes different tasks, the change from one type of job to another requires some time (the setup time), the number of switches between the different types of tasks (in this case T_2 and T_3) of the resources should be minimized: it should happen maximum once for each resource. Each resource type has to start its operation on the job type it is able to handle with smaller processing time. Based on the model of the problem in case of resources R_3-R_5 it is a task with type T_2, while for resource R_6 it is a task with type T_3.
It means that for resources R_3-R_5 the task sequence should be like T_2, T_2, ..., T_2, T_3, ..., T_3 and for resource R_6 that is T_3, T_3, ..., T_3, T_2, ..., T_2. It also has to be noted that if there is an R_3-R_5 resource that executes task with type T_3 and R_6 also has T_2 task in its queue then these tasks have to be swapped for reaching a better solution. It results, that the task queue of resources is:

1. T_2, T_2, ..., T_2, T_3, ..., T_3 for R_3-R_5 resources and T_3, T_3, ..., T_3 for R_6, or
2. T_2, T_2, ..., T_2 for R_3-R_5 resources and T_3, T_3, ..., T_3, T_2, ..., T_2 for R_6.

Suppose that the first case happens (in the opposite case the solution is very similar). Then the job sequences (and the definition of x and y) are as it is illustrated in Figure 3. As the first step a preemptive solution should be found, where

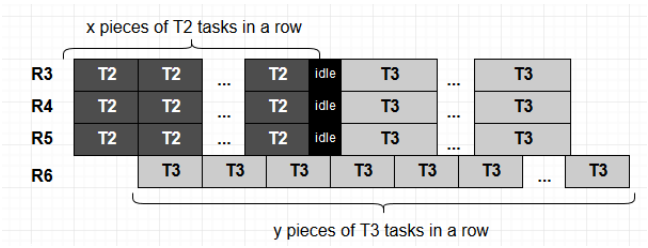


Fig. 3. A half-ready job sequence in $Room_2$ resulted by some steps of the heuristics.

all resources finish their job sequences at the same time. So, for determining the values of x and y the following equation system has to be solved:

$$p(R_3,T_3)*y = ((n1+n2)*p(R_2,T_2)+setup(T_2,T_3)*3+(n1-y)*p(R_3,T_3))/3 \quad (1)$$

$$x = n1/3 \quad (2)$$

Since x and y are not surely integers, a rounding process has to be done. Both numbers are rounded into the integer that is less or equal to them. Then the

remaining tasks are concatenated to the job sequences in the following way: the remaining T_3 jobs are added to the queue of resource R_6 and the remaining T_2 tasks are concatenated to the job sequence of resources R_3-R_5, starting from R_5 backwards to R_3.
In the final step local search is applied to improve the obtained solution: The makespan of the solution is calculated, then a random job is selected that is assigned to a different machine. If the makespan of the resulted solution is better, then it is kept, otherwise the change is discarded.

3.3 *Room*$_3$

The $3rd$ sub-problem is similar to the first sub-problem: there are two machines with identical capabilities. However, for starting the execution of a task not only a free machine is required but ready raw material pair from the previous rooms is also needed. Nevertheless, finding the optimal solution for this sub-problem is easy, the classical list scheduling is applied for this room.

4 Results

The proposed heuristic was tested both on deterministic data and on stochastic data, too. For the deterministic case three different problem classes were created that differ in the size of the problem: in the small sized problem $n1 = 20, n2 = 15$, in the medium sized problem $n1 = 50, n2 = 50$ and in the large sized problem $n1 = 150, n2 = 100$. For the stochastic case both numbers were selected in such a way that $n1 + n2 = 100$. The details of the parameters (the constant values for the deterministic way, and the interval from were the random values were selected in the stochastic way) are highlighted in Table 2.
In the calculation it turned out that local search phase of the heuristic is im-

Table 2. Parameter values of the test cases /values mean time unit; $p(R, T)$ denotes processing time of task T by resource R/.

	deterministic case	**stochastic case**
$p(R_1, T_1)$	6	$[2, 6]$
$p(R_3, T_2)$	6	$[4, 8]$
$p(R_3, T_3)$	10	$[10, 15]$
$p(R_6, T_2)$	10	$[9, 13]$
$p(R_6, T_3)$	6	$[5, 7]$
$p(R_7, T_4)$	4	$[2, 4]$
$setup(T_2, T_3) = setup(T_3, T_2)$	6	$[0, 5]$

portant, it can improve the quality of the result. It was obtained, too, that 10 iterations of local search were enough to find a good solution for the 2^{nd}

sub-problem in all the small, medium and large sized problem cases. The same problems were solved also using CPLEX that helped to judge the quality of the heuristics: in most of the cases the distance of the results to the optimum was less than 5% related to the optimal makespan and in more than 50% of the cases the proposed algorithm resulted in the optimum. Moreover, the heuristic algorithm provided the result quickly: in all cases (even for the large sized problems) the execution time was below 1 second.

5 Conclusion

In this paper a heuristic algorithm has been presented for scheduling a complex problem of unrelated machines. For the solution the original problem was divided into smaller sub-problems. Some of the sub-problems could be solved in a trivial way (greedy algorithm and list scheduling), while one sub-problem required a more complex heuristic. In this heuristic a preemptive solution was created, then that was improved by rounding and local search. Some computation were made with different problem classes (both the problem size and the deterministic/stochastic manner of the problem were varied). The proposed heuristic produces the result fast.
A challenge for the future is to examine more general cases (with arbitrary number of resources and with different process structure).
One interesting direction of further research could be the investigation of the separability of complex workflows into smaller sub-problems.

Acknowledgement

The Authors acknowledge the financial support of Széchenyi 2020 under the EFOP-3.6.1-16-2016-00015.

References

1. Correa, J.R., Schulz, A.S.: Single-Machine Scheduling with Precedence Constraints. Mathematics of Operations Research **30**(4), 1005–1021 (2005)
2. Jans, R., Degraeve, Z.: Meta-heuristics for dynamic lot sizing: A review and comparison of solution approaches. European Journal of Operational Research **177**, 1855–1875 (2007)
3. Wagner, H.M., Whitin, T.M.: Dynamic Version of the Economic Lot Size Model. Management Science **5**(1), 89–96 (1958)
4. Herrmann, J., Proth, J.M., Sauer, N.: Heuristics for unrelated machine scheduling with precedence constraints. European Journal of Operational Research **102**(3), 528–537 (1997)
5. Rocha, P.L., Ravetti, M.G., Mateus, G.R., Pardalos, P.M.: Exact algorithms for a scheduling problem with unrelated parallel machines and sequence and machine-dependent setup times. Computers and Operations Research **35**, 1250–1264 (2008)
6. Liu, C., Yang, S.: A heuristic serial schedule algorithm for unrelated parallel machine scheduling with precedence constraints. Journal of Software **6**(6), 1146–1153 (2011)

An exponential lifetime model: Stochastic order relations and a copula approach for modelling systemic risk

Sándor Guzmics

Univ. of Vienna, Department of Statistics and Operations Research, Vienna, Austria
sandor.guzmics@univie.ac.at

Abstract. We reconsider an exponential lifetime model obtained by modifications of exponential distributions. We investigate the copula of the resulting multivariate distribution function, and we provide some results concerning certain stochastic orderings, namely the convex order, the increasing convex order and the Lorenz order. Furthermore, we employ the joint lifetime model for quantifying systemic risk in financial systems.

Keywords: Copulas, Stochastic dominance, Convex order, Lorenz order, Systemic Risk

1 Introduction

We consider a system of n entities and their dependent lifetimes as it has been already presented in Pflug and Guzmics [8], and first has been published (for $n = 2$) by Freund [2]. Our overall motivation is to suggest a model for banking systems, where the default of an institution affects the remaining lifetime of other institutions. The lifetime variables are linked to loss variables, and we define risk functionals as functions of the loss variables in order to quantify systemic risk.

The lifetime model has the following setting: initially we assume individual exponential lifetimes for each entity, and when an institution defaults, the lifetime distributions of all other institutions are modified, as an effect of this default. Regarding the underlying exponential marginal distributions, our model can be understood as a multivariate extension of the exponential distribution, even though the resulting marginals, in general, will not be exponentially distributed any more, but they are mixtures of two exponential distributions.

The paper is organized as follows. In Section 2 we recall the model and its basic properties. In Section 3 we give a brief review about the convex, increasing convex and the Lorenz order, which are going to be relevant for our analysis. In Section 4 we explore some properties of our lifetime model regarding the above mentioned stochastic order relations. In Section 5 we suggest a model for systemic risk based on the lifetime model presented in Section 2.

2 The bivariate model

Pflug and Guzmics [8] have recently suggested a multivariate lifetime model, which for $n = 2$ had been worked out by Freund [2]. Consider two independent exponential lifetime variables, $Y_1 \sim Exp(\lambda_1)$, $Y_2 \sim Exp(\lambda_2)$. Assume that in a certain realization the first institution defaults earlier ($Y_1 < Y_2$). Then the remaining lifetime of the second institution will be continued according to another variable $Z_2 \sim Exp(\lambda_2 + a_2)$, for some fixed $a_2 \geq 0$, where a_2 is called shock parameter. The r.v. Z_2 is independent of both Y_1 and Y_2. We impose an analogous modification, when $Y_2 < Y_1$, by introducing a r.v. $Z_1 \sim Exp(\lambda_1 + a_1)$, where $a_1 \geq 0$ is a fixed shock parameter. The resulting, already modified lifetime variables X_1, X_2 are given by

$$\begin{cases} X_1 = Y_1 \cdot \mathbb{1}_{\{Y_1<Y_2\}} + (Y_2 + Z_1) \cdot \mathbb{1}_{\{Y_2<Y_1\}}\,, \\ X_2 = Y_2 \cdot \mathbb{1}_{\{Y_2<Y_1\}} + (Y_1 + Z_2) \cdot \mathbb{1}_{\{Y_1<Y_2\}}\,. \end{cases} \tag{1}$$

The joint cdf of (X_1, X_2) is given by (2). For more details look at [8].

$$H(x,y) = \begin{cases} 1 + \dfrac{\lambda_1}{\lambda_1 - a_2} \cdot e^{-(\lambda_1 - a_2)x} \cdot e^{-(\lambda_2+a_2)y} + \dfrac{a_1}{\lambda_2 - a_1} \cdot e^{-(\lambda_1+\lambda_2)x} - \\ \quad \dfrac{\lambda_2}{\lambda_2 - a_1} \cdot e^{-(\lambda_1+a_1)x} - \dfrac{\lambda_1}{\lambda_1 - a_2} \cdot e^{-(\lambda_2+a_2)y} \quad \text{when} \quad 0 \leq x \leq y\,, \\ 1 + \dfrac{\lambda_2}{\lambda_2 - a_1} \cdot e^{-(\lambda_2 - a_1)y} \cdot e^{-(\lambda_1+a_1)x} + \dfrac{a_2}{\lambda_1 - a_2} \cdot e^{-(\lambda_1+\lambda_2)y} - \\ \quad \dfrac{\lambda_1}{\lambda_1 - a_2} \cdot e^{-(\lambda_2+a_2)y} - \dfrac{\lambda_2}{\lambda_2 - a_1} \cdot e^{-(\lambda_1+a_1)x} \quad \text{when} \quad 0 \leq y \leq x\,. \end{cases} \tag{2}$$

We are interested in exploring the dependence structure of this bivariate distribution, and examine how the dependence structure changes as we let the shock parameters a_1, a_2 vary. To this aim we have considered the copula of (X_1, X_2) for a symmetric parameter setting ($\lambda_1 = \lambda_2 = 1, a_1 = a_2 = a$), and showed that $C_{a'}$ dominates C_a for $0 \leq a \leq a'$ in the upper orthant order, which clearly justifies, that as parameter a gets larger, the lifetime variables X_1, X_2 are more positively dependent. For the details of this analysis look at [8].

In this current paper we will investigate the lifetime variables (X_1, X_2) with respect to three closely related stochastic orders: the convex order, the increasing convex order and the Lorenz order.

3 Convex, increasing convex and Lorenz order

Stochastic order relations have an extensive literature, here we only mention some selected items. In the bibliographies of the listed works one can find numerous other sources. The books by Shaked and Shantikumar [3], [5] are general

comprehensive works; Denuit et al. use the notion of stochastic dominance (of several kinds) for applications in actuarial theory; concerning the topic of our present work (convex order and Lorenz order) the most highlighted references are Scarsini [4], and Kämpke and Radermacher [7].

Throughout this section $\underline{x}, \underline{y} \in \mathbb{R}^n$; $\underline{X} = (X_1, \ldots, X_n)$ and $\underline{Y} = (Y_1, \ldots, Y_n)$ denote n-variate random vectors, Z and W denote random variables.

3.1 The multivariate convex and increasing convex orders

The definition of convex order and increasing convex order is given in a merged form in Definition 1. (We use the term 'increasing function' in the weak sense.)

Definition 1. *A random vector $\underline{X}$ is said to be smaller than the random vector $\underline{Y}$ in the (increasing) convex order, in notation $\underline{X} \preceq_{CVX} Y$ resp. $\underline{X} \preceq_{ICVX} Y$, if for all (increasing) convex functions $g : \mathbb{R}^n \to \mathbb{R}$*

$$\mathbb{E}(g(\underline{X})) \leq \mathbb{E}(g(\underline{Y})). \tag{3}$$

Remark 1. It is easy to see that if two random vectors are ordered in the (increasing) convex order, then each of their components are also ordered in the (increasing) convex order. The converse does not hold, look at Example 1.

Remark 2. It is also easy to verify that both the convex order and the increasing convex order are partial order relations.

3.2 The Lorenz order and its relation to the convex order

For the sake of simplicity, in the rest of the paper we restrict our discussion to random variables which are non-negative and not almost surely 0. We use the same assumptions for the components of the appearing random vectors, too.

The notion of Lorenz curve [1] has been originally established in economics to compare the households by their income, wealth, etc. distributions.

Definition 2. *For a r.v. Z with cdf F the Lorenz curve is defined as $L : [0,1] \to [0,1], L_Z(u) = \frac{\int_0^u F^{-1}(s)ds}{\mathbb{E}(Z)}$, where F^{-1} is the generalized inverse of F.*

Remark 3. The Lorenz curve of a non-negative (and not almost surely 0) r.v. is an increasing convex function on $[0,1]$ and it can be seen as a way of describing unevenness. For an egalatarian distribution (where each household has exactly the same income, the Lorenz curve is given by $L(u) = u$. The larger the differences in the incomes are, the more the Lorenz curve deviates from $L(u) = u$. Notice that the Lorenz curve is invariant under positive scaling, i.e. if Z is a non-negative r.v. and $\lambda > 0$, then Z and $\lambda \cdot Z$ have the same Lorenz curves.

There is a univariate stochastic order relation which is obtained by comparison of Lorenz curves, that is the Lorenz order. For a reference look at [7].

Definition 3. *Let W and Z be non-negative random variables with $\mathbb{E}(W) > 0$, $\mathbb{E}(Z) > 0$. W is said to be dominated by Z in the Lorenz order, in notation $W \preceq_L Z$, if $L_W(u) \geq L_Z(u)$ for all $u \in [0,1]$.*

Remark 4. Notice that all (positive) egalatarian distributions are dominated by any other distributions in the Lorenz order. However, there is no such distribution that would dominate all other distributions in the Lorenz order.

We cite an important equivalence relation between the univariate convex order and the Lorenz order, which is in the core of our upcoming investigations in Section 4. The statement can be found in [7], Theorem 3.

Proposition 1. *Let W and Z be non-negative r.v.-s with $\mathbb{E}(W) = \mathbb{E}(Z)$. Then $W \preceq_L Z$ if and only if $W \preceq_{CVX} Z$.*

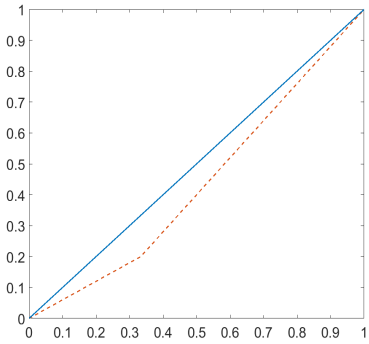
Finally, we give an example as it has been announced in Remark 1.

Example 1. Let us consider the bivariate vectors $\underline{X}$ and $\underline{Y}$ with the joint distributions given in following tables.

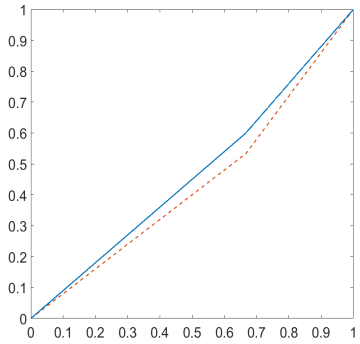
X_1 \ X_2	9/10	12/10
1	2/3	1/3

Y_1 \ Y_2	4/5	7/5
3/5	0	1/3
6/5	2/3	0

One can show that $X_i \preceq_{CVX} Y_i$ for $i = 1, 2$, but $\underline{X} \not\preceq_{CVX} \underline{Y}$. The Lorenz curves of X_i, Y_i $(i = 1, 2)$ are plotted in Figure 1. According to Proposition 1 the pointwise orderedness of the corresponding Lorenz curves means that $X_i \preceq_{CVX} Y_i$ for $i = 1, 2$. The bivariate convex functions $g(z_1, z_2) = z_1^2 + z_2^2$ and $h(z_1, z_2) = \max\{0, -z_1 - z_2 + 2\}$ show that $\underline{X}$ and $\underline{Y}$ are not comparable in the (multivariate) convex order.



(a) $L_{X_1}(u) \geq L_{Y_1(u)}$ for all $u \in [0,1]$, therefore $X_1 \preceq_L Y_1$ and $X_1 \preceq_{CVX} Y_1$



(b) $L_{X_2}(u) \geq L_{Y_2(u)}$ for all $u \in [0,1]$, therefore $X_2 \preceq_L Y_2$ and $X_2 \preceq_{CVX} Y_2$

Fig. 1: Componentwise Lorenz dominance illustrated by Lorenz curves

4 Convex and Lorenz order relations in our model

We consider the model (2) for a symmetric parameter setting, i.e. when $\lambda_1 = \lambda_2 = 1, a_1 = a_2 = a$. (Although the formula (2) is not valid for $\lambda = a = 1$, this case does not need to be excluded, neither, as [8] explains in details.)

Now we are ready to examine the model w.r.t the following three order relations. We denote the vector of the lifetime variables by $\underline{X}^a = (X_1^a, X_2^a)$.

Proposition 2. *Let $0 \leq a < a'$. Then $\underline{X}^a$ and $\underline{X}^{a'}$ are not ordered in the multivariate convex order.*

Proof. Pflug and Guzmics [8] as well as Freund [2] showed that $\mathbb{E}_a(X_1) = \mathbb{E}_a(X_2) = \frac{1}{2} \cdot \frac{a+2}{a+1}$. The necessary condition for the convex order i.e. $\mathbb{E}(\underline{X}^a) = \mathbb{E}(\underline{X}^{a'})$, does not hold. (Cf. Remark 2.)

Proposition 3. *Let $0 \leq a < a'$. Then X_i^a and $X_i^{a'}$ $(i = 1, 2)$ are not ordered in the Lorenz order.*

Sketch of the Proof. The Lorenz curves should have pointwise orderedness in order that the Lorenz order relation holds, which is not case, as the two panels of Figure 2 show.

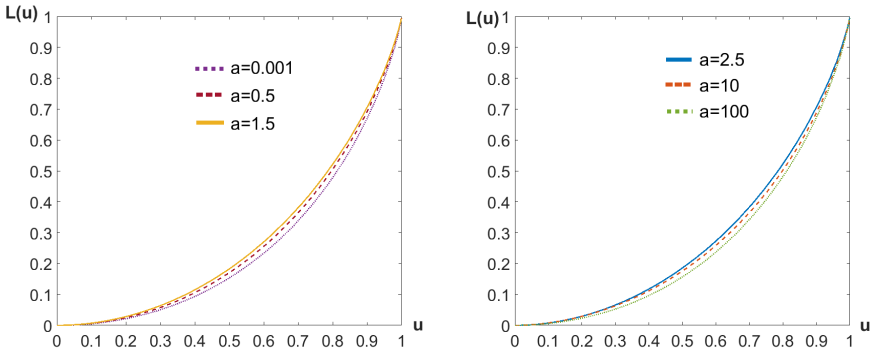


Fig. 2: Lorenz curves of X_i^a for different a values. For $0 \leq a < a' \leq a_0$: $X_i^{a'} \preceq_L X_i^a$ (left). For $a_0 \leq a < a'$: $X_i^a \preceq_L X_i^{a'}$ (right).

Remark 5. There exists a certain turning point a_0, such that for $0 \leq a < a' \leq a_0$ the relation $X_i^{a'} \preceq_L X_i^a$ $(i = 1, 2)$ holds, and for $a_0 \leq a < a'$ the relation $X_i^a \preceq_L X_i^{a'}$ $(i = 1, 2)$ holds. According to our investigations up to now, $a_0 \approx 2.4$.

Proposition 4. *Let $0 \leq a < a'$. Then X_i^a and $X_i^{a'}$ $(i = 1, 2)$ are ordered in the increasing convex order: $X_i^{a'} \preceq_{ICVX} X_i^a$ $(i = 1, 2.)$*

Proof. According to Denuit et al. [6], Proposition 3.4.6 it is enough to examine, whether $\mathbb{E}((X_i^{a'} - t)_+) \leq \mathbb{E}((X_i^a - t)_+)$ for all $t \in \mathbb{R}$. One sees by elementary computation that $a \mapsto \mathbb{E}((X_i^a - t)_+) = \frac{e^{-(1+a)t}}{1-a^2} - \frac{a}{2(1-a)} \cdot e^{-2t}$ is a decreasing function of a for each fixed t, which proves the statement.

5 A possible model for systemic risk

In this section we relate our lifetime variables to loss variables via a simple idea: the longer the lifetime is, the less the losses are. Look at Example 2 as an illustration. For more examples see [8]. Such a relation enables us to translate the notion of 'financial risk' into the 'volume of losses', which can be quantified.

Definition 4. Let us consider a risk functional $R : \mathcal{S} \to \mathbb{R}$, where $\mathcal{S}$ is a suitable space of real-valued random variables. Let $(L_1, \ldots, L_n)$ be the vector of losses, which are defined through the lifetime variables $(X_1, \ldots, X_n)$, and let C the copula of $(X_1, \ldots, X_n)$. Then the systemic risk $\mathcal{R}$ is defined as

$$\mathcal{R}(C, R; \mathbf{L}) = R\left(\bigoplus\nolimits_C L_i\right) - R\left(\bigoplus\nolimits_{\Pi} L_i\right), \tag{4}$$

where $\mathbf{L}$ defines the loss variables in terms of the lifetime variables, and $\bigoplus_C$ denotes the sum of L_i-s, when X_i-s are coupled by copula C.

Example 2. Let $(X_1, X_2) \sim H$, where H is according to (2) with $\lambda_1 = \lambda_2 = 1$. Let us define now the loss variables via $L_i = c_i \cdot e^{-r \cdot X_i}$ $(i = 1, 2)$, where c_i is the initial capital of institution i, and r is the risk-free interest rate. In the numerical study $c_1 = c_2 = 1$, and $r = 0.05$. For the risk functional R let us choose $R_t(L) := \mathbb{E}(L - t \,|\, L > t)$, where L is a loss variable, and t is a threshold, whose excess is considered as a "bad" event. In this example we observe a moderate increase in systemic risk as the shock parameters a_1, a_2 are getting increased.

(a_1, a_2)	(0,0)	(0,1)	(1,1)	(1,3)	(2,3)	(5,5)	(10,10)	(100,100)
$R(\oplus_C L_i)$	1.9049	1.9163	1.9275	1.9334	1.9373	1.9432	1.9466	1.9508
$\mathcal{R}(C, R; \mathbf{L})$	0	0.0114	0.0226	0.0285	0.0324	0.0383	0.0417	0.0459
rel.incr.	0%	0.59%	1.19%	1.49%	1.70%	2.01%	2.19%	2.41%

References

1. Lorenz, M.O.: Methods of Measuring the Concentration of Wealth, Publications of the American Statistical Association. Vol. 9 (New Series, No. 70) 209219. doi:10.2307/2276207 (1905)
2. Freund J. E.: A bivariate extension of the exponential distribution. In: Journal of the American Statistical Association, Vol. 56, No. 296 pp. 971-977 (1961)
3. Shaked, M., Shanthikumar, J.G.: Stochastic orders and their applications. Academic Press, Boston (1994)
4. Scarsini, M.: Multivariate convex orderings, dependence, and stochastic equality. In: Journal of Applied Probability, Vol. 35, pp. 93-103 (1998)
5. Shaked, M., Shanthikumar, J.G.: Stochastic orders. Springer, New York (2007)
6. Denuit, M., Dhaene, J., Goovaerts, M., Kaas, R.: Actuarial Theory for Dependent Risks: Measures, Orders and Models. Hohn Wiley & Sons Ltd (2005)
7. Kämpke, T., Radermacher, F.J.: Lorenz Curves and Partial Orders. In: Income Modeling and Balancing. Lecture Notes in Economics and Mathematical Systems, vol 679. Springer, Cham (2015)
8. Guzmics, S., Pflug, G. Ch.: Modelling cascading effects for systemic risk: Properties of the Fan Yu copula, Submitted to Dependence Modeling (2018)

Interlacing in cyclic scheduling

Máté Hegyháti and Olivér Ősz

Széchenyi István University, Győr, Hungary
hegyhati@sze.hu

Abstract. From the practical point of view, long-term scheduling, i.e. planning for a few months, holds a great importance. However, the scheduling of most production systems is a mathematically complex problem, thus, approaches relying on exact methods can usually provide the optimal solution for only up to a few days of planning period within a reasonable time.
The approach of cyclic scheduling assumes the schedule of the whole planning horizon to have a repeating nature. While the optimal solution may be lost if the cycle time is limited, this approach brings the problem to a manageable level, and provides solutions that are easier to execute and manage in practice.
Cyclic scheduling is a reasonable trade-off between solution quality and computational complexity, however, it also introduces new modeling challenges. This work focuses on interlacing, which is the overlapping of consecutive cycles. Allowing such overlaps may provide significantly better schedules, however, modeling them appropriately is not self-evident.
In this work, we emphasize the importance of interlacing, and present a novel model to address this issue.

