CTW16

14th Cologne-Twente Workshop on Graphs and Combinatorial Optimization

Università degli Studi di Milano
Gargnano (BS), Italy
June 6-8, 2016
ctw16.di.unimi.it

Proceedings of the Conference

Alberto Ceselli, Roberto Cordone, Giovanni Righini (Eds.)

UNIVERSITÀ DEGLI STUDI
DI MILANO
DIPARTIMENTO DI INFORMATICA

European Association for
Theoretical Computer Science
Italian Chapter

Organization

The CTW16 is organized by the Optimization Laboratory (OptLab) at Department of Computer Science of Università degli Studi di Milano.

Scientific Committee:

- Ali Fuat Alkaya (U Marmara)
- Alberto Ceselli (U Milano)
- Roberto Cordone (U Milano)
- Ekrem Duman (U Ozyegin)
- Ulrich Faigle (U Cologne)
- Johann L. Hurink (U Twente),
- Leo Liberti (École Polytechnique, Paris)
- Bodo Manthey (U Twente)
- Gaia Nicosia (U Roma Tre)
- Andrea Pacifici (U Roma Tor Vergata)
- Stefan Pickl (UBw München)
- Giovanni Righini (U Milano)
- Rainer Schrader (U Cologne)
- Rüdiger Schultz (U Duisburg-Essen)

Local Organization:

- Saverio Basso
- Alberto Ceselli
- Roberto Cordone
- Marco Premoli
- Giovanni Righini
- Andrea Taverna

Table of Contents

Combinatorial Optimization Track.

Approximation Algorithm
Mon 6, 09:00-10:30.

Heuristics 1
Mon 6, 11:00-13:00

Heuristics 2
Mon 6, 14:30-16:30

Combinatorial Optimization
Mon 6, 17:00-18:30

Graph Theory Track

Connectivity on Graphs
Mon 6, 09:00-10:30

Graphs Coloring
Mon 6, 11:00-13:00

Graph Optimization
Mon 6, 14:30-16:30

Game Theory
Mon 6, 17:00-18:30

Computational Complexity
Tue 7, 09:00-10:30

Graph Structures
Tue 7, 11:00-13:00

Computational Complexity
Wed 8, 09:00-10:30

Algebraic Structures
Wed 8, 11:00-13:00

The Shared Broadcast Tree Problem and MST

Marika Ivanova [1]

Department of Infomatics,
University of Bergen,
Bergen, Norway

Abstract

The shared broadcast tree (SBT) problem in Euclidean graphs resembles the minimum spanning tree (MST) problem, but differs from MST in the definition of the objective function. The SBT problem is known to be NP-hard. In the current work, we analyse how closely the MST-solution approximates the SBT-solution, and we prove in particular that the approximation ratio is at least 6. Further, we conduct numerical experiments comparing the MST-solution and the optimum. The results show that the cost of the MST-solution is around 20% higher than the optimal cost.

Keywords: shared broadcast tree, MST, approximation algorithm

1 Introduction

The purpose of a broadcast communication in a wireless ad-hoc network is to route information from one source node to all other nodes. Given a set of devices and distances between them, the task is to assign a power to each node, so that the communication demands are met and the energy consumption is minimized, assuming their locations are fixed. The devices are able to both transmit and receive a signal, as well as dynamically adjust their power level.

[1] Email: `marika.ivanova@uib.no`

Omnidirectional antennas are used, and hence a message reaches all nodes within the communication range given by a power assigned to the sender, i.e. the maximum of the powers necessary to reach all intended recipients.

Minimum Energy Broadcast [3] (MEB) is the problem of constructing an optimal arborescence for broadcasting from a given source to all remaining nodes, such that the total power consumption is minimized. A separate tree has to be stored for each source. The idea of SBT [2][4] is to construct a common source-independent tree, instead of a set of individual arborescences. The power levels then depend merely on the immediate neighbour from which a message is received. This idea is based on the observation that a forwarded signal does not have to reach the neighbour from which it originally came.

The decentralized nature of wireless ad-hoc networks implies its suitability for applications, where it is not possible to rely on central nodes, or where network infrastructure does not exist. This is typical for various short-term events like conferences or fixtures. Simple maintenance makes them useful in emergency situations, military conflicts, and home networking.

We model a wireless network as a complete graph $G = (V, E)$, where V corresponds to the network nodes (points in $\mathbb{R}^2$), and the edges E correspond to the potential links between them. The energy requirement for transmission from i to j is denoted by $p_{ij} = \kappa d_{ij}^{\alpha}$, where d_{ij} is the Euclidean distance between i and j, α is an environment-dependent parameter (typically valued between 2 and 4) and κ is a constant. In this work, we use $\alpha = 2$ and $\kappa = 1$. Let $T = (V, E_T)$, $E_T \subseteq E$ be a spanning tree of G. Then $T_{i/j}$ denotes the subtree of T consisting of all vertices k such that the path from k to j visits i, as introduced in [4]. For a non-leaf node i in T, i_1 and i_2 denote the first and the second most distant neighbour of i in T, respectively. If i is a leaf, i_2 is not defined, and we let $p_{ii_2} = 0$. If a message is generated at a node k in $T_{i_1/i}$ then i needs power p_{ii_2} to relay the message to i_2 and other neighbours in $T \setminus T_{i_1/i}$. Power p_{ii_1} is needed to relay messages initiated in $T \setminus T_{i_1/i}$. Assuming that all nodes initiate messages equally frequently, the SBT problem is to construct a spanning tree T minimizing the objective function

$$P(T) = \sum_{i \in V} |T_{i_1/i}| p_{ii_2} + |T \setminus T_{i_1/i}| p_{ii_1}. \qquad (1)$$

2 MST as an approximate solution to the SBT problem

Since the SBT problem is NP-hard, inexact solutions are often considered. Because any spanning tree is a feasible solution, the MST-solution yields one such approximation. This approach is also valid for MEB, where MST approximates the optimum with factor 6 [1]. We define the *MST approximation ratio* ρ as the supremum, taken over all SBT instances, of the ratio between the power consumptions in the MST solution and an optimal SBT.

Theorem 2.1 *The MST approximation ratio for SBT is at least 6.*

Proof. For an integer $k \geq 2$, let G_k be a complete Euclidean graph with a node o located in the center of a unit circle, nodes $t_1, \ldots, t_6$ evenly distributed on the circumference, and nodes $s_{i1}, \ldots, s_{ik}$, $(i = 1, \ldots, 6)$, evenly distributed on the radial line $[o, s_{ik}] \subset [o, t_i]$, where s_{ik} is located $1/k$ units from o. Thus, since arc costs p_{uv} are the square of arc lengths d_{uv}, we have $p_{uv} = 1/k^4$ for $u = s_{ij}$, $v = s_{i,j+1}$, whereas $p_{uv} = (1 - 1/k)^2$ for $u = s_{ik}$, $v = t_i$. A possible MST (denoted T_k) of G_k consists of the 6 paths $(o, s_{i1}, \ldots, s_{ik}, t_i)$. For this tree, the objective function (1) evaluates to

$$P(T_k) = \underbrace{6\left(1 - k^{-1}\right)^2}_{t_i} + \underbrace{6\left[(6k+6)\left(1 - k^{-1}\right)^2 + k^{-4}\right]}_{s_{ik}} + \underbrace{(6k-5)(6k+7)k^{-4}}_{o, s_{i1}, \ldots, s_{i,k-1}}.$$

Another spanning tree of G_k is the star T_k^* centered at node o. For this solution, (1) evaluates to

$$P(T_k^*) = \underbrace{6}_{t_i} + \underbrace{6\sum_{i=1}^{k}\left(i\frac{1}{k^2}\right)^2}_{s_{i1}, \ldots, s_{ik}} + \underbrace{6k+7}_{o}.$$

Thus, the MST-approximation ratio satisfies $\rho \geq \frac{P(T_k)}{P(T_k^*)}$. Since $\lim\limits_{k \to \infty} \frac{P(T_k)}{P(T_k^*)} = 6$, the claim follows. □

3 Numerical Experiments

The SBT problem can be modelled as a MILP [2][4], and moderately sized instances can be solved. We have generated instances of a specific number of nodes with random coordinates distributed uniformly on a square, and compared the MST-solution to the optimal one. The MILP solver CPLEX is used to compute the optimal solution. Each number of nodes is tested in 100 instances. Although the theoretical approximation ratio suggests that MST is not very suitable for SBT, the experimental results summarized in Tab. 1 reveal that in practice, MST represents a feasible solution with objective value approximately 1.2 times the optimum. This factor does not seem to change much with growing number of nodes. However, calculation of the optimum

Table 1
Average SBT costs of MST and optimal solutions for various instance sizes.

Number of nodes	10	12	14	16	18	20
$P(\text{OPT})$	46268	56060	66747	69727	84250	94039
$P(\text{MST})$	9432	68833	80195	84262	101816	119679
$P(\text{MST})/P(\text{OPT})$	1.198	1.232	1.206	1.210	1.209	1.271

for larger instances takes prohibitively long time, so we have access only to limited data. The largest ratio observed in the experiments is 1.59.

4 Conclusion and Future Work

This paper studies the relation between MST and the optimal solution to SBT in terms of the objective value. It has been shown that the MST approximation ratio is at least 6. Numerical experiments suggest that even though there are instances where MST is nearly 60% above the optimum, it represents a good solution in the vast majority of cases. The current research leads to several interesting questions that merit further investigation. A prominent question is whether there exists a constant upper bound on the MST-approximation ratio. For the related MEB problem, approximation algorithms with constant performance guarantee are well studied. Adapting these methods and the corresponding analysis to SBT is a research question to be pursued.

References

[1] Ambühl, C.: An Optimal Bound for the MST Algorithm to Compute Energy Efficient Broadcast Trees in Wireless Networks. Automata, Languages and Programming. 32nd International Colloquium, ICALP 2005, Lisbon, Portugal. 1139–1150 (2005)

[2] Papadimitriou, I., and Georgiadis, L.: Minimum-energy Broadcasting in Multi-hop Wireless Networks Using a Single Broadcast Tree. Mobile Networks and Applications, 11, 3 361–375 (2006)

[3] Wieselthier, J. E., Nguyen, G. D., Ephremides, A.: Energy-efficient broadcast and multicast trees in wireless networks. Mob. Netw. Appl. 7, 6, 481-492 (2002)

[4] Yuan, D., Haugland, D.: Dual Decomposition for Computational Optimization of Minimum-Power Shared Broadcast Tree in Wireless Networks. IEEE Transactions on Mobile Computing, Vol 12, no 11. 2008–2019 (2012)

Bounded-Degree Spanning Trees with Nodes of Degree One

Walter Kern and Bodo Manthey

University of Twente
Enschede, Netherlands

Abstract

We present a constant factor approximation algorithm for the following problem: given a connected graph $G = (V, E)$ with non-negative edge weights that satisfy the triangle inequality, find a minimum weight spanning tree that respects prescribed upper bounds on the vertex degrees. Using this approximation algorithm, we obtain constant factor approximation algorithms for the problem of computing connected d-factors of minimum weight for prescribed vertex degrees $d = (d_i)_{i \in V}$. Constant factor approximation algorithms for these problems were known only for the case that $d_i \geq 2$ for all $i \in V$.

1 Introduction

Finding low-cost spanning subgraphs with prescribed degree and connectivity requirements is a fundamental problem in the area of network design. The goal is to find a cheap, connected subgraph that meets the degree constraints. Most variants of such problems are NP-hard. Because of this, finding good approximation algorithms for such network design problems has been the topic of a significant amount of research. In this paper, we study the problem of finding low-cost spanning connected subgraphs with degree constraints, where violation of the degree constraint is not allowed. The degree constraints are either upper bounds or have to be met exactly.

Minimum-weight subgraphs with prescribed vertex degrees can be found efficiently, but asking for connectedness in addition makes the problem NP-hard [1]. Also finding spanning trees with given upper bounds for the degrees of the nodes is NP-hard [5].

A main obstacle for approximation algorithms for these problems seem to be vertices that are required to have degree 1. In fact, existing approximation algorithms [3,4,6] only work when the minimum degree requirement is at least

Preprint submitted to CTW 2016 *26 February 2016*

2, and it has been raised as an open problem [4,6] to approximate network design problems in the presence of vertices that must have degree 1.

Problem Definition. Instances of the problems that we consider consist of a simple undirected complete graph $G = (V, E)$ with edge weights w that satisfy the triangle inequality and given $d = (d_i)_{i \in V}$ to be interpreted as either prescribed vertex degrees or upper bounds thereof. For $F \subseteq E$, let $\deg_F(i)$ be the degree of node $i \in V$ in the graph (V, F). Furthermore, $w(F) = \sum_{e \in F} w(e)$ is the total weight of the edge set F.

In the *bounded-degree minimum spanning tree problem* (denoted by BMST), we are to compute a tree $T \subseteq E$ of minimum weight with the additional condition that $\deg_T(i) \leq d_i$ for all $i \in V$. We call such a tree *d-bounded*. We denote a minimum weight d-bounded tree by Tree_d, breaking ties arbitrarily.

In the *connected factor problem* (denoted by $\mathsf{ConnFact}$), our goal is to compute a connected, simple d-factor F of minimum weight. This means that (V, F) must be connected and $\deg_F(i) = d_i$ for all vertices $i \in V$.

Our Contribution. We give an affirmative answer to the question raised by Fukunaga and Nagamochi [4] and Cornelissen et al. [2] whether constant factor approximation algorithms also exist in case some of the d_i are equal to 1.. We present a factor 3-approximation algorithm for BMST. Then we use this algorithm to get factor 7 approximation algorithms for $\mathsf{ConnFact}$.

2 Bounded-Degree Spanning Trees

We start with a simple observation, based on the standard construction of Hamilton paths by doubling a minimum spanning tree.

Lemma 1 *Given an undirected, complete graph G with edge weights w and an edge $e_0 = \{i_0, j_0\} \in E$, we can compute in polynomial time a Hamiltonian path P with endpoints i_0 and j_0 such that $w(P) \leq 2w(T)$, where $T \subseteq E$ is a minimum weight spanning tree containing e_0.*

In what follows, we distinguish between nodes with prescribed degree $d_i = 1$ and other nodes. Therefore, we define $V_{=1} = \{i \in V \mid d_i = 1\}$ and $V_{\geq 2} = \{i \in V \mid d_i \geq 2\}$. Any d-bounded tree T consists of an *interior tree* T_{int} that connects only the $V_{\geq 2}$ nodes and to which the $V_{=1}$ nodes are attached. We may assume that T_{int} connects at least two nodes. Otherwise, $|V_{\geq 2}| \leq 1$ and the problem becomes trivial. The most challenging part is to determine how the vertices in $V_{=1}$ are attached to the interior tree.

To address this problem, we proceed in two steps. In the first step, we compute a forest that spans all of $V_{=1}$ and a subset of $V_{\geq 2}$ without violating the degree constraints. In the second step, we connect the connected components of this forest along a Hamiltonian path through a subset of the $V_{\geq 2}$ nodes. The forest in the first step is computed by solving an appropriate minimum-cost flow problem.

Let us describe the first step. In what follows, we assume that we know an edge $e_0 = \{i_0, j_0\} \in \mathrm{Tree}_d$ in the interior tree of the unknown optimum solution Tree_d. (In our algorithm, we fix $i_0 \in V_{\geq 2}$ arbitrarily, try all possible choices of $j_0 \in V_{\geq 2} \setminus \{i_0\}$, and take the best outcome.) Removing e_0 splits the unknown tree Tree_d into two subtrees. To outline the intuition behind our approach, consider i_0 and j_0 as the roots of these subtrees, and direct all edges in these two subtrees towards i_0 and j_0, respectively. We may interpret the subtrees as "flows" from the $V_{=1}$ nodes towards the roots i_0 and j_0, respectively. In this sense, the two subtrees define a solution to the flow problem (with node capacities) described below.

Consider the following flow problem MCF_{e_0}: The underlying graph has vertex set $V \cup \{r\}$, where $r \notin V$ is a new node, and edge set $(E \setminus \{e_0\}) \cup \{\{i_0, r\}, \{j_0, r\}\}$. All edges $e \in E \setminus \{e_0\}$ have a capacity of 1 in both directions and costs of w_e per unit of flow. Each node $i \in V_{\geq 2}$ has a node capacity of $d_i - 1$ (this means that at most $d_i - 1$ units of flow may pass through i). In addition, there are *overflow edges* $\{i, r\}$ for $i \in V_{\geq 2}$, which have cost 0. For $i \in V_{\geq 2} \setminus \{i_0, j_0\}$, edge $\{i, r\}$ has a capacity of $d_i - 2$. For $i \in \{i_0, j_0\}$, edge $\{i, r\}$ has a capacity of $d_i - 1$. The task is to find a min cost flow from the $V_{=1}$ nodes, each having a supply of 1, to the new root node r, which has a demand of $|V_{=1}|$.

The set $\mathrm{Tree}_d \setminus \{e_0\}$ defines a solution f_{Tree} of this flow problem as follows: Recall that we direct all edges in the two subtrees of $\mathrm{Tree}_d \setminus \{e_0\}$ towards their roots i_0 or j_0, respectively. On every arc $e = \{i, j\}$ in the directed tree $\mathrm{Tree}_d \setminus \{e_0\}$, we have a flow of 1 (towards i_0 or j_0). Thus, in particular, each $i \in V_{=1}$ has an outflow of 1. If a node $i \in V_{\geq 2} \setminus \{i_0, j_0\}$ has degree ℓ $(2 \leq \ell \leq d_i)$ in Tree_d, then in the directed tree, it has $\ell - 1$ incoming arcs and one outgoing arc (in direction to the root i_0 or j_0). Thus its total inflow equals $\ell - 1$ and we send $\ell - 2$ units of outflow directly to r on the overflow arc from i to r. Note that the node capacity constraint (throughput at most $d_i - 1$) is met. If $i \in \{i_0, j_0\}$ has degree ℓ $(2 \leq \ell \leq d_i - 1)$ in Tree_d, then its inflow equals ℓ units, which we route to r on the overflow arc $\{i, r\}$. This, again, also respects the node capacity constraints.

Lemma 2 *Let $f^\star$ be an integral optimum solution of MCF_{e_0} with minimum support $S^\star$ (which can be computed efficiently). Then we have the following properties:*

(i) $w(S^\star) \leq w(\mathrm{Tree}_d)$.

(ii) $S^\star$ *is a forest.*

(iii) $\deg_{S^\star}(i_0) \leq d_{i_0} - 1$ *and* $\deg_{S^\star}(j_0) \leq d_{j_0} - 1$.

(iv) Each connected component of $S^\star$ contains i_0 or j_0 or a node $i \in V_{\geq 2}$ with

$\deg_{S^\star}(i) \leq d_i - 2$.

Given $S^\star$, as in Lemma 2, we connect the connected components via a Hamilton path P with endpoints i_0 and j_0 as in Lemma 1: In each component of $S^\star$ that contains neither i_0 nor j_0, we pick a "root" node i of degree at most $d_i - 2$ in $S^\star$. Then we connect the components of $S^\star$ by following P, starting in i_0, ending in j_0 and skipping all other vertices except the root nodes chosen. This yields a d-bounded tree T of weight $w(T) \leq w(S^\star) + w(P) \leq w(\text{Tree}_d) + 2(\text{Tree}_d) \leq 3w(\text{Tree}_d)$. Putting this together yields the following result.

Theorem 3 *There is a polynomial-time 3 approximation for* BMST*.*

3 Connected Factors

The idea to approximate connected factors is as follows: we compute a d-factor F, which is not necessarily connected, and a d-bounded tree T. As long as the d-factor F is not connected, there exists an edge $e \in T \setminus F$ that we can add. In order to maintain the degrees, we remove one edge of each endpoint of e and add the edge e plus another edge.

Theorem 4 *There is a polynomial-time 7 approximation for* ConnFact*.*

The algorithm above also works for the variant of ConnFact, where we allow multiple edges. We just have to replace the initialization of F by a minimum-weight d-factor where multiple edges are allowed. In this way, we obtain a factor 7 approximation also for this variant of the problem.

References

[1] F. Cheah and Derek G. Corneil. The complexity of regular subgraph recognition. *Discrete Appl. Math.*, 27(1-2):59–68, 1990.

[2] Kamiel Cornelissen, Ruben Hoeksma, Bodo Manthey, N. S. Narayanaswamy, and C. S. Rahul. Approximability of connected factors. *Proc. WAOA 2013*, LNCS 8447, pp. 120–131. Springer, 2014.

[3] Sándor P. Fekete, Samir Khuller, Monika Klemmstein, Balaji Raghavachari, and Neal E. Young. A network-flow technique for finding low-weight bounded-degree spanning trees. *J. Algorithms*, 24(2):310–324, 1997.

[4] Takuro Fukunaga and Hiroshi Nagamochi. Network design with edge-connectivity and degree constraints. *Theory Comput. Syst.*, 45(3):512–532, 2009.

[5] Martin Fürer. Degree-bounded trees. In Ming-Yang Kao, editor, *Encyclopedia of Algorithms*, pages 231–233. Springer, 2008.

[6] Bodo Manthey and Marten Waanders. Approximation algorithms for k-connected graph factors. *Proc. WAOA 2015*, LNCS. Springer, to appear.

Scheduling parallel jobs on heterogeneous platforms

Klaus Jansen [1,2]

Department of Computer Science
University of Kiel
Kiel, Germany.

Denis Trystram [3]

Laboratoire d'Informatique
University of Grenoble
Grenoble, France.

Abstract

We consider the problem of scheduling parallel jobs on heterogeneous platforms. Given a set $\mathcal{J}$ of n jobs where each job $j \in \mathcal{J}$ is described by a pair (p_j, q_j) with a processing time p_j and number q_j of processors required and a set of N heterogeneous platforms P_i with m_i processors, the goal is to find a schedule for all jobs on the platforms minimizing the maximum completion time. The problem is directly related to a two-dimensional multi strip packing problem. Unless $P = NP$ there is no approximation algorithm with absolute ratio better than 2 for the problem. We propose an approximation algorithm with absolute ratio 2 improving the previously best known approximation algorithms. This closes the gap between the lower bound of < 2 and the best approximation ratio.

Keywords: scheduling parallel tasks, strip packing, approximation algorithms.

1 Introduction

We study the problem of scheduling parallel jobs on heterogeneous platforms. The input consists of a set $\mathcal{J} = \{1, \ldots, n\}$ of n jobs and a set $\mathcal{B}$ of N platforms $P_1, \ldots, P_N$, where each P_i consists of a set $M_i = \{1, \ldots, m_i\}$ of processors for $i \in [N] := \{1, \ldots, N\}$. The width of the platform P_i is the number m_i of processors. Each job $j \in \mathcal{J}$ is described by a pair (p_j, q_j) with a processing time (or height) $p_j \in \mathbb{N}$ and number of processors (or width) $q_j \in \mathbb{N}$ required to execute j. If all numbers m_i are equal, we have identical platforms. In the general case the numbers m_i may be different and the machines are called heterogeneous platforms. For simplification we suppose that $m_1 \geq m_2 \geq \ldots \geq m_N$. A schedule is an assignment $a : \mathcal{J} \rightarrow \mathbb{Q}_{\geq 0} \times \cup_{i=1}^{N} 2^{M_i}$ that assigns every job j to a starting time $t_j = a_1(j)$ and to a subset $A_j = a_2(j) \subseteq M_i$ of processors of a platform P_i such that $|A_j| = q_j$. A job j can only be executed in platform P_i if the width of the platform $m_i \geq q_j$. A schedule is feasible if every processor in every platform executes at most one job at any time. The goal is to find a feasible schedule with minimum total length or makespan $\max_{i \in [N]} C_{max}(P_i)$ where $C_{max}(P_i) = \max_{j | A_j \subseteq M_i} t_j + p_j$ is the local makespan on platform P_i (or height of platform P_i). The optimum value for an instance $(\mathcal{J}, \mathcal{B})$ is denoted by $OPT(\mathcal{J}, \mathcal{B})$.

2 Previous and new Results

Table 1
Approximation algorithms for heterogeneous platforms.

		ratio	constraints
Tchernykh et al. [7]	2005	10	none
Schwiegelshohn et al. [6]	2008	3	non-clairvoyant
Tchernykh et al. [8]	2010	$2e + 1$	release dates
Bougeret et al. [1]	2010	2.5	$\max q_j \leq \min m_i$
Dutot et al. [2]	2013	$(2 + \epsilon)$	none
Jansen and Trystram**(new result)**	2016	2	none

[1] Research supported by German Research Foundation (DFG), project Ja 612/12-2.

[2] Email: kj@informatik.uni-kiel.de
[3] Email: trystram@imag.fr

A typical application in grid computing is described in [8]. First, parallel jobs are allocated to suitable clusters and, afterwards, the allocated jobs are locally scheduled on the corresponding assigned clusters. The scheduling problem is also related to a multiple strip packing problem. By a reduction from 3-Partition, Zhuk [9] proved that there is no approximation algorithm with absolute approximation ratio better than 2 for packing rectangles with height 1 into multiple strips. This reduction shows also that there is no approximation algorithm with ratio better than 2 for scheduling parallel jobs on identical platforms, where $m_1 = \ldots = m_N$. For the general problem, Tchernykh et al. [7] presented an algorithm with absolute ratio 10. Earlier Remy [5] claimed that the approximation ratio 2 of List Schedule is preserved when applied to the problem with identical platforms while in [7] and again later in [6] it is shown that List Schedule cannot even guarantee a constant approximation ratio for this problem. On the other hand, several improved approximation algorithms for the scheduling problem have been proposed. In Table 1 we give an overview about the known approximation algorithms for heterogeneous platforms. Remark that in [6], the algorithm is an online non-clairvoyant algorithm where processing times are not available in advance. The algorithm in [1] works only under the constraint where the maximum required number of processors $\max q_j$ is at most the minimum number of processors $\min m_i$ among all platforms, while the algorithm in [8] works for the general problem with additional release dates. Currently, the best known absolute ratio of an approximation algorithm [2] for the general problem with heterogeneous platforms is $(2+\epsilon)$. The running time of the algorithm is $g(1/\epsilon)n^{O(f(1/\epsilon))}$ for some functions f and g. In this paper we propose a polynomial time algorithm with absolute ratio 2. This closes the gap between the lower bound of < 2 and the currently best absolute ratio $(2+\epsilon)$.

Theorem 2.1 *There is an approximation algorithm that for a set $\mathcal{J}$ of n parallel jobs and a set $\mathcal{B}$ of N heterogeneous platforms generates a schedule for the jobs with makespan at most $2\ OPT(\mathcal{J}, \mathcal{B})$. The running time is polynomial in n.*

3 Methods and Techniques

In order to obtain an approximation algorithm with absolute ratio 2, we use the following approach. Our new algorithm works in two phases. By scaling we may assume that $OPT \leq 1$. In the first phase we use a slight modification of the $(2+\epsilon)$-approximation algorithm in [2]. Depending on four cases, the algorithm in the first phase generates a solution where the makespan on some platforms is bounded by $(1+\epsilon)$ and on other platforms by 2 while a constant number of sets of jobs is non-assigned to the platforms. Our previous algorithm places these sets onto the first group of platforms causing a makespan of $(2+\epsilon)$. Instead of this approach, our new developed method converts the approximate solution of the first phase with fixed $\epsilon = 1/10$ into a 2-approximate solution. To achieve this goal we clever re-schedule in the second phase jobs on the platforms and insert the sets of non-assigned jobs of the first phase. The details can be found in [3].

References

[1] Bougeret, M., P.-F. Dutot, K. Jansen, C. Otte, and D. Trystram., *A fast* 5/2-*approximation algorithm for hierarchical scheduling*, European Conference on Parallel and Distributed Computing (Euro-Par 2010), Ischia, Springer LNCS 6272, 157–167.

[2] Dutot, P.F., K. Jansen, C. Robenek, and D. Trystram., *A* $(2+\epsilon)$ *- approximation for scheduling parallel jobs in platforms*, European Conference on Parallel and Distributed Computing (Euro-Par 2013), Aachen, Springer LNCS 8097, 78–89.

[3] Jansen, K., and D. Trystram, *Scheduling parallel jobs on a network of heterogeneous platforms*, University of Kiel, Dept. of Computer Science, Technical Report No. 1502 (2015).

[4] Remy, J., *Resource constrained scheduling on multiple machines*, Information Processing Letters, **91** (2004), 177–182.

[5] Schwiegelshohn, U., A. Tchernykh, and R. Yahyapour, *Online scheduling in grids*, IEEE International Parallel and Distributed Processing Symposium (IPDPS 2008), Miami, 1–10.

[6] Tchernykh, A., J. Ramirez, A. Avetisyan, N. Kuzjurin, D. Grushin, and S. Zhuk, *Two level job-scheduling strategies for a computational grid*, Conference on Parallel Processing and Applied Mathematics (PPAM 2005), Poznan, Springer LNCS 3911 (2005), 774–781.

[7] A. Tchernykh, A., U. Schwiegelshohn, R. Yahyapour, and N. Kuzjurin, *Online hierarchical job scheduling on grids with admissible allocation*, Journal of Scheduling **13(5)** (2010), 545–552.

[8] Zhuk, S.N., *Approximation algorithms to pack rectangles into several strips*, Discrete Mathematics and Applications **16(1)** (2006), 73–85.

Experiments with two heuristic algorithms for the Maximum Algebraic Connectivity Augmentation Problem

Claudia Justel [a,1] Carlos Rocha [a,2] and Emanuelle Chaves [a,3]

[a] *Instituto Militar de Engenharia, Rio de Janeiro, Brasil*

Abstract

In this work we present a heuristic algorithm to solve the Maximum Algebraic Connectivity Augmentation Problem (MACAP). This is an NP-complete problem (proved by Mosk-Aoyama in 2008) and consists in, given a graph, determining the smallest set of edges not belonging to it in such a way that the value of the algebraic connectivity of the augmented graph is maximum. In 2006, Ghosh and Boyd presented a heuristic procedure to solve this problem. This heuristic is an iterative method that selects one edge at a time based on the values of the components of a Fiedler vector of the graph. Our goal is to increase the value of the algebraic connectivity of a given graph by inserting edges based on the eccentricity of vertices. In order to evaluate our algorithm, computational tests comparing it with the Ghosh and Boyd procedure are presented.

Keywords: Graph, Laplacian matrix, Algebraic connectivity, Approximated algorithm.

[1] Email: `cjustel@ime.eb.br`, speaker

[2] Email: `edugaspar@hotmail.com`

[3] Email: `manuenlc@gmail.com`

1 Introduction

The algebraic connectivity, defined as the second smallest eigenvalue of the Laplacian matrix of a graph G, is a spectral invariant widely studied in the literature. This parameter is related to the connectivity of the graph. There are different applications of this parameter in several problems ([4])[7],[8]). In this work, we deal with a NP-complete problem known as the Maximum Algebraic Connectivity Augmentation (MACAP). We present a heuristic procedure, which is an iterative method where edges are added depending on the values of the eccentricities of its endpoints. This strategy is different from the heuristic procedure proposed by Ghosh and Boyd in 2006 [2]. The rest of the paper is organized as follows: in Section 2, the definition of the MACAP and the heuristic algorithm proposed by Ghosh and Boyd are given. Section 3 presents a new heuristic algorithm and experimental results comparing this new algorithm with the approximate algorithm in Section 2. At last, final remarks are presented in Section 4. Basic concepts and notation in Graph and Spectral Graph Theory can be found in [1] and [3].

2 MACAP: Complexity, Ghosh and Boyd's heuristic

Given a graph $G = (V, E)$ and a non-negative integer k, the MACAP consists in determining, among all the subsets of edges in G^C of size at most k, the subset that increases the algebraic connectivity as much as possible. In [4] Mosk-Aoyama proved that the decision problem associated to the MACAP is NP-Complete. The heuristic procedure presented in [2] by Ghosh and Boyd, uses a Fiedler vector in order to determine a set of edges to be included in the input graph G to increase the value of the algebraic connectivity. The notation used is: $G_{base} = (V, E_{base})$ a graph with $|V| = n$, $E_{cand} \subset E^C_{base}$ a subset of candidate edges of size m_c, and a non-negative integer number k, $0 \leq k \leq m_c$. The heuristic, denoted Perturbation Heuristic, PH, chooses k edges in E_{cand} ($E \subseteq E_{cand}$, $|E| = k$) to be inserted in G_{base}. Let $L = L(G)$ be the Laplacian matrix of G, $\lambda_2(G)$ the algebraic connectivity of G and $\mathbf{w} = (w_1, \ldots, w_n)$ a Fiedler vector, an eigenvector associated to $\lambda_2(G)$. The greedy strategy used selects k edges, one at a time, being the edge $(i, j) \in E^C_{base}$ for which the components i and j of the Fiedler vector $\mathbf{w}$ has the greatest value of $(w_i - w_j)^2$. In this case, a set of candidates edges E_{cand} that may be different than the set E^C is considered. Ghosh and Boyd ([2]) discuss the results obtained with their algorithm for 3 types of graphs (randomly generated) and suggest that "a large increase in algebraic connectivity can be obtained by adding a few edges carefully". This suggestion is what motivates our work.

3 Contribution

The Fiedler vector is widely used to produce efficient bipartition of vertices in sets of almost the same size. But the use of the Fiedler vector, as proposed by Ghosh and Boyd, does not guarantee an increase in the value of the algebraic connectivity. It is well known that $\frac{4}{n.diam(G)}$ is a lower bound of the algebraic connectivity of G ([3]). We propose a new procedure, the Eccentricity Heuristic EH, to choose an edge in the complement of the input graph by decreasing the diameter of the graph, in order to improve the lower bound for $\lambda_2(G)$. Following this idea, we determine some conditions to eccentricities at the endpoints of the candidate edge, so that the diameter of the resulting graph will decrease. Given a connected graph $G = (V, E)$, select an edge $e = (a, b) \in E^C$ such that: $e_G(a) = diam(G)$, $e_G(b) = diam(G) - 1$, and $d_G(a, b) = diam(G)-1$; to break ties, we consider $d_G(a)$ and $d_G(b)$ the highest. EH is an iterative procedure, where graph G and non-negative integer k are inputs. At each iteration of the main loop one edge is chosen by the criterion above and then graph G and set E^C_{base} are updated. After k iterations, a set with k edges is returned. Algorithm 1 shows pseudo-code of EH.

```
ALGORITHM 1: EH
Input: $k$, $G_{base} = (V, E_{base})$
Output: $E \subseteq E^C$, $|E| = k$;
G = G_{base};
If $k > |E^C|$ then stop;
$E = \emptyset$;
For $i = 1, \ldots, k$ do
    For $v \in V$ do compute $e_G(v)$;
    $d = diam(G) = \max_{v \in V} e_G(v)$;
    For $v \in V$ do compute $d_G(v)$;
    Determine $(a, b) \in E^C_{base}$ such that
        $d_G(a, b) = d - 1$
        $e_G(a) = d$ and $e_G(b) = d - 1$
        $d_G(a)$ maximum or maximum minus one
        $d_G(b)$ maximum or maximum minus one;
    $E = E \cup \{(a, b)\}$;
    $G = G \cup \{(a, b)\}$;
    $E^C_{base} = E^C_{base} - \{(a, b)\}$;
Return $E$;
```

Some observations about the input graph G for EH are necessary: i) the connected graph G can't be a complete graph; ii) the graph must have vertices with different values of eccentricities (we choose two vertices with conditions over their eccentricities). In order to evaluate the quality of the solutions of the EH, we executed a series of experiments with $k = 1$ and input graphs randomly generated. We consider groups of graphs with 40, 60, 80 and 100 vertices and use the *Nauty* generator of random graphs (*genrang* [6]) to obtain 10 samples of each type of instance for fixed n and m. It is important to notice that, in the case of sparse graphs with $m = n$ generated by *Nauty*, the number of disconected graphs obtained is high. On the other hand, the generation of graphs with $m = \frac{n(n-1)}{4}$ gives a high number of graphs with all the vertices with the same eccentricity. After that, we create two classes of instances. For each value of n, we generate 30 samples of graphs with $m = 6n$ and $m = 10n$, and only consider 10 graphs that verify all the conditions of the algorithm EH (connected and with different values of eccentricities for the vertices) except for the case $n = 80$ and $m = 6n$ in which only 9 of the 30 graphs generated verify all the conditions. For result analysis, we execute two type of experiments. First, we use EH without the lines 11 and 12. The set of edges considered in line 8 of this case is denoted E_1, and the best edge determined is denoted e_1. The second experiment considers EH with all the conditions in lines 9, 10, 11 and 12. In this second experiment, the set of edges considered in line 8 is denoted E_2, and the best edge determined is e_2. Our experiments were run on a PC with Intel(R)Core 5i processor running at 2.4 GHz with 4 GB of RAM and use *Matlab* ([5]) for computing eigenvectors in PH. From the experiments, the instance $n = 100$ and $m = 6n$ achieves the best results for $\Delta\lambda_2$ (the difference between $\lambda_2(G + e)$, with $e = e_1, e_2$ obtained by EH and $\lambda_2(G + e_{PH})$, with e_{PH} obtained by PH), in 90% of the graphs, with average value 0.081 for the first experiment. In all the instances considered, the first experiment presents better results than the second one, and the size of the sets E_1 are in general relatively small (compared to E^C).

4 Final Remarks

In this paper we compare the results obtained with two heuristic algorithms for the MACAP, the EH and the PH, using random graphs generated by *Nauty*. The experiments described show good results with the EH for the graphs considered. We propose as future work to investigate the structures of graphs which present better results in our experiments and perform new experiments with particular families of graphs.

Acknowledgement: The first author was partially supported by CNPq with Grant 305677/2013-6, and the others by PIBITI-CNPq.

References

[1] Diestel, R.D., *Graph Theory*. Springer (2000).

[2] Ghosh, A., Boyd, S. Growing well-connected graphs. *Proceedings of the 45th IEEE Conference on Decision and Control* (2006) pp. 6605-6611.

[3] Moliterno, J.J. *Applications of Combinatorial Matrix Theory to Laplacian Matrices of Graphs*. CRC Press (2012).

[4] Mosk-Aoyama, D., Maximum algebraic connectivity augmentation is NP-hard, *Operations Research Letters* **(**36**)** (2008) pp. 677-679.

[5] Matlab. http://www.mathworks.com/products/matlab/ .

[6] Nauty. http://users.cecs.anu.edu.au/ bdm/nauty/

[7] Olfati Saber,R. Ultrafast Consensus in Small World Networks, *Proceedings of the American Control Conference* **4** (2005) pp. 2371-2378.

[8] Wang, H., Van Mieghem, P. *Algebraic connectivity optimization via link adition Bionetics* (2008), Hyogo, Japan.

Multimode extensions of Combinatorial Optimization problems.

Roberto Cordone[1]

University of Milano, Department of Computer Science, via Comelico 39, 20135 Milano, Italy

Guglielmo Lulli[2]

Lancaster University Management School, LA1 4YX Lancaster, UK

Abstract

We review some complexity results and present a viable heuristic approach based on the Variable Neighborhood Search (VNS) framework for multimode extension of combinatorial optimization problems, such as the the Set Covering Problem (SCP) and the Covering Location Problem (CLP)

Keywords: Combinatorial Optimization, covering problems, complexity, Variable Neighborhood Search.

1 Introduction

Different real life applications motivate the generalization of some combinatorial optimization problems to a *multimode* setting. A primer in this category is the Resource Constrained Project Scheduling problem (RCPSP) [1], which requires to determine the starting times of all activities of a project so as to minimize its total completion time. The multimode extension allows each activity to be executed in one out of a set M of modes.

[1] Email: `roberto.cordone@unimi.it`

[2] Email: `g.lulli@lancaster.ac.uk`

Recently, we have been investigating the multimode setting in other combinatorial optimization problems, e.g., the Set Covering Problem (SCP) [2] and the Covering Location Problem (CLP) [3]. The former combines $|M|$ single mode SCP instances, defined on the same ground set I, which must be covered in all modes. The single mode subproblems are linked by cardinality constraints, which limit the number of modes in which the same column from J can be used. The generalization of the CLP to the multimode setting consists in placing a given number K_m of facilities of each mode $m \in M$ to serve a set I of demand centers that require different types of service. The goal is to maximize the demand coverage over all centers and modes with a cardinality constraint that limits the number of active modes in each facility site.

The decision version of the multimode problems listed above are all $\mathcal{NP}$-complete because they include, as a special case, the corresponding single mode version. However, the introduction of additional modes makes them much more challenging to solve. The higher complexity does not refer only to the worst case analysis (i.e., computational complexity and approximability), but also to the average case (i.e., exact and heuristic algorithms).

In what follows, we focus on the SCP and CLP. We first review some complexity results for their multimode generalizations and then present the computational challenges and a viable heuristic approach based on the Variable Neighborhood Search (VNS) framework.

2 Complexity results

Set covering

The single mode SCP admits a logarithmic approximation guarantee and approximation results are also available for more general covering problems [4]. By contrast, even the feasibility of the multimode SCP is $\mathcal{NP}$-complete.

Theorem 2.1 *It is $\mathcal{NP}$-complete to determine whether a given instance of the multimode SCP is feasible or not, even if $|M| = 2$.*

With the same construction it is possible to prove the inapproximability.

Corollary 2.2 *The* MM-SCP *does not admit any polynomial algorithm with an approximation guarantee, unless $\mathcal{P} = \mathcal{NP}$.*

Facility location

The CLP has a constant approximation guarantee, as discussed in Vohra and Hall [5]. Under mild technical assumptions, we provide two greedy algorithms that compute feasible solutions with a guaranteed approximation for the multimode CLP. Algorithm *Greedy1* selects one column at a time which covers the uncovered set of rows of maximum weight, satisfying the cardinality constraint. Algorithm *Greedy2* first builds a solution like *Greedy1*, but relaxing the cardinality constraints, then retrieves a feasible solution by removing the facilities which leave uncovered the minimum weight set of rows as necessary.

Theorem 2.3 *Algorithm* Greedy1 *computes a solution of* MM-CLP *with a guaranteed approximation factor of*

$$\alpha_1 = \frac{\sum\limits_{m\in M} K_m W_m}{|J|\, W_{\text{tot}}}$$

where $W_m = \sum_{i\in I} w_{im}$ *is the total weight of all rows in mode* $m \in M$ *and* $W_{\text{tot}} = \sum_{m\in M} W_m$ *is the total weight of all rows in all modes.*

When all modes have the same total weight ($W_m = W$) and require the same number of facilities ($K_m = K$), $\alpha_1 = K/|J|$. If all columns can be selected in one single mode ($b_j = 1$), the approximation can be refined.

Corollary 2.4 *If* $K_m = K$ *and* $W_m = W$ *for all* $m \in M$*, and* $b_j = 1$ *for all* $j \in J$*, Algorithm* Greedy1 *provides a constant approximation factor equal to*

$$\alpha_1' = \frac{K}{|J|}\left(\frac{1}{|M|} + \frac{|J|}{K|M|}\ln\frac{1}{1-\frac{K}{|J|}(|M|-1)}\right)$$

Theorem 2.5 *If* $b_{\min} = \min_{j\in J} b_j$ *and* $K_{\min} = \min_{m\in M} K_m$*, Algorithm* Greedy2 *computes a solution of* MM-CLP *with a guaranteed approximation factor of*

$$\alpha_2 = \frac{b_{\min}}{|M|}\left[1-\left(1-\frac{1}{K_{\min}}\right)^{K_{\min}}\right]$$

In Table 1, we summarize the complexity results described so far.