Keywords: cyclic scheduling · no intermediate storage · mixed-integer linear programming.

1 Introduction and literature review

Cyclic scheduling assumes that a long-term schedule consists of the same repeating parts. As the exact methods for short-term scheduling problems are inadequate for long-term problems, this approach was introduced to bring them to a manageable complexity, while sacrificing optimality by only considering a subset of solutions. Moreover, cyclic schedules have the additional benefit of simpler manageability over non-cyclic schedules that may be more profitable.

The usual objective in cyclic scheduling is to maximize the hourly profit, however, that introduces an undesired non-linearity in the objective function. Many approaches tackle this difficulty by solving a series of MILP models, where either the cycle time or the quantity of products is fixed. Thus, a well-studied subproblem in the field is minimizing the cycle time for a given amount of products, which is the focus of this paper.

Cyclic scheduling, however, introduces another modeling difficulty: it does not assume anything about the "continuity" of a cycle within a machine. In a

repeatable solution, the execution of the operations of a cycle may overlap with the same operations in the next cycle, or any other cycles, a phenomenon called interlacing. Disregarding solutions with interlacing may lose optimality, however, modeling them is not a trivial feat.

Brucker and Kampmeyer [1] proposed a general model for different variations of the cyclic machine scheduling problem. They presented 3 methods for modeling the interlacing of subsequent repetitions of the cyclic schedule.

The first method, first published by Hanen [2] has a parameter H_{*0}, which tells how many different repetitions can coincide with each other. When $H_{*0} = 1$, the first task of repetition $k + 1$ can only start after the last task of repetition k is finished. This way, the problem is equivalent to non-cyclic makespan minimization. If $H_{*0} = h$, then h subsequent cycles may coincide such that any task in repetition $k + h - 1$ may precede any task in repetition k.

The second method, job repetition was first introduced in [3] but it is not widely used in the literature. Here, each job has its own cycle, that may be shifted from each other. The cycle time is the maximum flow time among jobs. Parameter $H_{job} = h$ is used to limit the interlacing between tasks of h consecutive repetitions of a job.

The third method is called machine repetition, and was first presented in [4]. It is widely used for periodic scheduling applications, along with the idea of MPS (Minimal Part Set). An MPS includes one or more batches from each job, and each cycle produces 1 MPS. The operations of an MPS-cycle on a certain machine may start later than the tasks of the same MPS on a different machine. This way, different machines may work on distinct MPSes at the same time but one machine cannot start operating on the next MPS until it finishes its last task from the current MPS (unless interlacing is allowed). It follows that the cycle time is the maximum workload among machines. In the model of [1], parameter H_{MPS} denotes the number of MPSes that can be present in the system in unfinished state at any time.

As it can be observed, interlacing cycles often require intermediates to be stored for a long time. To the case where No Intermediate Storage is available (NIS), [5] proposed a model, which is generalized to the above 3 interlacing methods in [1]. If a machine has no storage buffers, materials must be stored inside the machine after they are processed, until they are transported to the next stage. During this time, the machine cannot perform any processing tasks.

The model of [1] introduces the following constraint to ensure task i is blocking its machine:

$$t_{i'} - t_{s(i)} \geq p_i - \alpha K_{s(i)i'} \quad \forall i, i' \in O, M(i) = M(i')$$

Here, t_i is the start time of task i, p_i is its processing time, $s(i)$ is its successor, α is the cycle time (to be minimized), $K_{ii'}$ is the binary variable denoting the disjunctive arc between tasks i and i', and $M(i)$ is the machine which processes i. $K_{ii'} = 1$ if task i is started before i' during a cycle, and $K_{ii'} + K_{i'i} = 1$. The possible values of variables K can be efficiently enumerated by a tabu search metaheuristic or a search with constraint propagation technique, and with K fixed, the problem becomes an LP.

While above constraint ensures that $M(i)$ cannot process any other tasks until $s(i)$ is started, it allows cross-transfer of materials between $M(i)$ and $M(s(i))$. As it was discussed in [6], this may be infeasible in practice. Switching the contents of 2 machines without intermediate storage is impossible in certain production systems. For example, if the materials are liquids, or if one of the machines is single-gripper robotic arm, or hoist, simultaneously loading and unloading a machine.

2 Mathematical model

2.1 Input data

The proposed model is designed for the cyclic scheduling of job-shop problems. To ease the presentation and understanding of the model, the following assumptions are made:

- each operation has dedicated units, e.g., no unit-assignment is to be made during the optimization process
- each job has a single batch in the MPS
- no intermediate has a dedicated storage unit

Lifting these assumption would require minor modifications of the model, that could be done by anybody having moderate experience in scheduling and MILP formulations.

The exact input sets and parameters describing the problem are:

J set of jobs
M set of machines
$s_j \in \mathbb{Z}^+$ the number of steps needed to produce $j \in J$
$m_{j,s} \in M$ the machine needed for the s-th step of $j \in J$ ($s \in \{1, 2, \ldots, s_j\}$)
$pt_{j,s} \in \mathbb{R}^{0,+}$ the processing time of the s-th step of $j \in J$ ($s \in \{1, 2, \ldots, s_j\}$)

To further ease notation, the following sets are derived from the above:

$O = \{(j, s) \mid j \in J, s \in \{1, \ldots, s_j\}\}$ the set of all operations
$C = \{((j_1, s_1), (j_2, s_2)) \in O \times O \mid (j_1, s_1) \neq (j_2, s_2) \wedge m_{j_1,s_1} = m_{j_2,s_2}\}$ the set of operations sharing the same unit

2.2 Variables

The most important continuous variable in the model is T^{CT} that denotes the time offset, by which a cycle can be repeated without any overlapping in the schedule. The goal of the model is to minimize this variable.

The continuous variables $T^s_m \in [0, \infty]$ denote the start of a cycle in machine $m \in M$. No operations can be started in m before T^s_m. The interval $[T^s_m, T^s_m + T^{CT}[$ is regarded as the "own cycle" of the schedule. If an operation is scheduled within the interval $[T^s_m + k \cdot T^{CT}, T^s_m + (k+1) \cdot T^{CT}[$, it is said to overlap with

the kth cycle, i.e., it will be carried out during the cycle of the kth repetition of the schedule.

In the proposed model, there are two sets of non-negative continuous timing variables assigned to each operation to indicate this:

$t^{s,R}_{j,s}, t^{f,R}_{j,s}$ the real starting and finishing times of the operation $(j,s) \in O$
$t^{s,V}_{j,s}, t^{f,V}_{j,s}$ the *virtual* starting and finishing times of the operation $(j,s) \in O$

If an operation is within the cycle of the schedule, then its virtual and real times are equal, otherwise, they have the difference of $k \cdot T^{CT}$. The core idea of the proposed model is, that while the real timing of operations are calculated ordinarily, the starting and finishing times are "shifted back" to the cycle of the schedule to express sequencing constraints and avoid overlaps within the unit.

There are two types of discrete variables used in the model:

$k_{j,s} \in \mathbb{N}$ is the offset of operation $(j,s) \in O$
$x_{(j_1,s_1),(j_2,s_2)} \in \{0,1\}$ are the sequencing variables

Variable $x_{(j_1,s_1),(j_2,s_2)}$ takes the value of 1, if and only if operation (j_1,s_1) precedes (j_2,s_2) if shifted to the same cycle.

2.3 Constraints and objective function

The first constraints set the relation between the real and virtual starting times, and the shifting variables:

$$t^{s,V}_{j,s} = t^{s,R}_{j,s} + k_{j,s} \cdot T^{CT} \qquad \forall (j,s) \in O \tag{1}$$

$$t^{f,V}_{j,s} = t^{f,R}_{j,s} + k_{j,s} \cdot T^{CT} \qquad \forall (j,s) \in O \tag{2}$$

The following timing constraints need to be added to ensure processing times and NIS policy:

$$t^{f,R}_{j,s} \geq t^{s,R}_{j,s} + pt_{j,s} \qquad \forall (j,s) \in O \tag{3}$$

$$t^{s,R}_{j,s+1} = t^{f,R}_{j,s} \qquad \forall (j,s) \in O, s \neq s_j \tag{4}$$

As mentioned above, the scheduling constraints are made on the shifted, virtual starting and finishing times:

$$t^{s,V}_{j_2,s_2} \geq t^{f,V}_{j_1,s_1} - M \cdot (1 - x_{(j_1,s_1),(j_2,s_2)}) \qquad \forall ((j_1,s_1),(j_2,s_2)) \in C \tag{5}$$

M is a sufficiently large number. To enforce scheduling in one direction the following constraints are needed:

$$x_{(j_1,s_1),(j_2,s_2)} + x_{(j_2,s_2),(j_1,s_1)} = 1 \qquad \forall ((j_1,s_1),(j_2,s_2)) \in C \tag{6}$$

Finally, the starting of the own cycle are set by the following to constraints:

$$T^s_m \leq t^{s,V}_{j,s} \qquad \forall (j,s) \in O \tag{7}$$

$$T^s_m + T^{CT} \geq t^{f,V}_{j,s} \qquad \forall (j,s) \in O \tag{8}$$

Last, the objective is to minimize the variable T^{CT}.

2.4 Linearizing the model

Due to constraints (1) and (2), the model described in the previous chapter is quadratic. Let us refer to this MIQP model as $\mathbb{M}$. An MILP model, $\mathbb{M}^*$ is proposed based on $\mathbb{M}$ as follows:

1. The variable T^{CT} is replaced by a parameter with the same name.
2. A new continuous variable T^I is introduced, that refers to the minimal idle time at the end of the own cycle the machines have
3. Constraint (10) is slightly altered: $T^s_m + T^{CT} - T^I \geq t^{f,V}_{j,s} \qquad \forall (j,s) \in O$
4. The objective is to maximize T^I

A feasible solution to $\mathbb{M}^*$ proves, that the cycle time T^{CT} is feasible for the required products, thus the optimal solution of $\mathbb{M}$ is less than T^{CT} Moreover, it is easy to see, that an upper bound for $\mathbb{M}^*$, t^u provides the lower bound of $T^{CT} - t^u$ for the minimal cycle time.

Consequently, if the optimal solution of $\mathbb{M}^*$ is 0, then T^{CT} is the minimal cycle time.

Note, that a non-zero optimal solution for $\mathbb{M}^*$ does not guarantee that there exists a solution for $\mathbb{M}$ with less cycle time than T^{CT}.

3 Illustrative example

The capabilities of $\mathbb{M}^*$ is illustrated via an illustrative literature example by Brucker and Kampmeyer[1]. This problem has three products, all of which take 3 steps to be produced in the 3 available units.

In the subsections below, two cases with different number of products are investigated. All of the test runs were executed on a computer with 4 GB or RAM, and a 4-core Intel i5 processor running Ubuntu 18.04. The applied MILP solver is Gurobi 8.0.1.

The (1,1,1) case The sum of the processing times is 15, thus the optimal cycle time is definitely below that, so it can be used for T^{CT} in the first run.

It took 29.13 seconds for the solver to find the optimal solution of 8. This result ensures, that the minimal cycle time is between 7 and 15.

In the second run, T^{CT} was changed to 7. The purpose of this run is to verify, whether 7 is a feasible cycle time or not, which means, that if a feasible solution is found, there is no need to wait until it is proven optimal. The solver found the optimal solution and proved its optimality in 0.75 seconds.

The (1,3,2) case To test a more complex case, 2 and 1 additional batches of products 2 and 3 are added respectively. Similarly to the (1,1,1) case, T^{CT} is set to the sum of the processing times in the first run, i.e., 30.

The solver could find a feasible solution with $T^I = 17$, and reduce the upper bound to 18, however, it could not finish within 60 seconds. From this run, we

know, that 30 is a feasible cycle time, and the optimal cycle time is greater or equal to 12.

Running the model the second time with $T^{CT} = 12$ did not provide a feasible solution within 60 seconds. When T^{CT} was changed to 13, the solver reported the expected optimal solution of 1 in 34.09 seconds.

Although the proposed model could not find the optimal solution with this approach, it provided a solution 1 hour close to optimality within around two minutes.

4 Conclusion

Interlacing is a feature of cyclic schedules, that is not trivial to model, however, allowing it may result in better solutions. In this paper, a MIQP model was presented for cycle time minimization. A modified, linear version of the model was presented to check the feasibility of a provided cycle time, and provide a lower bound on the optimal cycle time. The proposed approach of running the linear model iteratively proved to be efficient in finding schedules with the optimal or close-optimal cycle time, that allow interlacing.

5 Acknowledgement

This research was supported by the EFOP-3.6.1-16-2016-00017; "Internationalization, initiatives to establish a new source of researchers and graduates, and development of knowledge and technological transfer as instruments of intelligent specializations at Szechenyi University" grant.

References

1. Brucker, P., Kampmeyer, T.: A general model for cyclic machine scheduling problems. Discrete Applied Mathematics **156**(13), 2561–2572 (2008)
2. Hanen, C.: Study of a NP-hard cyclic scheduling problem: The recurrent job-shop. European Journal of Operational Research **72**(1), 82–101 (1994)
3. Brucker, P., Kampmeyer, T.: Tabu search algorithms for cyclic machine scheduling problems. Journal of Scheduling **8**(4), 303–322 (2005)
4. Hitz, K.L.: Scheduling of flexible flow shops II. Technical Report LIDS-R-1049, Laboratory for Information and Decision Systems, MIT, Cambridge (1980)
5. Brucker, P., Kampmeyer, T.: Cyclic job shop scheduling problems with blocking. Annals of Operations Research **159**(1), 161–181 (2007)
6. Hegyháti, M., Majozi, T., Holczinger, T.: Practical infeasibility of cross-transfer in batch plants with complex recipes: S-graph vs MILP methods. Chemical Engineering Science **64**(3), 605–610 (2009)

Generating sufficient matrices

Tibor Illés[1] and Sunil Morapitiye[1]

Budapest University of Technology and Economics, Budapest, Hungary
sunil@math.bme.hu

Abstract. The class of sufficient matrices (**SU**) are important in the theory and solvability of the linear complementarity problems (LCP) as it was proven that SU-LCPs can be solved in polynomial number of iterations using interior point algorithms (IPA) that depends on the size of problem n, bit length L and the value $\kappa \geq 0$ that characterise the matrix property. Furthermore, the SU-matrices are the wider class of matrices for which criss-cross algorithms (CCA) solves the problem in finite number of iterations. Important deficiency of the published IPAs for SU-LCPs is that in most publications there are no numerical examination at all. Main reason for this might lie in the fact that only few SU matrices are known that does not fall into the classes of **PSD** and **P** matrices. Our goal is to generate different **SU** (but not **PSD** or **P**) matrices and test problems on which the different IPAs can be tested and the results can be compared.

Keywords: sufficient matrices · linear complementarity problems · interior-point algorithms

1 Introduction

The class of $\mathbf{P}_0$ matrices and its subclasses play an important role in the theory of the LCP. We say that a matrix is in $\mathbf{P}_0$ if all the principal minors are nonnegative. Couple of decades ago two subclass were defined: the class of **SU** matrices in 1989 by Cottle et al. [2] and the class of $\mathbf{P}_*(\kappa)$ matrices in 1991 by Kojima, et al. [1]. Kojima, et al. [1], Guu and Cottle [3] and Valiaho [4] in a series of publication proved that these matrix classes are equivalent, i.e. $\mathbf{P}_* = \mathbf{SU}$.

For different variants of CCAs and IPAs for SU-LCPs good survey can be found in Csizmadia [7] and M. Nagy [8], respectively.

Our goal is to generate sufficient matrices, therefore we need to find different ways to construct **SU**-matrices. By building a set of **SU**-matrices, test set problems for SU-LCPs can be defined, thus practical, computational performance of the IPAs can be tested. Important definitions, some lemmas and some constructions for **SU**-matrices are summarized. Finally, we illustrate a way we generated **SU**-matrices using the discussed lemmas and constructions.

In this paper we omit the proofs of the known lemmas and Construction 1, because those can be found in the literature. For our own, new results, sketch of the proofs are included for most of the cases.

We distinguish between the *scalar product* ($\mathbf{x}^T\mathbf{y} \in \mathbb{R}$), and the *Hadamard (coordinate-wise) product* ($\mathbf{x}\,\mathbf{y} \in \mathbb{R}^n$) of two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$.

2 Definitions, lemmas, constructions

Definition 1. *A matrix $A \in \mathbb{R}^{n\times n}$ is called a* ***PSD***-matrix *if for every vector* $\boldsymbol{x} \in \mathbb{R}^n$

$$\boldsymbol{x}^T A\, \boldsymbol{x} \geq 0. \tag{1}$$

Now, we are ready to introduce the classes $\mathbf{P}_*(\kappa)$ and $\mathbf{P}_*$ as a generalization of **PSD**-matrices (for details see Kojima et al. [1]).

Definition 2. *A matrix $A \in \mathbb{R}^{n\times n}$ is called a* $\boldsymbol{P}_*(\kappa)$-matrix *(for some $\kappa \geq 0$) if for every vector $\boldsymbol{x} \in \mathbb{R}^n$*

$$(1+4\kappa) \sum_{i\in I_+(\boldsymbol{x})} x_i\, y_i + \sum_{i\in I_-(\boldsymbol{x})} x_i\, y_i \geq 0 \tag{2}$$

where $\boldsymbol{y} = A\,\boldsymbol{x}$, $I_+(\boldsymbol{x}) = \{i : x_i\, y_i > 0\}$ and $I_-(\boldsymbol{x}) = \{i : x_i\, y_i < 0\}$.

If $\kappa = 0$ we get back the definition of **PSD**-matrices. Now, we can introduce

$$\mathbf{P}_* = \cup_{\kappa\geq 0}\mathbf{P}_*(\kappa). \tag{3}$$

The classes **CSU**, **RSU** and **SU** were defined by Cottle et al. [2].

Definition 3. *A matrix $A \in \mathbb{R}^{n\times n}$ is called* column sufficient *matrix (or belongs to the* ***CSU*** *class of matrices) if for every vector $\boldsymbol{x} \in \mathbb{R}^n$ it satisfies the following condition*

$$\boldsymbol{x}\,\boldsymbol{y} \leq \boldsymbol{0} \qquad \Rightarrow \qquad \boldsymbol{x}\,\boldsymbol{y} = \boldsymbol{0}, \tag{4}$$

where $\boldsymbol{y} = A\,\boldsymbol{x}$.

It is easy to see that a matrix is a sufficient matrix, if $I_+(\mathbf{x}) = \emptyset$ implies that $I_-(\mathbf{x}) = \emptyset$. Furthermore, any sufficient matrix has the property that if $\exists i \in I_-(\mathbf{x})$ then there should be another index $j \in I_+(\mathbf{x})$.

Definition 4. *A matrix $A \in \mathbb{R}^{n\times n}$ is called* row sufficient *matrix (or belongs to the* ***RSU*** *class of matrices) if $A^T \in$* ***CSU****.*

A matrix $A \in \mathbb{R}^{n\times n}$ is called sufficient *matrix (or belongs to the* ***SU*** *class of matrices) if $A \in$* ***CSU*** $\cap$ ***RSU****.*

In most cases the complexity of the IPAs depends on the handicap of the matrix, therefore testing the algorithm on a matrix with zero handicap is not appropriate, thus our goal is to generate sufficient matrices with positive handicap.

Definition 5. *$A \in \mathbb{R}^{n\times n}$, $\boldsymbol{x} \in \mathbb{R}^n$ where $\boldsymbol{x}^T A\boldsymbol{x} < 0$ and let us define the following function*

$$F_A(\boldsymbol{x}) = -\frac{\boldsymbol{x}^T A\boldsymbol{x}}{\sum_{i\in I_+(\boldsymbol{x})} x_i(A\boldsymbol{x})_i}. \tag{5}$$

The handicap of a **SU** matrix A is denoted by $\kappa(A)$, and

$$\kappa(A) = \begin{cases} 0 & \text{if } A \in \mathbf{PSD} \\ \frac{1}{4}\sup\{F_A(\mathbf{x}) | \mathbf{x}^T A\mathbf{x} < 0\} & \text{otherwise} \end{cases} \tag{6}$$

If $A \notin \mathbf{PSD}$ then there exists a vector $\mathbf{x}$ for which $\mathbf{x}^T A\mathbf{x} < 0$ therefore $\kappa(A)$ is well defined. The handicap of a **SU** matrix A is basically the smallest κ for which $A \in \mathbf{P}_*(\kappa)$.

Definition 6. *The* principal pivot operation *(PPO) transforms the equation system $\boldsymbol{y} = A\boldsymbol{x}$ ($A \in \mathbb{R}^{n\times n}$, $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$) into an equivalent one, where the variables x_i and y_i are exchanged for certain indices $i \in R$.*

If $R = \{1, 2, ..., j\}$ for some $j \in \{1, 2, ..., n\}$, then the coefficient matrix of the new equation system is:

$$\mathcal{P}_R A = \begin{pmatrix} A_{RR}^{-1} & -A_{RR}^{-1}A_{R\overline{R}} \\ A_{\overline{R}R}A_{RR}^{-1} & A_{\overline{R}\overline{R}} - A_{\overline{R}R}A_{RR}^{-1}A_{R\overline{R}} \end{pmatrix}. \tag{7}$$

The Lemmas 1-5 and Construction 1 can be found in Cottle, Pang and Stone [5].

Lemma 1. *Every principal submatrix of a sufficient matrix is also sufficient.*

Lemma 2. *If $A \in \mathbb{R}^{n\times n}$, $P = diag(p_1, ..., p_n)$, $Q = diag(q_1, ..., q_n)$, where $p_i q_i > 0$ for all i, and $B = PAQ$, then*

1. *If $A \in \boldsymbol{SU}$ then so is B.*
2. *If $A \in \boldsymbol{P}_*(\kappa)$ for some $\kappa \geq 0$, then $B \in \boldsymbol{P}_*(\kappa')$, where $\kappa' \geq \kappa$ is such that*

$$\frac{1 + 4\kappa'}{1 + 4\kappa} = \frac{\max_i(p_i/q_i)}{\min_i(p_i/q_i)}. \tag{8}$$

Lemma 3. *The matrix classes **CSU**, **RSU**, **SU**, $\boldsymbol{P}_*(\kappa)$, $\boldsymbol{P}_*$ are closed under the PPO and the principal permutations of rows and columns.*

Lemma 4. *The handicap of a sufficient matrix is at least as large as the handicap of any of its principal submatrix.*

Lemma 5. *The handicap is invariant under the PPO.*

Lemma 6 (Construction 1). *If $A \in \mathbb{R}^{n\times n}$ is in **SU** then so is the following matrix*

$$\begin{pmatrix} A & I \\ -I & D \end{pmatrix}, \tag{9}$$

where $I, D \in \mathbb{R}^{n\times n}$, and I is the identity matrix and D is a diagonal matrix with nonnegative elements.

Now, we summarize two of our constructions which were used during the sufficient matrix generation process. From now on let $\mathcal{I} = \{1, 2, \ldots, n\}$ and $\mathcal{J}_k = \{n + 1, n + 2, \ldots, n + k\}$ be set of indices.

Lemma 7 (Construction 2). *Let a sufficient matrix $A \in \mathbb{R}^{n\times n}$ be given. Let us define the matrix $C \in \mathbb{R}^{(n+k)\times(n+k)}$ in the following way*

$$c_{ij} = \begin{cases} a_{ij} & \text{if } 1 \le i,j \le n \\ 1 & \text{if } i = 1 \text{ and } j \in \mathcal{J}_k \\ -1 & \text{if } j = 1 \text{ and } i \in \mathcal{J}_k \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where $k \in \mathbb{N}$ is arbitrary. Then the matrix C is sufficient.

Proof. First we prove the column sufficiency using the definition. Let $\mathbf{x} \in \mathbb{R}^{n+k}$, $\tilde{\mathbf{x}} = \mathbf{x}_{\mathcal{I}}$, $\mathbf{y} = C\mathbf{x}$ and $\tilde{\mathbf{y}} = A\tilde{\mathbf{x}}$. The Hadamard product

$$\mathbf{xy} = \begin{pmatrix} x_1\tilde{y}_1 + x_1 \sum_{i\in\mathcal{J}_k} x_i \\ x_2\tilde{y}_2 \\ \vdots \\ x_n\tilde{y}_n \\ -x_{n+1}\, x_1 \\ \vdots \\ -x_{n+k}\, x_1 \end{pmatrix}. \tag{11}$$

If $-x_i\, x_1 > 0$ for some $i \in \mathcal{J}_k$ then $I_+(\mathbf{x}) \neq \emptyset$. Otherwise $-x_i\, x_1 \le 0$ for all $i \in \mathcal{J}_k$ so $\sum_{i\in\mathcal{J}_k} x_i x_1 = x_1 \sum_{i\in\mathcal{J}_k} x_i \ge 0$. In this case (as $A \in \mathbf{SU}$) we know that among the first n coordinate of the vector $\mathbf{xy}$ there must be at least one positive, or every coordinate is zero. This proves the column sufficiency, and the row sufficiency can be proved exactly in the same way. □

Note, that if $A \notin \mathbf{PSD}$ then $C \notin \mathbf{PSD}$.

Lemma 8. *The matrix $E \in \mathbb{R}^{n\times n}$ of ones is sufficient.*

Proof. Let $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y} = E\mathbf{x}$ and the corresponding Hadamard product

$$\mathbf{xy} = \begin{pmatrix} x_1 \sum_{i=1}^n x_i \\ \vdots \\ x_n \sum_{i=1}^n x_i \end{pmatrix}. \tag{12}$$

If $\sum_{i=1}^n x_i = 0$ then $\mathbf{xy} = \mathbf{0}$. If $\sum_{i=1}^n x_i$ is positive (negative) then $\exists i \in \mathcal{I}$ for which x_i is positive (negative) so $i \in I_+(\mathbf{x})$, therefore $I_+(\mathbf{x}) \neq \emptyset$. □

Previous lemma gives us a useful tool in proving the following statement.

Lemma 9 (Construction 3). *Let a matrix $D \in \mathbb{R}^{2n\times 2n}$ is defined as follows*

$$d_{ij} = \begin{cases} 1 & \text{if } (i,j) \in \mathcal{I}\times\mathcal{I} \cup \mathcal{J}_n \times \mathcal{J}_n \cup (n, n+1) \\ -1 & \text{if } (i,j) = (n+1, n) \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

Then D is sufficient matrix.

Proof. Let $\mathbf{x} \in \mathbb{R}^{2n}$, $\mathbf{y} = D\mathbf{x}$ and the corresponding Hadamard product

$$\mathbf{xy} = \begin{pmatrix} x_1 \sum_{i=1}^{n} x_i \\ \vdots \\ x_{n-1} \sum_{i=1}^{n} x_i \\ x_n \sum_{i=1}^{n} x_i + x_n x_{n+1} \\ x_{n+1} \sum_{i=n+1}^{2n} x_i - x_n x_{n+1} \\ x_{n+2} \sum_{i=n+1}^{2n} x_i \\ \vdots \\ x_{2n} \sum_{i=n+1}^{2n} x_i \end{pmatrix}. \tag{14}$$

If $x_n x_{n+1} = 0$ then D is sufficient because of Lemma 8. Considering Lemma 8 we can also see that if $x_n x_{n+1}$ is positive (negative) there must be a positive element among the first (second) n coordinate of the vector $\mathbf{xy}$, and this is exactly what we needed. Again, the row sufficiency can be proved exactly the same way so we omit that. □

This construction can be generalized: if $A, B \in \mathbb{R}^{n \times n}$ are in $\mathbf{P}_0$ of rank 1, and $F = diag(A, B)$, then $f_{n,n+1}$ and $f_{n+1,n}$ can be chosen such that F is sufficient (and $f_{n,n+1}\, f_{n+1,n} < 0$).

3 Example: building a sufficient matrix

Matrices of order 1 are sufficient if the (only) element is nonnegative. Sufficient matrices of order 2 were characterized by Guu and Cottle in [3]. Deciding whether a matrix of order 3 is in **SU** can be calculated on paper, or even in head quite fast, using the lemmas in [6]. (It takes less than a minute after some practice.) We calculated several sufficient matrices of order 3, and then we applied the mentioned lemmas and constructions to increase its size and to hide the original structure.

Let us illustrate the construction of a larger sufficient matrix from smaller ones using Constructions 1-3. and some of Lemmas 1-5. Let us start with a given sufficient matrix A. (Sufficiency of A can be checked using the definition.)

$$A = \begin{pmatrix} 1 & 2 & -2 \\ -1 & 1 & -3 \\ 2 & 1 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 2 & -2 & 1 & 1 \\ -1 & 1 & -3 & 0 & 0 \\ 2 & 1 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \end{pmatrix} \quad C = \begin{pmatrix} 1 & 2 & -2 & 1 & 1 & 1 & 0 \\ -1 & 1 & -3 & 0 & 0 & 0 & 1 \\ 2 & 1 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 3 \end{pmatrix}$$

The sufficient matrix B can be obtained from A by applying Construction 2. From matrix B, the sufficient matrix C can be built by using Construction 1.

$$D = \begin{pmatrix} 1 & -2 & -3 & -1 & 1 & -1 & 2 \\ 1 & 1 & 5 & -1 & 1 & -1 & -1 \\ 3 & -3 & 0 & -3 & 3 & -3 & 3 \\ -1 & 2 & 3 & 1 & -1 & 1 & -2 \\ 2 & -4 & -6 & -2 & 2 & -2 & 4 \\ -1 & 2 & 3 & 1 & -1 & 1 & -2 \\ -1 & -1 & -5 & 1 & -1 & 1 & 10 \end{pmatrix}$$

Applying PPO (Lemma 3) to matrix C and some scaling (Lemma 2) the resulting matrix D is sufficient matrix, as well. Due to the scaling, the handicap of matrices C and D, might be different.

All our sufficient matrix examples can be downloaded from the internet. Currently there are 10 pieces of matrices of order 10 and 20, and one matrix of order 700. As every principal submatrix of a sufficient matrix is sufficient (Lemma 1), the 700×700 matrix grants us an immense amount of sufficient matrices. By the time of the Vocal conference we are going to choose additional test examples, so the IPAs can be tested uniformly.

Acknowledgement. This research has been partially supported by the Hungarian Research Fund, OTKA (grant no. NKFIH 125700) and the Higher Education Excellence Program of the Ministry of Human Capacities in the frame of Artificial Intelligence research area of Budapest University of Technology and Economics (BME FIKP-MI/FM).

References

1. M. Kojima, N. Megiddo, T. Noma and A. Yoshise, *A Unified Approach to Interior Point Algorithms for Linear Complementarity Problems*, Springer-Verlag, Berlin (1991)
2. R. W. Cottle, J. S. Pang and V. Venkateswaran, *Sufficient Matrices and the Linear Complementarity Problem*, Linear Algebra and Its Applications, 114/115:231-249 (1989)
3. S.-M. Guu and R. W. Cottle, *On a subclass of P_0*, Linear Algebra and Its Applications, 223/224:325-335 (1995)
4. H. Väliaho, *P_*-Matrices Are Just Sufficient*, Linear Algebra and Its Applications, 239:103-108 (1996)
5. R. W. Cottle, J. S. Pang and R. E. Stone, *The Linear Complementarity Problem*, Academic, Boston (1992)
6. H. Väliaho, *Criteria for Sufficient Matrices*, Linear Algebra and Its Applications, 233:109-129 (1996)
7. Zs. Csizmadia, *New pivot based methods in linear optimization, and an application in petroleum industry*, PhD thesis, ELTE, Budapest (2007)
8. M. Nagy, *Interior point algorithms for general linear complementarity problems*, PhD thesis, ELTE, Budapest (2009)

A new interior point algorithm for a class of market equilibrium problems

Tibor Illés[1] and Anita Varga[1]

Budapest University of Technology and Economics, Budapest, Hungary
vanita@math.bme.hu

Abstract. The Fisher type market exchange model is a special case of the Arrow-Debreu type market exchange model. In this case, the players are divided into two groups, consumers and producers. Producers sell their products for money, and the consumers have an initial amount of money that they can use to buy a bundle of goods which maximizes their utility functions. In his article in 2006, Yinyu Ye presented and analyzed an interior point algorithm to solve the Fisher type market exchange models (MEMs) with linear and Leontief utility functions. He generalized the Eisenberg-Gale convex optimization formulation of the Fisher type MEM to a weighted analytic center problem, and then presented an interior point algorithm to solve it. We introduce a new interior point algorithm (IPA) to solve the weighted analytic center problem (WACP) discussed by Yinyu Ye. We discuss a new way to find the solution of the Fisher type linear and Leontief MEMs.

Keywords: Fisher type market exchange models · Interior point algorithms · Convex optimization.

1 Introduction

The Arrow–Debreu MEM was first formulated by Walras in 1874 [1]. In this model, there are m players with an initial endowment of n divisible goods and a utility function for every players. Every player sells his or her goods at the market and then uses the revenue to buy a bundle of goods which maximizes his or her utility function value. Our goal is to find the market clearing prices (i.e. equilibrium solution), meaning that every product is sold and every money is spent in the end. We consider pure exchange economies, therefore no production is involved in the model.
The existence of the market clearing prices was proved by Arrow and Debreu in 1954 [2], under some mild conditions and assuming that the utility functions are concave. The proof is not constructive and therefore the algorithms to compute the equilibrium solution are still commonly investigated topics in the literature. Even in case of Leontief utility functions, determinining an equilibrium solution is at least as hard as computing a Nash-equilibrium for two-player nonzero sum games, where the latter class is PPAD-complete (Polynomial Parity Arguments on Directed graphs, [5]). However, there are efficient algorithms for special classes

of Arrow–Debreu MEMs. In case of linear utility functions, Jain gave a convex optimization formulation for the problem, and proved that the problem class is polynomially solvable [13]. Based on this formulation, a more efficient polynomial IPA was given by Ye in [4]. The convergence of this algorithm follows from the theory of Nesterov and Nemirovski on self-concordant barrier functions [7].
In the Fisher market equilibrium problem, the players are divided into two groups, producers and consumers. The producers bring their goods to the market, and each consumer has an initial amount of money. Similarly to the Arrow–Debreu case, every consumer has a utility function, which he or she wants to maximize. This problem can be considered as a special case of the Arrow–Debreu market exchange model, where money is the product brought to the market by the consumers.
Klafszky [15] in 1981 for Fisher type MEMs with linear utility functions introduced a combinatorial algorithm, very similar to that of the Hungarian Method [18] for the transportation problem. Furthermore, Klafszky studied the connection between the corresponding Eisenberg-Gale problem [3] with the geometric programming problem [17]. Recently, Eisenberg-Nagy et al. [16] generalized some of Klafszky's results [15] for Fisher type MEMs with homogenuos utility functions.
In this paper, we consider the Fisher MEM with linear utility functions. Although there are several efficient algorithms to solve this problem, we introduce a very simple, new IPA.

2 The Fisher exchange market equilibrium problem

In case of the Fisher type MEM ([8],[9]), the players are divided into two groups, producers and consumers. Let us denote the number of producers by pr, and the number of consumers by c. Without loss of generality we may assume that each producer has one unit of her good, and each consumer has an initial endowment of money (w_i, $i \in C = \{1, 2, \ldots, c\}$). The vector $\mathbf{x}_i$ represents the bundle of goods bought by consumer i, and $u_i(\mathbf{x}_i)$ is her utility function ($i \in C$). Every consumer uses her money to buy a bundle of goods which maximizes her utility function value. We would like to determine the prices (p_j, $j \in P = \{1, 2, \ldots, pr\}$) for the products so that in the end every money is spent and every product is sold (market clearing prices).
This problem is indeed a special case of the Arrow–Debreu MEM, if we consider money as the product brought to the market by the consumers.
With the above notations, and assuming that the utility functions are linear, the problem can be formulated as follows:

$$\left.\begin{array}{lll} \max\limits_{x_i} u_i(\mathbf{x_i}) = \sum_j u_{ij}x_{ij} & & \\ \left.\begin{array}{l} \sum_j p_j x_{ij} \le w_i, \\ x_{ij} \ge 0 \ \forall j \end{array}\right\} & (FC_i) & \forall i \\ \sum_i x_{ij} = 1 & \forall j \in P & \\ p_j \ge 0 & \forall j \in P & \end{array}\right\} (FL)$$

From now on, we make the following assumptions:

A1. Every consumer's initial endowment is positive ($w_i > 0 \; \forall i \in C$).
A2. Every consumer has at least one product that she like (at least one $u_{ij} > 0$ for every $i \in C$).
A3. Every good is valuable for at least one consumer (at least one $u_{ij} > 0$ for every $j \in P$).

Eisenberg and Gale [3] proved that the solution can be obtained by solving the following convex optimization problem:

$$\left.\begin{aligned} \max & \sum_{i \in C} w_i \log u_i & & \\ \text{subject to } & \sum_{i \in C} x_{ij} = 1 & & \forall j \in P \\ & u_i - \sum_{j \in P} u_{ij} x_{ij} = 0 & & \forall i \in C \\ & u_i, \; x_{ij} \geq 0 & & \forall i, j \end{aligned}\right\} (EGP)$$

and the the optimal Lagrange multipliers of this problem for the first *pr* conditions are the market clearing prices. In [4], Ye considered a more general problem, and gave a polynomial time IPA to determine the market clearing prices:

$$\left.\begin{aligned} \max & \sum_{j=1}^{n} w_j \log x_j \\ \text{subject to } & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned}\right\} (WCPP)$$

where $A \in \mathbb{R}^{mxn}$ has full row rank, $\mathbf{b} \in \mathbb{R}^m$, $w_j \geq 0$ is the nonnegative weight on the jth variable. The KKT-system is the following:

$$\begin{aligned} \mathbf{sx} &= \mathbf{w} \\ A\mathbf{x} &= \mathbf{b}, \; \mathbf{x} \geq \mathbf{0} \\ -A^T\mathbf{y} + \mathbf{s} &= \mathbf{0}, \; \mathbf{s} \geq \mathbf{0} \end{aligned} \tag{1}$$

where $\mathbf{y}$ and $\mathbf{s}$ are the Lagrange-multipliers.
Let $\mathcal{F} = \{(\mathbf{x}, \mathbf{y}, \mathbf{s}) \in \mathbb{R}^{\{n+m+n\}} : A\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}, \mathbf{s} = A^T\mathbf{y} \geq \mathbf{0}\}$ be the set of feasible solutions for (1), $\mathcal{F}^+ = \{(\mathbf{x}, \mathbf{s}) \in \mathbb{R}^{2n} : \exists \mathbf{y} \in \mathbb{R}^m, (\mathbf{x}, \mathbf{y}, \mathbf{s}) \in \mathcal{F}, \mathbf{x} > \mathbf{0}, \mathbf{s} > \mathbf{0}\}$ is the set of strictly feasible solutions. The following theorem of Illés [11] provides that system (1) has a unique solution:

Theorem 1 (Illés [11]). *Let $A \in \mathbb{R}^{mxn}$ with full row rank, $\mathbf{b} \in \mathbb{R}^m$ and $\mathcal{F}^+ \neq \emptyset$. Then $\forall \mathbf{w} \in \mathbb{R}^n_+ \; \exists! (\mathbf{x}, \mathbf{s}) \in \mathcal{F}^+ : \mathbf{sx} = \mathbf{w}$.*

Ye [4] used a more general version [12] of Theorem 1. to argue that (1) can be solved in polynomial time. Furthermore, Ye [4] pointed out that a strictly feasible solution $(\mathbf{x}_0, \mathbf{s}_0), \mathbf{y}_0$ always can be determined easily.

2.1 The IPA of Ye

In [4], Ye gave the following primal-dual path following IPA to solve the problem:

Input: A, $\mathbf{b}$, $\mathbf{w}$
$\mathbf{x} \in \mathcal{P}_+$, $(\mathbf{s}, \mathbf{y}) \in \mathcal{D}_+$
$\mu \geq 0$ error measure, $0 < \eta < 1$ constant parameter,
$\epsilon > 0$, $\hat{w}_j = \max\{\mu, w_j\}\ \forall j$
while $\eta\mu \geq \epsilon$ **do**
$\mathbf{x} := \mathbf{x} + \Delta x$;
$\mathbf{s} := \mathbf{s} + \Delta s$;
$\mathbf{y} := \mathbf{y} + \Delta y$;
$\mu := \left(1 - \frac{\eta}{\sqrt{n}}\right)\mu$;
$\hat{w}_j := \max\{\mu, \hat{w}_j\}\ \forall j$
end

The parameter settings suggested by Ye in [4] were $\eta = \frac{1}{4}$ and $\mu = \max_j w_j$. To determine the Δx, Δs, Δy Newton-directions, the following system of linear equations has to be solved:

$$\begin{aligned} \mathbf{s}\Delta x + \mathbf{x}\Delta s &= \mathbf{w} - \mathbf{x}\mathbf{s} \\ A\Delta x &= \mathbf{0} \\ -A^T\Delta y + \Delta s &= \mathbf{0} \end{aligned} \tag{2}$$

The algorithm of Ye can determine an ϵ-optimal solution for the Fisher type MEM with linear utility functions in at most

$$O\left(\sqrt{cp}\log\left((c+p)\max_i(w_i)/\epsilon\right)\right) + O(c)$$

iterations ([4],[10]).

3 The new interior point algorithm

Our goal is to give a new and simple IPA for the Fisher type MEM, that doesn't use the information obtained from the central path and furter refines the constructive proof of Theorem 1. So our main question is the following: under some additional special constraints, can we define an IPA without using the information provided by the central path, and ensure the polynomial computation time of the new IPA? The algorithm is based on formulation (1) by Ye, and uses the strictly feasible starting point from [4]. So from now on, we assume that we have a strictly feasible solution $(\mathbf{x}_0, \mathbf{s}_0)$: $\mathbf{x_0 s_0} = \mathbf{w}_0$.
Before introducing the algorithm, we need the definition of the hyperrectangle generated by two vectors:

Definition 1. *Let* $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$. *Then*

$$\mathcal{T}(\mathbf{a}, \mathbf{b}) = \{\mathbf{u} \in \mathbb{R}^n : a_i \leq u_i \leq b_i \text{ or } b_i \leq u_i \leq a_i\ \forall i\}$$

is the hyperrectangle *defined by the vectors* $\mathbf{a}$ *and* $\mathbf{b}$.

In each iteration, we consider the hyperrectangle generated by our current $\mathbf{w}_i$ and the $\mathbf{w}$ vector given by the model (or its subset if it is not included in the feasible set), and assume that $int(\mathcal{T}(\mathbf{w}_i, \mathbf{w})) \neq \emptyset \ \forall i$. We determine the Newton-directions from system (2).

Theorem 2 (Illés, [11]). *Let* $(\mathbf{x}_i, \mathbf{s}_i) \in \mathcal{F}^+$, $\mathbf{w_i} = \mathbf{x}_i\mathbf{s}_i$, *and* $int\mathcal{T}(\mathbf{w}_i, \mathbf{w}) \neq \emptyset$, *where vector* $\mathbf{w} \in \mathbb{R}^n_+$ *be given in (1). Then there exists an* $\alpha > 0$, *such that*

$$\mathbf{w}_{i+1} = (\mathbf{x}_i + \alpha\Delta x)(\mathbf{s}_i + \alpha\Delta s) \in int\mathcal{T}(\mathbf{w}_i, \mathbf{w}),$$

where $(\Delta x, \Delta y, \Delta s)$ *solves (2).*

The theorem provides that there exists an α steplength, for which if we take the $\mathbf{x}_i + \alpha\Delta x$, $\mathbf{s}_i + \alpha\Delta s$ steps, $\mathbf{w}_{i+1} = (\mathbf{x}_i + \alpha\Delta x)(\mathbf{s}_i + \alpha\Delta s)$ stays in the $\mathcal{T}(\mathbf{w}_i, \mathbf{w})$ hyperrectangle.
Based on this theorem, we think that with the fine tuning of the step length an algorithm can be introduced. The pseudo-code of the algorithm is the following:

Input: A, $\mathbf{b}$, $\mathbf{w}$
$(\mathbf{x}_0, \mathbf{s}_0) \in \mathcal{F}^+$, $\mathbf{y}_0 : (\mathbf{x}_0, \mathbf{s}_0, \mathbf{y}_0) \in \mathcal{F}$
$\epsilon > 0$, $i := 0$
$\mathbf{x} := \mathbf{x}_0, \mathbf{s} := \mathbf{s}_0$, $\mathbf{w}_0 := \mathbf{xs}$
while $\|\mathbf{w} - \mathbf{xs}\|_2 \geq \epsilon$ **do**
  Determine $\Delta x, \Delta y, \Delta s$ from (2) ;
  $\alpha := \text{argmin}_{\alpha>0} \left\{ \|\mathbf{w} - (\mathbf{x} + \alpha\Delta x)(\mathbf{s} + \alpha\Delta s)\|_2^2 : (\mathbf{x} + \alpha\Delta x, \mathbf{s} + \alpha\Delta s) \in \mathcal{T}(\mathbf{w}_i, \mathbf{w}) \cap \mathcal{F}^+ \right\}$;
  $\mathbf{x} := \mathbf{x} + \alpha\Delta x$;
  $\mathbf{s} := \mathbf{s} + \alpha\Delta s$;
  $\mathbf{y} := \mathbf{y} + \alpha\Delta y$;
  $i := i + 1$;
  $\mathbf{w}_i := \mathbf{xs}$
end

With the formulas given in [4], a strictly feasible $(\mathbf{x}_0, \mathbf{s}_0, \mathbf{y}_0)$ solution can be computed from the data exactly. The existence of a proper α at every iteration follows from Theorem 2. By construction, the feasibility of $(\mathbf{x}, \mathbf{s}, \mathbf{y})$ is always maintained.
If we consider the $\mathcal{T}(\mathbf{w}_i, \mathbf{w})$ series of hyperrectangles, it is clear from the construction (α is always strictly positive) that

$$\mathcal{T}(\mathbf{w}_i, \mathbf{w}) \supset \mathcal{T}(\mathbf{w}_{i+1}, \mathbf{w}) \quad \forall\, i \in \mathbb{N}.$$

The volumes of the $\mathcal{T}(\mathbf{w}_i, \mathbf{w})$ hyperrectangles form a strictly decreasing sequence in a compact set. The sequence constructed from the $\|\mathbf{w} - \mathbf{w}_i\|_2$ distances form a strictly decreasing sequence as well. Thus the convergence of the algorithm can be shown. The computational performance of this simple algorithm needs further investigations.

Acknowledgement
This research has been partially supported by the Hungarian Research Fund, OTKA (grant no. NKFIH 125700) and the Higher Education Excellence Program of the Ministry of Human Capacities in the frame of Artificial Intelligence research area of Budapest University of Technology and Economics (BME FIKP-MI/FM).

References

1. Walras, L.: Éléments d'économie politique pure, ou, Théorie de la richesse sociale. F. Rouge (1896)
2. Arrow, K. J., Debreu, G.: Existence of an equilibrium for a competitive economy. In: Econometrica: Journal of the Econometric Society, pp. 265-290. (1954)
3. Eisenberg, E., Gale, D.: Consensus of subjective probabilities: The pari-mutuel method. In: The Annals of Mathematical Statistics, 30(1), pp. 165-168. (1959)
4. Ye, Y.: A path to the Arrow–Debreu competitive market equilibrium. In: Mathematical Programming, 111(1-2), pp. 315-348. (2008)
5. Codenotti, B., Saberi, A., Varadarajan, K., Ye, Y.: Leontief economies encode nonzero sum two-player games. In: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm. Society for Industrial and Applied Mathematics. pp. 659-667. (2006)
6. Jain, K.: A polynomial time algorithm for computing an Arrow–Debreu market equilibrium for linear utilities. In: SIAM Journal on Computing, 37(1), pp. 303-318. (2007)
7. Nesterov, Y., Nemirovskii, A.: Interior-point polynomial algorithms in convex programming (Vol. 13). Siam. (1994)
8. Brainard, W. C., Scarf, H. E.: How to compute equilibrium prices in 1891. Cowles Foundation for Research in Economics. (2000)
9. Scarf, H.: The approximation of fixed points of a continuous mapping. In: SIAM Journal on Applied Mathematics, 15(5), 1328-1343. (1967)
10. Potra, F. A.: Weighted complementarity problems—a new paradigm for computing equilibria. In: SIAM Journal on Optimization, 22(4), 1634-1654. (2012)
11. Illés, T.: Lineáris optimalizálás elmélete és belsőpontos algoritmusai. (2014) `https://www.tankonyvtar.hu/hu/tartalom/tamop412A/2011_0025_mat_4`
12. Güler, O.: Existence of interior points and interior paths in nonlinear monotone complementarity problems. In: Mathematics of Operations Research, 18(1), 128-147. (1993)
13. Jain, K.: A polynomial time algorithm for computing an Arrow–Debreu market equilibrium for linear utilities. In: SIAM Journal on Computing, 37(1), 303-318. (2007)
14. Devanur, N. R., Papadimitriou, C. H., Saberi, A., Vazirani, V. V.: Market equilibrium via a primal-dual-type algorithm. In: Foundations of Computer Science, Proceedings. The 43rd Annual IEEE Symposium on (pp. 389-395). IEEE. (2002)
15. A lineáris cseremodell egyensúlyi árának meghatározása geometriai programozással, Alkalmazott Matematikai Lapok 7, 139-157. (1981)
16. Eisenberg-Nagy, M., Illés, T., Lovics, G.: Market exchange models and geometric programming. Central European Journal of Operations Research, 1-21. (2018)
17. Klafszky, E.: Geometric Programming, IIASA Systems Analysis and Related Topics No. 11 (1976)
18. Kuhn, H. W.: The Hungarian method for the assignment problem. Naval research logistics quarterly, 2(1-2), 83-97. (1955)

A comparison of matching algorithms for Kidney Exchange Programs

Tiago Monteiro[1], Xenia Klimentova[1], João Pedro Pedroso[1,2], and Ana Viana[1,3]

[1] INESC TEC, Porto, Portugal
[2] Faculdade de Ciências, Universidade do Porto, Porto, Portugal
[3] ISEP - School of Enginnering, Polytechnic of Porto, Porto, Portugal
tiago.p.monteiro@inesctec.pt

Abstract. Kidney Exchange Programs (KEP) allow an incompatible patient-donor pair, whose donor cannot provide a kidney to the respective patient, to have a transplant exchange with another pair in a similar situation if there is compatibility.
In this paper we propose two matching algorithms that address the waiting times of the pairs in a pool, by hierarchically maximizing the number of transplants giving preference to the pairs that have waited longer. The algorithms differ in the strategies used for finding feasible exchanges, as follows. One algorithm runs periodically (e.g. every 3 month); the other runs as soon as the pool is updated allowing for a new exchange. The two algorithms are compared to similar approaches in the literature, that aim at maximizing the number of transplants, through computational experiments.