3 Computational experience and heuristic approach

Although the mentioned problems are all NP-hard even in the single mode version, the multimode version experimentally proves much harder. Indeed, whilst state-of-the-art ILP solvers like CPLEX are able to quickly solve to optimality average-sized instances of the single mode versions, the same is not true for the multimode instances. Our computational experience shows that

	Single-mode	Multi-mode
SCP	Feas. easy	Feas. NP-complete
	NP-hard	NP-hard
	log-APX	not APX
MCLP	NP-hard	NP-hard
	APX with α_{sm}	APX with $\alpha_{mm} \approx \alpha_{sm}/\lvert M \rvert$

Table 1
Comparison of complexity between singlemode and multimode problems

the ILP solver is unable to close the gap in hours of computation on instances of a few thousands variables and constraints. More specifically, the average gap for the CLP is always around 10%, whereas the situation is even gloomier for the SCP, as the ILP in some cases is unable to find a feasible solution.

Given this experience, we developed a metaheuristic approach based on the VNS framework. We will discuss neighborhoods of different typologies and exploration strategies (exchange mechanisms), which are very promising to solve multimode problems. Computational results show that it is possible to achieve in a matter of minutes a 5% gap with respect to the known bound provided by the solver (which is unlikely to be tight).

References

[1] C. Artigues, S. Demassey and E. Néron (eds.) Resource-Constrained Project Scheduling: Models, Algorithms, Extensions and Applications. Wiley (2008)

[2] F. Colombo, R. Cordone and G. Lulli (2015). A Variable Neighborhood Search algorithm for the Multimode Set Covering Problem, J. of Global Optimization 65 (3) pp. 461–480.

[3] F. Colombo, R. Cordone and G. Lulli (2016). The Multimode Covering Location Problem, Computers & Operations Research 67 pp. 25–33.

[4] A. Srinivasan (2006) An Extension of the Lovász Local Lemma, and its Applications to Integer Programming. *SIAM Journal on Computing*, 36 (3), pp. 609–634.

[5] R.V. Vohra, N. Hall (1993) A probabilistic analysis of the maximal covering location problem *Discrete Applied Mathematics*, 43, pp. 175–183.

Heuristics for static cloudlet location

Alberto Ceselli, Marco Premoli[1]

Department of Computer Science, Università Degli Studi di Milano, Crema, Italy

Stefano Secci[2]

UPMC University Paris 06, UMR 7606, LIP6, F-75005, Paris, France

Abstract

Major interest is currently given to the integration of clusters of virtualization servers, also referred to as 'cloudlets', into the access network to allow higher performance and reliability in the access to mobile edge computing services. We tackle the facility location problem arising in the planning of these networks. Due to the complexity of the network topology, and the number of operational constraints, methods from the literature are hard to adapt. While in [1] we discussed the application issues, considering a real test case, in this paper we focus on the algorithmic ones, providing matheuristics solution algorithms for the static case, and an experimental insight on their computational behavior.

Keywords: telecommunications, facility location, matheuristics

Model. Let B be a set of access point (AP) locations. Let I, J and K be a set of sites where aggregation, core nodes and cloudlet facilities can be installed, resp.. Our static cloudlet location problem asks to design a two-level AP-aggregation-core network, to locate cloudlets on it, and to assign APs to cloudlets, minimizing installation costs, respecting cloudlet capacities and service level agreements on maximum delay and available bandwidth on paths between APs and cloudlets. We assume a superposition of stars topology: any AP is connected to a single aggregation node, and each aggregation node to a single core node, while a full mesh is built among cores. For each AP $s \in B$, let δ_s^u be the number of users connecting to s and δ_s^b their overall

[1] Email: {alberto.ceselli, marco.premoli}@unimi.it

[2] Email: stefano.secci@upmc.fr

bandwidth consumption. Let l_i, m_j, c_k be the fixed cost for activating an aggregation node in $i \in I$, a core node in $j \in J$ and a cloudlet facility in $k \in K$, resp.. Let C denote the number of users that each cloudlet can serve. Let $d_{i,j}$ and $u_{i,j}$ be the length and bandwidth capacity of each link $(i,j) \in E = (B \times I) \cup (I \times J) \cup (J \times J)$. We assume low latency to be enforced by imposing both a maximum sum of links' length ($\bar{D}$) and number of hops ($\bar{H}$) in a path from AP to its cloudlet, and a maximum distance ($\bar{d}$) between connected nodes. We define as S^{sk} the set of paths from APs to cloudlets such that $\sum_{(i,j)\in p} d_{(i,j)} \leq \bar{D}$, $|p| \leq \bar{H}$ and $d_{(i,j)} \leq \bar{d}$ for all $(i,j) \in p\}$, with $|p|$ denoting the number of links forming path p. We introduce three sets of variables. The first corresponds to binary location variables: x_i, y_j and z_k take value 1 if sites $i \in I$, $j \in J$ and $k \in K$, resp., are selected to host facilities. The second corresponds to binary routing variables: $r_p^{s,k}$ take value 1 if users in AP $s \in B$ are served by a cloudlet in $k \in K$, and the corresponding traffic is routed along path $p \in \bar{S}^{sk}$. The third corresponds to network topology binary variables: $t_{s,i}$, $w_{i,j}$ and $o_{m,n}$ take value 1 if a link is established between an AP s and an aggregation node i, an aggregation node i and a core node j, two core nodes m and n, resp.. Moreover, let $U \in [0,1]$, represent the maximum allowed link utilization ratio. We formulate our problem as follows.

$$\min \sum_{i\in I} l_i x_i + \sum_{j\in J} m_j y_j + \sum_{k\in K} c_k z_k \tag{1}$$

$$\text{s.t.} \sum_{p\in S^{sk}|i\in p} r_p^{s,k} \leq x_i \ , \ \forall s \in B, \forall k \in K, \forall i \in I \tag{2}$$

$$\sum_{p\in S^{sk}|j\in p} r_p^{s,k} \leq y_j \ , \ \forall s \in B, \forall k \in K, \forall j \in J \tag{3}$$

$$\sum_{p\in S^{sk}} r_p^{s,k} \leq z_k \ , \ \forall s \in B, \forall k \in K \tag{4}$$

$$\sum_{k\in K} \sum_{p\in S^{s,k}} r_p^{s,k} = 1 \ , \ \forall s \in B \tag{5}$$

$$\sum_{s\in B} \sum_{p\in S^{s,k}} \delta_s^u r_p^{s,k} \leq C z_k \ , \ \forall k \in K \tag{6}$$

$$\sum_{s\in B} \sum_{k\in K} \sum_{\substack{p\in S^{s,k} \\ |(i,j)\in p}} \delta_s^b r_p^{s,k} \leq u_{(i,j)} U (w_{i,j} + o_{i,j} + t_{i,j}) \ , \ \forall (i,j) \in E \tag{7}$$

We minimize installation costs (1); (2)-(4) impose that no path can be selected unless devices are installed on its sites; (5)-(7) ensure that each AP is assigned to a cloudlet; (6) impose that active cloudlets serve at most C users; (7) are link utilization constraints. Moreover a set of topology constraints need to be imposed, that are omitted here for the sake of brevity.

Algorithms. We devised matheuristics that consist of five phases: (i) clustering of the APs in B, aggregating their demands in *centers* (ii) dynamic generation of the center-cloudlet path variables $r_p^{s,k}$ (iii) retrieval of a feasible solution with a hierarchical rounding and pricing process (iv) refinement of the solution with local branching (v) restart.

During phase (i) we create $|B|/\alpha$ clusters of APs by selecting *centers*. To ensure feasibility we enforce that no AP is placed in a cluster if its distance from the center is greater than $\bar{d}$, and the distance between two centers is computed as the maximum distance between one center and each of the APs of the other cluster. To initialize the clustering we use a simplified model in which cloudlets, aggregation and core nodes coincide. Therefore, a routing path is always a direct link, the resolution process needs to find only cloudlet locations, and only capacity constraints need to be enforced.

During phase (ii), as the cardinality of feasible paths sets S^{sk} grows combinatorially, we perform column generation on the set of variables $r_p^{s,k}$. The pricing problem is a resource constrained shortest path problem on an acyclic network, that we solve in pseudo polynomial time by dynamic programming.

At the end of the column generation process (phase iii) we start rounding by selecting the location variable with highest fractional value, fix it to one, and propagate that fixing. If the solution is still fractional, we resume column generation to restore optimality, and we repeat the rounding and propagation process. If infeasibility is detected we backtrack, fixing the last rounding variable to zero, and column generation is resumed. If infeasibility is obtained also in this way, we stop in a FAIL status. Whenever a feasible integer solution is achieved, instead, we stop in a SUCCESS status. Instead of choosing an arbitrary location variable for rounding, we consider in sequence variables z_k, y_j, x_i and $r_p^{s,k}$. Variables related to the topology are never rounded explicitly: in case of SUCCESS, a small MILP problem remains to fix them. In case of FAIL, instead, the solution produced in phase (i) is considered. That is, in any case a feasible solution $\hat{S}$ is obtained after phase (iii), unless the instance itself is infeasible.

During phase (iv) we try to improve the feasible integer solution $\hat{S}$ with an ILP-based very large scale neighborhood search strategy, exploring a *κ-OPT neighborhood*: we consider the restricted model produced by the last column generation round, and we include the following local-branching con-

Table 1
Results on 100 nodes instances

		$\alpha = 2$			$\alpha = 3$			$\alpha 2 \to 3$	
	z^{init}	t	z^*	Δ z	t	z^*	Δ z	Δ z	Δ t
μ	12.65	9055.70	11.37	10.01%	320.50	11.85	6.21%	-4.43%	96.35%
σ	0.74	1884.33	0.70	5.26	75.80	0.64	4.33	5.41	0.95
max	13.32	11984.00	12.92	18.84%	489.00	13.12	10.73%	-11.10%	97.27%

straint: $\sum_{k \in K | \bar{z}_k = 1} (1 - z_k) + \sum_{k \in K | \bar{z}_k = 0} z_k \leq \lceil \kappa \cdot \sum_{k \in K} \bar{z}_k \rceil$ where parameters $\bar{z}_k$ represent the values of the variables z_k in $\hat{S}$, and parameter κ represents the fraction of z_k variables whose values are allowed to flip with respect to the current solution. We solve this restricted model with a general purpose ILP Solver, setting a limit τ on the execution time. As a restart strategy (phase v) we update the clustering and iterate steps (ii)–(iv). The information given by the fractional solution found at the end of phase (ii) is used to perform such an update: when a center i is fully associated to a cloudlet k through a single path, the two clusters represented by i and k are joined and a new representative is found by aggregating them; otherwise if center i is fractionally associated through multiple paths to different cloudlets, then the corresponding cluster is split, trying to improve a suitable *connectivity measure* that we devised, and whose formal definition is omitted for brevity. A fixed number of restarts are performed, and the best solution found is retained.

Computational results. We implemented our algorithms in C++, using CPLEX 12.6 to solve both LP and MILP problems. Our tests ran on an Intel Core 2 Duo 3 GHz workstation with 2 GB of RAM. Parameters are set as in [1]. We considered a dataset adapted from capacitated p-median instances from the literature. Table 1 reports an overview of results on instances with 100 nodes. We first report the value of the solution found by the initial clustering heuristics (z^{init}). Then we indicate average computing time, value of the solutions and gap with respect to the initial solutions, comparing two settings: clustering with $\alpha = 2$ (second block) and $\alpha = 3$ (third block). In the last block we summarize the effect of moving from $\alpha = 2$ to $\alpha = 3$. Our matheuristics lead to an average improvement of the initial solution of $\sim 10\%$ and $\sim 6\%$, resp.. We also note that using fewer clusters leads to major savings in CPU time ($\sim 96\%$) with a mild quality worsening ($\sim 4\%$).

References

[1] A. Ceselli, M. Premoli, S. Secci, "Cloudlet Network Design Optimization", in *Proc. of IFIP Networking 2015*, 20-25 May, 2015, Toulouse, France.

A preliminary analysis of the Distance Based Critical Node Problem

Roberto Aringhieri, Andrea Grosso, Pierre Hosteins[1]

Dipartimento di Informatica
Università degli Studi di Torino
Turin, Italy

Rosario Scatamacchia[2]

Dipartimento di Automatica e Informatica
Politecnico di Torino
Turin, Italy

Abstract

We discuss how to develop efficient heuristics for the distance based critical node problem, that is the problem of deleting a subset of k nodes from a graph G in such a way that the distance between each pair of nodes is as large as possible.

Keywords: Critical Node Problem, Graph Fragmentation, Shortest Paths.

1 Introduction

The Critical Node Problem (CNP) has been defined as a type of Interdiction Network Problem which aims at maximally fragmenting an undirected and

[1] Email: roberto.aringhieri@unito.it, grosso@di.unito.it, hosteins@di.unito.it

[2] Email: rosario.scatamacchia@polito.it

unweighted graph $G = (V, E)$ by deleting a subset of its nodes $S \subset V$ ($|S| = k$) according to a specific connectivity measure. This particular problem has raised a certain interest in the recent literature due its potential applicability to a vast number of real situations (see, e.g., [4]). Currently, the state of the art algorithms for solving the CNP are those presented in [1,2,3].

In the classic CNP, the connectivity is related to a pair-wise connectivity concept, that is either a path exists between a pair of nodes, or it does not. In [8], the authors introduces a more refined connectivity concept based on the shortest distance between each pair of nodes: the more distant the nodes, the lower their connectivity value. Therefore, the DB-CNP consists in minimizing the following objective function:

$$F(S) = \sum_{i,j \in V \setminus S \,:\, i \neq j} \frac{1}{d_{spt}(i,j)} \tag{1}$$

where d_{spt} is the value of the shortest path between the node i and the node j belonging to the weighted graph G.

Constructive and Local Search based heuristics usually build an incumbent solution step-by-step, that is, for instance, adding or deleting elements, or swapping a pair of elements respectively belonging and not belonging to a starting solution. As for the classic CNP, the development of efficient heuristic algorithms for the DB-CNP suffers from the non trivial evaluation of the incumbent new solution since we need to update the shortest path between each pair of nodes. In this paper we discuss how to develop efficient heuristics for the DB-CNP.

2 Shortest paths re-computation

The operations traditionally used to obtain an incumbent solution of the CNP consist in adding a node to S (i.e., deleting it from the graph), removing a node from S or swapping a node from S with a node from $V \setminus S$. As moving nodes from or to S can affect the length of shortest paths (SP), we are required to recompute all the SP values in the graph, which is known to have a computational cost of $\mathcal{O}(|V||E| + |V|^2 \log |V|)$ [6]. As such a complexity is usually prohibitive when thousands of incumbent solutions should be evaluated, we need to implement more efficient evaluations of the SP modifications.

It has been noted in computational works regarding all-SP re-computation that usually, if a very small number of edges' weights are modified, the time necessary to recompute only the shortest paths that are affected is actually much less than the theoretical worst case complexity. Since for the CNP we only modify the edges belonging to the backward and the forward start of one node at a time, such empirical results are encouraging for implementing efficient heuristics.

Moreover, some particular cases of interest to us can be demonstrated to require a lower worst case complexity than the general all-SP re-computation. For example, reintroducing a node $u \in S$ inside the graph amounts to consider that each SP can now go through u if it is profitable enough. Using the SP properties, we can show that computing the SP starting and ending at u can be done in $\mathcal{O}(D(G)(|V| - |S|))$ where $D(G)$ is the largest number of edges incident on any node in G. Then using those new SP lengths we can update all shortest paths in a maximum number of operations equal to $\mathcal{O}(|V|^2)$, which is inferior to the general case of edge weights modification [6].

Some dominance rules should also be devised, the simplest example being a node $v \in V \setminus S$ which does not belong to any shortest path in graph $G[V \setminus S]$: evidently such a node can never be an appropriate candidate for deletion since such a move would not lower the objective function. Similarly, the impact of removing a node belonging to a certain connected component would not change if our moves in the solution space only modifies other connected components.

3 Extension to directed graphs and weighted pair-wise connectivity

Since the SP definition is not limited to undirected graphs, the DB-CNP can be also applied to directed graphs, which opens the perspective of applying the critical node analysis to such situations that can be modelled by directed graphs only, contrary to the versions of the CNP previously considered in the literature [4].

We also note that the CNP based on weighted pair-wise connectivity is much more difficult to tackle with the existing heuristic algorithms since they tend to rely on the fact that not weighted pair-wise connectivity can be computed solely using the connected components' cardinality, a fact which is no longer true when weights are introduced between pairs of nodes. However, the heuristic framework developed for a DB-CNP, which tracks the SP values, allows us to evaluate solution moves for weighted pair-wise connectivity as a non infinite length means that the nodes are connected. Thus we see that algorithmic efforts in order to solve the DB-CNP can be beneficial for other formulations of the CNP as well.

4 Betweenness centrality

Betweenness centrality [7] can play also a fundamental role to devise efficient heuristics. Centrality would evaluate how important is a node for the connection of every pair of nodes. Betweenness centrality of the node j is the number of shortest paths from all vertices to all others that pass through the node j, and it can be computed using the Brandes' algorithm [5]. The basic idea is therefore to rank the nodes with respect to their betweenness value, and to consider first those having highest value in our heuristics. Note that heuristics for the classic CNP benefit of using such a rank as reported in [1,3].

References

[1] Addis, B., R. Aringhieri, A. Grosso and P. Hosteins, *Hybrid Constructive Heuristics for the Critical Node Problem*, Annals of Operations Research **238** (2016), pp. 637–649.

[2] Aringhieri, R., A. Grosso and P. Hosteins, *A genetic algorithm for a class of Critical Node Problems*, Electronic Notes in Discrete Mathematics (2015), proceedings of the INOC 2015 conference. To appear.

[3] Aringhieri, R., A. Grosso, P. Hosteins and R. Scatamacchia, *Local Search Metaheuristics for the Critical Node Problem*, Networks (2016), DOI:10.1002/net.21671.

[4] Arulselvan, A., C. W. Commander, L. Elefteriadou and P. M. Pardalos, *Detecting critical nodes in sparse graphs*, Computers & Operations Research **36** (2009), pp. 2193–2200.

[5] Brandes, U., *A faster algorithm for betweenness centrality*, Journal of Mathematical Sociology **25** (2001), pp. 163–177.

[6] Demetrescu, C. and G. F. Italiano, *Experimental analysis of dynamic all pairs shortest path algorithms*, ACM Trans. Algorithms **2** (2006), pp. 578–601.

[7] Freeman, L. C., *A Set of Measures of Centrality Based on Betweenness*, Sociometry **40** (1977), pp. 35–41.

[8] Veremyev, A., O. A. Prokopyev and E. L. Pasiliao, *Critical nodes for distance-based connectivity and related problems in graphs*, Networks **66** (2015), pp. 170–195.

Timetable Optimization for High-Speed Trains at Chinese Railways

Valentina Cacchiani [1]

DEI
University of Bologna
Bologna, Italy

Feng Jiang [2]

School of Transportation and Logistics
Southwest Jiaotong University
Chengdu, China

Paolo Toth [3]

DEI
University of Bologna
Bologna, Italy

Abstract

We study the Train Timetabling Problem (TTP) of the high-speed trains at the Chinese railways. TTP calls for determining, in the planning phase, an optimal schedule for a given set of trains, while satisfying track capacity occupation constraints. In this work, we are given on input a set of feasible timetables for the trains already planned along a double-track high-speed line, and the main goal consists of scheduling as many additional trains as possible. Beside the main goal, a second objective is to obtain a *regular schedule*, i.e. a schedule showing regularity in the train frequency. We model TTP on a time-space graph and propose a heuristic algorithm for it. Preliminary computational results on real-world instances of the high-speed line from Beijing to Shanghai in China are reported.

Keywords: Train Timetabling, Heuristic Algorithm, Timetable Regularity

1 Introduction

Railway networks are more and more utilized. The high-speed double-line from Beijing to Shanghai in China has shown an average increase of passenger volume of about 30% every year since it was put into operation in 2011. In 2013, 8.4 million of passengers travelled along the corridor and this number is expected to keep growing, mainly because new high-speed lines connected with the Beijing-Shanghai line will be put into operation.

The Train Timetabling Problem (TTP) is a crucial step to improve the capacity utilization of the railway network while guaranteeing a high quality of service. The TTP at the planning level consists of determining efficient train schedules in order to have a service with high frequency and short waiting and travel times. We focus on the non-periodic TTP, i.e. we do not require that trains are operated with the same schedule every given time period (e.g. every hour), since we consider a highly congested corridor (double-track railway line).

Given a corridor, described as a sequence of stations and a set of tracks connecting them, the TTP calls for scheduling arrivals and departures of trains at/from the stations, while respecting minimum travel, stopping and headway times, and track capacity constraints (related to overtaking and crossing of trains that must be avoided according to the physical structure of the corridor). The non-periodic TTP has been deeply investigated in the literature ([4], [8]), not only for deriving timetables from scratch ([1], [5]), but also for improving existing timetables by inserting additional trains ([2]), real-time rescheduling ([6]) and increasing the capacity utilization of the railway nodes ([3]).

In this work, we start from a given initial feasible schedule (described as departure and arrival train times at each of the visited stations) for the trains already planned along the double-track high-speed line. Given a set of new trains to be scheduled along the line, the main goal consists of scheduling as many additional trains as possible. As the number of passengers is expected to

[1] Email:valentina.cacchiani@unibo.it

[2] Email:feng.jiang@studio.unibo.it

[3] Email:paolo.toth@unibo.it

increase, inserting new passenger trains is important to guarantee comfortable travels. For each additional train, we are given its origin station and departure time, its destination station, the set of intermediate stations to be visited, the minimum stopping time at each of them and the travel time between each pair of consecutive stations. To determine an overall feasible train schedule, we are allowed to change the departure times and to increase the stopping times of the additional trains. With respect to the existing literature, we also investigate the possibility of modifying the timetables of the already planned trains and even of changing the stopping patterns (i.e. of canceling train stops at a given penalty) of both the planned and the additional trains. Beside the main goal, a second objective is to obtain a *regular schedule*, i.e. a schedule showing regularity in the train frequency at the main stations. A regular frequency allows passengers to easily remember the timetable and has uniform waiting times for the following train at different times of the day. This follows the aim of the periodic TTP ([7]), while keeping the flexibility of the non-periodic TTP.

2 Heuristic Algorithm

We model TTP on a time-space graph ([2], [3], [5]). Let T be the set of trains to be scheduled and $G = (V, A)$ the time-space graph: nodes in V represent time instants at which some train can arrive at and depart from a station. In addition, V includes an artificial source node σ and an artificial terminal node τ. The arc set A is partitioned into sets $A^1, \ldots, A^{|T|}$, one for each train $t \in T$. These arcs represent either the travel or the stop of a train, or correspond to artificial arcs connected to σ and τ. Given this graph representation, a path from σ to τ using arcs in A^t corresponds to a feasible timetable for train $t \in T$.

We propose an iterative heuristic algorithm based on a two-phase approach. In the first phase, we consider the main goal and insert additional trains: this is achieved by using a dynamic programming procedure that computes, for each additional train, its best path in the time-space graph, while respecting all the operational constraints induced by the previously scheduled trains. In particular, every change to the departure or stopping times and to the stopping patterns of the additional trains is penalized, and the already planned trains are initially kept as fixed. Successively, we allow to change the timetables and to cancel existing stops of the planned trains, so as to possibly schedule additional trains.

In the second phase, we focus on the regularity of the timetable, while keeping the set of scheduled trains as fixed. To improve the timetable regularity we assign appropriate penalties to the train graph nodes so as to favor regular time intervals between consecutive trains at the main stations.

3 Preliminary Computational Experiments

We consider the 2015 timetable of the Beijing-Shanghai high-speed corridor, that is a double-track line with 29 stations along which 304 trains run every day between 06:00 and midnight in both directions. We are given 42 additional trains to be scheduled. After the first phase, 20 additional trains are scheduled. In the second phase, the regularity of the scheduled trains from the main departure stations Beijing and Shanghai is improved. Current research is dedicated to further improve the line capacity utilization and the timetable regularity. In addition, we are studying different measures of regularity, and focusing on dealing with additional real-world constraints.

References

[1] Brännlund, U., Lindberg, P.O., Nöu, A. and J.E. Nilsson, *Allocation of scarce track capacity using Lagrangian relaxation*, Transportation Science. **32** (1998) 358–369.

[2] Cacchiani, V., Caprara, A., and P. Toth, *Scheduling Extra Freight Trains on Railway Networks*, Transportation Research Part B. **44(2)** (2009), 215–231.

[3] Cacchiani, V., Furini, F., and M.P. Kidd, *Approaches to a real-world train timetabling problem in a railway node*, Omega. **58** (2016), 97–110.

[4] Cacchiani, V., and P. Toth, *Nominal and Robust Train Timetabling Problems*, European Journal of Operational Research. **219(3)** (2012), 727–737.

[5] Caprara, A., Fischetti, M., and P. Toth, *Modeling and solving the train timetabling problem*, Operations Research. **50** (2002) 851–861.

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

[7] Kroon, L. G., Peeters, L. W., Wagenaar, J. C., and R. A. Zuidwijk, *Flexible connections in pesp models for cyclic passenger railway timetabling*, Transportation Science. **48(1)** (2013) 136–154.

[8] Lusby, R. M., Larsen, J., Ehrgott, M., and D. Ryan, *Railway track allocation: models and methods*, OR spectrum. **33(4)** (2011), 843–883.

Partitioning a graph into minimum gap components

Maurizio Bruglieri [1]

Dipartimento di Design, Politecnico di Milano, Milano, Italy

Roberto Cordone [2]

Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

Abstract

We study the computational complexity and approximability for the problem of partitioning a vertex-weighted undirected graph into p connected subgraphs with minimum gap between the largest and the smallest vertex weights.

Keywords: Graph partitioning, computational complexity, approximability

1 Introduction

Let $G = (V, E)$ be an undirected connected graph, w_v an integer *weight* coefficient defined on each vertex $v \in V$, and $p \leq |V|$ a positive integer number. Given a vertex subset $U \subseteq V$, we denote by $m_U = \min_{u \in U} w_u$ and $M_U = \max_{u \in U} w_u$ the minimum and maximum weight in U, respectively, and by *gap* their difference $\gamma_U = M_U - m_U$. The *Minimum Gap Graph Partitioning*

[1] Email: `maurizio.bruglieri@polimi.it`

[2] Email: `roberto.cordone@unimi.it`

Problem (*MGGPP*) requires to partition G into p vertex-disjoint connected subgraphs $G_r = (V_r, E_r)$, $(r = 1, \ldots, p)$ with at least two vertices each. Its *min-max* and *min-sum* versions minimize, respectively, the maximum gap f^{MM} and the sum of the gaps f^{MS} over all subgraphs:

$$f^{MM} = \max_{r=1,\ldots,p} \gamma_{V_r} \qquad f^{MS} = \sum_{r=1}^{p} \gamma_{V_r}$$

The *MGGPP* can find applications in agriculture (divide a land into parcels with limited difference in height [3]), in the location of gate houses along rivers, and in social network analysis (identify connected clusters of members with homogeneous features). It falls in the large field of graph partitioning problems [1,2], but, as far as we know, objective functions related to the differences between vertex weights in each subgraph have never been considered before.

2 Complexity

Theorem 2.1 *The* MGGPP *admits feasible solutions if and only if graph G contains a matching of cardinality at least p.*

Proof. Any maximum cardinality matching M induces on graph G a spanning forest of $|M|$ nondegenerate trees and $|V| - 2|M|$ isolated vertices. Each isolated vertex v has an incident edge e_v which is adjacent to an edge in M. Adding e_v to M for each isolated vertex v, we obtain a spanning forest of exactly $|M|$ trees. If $|M| > p$, we consider the edges connecting different trees, and we add them to M, stopping as soon as we obtain exactly p trees. This provides a feasible solution of the *MGGPP*. Vice versa, given a feasible solution, we can choose an edge from each subgraph (they all contain at least two vertices): these edges are nonadjacent, and yield a p-cardinality matching. □

Let $W_U = \{z \in \mathbb{Z} : \exists v \in U \text{ with } w_v = z\}$ be the set of values assumed by w on a subset of vertices $U \subseteq V$, and $\eta_U = |W_U|$ the number of such values.

Theorem 2.2 *The* MGGPP *with the* min-max *objective function is strongly $\mathcal{NP}$-hard even if $p = 2$ and $\eta_V = 3$.*

Proof. The decision version of the problem, obviously in $\mathcal{NP}$, amounts to verifying the existence of a solution such that the gap of all subgraphs is not larger than a given threshold. Given a generic instance of *SAT*, we build the following auxiliary graph. We introduce for each literal (x_i or $\bar{x}_i$) a vertex (v_i or $\bar{v}_i$) with $w_{v_i} = w_{\bar{v}_i} = 2$, and for each clause C_j a vertex c_j with weight $w_{c_j} = 1$;

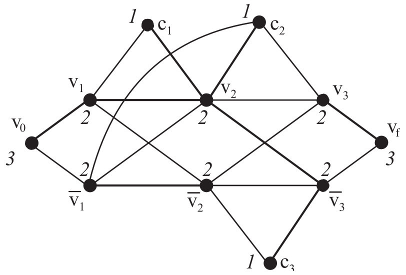


Fig. 1. Graph construction for the $\mathcal{NP}$-hardness proof of the min-max *MGGPP*

finally, we introduce two dummy vertices v_0 and v_f with weight $w_0 = w_f = 3$. Vertex v_0 is connected to v_1 and $\bar{v}_1$; vertex v_f is connected to v_n and $\bar{v}_n$; each vertex v_i (resp. $\bar{v}_i$) is connected to v_{i+1} and $\bar{v}_{i+1}$ $(i = 1, \ldots, n-1)$ and to all the clause vertices c_j such that literal x_i (resp. $\bar{x}_i$) occurs in clause C_j. We are looking for $p = 2$ connected subgraphs with gaps not larger than 1. Figure 1 shows the graph corresponding to $(x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_2 \vee x_3) \wedge (\bar{x}_2 \vee \bar{x}_3)$. If both subgraphs have gap ≤ 1, v_0 and v_f belong to the same subgraph, and this connects them through a path entirely made of vertices v_i or $\bar{v}_i$. By construction, this path contains at least one of v_i or $\bar{v}_i$ for each variable x_i. The other subgraph contains all the clause vertices c_j and connects them through adjacent vertices v_i or $\bar{v}_i$ which identify literals satisfying all clauses. Such a truth assignment is consistent because the subgraph includes at most one vertex for each variable x_i. Vice versa, any satisfying truth assignment identifies a partition of the graph into two subgraphs with gap ≤ 1. □

Theorem 2.3 *The* MGGPP *with the* min-sum *objective function is strongly $\mathcal{NP}$-hard even if $\eta_V = 2$.*

Proof (Sketch). The proof is by reduction from 3-*SAT*. □

3 Approximability

Theorem 3.1 *The* min-max MGGPP *cannot be approximated for any constant $\alpha < 2$ unless $\mathcal{P} = \mathcal{NP}$.*

Proof. Following Theorem 2.2, we can build an instance with optimum equal to 1 for any YES-instance of *SAT* and one with optimum equal to 2 for any NO-instance. By contradiction, a hypothetical α-approximated polynomial algorithm with $\alpha < 2$, would find on the former instances solutions with a value < 2 (by integrality, 1), and therefore solve *SAT* in polynomial time. □

Theorem 3.2 *The* MGGPP *is 2-approximable for $p = 2$.*

Proof. Let V_1^* and V_2^* be the unknown subsets of vertices of the optimal solution. The ranges of the weights in the two subgraphs, $\left[m_{V_1^*}; M_{V_1^*}\right]$ and $\left[m_{V_2^*}; M_{V_2^*}\right]$, are either separate or overlapping. In the former case, all the vertices in a subgraph have weights strictly smaller than those in the other. Then, the optimal solution can be found by exhaustively considering all pairs of intervals $[w_{\pi_1}, w_{\pi_k}]$ and $\left[w_{\pi_{k+1}}, w_{\pi_\eta}\right]$ $(k = 1, \dots, \eta_V - 1)$, and building the subgraphs induced on G by the vertices whose weights fall in the two intervals. In the latter case, the two ranges overlap, and $f^{*MS} = \gamma_{V_1^*} + \gamma_{V_2^*} \geq \gamma_V$, which implies $f^{*MM} = \max\left(\gamma_1^*, \gamma_2^*\right) \geq \gamma_V/2$. Generating any feasible solution with Theorem 2.1, we obtain $f^{MS} \leq 2\gamma_V \leq 2f^{*MS}$ and $f^{MM} \leq \gamma_V \leq 2f^{*MM}$. □

4 Some special cases

The *MGGPP* admits some polynomially solvable special cases.

Proposition 4.1 *The* min-max MGGPP *is polynomially solvable if* $\eta_V = 2$*.*

Proof (Sketch). If there is a vertex whose weight is different from that of the adjacent vertices, the optimal solution is γ_V. Otherwise, we merge all the adjacent vertices of equal weight and consider the resulting vertex set V'. If $|V'| > p$, the optimum is γ_V; otherwise, a procedure similar to that of Theorem 2.1 provides an optimal solution with p subgraphs of zero gap. □

Proposition 4.2 *The* min-sum *and* min-max MGGPP *are polynomially solvable on line graphs.*

Proof (Sketch). The proof is based on the computation by dynamic programming of the minimum bottleneck path on a suitable graph. □

We are currently investigating the complexity of other special cases and working on the design of exact and heuristic algorithms.

References

[1] D. A. Bader, H. Meyerhenke, P. Sanders, and D. Wagner, eds. *Graph Partitioning and Graph Clustering*, v. 588 of *Contemporary Mathematics.* AMS, 2013.

[2] C.-E. Bichot and P. Siarry, editors. *Graph Partitioning.* Wiley-ISTE, 2013.

[3] Li Xiao, Li Hongpeng, Niu Dongling, Wang Yan, and Liu Gang. Optimization of GNSS-controlled land leveling system and related experiments. *Transactions of the Chinese Society of Agricultural Engineering*, 31(3):48–55, 2015.

Algorithms for uniform centered partitions of trees

Isabella Lari[a], Justo Puerto[b], Federica Ricca[a], Andrea Scozzari[c]

[a] *Sapienza University of Rome, Italy*

[b] *IMUS, Instituto de Matemáticas de la Universidad de Sevilla, Spain*

[c] *University Niccolò Cusano, Rome, Italy*

Abstract

In this paper we provide polynomial time algorithms for the problem of finding uniform centered partitions of a tree, that is, partitions that are as balanced as possible either w.r.t. the costs or to the weights of their components.

Keywords: Tree partitioning, centered partitions, flat costs, min-max criteria, uniform partitions.

1 Introduction, notation and definitions

Let $T = (V, E)$ be a tree with $|V| = n$. Assume that V is partitioned into two subsets S and U such that $S \subset V$ with $|S| = p$. S is the set of *centers* (*facilities*) and $U = V \backslash S$ is the set of *units* (*clients*). We consider a cost function $c : U \times S \to \mathbb{R}^+ \cup \{0\}$ which associates a cost c_{is} to each pair (i, s), $i \in U$, $s \in S$. We assume that these costs are *flat*, i.e., they are independent of the topology of T. We also consider a nonnegative weight w_i associated to each i in U. A *centered partition* of T is a partition of the set V into p non empty subsets, $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$, such that each subset induces a subtree of T and contains exactly one center. The *cost of the component* $\mathcal{C}_s$ centered in s is defined as the sum of the costs c_{is} of the units $i \in \mathcal{C}_s$. The *weight of the component* $\mathcal{C}_s$ is given by the sum of the weights of the units $i \in \mathcal{C}_s$. We consider the flat costs and study the following two problems: i) *max-min cost* centered partition problem, that is, find a centered partition of T that maximizes the minimum cost of a component; ii) *min-max cost* centered partition problem, that is, find a centered partition of T that minimizes the maximum cost of a component.

Replacing the minimum and maximum cost by the minimum and maximum weight of a component we obtain the following variants of the above problems: iii) *max-min weight* centered partition problem; iv) *min-max weight* centered partition problem. For problems i)-iv) we provide polynomial time algorithms: for i), iii) and iv) we adapt already existing approaches, while for ii) we suggest a new procedure. This kind of problems are known as *uniform* partition problems, and they have been widely studied in the literature on trees [2,5]. In this paper we focus on the particular case of finding uniform centered partitions. In previous papers we already studied problems of this class on general graphs, providing several NP-completeness results for other types of graphs [1,3]. In particular, we proved that all the above problems are NP-complete even on planar bipartite graphs with vertex degree at most 3 and $p = 2$, and this motivates our interest for studying them now on trees.

2 Max-min centered partition of trees

In this section we study the max-min (cost/weight) centered partition problem of a tree T and we show how this problem can be solved in polynomial time by using results from [2], where Becker and Perl provide a general technique for partitioning trees with different objectives that is based on *shifting* operations and greedy decisions. Given the family $\mathcal{F}$ of all the possible subsets of V, they define a weighting function $H : \mathcal{F} \to \mathbb{R}^+ \cup \{0\}$ that assigns a weight $H(\mathcal{Z})$ to each subset $\mathcal{Z}$ in $\mathcal{F}$. Among the others, they solve the problem of finding a partition of T into p connected components, $\{\mathcal{Z}_1, \ldots, \mathcal{Z}_p\}$, that maximizes the minimum of the $H(\mathcal{Z}_j)$, $j = 1, \ldots, p$, by applying a shifting algorithm originally proposed in [5]. We refer to this problem as *BP-max-min problem* and observe that the only difference with our problem is that [2] does not consider centered partitions. For the BP-max-min problem the shifting algorithm applies when $H(\cdot)$ is a *basic* weighting function, i.e., a function satisfying the following property: if $\mathcal{Z}_1, \mathcal{Z}_2 \in \mathcal{F}$ are such that $\mathcal{Z}_1 \subseteq \mathcal{Z}_2$ then $H(\mathcal{Z}_1) \leq H(\mathcal{Z}_2)$.

Consider our problem of finding a max-min cost centered partition $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$ of T. Let $M = \sum_{i \in U} \max_{s \in S} c_{is}$. For a generic subset $\mathcal{C}$ of V we introduce the following weighting function:

$$H(\mathcal{C}) = \begin{cases} M|\mathcal{C} \cap S| + \sum\limits_{i \in \mathcal{C} \cap U} \max\limits_{s \in \mathcal{C} \cap S} c_{is} & \text{if } \mathcal{C} \cap S \neq \emptyset \\ \sum\limits_{i \in \mathcal{C}} \min\limits_{s \in S} c_{is} & \text{if } \mathcal{C} \cap S = \emptyset \end{cases} \tag{1}$$

It is easy to see that the above weighting function is basic. Notice that when $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$ is a centered partition, for a component $\mathcal{C}_s$ centered in s one has:

$$H(\mathcal{C}_s) = M + \sum_{i \in \mathcal{C}_s \cap U} c_{is} \tag{2}$$

Theorem 2.1 *A partition $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$ is an optimal solution of the max-min cost centered partition problem on a tree T if and only if it is an optimal solution of the BP-max-min problem with weighting function $H(\cdot)$.*

From Theorem 2.1 it follows that the max-min cost centered partition problem can be solved by the shifting algorithm for the BP-max-min problem in $O(p^2 r + pn)$ time, where r is the radius of T. We observe that the same basic weighting function (1) and the same shifting algorithm can be applied also to solve our max-min weight centered partition problem by setting for each $i \in U$: $c_{is} = w_i$, $\forall s \in S$.

3 Min-max centered partition of trees

In [2] Becker and Perl also provide a shifting algorithm for the problem of finding a partition of a tree into p connected components that minimizes the maximum weight of a component (*BP-min-max problem*) that applies when the weighting function $H(\cdot)$ is *invariant* (see [2] for the definition). Our min-max weight centered partition problem on T can be solved in polynomial time by exploiting the shifting algorithm in [2]. Let $W = \sum_{i \in U} w_i$ and assign the following weights:

$$\overline{w}_v = \begin{cases} w_v \text{ if } v \in U \\ W \text{ if } v \in S \end{cases} \tag{3}$$

It can be shown that the resulting weighting function that assigns to a component $\mathcal{C}$ a weight $\overline{H}(\mathcal{C}) = \sum\limits_{v \in \mathcal{C}} \overline{w}_v$ is invariant.

Theorem 3.1 *A partition $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$ is an optimal solution of the min-max weight centered partition problem on a tree T if and only if it is an optimal solution of the BP-min-max problem with weighting function $\overline{H}(\cdot)$.*

The most efficient implementation of the shifting algorithm for the BP-min-max problem was provided by Perl and Vishkin in [6] and requires $O(rp(p + \log d) + n)$ time, where r and d are the radius of T and the maximum degree of a vertex, respectively.

Finally, for the min-max cost centered partition problem we propose a new polynomial time algorithm based on the solution of a sequence of feasibility problems in which, at each iteration, a centered partition with maximum component cost bounded above by a quantity δ (*δ-centered partition*) must be identified. Since T is a tree, a unit i cannot be assigned to a center s such that the unique path from i to s contains another center $s' \neq s$. As a consequence, we can suppose that all leaves of T are centers. For a fixed value δ, if a δ-centered partition of T exists, it can be found by visiting bottom-up T rooted at a leaf r (denoted by T_r). Let T_i be the subtree of T_r rooted at i, S_i the set of its centers, and $p(i)$ the parent of i in T_r, $i \neq r$. The idea of the algorithm is to add as much cost as possible to the components in the bottom part of the tree without exceeding the given limit δ. If a unit i can be assigned to a center in T_i, such center is selected in S_i as the one that minimizes the sum of the assigning costs; if not, i must be assigned to the same center as its parent $p(i)$ in $S \backslash S_i$. In this way, during the algorithm, for the current vertex i and for each center $s \in S_i$ we are able to record the minimum cost of a component containing i and s. A δ-centered partition of T exists if, at the end, all these costs are smaller than or equal to δ. During the algorithm we compute the following quantities:

- $\bar{c}(i, s)$, $i \in V$ and $s \in S$: the sum of the costs c_{hs} of the units h in T_i that must be assigned to the same center as i in any δ-centered partition of T;
- $w^*(i, s)$, $i \in V$, $s \in S_i$: the minimum cost of a component containing i and s in a

centered partition of T_i whose components, but at most the one containing s, have cost at most δ. At the beginning we set $w^*(i,s) = M > \delta$, $i \in V$, $s \in S_i$.

We also introduce the binary indicator $r(p(i),i)$, $i \in U$, which is set to 1 when i must be necessarily assigned to the same center as its parent $p(i)$ in any δ-centered partition of T. For any given δ, after a suitable initialization of the above quantities, the algorithm performs the following visit of T_r:

```
visit $T_r$ bottom-up starting from its leaves
    if the visited vertex is a unit $i$
        for each $j$ such that $p(j) = i$ and $r(i,j) = 0$
            for each $s \in S_j$ such that $w^*(j,s) \leq \delta$ set $w^*(i,s) := w^*(j,s) + \bar{c}(i,s)$
        if $w^*(i,s) > \delta$ for all $s \in S_i$ then
            set $r(p(i),i) := 1$ and $\bar{c}(p(i),s) := \bar{c}(p(i),s) + \bar{c}(i,s)$ for all $s \in S \backslash S_i$
    else if the visited vertex is a center $s \in S$
        if $\bar{c}(s,s) > \delta$ then STOP: the problem is infeasible
return $r(p(i),i)$, $\forall\, i \in U$
```

If a δ-centered partition exists, it can be found by a top-down visit of T_r using $r(p(i),i)$.

Theorem 3.2 *A δ-centered partition of T can be found in $O(np)$ time.*

By a binary search on all the possible values of δ one can find the min-max cost centered partition in $O(np \log \bar{C})$ time, where $\bar{C}$ is an upper bound on the cost of a component (for example $\bar{C} = \sum_{i \in U} \max_{s \in S} c_{is}$). Let $f(\delta)$ be the maximum cost of a component in a δ-centered partition of T. It is easy to see that $f(\delta)$ is an increasing stepwise linear function of δ whose number of steps is bounded above by 2^n. Using the approach in [4] one can search over the different δ values in an overall time complexity of $O(n^2 p)$.

References

[1] N. Apollonio, I. Lari, J. Puerto, F. Ricca, and B. Simeone, Polynomial algorithms for partitioning a tree into single-center subtrees to minimize flat service costs, *Networks* **51** (2008) 78–89.