Keywords: Greedy hierarchical algorithm · Integer programming · Kidney exchange programs · Simulation · Waiting time

1 Introduction

Kidney transplantation is currently the most effective treatment for patients with end-stage renal disease, but finding a suitable kidney can be difficult. There are two different sources for kidneys: from deceased donors and from living donors. However, for the case of living donation the patient and willing donor often do not meet compatibility requirements. This deadlock can be overcome by kidney exchange programs (KEPs) that allow incompatible pairs to perform an exchange between them if the donor in one pair is compatible with the patient in the other pair and vice versa. This is the simplest case, resulting in the so called 2-cycle exchange. However, the size of exchange cycles can be increased, following the same reasoning.

Another possible organization for exchanges is a chain initiated by an altruistic donor, i.e. a donor with no associated patient that donates a kidney for no return. The altruistic donor donates a kidney to a patient of the first pair in the chain, his/her donor to the patient in the following pair and so forth. The last donor in the chain can either donate to the deceased donors waiting

list or act as a bridge donor for the next matching. Due to the several practical constraints the length of a cycle, and frequently of a chain have to be limited by some constant k. In most programs k is set to 3.

KEPs are managed by central or local authorities that collect the incompatible pairs or altruist's registrations and try to identify the exchanges that optimize a given objective. The dynamics of the evolving matching pool, where pairs enter and leave over time, is captured by several models. In [1] the authors conduct simulations that aim at maximizing the number of transplants performed under different time intervals between matches. In [2] authors study how dynamic policies affect the waiting times. Three different settings of feasible solutions are considered: only 2-cycles, 2 and 3-cycles, and a single unbounded chain. Average waiting time is considered as a measure of efficiency; results show that a greedy policy, where exchanges are done as soon as they are available, is nearly optimal.

In our work we propose to address the waiting time of pairs in the pool by hierarchically maximizing in the exchange the number of pairs that waited longer. We develop two matching algorithms that aim at addressing this objective. Matchings are performed either periodically, or as soon as possible (similarly to the work in [2]). We compare results with the cases where maximization of the number of transplants is the only objective considered.

2 Kidney exchange pool and hierarchical waiting times' optimization

We consider a dynamic KEP pool where pairs and altruistic donors appear over time. The pool is represented by a directed graph $G = (V, A)$. The set of vertices $V = P \cup N$ is composed by a set of incompatible pairs P and a set of altruistic donors N. The set of arcs A represent compatibility between vertices: $(i, j) \in A$ if the donor in vertex $i \in V$ and the patient in vertex $j \in V$ are compatible. A feasible exchange is a set of disjoint cycles or chains, where cycles are formed with vertices from set P and chains are initiated by an altruistic donor from set N, followed by vertices from set P. We assume that the maximum size of cycles and chains is limited by a value k (for the case of chains this limit is on the number of vertices involved in a chain, including the altruistic donor). The last donor in a chain donates to the deceased donor's waiting list.

In this sections we describe the two matching approaches proposed. The first one, presented in subsection 2.1, is referred in [2] as greedy. In this case the matching algorithm is run whenever a new incompatible pair or an altruistic donor joins the pool. In the second matching algorithm, subsection 2.2, exchanges are found periodically, i.e., the algorithm is run periodically.

2.1 Greedy hierarchical algorithm

A myopic greedy algorithm was proposed in [2] in the following way: exchanges are performed whenever a cycle or a chain is formed with an arriving altruistic

donor or incompatible pair. Possible ties are broken randomly, and the last donor in the chain acts as an altruistic donor in the following matching. We adapt this algorithm for our settings. Namely, we consider that the maximum length of cycles and chains is the same (k), and choose (randomly in case of multiple possibilities) the one with the largest number of transplants. Furthermore, we assume that the last donor in a chain donates to the deceased donors waiting list. We will refer to this version of the greedy algorithm as $Greedy_{max}$.

Furthermore, we propose a new algorithm where instead of randomly choosing a solution that maximizes the number of transplants, we aim at maximizing the number of transplants while reducing patient's waiting time in the pool. Upon arrival of a pair or of an altruistic donor to the pool, the algorithm checks if new cycles or chains are created with the new arrival. If they are, in case the arrival promotes the creation of more than one cycle or chain, preference is given to the one which contains the pair that has been in the pool for a longer time. Ties are broken by considering the pair with the second longest waiting time, and so forth. For the case of chains, the altruistic donor's waiting time is only considered after pairs are considered, i.e., when we have chains that only differ in the altruistic donor. This case happens when there is more than one altruistic donor in the pool and, upon a new pair arrival, more than one chain (initiated by different altruistic donors, but containing exactly the same pairs) is created. When a potential solution can be either a cycle or a chain and the solution only differs in the fact of the chain having the altruistic donor, priority is given to cycles. This case can happen when an unmatched altruistic donor is already in the pool when a new pair arrives. We will refer to this version of the greedy algorithm as $Greedy_{WT}$.

2.2 Hierarchical integer programming

In another approach we consider that the matching is performed periodically, within given intervals of time (this value is set to 3 months in many countries). In our work this problem is modeled and solved with integer programming (IP) using the cycle formulation [3,4].

Considering the compatibility graph $G = (V, A)$, let $\mathcal{C}$ be a set of cycles and chains with at most k vertices in G. Let $V(c)$ denote the set of vertices that belong to cycle/chain c. By associating the variable x_c for each $c \in \mathcal{C}$, where $x_c = 1$ if cycle/chain c is selected, 0 otherwise, we can write the following IP model:

$$\text{Maximize} \quad \sum_{c \in \mathcal{C}(k)} w_c x_c \tag{1a}$$

$$\text{Subject to:} \quad \sum_{c : i \in V(c)} x_c \leq 1 \qquad \forall i \in V \tag{1b}$$

$$x_c \in \{0, 1\} \qquad \forall c \in \mathcal{C}(k). \tag{1c}$$

The objective function (1a) maximizes the weighted sum of transplants to be performed and constraints (1b) ensure that each vertex is in at most one of the

selected cycles (i.e., each donor may donate, and each patient may receive only one kidney). We will refer to this IP model as IP_{max}.

Similarly to the previous greedy algorithms, we will also consider an IP model where the waiting times of the patients in the pool are addressed. For doing so, for each matching period t we first find exchanges that maximize the number of pairs that waited for t running periods. This can be done by replacing w_c in formulation (1a)-(1c) by w_c^t, where w_c^t is the number of pairs in cycle c that have been in the pool for t matching periods. The optimal value of this problem is denoted by v_t^*. Then, in case there are multiple optimal solutions that provide v_t^*, we choose the ones that maximize the number of patients that waited for $t-1$ periods, similarly, by considering coefficient w_c^{t-1}, that will provide optimum value v_{t-1}^*, in the objective function (1a) and imposing the following additional constraints:

$$\sum_{c \in \mathcal{C}} w_c^t x_c \geq v_t^* \tag{2}$$

The process is repeated until $t = 0$ by, adding to the IP problem constraints (2) in each iteration, with w_c^t replaced by $w_c^{t-1}, \ldots, w_c^0$.

3 Computational analysis

In this section we validate and compare the four matching policies described in section 2: $Greedy_{max}$, $Greedy_{WT}$ and IP_{max}, IP_{WT}. Computational results were obtained for 100 instances, generated with the simulator developed in [5], considering an horizon of 6 years. For the periodic matching algorithms, the interval between matchings was set to 90 days (3 months). We assumed that each patient has only one associated donor, and that pairs and altruistic donors do only leave the pool when matched. Furthermore, if more than one pair or altruistic donor enters the pool in the same day, we prioritize pairs that have O-blood type patients and, if necessary, with a higher value of Panel-reactive antibody (PRA). The maximum length of cycles and chains was set to 3.

Figure 1 illustrates the average waiting times within each period of 90 days for: 1) matched pairs (lower part of the graph), 2) pairs remaining in the pool after matching is performed (upper part of the graph) and, 3) all pairs in the program (matched and not matched). As shown, average waiting times of matched pairs for approaches that prioritize pairs that have been in the pool for a longer time is higher when compared to the other algorithms ($Greedy_{max}$ and IP_{max}). The reason for this is the fact, from all potential solutions, the selected solution is the one that has pairs with longer waiting times. We can also observe that at the beginning of the simulation the IP algorithms have higher waiting times. That can be justified by the time pairs have to wait until the match day arrives. The average waiting time for the pairs remaining in the pool is lower for approaches that take into account the waiting time. This can be justified by the reason presented before. The same happens for the average time of pairs in the program.

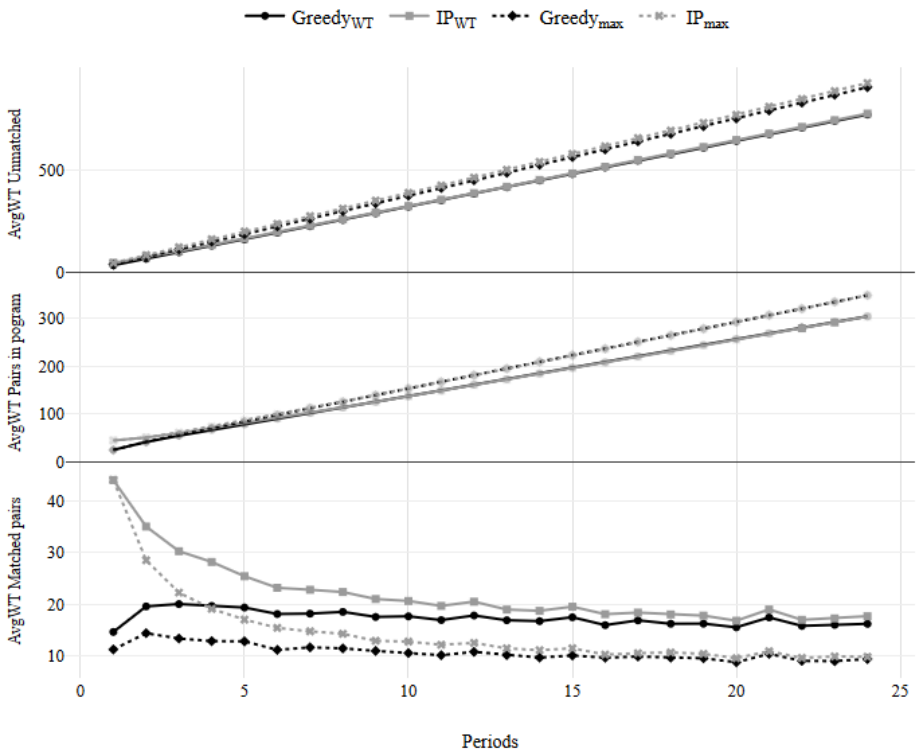


Fig. 1. Average waiting times (AvgWT) for matched pairs, unmatched pairs in the pool and all pairs in the program (matched and unmatched) using four different approaches. Greedy approaches are represented by black lines and IP approaches by grey. Solid lines represent algorithms that consider pairs' waiting times and dashed lines represent those that only consider the maximization of the number of transplants.

In table 1 we present the average number of matched pairs at the end of each simulation year. We can observe that, at the end of the simulation, the application of the *Greedy* algorithms result in less transplants when compared with *IP* algorithms. Moreover, the approaches that consider waiting times present in average a lower number of transplants when compared with those that neglect waiting times.

Table 1. Average number of pairs matched at the end of each simulation for each approach.

	Year						
	1	2	3	4	5	6	**Total**
$Greedy_{WT}$	153.53	163.92	165.06	165.71	163.89	166.53	**978.64**
$Greedy_{max}$	154.1	164.74	166.06	166.21	164.67	167.38	**983.16**
IP_{WT}	159.88	164.84	165.07	165.77	163.79	165.92	**985.27**
IP_{max}	163.32	166.4	166.48	167.03	165.48	168.49	**997.2**

4 Conclusions

In this work we simulate different matching policies in order to identify how they affect pairs waiting time. We propose approaches based on greedy and periodic matching and as evaluation criteria we consider the maximization of the number of transplants and minimization of waiting times of patients. Preliminary computational results show that average waiting times of unmatched patients are reduced for approaches that consider pairs waiting time. This is achieved by slightly sacrificing the total number of transplants performed and increasing average waiting times of matched pairs. As future work we intend to implement a lexicographic procedure that first maximizes the number of transplants and, in a second stage, select the solution involving the pair(s) with longer waiting times. Furthermore, we intend to extend computational results by considering graphs of different density and different frequencies for pair and altruistic donor's arrival.

Acknowledgements This work is financed by the ERDF European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme, and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within project "mKEP - Models and optimisation algorithms for multicountry kidney exchange programs" (POCI-01-0145-FEDER-016677), by FCT project SFRH/BPD/101134/2014 and by COST Action CA15210, ENCKEP, supported by COST (European Cooperation in Science and Technology) – http://www.cost.eu/.

References

1. M. Beccuti, V. Fragnelli, G. Franceschinis, and S. Villa. Dynamic simulations of kidney exchanges. In Bo Hu, Karl Morasch, Stefan Pickl, and Markus Siegle, editors, *Operations Research Proceedings 2010*, pages 539–544, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.
2. R. Anderson, I. Ashlagi, D. Gamarnik, and Y. Kanoria. Efficient dynamic barter exchange. *Operations Research*, 65(6):1446–1459, 2017.
3. D.J. Abraham, A. Blum, and T. Sandholm. Clearing algorithms for Barter exchange markets: Enabling nationwide kidney exchanges. *In: Proceedings of the 8th ACM conference on Electronic commerce, June 13-16*, pages 295–304, 2007.
4. M. Constantino, X. Klimentova, A. Viana, and A. Rais. New insights on integer-programming models for the kidney exchange problem. *European Journal of Operational Research*, 231(1):57–68, 2013.
5. N. Santos, P. Tubertini, A. Viana, and J. P. Pedroso. Kidney exchange simulation and optimization. *Journal of the Operational Research Society*, 68(12):1521–1532, 2017.

Mathematical model for power plant scheduling and its properties

Péter Naszvadi

Budapest University of Technology and Economics, Budapest, Hungary
vuk@math.bme.hu

Abstract. The power plant scheduling is one of the main tasks of the energy traders and electric transmission system operators worldwide. It is a cardinal question to achieve a feasible and suboptimal integer solution in a heck of time usually in a daily period regularly, where the objective is the cost of the produced energy to be minimized. Due to the more or less precise short-term electric consumers' forecasts, in practice the schedule generation is handled as a mixed integer programming problem. In this paper, it will be shown that the constructed optimization model contains a discretized form of a differential equation system covering the most important technical constraints, including the gradients and the bounds of the performance of each unit. It will also be proven that the mixed integer programming model has a network matrix, thus it can be solved in strongly polynomial time.
Most of the technical and/or financial constraints make the feasibility problem NP-complete as it will be shown here later. Some of them are very common - however, in practice, adding a strict subset of such constraints still keeps the necessary resources low for solving the schedule generation model. In 2004 the author proved in his MSc thesis that the polynomial part of the model, namely the LP relaxation, has a totally unimodular matrix. In 2006, he also proposed the results above including network property of the LP matrix.
Contributed relevant AMPL/GMPL model files to the open-source solver called GNU Linear Programming Kit (GLPK) in 2017, and merged since version 4.64 [1].

1 Introduction

Several models were constructed for generating electric power plants' schedules. They are quite varied and could be classified by several criteria, for example:

- model categories (LP, MILP, SP...)
- solution generating methods, selected algorithms etc.
- objective types and count
- objectives' perspective (economic, environmental, safety...)
- included (or ignored) constraints (network topology, spatial or temporal dependencies, economic, technical...)

For a general model with one objective function, the following optimization model stands:

$$\min \mathbf{C}(\mathbf{x}(t)) \tag{1}$$

$$\mathbf{x}(t) \in \mathbf{F} \tag{2}$$

It is assumed without loss of generosity that the objective function $\mathbf{C}$ is to be minimized. The $\mathbf{F}$ set of functions is the so-called feasibility region. The $\mathbf{x}(t)$ denotes a vector function, which maps time (t) to $\mathbb{R}^n$, where n is the total number of plant units to schedule. Basically $\mathbf{x}(t)$ is composed of $x_i(t) : \mathbb{R} \to \mathbb{R}$ functions, and each of them are piecewise analytical, and the dimension of $R_{f_i(x)}$ is in megawatt (MW) $\forall i \in [1, n] \cap \mathbb{Z}$.

1.1 Evolution of the model

Related models in this paper had been tested and/or used in Hungary before 2000[2][3] and around 2004-2008 at the balancing group department at MVM Rt. which was the Balancing Group Responsibe at that time. The latter title means a conductor-like role, and defines several responsibilities and tasks, one of them is to create and commit a daily electric power plant schedule on the preceeding day to the Transmission System Operator.

After 2000, all of the used models become more and less unusable due to several reasons including but not limited to: the increasing number of production units and consumers and the decreasing of time interval resolution of a schedule; the latter property will be defined later in this paper.

2 The model

An affordable assumption is that a schedule is determined by the measured generation or consumption within enough dense time intervals. According to the laws, contracts and regulations, it was 15 minutes around 2004 for a daily schedule, changed from 1 hours some years earlier. Due to the denser timescale, former models fail to work and the change produced several technical constraints.

2.1 Further allowed assumptions

According to the model composed by expressions (1),(2), the following assumptions had been made:

- The objective function is a sum composed of independent $\mathbb{R} \to \mathbb{R}$ linear mappings
- Network transmission system topology is contractive with no constraints - implying that the overall sum of input and output energy must be 0 in every moment

So the cost function become an ordinary inner product. Spinning reserves, resource barriers like reservoir levels, fuel etc. and all start-up and shut-down scenarios are ignored, the reason: other planning steps ensured most of them, keeping reliability. Some constraints had remained: the conservative property of the network - enforcing the power balance, signed performance bounds on consumers and units, and a new constraint type that is usually ignored in former planning models with hourly time resolution: ramp-down and ramp-up constraints - but only in nearest time units.

2.2 Discretization

Now comes the transformation scheme of objective(3) and constraints:

$$\min \mathbf{C}(\mathbf{x}(t)) \Longrightarrow \inf \int_t c_i(t) \cdot \mathbf{x}_i(t) dt \Longrightarrow \min c^T x \tag{3}$$

$$\mathbf{x}(t) \in \mathbf{F} \Longrightarrow \sum_i \mathbf{x}_i(t) = 0 \Longrightarrow \forall t : \sum_i \mathbf{x}_{i,t} = 0 \tag{4}$$

$$\mathbf{x}(t) \in \mathbf{F} \Longrightarrow \mathbf{x}(t) \in [\mathbf{b}_{LO}(t), \mathbf{b}_{UP}(t)] \Longrightarrow b_{LO,i,t} \leq \mathbf{x}_{i,t} \leq b_{UP,i,t} \tag{5}$$

$$\mathbf{x}(t) \in \mathbf{F} \Longrightarrow \tfrac{d}{dt}\mathbf{x}(t) \in [\mathbf{g}_{LO}(t), \mathbf{g}_{UP}(t)] \Longrightarrow g_{LO,i,t} \leq \Delta\mathbf{x}_{i,t} \leq g_{UP,i,t} \tag{6}$$

where $\Delta\mathbf{x}_{i,t} \stackrel{\text{def}}{=} \mathbf{x}_{i,t} - \mathbf{x}_{i,t-1} \wedge t > 0$, and $x \in \mathbb{R}^{P \times T}$, where P is the set of units and $T \stackrel{\text{def}}{=} \mathbb{Z} \cap [0, |T| - 1]$ is the set of discrete time interval endpoints. $x_{i,t}$ is the only variable in the model.

The above linear constraints defined by the rightmost parts of (4),(5) had been defined in [4] as second and fourth inequalities, respectively, but with an important remark: in the new model, it is assumed that the daily per time demand based on a very precise estimation in practice can be defined as a virtual consumer, with the same upper and lower performance bounds in (5) and without "gradient" barriers (no (6)).

There is only one thing to finish our model: determining the variable types. If we allow arbitrary types other than continuous or integers, the problem might contain the Subset-sum problem proving that it is NP-complete as a feasibility problem.

Lemma 1 Let use constraints (4), (5), let $|T| = 1$, and also let $x_{i,t} \in \{0, a_i\}\, \forall i \in \mathbb{Z} \cap [2..|P|]$ and let $x_{1,t} = -b$; the latter variable denotes the demand, the virtual consumer. With these restrictions, this model is equivalent with the Subset sum problem.

Proof Trivial. Let $a_i > 0 (i > 1)$ be given positive numbers and the sum is $b > 0$. Q.E.D.

Corollary The Start-up-Shut-down problem, where temporary shutdowns and/or unit starts are to schedule - is NP-hard, the proof is exactly the same. The same applies when there are thermal power plants and they have two different minima and each unit cannot work with performance values between them.

When there is a fully linear continuous model, but with each unit has exactly one optional choice when it can ramp up higher once a day, the Subset-sum problem can be still embedded with almost the same method. If only continuous variable types are allowed, the model can be solved in polynomial time as a pure linear programming model is in P, proven by Khachiyan[5]. But if variable types are restricted to integers, it is still not really a restriction due to the network property of the model's matrix, which will be proven below. The author gave a proof for it in his thesis [6]. The

3 The model's matrix

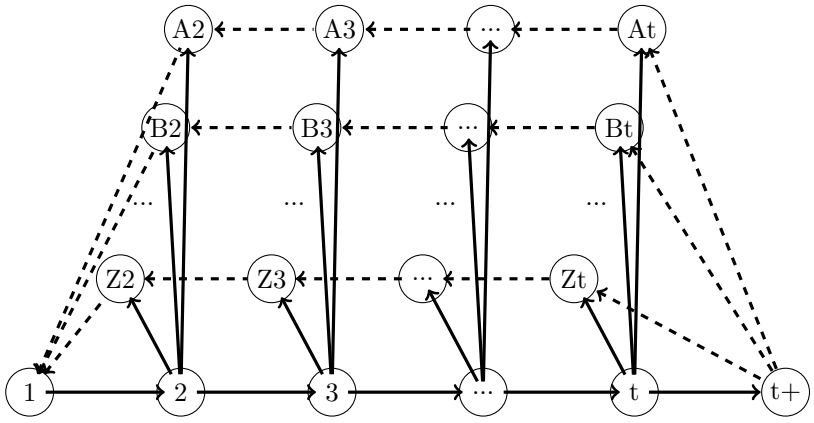
The main result of this paper is that the model's matrix is a network matrix. To prove this, it is enough to eliminate redundant rows and columns, and for the reduced parts a corresponding network graph and matchings between edges and matrix parts will be given, therefore Tutte-Gerards'[7] and Kotnyek's[9] appropriate matrix characterization theorems are applicable. After recognizing network property, Hoffman-Kruskal[8] theorem ensures that most common simplex LP algorithms will produce an integral solution at least on such a compact nonempty feasible set, and also some of them are even strongly polynomial with slight or no modifications; also referenced in PhD thesis[9].

The constraints in (5) can be omitted, because each row has at most 1 nonzero element, thus network property is invariant to this operation. The reduced M matrix has the following form after removing duplicates and normed with optional negations, where $M \in \mathbb{R}^{(|T|-1)\cdot(|P|+1)+1\times|P|\cdot|T|}$, $I \in \mathbb{R}^{|T|\times|T|}$ and $\Delta \in \mathbb{R}^{(|T|-1)\times|T|}$:

$$M = \begin{pmatrix} I & I & \cdots & I \\ \Delta & & & \\ & \Delta & & \\ & & \ddots & \\ & & & \Delta \end{pmatrix} \qquad I_{i,j} = \begin{cases} 1 \text{ if } i = j \\ 0 \text{ if } i \neq j \end{cases}$$

$$\Delta_{i,j} = \begin{cases} -1 \text{ if } i = j \\ 1 \quad \text{if } i = j - 1 \\ 0 \quad \text{otherwise} \end{cases} \tag{7}$$

In the following figure, there is a corresponding network graph. The spanning tree's edges are denoted with continuous, the non-tree edges are dashed. Every temporary addition of a dashed edge creates an exactly 3 or 4-length circle, in which the dashed neigbouring edges generate a Δ column's values, and the continuous edges with only numbered nodes are assigned to a balance-role element in the upper joint identity matrix-part of M. The inner nodes of a directed walks on dashed edges are matched to units, the numbered nodes originated from time interval endpoints. Extendable both ways: by number of dashed paths or length of paths. ✠



References

1. Makhorin A. O., GLPK 4.64 release information *http://lists.gnu.org/archive/html/info-gnu/2017-12/msg00002.html GNU official announcements mailing list (Dec 2017)*
2. Deák, I., Hoffer, J., Mayer, J., Németh, A., Potecz, B., Prékopa, A., Strazicky, B. Optimal daily scheduling of the electricity production in Hungary, *Operations Research in Progress, ed. G.Feichtinger, P.Kall, Wien, Austria, 1982, 103-114.*
3. Deák, I., Hoffer, J., Mayer, J., Németh, A., Potecz, B., Prékopa, A., Strazicky, B., Nagyméretű, vegyesváltozós, matematikai modell termikus villamosenergia-rendszer rövidtávú, optimális menetrendjének meghatározására hálózati feltételek figyelembevételével, *Alkalmazott Mat. Lapok 9 (1983) 221-337.*
4. Shahidehpour S. M., Tong S. K. A scheduling model for the daily operation of an electric power system *Applied Mathematical Modelling Volume 16, Issue 5, (May 1992) 226-244.*
5. Khachiyan L. G. A polynomial algorithm in linear programming, *Soviet Mathematics doklady, 20, 191-194 (1979)*
6. Naszvadi P. Áramelosztás operációkutatási elmélete, modelljei és megoldási módszerei, *Math. MSc. thesis, ELTE, Faculty of Nat. Sc., (2007)*
7. Gerards, A. M. H. A short proof of Tutte's characterization of totally unimodular matrices, *Linear Algebra and its Applications, 114/115: 207-212, (1989)*
8. Hoffman A. J., Kruskal J. B. Integral Boundary Points of Convex Polyhedra, Linear Inequalities and Related Systems *(H.W. Kuhn and A.J. Tucker, eds.), Princeton University Press, 1956, pp. 223–246*
9. Kotnyek B. A generalization of totally unimodular and network matrices, *PhD thesis, London School of Economics, (2002)*

Review and comparison of MILP approaches for cyclic scheduling of robotic cells

Ádám Papp, Olivér Ősz, and Máté Hegyháti

Széchenyi István University, Győr, Hungary
adam.papp.work@gmail.com

Abstract. Automation is a growing trend in modern manufacturing processes, and robotic cells are widely applied for that purpose in various industrial systems. Such cells consist of multiple production machines, material handling robot arms, and input/output buffers, are mainly used in mass production environments where demands are stable. In this scenario, it is common to address long-term production planning in a cyclic manner. This means that a short-term schedule - a cycle - is determined, which will be repeated periodically. In cyclic scheduling of robotic cells, the goal is to determine the sequence of robot movements and machine operations for the production cycle, which maximizes the profit divided by cycle time, while satisfying the practical feasibility constraints. This ratio inherently introduce an undesirable non-linearity, therefore, the problem is often decomposed to a series of subproblems, where the goal is to minimize the cycle time for a given quantity of parts.
This work aims to review the recent advancements in solving the cyclic scheduling of robotic cells with Integer Programming techniques. Problem classes were identified based on the most common parameters, such as production topology and allowed waiting times for intermediates. Selected MILP models were implemented and compared based on their modeling capabilities and solution performances.

Keywords: cyclic scheduling · robotic cell · mixed-integer linear programming

1 Introduction and literature review

Automation plays a more important role in the production industry nowadays, than ever before. Companies use computer controlled machines and systems for higher productivity, and to keep up with the increasing demands of the market. Achieving this goal requires appropriate modeling and scheduling of these machines, that was widely investigated in the literature of various fields, such as painting[3], wafer fabrication[4], assembly[5], or material handling[6].

In this work, the focus is on the scheduling of industrial manufacturing cells, in which the material handling is accomplished by automated robotic arms. These production cells may consist of several identical[9] or different[1] production machines, and input and output buffers which can be realized as conveyors

or storage buffers, often with unlimited capacity. Material handling can be accomplished by one[8] or several[2] automated robotic arms.

In case of stable demand, production planning is mostly addressed with *cyclic scheduling*. The aim in this methodology is to determine a short-term schedule - a *cycle* - which is a finite sequence of robot movements that can be repeated indefinitely. The usual objective is to maximize profit, or equivalently, to minimize the *mean cycle time*, i.e., the length of a cycle divided by the number of finished parts within a single period.

The scheduling of robotic manufacturing cells was first addressed in the literature by Sethi et al. [1]. From the modeling point of view, the scheduling of these systems show a lot of similarity with, and are often equivalent to the the so-called *Hoist Scheduling Problems*[6, 7, 10, 13, 14], where material handling is accomplished by an automated hoist instead of a robotic arm.

Over the years, various solution methods were developed and published for these problems. Abd et al.[5] used a Fuzzy logic approach for the online case. Zhou and Li[8] focused on a Tabu search heuristic algorithm. Batur et al.[9] modeled the main problem class as a TSP, and built a Simulated Annealing algorithm. Al-Ahmari[15] solved the base problem with Petri-nets.

The most widely used method in literature, however, is mixed integer linear programming (MILP) (see, e.g. [6, 7, 10, 13]). Thus, in this paper we focus on the capabilities and performance of the state-of-the-art MILP formulations found in the literature.

The rest of the paper is structured as follows: Section 2 presents the basic terminology of the field and highlights the most important parameters of robotic cells, providing a classification of related scheduling problems. Section 3 presents selected recent MILP models, and the comparison of their performance on variously scaled literature examples. Finally, Section 4 concludes the results of our investigation.

2 Classification of robotic cells

Robotic cells come in a great variety, and consequently, a lot of different problem classes can be identified. In this section, several distinguishing properties of such systems are mentioned, that are often used to differentiate between the capabilities of proposed models.

Type of the robotic arm: a *single gripper* can handle only one production part[12, 7, 10], while a *dual gripper* may carry two production parts simultaneously[16].

Product diversity: a *single part type cell* considers only one type of product and production recipe[14, 8, 2, 16]. In contrast, several products with different production recipes are produced in a *multiple part type cell*[10, 15, 9, 5].

Production path: In case of *flow shop* problems, the product parts must go through all of the machines in the same order[10, 6, 12, 7, 14], while the production paths may differ for *job shop* problems[5, 9, 13], and be arbitrary in the case of *open shop* problems.

Waiting policy: In the case of *no-wait* policy, the robot must transfer the part immediately to the next stage after its processing is finished, to avoid damages[2, 12, 1]. In *interval cells*, the processing times are given with time windows, the lower bound being the minimum required time, and the upper bound with the maximum allowed waiting time included[13, 7]. When there are no constraints for waiting, and the product can remain in the machine after its completion for unlimited time without getting damaged, the cell is termed as *free-pickup cell*[10].

An important parameter of a cycle is the amount of products produced by it. Che and Chu[2] introduced the term *r-degree cycle*, referring to a cycle where the number of parts entering and leaving the cell is equal to r. A different term, *k-unit cycle*[11] is used for flow shop problems, which means that all of the machines are visited by the robotic arm exactly k times. In case of different parts, an *MPS(Minimal Part Set)* is often defined, which is a minimal set of production parts with their relative proportion calculated from product demands[12].

3 Comparison of MILP models

There are many approaches published in the literature for robotic cell scheduling problems. Comparing them is not evident, as they usually address different classes, and (at least supposedly) work best on problems, that exploit all the features of that class, but nothing else. Thus, comparing models simply on the intersection of their capabilities cannot be counted as a fair comparison, though it still provides valuable information. Due to space limitations, only some of the results are shared here, focusing on single robot cells with single gripper arms. Subsection 3.1 provides necessary information about the selected models, the literature example used for the comparison, and some details about the test environment.

3.1 Models, problem and test environment

The following 4 models with different general model structures and publication dates were selected from the literature for this comparison:

Z03 An older, simple precedence based model from Zhou & Li (2003) [13]
L14 A more recent precedence based model from Li & Fung (2014) [14]
G18 A recent hybrid slot/precedence based model by Gultekin et al. (2018) [10]
F18 A recent precedence based model by Feng et al. (2018) [7]

Table 1 summarizes the capabilities of the models. Additionally, except for **G18**, all the models can address general transfer times and different movement types for empty movements of the robot, i.e., when it is not carrying any product.

The test instances were generated based on a well-known 12-stage literature example by Phillips and Unger[6]. For the different test cases described later, the

Table 1. Comparison of capabilities of the selected models

Model	Multiple parts	Interval pickup	r-degree cycle	Production path
Z03		×		flow-shop
L14		×	×	flow-shop
G18	×		×	flow-shop
F18	×	×	×	job-shop

original problem was modified accordingly, e.g., additional stages were generated, or specific parameters disregarded, as necessary.

Each model has been reimplemented based on the original publication[1], and compared on the same machine with Intel i5-7200U 3 GHz processor, 8 GB memory, and solved with Gurobi Optimizer 8.0.1 (with standard settings).

3.2 Test case 1: interval pickup & 1-degree cycles

The "intersection" of the capabilities of every model is a very simple problem class with interval pickup policy and 1-degree cycles. For the tests in this subsection, **G18** was excluded to include interval pickup policy. All of the test instances are 1-degree cycle problems, as **Z03** cannot handle r-degree cycles. To scale the problems, additional stages are introduced. The results are shown in Table 2.

Table 2. Comparison of **F18**, **L14**, and **Z03** on 1-degree, interval pickup problems

Stages	12	16	20	24
Model	Solution time			
F18	0.24 s	1.1 s	1.42 s	2.66 s
L14	0.34 s	0.93 s	2.84 s	4.45 s
Z03	0.26 s	1.16 s	2.85 s	2.96 s

All of the three models performed well on this specific problem class, even the more general ones. As there is no significant difference between them, there is no reason to select the more limiting **Z03**. Between **F18** and **L14**, the former performed usually better, and it is also a bit more general model, thus, **F18** could be considered as superior among the three for 1-degree cycles.

3.3 Test case 2: free pickup & r-degree cycles

Problems become more difficult as the number of batches is increased, i.e., r-degree cycles are considered. **Z03** must be excluded from these tests, and it is

[1] Where the original source was ambiguous or incomplete, the authors filled the gaps to their best knowledge.

important to highlight, that except for **L14** the remaining two models can address products with different types as well. For the tests below, a small change was made on the original test problem: the movement times are considered additive and the same for loaded/empty movements, as **G18** cannot handle the generalization of those. The time limit for all of the tests were set to 200 seconds, and the problem was scaled by increasing the number of cycles.

All of the models could find the optimal solution within the time limit for 1 cycle, though **G18**(13.21 s) was significantly slower than **F18**(0.23 s) or **L14**(0.46 s). For larger degree cycles, however, none of the models were able to find the optimal solution within 200 seconds. The best solutions found the optimality gaps are shown in Table 3.

Table 3. Comparison of **F18**, **L14**, and **G18** on r-degree, free pickup problems

	2-degree		3-degree		4-degree	
Model	Best obj	Gap	Best obj	Gap	Best obj	Gap
F18	2147 s	29.6%	3492 s	58.4%	4716 s	73.5%
L14	2177 s	30.4%	3406 s	50.4%	4650 s	62.2%
G18	3208 s	33.7%	-	∞	-	∞

It is evident that **G18** is a much slower model than the other two. To be fair, Gultekin et al.[10] also proposed a hybrid metaheuristic solution algorithm beside the MILP model. Further tests suggested, that the free pickup policy plays an important role in hanging the models for more degree cycles. To put this assumption to the test, **L14** and **F18** was tested on 2, 3 and 4 degree cycles, while gradually extending the pickup intervals. These tests assured that the lengths of these intervals is a key factor, and has an exponential effect on the computational time for both models. Omitting the detailed results, the maximum pickup interval per processing time ratios that the models could solve in time, were 1500%, 700%, and 500% for 2, 3, and 4 degree cycles respectively.

4 Conclusion

Robotic cell scheduling is becoming a more and more relevant topic for the industry. A wide range of problems can be identified, for which plenty of solution approaches has been published in the literature. In this work, four selected models were tested for different problem classes, and the recent precedence based model of Feng et. al.[7] turned out to be the most efficient for most of the cases, while still being one of the most general among the four. Additionally, it was identified, and verified, that pickup policy, and the ratio between the processing and waiting times has a huge impact on the computational time.

5 Acknowledgement

This work was financially supported by the ÚNKP-18-1 New National Excellence Program of the Ministry of Human Capacities of Hungary.

References

1. Sethi, S. P., Sriskandarajah, C., Sorger, G., Blazewicz, J., and Kubiak, W. (1992). Sequencing of parts and robot moves in a robotic cell. International Journal of Flexible Manufacturing Systems, 4(3-4), 331-358.
2. Che, A., and Chu, C. (2009). Multi-degree cyclic scheduling of a no-wait robotic cell with multiple robots. European Journal of Operational Research, 199(1), 77-88.
3. Kolakowska, E., Smith, S. F., and Kristiansen, M. (2014). Constraint optimization model of a scheduling problem for a robotic arm in automatic systems. Robotics and Autonomous Systems, 62(2), 267-280.
4. Lee, T. E. (2008). A review of scheduling theory and methods for semiconductor manufacturing cluster tools. In Proceedings of the 40th conference on winter simulation (pp. 2127-2135). Winter Simulation Conference.
5. Abd, K., Abhary, K., and Marian, R. (2016). Multi-objective optimisation of dynamic scheduling in robotic flexible assembly cells via fuzzy-based Taguchi approach. Computers and Industrial Engineering, 99, 250-259.
6. Phillips, L. W., and Unger, P. S. (1976). Mathematical programming solution of a hoist scheduling program. AIIE transactions, 8(2), 219-225.
7. Feng, J., Chu, C., and Che, A. (2018). Cyclic jobshop hoist scheduling with multi-capacity reentrant tanks and time-window constraints. Computers and Industrial Engineering.
8. Zhou, B. H., and Li, M. (2017). Scheduling method of robotic cells with robot-collaborated process and residency constraints. International Journal of Computer Integrated Manufacturing, 30(11), 1164-1178.
9. Batur, G. D., Erol, S., and Karasan, O. E. (2016). Robot move sequence determining and multiple part-type scheduling in hybrid flexible flow shop robotic cells. Computers and Industrial Engineering, 100, 72-87.
10. Gultekin, H., Coban, B., and Akhlaghi, V. E. (2018). Cyclic scheduling of parts and robot moves in m-machine robotic cells. Computers and Operations Research, 90, 161-172.
11. Dawande, M., Geismar, H. N., Sethi, S. P., and Sriskandarajah, C. (2005). Sequencing and scheduling in robotic cells: Recent developments. Journal of Scheduling, 8(5), 387-426.
12. Agnetis, A. (2000). Scheduling no-wait robotic cells with two and three machines. European Journal of Operational Research, 123(2), 303-314.
13. Zhou, Z., and Li, L. (2003). Single hoist cyclic scheduling with multiple tanks: a material handling solution. Computers and Operations Research, 30(6), 811-819.
14. Li, X., and Fung, R. Y. (2014). A mixed integer linear programming solution for single hoist multi-degree cyclic scheduling with reentrance. Engineering Optimization, 46(5), 704-723.
15. Al-Ahmari, A. (2016). Optimal robotic cell scheduling with controllers using mathematically based timed Petri nets. Information Sciences, 329, 638-648.
16. Jung, K. S., Geismar, H. N., Pinedo, M., and Sriskandarajah, C. (2015). Approximations to optimal sequences in single-gripper and dual-gripper robotic cells with circular layouts. IIE Transactions, 47(6), 634-652.

New trends in interior-point algorithms

Petra Renáta Rigó[1,2]

[1] Budapest University of Technology and Economics, Budapest, Hungary
[2] Babeş-Bolyai University, Cluj-Napoca, Romania
takacsp@math.bme.hu

Abstract. In this paper we deal with predictor-corrector interior-point algorithms (PC IPAs) for solving $P_*(\kappa)$-linear complementarity problems over Cartesian product of symmetric cones (Cartesian $P_*(\kappa)$-SCLCPs). We present the algebraic equivalent technique (AET) for determining search directions in case of interior-point algorithms (IPAs). After that, we give a unification of the scaled systems and Newton-systems in case of the PC IPAs. By using this general framework, new PC IPAs for Cartesian $P_*(\kappa)$-SCLCPs can be introduced.

Keywords: predictor-corrector interior-point algorithm · $P_*(\kappa)$-linear complementarity problem over Cartesian product of symmetric cones · algebraically equivalent technique

1 Introduction

Linear complementarity problems (LCPs) over symmetric cones have been extensively studied in the recent years. The LCPs belong to the class of NP-complete problems. In spite of this fact, Kojima et al. [5] showed that assuming that the problem's matrix has $P_*(\kappa)$-property, the IPAs solving these kind of LCPs usually have polynomial complexity in the handicap of the problem's matrix, the size of the problem and the bitsize of the data. The symmetric cone LCP (SCLCP) contains LCP, semidefinite linear complementarity problem (SDLCP) and second-order cone linear complementarity problem (SOCLCP), as special cases. The Cartesian $P_*(\kappa)$-property was first introduced by Luo and Xiu [6]. They also proved the existence and uniqueness of the central path for the Cartesian $P_*(\kappa)$-SCLCPs.

In the following we present some main aspects of the theory of the Euclidean Jordan algebras and symmetric cones [4].

Let V be an n-dimensional vector space over $\mathbb{R}$ and let us consider the following bilinear map: $\circ : (x, y) \to x \circ y \in V$. Then, $(V, \circ)$ is called a Jordan algebra iff for all $x, y \in V$ $x \circ y = y \circ x$ and $x \circ (x^2 \circ y) = x^2 \circ (x \circ y)$, where $x^2 = x \circ x$. A Jordan algebra is said to be Euclidean if there exists an inner product such that $\langle x \circ y, z\rangle = \langle x, y \circ z\rangle$. Throughout the paper it will be assumed that $(V, \circ)$ is an Euclidean Jordan algebra and it will be denoted it by V. For an element $x \in V$, let us introduce the linear operator $L(x) : V \to V$ such that for every $y \in V$, $x \circ y = L(x)y$. For each $x \in V$, we also define the quadratic representation of V: $P(x) := 2L(x)^2 - L(x^2)$, where $L(x)^2 := L(x)L(x)$.

Consider r as the smallest integer such that the set $\{e, x, \ldots, x^r\}$ is linearly dependent, for all $x \in V$. Then, r is named the degree of the element x and it is denoted as $deg(x)$. The rank of V, denoted as $rank(V)$, is the largest $deg(x)$, for all $x \in V$. Throughout the paper we will assume that V is an Euclidean Jordan algebra with rank r.

An idempotent element c is said to be primitive if it is nonzero and can not be expressed by sum of two other nonzero idempotents. A set of primitive idempotents $\{c_1, \ldots, c_r\}$ is called Jordan frame iff for all primitive idempotents c_i the following hold: $c_i \circ c_j = 0, i \neq j$ and $\sum_{i=1}^{r} c_i = e$.

Theorem 1. *(Spectral decomposition, Theorem III.1.2 in [4]) Let $x \in V$. Then, there exists a Jordan frame $\{c_1, \ldots, c_r\}$ and the real numbers $\lambda_1(x), \ldots, \lambda_r(x)$ such that*

$$x = \sum_{i=1}^{r} \lambda_i(x) c_i. \tag{1}$$

The numbers $\lambda_i(x)$ are the eigenvalues of x.

Consider the vector-valued function using the function φ, which is a real-valued univariate function differentiable on the interval $(0, +\infty)$ such that $\varphi'(t) > 0$ for all $t > 0$. Let $x \in V$ be a vector with the spectral decomposition given in (1). The vector-valued function φ is defined in the following way: $\varphi(x) := \varphi(\lambda_1(x))c_1 + \ldots + \varphi(\lambda_r(x))c_r$.
Let us introduce the following notation:

$$\langle x, s \rangle := tr(x \circ s), \tag{2}$$

where $x, s \in V$. The Frobenius norm, $\|\cdot\|_F$, which is induced by the trace inner product is defined by $\|x\|_F := \sqrt{\langle x, x \rangle}$.

We define the cone of squares $K := \{x^2 : x \in V\}$. This cone is a symmetric cone, i.e. it is self-dual and homogeneous.

The following hold: $x \in K \Leftrightarrow \lambda_i(x) \geq 0,\ i = 1, \ldots, r$ and $x \in \operatorname{int} K \Leftrightarrow \lambda_i(x) > 0,\ i = 1, \ldots, r$. Moreover, let us introduce the following notations: $x \succeq_K 0 \quad \Leftrightarrow \quad x \in K$ and $\quad x \succ_K 0 \quad \Leftrightarrow \quad x \in \operatorname{int} K$.
In the following section we present the Cartesian $P_*(\kappa)$-SCLCP problem.

2 The Cartesian $P_*(\kappa)$-SCLCP

Let us consider the Cartesian product space $V = V_1 \times V_2 \times \ldots \times V_m$ with its cone of squares $K = K_1 \times K_2 \times \ldots \times K_m$. Here, each V_i is a Euclidean Jordan algebra and each K_i is the associated cone of squares of V_i. For each $z = \left(z^{(1)}, z^{(2)}, \ldots, z^{(m)}\right)^T \in V$, where $z^{(i)} \in V_i$ consider the following:

$$Tr(z) = \sum_{i=1}^{m} Tr\left(z^{(i)}\right), \quad det(z) = \prod_{i=1}^{m} det\left(z^{(i)}\right),$$

$$\lambda_{min}(z) = \min_{1\le i\le m}\left\{\lambda_{min}\left(z^{(i)}\right)\right\}, \quad \lambda_{max}(z) = \max_{1\le i\le m}\left\{\lambda_{max}\left(z^{(i)}\right)\right\}.$$

The Frobenius norm of an element of V is defined as

$$\|x\|_F = \left(\sum_{i=1}^{m}\left\|x^{(i)}\right\|_F^2\right)^{\frac{1}{2}}.$$

Furthermore, we introduce the following notations

$$L(x) = diag\left(L\left(x^{(1)}\right), L\left(x^{(2)}\right), \ldots, L\left(x^{(m)}\right)\right),$$

$$P(x) = diag\left(P\left(x^{(1)}\right), P\left(x^{(2)}\right), \ldots, P\left(x^{(m)}\right)\right).$$

For any

$$x = \left(x^{(1)}, x^{(2)}, \ldots, x^{(m)}\right)^T \in V \text{ and } s = \left(s^{(1)}, s^{(2)}, \ldots, s^{(m)}\right)^T \in V$$

we will use

$$x \circ s = \left(x^{(1)} \circ s^{(1)}, x^{(2)} \circ s^{(2)}, \ldots, x^{(m)} \circ s^{(m)}\right)^T, \quad \langle x, s\rangle = \sum_{i=1}^{m}\left\langle x^{(i)}, s^{(i)}\right\rangle.$$

The Cartesian SCLCP has the following form. We should find a vector pair $(x, s) \in V \times V$ such that

$$-Mx + s = q, \quad \langle x, s\rangle = 0, \quad x \succeq_K 0, \quad s \succeq_K 0, \qquad (SCLCP)$$

where K is the symmetric cone of the Cartesian product space V, $M : V \to V$ is linear operator and $q \in V$.

Let $\kappa \ge 0$ be a constant. Then, we say that M has $P_*(\kappa)$-property, if for all $(x, s) \in V \times V$

$$-Mx + s = 0 \text{ implies } \langle x, s\rangle \ge -4\kappa \sum_{i\in I_+}\left\langle x^{(i)}, s^{(i)}\right\rangle,$$

where $I_+ = \left\{i : \left\langle x^{(i)}, s^{(i)}\right\rangle > 0\right\}$. In this case, we call problem $(SCLCP)$ Cartesian $P_*(\kappa)$-SCLCP.

Without loss of generality we can assume that the *interior-point condition* (IPC) holds. This means that there exists (x^0, s^0) so that:

$$\begin{aligned} -Mx^0 + s^0 &= q, \qquad x^0 \succ_K 0, \\ \langle x^0, s^0\rangle &= 0, \qquad s^0 \succ_K 0. \end{aligned} \qquad (IPC)$$

The central path can be characterized by the following system:

$$\begin{aligned} -Mx + s &= q, \qquad x \succeq_K 0, \\ x \circ s &= \mu e, \qquad s \succeq_K 0. \end{aligned} \qquad (3)$$

where $\mu > 0$.

Let us consider the subclass of Monteiro-Zhang family of search directions:
$C(x, s) = \left\{u | u \text{ is invertible and } L(P(u)x)L(P(u)^{-1}s) = L(P(u)^{-1}s)L(P(u)x)\right\}$.

Lemma 1. *(Lemma 28 in [8]) Let $u \in \operatorname{int} K$. Then,*

$$x \circ s = \mu e \quad \Leftrightarrow \quad P(u)x \circ P(u)^{-1}s = \mu e.$$

Choosing $u \in C(x,s)$, denoting $\tilde{M} = MP(u)^{-1}$ and $\tilde{I} = IP(u)$ and using Lemma 1, we can rewrite system (3) in the following way:

$$\begin{aligned} -\tilde{M}P(u)x + \tilde{I}P(u)^{-1}s = q, \qquad & P(u)x \succeq_K 0, \\ P(u)x \circ P(u)^{-1}s = \mu e, \qquad & P(u)^{-1}s \succeq_K 0. \end{aligned} \tag{4}$$

Luo and Xiu [6] and Asadi et al [1] proved that if the IPC holds, then system (4) has unique solution for each $\mu > 0$.

3 The AET technique in case of IPAs for Cartesian $P_*(\kappa)$-SCLCPs

In this section we present the AET technique for determining search directions proposed by Darvay [2] for linear optimization and generalized by Wang and Bai [9] for symmetric optimization problems. Consider the φ vector-valued function, which is induced by the real-valued univariate function $\varphi : (0, +\infty) \to \mathbb{R}$. Using this, system (4) can be written in the following way:

$$\begin{aligned} -\tilde{M}P(u)x + \tilde{I}P(u)^{-1}s = q, \qquad & P(u)x \succeq_K 0, \\ \varphi\left(\frac{P(u)x \circ P(u)^{-1}s}{\mu}\right) = \varphi(e), \qquad & P(u)^{-1}s \succeq_K 0. \end{aligned} \tag{5}$$

Using Newton's method and the technique presented by Wang and Bai [9] we define search directions. For the strictly feasible $x \in \operatorname{int} K$ and $s \in \operatorname{int} K$ our aim is to find the search directions $(\Delta x, \Delta s)$ so that

$$\begin{aligned} -\tilde{M}P(u)\Delta x + \tilde{I}P(u)^{-1}\Delta s = 0, \qquad & P(u)x \succeq_K 0, \\ P(u)x \circ P(u)^{-1}\Delta s + P(u)^{-1}s \circ P(u)\Delta x = a_\varphi, \qquad & P(u)^{-1}s \succeq_K 0, \end{aligned} \tag{6}$$

where

$$a_\varphi = \mu \left(\varphi'\left(\frac{P(u)x \circ P(u)^{-1}s}{\mu}\right)\right)^{-1} \circ \left(\varphi(e) - \varphi\left(\frac{P(u)x \circ P(u)^{-1}s}{\mu}\right)\right).$$

Using different φ functions we have different $\mathbf{a}_\varphi$ values.