[2] R.I. Becker and Y. Perl, The shifting algorithm technique for the partitioning of trees, *Disc. Appl. Math.* **62** (1995) 15–34.

[3] I. Lari, J. Puerto, F. Ricca and A. Scozzari, Partitioning a graph into connected components with fixed centers and optimizing cost-based objective functions or equipartition criteria, *Networks* **1** (2016) 69–81.

[4] M. Megiddo and A. Tamir, New results on the complexity of p-center problems, *SIAM J. Comput.* **12** (1983) 751–758.

[5] Y. Perl and S. Schach, Max-min tree partitioning, *J. of the ACM* **28** (1981) 5–15.

[6] Y. Perl and U. Vishkin, Efficient implementation of a shifting algorithm, *Disc. Appl. Math.* **12** (1985) 71–80.

Drezner's Exact Method for the Continuous $p-$Centre Problem Revisited

Becky Callaghan, Saïd Salhi, Gábor Nagy [1]

Centre for Logistics & Heuristic Optimisation (CLHO), Kent Business School,
The University of Kent, Canterbury, Kent, UK, CT2 7NZ

Abstract

Drezner's optimal algorithm for the $p-$centre problem is an elegant but somewhat slow method. We suggest some technical enhancements that significantly improve the method's efficiency.

Keywords: location, $p-$centre problem, Drezner's method

1 Introduction

The $p-$centre problem seeks to minimise the maximum distance or travel time whilst ensuring all the n demand points are covered by at least one of the p chosen facilities. This problem can be categorised as either the vertex $p-$centre problem or the absolute $p-$centre problem. In the former, which is the discrete case, the optimal facilities are part of a set of the potential facility sites which can be either the demand points or other known sites. However, in the latter the facilities can be located anywhere in the plane. In this work, we will explore the absolute $p-$centre problem by revisiting an early optimal algorithm proposed by Drezner (1984) to solve this problem.

[1] Email: `bc349@kent.ac.uk, s.salhi@kent.ac.uk, g.nagy@kent.ac.uk`

The idea of *maximal circles*, which we use here, is based on Drezner's algorithm. A circle is defined as maximal based on a given upper bound, Z. The set of maximal circles based on Z is then identified and their respective centres are then used as a subset for the potential facility locations.

Let us define the following notations.
I: set of demand points indexed by $i = 1 \dots n$;
J: set of all possible circles indexed by $j = 1 \dots m$;
r_j: the radius of circle C_j, $j \in J$;
C_j: circle j defined by its centre (x^c_j, y^c_j) and radius r_j;
$d_{i,j}$: Euclidean distance from demand point i to the centre of circle C_j, $i \in I, j \in J$;
p: number of facilities to locate;
Z: the upper bound at a given iteration;
J_Z: set of maximal circles defined by Z ($J_Z \subset J$);
$R(K)$: radius of the smallest circle encompassing all points in K, where $K \subset I$.

Definition 1.1 The **closure of circle** C_j is $Cl_j = \{i \ \in I| \ d_{i,j} \leq r_j\} \ \forall \ j = 1 \dots m$.

Definition 1.2 A circle C_j with radius r_j is said to be **maximal** with respect to Z if $r_j < Z$ and for every demand point i $\notin Cl_j$, $R(Cl_j \cup \{i\}) \geq Z$.

Definition 1.3 The **minimum covering circle** (MCC) is the smallest circle that encompasses all the demand points in the Euclidean plane.

Drezner proposed two formulations used to solve for the $p-$centre problem using maximal circles. The first uses the set covering problem to find the minimum number of maximal circles needed. The second imposes a new constraint so that the number of covering circles has to be equal to p, thus making it a feasibility problem. Drezner's optimal algorithm is given in Figure 1.

2 Two Possible Enhancements

Both formulations suggested by Drezner were found to take a relatively large amount of computational time and iterations. This observation led us to consider an alternative approach where the basic $p-$centre formulation was used instead. This approach has an advantage over Drezner's original suggestions as the optimal solution value is much tighter, leading to requiring a relatively smaller number of iterations. Drezner's original algorithm was also enhanced further which will now be explained.

```
Step 1. Find all circles made from one, two or three demand points. This creates
        three sets of circles : $C_J^1$, $C_J^2$ and $C_J^3$. Discard any circle in $C_J^3$ whose
        three points create an obtuse or right-angled triangle.
Step 2. Find an initial solution and set the solution value as the initial upper
        bound, $Z$.
Step 3. Eliminate all circles whose radii are $\geq Z$.
Step 4. Find all circles that are maximal according to Definition 1.2.
Step 5. Find a feasible solution using the set of current maximal circles $J_Z$.
        If a feasible solution is found, set the new upper bound $Z$ to the radius
            of the largest maximal circle found in the solution and go to Step 3.
          Else take the upper bound $Z$ as the optimal solution value of the planar
            $p-$centre problem and stop.
```

Fig. 1. Drezner's Original Algorithm (Drezner [1])

(i) **The Elzinga Hearn algorithm-Based Implementation**
The Elzinga-Hearn algorithm (1972), used to find the MCC of a set of demand points, was enhanced in two ways. Firstly, the algorithm was terminated early if the MCC found had a radius larger than Z. Secondly, more informative initial points were selected.

(ii) **Efficient Recording of the Maximal Circles**
If circle $C_j \in J_{Z_t}$, then $C_j \in J_{Z_{t^*}}$ where $t^* > t$ and so there is no need for further calculations in subsequent iterations.

As an example, Table 1 shows the number of maximal circles found at each iteration for the first 5 iterations of the original algorithm for the TSP-Library data set $pr439$ where $p = 100$. It can be shown that approximately 14% of the new maximal circles need to be identified at each iteration only as the other ones have already been found in previous iterations. Therefore, a technique to identify whether a circle is maximal or not in subsequent iterations is worthwhile constructing.

Our algorithm was tested on the TSP-Lib data set where $N = 439$, namely $pr439$. For illustration, we report the results found where $p = 80, 90$ and 100, see Table 2. The proposed enhanced algorithm found the optimal solution in an improved computational time compared to the initial Drezner algorithm. This modified algorithm will be tested on other data sets more extensively and, if need be, we may need to explore further enhancements.

Acknowledgments

The first author would also like to thank EPSRC for her PhD studentship.

Iteration #	# Original Circles	# Maximal Circles	# Circles Previously Identified	Extra % Required
1	9281	860	-	-
2	9189	855	780	8.77
3	8835	797	597	25.09
4	8796	805	758	5.84
5	8652	809	684	15.45
Average	**8951**	**825**	**705**	**13.79**

Table 1
Number of maximal circles required & previously identified for the first 10 iterations ($n = 439, p = 100$)

			Initial Algorithm	Enhanced Algorithm
p	Z^*	# Iterations	Total CPU Time (secs)	Total CPU Time (secs)
80	312.500	53	2186.27	173.85
90	280.903	38	1258.22	88.94
100	256.680	30	462.30	40.97
Average	**283.361**	**44**	**1302.26**	**101.25**

Table 2
Initial Algorithm vs Enhanced Algorithm for $n = 439$ TSP-Lib

References

[1] Drezner, Z. (1984). The $p-$centre Problem - Heuristic and Optimal Algorithms. Journal of the Operational Research Society, 35 (8), 741-748.

[2] Elshaikh, A., Salhi, S., & Nagy, G. (2015). The continuous $p-$centre problem: An investigation into variable neighbourhood search with memory. European Journal of Operational Research, 241, 606-621.

[3] Elzinga, J., & Hearn, D. (1972). Geometric Solutions for some Minimax Location Problems. Transportation Science, 6, 379-394.

New methods for the Distance Geometry Problem

Gustavo Dias and Leo Liberti [1,2]

CNRS LIX, Ecole Polytechnique, 91128 Palaiseau, France

Abstract

Given an integer K and a simple edge-weighted undirected graph $G = (V, E)$, the Distance Geometry Problem questions the existence of a vertex realization function $V \to \mathbb{R}^K$ such that each vertex pair adjacent to an edge is placed at a distance which is equal to the edge weight. This problem has many applications to science and engineering, and many methods have been proposed to solve it. We propose some new formulation-based methods.

Keywords: DGP, Semidefinite Programming, Diagonally dominant matrices.

1 Introduction

The problem studied in this paper is the

> Distance Geometry Problem (DGP). Given an integer $K \geq 1$ and a simple, edge-weighted, undirected graph $G = (V, E, d)$, where $d : E \to \mathbb{R}_+$, verify the existence of a vertex *realization* function $x : V \to \mathbb{R}^K$ such that:
>
> $$\forall \{i, j\} \in E \quad \|x_i - x_j\| = d_{ij}. \tag{1}$$

[1] Gustavo Dias is financially supported by a CNPq PhD thesis award.

[2] Email: {dias,liberti}@lix.polytechnique.fr

A recent survey on the DGP with the Euclidean norm is given in [2]. The DGP is **NP**-hard, by reduction from Partition. Three well-known applications are to clock synchronization ($K = 1$), sensor network localization ($K = 2$), and protein conformation ($K = 3$). A related problem, the Distance Matrix Completion Problem (DMCP), asks whether a partially defined matrix can be completed to a distance matrix. The difference is that while K is part of the input in the DGP, it is part of the output in the DMCP, in that a realization into *any* Euclidean space which allows the computation of the missing distances provides a certificate. It is remarkable that, by virtue of this seemingly minor difference, it is not known whether the Euclidean DMCP (EDMCP) is in **P** or **NP**-hard. It is currently thought to be "between the two classes".

In this short paper we sketch several new formulation-based methods for solving the DGP.

2 MILP formulations for 1- and ∞-norms

To the best of our knowledge, no method for solving DGPs with the 1- and ∞-norm currently exists.[3] Yet, since both norms can be linearized exactly, it is not difficult to derive Mixed-Integer Linear Programming (MILP) formulations for either. We first re-write Eq. (1) as follows:

$$\min_x \sum_{\{i,j\}\in E} |\, \|x_i - x_j\|_\ell - d_{ij}\,|, \tag{2}$$

for $\ell \in \{1, \infty\}$. Then, for $\ell = 1$ we write:

$$\min_x \sum_{\{i,j\}\in E} \left| \sum_{k\le K} |x_{ik} - x_{jk}| - d_{ij} \right|,$$

and equivalently for $\ell = \infty$. For $\ell = 1$, we apply some standard absolute value reformulations to obtain a MILP. The case $\ell = \infty$ is slightly more involved, but still easy to model. These formulations can be solved using any off-the-shelf MILP solver.

[3] We shall gladly take corrections to this statement!

3 SDP formulations for the 2-norm

Many Semidefinite Programming (SDP) formulations for the 2-norm case are well known from the sensor network localization literature (see [2]). Note that a realization x can be represented in matrix form by an $n \times K$ matrix where $n = |V|$, and where each of the n rows is a vector $x_i \in \mathbb{R}^K$ which places vertex $i \in V$. The Euclidean DGP (EDGP) can be modelled as follows:

$$\forall \{i,j\} \in E \quad \|x_i - x_j\|_2^2 = x_i \cdot x_i + x_j \cdot x_j - 2x_i \cdot x_j = d_{ij}^2. \tag{3}$$

Since the EDGP involves sums $x_i \cdot x_j$ of quadratic terms for various $i, j \in V$, we can linearize these sums by replacing them with variables X_{ij} organized in an $n \times n$ matrix, i.e. $X = xx^\top$. This provides an easy reformulation of Eq. (3):

$$\begin{aligned} \forall \{i,j\} \in E \quad X_{ii} + X_{jj} - 2X_{ij} &= d_{ij}^2 \\ X &= xx^\top. \end{aligned}$$

The rank constraint $X = xx^\top$ can be readily relaxed to $X \succeq xx^\top$, which in turn can be written as the Schur complement $\begin{pmatrix} I_K & x^\top \\ x & X \end{pmatrix} \succeq 0$, yielding a well-known pure feasibility SDP formulation. Usually, in an attempt to reduce the rank of the solution X, many papers propose the objective function $\min \mathsf{trace}(X)$. Some empirical experience suggests that this particular objective is suitable for instances from the sensor network localization application, since the so-called "anchor nodes" are usually evenly scattered among the sensors, and play a regularization role. For protein conformation instances, on the other hand, trace minimization yields poor results. A better formulation turns out to be:

$$\begin{aligned} \min_X \sum_{\{i,j\} \in E} (X_{ii} + X_{jj} - 2X_{ij}) & \\ \forall \{i,j\} \in E \quad X_{ii} + X_{jj} - 2X_{ij} &\geq d_{ij}^2 \\ X - xx^\top &\succeq 0. \end{aligned}$$

For the EDMCP, where the rank is of no importance, we only require that X should be the Gram matrix of a realization x (of any rank). Since Gram matrices are exactly positive semidefinite (PSD) matrices, the formulation is simplified to

$$\begin{aligned} \forall \{i,j\} \in E \quad X_{ii} + X_{jj} - 2X_{ij} &= d_{ij}^2 \\ X &\succeq 0. \end{aligned}$$

4 Diagonally dominant approximation

One serious drawback of SDP is that current solving technology is limited to instances of fairly low sizes. A. Ahmadi recently remarked [1] that diagonal dominance provides a useful tool for inner approximating the PSD cone. A matrix (Y_{ij}) is *diagonally dominant* (DD) if

$$\forall i \le n \quad Y_{ii} \ge \sum_{j \ne i} |Y_{ij}|. \tag{4}$$

It follows from Gershgorin's theorem that diagonally dominant matrices are PSD (the converse does not hold, hence the inner approximation). The crucial observation is that Eq. (4) is easy to linearize as follows:

$$\begin{aligned} \forall i \le n \quad & \sum_{j \ne i} T_{ij} \le Y_{ii} \\ \forall i, j \le n \quad & -T_{ij} \le Y_{ij} \le T_{ij}. \end{aligned}$$

This yields a new LP formulation related to the EDGP:

$$\left.\begin{array}{rr} \min\limits_{X,Y,T} \sum\limits_{\{i,j\} \in E} (X_{ii} + X_{jj} - 2X_{ij}) & \\ \forall \{i,j\} \in E & X_{ii} + X_{jj} - 2X_{ij} \ge d_{ij}^2 \\ & \begin{pmatrix} I_K & x^\top \\ x & X \end{pmatrix} = Y \\ \forall i \le n+K & \sum\limits_{\substack{j \le n+K \\ j \ne i}} T_{ij} \le Y_{ii} \\ & -T \le Y \le T. \end{array}\right\} \tag{5}$$

Let $\mathcal{D}(U) = \{U^\top M U \mid M \text{ is DD}\}$. The approximation Eq. (5) can be iteratively improved by requiring that $Y \in \mathcal{D}(U)$ with $U^0 = I$ and $U^h = \sqrt{\bar{Y}^{h-1}}$ for all $h > 0$, where $\bar{Y}^{h-1}$ is the solution of Eq. (5) at the previous iteration $h - 1$.

References

[1] Ahmadi, A. and G. Hall, *Sum of squares basis pursuit with linear and second order cone programming*, Technical Report 1510.01597v1, arXiv (2015).

[2] Liberti, L., C. Lavor, N. Maculan and A. Mucherino, *Euclidean distance geometry and applications*, SIAM Review **56** (2014), pp. 3–69.

Stochastic geometric programming with joint probabilistic constraints

Jia Liu[a,b], Abdel Lisser[b], Zhiping Chen[a]

a. School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an, Shaanxi, 710049, P. R. China

b. Laboratoire de Recherche en Informatique, Université Paris Sud, Bat 650 Ada Lovelace, Orsay, 91405, France

Abstract

This paper discusses the geometric programs with joint probabilistic constraints. When the coefficients are normally distributed and independent of each other, we approximate the problem by using piecewise linear function and transform the approximation problem into a geometric program. We prove that this approximation method provides a lower bound, and we use Bonferroni approximation to find an upper bound.

Keywords: Geometric programs; Joint probabilistic constraints; Piecewise linear approximation; Bonferroni approximation.

1 Introduction

Geometric programs are a type of optimization problems characterized by an objective and constraints functions which have a special form [2]. In real world

[1] Jia Liu's research was supported by the China Scholarship Council (CSC).

[2] E-mails: liu.jia@stu.xjtu.edu.cn; lisser@lri.fr; zchen@xjtu.edu.cn

applications, some of the coefficients in a geometric program may not be known precisely when the optimization is made. Hence, the stochastic geometric programming models are proposed to model geometric problems with random parameters. Individual probabilistic constraints have been applied to control the uncertainty level of geometric constraints [4,5,7].

In this paper, we furthermore consider the following joint probabilistic constrained stochastic geometric programs:

$$\min_t E\left[\sum_{i\in I_0} c_i \prod_{j=1}^{M} t_j^{a_{ij}}\right] \tag{1}$$

$$\text{s.t. } P\left(\sum_{i\in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq 1,\ k=1,\cdots,K\right) \geq 1-\epsilon. \tag{2}$$

Here, $\{I_k,\ k=0,\cdots,K\}$ is a decomposition of $\{1,\cdots,Q\}$ into $K+1$ disjoint index sets. Q is the total number of monomilas $c_i\prod_{j=1}^{M} t_j^{a_{ij}}$ in (1) and (2). Unlike [4,5,7], we require that the overall probability of meeting the K geometric constraints is above a certain probability level $1-\epsilon$, $\epsilon\in(0,0.5]$.

The stochastic geometric program with joint probabilistic constraints is a special kind of joint probabilistic constrained problems. The linear programs with joint probabilistic constraints are widely studied in [1,3,6].

2 Approximation methods

Similarly to [4], we suppose that a_{ij} is deterministic and c_i is normally distributed and independent of each other, i.e., $c_i \sim N(E_{c_i},\sigma_i^2)$.

As c_i are independent, (2) is equivalent to

$$\prod_{k=1}^{K} P(\sum_{i\in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq 1) \geq 1-\epsilon. \tag{3}$$

By introducing auxiliary variables $y_k\in\mathbb{R},\ k=1,\cdots,K$, (3) can be equivalently transformed into

$$P(\sum_{i\in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq 1) \geq y_k,\ k=1,\cdots,K, \tag{4}$$

and

$$\prod_{k=1}^{K} y_k \geq 1-\epsilon,\ y_k\geq 0. \tag{5}$$

From [4], we know that for independent normally distributed $c_i \sim N(E_{c_i}, \sigma_i^2)$, constraint (4) is equivalent to

$$\sum_{i \in I_k} E_{c_i} \prod_{j=1}^{M} t_j^{a_{ij}} + \Phi^{-1}(y_k) \sqrt{\sum_{i \in I_k} \sigma_i^2 \prod_{j=1}^{M} t_j^{2a_{ij}}} \leq 1, \;\; k = 1, \cdots, K. \tag{6}$$

Here, $\Phi^{-1}(y_k)$ is the quantile of the standard normal distribution $N(0,1)$. However, inequalities (6) are still very hard to handle in an optimization problem due to the nonlinear property of $\Phi^{-1}(\cdot)$.

2.1 Piecewise linear approximation

In this paper, we consider a piecewise linear approximation of $\Phi^{-1}(y_k)$ [3]. We choose S linear segments $F_s(y_k) = a_s y_k + b_s$ such that $a_s, b_s \geq 0$, $s = 1, \cdots, S$ and $F_s(y_k) \leq \Phi^{-1}(y_k)$ for all $y_k \in [1-\epsilon, 1]$. $\Phi^{-1}(y_k)$ is then approximated by

$$F(y_k) = \max_{s=1,\cdots,S} F_s(y_k).$$

Theorem 2.1 *Using the piecewise linear $F(y_k)$, we can found an approximation of the geometric problem with joint probabilistic constraints:*

$$\min_{t,y} \sum_{i \in I_0} E_{c_i} \prod_{j=1}^{M} t_j^{a_{ij}} \tag{7}$$

$$\text{s.t.} \sum_{i \in I_k} E_{c_i} \prod_{j=1}^{M} t_j^{a_{ij}} + (a_s y_k + b_s) \sqrt{\sum_{i \in I_k} \sigma_i^2 \prod_{j=1}^{M} t_j^{2a_{ij}}} \leq 1,$$

$$s = 1, \cdots, S, \;\; k = 1, \cdots, K, \tag{8}$$

$$\prod_{k=1}^{K} y_k \geq 1 - \epsilon, \;\; y_k \geq 0. \tag{9}$$

The optimal value of the approximation problem (7)-(9) is a lower bound of the problem (1)-(2).

Although problem (7)-(9) is not convex with respect to t and y, but it is convex with respect to $r_j = log(t_j)$, $j = 1, \cdots, M$ and $x_k = log(y_k)$, $k = 1, \cdots, K$, hence interior-point methods still can be efficiently used to solve the problem and provide a lower bound for the joint probabilistic problem (1)-(2).

2.2 Bonferroni approximation

In order to give an estimation about the upper bound of the joint probabilistic problem (1)-(2), we adopt the popular Bonferroni approximation, which gives probabilistic measure to the individual constraints. We set a group of estimations ϵ_k, $k = 1, \cdots, K$, such that $\epsilon_k \geq 0$, $k = 1, \cdots, K$, and $\sum_{k=1}^{K} \epsilon_k = \epsilon$. Then we use the following individual probabilistic constraint:

$$P(\sum_{i \in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq 1) \geq 1 - \epsilon_k, \; k = 1, \cdots, K. \tag{10}$$

to approximate the joint probabilistic constraint (2). By using the same method in [4], these individual constraints can be formulated as posynomial forms and the optimization problems with the individual constraints can also be solved by interior-point methods.

As Bonferroni approximation gives one possible allocation of the total tolerance probability, the optimal solution of the problem with constraints (10) is a feasible solution for the original problem (1)-(2). Hence, it provides an upper bound.

References

[1] Ackooij, W. V., R. Henrion, A. Moller, and R. Zorgati, *On joint probabilistic constraints with Guassian coefficient matrix*, Oper. Res. **39** (2011) 99–102.

[2] Boyd, S., S. J. Kim, L. Vandenberghe, and A. Hassibi, *A tutorial on geometric programming*, Optim. Eng. **8** (2007), 67–127.

[3] Cheng, J., and A. Lisser, *A second-order cone programming approach for linear programs with joint probabilistic constraints*, Oper. Res. Letters **40** (2012), 325–328.

[4] Dupačová, J., *Stochastic geometric programming: approaches and applications*, In V. Brožová, R Kvasnička, eds., Proceedings of MME09 (2009), 63 - 66.

[5] Iwata, K., Y. Murotsu, and T. Iwatsubo, *A probabilistic approach to the determination of the optimal cutting conditions*, J. of Engineering for Industry Trans. of ASME **94** (1972), 1099–1107

[6] Prékopa, A., "Stochastic Programming", Kluwer Academic Publishers, Dordrecht, Boston, (1995).

[7] Rao, S. S., "Engineering Optimization: Theory and Practice," 3rd Ed., Wiley-Interscience, New York, (1996).

Solving LP using random projections

Leo Liberti, Pierre-Louis Poirion, Vu Khac Ky [1,2]

CNRS LIX, Ecole Polytechnique, 91128 Palaiseau, France

Abstract

A celebrated result of Johnson and Lindenstrauss asserts that, in high enough dimensional spaces, Euclidean distances defined by a finite set of points are approximately preserved when these points are projected to a certain lower dimensional space. We show that the distance from a point to a convex set is another approximate invariant, and leverage this result to approximately solve linear programs with a logarithmic number of rows.

Keywords: Johnson-Lindenstrauss lemma, random projection.

1 Introduction

One of the computational “grand challenges” in Mathematical Programming is to solve ever larger Linear Programs (LP). We are currently able to routinely solve (sparse) LPs with a million variables and constraints. Developers of commercial solvers have seen customer LPs with up to a hundred million variables. What about a billion? This short paper is unfortunately *not* announcing such a breakthrough, but it possibly paves the way — if one is willing to accept an approximate solution with high probability.

[1] Email: {liberti,poirion,vu}@lix.polytechnique.fr

[2] Vu Khac Ky is supported by a Microsoft Research PhD grant.

We want to find approximate solution of LPs in standard form

$$\min\{cx \mid Ax = b \wedge x \geq 0\}, \tag{1}$$

with high probability, where A is an $m \times n$ matrix, $c \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$. The general idea is as follows: we pre-multiply A and b by a certain $k \times m$ matrix T (sampled randomly from certain distributions), with $k \ll m$. T is guaranteed with high probability to approximately preserve Euclidean distances among the columns of A and b. Since the worst-case complexity LP methods depends on both n and m, a large decrease in the number of rows is likely to have a beneficial impact on efficiency, and to allow for solving larger instances.

Such *random projection* methods are at the heart of the proof of the Johnson-Lindenstrauss Lemma (JLL), which states that, for any finite set $X \subseteq \mathbb{R}^m$ with $|X| = n$ and $\varepsilon \in (0,1)$ there exists a k of order $O(\frac{1}{\varepsilon^2} \ln n)$ and a mapping $T : \mathbb{R}^m \to \mathbb{R}^k$ such that:

$$\forall x, y \in X \quad (1-\varepsilon)\|x - y\|_2 \leq \|Tx - Ty\|_2 \leq (1+\varepsilon)\|x - y\|_2. \tag{2}$$

From here onwards, norms will always be Euclidean unless specified otherwise.

Random projections have been used previously to address optimization and/or learning algorithms involving the Euclidean norm only (see e.g. [2,1]). This is their natural setting, since a set of Euclidean distances is rotationally independent and rotational independence plays a prominent role in the original proof in the JLL [3]. As far as we know, this is the first application of the approximate preservation of the orthant $x \geq 0$ (which is definitely *not* rotationally independent), and is therefore interesting in its own right from a theoretical point of view.

For a matrix A we denote the i-th row by A_i and the j-th column by A^j. For a vector v and an index set J, we let $v_J = (v_j \mid j \in J)$. Let $\mathscr{C}(A) = \mathsf{cone}(A^j \mid j \leq n)$. For a problem P let $\mathcal{F}(P)$ be its feasible region.

2 A randomized algorithm for large LPs

Our proposed algorithm is as follows.

1. **Sample** a $k \times m$ *random projector* matrix T.
2. **Solve** $TP \equiv \min\{cx \mid TAx = Tb \wedge x \geq 0\}$,
   let c' be its optimal objective function value.
3. **Retrieve** an approximately optimal solution x^* of P as follows:
   a. let $A'x = b'$ be the system $TAx = Tb \wedge cx = c'$,

```
   let $\alpha$ be a uniform random vector in $\mathbb{R}^n$;
b. solve $TP_\alpha \equiv \min\{\alpha x \mid A'x = b' \wedge x \geq 0\}$,
   let $y'$ be its optimal dual vector and $y = T^\top y'$;
c. let $J$ be the set of indices $j \leq n$ such that $yA^j = \alpha_j$,
   set $x^*_i = 0$ for each $j \notin J$;
d. let $\bar{x}$ be the solution of the $k \times k$ system $(A^J)^\top A^J x_J = (A^J)^\top b$,
   let $x^*_j = \bar{x}_j$ for each $j \in J$.
```

In the rest of this paper, we shall sketch the reason why this algorithm works.

3 The random projector

Among the many distributions that T can be sampled from, the simplest has each component of T sampled independently from $\mathcal{N}(0, \frac{1}{\sqrt{k}})$. Since T is a linear map, it obviously preserves feasibility. In the (yet unpublished) report [4], we prove that, if b, A^j are unit vectors for $j \leq n$ and $b \notin \mathscr{C}(A)$, then $\exists \mathcal{C} > 0$ such that:

$$\mathsf{Prob}(Tb \notin \mathscr{C}(TA)) \geq 1 - 2n(n+1)e^{-\mathcal{C}(\varepsilon^2 - \varepsilon^3)k}$$

for all $\varepsilon > 0$ in a certain "reasonable" interval. Since $b \in \mathscr{C}(A)$ iff $\exists x \geq 0$ s.t. $Ax = b$, our result shows that if P is infeasible then TP highly likely to be infeasible, and this probability can be made arbitrarily close to 1 as k grows.[3]

4 Solving the projected LP

Since $\mathcal{F}(P) = \mathcal{F}(TP)$ with high probability, a bisection argument shows that P and TP both have objective function values c' with high probability. Thus, we can find c' by simply solving TP using a standard LP solver. On the other hand, we can prove that the primal solution x' of TP is infeasible in P with probability 1, so we need a different strategy to compute the certificate.

5 Solution retrieval

Steps a-d in the algorithm of Sect. 2 provide a primal solution retrieval method via the dual LP using complementary slackness. The dual y' of P_α is such that $y'A' \leq \alpha$. Since $A' = (TAc)^\top$, we write $y' = (\bar{y}, y^c)$ so that we have $\bar{y}TA + y^c c \leq \alpha$ $(\star)$. Letting $y = (\bar{y}T, y^c)$ we have $y(Ac)^\top \leq \alpha$ $(\dagger)$, which

[3] I.e. as m grows, which, since P is in standard form, also means that n grows.

means that y is a valid dual solution to the problem $P_\alpha = \min\{\alpha x \mid Ax = b \wedge cx = c' \wedge x \geq 0\}$. By complementary slackness of TP_α, at least k of the n inequalities in $(\star)$ are satisfied at equality (say those corresponding to the index set J), which means the same holds for $(\dagger)$. By complementary slackness of P_α, $\forall j \notin J$ we have $x^*_j = 0$. The nonzero components of x^* are those indexed by J, and we can find them by identifying the corresponding k columns of $Ax = b$ and then solving a $k \times k$ linear system.

6 Perspectives

So, how far are we down the road to solving large LPs? If we only consider dense, randomly generated feasibility problems $Ax = b \wedge x \geq 0$, the following table shows that this approach does actually save us some time.

Uniform	ε	$k \approx$	CPU savings	accuracy
$(0,1)$	0.1	$0.5m$	10%	100%
$(0,1)$	0.15	$0.25m$	90%	100%
$(0,1)$	0.2	$0.12m$	97%	100%
$(-1,1)$	0.1	$0.5m$	30%	50%
$(-1,1)$	0.15	$0.25m$	92%	0%
$(-1,1)$	0.2	$0.12m$	99.2%	0%

For sparse LPs, as expected, the issues concerning size, values of the constant $\mathcal{C}$, and values of ε (none of which we know how to estimate, much less compute) make it impossible to obtain any CPU time saving. For validation purposes, we ran a simple experiment on the `afiro` and `recipe` instances of the NetLib [5], and obtained a valid objective function value and primal solutions in around 10% and 20% of the total number of independent runs of our randomized algorithm.

References

[1] Ailon, N. and B. Chazelle, *Approximate nearest neighbors and fast Johnson-Lindenstrauss lemma*, in: *Proceedings of the Symposium on the Theory Of Computing*, STOC **06** (2006).

[2] Indyk, P. and A. Naor, *Nearest neighbor preserving embeddings*, ACM Transactions on Algorithms **3** (2007), p. Art. 31.

[3] Johnson, W. and J. Lindenstrauss, *Extensions of Lipschitz mappings into a Hilbert space*, in: G. Hedlund, editor, *Conference in Modern Analysis and Probability*, Contemporary Mathematics **26** (1984), pp. 189–206.

[4] Ky, V. K., P.-L. Poirion and L. Liberti, *Using the johnson-lindenstrauss lemma in linear and integer programming*, Technical Report 1507.00990v1[math.OC], arXiv (2015).

[5] NetLib, *LP instance library* (2015), `http://www.netlib.org/lp/`.

The Prize-collecting Scheduling Problem with Deadlines

Pierre Hosteins[1]

Computer Science Department
Università di Torino
Turin, Italy

Roberto Cordone[2] Giovanni Righini[3]

Computer Science Department
Università degli studi di Milano
Milan, Italy

Abstract

We study a prize-collecting single machine scheduling problem with hard deadlines, where the objective is to minimise the difference between the total tardiness and the total prize of selected jobs. This problem is motivated by industrial applications, both as a standalone model and as a pricing problem for column generation approaches to parallel machine scheduling problems. It is handled through the use of exact approaches, in the form of a Branch and Bound (B&B) algorithm and an Integer Linear Programming (ILP) formulation. The B&B and ILP formulation are compared in their efficiency on randomly generated benchmark instances.

Keywords: Prize-collecting, Single Machine Scheduling, Total Tardiness, Integer Linear Programming, Hard Deadlines, Branch and Bound.

1 Introduction

Even though the single machine scheduling problem has been given a tremendous amount of attention in the literature, the case where the jobs are subject to hard deadlines constraints ($1/\bar{d}_j/\sum_j T_j$) has been given a lot less attention: the work of [3] provides rules for the case of a single machine total tardiness minimisation with hard deadlines and proposes a B&B framework to solve the general case. Another recent trend of research is the prize-collecting generalization, where (at least part of) the jobs are no more obligatory and have a prize attached, that one can claim if the job is performed [2]. In such a setup, the processing time available is insufficient to perform all jobs and one has to select a subset so as to maximise profit, taking into account possible losses due to the total tardiness of the jobs. Examples of such a situation can be found in make-to-order production systems with limited production capacity and tight delivery requirements as well as scheduling with an outsourcing option. Another situation where a similar setup has been investigated is the pricing problem of multi-machine scheduling problems solved through Column Generation formulations, which ends up providing an example of a prize-collecting scheduling problem. However, such cases are mainly investigated for easier additive objective functions, such as the total (weighted) completion time of jobs or the (weighted) number of tardy jobs, see for example the works of [1]. The prize-collecting scheduling problems considered in these works are solved through pseudo-polynomial algorithms.
We provide dominance rules between the jobs in the prize-collecting case with total tardiness and a general B&B algorithm to solve the problem exactly.

2 Problem Formulation

The problem at hand consists of a set of jobs J (with cardinality $n = |J|$) that can be selected for processing by a machine with limited available working time T. Each job $j \in J$ has a certain number of characteristics: a processing time p_j; a due date d_j; a hard deadline $\bar{d}_j$; an associated prize λ_j. Finally, a subset $J_f \subseteq J$ of jobs that have to be processed obligatorily is introduced. The aim is to maximise the total prize of the jobs processed and to minimise the total tardiness while respecting the deadlines. The tardiness T_j of job j is, as usual, the difference between the completion time of the job and its due

[1] Email: `hosteins@di.unito.it`

[2] Email: `roberto.cordone@unimi.it`

[3] Email: `giovanni.righini@unimi.it`

date if the former exceeds the latter; otherwise, it is zero.

A linear formulation for the problem is obtained using positional date variables, taking into account the facultative nature of the jobs, in particular binary variables: $x_{jp} \in \{0,1\}$, equal to 1 if job $j \in J$ is in position $p \in \{1,...,n\}$ in the schedule and $y_p \in \{0,1\}$, equal to 1 if the scedule includes a job in position $p \in \{1,...,n\}$. With the help of tardiness variables T_p for job in position p the formula $\sum_{p\in\{1,...,n\}}\Big(T_p - \sum_{j\in J}\lambda_j x_{jp}\Big)$ defines the objective function as the difference between the total tardiness and the total prize. It is, however, more useful to define the objective function with an added constant $\sum_{j\in J}\lambda_j$ such that the objective function is always positive. The constraints of the problem (using the completion time variable C_p for position $p \in \{1,...,n\}$) can be written in the following manner:

$$y_p = \sum_{j\in J} x_{jp}, \quad p \in \{1,...,n\}, \qquad y_p \geq y_{p+1}, \quad p \in \{1,...,n-1\}. \tag{1}$$

$$C_p \geq \sum_{p'\leq p}\sum_{j\in J} p_j x_{jp'} - M(1-y_p), \quad p \in \{1,...,n\}, \tag{2}$$

$$C_p - \sum_{j\in J} d_j x_{jp} \leq T_p \leq \sum_{j\in J}(\bar{d}_j - d_j)x_{jp}, \quad p \in \{1,...,n\}, \tag{3}$$

$$\sum_{p\in\{1,...,n\}} x_{jp} \leq 1, \quad j \in J\setminus J_f, \qquad \sum_{p\in\{1,...,n\}} x_{jp} = 1, \quad j \in J_f. \tag{4}$$

3 Branch and Bound

An exact approach to optimise our prize collecting scheduling problem is designed by branching on the jobs that are not obligatory, in order to include or reject them from the schedule by fixing a binary variable $f_j = \sum_{p\in\{1,...,n\}} x_{jp}$ to 1 or 0 (defining respectively subsets of jobs J_1 and J_0). The general structure of the B&B for each given node ν of the branching tree is the following:

- compute the cost of ν by computing the total tardiness of jobs J_1 through the B&B procedure of [3] (tardiness subproblem);
- obtain a lower bound LB^ν through an ILP model which is a partial relaxation of the full problem where additional jobs are included into the schedule without taking into account the additional tardiness they might introduce, such that the objective function is decreased by the total sum of their prizes (prize subproblem);
- compute an upper bound UB^ν by inserting jobs of decreasing λ_j/p_j ratio

inside the optimal schedule of jobs J_1, at the position that introduces the smallest tardiness (starting from the solution of the tardiness subproblem for J_1); if $UB^\nu < UB^*$, update UB^*; if $LB^\nu > UB^*$, prune the node;

- if the node is not pruned, find the job $j \in J \setminus (J_0 \cup J_1)$ that belongs to the solution of the prize subproblem and has the highest λ_j/p_j (such a job always exists unless $LB^\nu = UB^\nu$); generate two children nodes fixing f_j to either 1 or 0;

4 Numerical Results

Our B&B algorithm has been tested on randomly generated instances with $|J|$ from 20 to 200 jobs against the resolution of ILP models using the C++ library of `CPLEX 12.5.1`. Among our 25 instances, the B&B could manage to close the gap between lower and upper bounds for all but one within a running time limit of 1000 seconds, while ILP models are usually unable to close the gap for instances with 80 jobs or more. Including a proportion of obligatory jobs confirms the relative velocity of our algorithm. When the time cutoff is lowered to 100 seconds, the results are even more interesting as the B&B solution remains near optimal for all instances while for the largest ones, the gap of ILP models gets much bigger, which prevents those models to be used for finding quickly good heuristic solutions.

References

[1] Z.L. Chen and W.B. Powell, *Solving Parallel Machine Scheduling Problems by Column Generation*, INFORMS Journal on Computing, 11:1 (1999) 78–94.
J.M. van den Akker, J.A. Hoogeveen and S.L. van de Velde, *Parallel Machine Scheduling by Column Generation*, Operations Research 47:6 (1999) 862–872.

[2] D. Shabtay, N. Gaspar and M. Kaspi, *A survey on offline scheduling with rejection*, Journal of Scheduling 16 (2013) 3–28.

[3] R. Tadei, A. Grosso and F. Della Croce, *Finding the Pareto-optima for the total and maximum tardiness single machine problem*, Discrete Applied Mathematics 124 (2002) 117–126.

Column generation for the variable cost and size bin packing problem with fragmentation

Marco Casazza[1]

UPMC University Paris 06, UMR 7606, LIP6, F-75005, Paris, France

Alberto Ceselli[2]

Dipartimento di Informatica, Università Degli Studi di Milano, Crema, Italy

Abstract

Bin Packing Problems with Item Fragmentation (BPPIF) are variants of classical Bin Packing in which items can be split at a price. We extend BPPIF models from the literature by allowing a set of heterogeneous bins, each potentially having a different cost and capacity. We introduce extended formulations and column generation algorithms to obtain good bounds with reasonable computing effort. We test our algorithms on instances from the literature. Our experiments prove our approach to be more effective than state-of-the-art general purpose solvers.

Keywords: Bin Packing, Item Fragmentation, Variable Cost and Size, Column Generation.

1 Introduction

Bin Packing Problems with Item Fragmentation (BPPIF) haves been introduced to model problems in diverse domains, like routing of consolidated traffic in optical networks and VLSI circuit design [1]. In their bin-minimization variant a set I of items, each having a size d_i, and a set of bins J, each having a capacity C, are given, together with a *fragmentation budget* F. The aim is to assign items to the minimum number of bins; up to F item splits are

[1] Email: `marco.casazza@upmc.fr`

[2] Email: `alberto.ceselli@unimi.it`, partially funded by Regione Lombardia - Fondazione Cariplo, grant n. 2015-0717, project "REDNEAT"

allowed: whenever an item is split, it is replaced by two fragments; the split point is arbitrary, but the sum of fragment sizes must equal the size of the original item. Recursive fragmentations are allowed, but each split counts in the budget. The final set of fragments need then to be assigned to the bins, in such a way that the sum of item (fragment) sizes assigned to the same bin do not exceed C. Recent contributions to the field include both approximation algorithms [3] and exact methods [2], both approaches proving to be effective. As stressed in [3], major interest is currently in making BPPIF models more flexible. In this paper we tackle the generalization of BPPIF, in its bin-minimization variant, in which each bin $j \in J$ has a potentially different cost v_j and capacity c_j, and the overall cost of the used bins needs to be minimized. We refer to our generalization as the Variable Cost and Size BPPIF (VCSB).

2 Model

We first observe the following.

Proposition 2.1 *An optimal VCSB solution always exists, in which (a) each item is split in at most two fragments (b) each bin contains at most two fragmented items (c) each set of k bins contains at most $k-1$ fragmented items.*

Any solution satisfying (a)–(c) is called *primitive* [1]. A formal proof is omitted, but intuitively given a set of k bins and a solution assigning a subset of items $\bar{I} \subseteq I$ to them, a *Next Fit with Fragmentations* procedure produces a fragmentation pattern that comply with (a)–(c). Fragmented items link one bin another in a *chain* structure, that includes a subset $\bar{I} \subseteq I$ of items and a subset $\bar{J} \subseteq J$ of bins. On feasible chains it always holds $\sum_{i \in \bar{I}} d_i \leq \sum_{j \in \bar{J}} c_j$. Let Ω be the set of all feasible chains. Following the framework of [2] we model the VSCB with the following chain-based extended formulation:

$$\min \sum_{p \in \Omega} (\sum_{j \in J} v_j \bar{y}_j^p) z^p \tag{1}$$

$$\text{s.t.} \sum_{p \in \Omega} \bar{x}_i^p z^p = 1 \qquad \forall i \in I \tag{2}$$

$$\sum_{p \in \Omega} \bar{y}_j^p z^p \leq 1 \qquad \forall j \in J \tag{3}$$

$$\sum_{p \in \Omega} \bar{f}^p z^p \leq F \tag{4}$$

$$z^p \in \{0,1\} \qquad \forall p \in \Omega. \tag{5}$$

Coefficient $\bar{x}_i^p$ (resp. $\bar{y}_j^p$) is 1 if item i (resp. bin j) is included in chain p, 0 otherwise. Coefficient $\bar{f}^p$ is the number of fragmentations performed in chain p. Binary variables z^p are 1 if chain p is selected, 0 otherwise. Since $(\sum_{j\in J} v_j \bar{y}_j^p)$ represents the cost of using the set of bins in chain p, the objective function (1) aims at minimizing the overall cost of selected chains. Constraints (2) ensure that each item is included in a selected chain. Constraints (3) ensure that each bin is included in at most one selected chain. Constraints (4) enforce the fragmentations budget to be respected.

3 Algorithms

Formulation (1)–(5) includes an exponential number of variables. In order to obtain dual bounds on the value of the optimal solution we relax integrality conditions and exploit column generation techniques. Without loss of quality in the bound, we also relax constraints (2) in $\geq$ form. Let $\lambda_i \geq 0$, $\mu_j \leq 0$ and $\eta \leq 0$ be the dual variables associated to constraints (2), (3) and (4), resp.. The associated pricing problem is the following.

$$\begin{aligned}
\min & \sum_{j\in J}(v_j - \mu_j) - \sum_{i\in I}\lambda_i x_i - \eta f \\
\text{s.t.} & \sum_{i\in I} d_i x_i \leq \sum_{j\in J} c_j y_j \\
& \sum_{j\in J} y_j \leq f + 1 \\
& x_i \in \{0,1\} \quad \forall i \in I, \qquad y_j \in \{0,1\} \quad \forall j \in J, \qquad f \geq 0
\end{aligned}$$

Since if $\sum_{j\in J} y_j = 0$ also $\sum_{i\in I} x_i = 0$, such a setting is never profitable. Therefore we set $f = (\sum_{j\in J} y_j) - 1$, obtaining a variant of a 0–1 Knapsack Problem (KP) in which capacity consumption has a (possibly non monotone) cost. We solve it with an ad-hoc pseudo-polynomial time algorithm, whose main idea is to find, for each value of capacity $c = 0 \ldots \sum_{j\in j} c_j$ (a) the combination of bins of minimum reduced cost giving at least overall capacity c (b) the combination of items of minimum reduced cost using at most capacity c (c) sum up these two contributions to obtain an optimal pricing solution using capacity c (d) return the best pricing solution over all values of c. The key observation is that both steps (a) and (b) can be performed by solving *a single* KP each, that in turn can be done in pseudo-linear time. Therefore, our pricing algorithm has pseudo-linear time as well. When column generation is over we also perform rounding to search for good primal solutions.