We will consider the NT-scaling scheme (Nesterov and Todd [7]). Let $u = w^{-\frac{1}{2}}$, where w is the NT-scaling point of x and s. Consider the notations:

$$v := \frac{P(w)^{-\frac{1}{2}}x}{\sqrt{\mu}} \left[= \frac{P(w)^{\frac{1}{2}}s}{\sqrt{\mu}}\right] \tag{7}$$

and

$$d_x := \frac{P(w)^{-\frac{1}{2}}\Delta x}{\sqrt{\mu}}, \quad d_s := \frac{P(w)^{\frac{1}{2}}\Delta s}{\sqrt{\mu}}, \quad \bar{M} = P(w)^{\frac{1}{2}} M P(w)^{\frac{1}{2}}. \tag{8}$$

Using the notations given is (7) and (8), we obtain the scaled system:

$$\begin{aligned} -\bar{M}d_x + d_s &= 0, \\ d_x + d_s &= p_v, \end{aligned} \tag{9}$$

where

$$p_v = v^{-1} \circ (\varphi'(v \circ v))^{-1} \circ (\varphi(e) - \varphi(v \circ v)).$$

We should define a proximity measure to the central path. This can be done in the following way: $\delta(v) = \delta(x, s, \mu) := \frac{\|p_v\|_F}{2}$. Furthermore, we define the neighbourhoods of the central path by using the τ threshold parameter:

$$\mathcal{N}(\tau, \mu) := \{(x, s) \in V \times V : -Mx + s = q, \quad x \succeq_K 0, \quad s \succeq_K 0 : \quad \delta(x, s, \mu) \leq \tau\}.$$

In Section 4 we generalize a method for determining the scaled predictor and corrector systems in case of the PC IPAs introduced in Darvay et al. [3].

4 General framework for determining search directions in case of PC IPAs

The PC IPAs use two kind of steps in each iteration, a predictor and several corrector steps. After a predictor step a certain amount of retirement from the central path is allowed. The aim of the corrector steps is to return in the τ-neighbourhood of the central path.

In this subsection we generalize the method introduced by Darvay et al. [3] in order to determine the scaled predictor and scaled corrector systems in case of PC IPAs. Note that system (9) coincides with the *scaled corrector system*.

In order to obtain the *scaled predictor system*, we should decompose a_φ in system (6) in the following way:

$$a_\varphi = h(x, s, \mu) + l(x, s),$$

where h and l are vector-valued functions and $h(x, s, 0) = 0$. After that, set $\mu = 0$ in this decomposition. Hence,

$$\begin{aligned} -\tilde{M}P(u)\Delta x + \tilde{I}P(u)^{-1}\Delta s = 0, \qquad & P(u)x \succeq_K 0, \\ P(u)x \circ P(u)^{-1}\Delta s + P(u)^{-1}s \circ P(u)\Delta x = l(x, s), \qquad & P(u)^{-1}s \succeq_K 0, \end{aligned} \tag{10}$$

We obtain the following scaled predictor system:

$$\begin{aligned} -\bar{M}d_x + d_s &= 0, \\ d_x + d_s &= (\mu v)^{-1} \circ l(x, s), \end{aligned} \tag{11}$$

where $\bar{M} = DMD$.

By using this general framework, we can introduce new PC IPAs for solving Cartesian $P_*(\kappa)$-SCLCPs.

5 Conclusion

In this paper we considered Cartesian $P_*(\kappa)$-SCLCPs. We presented the AET technique for determining search directions in case of IPAs. After that, we proposed a new general framework for defining search directions in case of PC IPAs.

Acknowledgement

This research has been partially supported by the Hungarian Research Fund, OTKA (grant no. NKFIH 125700) and by a grant of Romanian Ministry of Research and Innovation, CNCS - UEFISCDI, project number PN-III-P4-ID-PCE-2016-0190, within PNCDI III. Furthermore, this work has been also partially supported by the Higher Education Excellence Program of the Ministry of Human Capacities in the frame of Artificial Intelligence research area of Budapest University of Technology and Economics (BME FIKP-MI/FM).

References

1. S. Asadi, H. Mansouri, Zs. Darvay, and M. Zangiabadi. On the $P_*(\kappa)$ horizontal linear complementarity problems over Cartesian product of symmetric cones. *Optimization Methods and Software*, 31(2):233–257, 2016.
2. Zs. Darvay. New interior point algorithms in linear programming. *Adv. Model. Optim.*, 5(1):51–92, 2003.
3. Zs. Darvay, T. Illés, J. Povh, and P.R. Rigó. Predictor-corrector interior-point algorithm for sufficient linear complementarity problems based on a new search direction. 2018. submitted.
4. J. Faraut and A. Korányi. *Analysis on Symmetric Cones*. Oxford University Press, New York, 1994.
5. M. Kojima, N. Megiddo, T. Noma, and A. Yoshise. *A Unified Approach to Interior Point Algorithms for Linear Complementarity Problems*, volume 538 of *Lecture Notes in Computer Science*. Springer Verlag, Berlin, Germany, 1991.
6. Z. Luo and N. Xiu. Path-following interior point algorithms for the Cartesian $P_*(\kappa)$-LCP over symmetric cones. *Science in China Series A: Mathematics*, 52(8):1769–1784, 2009.
7. Yu. E. Nesterov and M. J. Todd. Self-scaled barriers and interior-point methods for convex programming. *Math. Oper. Res.*, 22(1):1–42, 1997.
8. S.H. Schmieta and F. Alizadeh. Extension of primal-dual interior point algorithms to symmetric cones. *Math. Program., Ser. A*, 96(3):409–438, 2003.
9. G. Q. Wang and Y. Q. Bai. A new full Nesterov-Todd step primal-dual path-following interior-point algorithm for symmetric optimization. *J. Optim. Theory Appl.*, 154(3):966–985, 2012.

Optimizing data collection: a data-driven approach for sea exploration

Davi Pereira dos Santos[2] and João Pedro Pedroso[1,2]

[1] Faculdade de Ciências, Universidade do Porto, Portugal
[2] INESC TEC, Porto, Portugal
davi.p.santos@inesctec.pt, jpp@fc.up.pt

Abstract. This paper describes an algorithm for a problem arising in sea exploration, where the aim is to schedule the expedition of a ship for collecting information about the resources on the seafloor. The aim of the algorithms is to decide locations where to collect data by probing. This way, after the expedition the information available is optimally enriched. The evaluation of a solution is done by comparing the estimation of the level of the resource on the given surface, which is done by regression using Gaussian processes, with the "true" level. The algorithms comprises elements from combinatorial optimization and from continuous optimization, and is influenced by evolutionary computation.

Keywords: Recognition problems · Tour planning · Orienteering · Surface exploration · Gaussian processes

1 Introduction

This work describes a problem with origins in sea exploration, though similar problems arise in other contexts. The identification of the contents of the seafloor is important in view of a possible exploitation of some of these resources. The aim of this problem is to schedule the journey of a ship for collecting information about the composition of the seafloor. We consider a bounded surface, through which some resource can be found with a given level. This "true value" is initially unknown, except for a limited number of points for which there is previous empirical information.

Optimal expedition planning involves three subproblems, each corresponding to a different phase in the process: assessment, planning and estimation.

Assessment consists of estimating the amount of information that would be conveyed by probing the surface at each point. This is done by means of an indicator function. Previous work assumed that actual information obtained by probing is not usable at the time of planning; here, we assume that after committing to probing at a certain place, the information obtained can immediately be used to change the course of the following decisions (in particular, the set of points used for building the indicator function is dynamically expanded).

Planning, the next phase in the solution process, consists of deciding on the position of points to probe until the end of the expedition; the point to probe next

is the only one to which we commit. The objective is to maximize the overall informational reward obtained, taking into account that the total duration of the trip is limited to a known bound. Hence, online planning involves using the previously available points together with the points newly probed in this trip, in order to decide the location of the next point to probe — though an estimation of the whole remaining trip is necessary for correctly taking this decision.

The third subproblem is estimation, which is related to the final aim of the problem: an estimation of the resource level available at any point on the surface, based on all the information available at the end of the trip. This is done through regression using both the initially available points and those collected during the expedition.

In this work we detail a hybrid algorithm for tackling this problem, including components of combinatorial optimization, machine learning and evolutionary computation. The objective is to carefully plan a data collection expedition that maximizes the information available at the end of the trip. This implies choosing the most profitable probing points, which are part of a ship trip that must respect a time limit. The trip length is determined by means of an integer programming model for orienteering [1]. We propose the estimation of the resource levels using a Gaussian processes model [4]. Points with high variance are initial candidates for probing, but their position will evolve through random distortion towards a solution which minimizes a measure of the variance allover the relevant surface.

Experiments with simulated data show that the proposed method improves the quality of the ship's schedule. We use an error measure involving the discrepancy between the "true value" and the predicted value, estimated in a fine grid over the relevant surface.

2 Method

The main difficulties for designing an algorithm for this problem concern the nonviability of direct evaluation of the objective function at the time of assessing the quality of a solution. This is due to the fact that the objective—*i.e.*, the error between the "true" value of contents and its evaluation through the function that is used to estimated them, along the relevant surface—is not known at the time some point is considered for probing. We deal with this difficulty in several layers.

We assume that the variance at each point is a surrogate for the interest of probing it, in order to obtain a better overall picture. However, assessing the variance just at the probing points was found to be insufficient; we verified that an estimation of the variance allover the surface was indeed necessary for having a meaningful measure of the interested of probing a point. Hence, when in the algorithm we attempt probing at a given point, the variance summed for a set of points representative of the whole surface is used; in practice, we evaluate it at the same points that are used for evaluating the error, by means of the variance estimated by Gaussian processes. This is depicted in Figure 1.

The main method is provided in Algorithm 1. The best kernel setting ϕ is selected according to a 5-fold cross-validation procedure that minimizes the

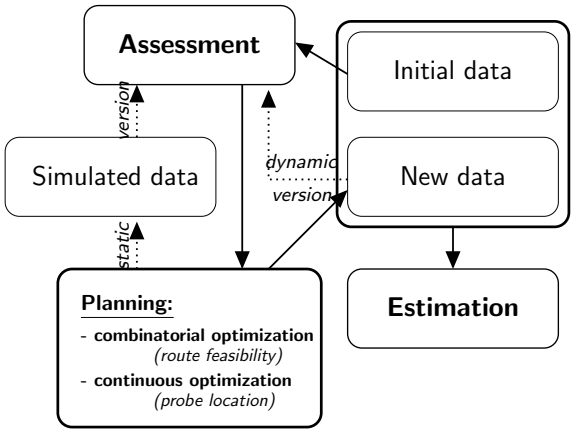


Fig. 1. Information flow and interactions between different parts of the algorithm.

error on the set of previously known data D. 25 kernel settings were used. They were based on the scikit-learn library's RationalQuadratic, RBF and Matern [2]. The parameters *length_scale_bounds* and *alpha_bounds* were set to respect one of the intervals in the set $\{(0.00001, 0.001), (0.001, 0.1), (0.1, 10), (10, 1000), (1000, 100000)\}$. The values for parameter *nu_bounds* were $\{0.5, 1.5, 2.5\}$. A noisy component was added to all kernels with bounds $(1e-5, 1e-2)$. The optimizer was allowed to restart 10 times.

The algorithm can be described as follows. A Gaussian Processes model w (with standard deviation function s) based on kernel setting ϕ is induced over D by function `fit`. While there is computational resource available, i.e. the elapsed time is within the time limit, the algorithm repeats a 4-step sequence:

- **orienteering** - insertion of points that maximize s over a grid G in accordance to the allowed trip costs;
- **disturbance** - one of the trip (a) points ($\boldsymbol{p} \in a$) is randomly chosen to be distorted according to the bivariate normal distribution[3] $\mathcal{N}_2(\boldsymbol{p}, \begin{bmatrix} 0.005 & 0 \\ 0 & 0.005 \end{bmatrix})$;
- **simulation** - the regression function w is applied to the distorted trip (a') to simulate probing values and to induce a new model able to calculate the sum of standard deviation values σ;
- **TSP** - discard of the previous steps results if the distorted trip (a') is not feasible and the unfeasibility is confirmed for the best TSP solution.

The function *cost* and the TSP solver take into account the trip duration and probing time.

[3] We adopted the value 0.005 for an available area S with dimensions 1×1.

function `Main`(D, T, S, E, v, L, K)

$\phi \leftarrow$ best kernel setting according to 5-fold cross-validation on D
$G \leftarrow \{(x_0 + \delta k, y_0 + \delta \ell), k = 1, \dots K, \ell = 1, \dots L\}$ ◁ *grid on S*
$\sigma_{min} \leftarrow \infty$
$\langle w, s \rangle \leftarrow$ `fit`(D, ϕ) ◁ *regression functions: w for prediction; s for standard dev.*
while *elapsed time* $<$ *time limit* **do**
  $a \leftarrow$ `Orienteering`(D, T, S, ϕ, G, a)
  $a' \leftarrow$ `Disturb`(a)
  $\langle \_, s \rangle \leftarrow$ `fit`$(D \cup$ `apply`$(w, a'), \phi)$ ◁ *apply w to a' to simulate probing*
  $\sigma \leftarrow \sum_{\langle x,y \rangle \in G} s(x, y)$
  $r \leftarrow$ TSP solution visiting a'
  if `cost`$(r) < T$ **and** $\sigma < \sigma_{min}$ **then**
    $\sigma_{min} \leftarrow \sigma$
    $a \leftarrow a'$
  end
end
$D' \leftarrow D \cup$ `apply`(v, a) ◁ *start of the testing part*
$\langle w, \_ \rangle \leftarrow$ `fit`(D', ϕ)
$\Delta \leftarrow \sum_{(x,y) \in E} |v(x, y) - w(x, y)|$
return Δ

Algorithm 1: Main procedure. Input: previously known data D, allowed trip costs including duration and probing time T, available area S, testing points E, "true function" evaluator v, grid dimensions $L \times K$. Output: estimated error value over the grid Δ.

function `Orienteering`(D, T, S, ϕ, G, a)

while True do
  $\langle w, s \rangle \leftarrow$ `fit`(D, ϕ)
  $\langle x, y \rangle \leftarrow \arg\max_{\langle x,y \rangle \in G} s(x, y)$ ◁ *return element with maximum s*
  $a' \leftarrow \{\langle x, y \rangle\} \cup a$
  $r \leftarrow$ TSP solution visiting a'
  if `cost` *(r)* $> T$ **then**
    break
  end
  $a \leftarrow a'$
  $z \leftarrow w(x, y)$
  D.**append**$(\langle z, x, y \rangle)$
end
return a

Algorithm 2: Orienteering. Input: previously known data D, allowed trip costs including duration and probing time T, available area S, kernel setting ϕ - grid where to take measurements G, list of points to visit for probing a. Output: expanded list of points to visit for probing a.

3 Computational results

Ten "true functions" were adopted to evaluate the method as described in [3]. From the same work, the time limit of 100 units was adopted for the trip duration (navigation and probing, each probing cost 1 unit). We compared the method to a previous work to evaluate the effect of adding the disturbance step after the orienteering step [3]. Moreover, a comparison to the use of a disturbance based on particle swarm optimization (PSO) was also included in the experiments [5].

Figure 2 (left) shows the progress of the model after the orienteering step for one of the "true functions". Variance clearly diminishes as the computational time increases for both methods: the proposed method (Distortion) and PSO (PSwarm). The behavior of the error curve is similar, despite its non-monotonicity. Overall, the proposed method has a better outcome than the version based on particle swarm optimization, as can be seen in Table 1. The table shows the sum of the standard deviation σ and the sum of the error Δ over the surface. Considering grids of initially known points 4x4, 7x7 and 10x10 and functions from 1 to 5 (used during the design of the method), the proposed method achieved the best error values in 14 out of 15 experiments; for PSwarm, the next best option, this number was 2. Considering functions from 6 to 10 (not used during the design of the method), the proposed method achieved the best error values in 10 out of 15 experiments against 7 of the (only-)Orienteering method.

Finally, Figure 2 (right) shows how the computational time is spent by the method. Most of the time is spent inducing the regressor and making predictions. The time spent calculating TSP solutions is negligible.

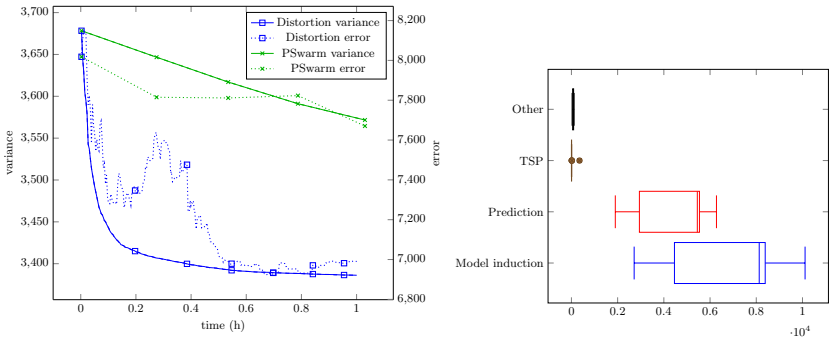


Fig. 2. Evolution of variance and error curves for Distortion and PSwarm for function 5 and grid of initial points 4x4 (left). Time spent on each step of the method (right).

4 Conclusions

In this work we propose a new algorithm for the problem of selecting the location of probings on an expedition whose aim is to improve the knowledge of content

Table 1. St. deviation σ and error Δ over surface of 10 functions (**F**)

F		4x4				7x7				10x10			
		Init	Orien	Dist	Pswa	Init	Orien	Dist	Pswa	Init	Orien	Dist	Pswa
1	σ	7838	2097	1849	*1438*	4114	1546	*1498*	1756	2263	1535	*1418*	1682
	Δ	31109	663	**532**	4104	7762	246	**218**	815	750	181	**60**	479
2	σ	10238	*10155*	*10155*	*10155*	3833	1468	*1439*	1624	2752	1926	*1701*	1908
	Δ	60381	**60376**	**60376**	**60376**	16273	6868	**6124**	7347	5122	3454	**2529**	3488
3	σ	6754	1719	*1395*	1631	4386	2204	*1882*	2151	2762	1938	*1699*	1920
	Δ	52273	14532	**8893**	13286	17307	7329	**6462**	7320	5424	3818	**2132**	3106
4	σ	7838	2097	*1849*	1995	4400	2334	*1922*	2246	2708	1906	*1842*	1888
	Δ	54245	8250	**6603**	7981	23692	6644	**5661**	6642	6885	5245	4563	**4552**
5	σ	8058	3678	*3386*	3572	4411	2044	*1942*	2026	2716	1685	*1659*	1671
	Δ	63666	8018	**6991**	7670	24731	6593	**5451**	6640	7318	3086	**3048**	3090
wins		0	1	5	1	0	0	5	0	0	0	4	1
6	σ	273	36	36	*35*	52	*35*	*35*	*35*	40	*34*	*34*	*34*
	Δ	10	**1**	**1**	**1**	2	**1**	**1**	**1**	**1**	**1**	**1**	**1**
7	σ	273	36	*35*	*35*	52	*35*	*35*	*35*	40	*34*	*34*	*34*
	Δ	**684**	874	1685	1963	676	**675**	**675**	**675**	**675**	**675**	**675**	**675**
8	σ	8660	7848	*7524*	7783	3841	2712	*2610*	2687	3030	2555	*2323*	2545
	Δ	44671	39017	**32066**	42733	19765	**6212**	6500	6417	9084	2491	**1711**	2462
9	σ	8690	7959	7953	*7932*	5555	3044	*2849*	2994	2722	1686	1661	*1068*
	Δ	70319	**49456**	49476	50492	29777	**6684**	8888	7111	9549	3587	4009	**3166**
10	σ	8687	7955	7949	*7889*	6624	4736	*4083*	4669	2825	2006	*1752*	1993
	Δ	71940	48707	**48642**	53043	36573	12511	**6586**	11517	12376	7146	**4682**	6841
wins		1	1	3	1	0	4	3	1	2	2	4	3

levels of some resource in the seafloor. The algorithm improved the state-of-the-art evolving the probing points. A potential direction for future work concerns reshaping the algorithm in the context of reinforcement learning. Indeed, as the reward related to a probing decision is only perceived after the result of the actual probing is obtained—ultimately, only after the whole trip is concluded—one can think of using real-world outcomes for rewarding algorithm's decisions.

References

1. Bruce L Golden, Larry Levy, and Rakesh Vohra. The orienteering problem. *Naval research logistics*, 34(3):307–318, 1987.
2. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
3. João Pedro Pedroso, Alpar Vajk Kramer, and Ke Zhang. The sea exploration problem: Data-driven orienteering on a continuous surface. *arXiv preprint arXiv:1802.01482*, 2018.
4. Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.
5. A Ismael F Vaz and Luís N Vicente. A particle swarm pattern search method for bound constrained global optimization. *Journal of Global Optimization*, 39(2):197–219, 2007.

The Problem of Using Remnants of Fabrics in Upholstered Furniture Factories

Bogdan Staruch and Bożena Staruch

University of Warmia and Mazury, Olsztyn, Poland
bstar@uwm.edu.pl

Abstract. We tackle the problem of the use of remnants of upholstery fabrics in medium-sized furniture factories. This is a significant problem that arises in the management of upholstered furniture plants. As upholstery fabrics cost is one of the main factors influencing production costs, there is a need to use the remnants of fabrics efficiently. Mathematically, this problem can be considered as very similar to the following NP-hard problems: a bin packing problem, a 1-dimensional cutting stock problem, a variable sized bin packing problem, a multi-knapsack problem. However, it is much more complicated, because the remnants of fabrics are of different lengths and the cuts are very diverse. Moreover, the objective is unclear. We present an integer linear model for solving this problem together with a heuristic algorithmic solution.

Keywords: remnants of upholstery fabrics · variable sized bin packing · integer linear programming · Lean Production · production management · multi-knapsack problem.

1 Introduction

Interviews with production managers in medium-sized upholstered furniture factories reveal their awareness of the need for efficient use of fabric remnants. The main reason for saving fabrics is their high cost, so it is worth using the remnants even at the expense of employees' time. In practice, two routines are used: 'emptying' of the warehouse and use of remnants in ongoing production. The first one consists of storing the remnants into the warehouse for future use and then emptying it if required. The disadvantage of this solution is the need for increased storage space and the risk that there will be no more orders for a given fabric. The advantage is that in the case of a sufficiently large number of orders for products made from a given fabric, the remnants could be used more optimally. The second routine includes checking whether remnants can be reused, before starting a new roll. This is particularly important in the case of customised production, which involves a large variety of rare or even individual products. The disadvantage of this solution is the greedy way of selecting remnants for production, which reduces the optimal use of remnants of fabrics.

Regardless of which routine is used, the form of the objective function is unclear and difficult to determine. It is known that the use of remnants leads

to savings, but it is difficult to decide what factors lead to these savings. The classic methods based on a bin packing problem and a 1-dimensional cutting stock problem (see [1]-[5]), or even on a variable sized bin packing problem ([6]-[7]), use objective in a form of minimization of the number of bins/rolls. This doesn't apply in the case of use of remnants. In the problem of this paper we can establish the aim of an algorithm in a rather heuristic form, that is, *use the remnants as efficiently as possible, saving longer remnants and avoiding the use of new rolls of fabric.* On the other hand, the use of a multi-knapsack problem (see [8]-[9]) assumes that having a large number of cuts choosing those that bring the highest total profit. This neither works, because in the discussed problem all the planned cuts are to be made. Choosing the order of cuts, we follow the heuristics in the form of: *longer cuts first* assuming that after the arrangement of longer cuts, shorter cuts will be easier to place in the remnants formed after cutting off the longer ones.

In this paper we consider both routines and propose a sub-optimal algorithmic solution for the stated problem. First step of the algorithm is to arrange the cuts using the Best-Fit(BF) principle (see Algorithm Best-Fit in [1]). This arrangement determines the remnants and eventually the whole rolls that are going to be used in the solution. The second step is to improve the solution. We also propose an objective function in the second step which, together with natural constraints on the length of remnants, forms an integer linear model. Hence, in case of a 'small' number of remnants and cuts, a solver based on known methods for solving Integer Linear Program (ILP) can be used to obtain an exact solution at a given stage.

2 Conventions and Assumptions

We assume the need for making some cuts from the given type of upholstery fabrics. The fabrics are uniquely determined by their attributes such as color, texture, producer. Hence, let us assume that the type of fabrics is fixed. We have n cuts, each of the given length $d_i, i = 1, ..., n$, to be done during the given production shift. The cuts are preventively sorted by decreasing lengths, *from the left to the right.*

We have m remnants at our disposal and we also treat full rolls as remnants. To be sure that the solution always exists, we take at the beginning a big number of full dummy rolls. Every remnant has its length $r_j, j = 1, ..., m$. The remnants are sorted by increasing lengths, *from the bottom to the top.*

Since the lengths of the remnants are rarely equal to the sum of a combination of lengths of the cuts, residual pieces will result, so-called *trim loss* (see [2]).