Instances		CPLEX			CG		
Cap.	Weight	S	G(%)	T(s)	S	G(%)	T(s)
T	L	0	8.57	0.11	9	0.49	0.64
T	M	0	10.00	0.04	0	9.15	1.06
T	S	0	16.67	0.03	0	16.53	0.99
L	L	0	25.00	0.10	10	0.00	0.44
L	M	0	12.12	0.10	4	2.24	0.60
L	S	1	10.00	0.06	0	10.43	0.90
Overall		1	13.73	0.073	23	6.47	0.771

Table 1
Results on instances adapted from [1].

We implemented our algorithms in C++, using SCIP 3.1 as framework and CPLEX 12.6.2 to solve LP subproblems. Our tests ran on a PC with a 2.1GHz CPU and 8GB of RAM. We compared to the branch-and-cut algorithm of CPLEX, with default parameter settings, exploiting a compact formulation of the VSCBPP adapted from [2], and stopping the computation at the root node. We considered a dataset adapted from the literature [1]. The dataset includes instances with either Tight (T) or Loose (L) capacities, and items whose size is either Small (S), Medium (M) or Large (L). Preliminary results on 6 classes of 10 instances each with $|I| = 20$ are reported in Table 1. Capacity and size distribution are indicated in the first two colums. The Table includes two blocks, one for CPLEX and one for our Column Generation algorithm (CG). Each block reports the number of instances whose optimality was proved (S) the average optimality gap obtained (G) and the time required to complete the computation (T). CPLEX turned out to be faster, but CG results were more accurate, requiring reasonable additional CPU time. In particular, CG was able to directly solve many more instances to proven optimality.

References

[1] M. Casazza, A. Ceselli “Mathematical programming algorithms for bin packing problems with item fragmentation ” Computers and Operations Research 46 (2014)

[2] M. Casazza, A. Ceselli “Exactly solving packing problems with fragmentation ” Optimization Online Tech. Rep. (2015)

[3] B. LeCun, T. Mautor, F. Quessette, M.-A. Weisser “Bin packing with fragmentable items ” Theoretical Computer Science 602 (2015)

Maximin Fairness in Project Budget Allocation

Maurizio Naldi [a,1] Gaia Nicosia [b,2] Andrea Pacifici [a,3]
Ulrich Pferschy [c,4]

[a] *Dipartimento di Ingegneria Civile e Ingegneria Informatica,
Università di Roma Tor Vergata, Roma, Italy*

[b] *Dipartimento di Ingegneria, Università Roma Tre, Roma, Italy*

[c] *Department of Statistics and Operations Research,
University of Graz, Graz, Austria*

Abstract

This work addresses a multi agent allocation problem in which multiple departments compete for shares of a company budget. Each department has its own portfolio of projects with given expected profits and costs and selects an optimal subset of its projects consuming its assigned budget share. Besides considering the total profit of the company a central decision maker should also take fairness issues into account. Thus, we introduce an equity criterion based on maximin fairness. The resulting trade-off between total profit and fairness indicators is studied in this contribution. To this purpose a bicriteria ILP model is presented where one of the objectives is the maximization of the overall profit and the other is the maximization of the minimum budget allocated to one of the departments. We perform an experimental analysis showing a nearly perfect linear anticorrelation between profit and fairness index values.

Keywords: Knapsack problem, project management, multi agent system, fair allocation, decision support system.

1 Introduction

We consider an allocation problem faced by the general manager of a company with a set D of d departments. A total budget b has be allocated to several projects, each belonging to one of the departments. We indicate by J_i the set of projects of department $i \in D$ and w.l.o.g. assume that $|J_i| = n$, for $i \in D$. The j-th project of the i-th department requires a budget s_{ij}, which must be obtained in full for the project to be undertaken, and yields an estimated return on investment (ROI) r_{ij} $(i = 1 \ldots d,\ j = 1 \ldots n)$. Since the overall required budget $\sum_{i \in D,\ j \in J_i} s_{ij}$ is usually larger than the available budget b, the company must select a subset of the projects submitted for funding.

Naturally, the company's objective is the maximization of the total profit obtained from the budget investment. However, this may correspond to selecting projects in a way that can be perceived as unfair by one or more departments, since the budget may well be allocated in an unbalanced way. To avoid such potentially biased solutions the company should take into account some idea of equity or *fairness* in the allocation decision.

In economic analysis, an axiomatic characterization of what might be a fair resource allocation has been the subject of several studies in the last decades. In the context of optimization, even though some studies date back to the Nineties ([4]), only recently "fairness" concepts received considerable attention (see e.g. [1], [6] and [7]). Here, we formulate the allocation decision as a bicriteria problem in which both profit and fairness maximization are considered. In this paper we follow the idea of Rawlsian justice and adopt a maximin fairness approach, i.e., we aim at maximizing, together with the overall profit, the minimum value of the budget allocated to a department. A similar problem has been addressed in [3], where the author uses *range* as a fairness indicator, i.e. the maximum difference between the budget allocated to any two departments. The author presents an ILP model and a two-phase algorithm for determining Pareto optimal solutions. In the future we will consider also other fairness measures, e.g. based on HHI or the Gini-coefficient.

Note that our problem is also strongly related to variants of the binary knapsack problems with multiple agents as, for instance, the knapsack sharing problem [2].

[1] Email: naldi@disp.uniroma2.it

[2] Email: nicosia@dia.uniroma3.it

[3] Email: andrea.pacifici@uniroma2.it (Corresponding author.)

[4] Email: pferschy@uni-graz.at Supported by the Austrian Science Fund (FWF): P 23829-N13.

2 Bicriteria model

We use a set of decision variables as in the standard 0-1 knapsack model: Variable $x_{ij} \in \{0,1\}$ equals 1 iff the j-th project of department i is accepted. Then the budget $b_i(x)$ allocated to the i-th department and the overall expected profit $\pi(x)$ for the company may be expressed respectively as $b_i(x) = \sum_{j \in J_i} s_{ij} x_{ij}$ and $\pi(x) = \sum_{i \in D} \sum_{j \in J_i} r_{ij} s_{ij} x_{ij}$. The total investment by the company is obviously limited through a knapsack constraint by the available budget b. We indicate by $\mathcal{X}$ the set of feasible allocations, i.e. $\mathcal{X} = \{x \in \{0,1\}^{d \times n} \ : \ \sum_{i \in D} b_i(x) \leq b\}$. Under any allocation $x \in \mathcal{X}$, the maximin fairness index is then expressed by $F(x) = \min_{i \in D} b_i(x)$ and the obvious resulting bicriteria knapsack model for the overall profit and fairness maximization can be written as follows:

$$\max \{(\pi(x), F(x)) \ : \ x \in \mathcal{X}\} \tag{1}$$

In this work we study $P(\lambda)$ as a surrogate model of (1) for any fairness bound λ:

$$P(\lambda): \quad \max\{\pi(x) \ : \ F(x) \geq \lambda, \ x \in \mathcal{X}\}. \tag{2}$$

3 Computational Experiments

We performed a large number of computational tests using Gurobi as ILP solver on randomly generated instances: individual project budgets are drawn from a lognormal distribution and the corresponding ROI values from a uniform distribution (whose lower bound reflects the assumption that only projects with a positive estimated return larger than the given interest rate are considered). The lognormal model has been validated in [5] for datasets spanning ten years of data. In particular, the following values have been set for the parameters of the lognormal model: $\mu = 5.2$ and $\sigma = 1.35$. These values are quite central in the range observed in the analyzed datasets and give a mean project size of 451 M\$. We generated 5 different classes of 1000 instances each. The instances in each class share the same budget value b, the number of departments $d \in \{2, 5, 10\}$ and the number of projects for each department $n \in \{10, 50, 100\}$.

We performed a statistical analysis on the test instances set to measure the distribution of fairness, normalized by the maximum fairness that can be achieved, i.e., the fairness that would result if all the departments got the same share of the budget ($\max F = b/d$). The resulting empirical probability density function (obtained through a Gaussian kernel approach) gets more and more slanted towards 1 and exhibits a diminishing dispersion as λ grows.

Most importantly, we investigate how the value of λ in (2) impacts the average profit and fairness in the solution of $P(\lambda)$. Since the optimization procedure aims at maximizing the profit, and the quest for fairness is considered as a constraint, we expect fairness to be achieved at the expense of profit, the more so the higher the λ-threshold is set. Such a behavior is convincingly verified by the data for all instance classes. In fact, the correlation between fairness and profit turns out to be a startling -0.9975, i.e., profit and fairness exhibit a nearly perfect linear anticorrelation. However, taking a different point of view, it can be shown that the relationship of profit and fairness to λ is not linear. Indeed, if we progressively raise the fairness bound λ, the average fairness grows more rapidly than its minimum guaranteed value λ especially as λ is increased over 50% of the budget value b. Moreover, we can observe that the simultaneous decrease in profits is also non linear.

At the time being, additional experiments are in progress aiming at better characterizing the trade-off between profit and fairness criteria.

References

[1] D. Bertsimas, V.F. Farias, N. Trichakis, The price of fairness, *Operations Research* 59, 17–31, 2011.

[2] M. Fujimoto, T. Yamada, An exact algorithm for the knapsack sharing problem with common items, *European J. of Operational Research* 171, 693–707, 2006.

[3] G. Kozanidis, Solving the linear multiple choice knapsack problem with two objectives: profit and equity, *Computational Optimization and Applications* 43, 261–294, 2009.

[4] M. Mandell, Modeling effectiveness-equity tradeoffs in public service delivery systems, *Management Science* 37, 467–482, 1991.

[5] M. Naldi, A Probability Model for the Size of Investment Projects, in: UKSim-AMSS 9th IEEE European Modelling Symposium on Mathematical Modelling and Computer Simulation, Madrid, 169–173, 2015.

[6] G. Nicosia, A. Pacifici, U. Pferschy, Competitive subset selection with two agents, *Discrete Applied Mathematics* 159, 1865–1877, 2011.

[7] G. Nicosia, A. Pacifici, U. Pferschy, Price of Fairness for Allocating Bounded Resources, arXiv Preprint Series, no. 1508.05253, 2015.

Heuristics for the General Multiple Non-linear Knapsack Problem

Luca Mencarelli [a,1,2], Claudia D'Ambrosio [a,3], Angelo Di Zio [b,4] and Silvano Martello [c,5]

[a] *CNRS, UMR 7161, LIX*
École Polytechnique
Palaiseau, France

[b] *Ferrari S.p.A.*
Maranello, Italy

[c] *DEI "Guglielmo Marconi"*
Bologna, Italy

Abstract

We propose heuristic algorithms for the multiple non-linear knapsack problem with separable non-convex profit and weight functions. First, we design a fast constructive algorithm that provides good initial solutions. Secondly, we improve the quality of these solutions through local search procedures. We compare the proposed methods with exact and heuristic algorithms for mixed integer non-linear programming problems, proving that our approach provides good-quality solutions in smaller CPU time.

Keywords: Multiple non-linear knapsack problem, Heuristic algorithm, Local search, Mixed-integer non-linear programming.

[1] The first author acknowledges the financial support provided by "MINO" Initial Training Network (ITN) under the Marie Curie 7th European Framework Programme.

[2] Email: `mencarelli@lix.polytechnique.fr`

[3] Email: `dambrosio@lix.polytechnique.fr`

[4] Email: `angelo.dizio@ferrari.com`

[5] Email: `silvano.martello@unibo.it`

1 Introduction

In the multiple non-linear knapsack problem, we are given n items and m knapsacks. We aim at deciding how many units of item j to load in knapsack i, i.e., our decision variables are represented by $x_{ij} \geq 0$ for each $i = 1, \ldots, m$, $j = 1, \ldots, m$. The units of some items are indivisible, thus integrality requirements on the corresponding x_{ij} $(i = 1, \ldots, m)$ have to be satisfied. For each item j, we have

- an upper bound on the item availability $u_j > 0$;
- a profit function $f_j(x) : \mathbb{R}_+ \rightarrow \mathbb{R}_+$;
- a weight function $g_j(x) : \mathbb{R}_+ \rightarrow \mathbb{R}_+$.

We assume that $f(x)$ and $g(x)$ are twice continuously differentiable, separable, non-linear, non-negative, non-decreasing functions. Note that there is no further assumption, thus, in general, f and g can be non-convex and non-concave.

The *Multiple Non-Linear Knapsack Problem* (MNLKP) can then be written as:

$$\begin{aligned}
\max \quad & \sum_{i \in M} \sum_{j \in N} f_j(x_{ij}) & & (1)\\
\text{s.t.} \quad & \sum_{j \in N} g_j(x_{ij}) \leq c_i & i \in M & \quad (2)\\
& \sum_{i \in M} x_{ij} \leq u_j & j \in N & \quad (3)\\
& x_{ij} \geq 0 & i \in M,\ j \in N & \quad (4)\\
& x_{ij} \text{ integer} & i \in M,\ j \in \overline{N} \subseteq N & \quad (5)
\end{aligned}$$

where $M = \{1, \ldots, m\}$ and $N = \{1, \ldots, n\}$. Objective function (1) aims at maximizing the profit given by the total amount of items inserted in the knapsacks. Constraints (2) impose that the knapsack maximum capacities are respected. The limit on the maximum availability of each item is represented by constraints (3). Constraints (5) ensure that, for the indivisible items, a discrete quantity is selected.

To the best of our knowledge, no author studied such variant of the non-linear knapsack before. For an extended reference on the classical 0-1 multiple knapsack the reader is referred to [6,5]. For a study on the single non-linear knapsack problem, we refer the reader to [3].

2 Constructive heuristic

The constructive heuristic is based on the discretization of the solution space.

Let $\delta_j = u_j/s$ for $j \in N$ (or $\delta_j = \max(1, \lfloor u_j/s \rfloor)$ if $j \in \overline{N}$) be the sampling step size, where s is the corresponding number of samplings.

We define the *profit-to-weight ratios*:

$$r_{jk} = \frac{f_j(k\delta_j)}{g_j(k\delta_j)} \qquad (j \in N, k = 1, \dots, s).$$

For every item j, let $\mu_j = \arg\max_{k=1,\dots,s}\{r_{jk}\}$. We assume, without loss of generality, that the items are sorted in non-decreasing order of their profit-to-weight ratio, i.e., so that $r_{1\mu_1} \geq r_{2\mu_2} \geq \cdots \geq r_{n\mu_n}$, and that the knapsacks are sorted in non-decreasing order of their capacities, i.e., so that $c_1 \geq c_2 \geq \cdots \geq c_m$.

Initially, the first two items, i.e., the two items with best largest profit-to-weight ratios, are considered. The algorithm selects $\bar{\mu}_1\delta_1$ units to the first item, where $\bar{\mu}_1$ is the largest sampling point, at which the ratio of first item is *higher* than the one of the second item. The capacity available for the first item is correspondly decreased. The algorithm updates the value of the profit-to-weight ratios and eventually re-sorts the items. The process considers then the next couple of items. The time complexity of this procedure is $O(n^2)$.

By iterating for all the knapsacks, we obtain a heuristic for the MNLKP of time complexity $O(mn^2)$.

3 Local Search

Given a knapsack i, we choose two items j and k and consider two potential modification of the current solution:

(i) $\Delta^1 = (f_j(x_{ij} + \varepsilon) - f_j(x_{ij})) + (f_k(x_{ik} - \varepsilon) - f_k(x_{ik}))$;

(ii) $\Delta^2 = (f_j(x_{ij} - \varepsilon) - f_j(x_{ij})) + (f_k(x_{ik} + \varepsilon) - f_k(x_{ik}))$,

where ε is a sufficiently *small* value, i.e., $\varepsilon < \min(\delta_j, \delta_k)$.

The local search selects the *best* modification, i.e., the variation Δ that leads to a *greater* objective value: $\Delta = \max(\Delta^1, \Delta^2)$. If a variation is infeasible with respect to (2)-(4), the corresponding Δ^i is set to 0. If $\Delta > 0$ the modification is performed and an improved solution is obtained. The process is iterated for all the knapsacks. The overall time complexity of the local search is $O(mn^2)$.

4 Computational Results

Extensive computational experiments show that the proposed heuristic approach produces a quick solution, outperforming the exact and heuristic solvers for non-convex (mixed-integer) non-linear programs, namely `Couenne` [2] and `Bonmin` [1] or `Ipopt` [4]. The experiments consider the cases in which the profits are described by general non-convex non-concave functions and the weights are either concave or linear.

We randomly generated 3360 instances similarly to [3] with the number of items between 10 and 1000 and the number of knapsacks between 2 and 10. However, $\forall i = 1, \ldots, m-1$ capacity c_i of the i-th knapsack is uniformly generated in either:

$$\left[0.4 \sum_{j=1}^{n} \frac{g_j(u_j)}{m}, 0.6 \sum_{j=1}^{n} \frac{g_j(u_j)}{m}\right] \text{ or } \left[0, \left(0.5 \sum_{j=1}^{n} g_j(u_j) - \sum_{k=1}^{i-1} c_k\right)\right]$$

while the m-th capacity is always set to $c_m = 0.5 \sum_{j=1}^{n} g_j(u_j) - \sum_{i=1}^{m-1} c_i$.

Preliminary computational results clearly show that the proposed methods outperform all the solvers in finding a good feasible solution in short CPU time.

References

[1] Bonmin. URL https://projects.coin-or.org/Bonmin, 02/2016.

[2] Couenne. URL https://projects.coin-or.org/Couenne, 02/2016.

[3] D'Ambrosio, C., and Martello, S., *Heuristic Algorithms for the General Nonlinear Separable Knapsack Problem*. Computers & Operations Research, **38** (2011), 505–513.

[4] Ipopt. URL https://projects.coin-or.org/Ipopt, 02/2016.

[5] Kellerer, H., and Pferschy, U., and Pisinger, D., "Knapsack Problems", Springer, Berlin, Germany, 2004.

[6] Martello, S., and Toth, P., "Knapsack Problems: Algorithms and Computer Implementations", John Wiley & Sons, Chichester, New York, 1990.

Dual approaches for a specific class of integer nonlinear programming problems

Marianna De Santis [1]

Institut für Mathematik
Alpen-Adria Universität Klagenfurt
Universitätsstrasse 65-67, 9020 Klagenfurt am Wörthersee, Austria

Abstract

In this work, we propose a strategy for computing valid lower bounds for a specific class of integer nonlinear programming problems, that includes integer quadratic programming problems. This strategy is used within a branch-and-bound scheme. Experimental results for randomly generated instances show that, in the quadratic case, the devised branch-and-bound method compares favorably to the MIQP solver of `CPLEX 12.6` when the number of constraints is small.

Keywords: integer programming, quadratic programming, global optimization

We consider integer optimization problems of the following form:

$$\begin{aligned} \min\ & f(x) = (x^\top Q x)^p + L^\top x \\ \text{s.t.}\ & Ax \le b \\ & x_i \in \mathbb{Z},\ i = 1, \dots, n \end{aligned} \tag{1}$$

where $Q \in \mathbb{R}^{n\times n}$ is a positive definite matrix, $L \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$ and $0.5 < p \le 1$.

The motivation for studying this class of problems is twofold. From a practical point of view, Problem (1) includes problems that arise in applications, such as portfolio optimization problems (see e.g. [1]). From a theoretical point of view, defining effective algorithms to solve to global optimality Problem (1) represents a big challenge in itself.

[1] Email: `marianna.desantis@aau.at`

In this work, following what has been done in recent papers by Buchheim et al. (see e.g. [3], [4]), we propose a strategy for computing valid lower bounds of Problem (1), with the idea of using this strategy within a branch-and-bound scheme for MINLP problems.

The branch-and-bound scheme we consider enumerates nodes very quickly: by fixing the branching order in advance, we gain the advantage of shifting expensive computations into a preprocessing phase. In each node, the dual problem of the continuous relaxation is solved in order to determine a local lower bound. Since all constraints of the continuous relaxation of (1) are affine, strong duality holds if the primal problem is feasible.

More precisely, assume that the variables with indices in $I \subseteq \{0, \dots, n\}$ have been fixed to values $s = (s_i)_{i \in I}$. Then, Problem (1) reduces to the minimization of

$$f_s : \mathbb{Z}^{n-|I|} \to \mathbb{R},\ x \mapsto (x^\top Q_s x + c_s^\top x + d_s)^p + L_s x + e_s \tag{2}$$

over the feasible region $\mathcal{F}_s = \{x \in \mathbb{Z}^{n-|I|} \mid A_s x \leq b_s\}$, where the matrix Q_s is obtained by deleting the rows and columns corresponding to I, the matrix A_s is obtained by deleting the columns corresponding to I, and the remaining terms are updated appropriately.

Let $\mathscr{L}_s(x, \lambda) : \mathbb{R}^{n-|I|} \times \mathbb{R}^m \to \mathbb{R}$ be the Lagrangian function associated to the continuous relaxation at a generic node. In Section 1 we show how to compute, for fixed λ, the unconstrained minimizer of the Lagrangian function, so that the dual problem we end up with is a continuous problem with non-negativity constraints:

$$\begin{aligned} &\max\ \mathscr{L}_s(x^\star(\lambda), \lambda) \\ &\text{s.t.} \quad \lambda \geq 0; \lambda \in \mathbb{R}^m, \end{aligned} \tag{3}$$

where $x^\star(\lambda) = arg \min_{x \in \mathbb{R}^{n-|I|}} \mathscr{L}_s(x, \lambda)$.

Problem (3) is then solved by the feasible active set method for box constrained problems proposed in [2]. Since we are considering the dual problem, it suffices to find an approximate solution, as each dual feasible solution yields a valid lower bound. We can thus prune the branch-and-bound node as soon as the current upper bound is exceeded by the value of any feasible iterate produced in a solution algorithm for the dual problem.

Experimental results for randomly generated instances show that, in the quadratic case (i.e. $p = 1$), the devised branch-and-bound method compares favorably to the MIQP solver of `CPLEX 12.6` when the number of constraints is small.

1 Unconstrained minimization of the Lagrangian

Let us consider a generic node of the branch-and-bound tree where the variables with indices in $I \subseteq \{0, \dots, n\}$ have been fixed. The continuous relaxation we end up with is the following continuous convex problem:

$$\begin{aligned} \min \quad & (x^\top Q_s x + c_s^\top x + d_s)^p + L_s^\top x + e_s \\ \text{s.t.} \quad & A_s x \leq b_s \ , x \in \mathbb{R}^{n-|I|} \ . \end{aligned} \tag{4}$$

Let $x_0 = -\frac{1}{2} Q_s^{-1} c_s$ and $\bar{d}_s = d_s - \frac{1}{4} c_s^\top Q_s^{-1} c_s$. Note that $\bar{d}_s \geq 0$ by construction. The Lagrangian function of (4) and its gradient can be written as

$$\mathscr{L}(x, \lambda) = \big((x - x_0)^\top Q_s (x - x_0) + \bar{d}_s\big)^p + L_s^\top x + e_s + \lambda^\top (A_s x - b_s);$$

$$\nabla_x \mathscr{L}(x, \lambda) = 2p\, Q_s (x - x_0) \left((x - x_0)^\top Q_s (x - x_0) + \bar{d}_s\right)^{p-1} + L_s + A_s^\top \lambda,$$

respectively, where $\lambda \in \mathbb{R}^m$.

After applying the bijective transformation $z = Q_s^{\frac{1}{2}}(x - x_0)$, we get

$$\nabla_z \mathscr{L}(z, \lambda) = 2p\, Q_s^{\frac{1}{2}} z (z^\top z + \bar{d}_s)^{p-1} + L_s + A_s^\top \lambda.$$

Therefore, the unconstrained minimizer of $\mathscr{L}(z, \lambda)$ satisfies the following equation

$$z(z^\top z + \bar{d}_s)^{p-1} = r, \tag{5}$$

where

$$r = \frac{Q_s^{-\frac{1}{2}}(-L_s - A_s^\top \lambda)}{2p}.$$

Equation (5) can be rewritten as

$$z = r(\|z\|^2 + \bar{d}_s)^{1-p}. \tag{6}$$

Let $y \in \mathbb{R}_+, \ y = \|z\|$. Taking the norm on both sides of (5) we get

$$y(y^2 + \bar{d}_s)^{p-1} = \|r\|. \tag{7}$$

Let $f : \mathbb{R}_+ \to \mathbb{R}_+$ be the continuous, unbounded and strictly monotonically increasing function $f(y) = y(y^2 + \bar{d}_s)^{p-1}$. It can be easily noticed that $f(0) = 0$, so that $f(y) = \|r\|$ has a unique solution $y \geq 0$.

We distinguish two cases:

Case 1: $\bar{d}_s = 0$, that is $x_i = 0$, for all $i \in I$. In this case equation (7) becomes

$$y^{2p-1} = \|r\|$$

and by (6), we get

$$z = r\|r\|^{\frac{2(1-p)}{2p-1}}. \tag{8}$$

Case 2: $\bar{d}_s > 0$, that is $\exists i \in I$ such that $x_i \neq 0$. In this case, the Newton method can be applied to solve $f(y) = \|r\|$. Let $\hat{y} > 0$ be the only positive value such that $f(\hat{y}) = \|r\|$. Then, the unconstrained minimizer with respect to z of $\mathscr{L}(z, \lambda)$ is

$$z = r(\hat{y}^2 + \bar{d}_s)^{1-p}. \tag{9}$$

We are then able to state the following result:

Proposition 1.1 *The unique solution of* $\nabla_x \mathscr{L}(x, \lambda) = 0$ *is*

$$x^\star(\lambda) = -\frac{\beta}{2p} Q_s^{-1}(L_s + A_s^\top \lambda) + x_0, \tag{10}$$

where $\beta := (\hat{y}^2 + \bar{d}_s)^{1-p}$ *and* $\hat{y} \in \mathbb{R}_+$ *is the solution of equation* (7).

Proof. By (8) and (9) and recalling that $x = Q_s^{-\frac{1}{2}} z + x_0$ we get the expression (10). □

References

[1] Cornuéjols, G. and Tütüncü, R. Optimization methods in Finance. Mathematics, Finance, and risk. Cambridge University Press, Cambridge, U.K., New York, 2006

[2] De Santis, M., Di Pillo, G. and Lucidi, S., *An active set feasible method for large-scale minimization problems with bound constraints*, Comput. Opt. Appl., **53(2)** (2012) 395–42.

[3] Buchheim, C., De Santis, M., Palagi, L. and Piacentini, M., *An Exact Algorithm for Nonconvex Quadratic Integer Minimization using Ellipsoidal Relaxations*, SIAM J. Optim. **23(3)** (2013), 1867–1889.

[4] Buchheim, C., De Santis, M., Lucidi, S., Rinaldi, F. and Trieu, L., *A Feasible Active Set Method with Reoptimization for Convex Quadratic Mixed-Integer Programming*, Optimization online, 2014.

A generalized Gilmore-Lawler procedure for the Quadratic Assignment Problem

Borzou Rostami [a,1,2] Federico Malucelli [b]

[a] *Fakultät für Mathematik, TU Dortmund, Germany*

[b] *Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Italy*

Abstract

In this paper we propose a new lower bounding procedure for the Quadratic Assignment Problem based on a generalization of the well-known Gilomore-Lawler procedure for a higher order reformulation. Computational results on some benchmark instances show the strength of the new approach compared with other lower bounds.

Keywords: Quadratic Assignment Problem, Lower bound, Gilomore-Lawler procedure

1 Introduction

The Quadratic Assignment Problem (QAP) is among the most difficult NP-hard combinatorial optimization problems. The general mathematical formu-

[1] The first author has been supported by the German Research Foundation (DFG) under grant BU 2313/2.

[2] Email: brostami@mathematik.tu-dortmund.de

lation of the problem is as follows:

$$\text{QAP: } \min\Big\{\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} q_{ijkl}x_{ij}x_{kl}: \ x \in X, \ x \text{ binary}\Big\}$$

where

$$X = \{x \geq 0 : \textstyle\sum_j x_{ij} = 1 \quad \forall i; \sum_i x_{ij} = 1 \quad \forall j\}.$$

Many solution methods, exact or heuristic algorithms have been proposed for solving the QAP. Because of the quadratic structure of the problem many attempts have been made in the literature to reformulate the problem as a quadratic 0-1 programming problem, a global concave minimization problem, or an Mixed Integer Programming (MIP).

In this paper, we are concerned with lower bounds for the QAP. Lower bounds play an important role in success of the Branch-and-Bound type algorithms for the QAPs. The ideal lower bound should be sharp (i.e., yielding a "small" gap with respect to the optimum solution) and fast enough to compute. The Gilmore-Lawler procedure presented by Gilmore [2] and Lawler [4] is one of the best known lower bounds for QAP given by the solution of the following linear assignment problem (LAP):

$$GLB = \min \ \Big\{\textstyle\sum_i \sum_j l_{ij}x_{ij}: \ x \in X, \ x \text{ binary}\Big\},$$

where for each i, j the coefficient l_{ij} are found by solving the following problem:

$$l_{ij} = \min \ \Big\{\textstyle\sum_k \sum_l q_{ijkl}x_{kl}: \ x \in X, \ x_{ij} = 1\Big\}.$$

Although the GL provides a lower bound for the QAP that can be computed very efficiently, the obtained bounds are not so close to the optimal solution and usually deteriorate as the size of the QAP increases.

2 A generalization of the Gilmore-Lawler procedure

Given a general mathematical formulation of QAP, we consider a generalization of the GL, called GGL, procedure which considers the minimum interaction cost not only of a single assignment but of two special assignments. To explain the idea, let us first define a reformulation of the QAP as follows:

$$\text{CAP: } \min\Big\{\textstyle\sum_{(i,j,k)\in A}\sum_{l,m} E_{ijk}^{lm}x_{ij}x_{jk}x_{lm}: \ x \in X, \ x \text{ binary}\Big\} \qquad (1)$$

where $A = \{(i,j,k):\ i=j=k \quad \text{or} \quad i \neq j, j \neq k\}$, and for each $(i,j,k) \in A$, and each l,m, $E_{ijk}^{lm} = 1/2\,(q_{ijlm} + q_{jklm})$.

Theorem 2.1 *Problems CAP and QAP are equivalent.* □

In order to compute a lower bound for the QAP, we apply a GL type procedure to CAP. More precisely, for each $(i,j,k) \in A$ potentially in the solution we consider a subproblem to solve an assignment problem which contains assignment of i to j and assignment of j to k, i.e.,

$$u_{ijk} = \min\left\{\sum_{l,m} E_{ijk}^{lm} x_{lm} : x \in X,\ x_{ij} = x_{jk} = 1\right\} \quad \forall (i,j,k) \in A$$

The new bound is then defined to be the solution of the following problem:

$$\mathrm{P}:\ GGLB = \min\left\{\sum_{(i,j,k)\in A} u_{ijk} x_{ij} x_{jk}:\ x \in X,\ x \text{ binary}\right\}.$$

To linearize this problem, we first multiply equations $\sum_j x_{ij} = 1$, for each i, and equations $\sum_j x_{ij} = 1$, for each j by variables x_{ki} and x_{jk}, respectively. All such quadratic equations are included within the formulation. Then we linearize the nonlinear terms $x_{ij}x_{jk}$ by substituting $y_{ijk} = x_{ij}x_{jk}$ for all $(i,j,k) \in A$. Following [5] we further simplify the linearized model and then project out the x variables to result the following Integer Programming formulation:

$$\begin{array}{lll} \mathrm{IP1}: & \min & \sum_{(i,j,k)\in A} (u_{ijk} + q_{ijjk}) y_{ijk} & \\ & \text{s.t.} & \sum_{i,k:(i,j,k)\in A} y_{ijk} = 1 & \forall\, j \\ & & \sum_{k:(k,i,j)\in A} y_{kij} - \sum_{k:(i,j,k)\in A} y_{ijk} = 0 & \forall (i,j) \\ & & y \text{ binary.} & \forall\ (i,j,k) \in A. \end{array}$$

Theorem 2.2 *Problems IP1 and P are equivalent.* □

Note that this problem has network properties and its optimal solution will yield the GGL bound. However, if the binary restrictions on variables y are relaxed in IP1, the problem provides a lower bound on the GGLB value.

3 Computational results

In this section we evaluate the quality of lower bounds in terms of relative gap in percent obtained by the GGL procedure and compare it with the GL. To show the effectiveness of the GGL, we also report the relative gap in percent obtained by the well-known RLT based approach [3]. We should note here

Table 1
Results for some chalanging instances from the QAPLIB

Instance		Gap(%)			Time (second)		
name	Opt.	GL	GGL	RLT1	GL	GGL	RLT1
bur26g	10117172	7.4	0.6	0.6	0.0	5.9	138.0
bur26h	7098658	7.8	0.7	0.7	0.0	5.8	129.8
Had20	6922	10.9	5.3	5.1	0.0	0.4	36.1
Rou20	725520	17.3	12.1	11.5	0.0	1.1	36.4
Nug20	2057	19.9	15.8	15.2	0.0	1.3	36.2
Nug30	6124	25.8	22.3	21.7	0.0	5.7	225.9

that, in our computation of GGLB we found it more efficient to apply the reformulation (1) to the quadratic 0-1 reformulation of the QAP proposed in [1]. Table 1 reports the results. The first two columns give the instance names and the optimal objective values (opt). The next three columns give the gap in percent obtained by the GL, GGL, and RLT1, respectively. The last three columns give the computing times. As you can observe, the bounds provided by the GGL outperform the GL bounds and very close to those of the RLT1. Note that our results have been obtained in small computational times; less than 6 seconds for all instances, while the RLT1 take much longer time to compute the bounds.

References

[1] Carraresi, P. and F. Malucelli, *A reformulation scheme and new lower bounds for the qap*, Quadratic assignment and related problems (1994), pp. 147–160.

[2] Gilmore, P. C., *Optimal and suboptimal algorithms for the quadratic assignment problem*, Journal of the Society for Industrial & Applied Mathematics **10** (1962), pp. 305–313.

[3] Hahn, P. and T. Grant, *Lower bounds for the quadratic assignment problem based upon a dual formulation*, Operations Research **46** (1998), pp. 912–922.

[4] Lawler, E. L., *The quadratic assignment problem*, Management science **9** (1963), pp. 586–599.

[5] Rostami, B. and F. Malucelli, *A revised reformulation-linearization technique for the quadratic assignment problem*, Discrete Optimization **14** (2014), pp. 97–103.

The Linear Extension Polytope of a Poset

Jean-Paul Doignon [a,1], Samuel Fiorini [a,2], Selim Rexhep [a,3]

[a] *Department of Mathematics, Université Libre de Bruxelles, Brussels, Belgium*

Abstract

Let P be a finite poset. By definition, the linear extension polytope of P has as vertices the characteristic vectors of all linear extensions of P. In case P is an antichain, it is the linear ordering polytope. The linear extension polytope appears in combinatorial optimization in the context of scheduling with precedence constraints, see e.g. [5]. It seems also relevant to order theory, being similar in spirit to other constructions such as the linear extension graph, see e.g. [4]. In this work, we relate the combinatorial properties of the poset P to the polyhedral structure of its linear extension polytope. Of particular interest is a natural relaxation of the linear extension polytope. We prove that the relaxation is exact in case P is a width-2 poset, and formulate a conjecture stating exactly when the relaxation is exact.

Keywords: poset, linear extension, polytope, linear ordering problem

1 Introduction

Consider a (finite, strict) poset $P = (X, <)$. A chain $L = (X, \prec)$ is said to be a *linear extension* of P if $i < j$ implies $i \prec j$ for all $i, j \in X$. The *linear*

[1] Email: doignon@ulb.ac.be

[2] Email: sfiorini@ulb.ac.be

[3] Email: srexhep@ulb.ac.be

extension polytope of P is defined as the convex hull of the characteristic vectors of the linear extensions of P. Letting $A := \{(i,j) \mid i,j \in X,\ i \neq j\}$ and letting χ^L denote the characteristic vector of the chain $L = (X, \prec)$, that is, the vector $\chi^L \in \mathbb{R}^A$ such that $\chi^L_{ij} = 1$ if $i \prec j$ and $\chi^L_{ij} = 0$ otherwise, we have $\mathrm{P}_{\mathrm{LO}}(P) := \operatorname{conv}\bigl(\{\chi^L \mid L \text{ is a linear extension of } P\}\bigr)$.

If P is antichain, then the linear extension polytope $\mathrm{P}_{\mathrm{LO}}(P)$ is the famous *linear ordering polytope* $\mathrm{P}^X_{\mathrm{LO}}$, which naturally appears in polyhedral approaches to the linear ordering problem, see e.g. [3]. For a general poset $P = (X, <)$, the linear extension polytope $\mathrm{P}_{\mathrm{LO}}(P)$ is the face of the linear ordering polytope $\mathrm{P}^X_{\mathrm{LO}}$ obtained by setting $x_{ij} = 1$ whenever $i < j$.

To our knowledge, Schulz in his PhD thesis [5] was the first to study the linear extension polytope. He proved that the affine hull of $\mathrm{P}_{\mathrm{LO}}(P)$ is defined by

$$x_{ij} = 1, \quad \text{whenever } i < j, \tag{1}$$

$$x_{ij} = 0, \quad \text{whenever } i > j, \tag{2}$$

$$x_{ij} + x_{ji} = 1, \quad \text{whenever } i \text{ and } j \text{ are incomparable.} \tag{3}$$

As a consequence, the dimension of the linear extension polytope $\mathrm{P}_{\mathrm{LO}}(P)$ equals the number of unordered pairs formed of incomparable elements of P.

2 Linear Description

Since the linear ordering problem is NP-hard, obtaining an explicit linear description of $\mathrm{P}_{\mathrm{LO}}(P)$ for a general poset P seems hopeless. However, the situation can change when posets P are restricted. In general, let $\mathrm{Q}_{\mathrm{LO}}(P)$ denote the polytope defined by (1)–(3) together with

$$x_{ij} \geqslant 0, \quad \forall i,j \in X \text{ with } i \neq j, \tag{4}$$

$$x_{ij} + x_{jk} - x_{ik} \leqslant 1, \quad \forall i,j,k \in X \text{ with } i \neq j \neq k \neq i. \tag{5}$$

Inequalities (4)–(5) define facets of the linear ordering polytope $\mathrm{P}^X_{\mathrm{LO}}$. The situation is however more complex for $\mathrm{P}_{\mathrm{LO}}(P)$. An incomparable pair of elements (i,j) of $P = (X,<)$ is *critical* if adding the pair (i,j) to $<$ still yields a partial order relation. For more information on critical pairs, see Trotter [6]. We proved that Inequality (4) defines a facet of $\mathrm{P}_{\mathrm{LO}}(P)$ if and only if the pair (j,i) is critical in P. Determining when Inequality (5) is facet defining for $\mathrm{P}_{\mathrm{LO}}(P)$ seems harder to do.

Note that the polytope $\mathrm{Q}_{\mathrm{LO}}(P)$ is a relaxation of $\mathrm{P}_{\mathrm{LO}}(P)$. It is natural to ask when this relaxation is exact, that is, when $\mathrm{Q}_{\mathrm{LO}}(P) = \mathrm{P}_{\mathrm{LO}}(P)$ holds. The answer is already known when P is an antichain:

Theorem 2.1 (Dridi [1]) *If P is an antichain, $\mathrm{Q_{LO}}(P) = \mathrm{P_{LO}}(P)$ if and only if $n \leqslant 5$.*

To prove Theorem 2.1, Dridi uses the following result. Recall that the *dimension* of the poset $P = (X, <)$ is the minimum number of linear orders of X such that their intersection is precisely $<$.

Theorem 2.2 (Dridi [1]) *Let $P = (X, <)$ be a poset and $Q = (X, \prec)$ be a (nonnecessarily linear) extension of P. Let also $\phi(Q)$ be the $\{0, \frac{1}{2}, 1\}$ point of $\mathbb{R}^A$ defined by $\phi(Q)_{ij} = 1$ if $i \prec j$, $\phi(Q)_{ij} = \frac{1}{2}$ if i and j are incomparable and $\phi(Q)_{ij} = 0$ otherwise. Then $\phi(Q) \in \mathrm{Q_{LO}}(P)$. Moreover, $\phi(Q) \in \mathrm{P_{LO}}(P)$ if and only if $\dim(Q) \leqslant 2$.*

Remark: Dridi [1] worked only on the linear ordering polytope $\mathrm{P}^X_{\mathrm{LO}}$, and Theorem 2.1 and 2.2 are reformulations of his results.

We propose the following conjecture generalizing Theorem 2.1:

Conjecture 2.3 *There holds $\mathrm{Q_{LO}}(P) = \mathrm{P_{LO}}(P)$ if and only if all the extensions of P have dimension at most 2.*

The implication $\Rightarrow$ of the conjecture follows easily from Theorem 2.2. On the other hand, because an antichain of size at least 6 has an extension of dimension 3, a poset without 3-dimensional extensions has width at most 5. The following result implies that one can focus on the cases where P has width 3, 4 or 5 in order to establish Conjecture 2.3:

Theorem 2.4 *If P has width at most 2, then $\mathrm{P_{LO}}(P) = \mathrm{Q_{LO}}(P)$.*

Our next result suggests to consider the structure of the incomparability graph $\overline{G}(P)$ of P when aiming to prove Conjecture 2.3.

Theorem 2.5 *All extensions of the finite poset $P = (X, \leqslant)$ have dimension at most 2 if and only the incomparability graph $\overline{G}(P)$ does not admit the antenna nor the co-rising sun (see Figure 1) as (nonnecessarily induced) subgraphs.*

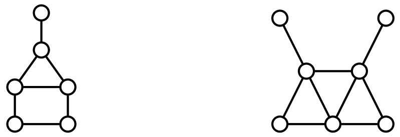


Fig. 1. The antenna and the co-rising sun

In order to prove Conjecture 2.3, we established that it suffices to consider posets P whose incomparability graph $\overline{G}(P)$ is connected, even 2-connected with the possible exception of some pendent edges. If $\overline{G}(P)$ admits a clique of size 5, then because of the absence of antenna the clique must be a connected component of $\overline{G}(P)$; hence the conjecture is true by Theorem 2.1 and the fact that we assume $\overline{G}(P)$ connected. Using a similar argument, we are able to handle the case where $\overline{G}(P)$ admits a clique of size 4. Hence, to prove Conjecture 2.3, there remains to handle only the case of a width-3 poset P.

3 Comparability Invariance

Two posets with the same comparability graphs are called *equivalent*. It is well known that the number of linear extensions of a poset is a *comparability invariant*, that is, if P and Q are equivalent then $e(P) = e(Q)$ (see for example Trotter [6]). Hence, for two equivalent posets, the associated polytopes have both the same number of vertices and the same dimension. Now a natural question is whether the combinatorial structure of $\mathrm{P_{LO}}(P)$ is a comparability invariant. (We recall that two polytopes are combinatorially equivalent if they have isomorphic face lattices). Suprisingly, the answer is no in general. We found a counter-example on only 6 points. However, we have a proof that the linear extension polytope of two equivalent width-2 posets are affinely equivalent.

References

[1] T. Dridi, *Sur les distributions binaires associées à des distributions ordinales*, Mathématiques et Sciences Humaines, 69, 15-31, 1980.

[2] E. Gawrilow, M. Joswig, *Polymake: a framework for analyzing convex polytopes*, Polytopes-combinatorics and computation (Oberwolfach, 1997), 43-73, DMV Sem., 29, Birkhäuser, Basel, 2000.

[3] M. Grötschel, M. Jünger, G. Reinelt, *Facets of the linear ordering polytope*, Mathematical Programming, 33, 43-60, 1984.

[4] M. Massow, *Linear extension graphs and linear extension diameter*, PhD thesis, TU Berlin, 2009.

[5] A. Schulz, *Polytopes and Scheduling*, Phd Thesis, TU Berlin, 1996.

[6] W.T. Trotter, *Combinatorics and partially ordered sets : Dimension theory*, The Johns Hopkins University Press, Baltimore, 1992.

Combinatorial Relaxation Bounds and Preprocessing for Berth Allocation Problems

Evellyn Cavalcante, Johan Oppen, Phillippe Samer [1]

Molde University College – Molde, Norway

Sebastián Urrutia [2]

Universidade Federal de Minas Gerais – Belo Horizonte, Brazil

Abstract

We investigate an optimization problem in container ports, for which previous models based on generalized set partitioning formulations have been studied. We describe two combinatorial relaxations based on computing maximum weighted matchings in suitable graphs, providing dual bounds and a variable reduction technique.

Keywords: Dual bounds, matching, probing, port operations, maritime logistics.

1 Introduction

In this work, we discuss graph-theoretical results for a discrete optimization problem in maritime logistics. The Berth Allocation and Quay Crane Assignment Problem (BACAP) aims to allocate berthing position/time, and a number of quay cranes (QCs) for arriving vessels in a seaport container terminal. Feasible assignments in the BACAP need to fulfil requirements on desired berthing period and position, and an agreement on the QCs availability.

[1] evellyn.cavalcante@gmail.com, johan.oppen@himolde.no, samer@dcc.ufmg.br

[2] surrutia@dcc.ufmg.br

Work supported by Norwegian Research Council project 227084/O70: Port-Ship Coordinated Planning.

Recent work formulate variations of this problem as a Generalized Set Partitioning Problem (GSPP) [1,2], where each column represents a feasible assignment for a vessel, and its cost is a linear combination of deviations from desired berthing and QCs allocation. The algorithms in [1] solve a GSPP model after generating all variables *a priori*. Effective variable reduction techniques are central in their effective results. The work of [2] stems from a similar approach, though they assume different application modeling and instances.

We demonstrate two novel dual bounds and, as it is done in [1], extend them into a preprocessing technique. The results can be exploited in algorithms building on variable enumeration approach [1,2]. Our companion full paper describes computational experiments, and a branch and cut algorithm separating valid inequalities from set partitioning and packing relaxations.

2 Set partitioning formulations for BACAP

We describe next the GSPP model for the BACAP presented by [1]. Let V be the set of vessels, T be the set of time slots in the horizon, and L be the set of berthing positions in the quay. Define $P = T \times L$, and K as the number of available QCs. Let Ω denote the complete set of feasible assignments; note that $|\Omega| \leq (|V| \times |P| \times K)$ since feasible assignments respect each vessel requirements in a problem instance. Decision variables $y \in \mathbb{B}^{|\Omega|}$ indicate which assignments are used in the solution. The coefficient matrices are as follows. $A \in \mathbb{B}^{|V|\times|\Omega|}$ associates each column j with a single vessel. $B \in \mathbb{B}^{|P|\times|\Omega|}$ represents berthing (time, space) positions: $b_{p,j}$ is one iff position $p \in P$ is used in y_j. An element of $Q \in \mathbb{Z}^{|T|\times|\Omega|}$ determines how many QCs are used by y_j in time period t. Then, the BACAP is defined as follows.

$$\min \sum_{j\in\Omega} c_j y_j \tag{1}$$

subject to (2,3,4,5)

$$\sum_{j\in\Omega} a_{ij} y_j = 1 \ \forall i \in V \tag{2}$$

$$\sum_{j\in\Omega} b_{pj} y_j \leq 1 \ \forall p \in P \tag{3}$$

$$\sum_{j\in\Omega} q_{tj} y_j \leq K \ \forall t \in T \tag{4}$$

$$y_j \in \{0,1\} \ \forall j \in \Omega \tag{5}$$

Set partition constraints (2) ensure that all vessels are served by exactly one assignment, while set packing in (3) forbid overlapping in time/space slots. Inequalities (4) guarantee that QCs availability in the terminal is respected.

3 Weighted matching in two interesting graphs

The GSPP formulations and algorithms we refer to are based on two steps: enumerating feasible assignments for individual vessels *a priori*, and solving the resulting model with a MIP solver. We consider next two suitable graphs, representing the set of assignments enumerated on the first step. In the following, let $\Omega_i \subseteq \Omega$ be the subset of assignments corresponding to a given vessel $i \in V$. We denote two assignments for different vessels as *compatible* if they have no overlap in berthing time and space.

We define the graph $G_1(V, E_1)$, with a vertex for each vessel. The set E_1 includes an edge (i, j) if the individual assignments of best cost for vessels i and j are not compatible with each other. Let c'_j denote the minimum cost assignment for vessel j, *i.e.* $c'_j = \min\{c(y_j) : y_j \in \Omega_j\}$. Analogously, let c''_j be the second minimum cost assignment for j. The cost $c_1(i, j)$ of an edge in G_1 is defined by the least difference among such costs, for the corresponding vessels i and j. That is: $c_1(i, j) = \min\{(c''_i - c'_i), (c''_j - c'_j)\}$. Then, the following bound on the cost of any feasible solution holds.

Theorem 3.1 *Let $M \subseteq E_1$ denote a maximum weighted matching in G_1, and $w(M)$ be its weight. Then $LB_1 \triangleq w(M) + \sum_{j\in V} c'_j$ is a lower bound to (1).*

Proof. The selection of the best individual assignments for each vessel corresponds to relaxing constraints (3) and (4). Therefore, this is a trivial lower bound to the cost of any feasible solution, and amounts to $\sum_{j\in V} c'_j$.

Starting with the trivial selection of best individual assignments, the weight of an edge $(i, j) \in E_1$ corresponds to the minimum cost increase due to exchanging one such assignment for the second best. Clearly, this new pair of assignments for vessels i and j can still be infeasible, but the sum of their costs is a lower bound to the cost of any compatible assignment for these vessels.

Any matching in G_1 correspond to disjoint pairs of vessels, whose best assignments are not compatible. Therefore, the weight of any matching is a required cost increase over $\sum_{j\in V} c'_j$, implied by the pairwise overlap of the corresponding individual assignments. In particular, a maximum weighted matching corresponds to the strongest such bound in G_1. □

Our second dual bound strengthens the information on the cost of compatible assignments between pairs of vessels. Let $G_2(V, E_2)$ denote a complete graph, with a vertex for each vessel. Define the cost $c_2(i, j)$ of an edge in E_2 as the cheapest compatible assignments for vessels i and j, *i.e.* $c_2(i, j) = \min\{c(y_i) + c(y_j) : y_i \in \Omega_i, y_j \in \Omega_j, y_i \text{ and } y_j \text{ are compatible}\}$. Then, we have the following result.

Theorem 3.2 *Let $M \subseteq E_2$ be a maximum weighted matching in G_2. Then, $LB_2 \triangleq \sum_{e\in M} c_2(e)$ is a lower bound to (1).*

Proof. The weight of a single edge $(i, j) \in E_2$ is the sum of the minimum cost assignments for vessels i and j, maintaining their non-overlapping constraints.

A selection of edges not sharing a vertex (*i.e.* a matching) thus corresponds to pairing up vessels and determining their best compatible assignments, which is required in any solution satisfying (3). Therefore, the weight of any matching in G_2 is a lower bound to the cost of a feasible solution, since this clearly relaxes constraints regarding the overlap of unpaired vessels. A maximum weighted matching thus provides the strongest such bound in G_2. □

Although this result holds for any number of vessels, it would be weaker for odd $|V|$. To circumvent this, we simply add to G_2 an artificial vertex s, with edges to every other vertex i, with costs $c_2(s, i) = \min\{c(y_i) : y_i \in \Omega_i\}$.

Due to space limitations, we only state the following two results here, and refer the reader to the companion full paper for their demonstrations.

Theorem 3.3 *The lower bound from graph G_2 is stronger than that from graph G_1,* i.e. *for any given problem instance, $LB_2 \geq LB_1$ holds.*

Finally, assuming that any primal solution bound is known, we can also extend the previous results into a variable probing technique. First, assuming that a given assignment $y_k \in \Omega_k$ is fixed in the solution, we can define the complete graph $G_{2,k}(V \backslash \{k\}, E_{2,k})$. The corresponding edge costs $c_{2,k}$ regard the best compatible assignments for two vessels, which are also compatible with y_k. That is: $c_{2,k}(i,j) = \min\{c(y_i) + c(y_j) : y_i \in \Omega_i, y_j \in \Omega_j, y_i$ and y_j are compatible with each other and with $y_k\}$.

Proposition 3.4 *Let $LB_{2,k}$ denote the lower bound from Theorem (3.2) determined over $G_{2,k}$. Given any upper bound UB to (1), if $c(y_k) + LB_{2,k} > UB$, then there is an optimal solution which does not include the assignment $y_k \in \Omega_k$, and the corresponding variable/column can be removed from the model.*

An analogous method can be derived from (3.1), but (3.3) implies that it cannot be stronger, *i.e.* it cannot remove a column which (3.4) does not.

References

[1] Ç. Iris, D. Pacino, S. Ropke, and A. Larsen. Integrated berth allocation and quay crane assignment problem: Set partitioning models and computational results. *Transportation Research Part E: Logistics and Transportation Review*, 81:75 – 97, 2015.

[2] Y. B. Türkoğulları and Z. C. Taşkın and N. Aras and İ. K. Altınel. Optimal berth allocation and time-invariant quay crane assignment in container terminals. *European Journal of Operational Research*, 235(1):88 – 101, 2014.

A new Mathematical Programming Model for the Green Vehicle Routing Problem

Maurizio Bruglieri[1]

Dipartimento di Design, Politecnico di Milano, Milano, Italy

Simona Mancini[2]

Dipartimento di Matematica e Informatica, Universitá di Cagliari, Cagliari, Italy

Ferdinando Pezzella[3]

DII, Universitá Politecnica delle Marche, Ancona, Italy

Ornella Pisacane[4]

Facoltá di Ingegneria, Universitá degli Studi e-Campus, Novedrate (Como), Italy

Abstract

A new MILP formulation for the Green Vehicle Routing Problem is introduced where the visits to the Alternative Fuel Stations (AFSs) are only implicitly considered. The number of variables is also reduced by pre-computing for each couple of customers an efficient set of AFSs, only given by those that may be actually used in an optimal solution. Numerical experiments on benchmark instances show that our model outperforms the previous ones proposed in the literature.

Keywords: Vehicle Routing, Refueling, Alternative Fuel Vehicles, Efficient set of Alternative Fuel Stations, Mixed Integer Programming

1 Introduction

In the Vehicle Routing Problem (VRP), fuel autonomy is usually assumed sufficient to serve customers in every route. However, with Alternative Fuel Vehicles (AFVs), refuels along the route are needed. Moreover, since Alternative Fuel Stations (AFSs) are not widespread on road networks, refueling stops should be a priori planned to prevent drivers to remain stuck along their routes. The Green VRP (G-VRP) [1] consists in serving a set of customers with a fleet of m AFVs that leave fully refueled from a single depot and can be refueled at AFSs, along their routes. The objective is to minimize the total travel distance. The refueling time is fixed. A fuel consumption rate (r) is given and tanks are totally replenished at AFSs. A maximum route duration (T_{max}) is imposed. The G-VRP is modeled in [1] including dummy copies of the AFSs to manage multiple visits at the same AFS. Hereafter we refer to such a formulation as the EMH model. A further formulation in which the AFSs are implicitly addressed is proposed in [2], together with a Branch-and-Cut method (hereafter, KK-B&C). In this paper, the G-VRP is modeled by Mixed Integer Linear Programming (MILP) without cloning AFSs, since this increases the number of nodes and, consequently, the problem complexity. The number of variables employed is also reduced by pre-computing for each couple of customers an efficient set of AFSs, including only those that may be actually used in an optimal solution. Our formulation is tested on two sets of benchmark instances taken by [1], showing that it outperforms both the EMH model and the KK-B&C.

2 A New MILP Model for the G-VRP

The G-VRP is defined on a directed complete graph $G = (N, A)$, where $N = I \cup \{0\}$, with I set of customers and 0 the depot, and $A = \{(i, j) : i \in N, j \in N,\ i \neq j\}$. The set F of available AFSs is known. The following data are given: $\forall i \in N \cup F, \forall j \in N \cup F, i \neq j$, travel time, t_{ij}, and travel distance, d_{ij}; $\forall i \in I$, p_i is the service time, while $\forall s \in F$ it represents the refueling time; Q is the refueling capacity. Our model is based on the computation of the sets L_{ij} of AFSs that may be convenient for an AFV to move in a feasible way from i to j, $\forall (i, j) \in A$. These sets are computed in the following way. Let

[1] Email:maurizio.bruglieri@polimi.it

[2] Email:simona.mancini@unica.it

[3] Email:pezzella@dii.univpm.it

[4] Email:ornella.pisacane@uniecampus.it

$s^* = \arg\min_{s \in F : d_{is} \leq \frac{Q}{r}, d_{sj} \leq \frac{Q}{r}} \{d_{is} + d_{sj}\}$. At the beginning $L_{ij} := F$ and for each $s \in F$ if $d_{is} > d_{is^*}$ and $d_{sj} > d_{s^*j}$ or $d_{is} > \frac{Q}{r}$ or $d_{sj} > \frac{Q}{r}$, the AFS s is removed from L_{ij}. Moreover, $\hat{t}_{ijs} = t_{is} + t_{sj} - t_{ij}$ is the detour time of an AFV to visit $s \in L_{ij}$ going from i to j and $\hat{d}_{ijs} = d_{is} + d_{sj} - d_{ij}$ is its detour distance. We model the G-VRP through the following binary variables: routing variables $x_{ij},\ \forall (i,j) \in A$, equal to 1 if node j is visited just after node i (directly or through an AFS), 0 otherwise; $z_{ijs},\ \forall (i,j) \in A,\ \forall s \in L_{ij}$ equal to 1 if AFS s is employed to go from i to j, 0 otherwise. Moreover, we use the following continuous variables $\forall i \in N$: the residual fuel level y_i of AFV at i; the time τ_i on which i is reached. The MILP model is detailed in the following:

$$(1) \qquad \min \sum_{(i,j) \in A} \left(d_{ij} x_{ij} + \sum_{s \in L_{ij}} \hat{d}_{ijs} z_{ijs} \right)$$

s.t.

$$(2) \qquad \sum_{s \in L_{ij}} z_{ijs} \leq x_{ij} \quad \forall\, (i,j) \in A$$

$$(3) \qquad \sum_{j \in N : j \neq i} x_{ij} = 1 \quad \forall\, i \in I$$

$$(4) \qquad \sum_{i \in N : i \neq j} x_{ji} = \sum_{i \in N : i \neq j} x_{ij} \quad \forall\, j \in N$$

$$(5) \qquad \sum_{j \in N : j \neq 0} x_{0j} \leq m$$

$$(6) \qquad \sum_{j \in N : j \neq 0} x_{j0} \leq m$$

$$(7) \qquad \tau_j \geq \tau_i + (t_{ij} + p_i) x_{ij} + \sum_{s \in L_{ij}} (\hat{t}_{ijs} + p_s) z_{ijs} - l_0 (1 - x_{ij}) \ \forall i \in N, \forall j \in I, i \neq j$$

$$(8) \qquad \tau_j \leq T_{max} - (t_{j0} + p_j) - \sum_{s \in L_{j0}} (\hat{t}_{j0s} + p_s) z_{j0s} \ \ \forall j \in N \setminus \{0\}$$

$$(9) \qquad y_j \leq \sum_{s \in L_{ij}} (Q - r \cdot \mathrm{d}_{sj}) z_{ijs} + Q(1 - \sum_{s \in L_{ij}} z_{ijs}) \ \forall j \in I, \forall i \in I, i \neq j$$

$$(10) \qquad y_j \leq y_i - r \cdot d_{ij} + 2Q(1 - x_{ij} + \sum_{s \in L_{ij}} z_{ijs}) \quad \forall j \in I, \forall i \in N, i \neq j$$

$$(11) \qquad y_i \geq r \cdot d_{i0} (x_{i0} - \sum_{s \in L_{ij}} z_{i0s}) \quad \forall i \, \in \, I$$

$$(12) \qquad \sum_{s \in L_{ij}} r \cdot d_{s0} z_{i0s} \leq Q \quad \forall i \, \in \, I$$

$$(13) \qquad y_i \geq \sum_{s \in L_{ij}} (r \cdot d_{is} z_{ijs}) \quad \forall (i,j) \in A$$

$$(14) \qquad y_0 \leq Q$$

$$(15) \qquad x_{ij} \in \{0,1\} \quad \forall (i,j) \in A$$

$$(16) \qquad y_i \geq 0,\ \tau_i \geq 0 \quad \forall i \, \in \, N$$

$$(17) \qquad z_{ijs} \in \{0,1\} \quad \forall (i,j) \in A, \forall s \in L_{ij}$$

Fig. 1. New formulation for the G-VRP.

Objective function (1) minimizes the total travel distance. Between each pair of customers, at most one AFS may be visited (2) while each customer must be visited exactly once (3). Route continuity is assured by (4). The number of AFVs is limited by (5)-(6). Arrival time at each node is ruled by (7) that also exclude sub-tours. Maximum route duration is enforced by (8). Fuel level, at each node, is ruled by (9)-(10). An AFV, after visiting its last customer, must have enough fuel to return to the depot either without refueling (11) or refueling (12). In case of refueling, (13) guarantee the AFV can reach the selected AFS. The AFVs leave fully recharged from the depot thanks to (14). Finally, the variables nature is specified in (15)-(16)-(17).

3 Some Numerical Results

We tested the performances of our model on the benchmark instance sets S1 and S3 of [1], each one with 10 instances and every instance with 20 customers, on average. Both our model and EMH one are solved with CPLEX12.5 with a CPU time limit of $3,600$ s. In the EMH model, for each AFS, m copies are introduced. Concerning S1, our model optimally solves 7 instances with an average CPU time of $1,772$ s and an average Relative MIP Gap (RMG) of 3.22%. Instead, EMH always reaches the CPU time limit (never certifying the optimality) with an average RMG of 29.97% and, for one instance, it is not able even to find a feasible solution. For three instances, it finds the same optimal value of ours, without certifying its optimality, thus showing that their lower bound is too weak. Comparing our results with those of KK-B&C (Table 5 of [2]), despite they use a more powerful computer, we detect two more optimal solutions with a lower average CPU time (theirs is $2,373$ s) and with a lower average RMG (theirs is 3.5%). About S3, our model performs better than on S1, probably because the greater number of AFSs allows exploiting better the variables saving given by the non-generation of dummy copies of the AFSs. Indeed, our model solves to the optimality 9 over 10 instances with an average CPU time of $1,040$ s and an average RMG of 0.68%. While, EMH always reaches the CPU time limit with an average RMG of 22.42%. On five instances it is not able even to find a feasible solution within the time limit and only on two instances, the optimal value is found. Comparing our results with those of the KK-B&C, two more optimal solutions are found, with a lower average CPU time (theirs is $1,204$ s) and average RMG (theirs is 1.38%).

References

[1] Erdogan, S., and E. Miller-Hooks, *A Green Vehicle Routing Problem*, Transportation Research Part E. **48** (2012), 100–114.

[2] Koç, C. and I., Karaoglan, *The green vehicle routing problem: A heuristic based exact solution approach*, Applied Soft Computing. **39** (2016), 154–164.

A Branch&Price&Cut algorithm for the Vehicle Routing Problem with Intermediate Replenishment Facilities

Paolo Gianessi [1]

École des Mines de Saint-Étienne, FAYOL-EMSE, CNRS:UMR6158, LIMOS, France

Alberto Ceselli [2]

Dipartimento di Informatica, Università degli Studi di Milano, Italy

Lucas Létocart [3] Roberto Wolfler Calvo [3]

Université Paris 13, Sorbonne Paris Cité, LIPN, CNRS, (UMR 7030), France

Abstract

We present a Branch&Price&Cut algorithm for the Vehicle Routing Problem with Intermediate Replenishment Facilities that relies on a new extended formulation. The aim of this latter is to tackle symmetry issues by dropping out the vehicle index. The linear relaxation is further strengthened by adding valid inequalities.

Keywords: Column Generation, Valid Inequalities, Branch&Price&Cut, Vehicle Routing Problem with Intermediate Replenishment Facilities.

1 Introduction

The *Vehicle Routing Problem with Intermediate Replenishment Facilities (VRPIRF)* is defined on a graph where the node set consists of a *central depot* Δ, a set C of n *customers*, and f *replenishment facilities*.

[1] Email: paolo.gianessi@emse.fr

[2] Email: Alberto.Ceselli@unimi.it

[3] Email: {lucas.letocart,roberto.wolfler}@lipn.univ-paris13.fr

The aim is to find a least cost set of *routes* that visits each client exactly once, the cost of a route being the sum of the costs of the visited arcs. Each client has a *demand* and can be served by one of the n_K homogeneous, fixed capacity *vehicles* based at the depot. Furthermore, vehicles can recharge at replenishment facilities so as to perform not one but a sequence of routes called a *rotation*. However, the rotation of a vehicle must start and end at the depot and its total *duration* (the sum of the travel, service and recharge times associated with the visited arcs, clients, and depots, respectively) must not exceed a given *shift length*.

VRPIRF [11] is the particular case of the *Multiple Depot VRP with Inter-Depot routes* (*MDVRPI*, [9]) with only one depot. MDVRPI itself is a generalization of the *Multi-Depot VRP (MDVRP)* in which each depot acts both as the base for the vehicles of its own fleet, and as a facility for vehicles based at other depots. Hence, VRPIRF turns out to belong to the family of Multi-Depot VRPs (see e.g. [2]), one of the most investigated families of VRPs. The multiple use of vehicles is an element that VRPIRF has also in common with the *Multi-Trip VRP (MTVRP)* [8].

In Section 2, we describe an extended formulation which makes use of *replenishment arcs* and *arrival times* together with valid connectivity inequalities, while Section 3 is devoted to the description of the Branch&Price&Cut algorithm.

2 Formulation

We propose a new Set-Partitioning formulation without the vehicle index for the VRPIRF. A solution to overcome vehicle-related symmetry issues, which affect some previous formulations, consists in using *arrival times* and *replenishment arcs*. Arrival times (inspired by e.g. [1], [6]) enable to keep track of the elapsed time along a rotation: the association between a vehicle and the routes it performs to compute its total service time can be disregarded, and the vehicle index removed. Further, arrival times assure the connection of a solution as a *side-effect*. However, in order to use them, a rotation must be represented as a sequence of arcs in which each intermediate has indegree and outdegree equal to 1. This representation shift is what *replenishment arcs* (see e.g. [6], [10]) $A_P = C \times C$ allow to do, as they model recharges in between two clients so that facility nodes are no more needed. We will use them along with *base arcs* $A_0 = V \times V$, where node set is $V = \{\varDelta\} \cup C$.

As to decision variables, we have three sets of binary variables, namely *route variables* x_r, *base arc variables* x_{ij}, $ij \in A_0$ and *replenishment arc variables* w_{ij}, $ij \in A_P$, whereas *arrival time variables* z_{ij}, $i, j \in V$, are real nonnegative. Along with problem-defining contraints, we introduce *connectivity inequalities* in order to refine the fractional solution of a node of the Branch & Bound tree and tighten the lower bound. They generalize *subtour elimination constraints (SECs)* in that both base and replenishment arcs are taken into account, so as to exploit the structural similarity between a rotation expressed with replenishment arcs, and a classical route. To separate such connectivity inequalities we use CVRPSEP (see [7]), a library of routines to separate various families of valid cuts for symmetric Capacitated VRP. Route $r \in \mathscr{R}$ is associated with cost c_r and binary terms a_r^i, b_r^{ij}, $e_r'^{\Delta}$ (resp. $e_r''^{\Delta}$), $e_r'^{p}$ (resp. $e_r''^{p}$), which denote whether $r \in \mathscr{R}$ visits $i \in C$ or $ij \in A_0$, starts (resp. ends) at Δ or at facility p. Terms $t_{ij}, ij \in A_0$ and $u_{ij}, ij \in A_P$ denote the time associated with a base or replenishment arc.

$$
\begin{array}{llll}
\min & \sum\limits_{r\in\mathscr{R}} c_r x_r & & \\
\text{s.t.} & \sum\limits_{r\in\mathscr{R}} a_r^i\, x_r = 1 & \forall i \in C & \text{clients service 1} \\
& \sum\limits_{r\in\mathscr{R}} e_r'^{p} x_r = \sum\limits_{r\in\mathscr{R}} e_r''^{p} x_r & \forall p \in F & \text{routes balance at facilities} \\
& \sum\limits_{r\in\mathscr{R}} e_r'^{\Delta} x_r = \sum\limits_{r\in\mathscr{R}} e_r''^{\Delta} x_r \leq n_K & & \text{routes bal. at } \Delta\text{, nr of vehicles} \\
& \sum\limits_{r\in\mathscr{R}} b_r^{ij} x_r = x_{ij} & \forall ij \in A_0 & \text{route-arc variables link} \\
& \sum\limits_{ji\in A_0} x_{ji} + \sum\limits_{ji\in A_P} w_{ji} = \sum\limits_{ij\in A_0} x_{ij} + \sum\limits_{ij\in A_P} w_{ij} = 1 & \forall i \in C & \text{clients service 2} \\
& z_{\Delta i} = t_{\Delta i} x_{\Delta i} & \forall i \in C & \text{arrival times} \\
& (t_{\Delta i} + t_{ij}) x_{ij} + (t_{\Delta i} + u_{ij}) w_{ij} \leq z_{ij} \leq (T - t_{j\Delta})(x_{ij} + w_{ij}) & \forall ij \in A_P & \\
& \sum\limits_{j\in V\setminus i} z_{ij} = \sum\limits_{j\in V\setminus i} z_{ji} + \sum\limits_{j\in V\setminus i} t_{ij} x_{ij} + \sum\limits_{j\in C\setminus i} u_{ij} w_{ij} & \forall i \in C & \text{arrival times propagation} \\
& (t_{\Delta i} + t_{i\Delta}) x_{i\Delta} \leq z_{i\Delta} \leq T x_{i\Delta} & \forall i \in C & \text{max shift length}
\end{array}
$$

3 Branch&Price&Cut algorithm

A Column Generation-based framework is considered, where the Pricing Problem (PP) consists of an *Elementary Shortest Path Problem with Resource Constraints* (*ESPPRC*) to determine new negative reduced cost route variables. The ESPPRC is solved by means of a Dynamic Programming algorithm inspired by the one presented in [5] and enhanced with *ng*-paths [3] and a q-paths-based [4] completion bound method to restrain the combinatorial explosion. Then, the exploration of the Branch&Price&Cut tree is guided by problem-tailored branching rules. In our CG-based framework, branching rules can only concern base arc variables x_{ij}, $ij \in A_0$, or replenishment arc variables w_{ij}, $ij \in A_P$, as route variables x_r cannot be branched on. On the other hand, branching on arc variables requires no transformation, since the link between route and arc variables is explicit in the model. The chosen branching strategy consists in branching on replenishment arc variables w_{ij} (total number, total number per facility, single variables) and then on base arc variables x_{ij} (node with the highest number of fractional outgoing arc variables, single variables). The propagation of branching decisions on x_{ij} variables to the PP requires to transform back a solution with routes variables x_r, whereas the branching decisions on w_{ij} variables have no effect on the PP as replenishment arcs do not appear in the graph of the PP.

References

[1] S.Almoustafa, S.Hanafi, N.Mladenovic, *New exact method for large asymmetric distance-constrained vehicle routing problem.* European Journal of Operational Research, 226(3), 386–394, 2013.

[2] R.Baldacci, A.Mingozzi, *A unified exact method for solving different classes of vehicle routing problems.* Mathematical Programming, 120(2):347–380, 2009.

[3] R.Baldacci, A.Mingozzi, R.Roberti, *New Route Relaxation and Pricing Strategies for the Vehicle Routing Problem.* Operations Research, 59(5), 1269–1283, 2011.

[4] N.Christofides, A.Mingozzi, P.Toth, *Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations.* Mathematical Programming, 20(1), 255–282, 1981.

[5] D.Feillet, P.Dejax, M.Gendreau, C.Gueguen, *An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems..* Networks, 44(3), 216–229, 2004.

[6] I. Kara, *Arc based integer programming formulations for the Distance Constrained Vehicle Routing problem.* 3rd IEEE International Symposium on Logistics and Industrial Informatics (LINDI), 33–38, 2011.

[6] I.Karaoglan, C.Koc, *A branch and cut algorithm for the vehicle routing problem with multiple use of vehicles.* Proceedings of the 41st International Conference on Computers & Industrial Engineering, 2011.

[7] J.Lysgaard, *CVRPSEP: A Package of Separation Routines for the Capacitated Vehicle Routing Problem.* Technical Report, Handelshøjskolen i Århus. Institut for Driftsøkonomi og Logistik, 2003.

[8] D.Cattaruzza, N.Absi, D.Feillet, *Vehicle routing problems with multiple trips.* 4OR: A Quarterly Journal of Operations Research, http://hal-emse.ccsd.cnrs.fr/emse-01250603, to appear, 2016.

[9] I.Muter, J.-F.Cordeau, G.Laporte, *A Branch-and-Price Algorithm for the Multidepot Vehicle Routing Problem with Interdepot Routes.* Transportation Science, 48(3):425–441, 2014.

[10] O.J.Smith, N.Boland, H.Waterer, *Solving shortest path problems with a weight constraint and replenishment arcs.* Computers & Operations Research, 39(5), 964–984, 2012.

[11] C.D.Tarantilis, E.E.Zachariadis, C.T.Kiranoudis, *A Hybrid Guided Local Search for the Vehicle-Routing Problem with Intermediate Replenishment Facilities.* INFORMS Journal on Computing, 20(1):154–168, 2008.

Linear Models and Computational Experiments for the Quadratic TSP

Anja Fischer [a,1] J. Fabian Meier [b,2] Ulrich Pferschy [c,3]
Rostislav Staněk [c,3]

[a] *Institute for Numerical and Applied Mathematics, University of Goettingen, Goettingen, Germany*

[b] *Continentale Krankenversicherung a.G., Dortmund, Germany*

[c] *Dept. of Statistics and Operations Research, University of Graz, Graz, Austria*

Abstract

We consider the Symmetric Quadratic Traveling Salesman Problem (SQTSP), which is a generalization of the classical TSP where each sequence of two consecutive edges in the tour gives rise to a certain cost value. For the standard linearization we apply a purely integral subtour elimination strategy which outperforms the usual fractional separation routine in computational experiments, even if strengthened inequalities are added. The maximization version of the problem is introduced and turns out to benefit from this strengthening. Finally, a new geometry-based linearization with only a linear number of additional variables is presented for the Angular Metric TSP and variants thereof. It is faster than the other approaches for medium-sized instances of one of the variants.

Keywords: TSP, Angular Metric TSP, subtour elimination, computational study

[1] Email: `anja.fischer@mathematik.uni-goettingen.de`

[2] Email: `brief@fabianmeier.de`

[3] Supported by the Austrian Science Fund (FWF): P 23829-N13.
Email: {`pferschy, rostislav.stanek`}`@uni-graz.at`

1 Introduction

In this contribution we consider a generalization of the classical *Traveling Salesman Problem* (TSP) in terms of the cost structure. For each pair of adjacent edges there is a cost coefficient representing the cost of using both edges in a tour. In other words, if the tour visits vertices i, j and k in this order, costs of d_{ijk} with $d_{ijk} = d_{kji}$ arise. This allows the modeling of symmetric transition costs such as turning costs of a physical path but also setup costs when moving from one activity to another. The resulting *Symmetric Quadratic Traveling Salesman Problem* (SQTSP) asks for a cost-minimal Hamiltonian cycle with respect to this quadratic objective function.

The SQTSP was previously studied in [3] and [2]. An important special case is the *Angular Metric TSP (AngleTSP)* [1] that arises in robotics where d_{ijk} represents the *turning angle* between edges (i, j) and (j, k). Indeed, the energy demand for the motion of a robot also depends on the turning angle.

We will also consider the *Maximum Symmetric Quadratic Traveling Salesman Problem* (MaxSQTSP) which asks for a *cost-maximal* tour T w. r. t. the quadratic objective function and has not been studied in the literature before.

2 Fractional vs. integral approach

Following the notation in [2] let $G = (V, V^{\{2\}})$ be a complete graph with vertex set $V = \{1, \dots, n\}$, $n \geq 3$, and edge set $V^{\{2\}} := \big\{(i, j) = (j, i) \colon i, j \in V, i \neq j\big\}$. A 2-edge $e^{\langle 3\rangle} := \langle i, j, k\rangle \in V^{\langle 3\rangle} := \big\{\langle i, j, k\rangle = \langle k, j, i\rangle \colon i, j, k \in V, |\{i, j, k\}| = 3\big\}$ is defined as a sequence of three distinct vertices where the reverse sequence is regarded as identical. Furthermore, for a set of 2-edges $V^{\langle 3\rangle}$ let $G = (V, V^{\langle 3\rangle})$ denote a complete 2-*graph*. Using binary edge variables $x_e, e \in V^{\{2\}}$, SQTSP can be formulated using the well-known *subtour elimination constraints* in (3).

$$\min \sum_{e^{\langle 3\rangle} = \langle i,j,k\rangle \in V^{\langle 3\rangle}} d_{e^{\langle 3\rangle}} x_{(i,j)} x_{(j,k)} \tag{1}$$

$$\text{s. t.} \sum_{j \in V \setminus \{i\}} x_{(i,j)} = 2, \qquad i \in V, \tag{2}$$

$$\sum_{e=(i,j) \in S^{\{2\}}} x_e \leq |S| - 1, \qquad S \subsetneq V,\ S \neq \emptyset, \tag{3}$$

$$x_e \in \{0, 1\}, \qquad e \in V^{\{2\}}. \tag{4}$$

We linearize this model by introducing a cubic number of variables $y_{e^{\langle 3\rangle}} = y_{ijk} \in \{0,1\}$ for all 2-edges $e^{\langle 3\rangle} = \langle i,j,k\rangle \in V^{\langle 3\rangle}$, where $y_{ijk} = 1$ if and only if the vertices i, j and k are visited in the tour consecutively, see [2]. The x-variables are coupled with the y-variables by constraints (5).

$$\begin{aligned}
\min \quad & \sum_{e^{\langle 3\rangle} \in V^{\langle 3\rangle}} d_{e^{\langle 3\rangle}} y_{e^{\langle 3\rangle}} \\
\text{s. t. } & (2), (3), (4), \\
& x_{(i,j)} = \sum_{k \in V \setminus \{i,j\}} y_{ijk} = \sum_{k \in V \setminus \{i,j\}} y_{kij}, \qquad (i,j) \in V^{\{2\}}, \qquad (5) \\
& y_{e^{\langle 3\rangle}} \in \{0,1\}, \qquad e^{\langle 3\rangle} \in V^{\langle 3\rangle}. \qquad (6)
\end{aligned}$$

This ILP can be used to solve the SQTSP by the “standard” TSP techniques of separating the subtour elimination constraints (3): Identify the violated constraints on fractional solutions during the branch and cut solution process by solving appropriate min-cut problems as it was done in [2].

In this contribution we focus on a different strategy which was already tested (with limited success) for the classical TSP in [4]: Relax all subtour constraints (3) first and then solve the remaining model to integral optimality using an ILP solver. In the resulting 2-matching cycles can be found by a simple scan. Now, we add a subtour elimination constraint for each such cycle and resolve the enlarged ILP model. This process is repeated until we get a solution consisting of only one cycle, i. e. an optimal tour.

3 Computational Experiments

We performed extensive computational experiments for several instance classes of SQTSP following benchmarks from the literature. We also tested different strengthened variants of subtour elimination constraints suggested in [2]. It turned out that the simple integral approach significantly outperforms the standard fractional separation procedure known from the literature for all types of test instances. Moreover, the standard versions were faster so that sophisticated separation strategies do not pay off in the minimization case.

A different picture appears for MaxSQTSP, where some of the strengthened subtour elimination constraints from [2] do speed up the solution process. The comparison between purely integral and fractional subtour elimination is less clear and depends on the particular type of test instances. For MaxAngleTSP the computational results showed a surprisingly dichotomous behavior for odd and even cardinalities of V. This will be subject of further study.

4 A geometry-based MILP linearization for AngleTSP

For the special case of the AngleTSP we can exploit the geometry of the problem and avoid the cubic number of additional binary variables $y_{e^{\langle 3 \rangle}}, e^{\langle 3 \rangle} \in V^{\langle 3 \rangle}$. Instead, the following linearization adds only a linear number of real-valued variables $y_j \in \mathbb{R}_0^+, j \in V$, expressing the turning angle of a tour in j. Thus, we replace (1) by $\min \sum_{j \in V} y_j$ and add the constraints

$$y_j \geq \sum_{\substack{i,k \in V \setminus \{j\} \\ i<k}} d_{ijk} x_{(i,j)} x_{(j,k)}, \qquad j \in V. \tag{7}$$

We can show that these inequalities are equivalent to the following linear inequalities if the degree two (2) and the integrality constraints (6) are satisfied.

$$y_j \geq \sum_{k \in V \setminus \{j\}} d_{ijk} x_{(j,k)} - \pi, \qquad i, j \in V, \ i \neq j. \tag{8}$$

Theorem 4.1 *The set of constraints (2) and (7) is equivalent to the set of constraints (2) and (8) for $x_{(i,j)} \in \{0,1\}, (i,j) \in V^{\{2\}}$, $y_j \in \mathbb{R}_0^+, j \in V$.*

Our computational tests show that using this formulation the running times can be improved for instances with up to $n = 55$ if we consider a variant of the AngleTSP where the turning angles are part of a linear combination with the distances between the vertices. For larger or classical AngleTSP instances the running times are often worse. One reason for this behavior might be the larger root node gaps.

References

[1] Aggarwal, A., D. Coppersmith, S. Khanna, R. Motwani and B. Schieber, *The angular-metric traveling salesman problem*, SIAM Journal on Computing **29** (2000), pp. 697–711.

[2] Fischer, A. and C. Helmberg, *The symmetric quadratic traveling salesman problem*, Mathematical Programming **142** (2013), pp. 205–254.

[3] Jäger, G. and P. Molitor, *Algorithms and experimental study for the traveling salesman problem of second order*, Springer LNCS **5165** (2008), pp. 211–224.

[4] Pferschy, U. and R. Staněk, *Generating subtour elimination constraints for the TSP from pure integer solutions*, Central European Journal of Operations Research (2016), to appear, doi: 10.1007/s10100-016-0437-8.

Graphs with large girth are b-continuous [2]

Ana Silva [a,b,1] Cláudia Linhares Sales [a,c,1]

[a] *ParGO Research Group - Parallelism, Graphs and Optimization*

[b] *Departamento de Matemática, Universidade Federal do Ceará, Brazil*

[c] *Departamento de Computação, Universidade Federal do Ceará, Brazil*

Abstract

A b-coloring of the vertices of a graph is a proper coloring where each color class contains a vertex which is adjacent to each other color class. The b-chromatic number of G is the maximum integer $b(G)$ for which G has a b-coloring with $b(G)$ colors. A graph G is b-continuous if G has a b-coloring with k colors, for every integer k in the interval $[\chi(G), b(G)]$. It is known that not all graphs are b-continuous. Here, we show that if G has girth at least 10, then G is b-continuous.

Keywords: b-chromatic number, b-continuity, graphs with large girth.

Let G be a simple graph. A *coloring of* G is a function $\psi : V(G) \rightarrow \mathbb{N}$ such that $\psi(u) \neq \psi(v)$ whenever $uv \in E(G)$. We say that $u \in V(G)$ is a *b-vertex in* ψ if for every color $c \neq \psi(u)$, there exists $v \in N(u)$ colored with c. Observe that if ψ has a color class c that has no b-vertices, then we can separatedly change the color of each vertex in c to obtain a proper coloring with fewer colors. But since the coloring problem is NP-complete, $\chi(G)$ cannot always be reached. Irving and Manlove [10], interested in the worst case scenario, defined a *b-coloring* as a coloring of G that has at least one b-vertex in each

[1] *Email addresses:* `anasilva@mat.ufc.br` (Silva), `linhares@lia.ufc.br` (Linhares Sales)

[2] Partially supported by CNPq and CAPES, Brazil.

of its color classes, and the *b-chromatic number of* G as the maximum number of colors $b(G)$ used by a b-coloring of G. Finding $b(G)$ is NP-complete [10], even if G is bipartite [13], chordal [9], or a line graph [5].

It is known that $K_{n,n}$ minus a perfect matching only admits b-colorings with 2 and n colors, for $n \in \mathbb{N}$ [13]. Also, for every finite $S \subset \mathbb{N} - \{1\}$, there exists a graph G that admits a b-coloring with k colors iff $k \in S$ [2]. This leads to the following definition: G is *b-continuous* if it has a b-coloring with k colors, for every $k \in \{\chi(G), \cdots, b(G)\}$. Deciding whether a given graph G is b-continuous is NP-complete, even if b-colorings with $\chi(G)$ and $b(G)$ colors are given [2]. Concerning positive results, we mention that the following graph classes are b-continuous: chordal graphs [8,12]; Kneser graphs $K(n, 2)$ for $n \geq 17$ [11]; P_4-sparse graphs [4] and P_4-tidy graphs [3]; and regular graphs with girth at least 6 and with no cycles of length 7 [1].

The *girth of* G is the minimum size $g(G)$ of a cycle in G. Let $m(G)$ be the maximum k for which G has at least k vertices of degree at least $k - 1$. It is not hard to see that $b(G) \leq m(G)$ [10]. In [7], it is conjectured that if G is a d-regular graph with $g(G) \geq 5$ and $d \geq 4$, then $b(G) = m(G)$. This conjecture has also motivated the investigation in [1] mentioned in the previous paragraph.

It is also known that if G is a tree, then (*) $b(G) \geq m(G) - 1$, and one can decide whether $b(G)$ equals $m(G) - 1$ or $m(G)$ in polynomial time [10]. Later, it was noted that in fact this property holds for graphs with large girth, and the most recent result regarding this aspect says that (*) holds whenever $g(G) \geq 7$ [6]. This and the numerous results on regular graphs with large girth indicate that, unlike the classic coloring problem, having large girth somehow helps in finding b-colorings of G. We therefore pose the question below, and give a partial answer to it.

Question 1 *What is the minimum* $\hat{g}$ *s.t.* G *is b-continuous when* $g(G) \geq \hat{g}$*?*

Theorem 1 *If* $g(G) \geq 10$*, then* G *is b-continuous.*

We mention that the known non-b-continuous graphs have girth 4; hence, $5 \leq \hat{g} \leq 10$. Also, if $g(G) \geq 7$, then finding $b(G)$ can be done in polynomial time [6], while finding $\chi(G)$ is NP-complete, even if G is a line graph [14]. Therefore, any proof that (partially) answers Question 1 must have a non-constructive part.