We introduce binary decision variables $x_{ij}, i = 1, ..., n, j = 1, ..., m$, where $x_{ij} = 1$ if the i-th cut is associated to the j-th remnant.

Hence, the following ILP problem, where objective is now unclear, can be written down:

minimize *losses due to the trim loss*

subject to $\sum_{i=1}^{n} x_{ij} \cdot d_i \leq r_j, \quad \sum_{j=1}^{m} x_{ij} = 1, \quad x_{ij} \in \{0,1\}$

3 Algorithmic solution

Step1. The Choice of Remnants First, we arrange cuts using the approximation algorithm Best-Fit (BF).

1. $x_{ij} = 0$ for every $i = 1, ..., n$ and $j = 1, ..., m$.
2. For $i = 1$ to n put the i-th cut on the first met feasible remnant j ascending from the bottom to the top, where the j-th remnant is said to be *feasible for the i-th cut* if $d_i + \sum_{k=1}^{i-1} x_{kj} \cdot d_k \leq r_j$. Put $x_{ij} = 1$.

As an output for this step we get an arrangement of all cuts on the remnants which is a solution, although not necessarily the optimal one. The remnants with $\sum_{i=1}^{n} x_{ij} \cdot d_i = 0$ are useless, so we put them aside. The rest of remnants is numbered again, from 1 to m, for simplicity.

Step2. Improvement The idea of the improvement is as follows:

1. Take K cuts from the top of the remnants. Namely, $K = \{i_1, ..., i_K\} \subset \{1, ..., n\}$ with $i_1 \leq i_2 \leq ... \leq i_K$. Put $x_{kj} = 0$ for every $k \in K$, $j = 1, ..., m$. These cuts will be called *the K-top cuts*.
2. The remnants with $\sum_{i=1}^{n} x_{ij} \cdot d_i = 0$ are now *cleared*. Let M denote *the top uncleared remnant*.
3. Use the K-top cuts to rearrange the cuts using the Push-Up procedure.

Push-Up Procedure

1. For $k = 1$ to $k = K - 1$ take the two cuts from the left from the set of K-top cuts and remove them from K. Let D_k be the sum of their lengths, let us treat this sum as a one joint cut, denoted by D_k, too.
2. Ascending from the bottom to the top find the first remnant j containing a cut i such that $d_i < D_k$ and j becomes feasible for D_k after subtracting the i-th cut. Chose the shortest cut i among all the cuts assigned to j satisfying the above conditions. We will say then that *D_k pushes-up the i-th cut from the j-th remnant* or *i is pushed-up by D_k* and if it holds put $x_{ij} = 0$, $x_{i_k j} = 1$, $x_{i_{k+1} j} = 1$.
   (a) If there is a cut i pushed-up by D_k, ascend with i to the top and push-up first possible cut. If there is a cut pushed-up by i continue ascending to the top with this cut. Add the last pushed-up cut to the set K placing it in accordance to the ordering.

(b) In opposite case, assign the i_k-th cut to the first feasible remnant (possibly the cleared one) and add the other to the set K.

3. If K has the one-element, assign this element to the first feasible remnant.

Notice that every iteration in the Push-Up procedure results in decreasing the cardinality of K by one, hence the procedure stops after K iterations. Moreover, after pushing-up a cut from the given remnant, the trim loss decreases.

It may happen that no improvement was obtained, that is, all the K-top cuts remain unchanged. Then we can run the Push-Up procedure taking the first three cuts instead of two in 1. If no improvement is obtained the algorithm stops. In case of any improvement we repeat the Push-Up procedure for $K-1$-top cuts.

The Value of K It can be easily seen that bigger value of K improves the solution. On the other hand, bigger value of K increases computational complexity. The most promising solution to this dilemma is to determine the K-value experimentally, taking into account the actual production data of the factory in question. A percentage value can be used, for example 30 percent of all the cuts. Alternatively, a different value depending on n can be taken, for example, K being $a \cdot \sqrt{n}$ for some positive a.

Computational Complexity Pessimistic time of execution of Step1 of the algorithm is $O(m \cdot n)$. By pre-processing of the choice of remnants we obtain an upper bound on m that is proportional with n. Hence, the computational complexity of Step1 is $O(n^2)$.

The computational complexity of Step2 of the algorithm is $O(K^2 \cdot n)$. Therefore, in case K is proportional to $\sqrt{n}$ the computational complexity of Step2 as well as the whole algorithm is $O(n^2)$.

4 The Objective Function

Having the choice of remnants obtained in Step1 we propose an adequate objective function. We introduce a *dummy remnant* for this purpose, which is assumed to be of unbounded length and at least one cut is assigned to this remnant. For example, the M-th remnant form Step2 (1) can play the role of the dummy one. Then the objective is a minimization of the total usage of the dummy remnant. Then the ILP(M) optimization model is as follows:

$$\text{minimize} \quad \sum_{i=1}^{n} x_{iM} \cdot d_i$$

$$\text{subject to} \quad \sum_{i=1}^{n} x_{ij} \cdot d_i \leq r_j \text{ for } j = 1, ..., M-1, \quad \sum_{j=1}^{M} x_{ij} = 1,$$

$$x_{ij} \in \{0, 1\}$$

As Step2 in the proposed algorithm realizes the above objective approximately, the ILP(M) is a tool for experimental determining of K.

Finally, we want to emphasise that under assumption that a fast algorithm for solving the above ILP(M) exists, we can propose the following iterative solution:

1. Take m remnants obtained in Step1. Let $M = m$.
2. Solve ILP(M).
   (a) If $\sum_{i=1}^{n} x_{iM} \cdot d_i > 0$ the solution is optimal.
   (b) If $\sum_{i=1}^{n} x_{iM} \cdot d_i = 0$ put $M = M - 1$ and go to 2.

The possibility of obtaining such a fast algorithm is high in case of small size of input data, what happens in medium-size furniture factories managed by Lean Production. Experiments with the GLPK (GNU Linear Programming Kit [10]) package show that an exact solution is obtained quickly when the number of remnants and cuts is around 20. In case of big size of input data the main algorithm presented is this paper efficiently produces an approximated solution that is fully acceptable by practitioners.

References

1. Delorme, M., Iori, M., Martello, S.: Bin Packing and Cutting Stock Problems: Mathematical Models and Exact Algorithms, EUR. J. OPER. RES., **255**(1), 1–20. (2016)
2. Dyckhoff, H., Finke, U.: Cutting and Packing in Production and Distribution. Physica- Verlag, Heidelberg, (1992).
3. Singh, A., Gupta, A.K.: Two heuristics for the one-dimensional bin-packing problem, OR Spectrum, **29**, 765-–781. (2007)
4. Sweeney, P.E., Paternoster, E.R.: Cutting and packing problems: a categorized, application-orientated research bibliography, J. OPER. RES. SOC., **43**, 691-–706. (1992)
5. Wäscher, G., Haußner, H., Schumann, H.: An improved typology of cutting and packing problems, EUR. J. OPER. RES., **183**, 1109-–1130. (2007)
6. Epstein, L., Levin, A.: An APTAS for Generalized Cost Variable-Sized Bin Packing, SIAM J. Comput., **38**(1), 411—428. (2008)
7. Holthaus, O.: Decomposition approaches for solving the integer one-dimensional cutting stock problem with different types of standard lengths, EUR. J. OPER. RES., **141**, 295—312. (2002)
8. Trick, M.A.: A dynamic programming approach for consistency and propagation for knapsack constraints, ANN. OPER. RES., **118**, 73-–84. (2003)
9. Martello, S., Toth, P.: Knapsack Problems: Algorithms and Computer Implementations. John Wiley and Sons, Chichester. (1990)
10. The GNU Linear Programming Kit package available at https://www.gnu.org/software/glpk/

Tasks Assignment to Workers on the Basis of Their Competencies

Bożena Staruch

University of Warmia and Mazury, Olsztyn, Poland
bostar@matman.uwm.edu.pl

Abstract. The paper is motivated by real problems concerning tasks assignment to workers in medium-sized upholstered furniture plants managed using the Lean Production method. We propose an integer linear optimization model for solving this problem. The model is further enhanced by competence coefficients that describe the skills or capabilities of each employee to perform each specific task. The competence coefficients are used to block the possibility of assigning the given task to a worker that has no skills to do it. Additionally, we involve a dummy worker to the model which guarantees existence of a solution of the problem. We also discuss the use of the presented model to solve real problems related to production supervision.

Keywords: tasks assignment · competence coefficient · integer linear programming · Lean Production · production management · generalized assignment problem.

1 Introduction

We will present the issue of assigning tasks to employees in medium-sized upholstered furniture plants managed using the Lean Production method. This problem is motivated by real-life challenges described by upholstered furniture plants managers in Poland. The main idea and a large part of this publication were obtained in 2012 in cooperation with ITM[1], which provides production management systems in furniture factories. IT implementation of the solution to the presented problem was awarded in 2016 with the 'Debut of the Year' award given by the *Rzeczpospolita* newspaper.

Lean Manufacturing (see [1], [2]), used in the factories under consideration requires a high degree of flexibility in production planning. The products are produced on-demand and are often customized. Hence production plans are very diverse, they can vary from one day to another or even during an ongoing shift. Other aspects such as unpredictable production disturbances, bad condition of the working person, or hidden defects in materials, which can only be seen during execution of the task, require an immediate reaction and reallocation. The problem of reallocation was considered, for example, in [3] and [4]. In addition,

[1] ITM Sp. z o.o. www: itm.com.pl

in the analysed upholstered furniture factories, color-coded task priorities based on the Kanban philosophy (see e.g [5]-[8]) are usually applied. Red indicates that a task must be performed during the current shift. Tasks marked in yellow are those that will be red during the next shift. Green indicates tasks that can wait but may be done if possible. Therefore, there is a great need to create an effective and flexible system supporting production management, in particular human resources management. As far as upholstered furniture factories are concerned, such a solution is most needed in two stages of production: covers sewing and upholstering.

We propose an integer linear optimization model for solving the problem of task assignment to workers. The model is further enhanced by competence coefficients with their 'normal' values from an interval $[MinC, MaxC]$ that describe the skills or capabilities of each employee to perform each specific task. The competence coefficients may be determined on the basis of a subjective appraisal by the superiors or on the basis of some automatic evaluation systems. Alternatively, these two ways can be mixed. It is worth considering the development of an automatic classification model (more on classifiers can be found in [9]-[11]). Regardless of the method of evaluation, the values of these factors are to be such that a lower value of the coefficient indicates greater predisposition of the worker to perform a given task. By setting the values of some competence factors on $3 \times MaxC$, we obtain a way of blocking the allocation of tasks to workers with insufficient competence to perform them. On the other hand the allocation of very easy tasks to highly qualified workers can be blocked the same way. The need for blockages is discussed in the domain called *task assignment*, where binary values of competence coefficients indicating skills or authorisation are used (see e.g. [12]).

An additional element of the presented model is a dummy worker (named *Dummy* for short) with its competence value equal to $2 \times MaxC$. Involving Dummy ensures the existence of an optimal solution of the modeled problem. Further analysis of tasks allocated to Dummy allows to decide if reallocation of tasks should be performed. In particular, a 'red' task assigned to Dummy means that this task will not be finished during the current shift. Then some of the 'green' or 'yellow' tasks should be removed from the production plan so that all urgent items are completed.

The problem discussed in this paper is a variant of a *generalized assignment problem* that is known to be NP-hard and even APX-hard to approximate it. There are many algorithmic methods in the literature to obtain a sub-optimal solution to this problem based on different types of heuristics (see [13]-[22]). Therefore, in this work we will focus on the model itself and its application in practical situations in production rather than on the search for an algorithmic solution.

2 The Model Description

2.1 Conventions and Assumptions

Assume that there are workers in a plant that are planned to perform a set of tasks: sewing covers or upholstering furniture. Let us use the following notations:

1. m is the number of workers, who are indicated by indices $i = 1, ..., m$, where Dummy is the m-th worker;
2. every worker i works for a given interval of time not exceeding L_i;
3. n is the number of tasks planned to be performed, which are indicated by indices $j = 1, ..., m$;
4. every item j has its fixed normative execution time t_j;
5. Dummy's time $L_m = \sum_{j=1}^{n} t_j$;
6. each worker may perform any number of tasks within her/his time limit L_i;
7. each task can be performed by exactly one worker;
8. each worker i has specific competencies to perform the task j. The level of these competencies is normally presented in the form of a competence coefficient $Cij \in [MinC, MaxC]$ so that the higher the competencies, the lower the competence coefficient value;
9. $C_{ij} = 3 \times MaxC$ if we want to block the possibility of performing the task j by the i-th worker;
10. $C_{mj} = 2 \times MaxC$ is applied to Dummy.

Decision Variables We use binary decision variables x_{ij}, where $x_{ij} = 1$ means that the j-th task is assigned to the i-th worker.

Constrains All the $m + n$ constraints are described above in points 6. and 7.

Objective function Our goal is to allocate m tasks to n workers so that the tasks are performed in 'the best' possible way. Therefore, when defining the objective function, it is necessary to define what the term 'the best' means. Typically, the longer normative time of a given item, the higher level of worker's skills is needed. This is why the objective function should depend on both normative times and competence coefficients. We decided to minimize the total cost of tasks, where the ij cost is equal to $t_j \cdot C_{ij}$.

2.2 Integer Linear Model

Mathematically the problem of tasks assignment to workers on the basis of their competencies (TAW problem) can be formulated as a generalized assignment problem in the form of an integer linear program:

$$\text{minimize} \quad \sum_{i=1}^{m} \sum_{j=1}^{n} t_j \cdot C_{ij} \cdot x_{ij}$$

$$\text{subject to} \quad \sum_{j=1}^{n} x_{ij} \cdot t_j \leq L_i, \quad \sum_{i=1}^{m} x_{ij} = 1, \quad x_{ij} \in \{0, 1\}$$

As can be easily seen the TAW problem is an instance of a minimized generalized assignment problem, where $t_j \cdot C_{ij}$ are costs of performing j-th task by i-th worker. Therefore, any of existing algorithmic solutions can be used for solving this problem. Notice that because the normative time of task performance is fixed regardless of the assigned employee, the TAW problem becomes practically useless if competence factors are not differentiated.

2.3 Competence Coefficients

Competence coefficients play an essential role in the TAW problem. As we mentioned earlier, they are used to block the assignment of tasks to selected workers and to introduce a dummy worker. In addition, competence factors express an employee's level of skills resulting in faster work and higher quality of workmanship. The right choice of these coefficients has a direct impact on higher production efficiency. For this reason, a well-thought-out system for determining competence factors, tailored to the needs of the production department in the factory, is needed.

Interviews with production managers in upholstered furniture factories show that it is normal practice to manually assign tasks to be performed. The shift manager, based on his own experience and knowledge of the skills of his subordinate workers, decides on the allocation of tasks.

An attempt to automate the determination of competence coefficients should start by writing down the knowledge of production managers in numerical form, for example by taking integers from the range of 80 to 120 in such a way that the lower value indicates higher skills. Subsequently, the factors influencing the quality of work should be identified and their numerical evaluation should be recorded in the form of an information table during long-term observation. For example, three of these factors can be quickly identified: the number of tasks of a given type performed so far, the reduction in execution time in relation to normative time, the number and importance of corrections (loss of time for corrections, material losses, etc.). With such an information table, an appropriate classification model can be used to determine the competence coefficients. Obviously, the obtained coefficients should be monitored on an ongoing basis and, if necessary, corrected.

There is no universal rule indicating which values of the competence coefficients are the best. One factory will opt for the quality of workmanship even at the expense of time, while for the other the most important will be the reduction of time. In the latter case, the competence factors may express a percentage saving of time for each task in relation to the normative time.

Competence coefficients may be used to obtain an assignment that realizes 'seriality' requirements. We understand seriality as allocation of series of the same tasks to a worker if possible. Such seriality gives saving of time: shortening the time of adjusting the workplace (changeover time), automation and personal optimization resulting from repeating the same activities several times. Tests shows that greater differentiation in the values of the competence coefficients increases the seriality effect.

3 TAW Problem in Production

In this section we discuss different aspects of application of the TAW problem in real-life production scheduling. Let us assume that a fast algorithmic method for obtaining a sub-optimal solution, denoted by AS, is applied to TAW. We must emphasise the fact that in the management of real production, an exact optimal solution is not necessary. Instead, speed of calculation and flexible response to various production events are required. Therefore, a flexible method of partial assigning of tasks is proposed in this paper. Even a very simple AS based on continuous relaxation and bounding is surprisingly effective. We will say that the tasks are *partially assigned* if a proper subset of the set of tasks is allocated to workers. Where the tasks are partially assigned, the TAW problem can be run again with a new subset of tasks and with L_is decreased by subtracting normative times of the previously allocated tasks. By repeating this procedure, we obtain a flexible method of reaction on different aspects and disturbances in production.

Preferences in Production As we mentioned in Introduction, preferences can be modeled using color indicators. Red tasks must be done during the current shift. After applying the TAW problem to all the workers of the given shift and all the possible tasks (including red, yellow and as well as green) the solution can be analyzed. Tasks that are allocated to Dummy should be treated as unassigned. In case of unassigned red items, a reaction is required, which could be as follows:

1. remove a part of green or yellow items, preferably those which are not in series (the rest is partially assigned) and run AS again,
2. fix red and serialized items that are already allocated (they are partially assigned), remove the rest and run AS again,
3. unblock some of the workers and run AS again,
4. start with partial assignment of red items, then allocate the yellow and/or the green ones.

Each of these reactions leads to a deterioration of the optimal solution, but the continuity of production is more important than the accuracy of the solution.

Disturbances in Production Many types of production perturbations can occur, but the reallocation of tasks as described above can always be applied.

References

1. Ohno, T.: Toyota Production System: Beyond Large-Scale Production. CRC Press. (1988). ISBN 978-0-915299-14-0.
2. Womack, J.P., Jones, D.T.: Lean Thinking: Banish Waste and Create Wealth in Your Corporation. First Free Press. (2003). ISBN 0-7432-4927-5.
3. Jain, A.K., Elmaraghy, H.A.: Production scheduling/ rescheduling in flexible manufacturing. INT. J. PROD. RES., **35**(1), 281–309. (1997)

4. Strickera, N., Lanzaa, G.: The concept of robustness in production systems and its correlation to disturbances. In: Procedia CIRP., **19**, 87–92. (2014)
5. Anderson, D.J.: Kanban: Successful Evolutionary Change for Your Technology Business. Blue Hole Press. (2010). ISBN 0-9845214-0-2.
6. Hammarberg, M., Joakim, S.: Kanban in Action. Shelter Island, NY: Manning Publications, (2014). ISBN 978-1-617291-05-0.
7. Ladas, C.: Scrumban: Essays on Kanban Systems for Lean Software Development. Modus Cooperandi Press. (2009). ISBN 978-0578002149.
8. Scotland, C.: http://www.methodsandtools.com/archive/archive.php?id=104. last accessed 2018/10/21.
9. Artiemjew, P.: On Classification of Data by Means of Rough Mereological Granules of Objects and Rules. LNAI 5009, Springer, Berlin, 221–228. (2008)
10. Staruch, B.: Classification model based on topological approximation space, LNCS, vol. 10314 LNAI, p.p. 570–578. (2017)
11. Staruch, B., Staruch, B.: A topological approximation space based on open sets of topology generated by coverings, LNCS, vol. 10314 LNAI, p.p. 130–137. (2017)
12. IBM Knowledge Center https://www.ibm.com/support/knowledgecenter/en. last accessed 2018/10/21.
13. Baykasoglu, A., Özbakır, L., Tapkan, P.: Artificial bee colony algorithm and its application to generalized assignment problem, in: F.T.S. Chan, M.K. Tiwari (Eds.), Swarm Intelligence: Focus on Ant and Particle Swarm Optimization, I-Tech Education and Publishing, Vienna, Austria, pp. 113–144. (2007)
14. Cattrysse, D., Salomon, M., Van Wassenhove, L.N.: A set partitioning heuristic for the generalized assignment problem, EUR. J. OPER. RES., **72**, 167–174. (1994)
15. Chu, P.C., Beasley, J.E.: A genetic algorithm for the generalized assignment problem, Computers and Operations Research, **24** 17–23. (1997)
16. Diaz, J.A., Fernandez, E.: A tabu search heuristic for the generalized assignment problem, EUR. J. OPER. RES., **132** 22–38. (2001)
17. Lourenço, H.R., Serra, D.: Adaptive search heuristics for the generalized assignment problem, Mathware and Soft Computing, **9** 209–234. (2002)
18. Nauss, R.M.: Solving the generalized assignment problem: an optimizing and heuristic approach, Informs Journal of Computing, **15** 249–266. (2003)
19. Osman, I.H.: Heuristics for the generalized assignment problem: simulated annealing and tabu search approaches, OR Spektrum, **17** 211–225. (1995)
20. Randall, M.: Heuristics for ant colony optimisation using the generalised assignment problem, in: Proceedings of IEEE Congress on Evolutionary Computation, Portland, Oregon, USA, pp. 1916–1923. (2004)
21. Savelsbergh M.: A branch-and-price algorithm for the generalized assignment problem, OPER. RES., **45** 831–841. (1997)
22. Yagiura, M., Ibaraki, T., Glover, F.: A path relinking approach with ejection chains for the generalized assignment problem, EUR. J. OPER. RES., **169** 548–569. (2006)

A framework for defining scheduling problems

Attila Tóth[1] and Miklós Krész[2, 3]

[1,3] University of Szeged, Szeged, Hungary
[2] InnoRenew CoE and UP IAM, Slovenia
[1] attila@jgypk.szte.hu
[2] kresz@jgypk.szte.hu

Abstract. Scheduling is one the most classical optimization problems with having several types according to the resources, the cost function and the defined constraints for the jobs. In practice many optimization problems can be formalized in a scheduling framework, but direct applications are frequently difficult as the adaption of the general methods to the given field is typically far from trivial. Usually real life rules are much more complicated than the published constraints and these rules can be significantly varied in different application areas. Furthermore, industrial applications must be flexible to adapt to the changing circumstances. In this study, we present a general definition structure for the rules by which the optimization methods can be flexible in managing the different constraints of the application areas.

Keywords: Scheduling, General Merge Model, Industrial applications.

1 Introduction

Scheduling is one of the most popular optimization problems with the applications in many fields of industry. The aim of this type of problems is to allocate the resources to activities over a time period considering predefined constraints minimizing a given objective. The resources are usually called machines and the activities called jobs or tasks. Early works dealt with only some simple cases of the problem in the aspect of machine and job environment, constraints and objective, which are rarely found in industry. The scheduling problems frequently appear in real life application in many different forms, like production scheduling in manufacturing, crew scheduling or time-table scheduling. Some early results can be found in Lenstra et al. [8] and excellent reviews of the scheduling problems are Chen et al. [2] and Pinedo et al. [11].

Scheduling problems are widely studied in academic research; many problem variants, analyses and solution methods are published in the last decades. Though, the industrial demand is intensive for high quality solutions, there is still a big gap between the

theoretical results and the real life applications. The problem is that the published solution methods can be rarely applied to direct industrial use. There are two reasons which mainly cause this duality. On one hand, in the research studies usually the problem is relaxed by omitting the difficult constraints[1] or using a simplified objective which leads unacceptable solutions for real applications. On the other hand, the published methods are too rigid for real usage in the sense that they are based on some fixed structures and rules that are not possible to be changed without rebuilding a significant part of the method. However, scheduling problems in real life applications are usually part of a decision making framework which require flexibility for the method to be able to be customized in changing circumstances. Nevertheless, in the last few years, there were a few attempts to handle scheduling problems from a point of view of application-oriented frameworks (see e.g. [10][12]) .

Usually, in real life scheduling problems the most difficult part is fulfilling the practical constraints defined by highly complex rules. Moreover, these constraints are very difficult to be categorized as the set of rules can be rather distinct even within the same problem type if concerning different applications. For example, the regulations for crew scheduling might strictly depend on national and company rules. Therefore, a practical solution on a problem type can be very rarely directly adapted to other environment or business case.

In this study, we give a general rule definition method for scheduling problems which provides the basis of a framework to bridge the gaps outlined above. Using this form, the optimization methods are more independent from the constraint set of a given application area and can be more easily used to another family of problems. Each rule is defined as an independent unit which is responsible for its effective verification during the optimization. By this way the solution methods become suitable to be the part of a flexible framework: the rule set can be easily modified without rebuilding the optimization algorithm.

2 Scheduling

In the scheduling problem there are m machines (resources) and n jobs. A schedule is considered as an assignment between machines and jobs. A schedule is feasible if each job is performed and there is no time overlapping, i.e. each job can be performed at most one machine at once and a machine does not work on two or more jobs in the same time. Moreover, the schedule must satisfy other given rules which are defined by the specific problem. Usually a deterministic scheduling is assumed where each property and all data of the problem are known with certainty. In industrial case generally the offline scheduling is typical where the jobs, machines and all other data are known in advance.

The three main points of scheduling problems are the machine environment, the job characteristics and the optimality criterion. Many types of the scheduling problems can be

[1] from modelling or algorithmic point of view

distinguished by the above aspects. In the machine environment point of view the problem can be single stage or multi stage. For single stage case either single machine is considered or using parallel machines which can be identical, uniform or unrelated. For multi stage case flow shop, open shop and job shop problem types can be identified. Another categorization is the job processing time which can depend on the job type as well as on the performing machine. Further job characteristics are the release date, due date or deadline. Precedence constraints among the jobs can be also defined which prescribes a partial order among the jobs. Weights are frequently assigned to the jobs and pre-emption is also a natural constraint (when jobs can be interrupted and continued later). The objective of the scheduling provides a further classification of these problems: minimizing the makespan, the total competition time, the flow time, the maximum lateness, the total tardiness or a weighted combination of any are all the typical cases. Some scheduling types can be solved in polynomial time, but most of them are NP-hard.

Formally, scheduling problems are described by a three-field representation $\alpha \mid \beta \mid \gamma$, where α defines the machine environment, β represents the job characteristics and γ is the optimality criterion [4]. For example, $1 \mid r_j, prec \mid \sum w_j C_j$ defines a single machine scheduling problem where the jobs have release date with a precedence constraint among them and the objective is the minimization of the weighted completion time.

Many industrial problems can be handled as a scheduling problem, e.g. crew scheduling (bus driver, pilot, call center, etc.), manufacturing (car, electronic devices, etc.), CPU operation, time table scheduling. These practical problems can be formulized with the above methodology and categorized to a scheduling problem type.

3 Rule definition

The main reason of the diversity of the industrial scheduling problems is the high variability of the constraint sets. In the literature the presented solution methods are usually specialized to the rules defined by the given industrial demand. Because of this, it is hard to adapt the considered method to other problem circumstances with different regulations.

One way is to increase the flexibility of the solution methods is to make them more independent from the defined rules. The above goal can be reached within a framework in such a way that a rule is defined as an individual part and the methods should use them as a black-box unit. In this way, the set of rules can be varied without modifying the solution method or the method can be exchanged for the same rule set.

Usually the most time consuming part of an optimization process for real life scheduling problems is verifying the solutions (or sub-solutions) by checking the constraints. Therefore, it is an obvious demand that a constraint should be checked as easy and fast as possible.

In this study we propose a universal form for rule definition which fulfils the above requirements. This model is a generalization of the so-called Merge Model for graph colouring published by Juhos et al. [6].

3.1 Merge Model for graph colouring

Many scheduling problems can be formalized as graph colouring if some complex constraints are ignored. In the graph colouring problem the goal is to colour the nodes such that the neighboured nodes must not get the same colour. Generally, the objective is to find the minimal number of colours with which the graph can be coloured, or if the number of colours is given, to find a colouring with the given colour set (it is called *k*-colouring). In scheduling problems the nodes represent the jobs and the colours are the machines. The edges of the graph represent the constraints, i.e. which jobs cannot be assigned to the same machine (e.g. because of time overlapping).

For the graph colouring problem Juhos et al. [6] published a general model, called *Merge Model*, which makes the colouring more effective reducing the number of *constraint checks*. During the colouring process the nodes are coloured one by one. In each step, there must be checked if the current node can be coloured with a certain colour. To check this constraint the Merge Model uses a *Merge Table* which is the adjacency matrix of a hyper graph. The single nodes of the hyper graph are the uncoloured nodes and the hyper nodes are the coloured ones with containing all the single nodes using the corresponding colour. In this table each column represents one node of the original graph and each row stands for a node of the hyper graph (normal and hyper nodes also). Here, a constraint check is only checking one value in the matrix. Since, a colour can be assigned to a node if the cross of the row of the colour and the column of the uncoloured node contains zero, i.e. there is no edge between the normal node and the hyper node. When a node is coloured, i.e. a single node is involved in a hyper node, a *merge operation* is performed in the Merge Table which makes one row from the two corresponding rows of the nodes performing dot product on them. This operation merge the edges of the normal nodes and that of the hyper nodes. This method is proved to be efficient for graph colouring problems [7].

3.2 General Merge Model for scheduling rules

Solving a scheduling problem goes through the similar steps as in the case of graph colouring. Basically, the algorithm takes the jobs (nodes) one by one and assigns them to a machine (colour). Obviously, these algorithms can have more complicated operations too, but it is not important in the rule checking point of view. The merge operation can be generalized such that two hyper nodes can be also merged, which can also reduce the number of colours. For example, some methods create job groups first and then assign them to the machines together but these groups can be considered as a machine and later the machines can be merged together to reduce their number to the required value.

Through the scheduling process in each iteration all the rules must be checked if it is fulfilled in the current solution or not. Since, the rules can be various and change time to time, a general rule definition is necessary. Let the constraints be formulized in individual objects for the scheduling algorithm, which is called *General Merge Model* (GMM). This model contains three general component denoted with a triple (R,f,g). The R is the base *merge structure* of the rule on which the constraint can be checked and the step operation can be performed. Here, R represents the solution in this rule point of view. The f: $R \rightarrow$ {true, false} is the *checking function* which gives true if the solution satisfies the rule and false otherwise. Finally, the g: $R \rightarrow R$ is the *merge operation* which produce the next state of the solution after one step in point of view this rule. Notice that, the feasibility checking f requires only the structure R and the merge operation g. Update R to follow the changing of the solution through the optimization process. Therefore, in this model the rule forms an individual object and it is responsible for its own feasibility check. Obviously, the details of the components depend on the problem type and the considered rule and should be defined in an effective form for the application.

Graph colouring contains only one rule which can be formulated with GMM in this way. Let R be the adjacency matrix of the hyper graph which is at first the original graph. When a node j is about to be coloured with colour i, the checking function f gives true iff $R(i,j)$ is equal to 1. Finally, operation g substitutes the rows i and j with the dot product of the two rows.

Using GMM, the scheduling process is the following. At first, the merge structures R are generated for each rule. Then in each iteration, the algorithm checks the feasibility of the candidate solution using the checking functions f of each rule. If every rule is satisfied the algorithm performs each merge operations g on the corresponding R structures and move to the next step. The rest of the steps and decisions of the algorithm (e.g. how to choose jobs and machines, backtracks, etc.) are the responsibility of the solution method.

In this way, the solution method uses a separated rule set which contains the individual rule objects and which can be modified without changing other part of the program. If a new rule comes to the problem, just its merge object have to be defined with its three components and added to the rule set. This model is flexible enough to use in optimization algorithms for industrial scheduling problems.

4 Real life example: rostering problem

Rostering is a popular practical scheduling problem. Here, there are daily shifts for a given time period which have to be performed by employees. The goal is to assign the employees to the shifts with minimizing the operation cost and fulfilling some predefined regulations. This problem arises in many industrial field like public transportation, nurse rostering, call centers, etc. Besides the trivial rules (each shift must be performed and an employee can work only at most one shift at a day) there can be totally different rules

depending on the application area. Even concentrating only one rostering type, like driver rostering, the regulations can be various by the country or the transportation company.

The rostering is a single stage scheduling problem with parallel machines where the objective is usually minimizing the number of employees and the salary cost. Sometimes some penalty values are also involved in the cost function by soft constraints. This optimization problem is NP-hard [9] and many solution methods are published (pilot rostering [3], nurse rostering [1], call center [5] and driver rostering).

In rostering problem there can be several different regulations, but some of them are common. Here, we show a short list of them using GMM for demonstration purpose.

1. Each employee can have at most one shift per day. Let merge structure $R^{(1)}$ be a matrix where each column stands for one day in the planning period and each row represents one employee. The matrix has value *1* in (*i,k*) if employee *i* has a shift on day *k* and zero otherwise. At first, each shift is assigned to one fictive employee, so each row contains one *1*. Here, a shift assignment to employees means the merge of two rows (*i* and *j*) of fictive or real employees. The checking function $f^{(1)}$ returns true if the dot product of the two corresponding rows equal to zero, i.e. the rows cannot have 1 in the same column. The merge operation $g^{(1)}$ is equal to $(R^{(1)}_i \vee R^{(1)}_j$ (OR operation of the two merged rows).
2. The number of continuous working days is bounded with value *D*. The structure $R^{(2)}$ is the same as $R^{(1)}$ and $g^{(2)}$ is the same as $g^{(1)}$. The $f^{(2)}$ merges row *i* and *j* with $g^{(2)}$ and checks all parts with the length of D+1. If any of the above subrow does not contain at least one zero, $f^{(2)}$ returns false.
3. There must be at least *T* resting time between any two consecutive shifts for each employee. Let $R^{(3)}$ be similar to $R^{(1)}$ but here each component contains two values as start (*ws*) and end (*we*) working time on the given day (it comes from the start and end time of the shifts). If an employee has no shift in a day, *ws* is 1440 and *we* is 0. The $f^{(3)}$ merges the two corresponding rows with $g^{(3)}$ and checks for each day if the difference of *ws* in the day and *we* in the previous day is at least *T*. If this value is less than *T* on any day $f^{(3)}$ return false. The $g^{(3)}$ generates the merged row setting *ws* to the minimum of *ws* values in the two rows and setting *we* to the maximum of *we* values correspondingly.

In the rostering problem there can be many other rules about resting days, free weekends, shift types, etc., which can be similarly formulated.

5 Conclusion

Scheduling is one of the most studied optimization problems for decades and there is a significant demand in industrial area for effective optimization methods. However, the published methods are rarely used for practical cases. The main reason of this is that the artificial soliton methods are too rigid for the various real life circumstances. Here, a

general model is presented for rule definition for scheduling problems which is the most crucial and time consuming part of the practical problem solving. In this model, each rule is defined as an individual unit which is independent from other rules as well as from the optimization method itself. In this way, the problem definition is more flexible and suitable for real applications.

Acknowledgment

Attila Tóth acknowledges the support of the National Research, Development and Innovation Office - NKFIH Fund No. SNN-117879.
Miklós Krész acknowledges the European Commission for funding the InnoRenew CoE project (Grant Agreement #739574) under the Horizon2020 Widespread-Teaming program and the support of the EU-funded Hungarian grant EFOP-3.6.2-16-2017-00015.

References

1. E. Burke, P. De Causmaecker, G. Berghe, H. Van Landeghem, The State of the Art of Nurse Rostering, J. Sched. 7 (2004) 441–499.
2. Chen B., Potts C.N., Woeginger G.J. (1998) A Review of Machine Scheduling: Complexity, Algorithms and Approximability. In: Du DZ., Pardalos P.M. (eds) Handbook of Combinatorial Optimization. Springer, Boston, MA
3. M. Gamache, F. Soumis, A method for optimally solving the rostering problem, in: G. Yu (Ed.), OR in Airline Industry, Kluwer Academic Publishers, Boston, 1998, pp. 124–157.
4. R.L. Graham, E.L. Lawler, J.K. Lenstra and A.H.G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: a survey, Annals of Operations Research 5 (1979), 287-326.
5. T. Grossman, D. Samuelson, S. Oh, T. Rohleder, Call centers, Technical Report, Haskayne Sch. Business, Univ. Calgary. (1999).
6. Juhos, I., Tóth, A., van Hemert, J.: Binary merge model representation of the graph colouring problem. Evolutionary Computation in Combinatorial Optimization. Volume 3004 of LNCS., Springer (2004) 124–134
7. Juhos, I., van Hemert, J.I.: Improving graph colouring algorithms and heuristics using a novel representation. Lecture Notes in Computer Science, vol. 3906, pp. 123–134. Springer, Berlin (2006)
8. J.K. Lenstra, A.H.G. Rinnooy Kan and P. Brucker, Complexity of machine scheduling problems, Annals of Operations Research 1 (1977), 343-362.
9. J.K. Lenstra, A.H.G.R. Kan, Complexity of vehicle routing and scheduling problems, Networks. 11 (1981) 221–227.
10. K. Nurmi, J. Kyngas, and G. Post. Driver rostering for bus transit companies. Engineering Letters, 19(2):125–132, December 2011.
11. Michael L. Pinedo, Scheduling: Theory, Algorithms, and Systems, Springer International Publishing, 2016
12. Tóth, A., Krész, M. A flexible optimization framework for driver scheduling, Proceedings of the 11th International Symposium on Operational Research SOR '11. (2011) 346 p.

A heuristic approach for kidney exchange program

Utkarsh Verma and Narayan Rangaraj

Department of Industrial Engineering and Operations Research, Indian Institute of Technology, Bombay, India
utkarshverma@iitb.ac.in

Abstract. Kidney exchange programs are designed to find compatible matches through exchange cycles within an incompatible donor-recipient registry. Execution of long exchange cycles are logistically challenging - thus a bound on cycle length is required. Solving kidney exchange program with a bound on cycle length more than 2 is NP-hard. So we propose a heuristic approach to solve large size kidney exchange program by identifying the crucial pairs for creating 3-way cycles. It uses the blood group distribution to find the crucial pairs and prioritize them in the solution. It was observed that these crucial pairs were having hard to match recipients as well. Prioritizing hard to match recipients increases the weights for those recipients and they will have better HLA matches, PRA matches and smaller age differences. These parameters defines the success probability of execution and it will increase with higher weights. Thus overall probability of successful execution of the proposed transplants will increase with this approach. We compare our heuristic with exact approaches and it was observed that the difference between the heuristic and exact methods was minimal. The heuristic approach has a polynomial running time which allows us to run the algorithm multiple times as required.

Keywords: Kidney exchange program · Heuristic methods · Health-care management

1 Introduction

Kidney transplant is the most effective treatment for patients with kidney failure. It can happen either via a living donor or a deceased donor. If a patient has a willing living donor who is compatible with the patient, the transplant is performed, else they are registered in Paired Kidney Exchange (PKE) registry. Here the aim is to find compatible matches within the registry through swap and cycle exchanges. All the operations in a cycle need to be performed simultaneously to avoid the chances of donors backing out. This creates a bound of the cycle length as too many simultaneous transplants are practically unfeasible. Altruistic donations are another way to get a living donor kidney transplant and people donates a kidney to save the life of other without taking any incentive

from them. Altruistic donor initiated chains are also part of kidney exchange programs.

There are several factors which affect the graft survival of the transplanted kidney. Graft survival is the time duration which the kidney survived after transplantation. Factor like, HLA antigen, age difference, GFR level, waiting time on dialysis, failed vascular access etc. are considered to define the quality of a match. So the aim of the kidney exchange program is to find the optimal number of exchanges with a bound on cycle length considering the quality of matches.

Researchers have developed mechanisms for finding the optimal number of transplants within such registries. Different Integer Programming models have been proposed to solve the KEP (Kidney Exchange Program) for optimality. The problem of finding maximum weighted bounded cycles over a network is an NP-hard problem which can take an exponential amount of time in worst case. Also, it was observed that most of the proposed transplants were executed due to various reasons and approximately only *7% of the proposed transplants were successfully performed* in UNOS kidney exchange between Oct 2010 to Nov 2012 [1].

This showed that finding optimal number of matches may not result in actual transplants. So different heuristic methods were proposed to find solution which might increase the probability of execution. One such heuristic is the Spanish exchange model where they gave preference to 2-way cycles over 3-way cycles, cycles with higher weights and robust cycles [2]. This model was easy to implement and found solutions close to optimal solutions found through IP models. But giving preferences to 2-way cycle might reduce the possibility of 3-way exchange which can potentially increase the number of transplants. Also, there is no priority given to sensitized recipients and this might reduce the probability of success for those recipients. We propose a data-driven approach to find the maximum number of transplants where priority is given to sensitized recipients. We have bounded our algorithm to 2-way and 3-way cycles only as performing long cycles are logistically challenging. Our algorithm can be easily implemented and will increase the probability of execution for sensitized recipients.

2 Literature review

Kidney exchange programs were started around 2000 in a few countries and later many other countries also developed their own exchange programs. Initially the exchanges were done as swaps but later it was extended to longer cycles as well. In 2004, Roth et al. [3] considered only two-way exchanges and recipients having 0-1 preferences over available donors. This was the way KEPs were designed and implemented, and later k-length cycles were also considered with weighted preferences over available donors. In 2007, Abraham et al. [4] show that 2-way kidney exchanges can be solved in polynomial time but when the length of exchanges are more than or equal to 3 then this problem of exchange becomes an NP-hard problem and there is no known polynomial time algorithm to solve this problem. They have proposed two IP (Integer Programming) formulations based

on an edge formulation and a cycle formulation. They showed that in restricted cycle length cases, the cycle formulation dominates the edge based formulation. Anderson et. al [5] have proposed two IP formulations for finding long chains and cycles in KEP. They proposed an incremental formulation approach to solve the model. Constantino et al. [6] proposed compact formulations for KEP which reduces the number of constraints by adding some more variables in the model, it had some advantages over earlier proposed IP models.

All these models were IP based and in the worst case, they can take an exponential amount of time to solve to optimality. Researchers have also come up with different heuristic solutions for KEP. In 2013, Dickerson et al. [1] proposed failure aware kidney exchange where they consider the probability of failure after a transplant is proposed. They defined expected utility of edges, cycles and chains, and showed that *there exists a non-maximum cardinality matching that provides linearly more utility than all maximum cardinality matchings.* In 2015, Nickholds and Mak-Hau [7] consider KEP as a multi-criteria problem and proposed a two-stage heuristic for it. In the first stage, they find the optimal number of transplants without considering the weight of matches and in the second stage, they maximize the sum of weighted matches subject to the constraint that the number of transplants should be equal to that of the first stage. They proposed Random Ascent and Steepest Ascent heuristic to solve multi-criteria KEP.

Another heuristic approach for finding good quality solutions was used in the Spanish kidney exchange program. Bofill et al [2017] proposed a greedy algorithm which gives preference to 2-way cycles over 3-way cycles as their probability of failures is lower.

The heuristic works as follows:

Step 1: Create a list C of all possible 2-way cycles followed by all possible 3-way cycles in order of decreasing weights.
Step 2: Select the top cycle from C and add it to the solution set S and remove cycles which are non-disjoint to the selected cycle.
Step 3: Repeat this process until no cycle remains in C.
Step 4: Try to convert 2-way cycle to 3-way cycle by adding a non assigned vertex.
Step 5: Try to convert 3-way cycles to two disjoint 2-way cycles by adding an unassigned vertex to them.

Not surprisingly, IP based formulations (with restrictions on cycle lengths) sometimes works slightly better than their greedy approach in terms of the number of transplants because they allow all possible 3-way and 2-way cycles. The heuristic algorithm works well as it tries to maximize the number of 2-way cycles, which would have lower failure probabilities.

The above heuristics do not prioritize sensitized recipients. Sensitized recipients who are blood group compatible, but are hard to match because of cross match and HLA factors are crucial participants in multi-way (including 3 way) cycles. Since these are hard-to-match recipients, we need to prioritize them and

find solutions which can increase the probability of a successful transplant which involves them. This is the basis of our heuristic, described in the next section.

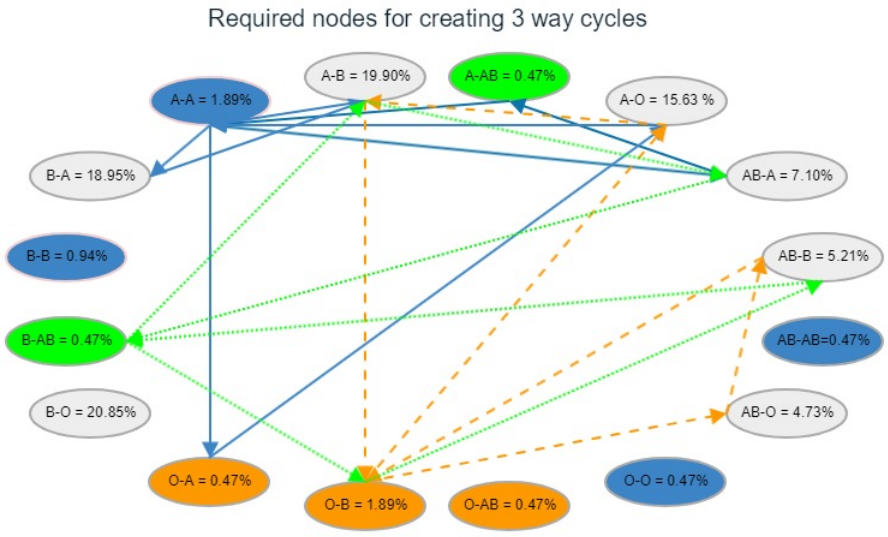
3 A heuristic solution based on blood group distribution

In a typical PKE registry, there are a large number of O blood group (BG) type recipients and very few O BG type donors. This happens due to the compatibility of blood groups as O is a universal donor which allows a donation to any BG recipient. Similarly, there would be many AB type donors and very few AB type recipients, this also happens because of BG compatibility as AB is a universal recipient. Pairs involving O donors or AB recipients usually come to the incompatible registry if the cross-match between the donor and the recipient become positive which restrict the transplant to be performed. In these cases, recipients becomes sensitized to a certain profile of the donor. More care is required to find matches for these recipients.

Now in such registries, for creating a 3-way cycle, three types of pairs are needed as part of such cycles.

- a pair with an O donor
- a pair with an AB recipient
- similar BG type donor-recipient pairs (A-A, B-B)

These type of pairs join the registry either due to cross-match positivity or to improve the quality of matches. In both cases, we need to ensure that these recipients get a priority as they come up with a possibility of an increased number of transplants within the registry. Also, this will increase the probability of successful transplant as these are hard to match recipients. A typical blood group distribution of a paired exchange registry is as follows:



We use this information to develop a good quality solution. Pairs which are needed to create 3-way exchanges are considered as crucial pairs and they should be given a priority as they can increase the welfare of the registry. Prioritizing hard to match recipients will increase the compatibility and weights for them and they will receive a better HLA (Human Leukocyte Antigen) match, PRA (Panel-Reactive Antibody) levels and smaller age differences. These are a few crucial parameters which defines the quality of matches and thus it will increase the probability of successful execution of the proposed transplants. Now we propose an algorithm which prioritize these pairs to create 3-way exchanges and remaining pairs can be solved for 2-way exchanges within registry. The algorithm works as follows:

Step 1: Find a maximum weighted 3-way cycle containing sensitized recipient and add it to a solution set S.
Step 2: Repeat this process until no 3-way cycle is remaining.
Step 3: If there are sensitized recipients left in the registry, find maximum weighted 2-way cycle and add it to the solution set S and repeat this process until no possible match is there for the sensitized recipient.
Step 4: The remaining data will have only 2-way swaps and this is solved for max weighted cardinality.

Finding a 3-way cycle over a network is a polynomial time problem and for 2-way exchanges, maximum weighted and maximum cardinality is also a polynomial time problem. Thus the overall procedure will be polynomial time problem which can typically be solved faster than an IP based formulation. Since we are prioritizing the sensitized recipients, the probability of execution of the proposed solution will be high.

4 Simulation study

The proposed heuristic was compared with the IP formulations for a few data sets. Since the heuristic tries to maximize the weights for hard to match recipients, it was not expected to be optimal in terms of the number of transplants but it should be close enough to optimal solution and have a higher chance of successful execution. The following table compared the heuristic approach with exact IP formulations in terms of the number of 2-way and 3-way cycles.

Table 1

	Proposed heuristic		Exact IP formulation	
Data set	$T(C_3)$	$T(C_2)$	$T(C_3)$	$T(C_2)$
Pairs = 10	0	3	0	3
Pairs = 20	0	4	0	4
Pairs = 30	2	4	3	3
Pairs = 40	2	4	2	5

$T(C_3)$ = Total number of 3 way cycles, $T(C_2)$ = Total number of 2 way cycles

Here it can be observed that the difference between the proposed heuristic and the exact IP is small. The way the heuristic approach works will increase the probability of successful execution of transplants for hard to match recipients. Prioritizing hard to match recipients will increase the overall weights for these recipients and thus they will have better HLA matches, PRA levels and small age differences. Better PRA matches increases the possibility of successful execution of proposed transplants. Thus in comparison to non prioritized methods, this method is expected to have higher number of successful transplants for hard to match recipients which will increase the overall number of successful executions.

5 Conclusion

As successful implementation of proposed exchanges are very few in practice due to several reasons, we try to increase the probability of successful transplant by prioritizing the hard-to-match recipients, who are essential parts of solutions which contain 3-way cycles. We propose a heuristic algorithm which uses the blood group distribution of the PKE registry and find a solution which increase the weights for hard to match recipients. Integer Programming based formulations have exponential running time in the worst case, whereas the proposed algorithm has a smaller running time as compared to earlier proposed algorithms. Comparison of the heuristic algorithm and exact methods show that the difference between both approaches very small in terms of the number of transplants, while leading to a higher probability of successful transplants in practice.

References

1. Dickerson, J.P., Procaccia, A.D. and Sandholm, T., Failure-aware kidney exchange, *EC '13 Proceedings of the fourteenth ACM conference on Electronic commerce*, 323-340, 2013.
2. Bofill, M., Marcos, C., Francesc, C., et al. The Spanish Kidney Exchange Model: Study of Computation-Based Alternatives to the Current Procedure, *Artificial Intelligence in Medicine*, 272-277, 2017.
3. Roth, A.E., Sonmez, T. and Unver, M.U., Kidney Exchange, *Quarterly Journal of Economics*, 457-488, 2004.
4. Abraham, D.J., Blum, A. and Sandholm, T., Clearing Algorithms for Barter Exchange Markets: Enabling Nationwide Kidney Exchanges, *8th ACM conference on Electronic commerce*, 295-304, 2007.
5. Anderson, R., Ashlagib, I., Gamarnik, D. and Roth, A.E., Finding long chains in kidney exchange using the traveling salesman problem, *PNAS Early Edition* 2014.
6. Constantino, M., Klimentova, X., Viana, A. and Rais, A., New insights on integer-programming models for the kidney exchange problem, *European Journal of Operational Research*, 57-68, 2013.
7. Nickholds, L. and Mak-Hau, V., Heuristic approaches for multi-criteria optimisation in kidney exchange programs, *Proceedings of the 21st International Congress on Modelling and Simulation, Modelling and Simulation Society of Australia and New Zealand*, 1780-1786, 2015.