1 Outline of the proof

Let G be any graph, and ψ be a b-coloring of G with k colors. Let $\mathcal{B}(\psi)$ denote the set of all b-vertices in ψ, and for each color i denote by B_i the set of b-vertices in color class i. We want to change the color of some $x \in V(G) \setminus \mathcal{B}(\psi)$, while ensuring that exactly one color class loses b-vertices, and that no new b-vertex is created. For each $x \in V(G) \setminus \mathcal{B}(\psi)$, let $U(x)$ contain $w \in \mathcal{B}(\psi)$ if x is the only neighbor of w colored with $\psi(x)$. We say that x is *mutable* if, for every $w \in N(x) \setminus \mathcal{B}(\psi)$, there exists $i \in \{1, \cdots, k\} \setminus \psi(N[x])$ such that $\{1, \cdots, k\} \setminus \psi(N[w] \setminus \{x\}) \neq \{i\}$. This means that changing the color of x to i does not create new b-vertices.

Lemma 2 *If $x \in V(G) \setminus \mathcal{B}(\psi)$ is mutable and $|\psi(U(x))| = 1$, then there exists a b-coloring of G with $k-1$ colors.*

For each $u \in \mathcal{B}(\psi)$ and each color $i \in \{1, \cdots, k\} \setminus \psi(u)$, let $B_i(u) = N(u) \cap B_i$, and $R_i(u)$ be the remaining neighbors of u colored with i. We say that color i is *weak in* $N(u)$ if, $\forall x \in R_i(u)$, x is mutable and:

(**) For every $w \in U(x) \setminus \{u\}$, there exists $w' \in \mathcal{B}_{\psi(w)} \setminus N(R_i(u))$.

Condition (**) means that, if the color of every $x \in R_i(u)$ gets changed, then not every b-vertex of color $\psi(w)$ is lost.

Lemma 3 *Let $u \in \mathcal{B}(\psi)$ and $i \in \{1, \cdots, k\} \setminus \{\psi(u)\}$. If i is weak in $N(u)$ and $B_i(u) = \emptyset$, then there exists a b-coloring of G with $k-1$ colors.*

Let $D_k(G)$ be the subset of vertices of degree at least $k-1$ in G, and for each $u \in V(G)$, let $N_2(u)$ ($N_{\leq 2}(u)$) denote the set of vertices at distance 2 (at most 2) from u. We say that $u \in V(G)$ is a *k-iris* in G if there exists $S \subseteq N(u) \cap D_k(G)$ with cardinality $k-1$; and we say that u is a *dilated k-iris* if there exists a subset $S \subseteq N_{\leq 2}(u) \cap D_k(G)$ with k vertices such that: $N(v) \cap N(w) = \emptyset$, for every $v, w \in S \cap N_2(u)$, $v \neq w$; and $|S \cap N_2(u)| \geq 1$. The next two lemmas finish the proof. We mention that the proof in [1] actually proves the k-iris part of the next lemma, and works for any graph with girth 6 that has no cycles of length 7. Also, girth 5 suffices for the recoloring part, i.e., the condition $g(G) \geq 10$ is necessary only for coloring the dilated k-iris.

Lemma 4 *Let G be a graph with girth at least 10. If G has a k-iris or dilated k-iris, where $k \geq \chi(G)$, then G has a b-coloring with k colors.*

Lemma 5 *Let G be a graph with $g(G) \geq 5$, ψ be a b-coloring of G with k colors, where $k \geq \chi(G) + 1$, and $u \in \mathcal{B}(\psi)$. Then there exists a weak color i in $N(u)$ such that $B_i(u) = \emptyset$, or u is a $(k-1)$-iris, or a dilated $(k-1)$-iris.*

References

[1] R. Balakrishnan and T. Kavaskar. b-coloring of Kneser graphs. *Discrete Appl. Math.* 160 (2012) 9–14.

[2] D. Barth, J. Cohen and T. Faik. On the b-continuity property of graphs. *Discrete Appl. Math.* 155, 1761–1768, 2007.

[3] C.I. Betancur Velasquez, F. Bonomo, and I. Koch. On the b-coloring of P_4-tidy graphs. *Discrete Appl. Math.* 159 (2011) 67–76.

[4] F. Bonomo, G. Duran, F. Maffray, J. Marenco and M. Valencia-Pabon. On the b-coloring of cographs and P4-sparse graphs. *Graphs and Combin.* 25 (2), 153–167, 2009.

[5] V. Campos, C. Lima, N.A. Martins, L. Sampaio, M.C. Santos and A. Silva. The b-chromatic index of graphs. *Discrete Mathematics* 338 (11) (2015) 2072–2079.

[6] V. Campos, C. Lima and A. Silva. Graphs with girth at least 7 have high b-chromatic number. *European Journal of Combinatorics* 48 (2015), 154–164.

[7] A. El Sahili and H. Kouider. About b-colouring of regular graphs. *Utilitas Math.* 80 (2009) 211–215.

[8] T. Faik, About the b-continuity of graphs. *Electron. Notes in Discrete Math.* 17 (2004) 151–156.

[9] F. Havet, C. Linhares-Sales and L. Sampaio. b-coloring of tight graphs. *Discrete Appl. Mathematics* 160 (18) (2012) 2709–2715.

[10] R.W. Irving and D.F. Manlove. The b-chromatic number of a graph. *Discrete Appl. Math.* 91, 127–141, 1999.

[11] R. Javadi and B. Omoomi. On b-coloring of the Kneser graphs. *Discrete Math.* 309, 4399–4408, 2009.

[12] J. Kára, J. Kratochvíl and M. Voigt. b-Continuity. Preprint no. M14/04, Faculty for Mathematics and Natural Science, Technical University Ilmenau, 2004.

[13] J. Kratochvíl, Zs. Tuza, and M. Voigt. On the b-chromatic number of graphs. WG 2002, *Lecture Notes In Computer Science* 2573 (2002) 310–320.

[14] V.V. Lozin and M. Kaminski. Coloring edges and vertices of graphs without short or long cycles. *Contributions do Discrete Mathematics* 2 (1) (2007).

Proper connection number 2, connectivity, and forbidden subgraphs

Christoph Brause[a], Trung Duy Doan[a,b,1], Ingo Schiermeyer[a]

[a] *Institute of Discrete Mathematics and Algebra*
TU Bergakademie Freiberg
Freiberg, Germany
brause@math.tu-freiberg.de, ingo.schiermeyer@tu-freiberg.de

[b] *School of Applied Mathematics and Informatics*
Hanoi University of Science and Technology
Hanoi, Vietnam
trungdoanduy@gmail.com

Abstract

An edge-coloured graph G is called *properly connected* if any two vertices are connected by a path whose edges are properly coloured. The *proper connection number* of a graph G, denoted by $pc(G)$, is the smallest number of colours that are needed in order to make G properly connected. In this paper we consider sufficient conditions in terms of connectivity and forbidden subgraphs, implying a graph to have proper connection number 2.

Keywords: proper connection number, 2-connected, forbidden subgraphs

[1] Financial support by the Free State of Saxony (Landesgraduiertenstipendium) is thankfully acknowledged

1 Introduction

We use [4] for terminology and notation not defined here and consider simple and undirected graphs only.

The concept of proper connections in graphs is an extension of proper colourings and is motivated by rainbow connections of graphs. Andrews et al. [1] and, independently, Borozan et al. [3] introduced the concept as follows:

An edge-coloured graph G is called *properly connected* if every two vertices $u, v \in V(G)$ are connected by a path whose edges are properly coloured. The proper connection number $pc(G)$ is the smallest number of colours needed to colour a graph G properly connected. We say, an edge-colouring c has the *strong property* if for every two vertices $u, v \in V(G)$ there exists two properly coloured paths $P_1 : u = w_1w_2 \dots w_k = v$ and $P_2 : u = z_1z_2 \dots z_l = v$ such that $c(w_1w_2) \neq c(z_1z_2)$ and $c(w_{k-1}w_k) \neq c(z_{l-1}z_l)$. We note that $pc(G) = 1$ if and only if G is complete [3].

For simplifying notation, let $[k]$ be the set $\{1, 2, \dots, k\}$ for some positive integer k. Following common notation, we say G contains an *induced subgraph* F if there is a vertex subset $U \subseteq V(G)$ such that $G[U] \cong F$. Therefore, G is *F-free* ($\mathcal{F}$-free) if and only if G contains F (all graphs of $\mathcal{F}$) not as an induced subgraph. Let $S_{i,j,k}$ be the graph consisting of three induced paths of lengths i,j, and k with a common initial vertex, and $\mathcal{S}$ be the set of graphs whose every component is of the form $S_{i,j,k}$ for some $0 \leq i \leq j \leq k$.

In many fields of graph theory, forbidden subgraphs and the connectivity of a graph play an important role. In [2], Bedrossian characterized pairs of forbidden subgraphs for 2-connected graphs implying hamiltonicity. Thus, since every noncomplete, hamiltonian graph has proper connection number 2 [3], his characterization is the starting point for our work to find sufficient conditions in terms of connectivity and forbidden subgraphs such that $pc(G) = 2$ holds for a graph G. We note that all pairs in Bedrossian's characterization contain the claw. Our first result improves that observation by forbidding only the claw.

Theorem 1.1 *Let G be a connected, claw-free, and noncomplete graph. Then $pc(G) = 2$.*

Sketch of the Proof. Suppose, to the contrary, that there exists a connected, claw-free graph of proper connection number at least 3. Moreover, all those graphs are noncomplete. Then, let G be a counterexample of minimum order, i.e. G is connected, claw-free, but $pc(G) \geq 3$, and for all noncomplete but connected induced subgraphs G' of G, it holds $pc(G') = 2$. Now, let H be a connected induced subgraph in G such that

(i) $pc(H) = 2$, and (ii) subject to (i), $n(H)$ is maximum.

Therefore, there exists a vertex v in $V(G-H)$ which is adjacent to at least one vertex of $V(H)$, say u. Now, by some small case to case analysis, one can show that $G[V(H) \cup \{v\}]$ is properly connected, contradicting the minimality of G. □

Further, we find necessary conditions on forbidden subgraphs, implying a proper connection number 2.

Proposition 1.2 (i) *Let $\mathcal{F}$ be a finite set of graphs. If $\mathcal{F} \cap \mathcal{S} = \emptyset$, then there exists a 2-connected, $\mathcal{F}$-free graph G such that $pc(G) = 3$.*

(ii) *Let $0 \leq i \leq j \leq k$. If $i \geq 3$ or $j + k \geq 15$, then there exists a 2-connected, $S_{i,j,k}$-free graph G such that $pc(G) = 3$.*

Using this characterization, it is quit natural to forbid $S_{i,j,k}$ with small i,j, and k, for example $S_{1,1,3}$.

Theorem 1.3 *Let G be a noncomplete, 2-connected, $S_{1,1,3}$-free graph of minimum degree at least 3. Then $pc(G) = 2$.*

Some basic results, which are important for our proofs, make only use of the connectivity of a graph.

Theorem 1.4 (Borozan et al. [3]) *Let G be a 2-connected graph. Then there exists an edge-colouring $c : E(G) \to [3]$ having the strong property.*

Theorem 1.5 (Borozan et al. [3]) *Let G be a 2-connected bipartite graph. Then there exists an edge-colouring $c : E(G) \to [2]$ having the strong property.*

The authors claim that their results still hold if one replaces 2-connectivity by 2-edge-connectivity. As a further consequence, by a result of Paulraja in [5], every 3-connected graph G has a 2-connected bipartite spanning graph. Therefore, Borozan et al. deduced the following result.

Theorem 1.6 (Borozan et al. [3]) *Let G be a 3-connected graph. Then there exists an edge-colouring $c : E(G) \to [2]$ having the strong property.*

There are 2-connected graphs having proper connection number 3, for example graph B in Figure 1 [3]. Since all known graphs have a 3-cut, we study the proper connection number of 3-edge-connected graphs.

Theorem 1.7 *Let G be a 3-edge-connected and noncomplete graph. Then $pc(G) = 2$.*

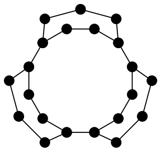


Fig. 1. Graph B

We note that Theorem 1.7 closes the gap in transforming Theorems 1.4, 1.5, and 1.6 to their edge-connected version.

2 Basic sketch of the proofs of Theorems 1.3 and 1.7

The idea of both proof can be described in its basic form as follows:

Suppose, to the contrary, there is a graph G fulfilling all conditions of the corresponding theorem, but has proper connection number at least 3. Since G is 2-edge-connected in both theorems, it contains a cycle as a subgraph. Therefore, G contains a subgraph of proper connection number 2. Now let H be a connected subgraph in G such that

(i) $pc(H) = 2$, and (ii) subject to (i), $n(H)$ is maximum.

Moreover, we can assume, without loss of generality, that H is induced. Now by some case to case analysis, we obtain that the 2-edge-colouring making H properly connected is extendable, contradicting the maximality of H, or we find an induced subgraph $S_{1,1,3}$, in case of Theorem 1.3, or a 2-cut in case of Theorem 1.7, a contradiction.

References

[1] E. Andrews, C. Lumduanhom, E. Laforge, and P. Zhang, *On Proper-Path Colourings in Graphs*, JCMCC, to appear.

[2] P. Bedrossian, *Forbidden Subgraph and Minimum Degree Conditions for Hamiltonicity*, Thesis, Memphis State University, USA, 1991.

[3] V. Borozan, S. Fujita, A. Gerek, C. Magnant, Y. Manoussakis, L. Montero, and Z. Tuza, *Proper connection of graphs*, Discrete Math. 312(17) (2012), 2550–2560.

[4] F. Harary, *Graph Theory*, Addison-Wesley, 1969.

[5] P. Paulraja, *A characterization of Hamiltonian prisms*, J. Graph Theory 17 (2) (1993), 161–171.

On the minimum degree and the proper connection number of graphs

Christoph Brause[a], Trung Duy Doan[a,b,1], Ingo Schiermeyer[a]

[a] *Institute of Discrete Mathematics and Algebra*
TU Bergakademie Freiberg
Freiberg, Germany
brause@math.tu-freiberg.de, ingo.schiermeyer@tu-freiberg.de

[b] *School of Applied Mathematics and Informatics*
Hanoi University of Science and Technology
Hanoi, Vietnam
trungdoanduy@gmail.com

Abstract

An edge-coloured graph G is called *properly connected* if any two vertices are connected by a path whose edges are properly coloured. The *proper connection number* of a graph G, denoted by $pc(G)$, is the smallest number of colours that are needed in order to make G properly connected. In this paper we consider sufficient conditions in terms of the ratio between minimum degree and order of a 2-connected graph G implying that G has proper connection number 2.

Keywords: proper connection number, 2-connected, minimum degree

[1] Financial support by the Free State of Saxony (Landesgraduiertenstipendium) is thankfully acknowledged

1 Introduction

We use [3] for terminology and notation not defined here and consider simple and undirected graphs only.

As an extension of proper colourings and motivated by rainbow connections of graphs, Andrews et al. [1] and, independently, Borozan et al. [2] introduced the concept of proper connections in graphs. An edge-coloured graph G is called *properly connected* if every two vertices $u, v \in V(G)$ are connected by a path whose edges are properly coloured. The proper connection number $pc(G)$ of a graph is the smallest number of colours needed to colour a graph G properly connected. We say, an edge-colouring c has the *strong property* if for every two vertices $u, v \in V(G)$ there exists two properly coloured paths $P_1 : u = w_1 w_2 \dots w_k = v$ and $P_2 : u = z_1 z_2 \dots z_l = v$ such that $c(w_1 w_2) \neq c(z_1 z_2)$ and $c(w_{k-1} w_k) \neq c(z_{l-1} z_l)$.

For simplifying notation, let $[k]$ be the set $\{1, 2, \dots, k\}$ for some positive integer k. Further, let G be a graph, $u, v \in V(G)$ be two distinct vertices, and $P : w_1 w_2 \dots w_k$ be a path, vertex disjoint from G. We say, we *add the ear* P to G by adding the edges uw_1 and vw_k. Hence, for a Θ-*graph* G, there is a cycle C, two distinct vertices $u, v \in V(C)$, and a path P such that G is obained by adding ear P to C. Moreover, we define a 2-*ear cycle* to be a graph obtained by adding some ear to a Θ-graph.

Borozan et al. [2] proved the following results.

Theorem 1.1 (Borozan et al. [2]) *Let G be a 2-connected graph. Then there exists an edge-colouring $c : E(G) \to [3]$ having the strong property.*

Theorem 1.2 (Borozan et al. [2]) *Let G be a 2-connected bipartite graph. Then there exists an edge-colouring $c : E(G) \to [2]$ having the strong property.*

Further, Borozan et al. [2] introduced a construction to obtain 2-connected graphs having proper connection number 3, for example graph B in Figure 1. We note that all those graphs contain odd cycles.

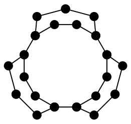


Fig. 1. Graph B with proper connection number 3

By a result of Paulraja in [6], every 3-connected graph G has a 2-connected bipartite spanning graph. Therefore, Borozan et al. deduced the following result.

Theorem 1.3 (Borozan et al. [2]) *Let G be a 3-connected graph. Then there exists an edge-colouring $c : E(G) \to [2]$ having the strong property.*

Based on their construction for 2-connected graphs of proper connection number 3, the authors conjectured the following in [2].

Conjecture 1.4 (Borozan et al. [2]) *Let G be a graph of connectivity $\kappa(G) = 2$ and minimum degree $\delta(G)$ at least 3. Then $pc(G) = 2$.*

In this paper we study sufficient conditions related to the minimum degree and implying a proper connection number at most 2. In particular, we disprove Conjecture 1.4 by constructing a series of 2-connected graphs G_i such that $\delta(G_i) = i$, $n(G_i) = 42i$, and $pc(G_i) \geq 3$.

Proposition 1.5 *For every integer $d \geq 2$, there exists a 2-connected graph G of minimum degree d and order $n = 42d$ such that $pc(G) \geq 3$.*

Further, using our construction technique in a slightly different way, we can prove the following result.

Proposition 1.6 *For all integers $d, k \geq 2$, there exists a connected graph G of minimum degree d and order $n = (d + 1)(k + 1)$ such that $pc(G) = k$.*

By Proposition 1.5, one cannot bound the minimum degree of a 2-connected graph G from below by a constant such that $pc(G) \leq 2$ follows. Therefore, it is quiet natural to ask for a ratio between minimum degree and order of a 2-connected graph, implying $pc(G) \leq 2$.

Theorem 1.7 *Let G be a 2-connected graph of order $n = n(G)$ and minimum degree $\delta(G) > \frac{n+8}{20}$. Then $pc(G) \leq 2$.*

2 Sketch of the proof of Theorem 1.7

Before we start sketching the main steps of our proof, let us mention Menger's theorem, since we shall use it, as well as Theorem 1.2, frequently as a basic tool at several points throughout the proof.

Theorem 2.1 (Menger's theorem [5]) *Let G be a graph, $u, v \in V(G)$ be two distinct vertices. Then the size of a minimum vertex cut for u and v equals the maximum number of internally pairwise disjoint $u - v$ paths.*

Now we are able to sketch our proof by starting with the following claim.

Claim 2.2 *If G is a cycle, Θ-graph, or 2-ear cycle, then $pc(G) \leq 2$.*

Now suppose, to the contrary, that G is a 2-connected graph of order $n = n(G)$, minimum degree $\delta(G) > \frac{n+8}{20}$, and proper connection number at least 3. Trivially, G is no cycle, Θ-graph, or 2-ear cycle by Claim 2.2. Therefore, let Q be a subgraph of G such that

(i) Q is a 2-ear cycle, and (ii) subject to (i), $n(Q)$ is maximum.

Further, let H be a subgraph of G such that

(i) $pc(H) \leq 2$, Q is a subgraph of H, and

(ii) subject to (i), $n(H)$ is maximum.

By our supposition, $G - V(H)$ is a nonempty graph. Now the following series of claims lead us to the nonexistence of G.

Claim 2.3 *Any component of $G - V(H)$ is bipartite.*

Claim 2.4 *Any component of $G - V(H)$ has a bridge.*

Using a result of Jackson in [4], we obtain the next claim.

Claim 2.5 *There exist two distinct vertices u and v in $G - V(H)$ such that*

(i) *u and v are adjacent to vertices in $V(H)$,*

(ii) *there exists a path P connecting u and v in $G - V(H)$ of order at least $4\delta(G) - 2$.*

Claim 2.6 $n(H) \geq 16\delta(G) - 6$.

By Claims 2.5 and 2.6, and the assumption on the minimum degree of G, $n(G) \geq n(H) + n(P) \geq 20\delta(G) - 8 > n(G)$, a contradiction.

References

[1] E. Andrews, C. Lumduanhom, E. Laforge, and P. Zhang, *On Proper-Path Colourings in Graphs*, JCMCC, to appear.

[2] V. Borozan, S. Fujita, A. Gerek, C. Magnant, Y. Manoussakis, L. Montero, and Z. Tuza, *Proper connection of graphs*, Discrete Math. 312(17) (2012), 2550–2560.

[3] F. Harary, *Graph Theory*, Addison-Wesley, 1969.

[4] B. Jackson, *Long cycles in bipartite graphs*, J. of Combinatorial Theory, Series B, 38 (2) (1985), 118–131.

[5] K. Menger, *Zur allgemeinen Kurventheorie*, Fund. Math. 10 (1927), 96–115.

[6] P. Paulraja, *A characterization of Hamiltonian prisms*, J. Graph Theory 17 (2) (1993), 161–171.

A Note on Fractional Coloring and the Integrality gap of LP for Maximum Weight Independent Set

Parinya Chalermsook [a,2], Daniel Vaz [a,b,1]

[a] *Max-Planck-Institut für Informatik, Saarbrücken, Germany*

[b] *Graduate School of Computer Science, Saarland University, Saarbrücken, Germany*

Abstract

We prove a tight connection between two important notions in combinatorial optimization. Let $\mathcal{G}$ be a graph class (i.e. a subset of all graphs) and $r(\mathcal{G}) = \sup_{G \in \mathcal{G}} \frac{\chi_f(G)}{\omega(G)}$ where $\chi_f(G)$ and $\omega(G)$ are the fractional chromatic number and clique number of G respectively. In this note, we prove that $r(\mathcal{G})$ tightly captures the integrality gap of the LP relaxation with clique constraints for the Maximum Weight Independent Set (MWIS) problem. Our proof uses standard applications of multiplicative weight techniques, so it is algorithmic: Any algorithm for rounding the LP can be turned into a fractional coloring algorithm and vice versa. We discuss immediate applications of our results in approximating the fractional chromatic number of certain classes of intersection graphs.

Keywords: Fractional coloring, maximum weight independent set, linear programming.

1 Introduction

In the Maximum Weight Independent Set Problem (MWIS), we are given graph G and weight function $w : V(G) \to \mathbb{R}_{\geq 0}$. A set $J \subseteq V(G)$ is independent if there is no edge in J. Define $w(J) = \sum_{v \in J} w(v)$. Our goal is to compute the maximum weight independent set in G. We denote the weight of a maximum

[1] Email: ramosvaz@mpi-inf.mpg.de

[2] Email: parinya@mpi-inf.mpg.de

weight independent set by $\alpha(G, w)$. This problem is cornerstone in combinatorial optimization and has been extensively studied.

We consider the LP relaxation with clique constraints for MWIS. For each vertex $v \in V$, there is a variable x_v indicating whether vertex v is included.

$$\text{(LP)} \quad \max \sum_{v \in V(G)} w(v) x_v$$
$$\text{s.t.} \sum_{v \in C} x_v \leq 1 \text{ for each clique } C \text{ in graph } G$$

In general, the number of cliques can be exponentially large, but for restricted graph classes (e.g. intersection graphs of rectangles in higher dimensional boxes [4,3]), there is only a polynomial number of maximal cliques. Moreover, it is known that all clique constraints are implied by the canonical SDP relaxation of MWIS, as well as the Lovasz theta function [6]. The main question of our interest is:

How good is (LP) in approximating the maximum-weight independent set?

For each G and weight function w, define $\mathsf{LP}(G, w)$ as the value of an optimal solution for the above LP. The integrality gap $\mathsf{gap}(G, w)$ is the ratio $\frac{\mathsf{LP}(G,w)}{\alpha(G,w)}$.

In this note, we show a tight connection between the integrality gap of (LP) and the fractional chromatic number of a graph. A valid *fractional coloring* for G is a function $\sigma : 2^{V(G)} \rightarrow [0, 1]$ such that (i) the support of σ contains only independent sets, and (ii) for each $v \in V(G)$, we have $\sum_{I : v \in I} \sigma(I) \geq 1$. The *fractional chromatic number* $\chi_f(G)$ is defined as the minimum real number k such that there exists a valid fractional coloring σ, $\sum_I \sigma(I) \leq k$.

For any graph G, a *clique replacement operation* on v is performed by creating graph $G' : V(G') = (V(G) \setminus v) \cup \{v_1, \ldots, v_\ell\}$ and $E(G') = E(G \setminus v) \cup \{v_i u : vu \in E(G)\} \cup \{v_i v_j : i, j \in [\ell]\}$. In words, this operation replaces vertex v with a clique K_ℓ. Let $\mathcal{G}$ be a class of graphs. We say that $\mathcal{G}$ is closed under clique replacement if for any $G \in \mathcal{G}$, a clique replacement operation at v gives us $G' \in \mathcal{G}$. Many natural graph classes are closed under clique replacement, e.g., interval graphs, d-dimensional box graphs, disk graphs, and perfect graphs.

Theorem 1.1 *Let $\mathcal{G}$ be any class of graphs that is closed under clique replacement. The following statements hold:*

- *Suppose that, for any n-vertex graph $G \in \mathcal{G}$, we have $\chi_f(G) \leq \gamma(n)\omega(G)$. Then, for any $G \in \mathcal{G}$ and any weight function w, we have $\mathsf{LP}(G, w) \leq \gamma(N)\alpha(G, w)$ for some N. Moreover, given a fractional coloring with polynomial support, there is a $(1 + \epsilon)\gamma(N)$ approximation for MWIS via rounding (LP), for $N = O(n^2/\epsilon)$.*
- *Assume $\mathsf{LP}(G, w) \leq \gamma(n)\alpha(G, w)$ for all w. Then we have $\chi_f(G) \leq \gamma(n)\omega(G)$. Moreover, given a polynomial-time $\gamma(n)$-approximation LP rounding algorithm for MWIS, we can efficiently compute a fractional coloring using at most* $(1 +$

$\epsilon)\gamma(n)\omega(G)$ colors for any $\epsilon > 0$.

The gap between $\chi(G)$ and $\omega(G)$ has received a lot of attention in the context of intersection graphs. In particular, many old problems in mathematics are related to χ-boundedness[3] of intersection graphs (see for instance [2,5] and references therein). We hope that this work will encourage the study of $\chi_f(G)/\omega(G)$. Our results have many immediate applications, giving both new algorithmic and integrality gap results. Due to the space limit, we omit the applications.

2 The Equivalence

Fractional Coloring $\Longrightarrow$ LP Gap: Consider any graph $G = (V, E)$, $n = |V|$, and $G \in \mathcal{G}$. We will show that $\alpha(G, w) \geq \mathsf{LP}(G, w)/\gamma(n)$.

Let x be an optimal LP solution for (LP). First, assume that x_v is in an integral multiple of $1/q$ for some integers q. By standard LP theory, this is possible. Let $x_v = q_v/q$. We create a graph G' from G as follows: For each vertex $v \in V(G)$, perform a clique replacement operation on v by replacing v with a clique X_v of size q_v. Observe that $\omega(G') \leq q$: Let C' be a clique in G'. Consider the set $C = \{v \in V(G) : X_v \cap C' \neq \emptyset\}$. The LP constraint guarantees that $\sum_{v \in C} x_v \leq 1$ and therefore $|C'| \leq \sum_{v \in C} |X_v| = \sum_{v \in C} q_v \leq q$.

Since $\mathcal{G}$ is closed under clique replacement operation, we have $G' \in \mathcal{G}$ and that $\chi_f(G') \leq \gamma(N)q$. Let σ be an optimal fractional coloring of G'. We sample an independent set J where each $J \subseteq V(G')$ is sampled with probability $\sigma(J)/\chi_f(G)$. Therefore, each vertex $v \in V(G')$ is sampled with probability $\sum_{I:v \in I} \sigma(I) \geq 1/\chi_f(G')$. So we get an independent set J: $\mathbb{E}[w(J)] = \sum_{v \in V(G')} w(v)Pr[v \in J] \geq \frac{1}{\chi_f(G')} \sum_{v \in V(G')} w(v)$. This is at least $\frac{1}{\gamma(N)q} \sum_{v \in V(G)} w(v)q_v = \mathsf{LP}(G, w)/\gamma(N)$.

This concludes the proof. Remark that N can be very large compared to n, but this does not affect the ratio if γ is a constant function. If γ is not a constant function, we can reduce the value of N to $O(n^2/\epsilon)$, while preserving the ratio within a factor of $(1 + \epsilon)$. The proof is omitted, due to space limitation.

LP Gap $\Longrightarrow$ Coloring: Let G be a graph on n vertices. If $\mathsf{gap}(G, w) \leq \gamma(n)$ for all weight vectors w, then $\chi_f(G) \leq \gamma(n)\omega(G)$. Moreover, we show how to compute a fractional coloring using at most $(1 + \epsilon)\gamma(n)\omega(G)$ colors for any $\epsilon > 0$.

The following linear constraints check whether the graph is $1/\eta$-colorable.

$$\text{(P)} \quad \sum_{I:v \in I} \sigma(I) \geq \eta \text{ for all } v \in V(G)$$

$$\sum_{I} \sigma(I) \leq 1$$

[3] A graph is χ-bounded if $\chi(G) \leq f(\omega(G))$ for some function f.

Our goal is to find a feasible solution σ that satisfies every constraint. Applying a standard multiplicative weight framework, our algorithm does the following steps:

(i) Start with initial weight function $w^{(1)}$ where $w_v^{(1)} = 1$ for all v.

(ii) In iteration t, compute a solution $\sigma^{(t)}$ that satisfies the "weighted average constraint" $\sum_v w_v^{(t)}(\sum_{I:v\in I}\sigma(I) - \eta) \geq 0$.

(iii) Update the weight $w^{(t)}$ to $w^{(t+1)}$. Then return to Step (ii).

Theorem 2.1 *[1] There is an update strategy such that, after T rounds, solution $\sigma = \frac{1}{T}\sum_{t=1}^{T}\sigma^{(t)}$ $(1-\epsilon)$-satisfies all constraints, i.e. for all v, $\sum_{I:v\in I}\sigma(I) \geq (1-\epsilon)\eta$.*

It only remains to show that we can compute a solution that satisfies the "weighted average constraint", which means finding I with $w(I) \geq \eta\, w^{(t)}(V)$ on $(G, w^{(t)})$. Consider the linear program for MWIS, (LP), using weights $\{w_v^{(t)}\}_{v\in V}$. We obtain a fractional solution x with weight $\frac{1}{\omega(G)}\sum_{v\in V} w_v^{(t)}$ by setting $x_v = 1/\omega(G)$ for all $v \in V$. Since $\sum_{v\in C} x_v = |C|/|\omega(G)| \leq 1$, for every clique C, it is clear that this is a solution to the LP. This implies that there is an integer solution with weight $\frac{1}{\gamma(n)\omega(G)}\sum_{v\in V} w_v^{(t)} = \eta\sum_{v\in V} w_v^{(t)}$, that is, there is an independent set I' with the desired weight. Furthermore, we use a $\gamma(n)$-approximation LP rounding algorithm to find I' of total weight $\frac{1}{\gamma(n)}\sum_v w_v^{(t)} x_v = w^{(t)}(V)/\omega(G)\gamma(n)$.

References

[1] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(6):121–164, 2012.

[2] Edgar Asplund and Branko Grünbaum. On a coloring problem. *Mathematica Scandinavica*, 8:181–188, 1960.

[3] Parinya Chalermsook and Julia Chuzhoy. Maximum independent set of rectangles. In *Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009, New York, NY, USA, January 4-6, 2009*, pages 892–901, 2009.

[4] Timothy M. Chan and Sariel Har-Peled. Approximation algorithms for maximum independent set of pseudo-disks. *Discrete & Computational Geometry*, 48(2):373–392, 2012.

[5] András Gyárfás. On the chromatic number of multiple interval graphs and overlap graphs. *Discrete mathematics*, 55(2):161–166, 1985.

[6] László Lovász. On the shannon capacity of a graph. *Information Theory, IEEE Transactions on*, 25(1):1–7, 1979.

List Coloring of Planar Graphs with Forbidden Cycles

Sreekanth Gorla, Meghana Nasre

Indian Institute of Technology, Madras, India.
{gsrikanth,meghana}@cse.iitm.ac.in

Abstract

We consider list coloring of planar graphs without cycles of length in $\{4, \ldots, 8\}$. List coloring is a generalization of the classical vertex coloring problem where each vertex has a list of colors associated with it. The goal is to proper vertex color the graph, such that each vertex gets a color available in its list. In this note, we prove that it is possible to 3-list color planar graphs without cycles of length in $\{4, \ldots, 8\}$ and with restrictions on 9-cycles.

Keywords: planar graphs; list coloring; discharging method

1 Introduction

Coloring is possibly one of the most well-studied problems in graph theory. A generalization of the coloring problem is *list coloring* where the input is a graph G and associated with every vertex is a list $l(v)$ of distinct colors. It is common to assume that the lists associated with all vertices are of the same size. The goal is to proper vertex color the graph such that every vertex gets a color from its list. The choosability number of a graph denoted by $\chi_l(G)$ is the minimum size of the lists such that with any assignment of colors to the lists, it is possible to get a proper vertex coloring of G. Since all the lists can be same, it is clear that the chromatic number of G denoted by $\chi(G) \leq \chi_l(G)$.

Coloring and list coloring have been extensively investigated on planar graphs. It was conjectured that for a planar graph $\chi_l(G) \leq \chi(G)+1$. Supporting this, Alon and Tarsi [1] proved that planar bipartite graphs are 3-choosable. Thomassen [6] proved that planar graphs are 5-choosable. However, Voigt [7] constructed a 3-colorable graph which is not 4-choosable. In this paper, we consider planar graphs with forbidden cycles.

Salvatipour [5] and Borodin et al.[2] in a series of results have proved that planar graphs without $\{4, \ldots, 7\}$ cycles are 3-colorable. However, in terms of choosability, the current best known result is by Borodin [3] where he shows that planar graphs without $\{4, \ldots, 9\}$ cycles are 3-choosable. In the same paper, Borodin also posed the open question of proving that planar graphs without cycles of length in $\{4, \ldots, 8\}$ are 3-choosable. In this note, we make partial progress towards this question. Our main result is stated as Theorem 1.1 which we prove using the discharging method. At the heart of the proof are reducible configurations with respect to 3-choosability, which may be of independent interest.

Theorem 1.1 *Let G be a planar graph. If in G,*

(i) *there is no cycle of length in $\{4, \ldots, 8\}$ and*

(ii) *9-cycles do not have vertices of degree 4 or 5 on its boundary,*

then G is 3-choosable.

Preliminaries: All graphs considered here are planar, simple, and connected. For a graph G, the sets $V(G), E(G), F(G)$ denote the set of vertices, edges, and faces of G respectively. Let d_v denote the degree of vertex v and l_f denote the length of a face f. The chromatic number and choosability number of G are denoted by $\chi(G)$ and $\chi_l(G)$ respectively. Recall Euler's formula which holds for planar graphs, $V(G) + F(G) = E(G) + 2$.

Our proof is using the discharging method which is most popularly known for the proof of the four color theorem. The main ingredients of any proof by discharging method is a set of *reducible configurations, charging rules*, and *discharging rules*. These terms will be clarified in the proof of Theorem 1.1.

2 Proof of Theorem 1.1

In order to prove the theorem, we first list the set of reducible configurations. A reducible configuration is a subgraph that cannot occur in a minimal counterexample for the desired property (in this case, Theorem 1.1).

(R_1) : *Vertices of degree less than 3.*

(R_2) : *Induced even cycle with all degree-3 vertices.*

(R_3) : *9 cycle with all degree-3 vertices and adjacent to four triangles.*

To see that R_1 is reducible, assume that a minimal counterexample G contains a vertex v with $d_v < 3$. Consider the graph $H = G \setminus \{v\}$. By minimality of G, the graph H is 3-choosable. Recall that each vertex has lists of size 3 associated with it. Thus, it is possible to extend the coloring of H and assign a color to v from $l(v)$ such that G is 3-choosable. To see that R_2 is 3-choosable, we employ a similar argument and the fact that every even cycle is 2-choosable [4]. Proving R_3 is reducible is significantly non-trivial and we show it using Theorem 2.1.

Assuming these reducible configurations, we prove Theorem 1.1. We show that any planar graph satisfying the conditions of Theorem 1.1 has one of the reducible configurations. If not, then there exists a minimal counterexample G. We now assign charge (see below) to the vertices and faces of G such that the total charge is negative. Using suitable discharging rules (see below) and the fact that G does not contain any reducible configuration as an induced subgraph, we show that the final charge is positive. This contradicts the existence of the minimal counterexample G.

Charging rules: For a vertex v of degree d_v, assign a charge of $d_v - 6$. For a face f of length l_f, assign a charge of $2l_f - 6$. With this charging scheme and using Euler's formula, it is easy to see that the total initial charge for any graph is -12. We also note that, all faces have non-negative charge. On the other hand, vertices of degree 3, 4, and 5 get negative charge. We call a degree-3 vertex as a type-1 vertex, if it belongs to two non-triangular faces and one triangular face. We call a degree-3 vertex as a type-2 vertex, if it has all non-triangular faces surrounding it.

Discharging rules: We set up the discharging rules such that every non-triangular face (having positive charge) gives excess charge to vertices which have negative charge. Every non-triangular face f gives:

(i) a charge of $\frac{3}{2}$ to type-1 degree-3 vertices on its boundary.

(ii) a charge of 1 to type-2 degree-3 vertices on its boundary.

(iii) a charge of $\frac{1}{2}$ to each degree-4 vertex v on its boundary, if either (a) v has exactly one triangle incident on it and that triangle shares an edge with f or (b) v has no triangle incident on it.

(iv) a charge of 1 to all degree-4 vertices on its boundary not satisfying (iii).

(v) a charge of $\frac{1}{3}$ to all degree-5 vertices on its boundary.

As mentioned earlier these charging, discharging rules and the reducible configurations allow us to contradict the existence of a minimal counterexample, or in fact any counterexample. Thus, this proves the theorem.

Theorem 2.1 *A 9 cycle with all degree-3 vertices and adjacent to four triangles is reducible with respect to Theorem 1.1.*

Proof. (*sketch*) Let H denote a 9 cycle as mentioned above, see Figure 1(i). If H occurs in any minimal counterexample G, we remove v_1 (degree 3 vertex of H not a part of any triangle) from G. By minimality of G, $G \setminus \{v_1\}$ has a 3-list coloring say ϕ. We prove that, either ϕ can be extended to G or the colors on vertices in H form two equivalence classes. That is, for each $i = 1, \ldots, 9$, vertex v_i has a $l(v_i) = \{c_1, c_2, x(j)\}$, where $x(j)$ denotes the color that ϕ assigned to w_j $(j = 1, \ldots, 5)$, the neighbour of v_i not in the cycle. If the lists are of this type and we are not able to complete the coloring, it must be the case that for each $j = 1, \ldots, 5$, w_j is assigned $x(j)$. We then introduce a suitable gadget to show that this bad case cannot happen and therefore, we can always extend the 3-list coloring to G. □

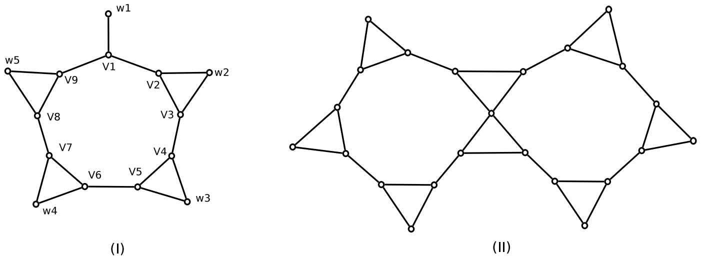


Fig. 1. Reducible configurations

Discussion: We remark that in addition to the reducible configurations mentioned earlier, we can also prove that a particular case of two 9-cycles sharing a degree-4 vertex (see Figure 1(ii)) is also reducible. We believe that these reducible configurations are a useful contribution which may help in completing the proof of Borodin's open question.

References

[1] N. Alon and M. Tarsi, *Colorings and orientations of graphs*, Combinatorica, 12 (1992), pp. 125–134.

[2] O. Borodin, A. Glebov, A. Raspaud, and M. Salavatipour, *Planar graphs without cycles of length from 4 to 7 are 3-colorable*, Journal of Combinatorial Theory, Series B, 93 (2005), pp. 303 – 311.

[3] O. V. Borodin, *Structural properties of plane graphs without adjacent triangles and an application to 3-colorings*, Journal of Graph theory, (1996), pp. 183 – 186.

[4] P. Erdős, A. L. Rubin, and H. Taylor, *Choosability in graphs. (in english)*, Combinatorics, graph theory and computing, (1980), pp. 125–157.

[5] M. R. Salavatipour, *Graph Colouring via the Discharging Method*, PhD thesis, Toronto, Ont., Canada, Canada, 2003.

[6] C. Thomassen, *Every planar graph is 5-choosable*, Journal of Combinatorial Theory, Series B, 62 (1994), pp. 180 – 181.

[7] M. Voigt and B. Wirth, *On 3-colorable non-4-choosable planar graphs*, Journal of Graph Theory, 24 (1997), pp. 233–235.

Subset matching and edge coloring in bipartite graphs

Ömer Can Yavuzyılmaz*, Enver Kayaaslan

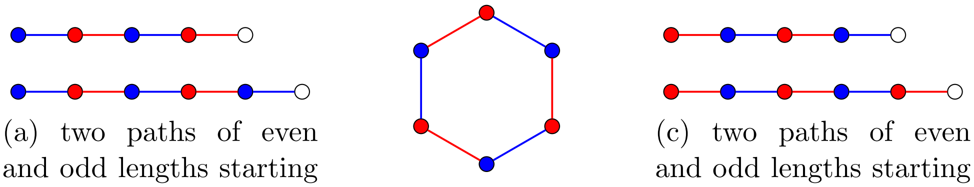
Turgut Ozal University, Ankara, Turkey
omercanyy@gmail.com, ekayaaslan@turgutozal.edu.tr

Abstract

The focus of this paper is on finding a matching in a bipartite graph such that a given subset of vertices are matched. This is called subset matching and generalizes perfect matchings. We prove a necessary and sufficient condition for the existence of a subset matching in bipartite graphs. The proof is algorithmic and based on combination of two matchings. Remarkably, the necessary and sufficient condition always holds when the subset is composed of the vertices with maximum degree. This in turn leads to a simple algorithm that finds an optimal edge coloring in bipartite graphs with no need to transform the bipartite graph into a regular one.

Keywords: bipartite graph, matching, edge coloring.

Bipartite graphs represent the class of (undirected) graphs without odd cycles, and a matching is a subgraph in which every vertex is incident to at most one edge. Throughout the presentation of our findings, we assume that the reader is familiar with bipartite graphs and matchings [10]. A closely related study is by Alon and Yuster [2], which introduces maximum subset matching in general graphs and gives an approximation algorithm. In our study, we are interested in subset matching in bipartite graphs, where a given subset of vertices are all matched. Notice that a subset matching is, in fact, a perfect matching for the bipartite graph when the subset contains all vertices.



(a) two paths of even and odd lengths starting from the blue vertices.

(b) a cycle.

(c) two paths of even and odd lengths starting from the red vertices.

Fig. 1. Possible components of the union graph of two matchings.

Problem 1 Subset matching in bipartite graphs *Given a bipartite graph $\mathcal{B}(\mathcal{V}_1 \cup \mathcal{V}_2, \mathcal{E})$ and a subset $S \subset \mathcal{V}_1 \cup \mathcal{V}_2$ of vertices, a matching $\mathcal{M}$ is called a subset matching for S if there is an edge $\{u, v\} \in \mathcal{M}$ for every $v \in S$.*

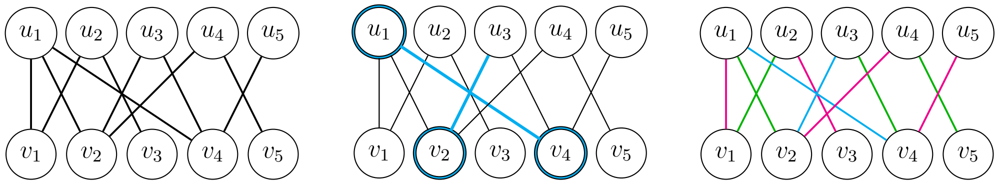
The following theorem presents a necessary and sufficient condition for a bipartite graph to have a subset matching for a given subset of vertices.

Theorem 1 Necessary and sufficient condition *Given a bipartite graph $\mathcal{B}(\mathcal{V}_1 \cup \mathcal{V}_2, \mathcal{E})$ and $S \subset \mathcal{V}_1 \cup \mathcal{V}_2$, there is a subset matching $\mathcal{M}$ for S if and only if $|X_i| \leq |\mathcal{N}(X_i)|$ for any $X_i \subset S \cap \mathcal{V}_i$, for each $i = 1, 2$.*

Proof. Hall's Theorem [6] implies that there exists a matching that covers $S \cap \mathcal{V}_i$ if and only if $|X_i| \leq |\mathcal{N}(X_i)|$ for any $X_i \subset S \cap \mathcal{V}_i$, for each $i = 1, 2$.

Assume that there is a subset matching $\mathcal{M}$ for S. Then, $\mathcal{M}$ covers both $S \cap \mathcal{V}_1$ and $S \cap \mathcal{V}_2$, and due to Hall's Theorem, $|X_i| \leq |\mathcal{N}(X_i)|$ for any $X_i \subset S \cap \mathcal{V}_i$, for each $i = 1, 2$.

Assume that $|X_i| \leq |\mathcal{N}(X_i)|$ for any $X_i \subset S \cap \mathcal{V}_i$, for each $i = 1, 2$. By Hall's Theorem, there are two matchings $\mathcal{M}_1$ and $\mathcal{M}_2$ that cover $S \cap \mathcal{V}_1$ and $S \cap \mathcal{V}_2$, respectively. Now, we show that there is a subset matching $\mathcal{M}$ for S. Consider the union graph $\mathcal{M}_1 \cup \mathcal{M}_2$, which is also a subgraph of the bipartite graph. We categorize vertices into three groups: the vertices of $S \cap \mathcal{V}_1$ (shown as blue circles in Fig. 1), the vertices of $S \cap \mathcal{V}_2$ (red circles), and the vertices of $\mathcal{V}_1 \cup \mathcal{V}_2 - S$ (empty circles). Notice that any vertex in the union graph is either isolated, or incident to exactly one edge, or incident to exactly two edges (one from $\mathcal{M}_1$ and the other from $\mathcal{M}_2$). Thus a connected component of the union graph is either a path, or a cycle of even length [9,11,4], as depicted in Fig. 1, where the blue and red edges represent edges of $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively. For a path that starts from a (blue) vertex in $S \cap \mathcal{V}_1$ (Fig. 1a), we pick the corresponding (blue) edges of $\mathcal{M}_1$ to be included in $\mathcal{M}$. For a cycle (Fig. 1b), we pick either the corresponding (blue) edges of $\mathcal{M}_1$ or (red) edges of $\mathcal{M}_2$ for $\mathcal{M}$. Finally, for a path that starts from a (red) vertex in $S \cap \mathcal{V}_2$ (Fig. 1c), we pick the corresponding (red) edges of $\mathcal{M}_2$ to be included in $\mathcal{M}$. Then, $\mathcal{M}$ is a matching and covers all vertices of S. Thus, $\mathcal{M}$ is a subset matching for S.□



u_1 u_2 u_3 u_4 u_5
v_1 v_2 v_3 v_4 v_5
u_1 u_2 u_3 u_4 u_5
v_1 v_2 v_3 v_4 v_5
u_1 u_2 u_3 u_4 u_5
v_1 v_2 v_3 v_4 v_5

Fig. 2. A sample bipartite graph with maximum degree three (left), a subset matching for the maximum degree vertices (middle), an optimal edge coloring (right).

The following theorem relates the edge coloring problem in bipartite graphs to subset matching for the maximum degree vertices.

Theorem 2 Subset matching for maximum degree vertices *Given a bipartite graph $\mathcal{B}(\mathcal{V}_1 \cup \mathcal{V}_2, \mathcal{E})$, there is a subset matching $\mathcal{M}_\Delta$ for $S_\Delta = \{v \in \mathcal{V}_1 \cup \mathcal{V}_2 : deg(v) = \Delta\}$, where Δ refers to the maximum degree.*

Proof. We show that $|X_i| \leq |\mathcal{N}(X_i)|$ for any $X_i \subset S_\Delta \cap \mathcal{V}_i$, for each $i = 1, 2$. Without loss of generality, take any $X_1 \subset S_\Delta \cap \mathcal{V}_1$. For the sake of contradiction, suppose $|X_1| > |\mathcal{N}(X_1)|$. Consider the edges incident to any of the vertices in X_1. There are $\Delta |X_1|$ such edges as $deg(v) = \Delta$ for each $v \in X_1$. Notice that each of those edges is also incident to a vertex in $\mathcal{N}(X_1)$. Since we suppose $|X_1| > |\mathcal{N}(X_1)|$, there is a vertex $w \in \mathcal{N}(X_1)$ such that $deg(w) > \Delta$, by the pigeonhole principle. This contradicts with that Δ is maximum. □

The edge chromatic number of a bipartite graph equals to the maximum degree Δ, and this fact dates back to König [8]. In the literature, there are a number of algorithms for finding an optimal edge coloring of a bipartite graph [5,7,3,12,1]. However, those algorithms are, in general, based on edge coloring of regular bipartite graphs.

The proof of Theorem 1 is algorithmic, and thus, we can find a subset matching, if exists, using bipartite graph matching algorithms as a subroutine, and a subset matching exists for the maximum degree vertices, due to Theorem 2. This leads to an algorithm for finding an optimal edge coloring of bipartite graphs without introducing new vertices or edges. Our algorithm takes Δ steps, and initially, we consider the bipartite graph $\mathcal{B}_1 = \mathcal{B}$. At each step $k < \Delta$, the algorithm finds a subset matching $\mathcal{M}_k$ for the maximum degree vertices in $\mathcal{B}_k$, assigns a color c_k to those edges of $\mathcal{M}_k$, and, for the next step, considers the bipartite graph $\mathcal{B}_{k+1} = \mathcal{B}_k - \mathcal{M}_1$. Notice that the maximum degree decreases exactly by one after each step. At the final step, the bipartite graph $\mathcal{B}_\Delta$ is itself a matching since the maximum degree becomes one. Fig. 2 illustrates a bipartite graph with maximum degree $\Delta = 3$, a subset matching for $\{u_1, v_2, v_4\}$, which are the vertices of maximum degree, and an edge coloring with three colors obtained by our algorithm. Note that our algorithm computes an optimal edge coloring also for bipartite multigraphs, if we identify the maximum degree vertices by taking multiple edges into account, and consider multiple edges as a single one when finding a subset matching.

References

[1] Noga Alon. A simple algorithm for edge-coloring bipartite multigraphs. *Information Processing Letters*, 85(6):301–302, 2003.

[2] Noga Alon and Raphael Yuster. Fast algorithms for maximum subset matching and all-pairs shortest paths in graphs with a (not so) small vertex cover. In *Algorithms–ESA 2007*, pages 175–186. Springer, 2007.

[3] Richard Cole, Kirstin Ost, and Stefan Schirra. Edge-coloring bipartite multigraphs in O(E log D) time. *Combinatorica*, 21(1):5–12, 2001.

[4] Fanny Dufossé, Kamer Kaya, and Bora Uçar. Bipartite matching heuristics with quality guarantees on shared memory parallel computers. In *Parallel and Distributed Processing Symposium, 2014 IEEE 28th International*, pages 540–549. IEEE, 2014.

[5] Harold N Gabow and Oded Kariv. Algorithms for edge coloring bipartite graphs and multigraphs. *SIAM Journal on Computing*, 11(1):117–129, 1982.

[6] Philip Hall. On representatives of subsets. *J. London Math. Soc*, 10(1):26–30, 1935.

[7] Ajai Kapoor and Romeo Rizzi. Edge-coloring bipartite graphs. *Journal of Algorithms*, 34(2):390–396, 2000.

[8] Dénes König. Graphok és alkalmazásuk a determinánsok és a halmazok elméletére. *Mathematikai és Természettudományi Ertesito*, 34:104–119, 1916.

[9] Eugene L Lawler. *Combinatorial optimization: networks and matroids.* Courier Corporation, 2001.

[10] László Lovász and Michael D Plummer. *Matching theory*, volume 367. American Mathematical Soc., 2009.

[11] Fredrik Manne and Mahantesh Halappanavar. New effective multithreaded matching algorithms. In *Parallel and Distributed Processing Symposium, 2014 IEEE 28th International*, pages 519–528. IEEE, 2014.

[12] Alexander Schrijver. Bipartite edge coloring in O(Δm) time. *SIAM Journal on Computing*, 28(3):841–846, 1998.

On the chromatic number of $(P_5, K_{2,t})$-free graphs [⋆]

Christoph Brause[a], Trung Duy Doan[a,b], Ingo Schiermeyer[a]

[a] *Institute of Discrete Mathematics and Algebra*
TU Bergakademie Freiberg
Freiberg, Germany
brause@math.tu-freiberg.de, ingo.schiermeyer@tu-freiberg.de

[b] *School of Applied Mathematics and Informatics*
Hanoi University of Science and Technology
Hanoi, Vietnam
trungdoanduy@gmail.com

Abstract

In this paper we study the chromatic number of $(P_5, K_{2,t})$-free graphs with $t \geq 2$. It is still an open question whether there are polynomial (χ-binding) functions f_k for $k \geq 5$ such that every P_k-free graph G satisfies $\chi(G) \leq f_k(\omega(G))$, where P_k is an induced path on k vertices. Our main result is that every $(P_5, K_{2,t})$-free graph G admits a polynomial χ-binding function. Moreover, we will present polynomial χ-binding functions for several other subclasses of P_5-free graphs.

Keywords: chromatic number, χ-binding function, P_5-free graphs

[⋆] Part of this research has been financially supported by the DAAD PPP-project 57210296 Freiberg-Pilsen. Financial support by the Free State of Saxony (Landesgraduiertenstipendium of Trung Duy Doan) is thankfully acknowledged.

1 Introduction

We consider finite, simple, and undirected graphs, and use standard terminology and notation.

Let G be a graph. An *induced subgraph* of G is a graph H such that $V(H) \subseteq V(G)$, and $uv \in E(H)$ if and only if $uv \in E(G)$ for all $u, v \in V(H)$. Given graphs G and F we say that G *contains* F if F is isomorphic to an induced subgraph of G. We say that a graph G is F-free, if it does not contain F. For two graphs G, H we denote by $G + H$ the disjoint union and by $G \vee H$ the join of G and H, respectively.

A graph G is called *k-colourable*, if its vertices can be coloured with k colours so that adjacent vertices obtain distinct colours. The smallest k such that a given graph G is k-colourable is called its *chromatic number*, denoted by $\chi(G)$. It is well-known that $\omega(G) \leq \chi(G) \leq \Delta(G) + 1$ for any graph G, where $\omega(G)$ denotes its clique number and $\Delta(G)$ its maximum degree. A graph G is *perfect* if $\chi(H) = \omega(H)$ for every induced subgraph H of G.

A family $\mathcal{G}$ of graphs is called χ-bound with binding function f if $\chi(G') \leq f(\omega(G'))$ holds whenever $G \in \mathcal{G}$ and G' is an induced subgraph of G. For a fixed graph H let $\mathcal{G}(H)$ denote the family of graphs which are H-free. The following theorems are well known in chromatic graph theory.

Theorem 1.1 (Erdős [4]) *For any positive integers $k, l \geq 3$ there exists a graph G with girth $g(G) \geq l$ and chromatic number $\chi(G) \geq k$.*

Theorem 1.2 (The Strong Perfect Graph Theorem [3]) *A graph is perfect if and only if it contains neither an odd cycle of length at least five nor its complement.*

In this paper we study the chromatic number of P_5-free graphs. Our work was motivated by the following conjecture of Gyárfás.

Conjecture 1.3 (Gyárfás' conjecture [6])) *Let T be any tree (or forest). Then there is a function f_T such that every T-free graph G satisfies $\chi(G) \leq f_T(\omega(G))$.*

Gyárfás [6] proved this conjecture when T is a path P_k for all $k \geq 4$ by showing $f_{P_k}(\omega) \leq (k-1)^{\omega(G)-1}$. One may wonder whether this exponential bound can be improved. In particular:

Question: Are there polynomial (χ-binding) functions f_k for $k \geq 5$ such that every P_k-free graph G satisfies $\chi(G) \leq f_k(\omega(G))$?

2 Known results for P_5-free graphs

The following results have been shown for P_5-free graphs.

Theorem 2.1 ([5]) *Let G be a connected $(P_5, House)$-free graph of order n and clique number $\omega(G)$. Then $\chi(G) \leq \binom{\omega(G)+1}{2}$.*

Theorem 2.2 ([2]) *Let G be a connected (P_5, Gem)-free graph of order n and clique number $\omega(G)$. Then $\chi(G) \leq 6\omega(G)$.*

Corollary 2.3 *Let G be a connected (P_5, H)-free graph of order n and clique number $\omega(G)$, where $H \in \{Paw, Diamond\}$. Then $\chi(G) \leq 6\omega(G)$.*

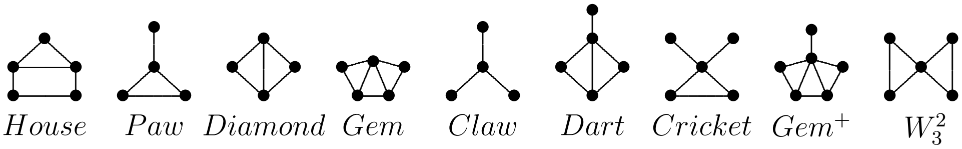


Fig. 1. The graphs *House*, *Paw*, *Diamond*, *Gem*, *Claw*, *Dart*, *Cricket*, Gem^+, and *Windmill* W_3^2.

In [7] the subgraph *Gem* was replaced by the supergraph $Gem^+ = K_1 \vee (K_1 + P_4)$.

Theorem 2.4 *Let G be a (P_5, Gem^+)-free graph of order n and clique number $\omega(G)$. Then $\chi(G) \leq \omega^2(G)$.*

Corollary 2.5 *Let G be a connected (P_5, H)-free graph of order n and clique number $\omega(G)$, where $H \in \{Claw, Dart, Cricket\}$. Then $\chi(G) \leq \omega^2(G)$.*

For integers $r, p \geq 2$ the *windmill graph* $W_{r+1}^p = K_1 \vee pK_r$ is the graph obtained by joining a single vertex (the center) to the vertices of p disjoint copies of a complete graph K_r (the Windmill W_3^2 is shown in Figure 2).

Theorem 2.6 ([8]) *Let G be a (P_5, W_{r+1}^p)-free graph for some $r, p \geq 2$. Then $\chi(G) \leq c(p, r) \cdot \omega^{(p-1)r+1}$ for a constant $c(p, r)$.*

3 Main results

Our first main result is the following:

Theorem 3.1 *Let G be a $(P_5, K_{2,t})$-free graph for some $t \geq 2$. Then $\chi(G) \leq c_t \cdot \omega^t$ for a constant c_t.*

Sketch of proof: By the Strong Perfect Graph Theorem every non perfect $(P_5, K_{2,t})$-free graph contains an induced C_5 or an induced $\overline{C}_{2p+1}$ for some

$3 \leq p \leq \omega$. We first consider the neighbourhood $N(C_5)$ of the C_5, which can be partitioned into 21 distinct subsets depending on the neighbours on an induced C_5. Using the Ramsey number $R(K_\omega, K_t)$ we manage to find an induced $K_{2,t}$ or to show that $\chi(G) \leq c(n_1) \cdot \omega^t$. Next we consider the neighbourhood of an induced $\overline{C}_{2p+1}$ and proceed in a similar way. □

Next we make use of a structural result for connected P_5-free graphs.

Theorem 3.2 (Bacsó and Tuza[1]) *Every connected P_5-free graph contains a dominating clique or a dominating P_3.*

This admits the following result for P_5-free graphs (cf. [7]).

Theorem 3.3 *Let H be a graph such that $\mathcal{G}(H)$ has an $O(\omega^s)$ χ-binding function for some $s \geq 1$, and let G be a connected $(P_5, K_1 \vee H)$-free graph with clique number $\omega(G)$. Then G has an $O(\omega^{s+1})$ χ-binding function.*

So we can apply Theorem 3.3 to obtain the following result for $(P_5, K_{2,t})$-free graphs.

Theorem 3.4 *Let G be a $(P_5, K_p \vee K_{2,t})$-free graph for some $p \geq 1, t \geq 2$. Then $\chi(G) \leq c(p,t) \cdot \omega^{t+p}$ for a constant $c(p,t)$.*

References

[1] G. Bacsó and Zs. Tuza, Dominating cliques in P_5-free graphs, **Period. Math. Hungar.** 21 (3) (1990) 303–308.

[2] S. A. Choudum, T. Karthick, and M. A. Shalu, Perfect Coloring and Linearly χ-Bound P_6-free Graphs, **J. Graph Theory** 54 (4) (2006) 293–306.

[3] M. Chudnovsky, N. Robertson, P. Seymour, and R.Thomas, The Strong Perfect Graph Theorem, **Ann. of Math.** 164 (2006) 51–229.

[4] P. Erdős, Graph theory and probability, **Canad. J. Math.** 11 (1959), 3438.

[5] J. L. Fouquet, V. Giakoumakis, F. Maire, and H. Thuillier, On graphs without P_5 and $\overline{P}_5$, **Discrete Math.** 146 (1995) 33–44.

[6] A. Gyárfás, Problems from the world surrounding perfect graphs. In **Proc. Int. Conf. on Comb. Analysis and Applications** (Pokrzywna, 1985), Zastos. Mat. 19 (1987) 413–441.

[7] I. Schiermeyer, Chromatic number of P_5-free graphs: Reed's conjecture, **Discrete Math.**, to appear.

[8] I. Schiermeyer, On the chromatic number of $(P_5, windmill)$-free graphs, Preprint 2015.

On Minimum Average Stretch Spanning Trees in Grid Graphs

Pankaj Pundir and G. Ramakrishna

Indian Institute of Information Technology Chittoor, Sri City, India
{pankaj.pundir,ramakrishna.g}@iiits.in

Abstract

A minimum average stretch spanning tree of a graph is a spanning tree that minimizes the average stretch. We present a necessary-and-sufficient condition for a spanning tree to be a minimum average stretch spanning tree (MAST) in grid graphs and provide a linear-time algorithm to construct an MAST.

Keywords: Minimum average stretch spanning tree, Grid graphs

1 Introduction

A *minimum average stretch spanning tree* (MAST) of a graph is a spanning tree that minimizes the average stretch and has various applications in network design and solving symmetric diagonally dominant linear systems [2]. Let $G = (V(G), E(G))$ be an unweighted graph and T be a spanning tree of G. For an edge $(u, v) \in E(G)$, $d_T(u, v)$ denotes the distance between u and v in T. For a *non-tree edge* $e = (u, v)$ with respect to T, i.e., $e \in E(G) \setminus E(T)$, stretch$(e)$ is $d_T(u, v)$. The average stretch of T is defined as

$$\mathrm{AvgStr}(T) = \frac{1}{|E(G)| - |E(T)|} \sum_{e \in E(G) \setminus E(T)} \mathrm{stretch}(e) \qquad (1)$$

Motivation and Our contribution. To the best of our knowledge, there is no necessary-and-sufficient condition for a spanning tree to be an MAST in grid graphs. Also, there is no published work to construct an MAST in any subclass of planar graphs except polygonal 2-trees. We state our main theorem, which is based on edge-missing property (defined later).

Theorem 1.1 *A spanning tree T of an unweighted grid graph G is an* MAST *if and only if T satisfies the edge-missing property. Further, an* MAST *of an unweighted grid graph can be constructed in linear time.*

We also derive an expression to find the number of MASTs in grid graphs.

Related work. The MAST problem is equivalent to the problem of finding a minimum fundamental cycle basis (MFCB) [2] and is shown as NP-complete in general graphs [1]. Recently, Reich has shown that MFCB is NP-complete in planar graphs as well [4]. In case of polygonal 2-trees, a subclass of planar graphs, Narayanaswamy et al. designed a polynomial-time algorithm for MAST [3]. The routing region in a microchip can be modeled as a grid graph. Also, combinatorial problems such as the existence of a Hamiltonian path between given two vertices and various routing problems are studied in grid graphs.

2 Structural Properties of MAST

We first introduce the necessary terminology on grid graphs. Then, we provide proofs for necessary and sufficient conditions in Lemma 2.2 and Lemma 2.4, respectively, and these lemmas leads to our main result.

Grid Graphs Preliminaries. For a positive integer n, $[n]$ denotes the set $\{1, \ldots, n\}$. A graph G is a *grid graph* if for some fixed $m \geq 1$ and $n \geq 1$, $V(G) = [m] \times [n]$, and $E(G) = \{(p_1, p_2) \mid p_1 = (x_1, y_1), p_2 = (x_2, y_2) \in V(G), |x_1 - x_2| + |y_1 - y_2| = 1\}$. Let G be a grid graph. The bounded regions defined by the planar embedding of G are *cells* of G. For each cell C, $V(C)$ and $E(C)$ denote the set of vertices and edges of C, respectively and $|V(C)| = |E(C)| = 4$. For a set $X \subseteq V(G)$, $G[X]$ denotes the induced subgraph on vertices in X. An edge $e \in E(G)$ is *external* if e appears exactly in one cell, otherwise it is *internal*. A cell C is *E-cell* if and only if C has at least one external edge. The vertices appear on the exterior boundary of G are *external* and the remaining vertices are *internal*. Let G' be the graph obtained from G after the removal of external vertices in G. If all the cells in G are E-cells, then the number of *layers* in G is one; Otherwise, the number of layers in G is defined as one plus the number of layers in G'. A grid graph on ℓ layers is denoted by G_ℓ. From a grid graph G_i on i layers, the grid graph

G_{i-1} on $i-1$ layers can be obtained by removing the external vertices in G_i.

Consider a spanning tree T of G_ℓ. For $i \in [\ell]$, an E-cell C in G_i is said to satisfy *edge-missing* property, with respect to T, if exactly one external edge among all the external edges in C does not appear in T. T is said to satisfy *edge-missing* property, if for each $1 \leq i \leq \ell$, each E-cell in G_i satisfies the edge-missing property. T is said to satisfy *external-edge-missing* property, if each E-cell in G_ℓ satisfies edge-missing property. A *fundamental cycle* with respect to T is a cycle formed by a non-tree edge (u, v), and the unique path between u and v in T, where $(u, v) \in E(G) \setminus E(T)$. A fundamental cycle C is an *external fundamental cycle* if the associated non-tree edge in C is external.

For the rest of the paper, G denotes a grid graph and G_ℓ denotes a grid graph on ℓ layers. Due to the page limit, proofs of few lemmas are omitted.

Lemma 2.1 *Let T be a spanning tree of G_ℓ. (a) If $\ell = 1$, then T is an* MAST *if and only if T satisfies external-edge-missing property. (b) If T is an* MAST*, then T satisfies external-edge-missing property.*

The *distance* between two cells C_u and C_v in a grid graph G is the distance between two vertices u and v in $\tilde{G}$, where $\tilde{G}$ is the dual graph of G, u and v are the dual vertices corresponding to C_u and C_v, respectively.

Lemma 2.2 *Let T be an* MAST *of G_ℓ. Then T satisfies edge-missing property.*

Proof. (**Sketch.**) By Lemma 2.1.(b), from every E-cell of G_ℓ exactly one external edge does not appear in T. Then, each E-cell in G_ℓ satisfies the edge-missing property. We assume that there is an E-cell C in G_i, $i \leq \ell - 1$, such that C violates the edge-missing property and for every k, $i+1 \leq k \leq \ell$, every E-cell in G_k, satisfies the edge-missing property. Let e be an external edge of G_i that appears in C. Let C' denote an E-cell in G_i, such that both C and C' appears in a same external fundamental cycle of T.

Since C violates the edge-missing property and T is a tree, an internal edge e' of G_i that appears in C is not available in T. Let $T' = T + e - e'$ be a new spanning tree obtained from T. We can observe that tot-Stretch(T) − tot-Stretch$(T') \geq 2r$, where r denotes the distance between C and C'. As $r \geq 1$, tot-Stretch$(T') <$ tot-Stretch(T) and thus T is not an MAST. Consequently, our assumption is incorrect, and hence the lemma is true. □

An edge $e \in E(G)$ is in *level* i if and only if e is an external edge in G_i. The stretch of a spanning tree T in level i is defined as the summation of stretches of the non-tree edges in level i, and is denoted by stretch$_i(T)$. A grid graph G is a *chain* if all the vertices in G are external. If a spanning tree T of G

satisfies edge-missing property, then we observe that for every fundamental cycle C with respect to T, $G[V(C)]$ is a chain. This observation helps to find the level-wise stretch in Lemma 2.3. Let n_i denote the number of E-cells in G_i. The parameter *base structure* value b of G, is defined as n_1 if G_1 is chain; Otherwise the value of b is 2.

Lemma 2.3 *Let T and T' be two spanning trees of G_ℓ that satisfy edge-missing property. Then for each $i \in [\ell]$, $\text{stretch}_i(T) = \text{stretch}_i(T')$. Further, $\text{stretch}_1(T) = 3n_1$, $\text{stretch}_2(T) = 5n_1 + 3(b+6)$, and for each $3 \leq i \leq \ell$, $\text{stretch}_i(T) = (2i+1)n_1 + (2(i-1)+1)(b+6) + 8((i-1)^2 - 1)$.*

Lemma 2.4 *Let T_ℓ be a spanning tree of G_ℓ. If T_ℓ satisfies edge-missing property, then T_ℓ is an* MAST.

Proof. Let T^* be an MAST of G. By Lemma 2.2, T^* satisfies edge-missing property. Also, T_ℓ satisfies edge-missing property. Then by applying Lemma 2.3, we observe that for each $i \in [\ell]$, $\text{stretch}_i(T_\ell) = \text{stretch}_i(T^*)$. Consequently, $\text{AvgStr}(T_\ell)$ is equal to $\text{AvgStr}(T^*)$, and thus T_ℓ is an MAST. □

The first part of Theorem 1.1 follows from Lemma 2.2 and Lemma 2.4.
MAST Construction. For each $1 \leq i \leq \ell$, for each E-cell C in G_i, remove an external edge of C, and from the first part of Theorem 1.1, the resultant graph is an MAST. Second part of Theorem 1.1 follows from this construction.

Theorem 2.5 *Let G_ℓ be a grid graph on ℓ layers, $\{C_1, \ldots, C_r\}$ be the set of E-cells in G_1, and b_i denotes the number of external edges in C_i. Then the number of* MASTs *in G_ℓ is $16^{\ell-1} \times b_1 \times \ldots \times b_r$.*

Remark. Characterizing a graph class in which an MAST is also a spanning tree whose maximum stretch is minimum, is an interesting open question.

References

[1] N. Deo, G. Prabhu, and M.S. Krishnamoorthy. Algorithms for generating fundamental cycles in a graph. *ACM Trans. on Math. Soft.*, 8:26–42, 1982.

[2] C. Liebchen and G. Wünsch. The zoo of tree spanner problems. *Discrete Appl. Math.*, 156:569–587, 2008.

[3] N.S. Narayanaswamy and G. Ramakrishna. On minimum average stretch spanning trees in polygonal 2-trees. *Theor. Comput. Sci.*, 575(C):56–70, 2015.

[4] A. Reich. Minimum strictly fundamental cycle bases of planar graphs are hard to find. *Discr. Appl. Math*, 205:150 – 159, 2016.

The Capacitated Budgeted Minimum Cost Flow Problem with Unit Upgrading Costs

Christina Büsing, Sarah Kirchner, Annika Thome [1]

Lehrstuhl II für Mathematatik, Lehrstuhl für Operations Research
RWTH Aachen University
D-52062 Aachen, Germany

Abstract

We consider a constraint minimum cost flow problem and show that it is in general $\mathcal{NP}$–complete. For special graph classes we give (pseudo–)polynomial algorithms.

Keywords: minimum cost flow, complexity, polynomial, pseudo–polynomial

1 Introduction and Problem Definition

Many polynomially solvable optimization problems become $\mathcal{NP}$-hard when resource constraints are added, e.g. minimum spanning tree or shortest path problems (see [1] and [4]). However if the resource consumption is of unit size these problems are solvable in polynomial time. This holds for the $\mathcal{NP}$-complete knapsack problem (see [5]). We consider a constraint minimum flow problem – the Budgeted Minimum Cost Flow Problem with resource limit K (BMCF(K)). We assume that each arc is associated with two costs: high cost $\bar{c}$ and low cost $\underline{c}$. The budget K now allows to pay the low instead of the high cost for flow on up to K arcs.

Definition 1.1 [**The BMCF(K) Problem**] Let $G = (V, A)$ be a digraph. Let s be the single supply vertex with $b(s) > 0$. Let $b(v) \in \mathbb{Z}_0^-$ be the demand of any $v \in V\backslash\{s\}$, let $\bar{c}(a)$, $\underline{c}(a)$ be upper and lower costs for all $a \in A$ such that there are no negative cost cycles, and let $u(a)$ be capacities for all $a \in A$.

[1] Email: `buesing@math2.rwth-aachen.de`, `{kirchner,thome}@or.rwth-aachen.de`

Finally, let $K \in \mathbb{N}$ be the budget parameter. A solution to the BMCF(K) problem is a b–flow f^* and a set $A_K^* \subseteq A$ with $|A_K^*| \leq K$. The objective is to find such a pair (f^*, A_K^*) that minimizes the cost function

$$c(f^*, A_K^*) = \sum_{a \in A_K^*} \underline{c}(a) \cdot f^*(a) + \sum_{a \in A \setminus A_K^*} \bar{c}(a) \cdot f^*(a)$$

The BMCF(K) problem is strongly $\mathcal{NP}$-complete for arbitrary resource consumption (see [3]). Even if the resource consumption is bounded to 1 per arc and no capacities are considered, the BMCF(K) problem is strongly $\mathcal{NP}$-complete (see [2]). Adding upper capacities this result transfers to the single source single sink BMCF(K) problem.

2 Results on Special Graph Classes

We introduce a polynomial algorithm for the BMCF(K) problem on cycles and pseudo-polynomial algorithms for parallel-arc trees and series-parallel graphs. We start with a first general observation: Let f be a flow. Then the *cost contribution* of an arc a is determined by $\text{con}_f(a) = (\bar{c}(a) - \underline{c}(a)) \cdot f(a)$. When all cost contributions are known and sorted non-decreasingly, the first K arcs are selected to be upgraded. This selection $A_K^*(f)$ of upgraded arcs is optimal for the given flow f as there are no arcs with higher cost contributions.

2.1 Cycles

A *cycle* $C = \{v_1, \ldots, v_n\}$ is a sequence of vertices where each two consecutive vertices are connected to each other by two antiparallel arcs (v_i, v_{i+1}) and (v_{i+1}, v_i) for all $i = 1, \ldots, n$ and $v_{n+1} = v_1$. We assume that $s = v_1$. In a given solution on a cycle C, there are three options to send flow to a specific demand vertex. The first two options are to send the entire flow clockwise or anticlockwise. The third one is to split the flow and send one portion clockwise and the rest anticlockwise. In case of the third option, we say that a demand vertex is a *split vertex*. We can show that there is always an optimal solution with at most one split vertex.

One further structural property of an optimal solution is that it can be transformed into a *neat* solution. A *neat* solution is characterized by a specific vertex v_i such that the demand of all demand vertices v_j with $j < i$ is satisfied by flow sent clockwise and demand of demand vertices v_j with $j > i$ is satisfied by flow sent anticlockwise. If there is a split vertex, then this vertex is the specific vertex v_i of a neat solution.

Exploiting the structure of a neat solution we propose a poynomial time algorithm to solve the BMCF(K) problem on cycles: We declare one of the demand vertices as the split vertex, say v_i and obtain a neat solution specified by v_i. There are two potentially optimal ways to send flow to satisfy the demand of v_i, i.e. two possible flows f_i^1 and f_i^2. We compute the cost $c(f_i^1, A_K^*(f_i^1))$ and $c(f_i^2, A_K^*(f_i^2))$. This is repeated for all demand vertices. An optimal solution to the BMCF(K) problem on a cycle is determined by choosing $(f^*, A_K^*) = \text{argmin}_{t_i \in V, j=1,2}\{c(f_i^j, A_K^*(f_i^j))\}$.

2.2 Parallel-Arc Trees

The capacitated BMCF(K) is easy on directed trees because the flow in a tree is unique. Given this unique flow we can obtain an optimal solution as described at the beginning of this section. We extend the defintion of directed trees and introduce the graph class of parallel-arc trees. A parallel-arc tree is a directed tree T on a macro level whose arcs represent batches of parallel arcs. In any BMCF(K) instance on a parallel-arc tree the supply vertex s is w.l.o.g. located at the root of the underlying tree. The amount of flow d_i that is sent through each batch i is fixed as the flow in the underlying tree is unique. Hence, we need to decide which arcs to upgrade and which arcs of a batch have positive flow. To do so, we propose a two-step dynamic program (DP) that solves the BMCF(K) on parallel-arc trees to optimality in polynomial time. In the first step, the DP considers each batch separately. It calculates for each batch i the cost $c_i(k', f, j)$ of sending f units of flow through batch i only via the first j arcs when k' arcs can be upgraded. After the DP calculated these costs $c_i(k', f, j)$ for all $k' \leq K$, $f \leq d_i$ and $j \leq i$ for all $i \leq |A_T|$, the DP determines in the second step how many arcs to upgrade in each batch.

2.3 The BMCF on series-parallel Graphs

Finally, we propose a pseudo-polynomial dynamic program (DP) that solves the BMCF(K) problem on series-parallel digraphs. The used definition can be found in [2]. The DP is based on the SP-tree T of the graph. The *induced subgraph* G_i of an SP node i is the subgraph obtained by applying all series and parallel operations of the subtree rooted at i.

Let h_i denote the *inner demand* in G_i which is the demand in G_i exluding the demand of t_i for each node i of T. We denote by $c_i(k, f, d)$ the cost of sending flow through G_i where k arcs are upgraded such that all inner demand h_i is satisfied; f units of flow exceed the total demand of G_i; and the demand of t_i is possibly partially met by d units of flow. In the intialization for the leaves i of T associated with a single arc, depending on whether the arc is upgraded or not, either the upper or lower cost applies to the total flow. In case the flow exceeds the capacity or $k > 1$ the cost is set to infinity. Let us consider an SP node i that represents a series operation where j and l are the child nodes of i and t_j is contracted with s_l. We know that the entire flow present in G_l must have been flow that exceeded the total demand of G_j. Hence $f_j = f_l + h_l + d_l$ and also $f_l = f$ and $d_l = d$. The demand of t_j must be satisfied by flow through G_j after the series operation since it will not be a target in future operations. Therefore, we only need to consider costs where $d_j = b(t_j)$. Since we must divide the budget k between the two subgraphs G_j and G_l, we can obtain $c_i(k, f, d)$ as follows.

$$c_i(k, f, d) = \min_{0 \leq k' \leq k} \{c_j(k', f + h_l + d, b(t_j)) + c_l(k - k', f, d)\} \tag{1}$$

Let us now consider an SP node i that represents a parallel operation where j and l are the child nodes of i. Due to the capacity constraints we must divide the budget k between the two subgraphs G_j and G_l and obtain $c_i(k, f, d)$ as follows.

$$c_i(k, f, d) = \min_{k' \leq k, x \leq f, y \leq d} \{c_j(k', f - x, d - y) + c_l(k - k', x, y)\} \tag{2}$$

The DP for the uncapacitated BMCF(K) problem on graphs with bounded tree-width as described in [2] can be easily modified for the capacitated case.

References

[1] Aggarwal, V., Y. Aneja and K. Nair, *Minimal spanning tree subject to a side constraint*, Computers & Operations Research **9** (1982), pp. 287 – 296.

[2] Büsing, C., S. Kirchner, A. Koster and A. Thome, *The budgeted minimum cost flow problem with unit upgrading cost*, http://www.optimization-online.org/DB_HTML/2015/10/5173.html (2015).

[3] Coene, S., P. Goos, P. Maya Duque, K. Sörensen and F. Spieksma, *The accessibility arc upgrading problem*, European Journal of Operational Research **224** (2013), pp. 458 – 465.

[4] Garey, M. and D. Johnson, *A guide to the theory of NP-completeness*, WH Freemann, New York (1979).

[5] Magazine, M. J., G. L. Nemhauser and L. E. Trotter Jr., *When the greedy solution solves a class of knapsack problems.*, Operations Research **23** (1975), p. 207.

Intersection of Longest Paths in Graph Classes

Márcia R. Cerioli [1,3]

COPPE - Sistemas e Computação and Instituto de Matemática
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brazil

Paloma Lima [1,2]

COPPE - Sistemas e Computação
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brazil

Abstract

Let G be a graph and $lpt(G)$ be the size of the smallest set $S \subseteq V(G)$ such that every longest path of G has at least one vertex in S. If $lpt(G) = 1$, then all longest paths of G have non-empty intersection. In this work, we prove that this holds for some graph classes, including ptolemaic graphs, P_4-sparse graphs, and starlike graphs, generalizing the existing result for split graphs.

Keywords: intersection of longest paths, graph classes, ptolemaic graphs, P_4-sparse graphs, starlike graphs

[1] This work was partially supported by CNPq, Brazil.

[2] Email: `paloma@cos.ufrj.br`

[3] Email: `cerioli@cos.ufrj.br`

1 Introduction

It is a well-known fact that in every simple connected graph, every two longest paths intersect, that is, they share a common vertex. In 1966, Gallai asked whether it is true that all longest paths of a connected graph share a common vertex. Even though the answer for Gallai's question is known to be negative for general graphs [4], many graph classes answer positively to this question. This is the case for split graphs [7], interval graphs [1], outerplanar graphs and 2-trees [4], circular-arc graphs [6], and graphs with matching number smaller than three [3]. In order to approach this problem, Rautenbach and Sereni [8] defined $lpt(G)$ to be the size of the smallest set $S \subseteq V(G)$ such that every longest path of G intersects S, where *lpt* stands for *longest path transversal*. Let $\mathcal{C}$ be a class of graphs. If $lpt(G) = 1$ for all $G \in \mathcal{C}$, then the answer for Gallai's question is positive in $\mathcal{C}$. In [8], the authors also provide upper bounds on $lpt(G)$ for general graphs and for some specific graph classes, such as planar and bounded treewidth graphs.

In this work, we determine graph classes that have positive answer for Gallai's question, including ptolemaic graphs, starlike graphs and P_4-sparse graphs. We also prove that if the blocks of a given graph are a split graph, an interval graph or have a universal vertex, then $lpt(G) = 1$.

2 Ptolemaic Graphs

A connected graph is a *ptolemaic graph* if for every four vertices u_1, u_2, u_3, u_4 of G, the inequality $d_{12}d_{34} \leq d_{13}d_{24} + d_{14}d_{23}$ is satisfyed, where d_{ij} is the distance between vertices u_i and u_j in G. The following characterization of ptolemaic graphs has been given by Howorka:

Theorem 2.1 (Howorka [5]) *The following conditions are equivalent:*

1. G is a ptolemaic graph.

2. G is a gem*-free graph and a chordal graph.*

3. G is a distance-hereditary graph and a chordal graph.

4. For every two non-disjoint maximal cliques Q and Q' of G, $Q \cap Q'$ separates $Q \setminus Q'$ and $Q' \setminus Q$.

A *minimal vertex separator* of G is a *uv*-minimal separator for some pair of vertices $u, v \in V(G)$. A family of sets $F = \{F_1, F_2, ..., F_k\}$ is said to be *laminar* if $F_i \cap F_j \neq \emptyset$ implies $F_i \subseteq F_j$ or $F_j \subseteq F_i$ $\forall i, j$. Uehara and Uno [9] proved the following theorem concerning the minimal separators of ptolemaic graphs:

Theorem 2.2 (Uehara and Uno [9]) *Let G be a chordal graph. G is a ptolemaic graph if and only if the family of minimal vertex separators contained in each maximal clique of G is laminar.*

The following result was suggested in the final remarks of [1]:

Theorem 2.3 (Balister et al. [1]) *Let G be a chordal graph. There exists a maximal clique K such that each longest path of G has at least one vertex in K.*

Using the Characterisation 2.1 and Theorems 2.2 and 2.3 we are able to prove the following:

Theorem 2.4 *If G is a ptolemaic graph, then $lpt(G) = 1$.*

3 Split-like Graphs

A graph is a *split graph* if $V(G)$ can be partitioned into an independent set and a clique. Split graphs have many different characterisations, for example, as intersection graphs of distinct substars of a star or as $(2K_2, C_4, C_5)$-free graphs. Klavžar and Petkovšek [7] proved that if G is a split graph, then $lpt(G) = 1$.

We prove the following stronger result:

Theorem 3.1 *Let G be a graph such that $V(G)$ can be partitioned into $k+1$ sets $(V_1, ..., V_k, K)$, such that*

1. K is a clique;

2. For all $x \in V_i$ and $y \in V_j$, $i \neq j$, it holds that $xy \notin E(G)$;

3. The vertices of V_i can be ordered $v_{i1}, v_{i2}, ..., v_{i|V_i|}$ in such a way that for all $x \in K$, if $xv_{ij} \in E(G)$, then $xv_{ik} \in E(G)$ for all $k < j$;

4. For all $x \notin K$, there exists $y \in K$ such that $xy \in E(G)$.

Then $lpt(G) = 1$.

It is worth noticing that split graphs satisfy the conditions in Theorem 3.1, as they are exactly the graphs obtained when $|V_i| = 1$, for every i. We are now able to use Theorem 3.1 to prove the following, where a starlike graph is the intersection graph of substars of a star [2].

Theorem 3.2 *If G is a $(2K_2, C_4)$-free graph, then $lpt(G) = 1$.*

Theorem 3.3 *If G is a starlike graph, then $lpt(G) = 1$.*

Theorem 3.4 *If G is a P_4-sparse graph, then $lpt(G) = 1$.*

4 Conditions on the Blocks of a Graph

Balister et al. [1] proved that $lpt(G) = 1$ if G is an interval graph. Given $x, y \in V(G)$, we say P is a xy-longest path if P is a longest path connecting x and y. We modify the proof presented in [1] in order to obtain the following:

Theorem 4.1 *Let G be an interval graph and $\{P_1, P_2, ..., P_k\}$ a set of paths such that P_i is a x_iy_i-longest path. If $P_i \cap P_j \neq \emptyset$ for all $i \neq j$, then $\cap_{i=1}^{k} P_i \neq \emptyset$.*

We also prove an analogous result for split graphs and then obtain some conditions on the blocks of a graph G that force $lpt(G) = 1$.

Theorem 4.2 *Let $G = (K \cup S, E)$ be a 2-connected split graph with maximal S. For all $x, y \in V(G)$, every xy-longest path contains all vertices of K.*

Theorem 4.3 *If every block of G is a split graph, an interval graph, or a graph with a universal vertex, then $lpt(G) = 1$.*

References

[1] P. Balister, E. Györi, J. Lehel, R. Schelp, Longest paths in circular arc graphs, *Comb. Probab. Comput.* 13 (2004) 311–317.

[2] M. R. Cerioli and J. L. Szwarcfiter, Characterizing intersection graphs of substars of a star, *Ars Combinatoria* 79 (2006) 21–31.

[3] F. Chen, Nonempty intersection of longest paths in a graph with a small matching number, *Czech. Math. J.* 65 (2015) 545–553.

[4] C. G. Fernandes, D. M. Martin, S. F. Rezende, and Y. Wakabayashi, Intersecting longest paths, *Discrete Math.* 313 (2013) 1401–1408.

[5] E. Howorka, A characterization of ptolemaic graphs, *J. Graph Theor.* 5 (1981) 323–331.

[6] F. Joos, A note on longest paths in circular arc graphs, *Discuss. Math. Graph Theory* 35 (2015) 419–426.

[7] S. Klavžar and M. Petkovšek, Graphs with nonempty intersection of longest paths, *Ars Combinatoria* 29 (1990) 43–52.

[8] D. Rautenbach and J. S. Sereni, Transversals of longest paths and cycles, *SIAM J. Discrete Math.* 28 (2014) 335–341.

[9] R. Uehara and Y. Uno, Laminar structure of ptolemaic graphs with applications,*Discrete Appl. Math.* 157 (2009) 1533–1543.

On Longest Cycles in Essentially 4-connected Planar Graphs

Jochen Harant [1]

Institut for Mathematics
Ilmenau University of Technology
Germany

Igor Fabrici [2]

Institut for Mathematics
P.J. Šafárik University
Košice, Slovakia

Stanislav Jendrol' [3]

Institut for Mathematics
P.J. Šafárik University
Košice, Slovakia

Abstract

A planar 3-connected graph G is essentially 4-connected if, for any 3-separator S of G, one component of the graph obtained from G by removing S is a single vertex. Jackson and Wormald proved that an essentially 4-connected planar graph on n vertices contains a cycle C such that $|V(C)| \geq \frac{2n+4}{5}$. For a cubic essentially 4-connected planar graph G, Grünbaum with Malkevitch, and Zhang showed that G has a cycle on at least $\frac{3}{4}n$ vertices. In the present paper the result of Jackson and Wormald is improved. Moreover, new lower bounds on the length of a longest cycle of G are presented if G is an essentially 4-connected planar graph of maximum degree 4 or G is an essentially 4-connected maximal planar graph.

Keywords: planar graph, longest cycle

We use standard notation and terminology of graph theory ([1]) and consider a finite simple 3-connected planar graph G with vertex set $V(G)$ and maximum degree $\Delta(G)$. A subset $S \subset V(G)$ is an s-*separator* of G if $|S| = s$ and $G - S$ is disconnected. It is well-known that $G - S$ has exactly two components if G is a 3-connected planar graph and S is a 3-separator of G. If S is a 3-separator of a 3-connected planar graph G and one component of $G - S$ is a single vertex, then S is a *trivial* 3-*separator* of G. If G is planar, 3-connected, and each 3-separator of G is trivial, then G is *essentially* 4-*connected*.

It is known ([6]) that there are infinitely many 3-connected planar graphs G on n vertices such that the length of a longest cycle of G is at most $9n^{\log_3 2}$. The constant 9 is improved several times, however, the exponent $\log_3 2$ is best possible ([2]). On the other hand, a 4-connected planar graph always contains a hamiltonian cycle ([9]). Since an essentially 4-connected planar graph is 3-connected but not necessarily 4-connected, we are interested in lower bounds on the length of longest cycles of an essentially 4-connected planar graph.

Jackson and Wormald [5] proved that every essentially 4-connected planar graph on n vertices contains a cycle C such that $|V(C)| \geq \frac{2n+4}{5}$. For a cubic essentially 4-connected planar graph G, Grünbaum and Malkevitch [4], and Zhang [10] showed that G has a cycle on at least $\frac{3}{4}n$ vertices. Given a real constant $c > \frac{2}{3}$, Jackson and Wormald [5] presented an infinite family of essentially 4-connected planar graphs G such that G does not contain a cycle on more than $c \cdot n$ vertices. This observation is even true for essentially 4-connected maximal planar graphs. To see this, let G' be a 4-connected maximal planar graph on $n' \geq 6$ vertices embedded into the plane and G be obtained by inserting a new vertex into each face of G' and connecting it with all three vertices of that face by an edge. Obviously, G is an essentially 4-connected maximal planar graph on $n = n' + (2n' - 4)$ vertices and the $2n' - 4$ vertices in $V(G) \setminus V(G')$ are pairwise independent. Hence each cycle of G contains at most $2n' = \frac{2}{3}(n+4)$ vertices. One can show easily that G contains a cycle on exactly $2n' = \frac{2}{3}(n+4)$ vertices.

It is well-known, that a 3-connected planar graph on $4 \leq n \leq 10$ vertices is

[1] Email: Jochen.Harant@tu-ilmenau.de
[2] Email: Igor.Fabrici@upjs.sk
[3] Email: stanislav.jendrol@upjs.sk

hamiltonian. It remains open whether a maximal planar (or even an arbitrary planar) essentially 4-connected graph on $n \geq 11$ vertices contains a cycle C such that $|V(C)| \geq \frac{2}{3}(n+4)$.

Our results are presented in the following Theorem 0.1 improving the result of Jackson and Wormald. Moreover, new lower bounds on the length of a longest cycle of G are presented if G is an essentially 4-connected planar graph of maximum degree 4 or G is an essentially 4-connected maximal planar graph.

Theorem 0.1 ([3])
Let G be an essentially 4-connected planar graph on $n \geq 11$ vertices and C be a longest cycle of G.
(i) $|V(C)| \geq \frac{1}{2}(n+4)$.
(ii) If $\Delta(G) = 4$, then $|V(C)| \geq \frac{3}{5}n$.
(iii) If G is maximal planar, then $|V(C)| \geq \frac{13}{21}(n+4)$.

To sketch a proof of Theorem 0.1, we need some definition.

Let C be a cycle of a graph G and B be a component of $G - V(C)$. A vertex $x \in V(C)$ is a *touch vertex* of B if x is adjacent to a vertex of $V(B)$. Note that B has at least 3 touch vertices, if G is a 3-connected planar graph.

In [9], Tutte proved a remarkable and famous result on cycles in 2-connected planar graphs implying that a 4-connected planar graph is hamiltonian. This result has been extended several times ([7], [8]). We use the following Lemma 0.2 of Sanders ([7]) as a version of Tutte's result for 3-connected planar graphs.

Lemma 0.2 *Every 3-connected planar graph G with two prescribed edges a and b contains a cycle C through a and b such that each component of $G-V(C)$ has exactly 3 touch vertices.*

A cycle C of G is an *outer-independent-3-cycle* (OI3-cycle), if $V(G) \setminus V(C)$ is an independent set of vertices and $d(x) = 3$ for all $x \in V(G) \setminus V(C)$.

We prove Lemma 0.3.

Lemma 0.3 *Let G be an essentially 4-connected planar graph G and let a and b be non-adjacent edges of G. If a and b belong to a common face of G or all end vertices of a and b have degree at least 4 in G, then G contains an* OI3*-cycle C through a and b.*

Note that a hamiltonian cycle of a graph is an OI3-cycle. Let $a = yz$ be an edge of an OI3-cycle C of a graph G and assume that y and z have a common neighbor $x \in V(G) \setminus V(C)$. Then let C' be the cycle of G obtained from C by replacing the edge a with the path (y, x, z). In this case, a is an *extendable edge of C*. Note that C' is again an OI3-cycle of G, $|V(C')| = |V(C)| + 1$, and that C' has less extendable edges than C. Obviously, a longest OI3-cycle of G does not contain an extendable edge.

Using Lemma 0.3, we prove Lemma 0.4 implying Theorem 0.1.

Lemma 0.4 *Let G be an essentially 4-connected planar graph on $n \geq 11$ vertices.*

(i) *G contains an OI3-cycle.*

(ii) *If C is an OI3-cycle of G without extendable edges, then $|V(C)| \geq \frac{1}{2}(n + 4)$.*

(iii) *If $\Delta(G) = 4$ and C is an OI3-cycle of G, then $|V(C)| \geq \frac{3}{5}n$.*

(iv) *If G is maximal planar and C is a longest OI3-cycle of G, then $|V(C)| \geq \frac{13}{21}(n + 4)$.*

References

[1] J.A. Bondy, U.S.R. Murty, Graph Theory, Springer 2008.

[2] G. Chen, X. Yu, Long cycles in 3-connected graphs, *J. Combin. Theory Ser. B* 86(2002)80-99.

[3] I. Fabrici, J. Harant, S. Jendrol', On Longest Cycles in Essentially 4-connected Planar Graphs, accepted in *Discussiones Mathematicae Graph Theory.*

[4] B. Grünbaum, J. Malkevitch, Pairs of edge-disjoint Hamilton circuits, *Aequationes Math.* 14(1976)191-196.

[5] B. Jackson, N.C. Wormald, Longest cycles in 3-connected planar graphs, *J. Combin. Theory Ser. B* 54(1992)291-321.

[6] J.W. Moon, L. Moser, Simple paths on polyhedra, *Pacific J. Math.* 13(1963)629-631.

[7] D.P. Sanders, On paths in planar graphs, *J. Graph Theory* 24(1997)341-345.

[8] C. Thomassen, A theorem on paths in planar graphs *J. Graph Theory* 7(1983)169-176.

[9] W.T. Tutte, A theorem on planar graphs, *Trans. Amer. Math. Soc.* 82(1956)99-116.

[10] C.-Q. Zhang, Longest cycles and their chords, *J. Graph Theory* 11(1987)521-529.

Dimension and codimension of simple games

Sascha Kurz

LS Wirtschaftsmathematik, Universitaet Bayreuth, Germany

Xavier Molinero[1]

Department of Mathematics,Technical University of Catalonia, Manresa, Spain

Martin Olsen

BTECH, Aarhus University, Denmark

Maria Serna[2]

Department of Computer Science, Technical University of Catalonia, Spain

Abstract

This paper studies the complexity of computing a representation of a simple game as the intersection (union) of weighted majority games, as well as, the dimension or the codimension. We also present some examples with linear dimension and exponential codimension with respect to the number of players.

Keywords: Simple games, Dimension, Codimension, Computational complexity

1 Introduction and preliminaries

We consider the so-called *simple games* and the computational complexity of representing them as unions or intersections of weighted majority games.

[1] Partially funded by Grant MTM2015-66818-P from MINECO.

[2] Partially funded by MINECO and FEDER funds under grant TIN2013-46181-C2-1-R, and AGAUR grant SGR 2014–1034.

[3] Emais: sascha.kurz@uni-bayreuth.de, xavier.molinero@upc.edu, martino@btech.au.dk, mjserna@cs.upc.edu

Simple games and its dimension, as well as, weighted majority games, were defined by Taylor and Zwicker [10]. Later, Freixas and Marciniak [3] introduced a new concept, the codimension of simple games.

A *simple game* is a tuple $\Gamma = (N, \mathcal{W})$, where N is a finite set of *players* and $\mathcal{W} \subseteq \mathcal{P}(N)$ is a monotonic family of subsets of N. Furthermore, *its dual* $\Gamma^* = (N, \mathcal{W}^*)$ is the game such that $\mathcal{W}^* = \{S \subseteq N : N \setminus S \notin \mathcal{W}\}$. Γ is said to be *self-dual* if $\Gamma = \Gamma^*$. Note that $(\Gamma^*)^* = \Gamma$. Given two simple games $\Gamma_1 = (N_1, \mathcal{W}_1)$ and $\Gamma_2 = (N_2, \mathcal{W}_2)$, they are *equivalent* if $N_1 = N_2$ and $\mathcal{W}_1 = \mathcal{W}_2$. The subsets of N are called *coalitions*, the set N is the *grand coalition* and each $X \in \mathcal{W}$ is a *winning coalition*. The complement of the family of winning coalitions is the family of *losing coalitions* $\mathcal{L}$, i.e., $\mathcal{L} = \mathcal{P}(N) \setminus \mathcal{W}$. Any of those set families determine uniquely the game Γ and constitute one of the usual forms of representation for simple games [10], although the size of the representation is not, in general, polynomial in the number of players [8].

A simple game Γ is a *weighted majority game* (WMG) if it admits a representation by means of $n+1$ nonnegative real numbers $[q; w_1, \ldots, w_n]$ such that $S \in \mathcal{W} \iff w(S) \geq q$ where, for each coalition $S \subseteq N$, $w(S) = \sum_{i \in S} w_i$. The number q is called the quota and w_i the weight of the player i. It is well known that any WMG admits a representation with integer numbers. The *dimension* of a simple game Γ is the least k such that there exists WMGs $\Gamma_1, \ldots, \Gamma_k$ such that $\Gamma = \Gamma_1 \cap \ldots \cap \Gamma_k$. On the other hand, the *codimension* of a simple game Γ is the least k such that there exists WMGs $\Gamma_1, \ldots, \Gamma_k$ such that $\Gamma = \Gamma_1 \cup \ldots \cup \Gamma_k$.

There are many theoretical results and examples about dimension and codimension [9,7,3,6,4,10,5] including computational complexity results [1]. We present some results that will be used later on.

Lemma 1.1 *The dimension of a simple game v is bounded above by $\left|\mathcal{L}^M\right|$ and the co-dimension is bounded above by $|\mathcal{W}^m|$.*

Lemma 1.2 *Let Γ be a simple game. Γ is the intersection of t weighted games if and only if Γ^* is the union of t weighted games. Furthermore a representation, as union (intersection), of Γ^* can be obtained from a representation, as intersection (union), of Γ in polynomial time. Moreover, dim(Γ)=codim(Γ^*), and if Γ is self-dual then dim(Γ) = codim(Γ).*

Note that the converse statement of the last sentence is not true in general as there are weighted games which are not self-dual.

2 Computational complexity of related problems

First, we present a simple game with $2n$ players, dimension n and codimension 2^{n-1}. Other examples of high dimensional games can be found in [10,9].

Example 2.1 Given a positive integer n, Freixas and Marcinicak (Theorem 2 of [3]) define a simple game with $2n$ players and dimension n. Let $\Gamma = (N, \mathcal{W})$ be a simple game defined by $N = \{1, 2, \ldots, 2n\}$ and $S \in \mathcal{W}$ iff $S \cap \{2i-1, 2i\} \neq \emptyset$, $i \in \{1, \ldots, n\}$, then Γ has dimension n.

As S is a winning coalition in Γ^* iff $N \setminus S$ is a losing coalition in Γ, $\Gamma^* = (N, \mathcal{W}_1^* \cup \ldots \cup \mathcal{W}_n^*)$, where $\mathcal{W}_i^* = \{S \subseteq N : \{2i-1, 2i\} \subseteq S\}$, $i \in \{1, \ldots, n\}$. As Γ^* is a composition of n unanimity games, Γ^* has dimension 2^{n-1} [4] and Γ has codimension 2^{n-1} (by Lemma 1.2).

Proposition 2.2 *Given a simple game Γ as union (intersection) of weighted games, computing a representation of Γ as intersection (union) of weighted games requires exponential time.*

The complexity of several problems about representations of simple games as intersections of WMGs were analyzed in [1]. We provide here a new reduction from the NP-hard *Subset Sum Problem (*SSP*)*. Our reduction differs in the fact that, for the game $\Gamma(I, d)$ associated to an instance I, we know both the dimension and the codimension.

Lemma 2.3 *Let $d > 1$. When I is a yes instance of* SSP *then $dim(\Gamma(I, d)) = d$ and $codim(\Gamma(I, d)) = 2^d$, otherwise, $dim(\Gamma(I, d)) = codim(\Gamma(I, d)) = 1$.*

Combining lemmas 2.3 and 1.2 we can prove the following results.

Proposition 2.4 *Let d_1 and d_2 be two integers with $1 \leq d_2 < d_1$. Then the problem of deciding whether the union of d_1 given WMGs can also be represented as the union of d_2 WMGs is NP-hard.*

Proposition 2.5 *Let d_1 and d_2 be two integers with $1 \leq d_1, d_2$. Then the problem of deciding whether the intersection (union) of d_1 given WMGs can also be represented as the union (intersection) of d_2 WMGs is NP-hard.*

As a consequence of the previous results, given a simple game Γ as union or intersection of WMGs, to compute $dim(\Gamma)$, $codim(\Gamma)$ or deciding whether Γ is weighted are NP-hard problems. Recall that two game representations are said to be *equivalent* whenever the represented games have the same set of winning coalitions. We can extend several results on equivalence problems from [2] to games given as unions of WMG, in particular we have.

Proposition 2.6 *Checking whether a given union of* WMGs *is equivalent to a given union of* WMGs *is co-NP-complete, even if all weights are equal to 0 or 1.*

It remains open to exhaustively classify the dimension and codimension of all complete simple game up to n players. Some bounds about dimension are given by Freixas and Puente [4] and Olsen *et al.* [9]. As well as to find complete simple games with *small* dimension (codimension), but with *large* codimension (dimension), and to construct analytical examples with specific dimension and codimension. It is also interesting to find real simple games with *large* dimension or codimension as the example given by Kurz and Napel [7].

References

[1] V.G. Deĭneko and G.J. Woeginger. On the dimension of simple monotonic games. *European Journal of Operational Research*, 170:315–318, 2006.

[2] E. Elkind, L.A. Goldberg, P.W. Goldberg, and M. Wooldridge. On the dimensionality of voting games. In *AAAI Conference on Artificial Intelligence*, pages 69–74, 2008.

[3] J. Freixas and D. Marciniak. On the notion of dimension and codimension of simple games. *Cont. to Game Theory and Management*, 3:67–81, 2010.

[4] J. Freixas and M. A. Puente. A note about games-composition dimension. *Discrete Applied Mathematics*, 113(2–3):265–273, 2001.

[5] J. Freixas and M.A. Puente. Complete games with minimum. *Annals of Operations Research*, 84:97–109, 1998.

[6] J. Freixas and M.A. Puente. Dimension of complete simple games with minimum. *European Journal of Operational Research*, 188(2):555–568, 2008.

[7] S. Kurz and S. Napel. Dimension of the Lisbon voting rules in the EU Council: a challenge and new world record. *Optimization Letters*, to appear, 2015.

[8] X. Molinero, F. Riquelme, and M. J. Serna. Forms of representations for simple games: sizes, conversions and equivalences. *Mathematical Social Sciences*, 76:87–102, 2015.

[9] M. Olsen, S. Kurz, and X. Molinero. On the construction of high dimensional simple games. *CoRR*, abs/1602.01581, 2016.

[10] A.D. Taylor and W.S. Zwicker. *Simple games: desirability relations, trading, and pseudoweightings*. Princeton University Press, New Jersey, USA, 1999.

Equivalent Nonlinear Complementarity Problem for Chance-constrained Games

Vikas Vikram Singh [1]

Laboratoire de Recherche en Informatique, Université Paris Sud XI, Bât 650, 91405, Orsay, France

Oualid Jouini [2]

Laboratoire Génie Industriel, Ecole Centrale Paris,Grande Voie des Vignes, 92290, Châtenay-Malabry, France

Abdel Lisser [3]

Laboratoire de Recherche en Informatique, Université Paris Sud XI, Bât 650, 91405, Orsay, France

Abstract

We formulate a random bimatrix game, where the entries of the payoff matrix of each player jointly follow a multivariate elliptically symmetric distribution, as a chance-constrained game. We show that a Nash equilibrium of a chance-constrained game can be computed by solving an equivalent nonlinear complementarity problem.

Keywords: Chance-constrained game, Nash equilibrium, Elliptically symmetric distribution, Nonlinear complementarity problem.

[1] Email: `vikas.singh@lri.fr`

[2] Email: `oualid.jouini@ecp.fr`

[3] Email: `abdel.lisser@lri.fr`

1 Introduction

The games with deterministic payoffs have been extensively studied in the literature. However, there can be practical cases where the players' payoffs are better modeled by random variables. The wholesale electricity markets are the good examples that capture this situation [2]. We refer [5] and references therein for recent literature on games with random payoffs. The situation where players are risk neutral can be handled using expected payoff criterion [3]. The risk averse situation is better modeled by formulating the game as a chance-constrained game, e.g., see [5], [1], [4] and references therein. The existence of Nash equilibrium for chance-constrained game has been shown in [5]. To compute the Nash equilibria of a chance-constrained game, where the players' payoffs are independent random variables, a mathematical programming based approach have been proposed in [4]. In general, the payoffs can be dependent random variables following a certain joint probability distribution.

In this paper, we consider a two player random bimatrix game where the entries of the payoff matrix of each player follow a multivariate elliptically symmetric distribution. We formulate the corresponding chance-constrained game as an equivalent nonlinear complementarity problem (NCP).

2 Chance-constrained Game

We consider a bimatrix game (A, B) where the entries of A and B are random variables. Let $I = \{1, 2, \cdots, m\}$ and $J = \{1, 2, \cdots, n\}$ be the action sets of player 1 and player 2 respectively. Let X and Y be the sets of all probability distributions over action sets I and J respectively. The generic elements $x \in X$ and $y \in Y$ are mixed strategies of player 1 and player 2 respectively. We consider the situation where players are risk averse. Let $\alpha_1 \in [0, 1]$ and $\alpha_2 \in [0, 1]$ be the confidence (risk) levels of player 1 and player 2 respectively. Let $\alpha = (\alpha_1, \alpha_2)$ be a confidence level vector which is of common knowledge. For given strategy pair (x, y) and α, the payoff of player 1 and player 2, defined using chance constraint, is respectively given by,

$$u_1^{\alpha_1}(x, y) = \sup\{u | P(x^T Ay \geq u) \geq \alpha_1\}, \tag{1}$$

$$u_2^{\alpha_2}(x, y) = \sup\{v | P(x^T By \geq v) \geq \alpha_2\}. \tag{2}$$

We assume that the probability distributions of the payoff matrix of one player are known to another player. Then, for a given α the chance constrained game defined above is a non-cooperative game with complete information. The definition of Nash equilibrium of a chance-constrained game is a standard definition of Nash equilibrium for payoff functions $u_1^{\alpha_1}(\cdot)$ and $u_2^{\alpha_2}(\cdot)$.

3 Nonlinear Complementarity Problem Formulation

We represent the entries of A (resp. B) by an $mn \times 1$ vector $a = (a_1, a_2, \cdots, a_m)^T$ (resp. $b = (b_1, b_2, \cdots, b_m)$), where $a_i = (a_{i1}, a_{i2}, \cdots, a_{in})$ (resp. $b_i = (b_{i1}, b_{i2}, \cdots, b_{in})$) for all $i \in I$. We assume that the vector a (resp. b) follow a multivariate elliptically symmetric distribution with parameters μ_1 and Σ_1 (resp. μ_2 and Σ_2). Let Σ_1 and Σ_2 be positive definite matrices. Denote, $\mu_1 = (\mu_{1,1}, \mu_{1,2}, \cdots, \mu_{1,m})^T$, where $\mu_{1,i} = (\mu_{1,i1}, \mu_{1,i2}, \cdots, \mu_{1,in})$ for all $i \in I$. The vector μ_2 is defined similarly. For a given (x, y), define $\eta(x, y) = (\eta_1, \eta_2, \cdots, \eta_m)^T$, where $\eta_i = (\eta_{i1}, \eta_{i2}, \cdots, \eta_{in})$, $i \in I$, such that $\eta_{ij} = x_i y_j$. From [5], the payoff functions of player 1 and player 2, defined by (1) and (2) respectively, are given by,

$$u_1^{\alpha_1}(x, y) = \mu_1^T \eta(x, y) + ||\Sigma_1^{1/2} \eta(x, y)|| F_{Z_1}^{-1}(1 - \alpha_1), \tag{3}$$

$$u_2^{\alpha_2}(x, y) = \mu_2^T \eta(x, y) + ||\Sigma_2^{1/2} \eta(x, y)|| F_{Z_2}^{-1}(1 - \alpha_2), \tag{4}$$

where Z_1 and Z_2 follow a univariate spherically symmetric distribution, and $F_{Z_1}^{-1}(\cdot)$ and $F_{Z_2}^{-1}(\cdot)$ denote its quantile functions, and $|| \cdot ||$ is the Euclidean norm. Let $\mu_1(y) = \left(\sum_{j \in J} \mu_{1,ij} y_j \right)_{i \in I}$ be an $m \times 1$ vector for fixed y, and $\mu_2(x) = \left(\sum_{i \in I} \mu_{2,ij} x_i \right)_{j \in J}$ be an $n \times 1$ vector for fixed x. For fixed (x, y), let $f_1(x, y) = \Sigma_1^{1/2} \eta(x, y)$ and $f_2(x, y) = \Sigma_2^{1/2} \eta(x, y)$. Then, $f_1(\cdot)$ and $f_2(\cdot)$ are vectors of functions. Let $\mathcal{J}_{f_1(\cdot, y)}(x)$ be a Jacobian matrix of $f_1(\cdot, y)$ for fixed y, and $\mathcal{J}_{f_2(x, \cdot)}(y)$ be a Jacobian matrix of $f_2(x, \cdot)$ for fixed x. For a given vector ν, $\nu \geq 0$ implies componentwise non-negativity, and $\perp$ means that elementwise equality must hold at one or both sides. We have the following result.

Theorem 3.1 *Consider a bimatrix game (A, B) where the entries of A (resp. B) jointly follow a multivariate elliptically symmetric distribution with parameters μ_1 and Σ_1 (resp. μ_2 and Σ_2). Let Σ_1 and Σ_2 be positive definite matrices. Consider a point $\zeta^{*T} = (x^{*T}, y^{*T}, \lambda_1^*, \lambda_2^*, \lambda_3^*, \lambda_4^*)$. Then, the strategy part (x^*, y^*) of ζ^* is a Nash equilibrium of a chance-constrained game, for a given $\alpha \in (0.5, 1]^2$, if and only if ζ^* is a solution of the NCP given below:*

$$0 \leq x \perp -\mu_1(y) - \frac{(\mathcal{J}_{f_1(\cdot, y)}(x))^T f_1(x, y) F_{Z_1}^{-1}(1 - \alpha_1)}{||f_1(x, y)||} - \lambda_1 e_m + \lambda_2 e_m \geq 0,$$

$$
\begin{aligned}
&0 \leq y \perp -\mu_2(x) - \frac{(\mathcal{J}_{f_2(x,\cdot)}(y))^T f_2(x,y) F_{Z_2}^{-1}(1-\alpha_1)}{||f_2(x,y)||} - \lambda_3 e_n + \lambda_4 e_n \geq 0,\\
&0 \leq \lambda_1 \perp \sum_{i\in I} x_i - 1 \geq 0, \quad 0 \leq \lambda_2 \perp 1 - \sum_{i\in I} x_i \geq 0,\\
&0 \leq \lambda_3 \perp \sum_{j\in J} y_j - 1 \geq 0, \quad 0 \leq \lambda_4 \perp 1 - \sum_{j\in J} y_j \geq 0.
\end{aligned}
$$

The chance-constrained game proposed in this paper can be used to model the Cournot or Bertrand competition among electricity firms, where, using discretization the action sets of the firms are finite, and the demand or cost functions are uncertain. We will provide the proof of Theorem 3.1, and the application of chance-constrained game in the electricity market together with numerical results using the proposed NCP, in the extended version of the paper.

Acknowledgements

This research was supported by Fondation DIGITEO, SUN grant No. 2014-0822D.

References

[1] R. A. Blau, Random-payoff two person zero-sum games, Operations Research 22 (6) (1974) 1243–1251.

[2] M. Mazadi, W. D. Rosehart, H. Zareipour, O. P. Malik, M. Oloomi, Impact of wind integration on electricity markets: A chance-constrained Nash Cournot model, International Transactions on Electrical Energy Systems 23 (1) (2013) 83–96.

[3] U. Ravat, U. V. Shanbhag, On the characterization of solution sets of smooth and nonsmooth convex stochastic Nash games, Siam Journal of Optimization 21 (3) (2011) 1168–1199.

[4] V. V. Singh, O. Jouini, A. Lisser, Chance-constrained games: A mathematical programming approach (2015).
URL http://www.optimization-online.org/DB_FILE/2015/12/5270.pdf

[5] V. V. Singh, O. Jouini, A. Lisser, Existence of Nash equilibrium for chance-constrained games (2015).
URL http://www.optimization-online.org/DB_FILE/2015/06/4977.pdf

Determining the Optimal Strategies for Zero-Sum Average Stochastic Positional Games

Dmitrii Lozovanu

Applied Mathematics
Institute of Mathematics and Computer Science of Moldova Academy of Sciences
Chisinau, Moldova

Stefan Pickl

Institute for Theoretical Computer Science, Mathematics and Operations Research
Universität der Bundeswehr, München,
Neubiberg-München, Germany

Abstract

We consider a class of zero-sum stochastic positional games that generalizes the deterministic antagonistic positional games with average payoffs. We prove the existence of saddle points for the considered class of games and propose an approach for determining the optimal stationary strategies of the players.

Keywords: Average Markov decision process, Zero-sum stochastic positional games, Optimal strategies, Saddle points.

[1] Email: dmitrii.lozovanu@math.md
[2] Email: stefan.pickl@unibw.de

1 Introduction and Problem Formulation

The aim of this paper is to prove the existence of saddle points for a class of antagonistic stochastic games that extends deterministic positional games with average payoffs from [1,2,5]. The considered class of games we formulate by using the framework of a Markov decision process (X, A, c, p, x_0) with a finite set of states X, a finite set of actions $A = \cup_{x \in X} A(x)$ where $A(x)$ is the set of actions in the state $x \in X$, a probability transition function $p : X \times A \times X \to [0, 1]$ that satisfies the condition $\sum_{y \in X} p^a_{x,y} = 1, \ \forall x \in X, a \in A(x)$, and the cost function $c : X \times X \to R$. We assume that the Markov process describes the evolution of a dynamic system that is controlled by two players as follows: The set of states is divided into two subsets $X = X_1 \cup X_2$ with $X_1 \cap X_2 = \emptyset$, where X_1 represents the position set of the first player and X_2 represents the position set of the second player. At time moment $t = 0$ the dynamical system is in the state x_0. If this state belongs to the set of positions of the first player then the action $a_0 \in A(x_0)$ in this state is selected by the first player, otherwise the action $a_0 \in A(x_0)$ is chosen by the second player. After that the dynamical system passes randomly to an another state according to the probability distribution $\{p^{a_0}_{x_0,y}\}$. At time moment $t = 1$ the players observe the state $x_1 \in X$ of the system. If x_1 belongs to the set of positions of the first player then the action $a \in A(x_1)$ is chosen by the first player, otherwise the action is chosen by the second one and so on, indefinitely. In this process the first player intends to maximize $\lim_{t\to\infty} \inf \frac{1}{t} \sum_{\tau=1}^{t} \mu(x_{\tau-1})$, where $\mu(x_\tau) = \sum_{y \in X} p^{a_\tau}_{x_\tau,y} c_{x_\tau,y}$, and the second player intends to minimize $\lim_{t\to\infty} \sup \frac{1}{t} \sum_{\tau=1}^{t} \mu(x_{\tau-1})$. We assume that the players use stationary strategies of a selection of the actions in their position sets. We define the stationary strategies of the players as maps: $s^1 : x \to a \in A(x)$ for $x \in X_1$; $\quad s^2 : x \to a \in A(x)$ for $x \in X_2$. Let s^1, s^2 be arbitrary strategies of the players.Then $s = (s^1, s^2)$ determines a Markov process induced by probability distributions $\{p^{s^i(x)}_{x,y}\}$ in the states $x \in X_i$, $i = 1, 2$ and a given starting state x_0. For this Markov process with transition costs $c_{x,y}, \ x, y \in X$ we can determine the average cost per transition $F_{x_o}(s^1, s^2)$. The function $F_{x_o}(s^1, s^2)$ on $S = S^1 \times S^2$, where S^1 and S_2 represent the corresponding sets of the stationary strategies of player 1 and player 2, defines an antagonistic game. This game is determined by position sets X_1, X_2, the set of actions A, the probability function p, the cost function c and the starting state x_0. We denote this game by (X_1, X_2, A, c, p, x_0) and call it zero-sum average stochastic positional game. In this game, we are seeking for the saddle points.

2 The Main Results

We show that for the considered zero-sum game there exists a saddle point, i.e there exist s^{1^*}, s^{2^*} for which

$$F_{\overline{x}}(s^{1^*}, s^{2^*}) = \max_{s^1 \in S^1} \min_{s^2 \in S^2} F_{\overline{x}}(s^1, s^2) = \min_{s^2 \in S^2} \max_{s^1 \in S^1} F_{\overline{x}}(s^1, s^2).$$

Theorem 2.1 *Let $(X_1,\ X_2,\ A,\ c,\ p, \overline{x})$ be an antagonistic stochastic positional game with average payoff $F_{\overline{x}}(s^1, s^2)$. Then the system of equations*

$$(1) \qquad \begin{cases} \varepsilon_x + \omega_x = \max\limits_{a \in A(x)} \Big\{ \mu_{x,a} + \sum\limits_{y \in X} p^a_{x,y} \varepsilon_y \Big\}, & \forall x \in X_1; \\ \varepsilon_x + \omega_x = \min\limits_{a \in A(x)} \Big\{ \mu_{x,a} + \sum\limits_{y \in X} p^a_{x,y} \varepsilon_y \Big\}, & \forall x \in X_2; \end{cases}$$

has a solution under the set of solutions of the system of equations

$$(2) \qquad \begin{cases} \omega_x = \max\limits_{a \in A(x)} \Big\{ \sum\limits_{y \in X} p^a_{x,y} \omega_y \Big\}, & \forall x \in X_1; \\ \omega_x = \min\limits_{a \in A(x)} \Big\{ \sum\limits_{y \in X} p^a_{x,y} \omega_y \Big\}, & \forall x \in X_2, \end{cases}$$

i.e. the system of equations (2) has such a solution $\omega^*_x,\ x \in X$ for which there exists a solution $\varepsilon^*_x,\ x \in X$ of the system of equations

$$\begin{cases} \varepsilon_x + \omega^*_x = \max\limits_{a \in A(x)} \Big\{ \mu_{x,a} + \sum\limits_{y \in X} p^a_{x,y} \varepsilon_y \Big\}, & \forall x \in X_1; \\ \varepsilon_x + \omega^*_x = \min\limits_{a \in A(x)} \Big\{ \mu_{x,a} + \sum\limits_{y \in X} p^a_{x,y} \varepsilon_y \Big\}, & \forall x \in X_2. \end{cases}$$

The optimal stationary strategies s^{1^*}, s^{2^*} of the players can be found by fixing arbitrary maps $s^{1^*}(x) \in A(x)$ for $x \in X_1$ and $s^{2^*}(x) \in A(x)$ for $x \in X_2$ such that

$$s^{1^*}(x) \in \Big(Arg \max_{a \in A(x)} \Big\{ \sum_{y \in X} p^a_{x,y} \omega^*_y \Big\} \Big) \bigcap \Big(Arg \max_{a \in A(x)} \Big\{ \mu_{x,a} + \sum_{y \in X} p^a_{x,y} \varepsilon^*_y \Big\} \Big),\ x \in X_1,$$

$$s^{2^*}(x) \in \Big(Arg \max_{a \in A(x)} \Big\{ \sum_{y \in X} p^a_{x,y} \omega^*_y \Big\} \Big) \bigcap \Big(Arg \max_{a \in A(x)} \Big\{ \mu_{x,a} + \sum_{y \in X} p^a_{x,y} \varepsilon^*_y \Big\} \Big),\ x \in X_2.$$

Proof (Sketch) Let $\overline{x} \in X$ be an arbitrary state and consider the stationary strategies $\overline{s}^1 \in S^1$, $\overline{s}^2 \in S^2$ for which $F_{\overline{x}}(\overline{s}^1, \overline{s}^2) = \min_{s^2 \in S^2} \max_{s^1 \in S^1} F_{\overline{x}}(s^1, s^2)$. We show that $F_{\overline{x}}(\overline{s}^1, \overline{s}^2) = \max_{s^1 \in S^1} \min_{s^2 \in S^2} F_{\overline{x}}(s^1, s^2)$, i.e. we show that $\overline{s}^1 = s^{1*}$, $\overline{s}^2 = s^{2*}$.

If we consider the Markov decision process induced by strategies $\overline{s}^1, \overline{s}^2$ then according to the properties of the bias equations for this decision process the system of linear equations

$$(3) \qquad \begin{cases} \varepsilon_x + \omega_x = \mu_{x,a} + \sum\limits_{y \in X} p_{x,y}^a \varepsilon_y, & \forall x \in X_1,\ a = \overline{s}^1(x); \\ \varepsilon_x + \omega_x = \mu_{x,a} + \sum\limits_{y \in X} p_{x,y}^a \varepsilon_y, & \forall x \in X_2,\ a = \overline{s}^2(x); \\ \omega_x = \sum\limits_{y \in X} p_{x,y}^a \omega_y, & \forall x \in X_1,\ a = \overline{s}^1(x); \\ \omega_x = \sum\limits_{y \in X} p_{x,y}^a \omega_y, & \forall x \in X_2,\ a = \overline{s}^2(x) \end{cases}$$

has the solution ε_x^*, ω_x^* $(x \in X)$ which for a fixed strategy $\overline{s}^2 \in S^2$ satisfies the condition:

$$\begin{cases} \varepsilon_x^* + \omega_x^* \geq \mu_{x,a} + \sum\limits_{y \in X} p_{x,y}^a \varepsilon_y^*, & \forall x \in X_1,\ a \in A(x); \\ \varepsilon_x^* + \omega_x^* = \mu_{x,a} + \sum\limits_{y \in X} p_{x,y}^a \varepsilon_y^*, & \forall x \in X_2,\ a = \overline{s}^2(x); \\ \omega_x^* \geq \sum\limits_{y \in X} p_{x,y}^a \omega_y^*, & \forall x \in X_1,\ a \in A(x); \\ \omega_x^* = \sum\limits_{y \in X} p_{x,y}^a \omega_y^*, & \forall x \in X_2,\ a = \overline{s}^2(x) \end{cases}$$

and $F_x(\overline{s}^1, \overline{s}^2) = \omega_x^*$, $\forall x \in X$.

Taking into account that $F_x(\overline{s}^1, \overline{s}^2) = \min_{s^2 \in S^2} F_x(\overline{s}^1, s^2)$ then for a fixed strategy $\overline{s}^1 \in S^1$ the solution ϵ_x^*, ω_x^* $(x \in X)$ satisfies the condition

$$\begin{cases} \varepsilon_x^* + \omega_x^* = \mu_{x,a} + \sum\limits_{y \in X} p_{x,y}^a \varepsilon_y^*, & \forall x \in X_1,\ a = \overline{s}^1(x); \\ \varepsilon_x^* + \omega_x^* \leq \mu_{x,a} + \sum\limits_{y \in X} p_{x,y}^a \varepsilon_y^*, & \forall x \in X_2,\ a \in A(x); \\ \omega_x^* = \sum\limits_{y \in X} p_{x,y}^a \omega_y^*, & \forall x \in X_1,\ a = \overline{s}^1(x); \\ \omega_x^* \leq \sum\limits_{y \in X} p_{x,y}^a \omega_y^*, & \forall x \in X_2,\ a \in A(x) \end{cases}$$

So, the following system

$$\begin{cases} \varepsilon_x + \omega_x \geq \mu_{x,a} + \sum\limits_{y \in X} p^a_{x,y} \varepsilon_y, & \forall x \in X_1,\ a \in A(x); \\ \varepsilon_x + \omega_x \leq \mu_{x,a} + \sum\limits_{y \in X} p^a_{x,y} \varepsilon_y, & \forall x \in X_2,\ a \in A(x); \\ \qquad \omega_x \geq \sum\limits_{y \in X} p^a_{x,y} \omega_y, & \forall x \in X_1,\ a \in A(x); \\ \qquad \omega_x \leq \sum\limits_{y \in X} p^a_{x,y} \omega_y, & \forall x \in X_2,\ a \in A(x) \end{cases}$$

has a solution, which satisfies condition (3). This means that $\overline{s}^1 = s^{1^*}$, $\overline{s}^2 = s^{2^*}$ and

$$F_{\overline{x}}(s^{1^*}, \overline{s}^2) = \max_{s^1 \in S^1} \min_{s^2 \in S^2} F_{\overline{x}}(s^1, s^2) = \min_{s^2 \in S^2} \max_{s^1 \in S^1} F_{\overline{x}}(s^1, s^2),\ \forall \overline{x} \in X,$$

i.e. , the theorem holds. □

The obtained saddle point condition for zero-sum stochastic games generalizes the saddle point condition for deterministic average positional games from [1,5]. Based on Theorem 2.1 we may conclude that the optimal strategies of the players in the considered game can be found if we determine a solution of equations (1),(2). We have shown that a solution of these equations can be determined using iterative algorithms like algorithms for determining the optimal solutions of an average Markov decision problem [3,4].

References

[1] Ehrenfeucht, A., Mycielski, J., *Positional strategies for mean payoff games* International Journal of Game Theory. **8** (1979), 109–113.

[2] Lozovanu, D., Pickl, S., ”Optimization and Multiobjective Control of Time-Discrete Systems”, Springer, 2009.

[3] Lozovanu, D., Pickl, S., ”Optimization of Stochastic Discrete Systems and Control on Complex Networks”, Springer, 2015.

[4] Puterman, M., ”Markov Decision Processes: Discrete Stochastic Dynamic Programming”, John Wiley, New Jersey, 2005.

[5] Zwik U., Paterson M., *The complexity of mean payoff games on graphs,* TCS **158** (1996), 344-359.

The Complexity of SIMPLE MAX-CUT on Comparability Graphs

Rafael Veiga Pocai [1,2]

Instituto de Matemática e Estatística
Universidade de São Paulo
São Paulo, Brazil

Abstract

We adapt a result by Masuda *et al.* [3] on FIXED LINEAR CROSSING NUMBER to show that SIMPLE MAX-CUT is NP-hard on comparability graphs.

Keywords: max-cut, comparability graphs, computational complexity

1 Introduction

SIMPLE MAX-CUT is the problem of, given a graph $G = (V, E)$, finding a set $U \subset V$ such that $\delta(U)$ has maximum cardinality.

Masuda *et al.* [3] showed that FIXED LINEAR CROSSING NUMBER is NP-hard. The problem consists on minimizing crossings in a drawing of a graph such that the vertices are drawn on a line in a fixed order and the edges are drawn as semicircles. This problem can be interpreted as SIMPLE MAX-CUT on circle graphs, as observed by Buchheim and Zheng [1]. Therefore,

[1] This work was supported by CNPq, Brazil (131103/2014-8)

[2] Email: `rvpocai@ime.usp.br`

Masuda's result also proves that SIMPLE MAX-CUT is NP-hard on circle graphs.

A comparability graph is a graph that can be generated by a partial order $\prec$ over a set V, where the vertices are the elements of V and $u, v \in V$ are adjacent if and only if $u \prec v$ or $v \prec u$. Inspired on Masuda's proof, we show that SIMPLE MAX-CUT is NP-hard also on comparability graphs.

2 Reduction from SET SPLITTING

In SET SPLITTING, we are given a set X and a collection $\mathcal{S}$ of subsets of X, and the goal is to decide whether there is a partition X_1, X_2 of X such that for all $S \in \mathcal{S}$, $S \cap X_1 \neq \emptyset$ and $S \cap X_2 \neq \emptyset$. We call such an instance *affirmative*.

This problem is NP-complete even if the sets in $\mathcal{S}$ have only either two or three elements [2]. Let $(X, \mathcal{S})$ be an instance of this kind, and let $m = |X|$, $n = |\mathcal{S}|$, $n_2 = \{S \in \mathcal{S} : |S| = 2\}$ and $n_3 = \{S \in \mathcal{S} : |S| = 3\}$. We also define $M = \binom{2n_2+6n_3}{2} + 1$.

We construct a comparability graph $G = (V, E)$ such that the answer of SIMPLE MAX-CUT on G tells us whether $(X, \mathcal{S})$ is an affirmative instance or not. The set of vertices V is the following:

- For each set $S \in \mathcal{S}$ such that $S = \{x_p, x_q\}$, we create two disjoint sets A^S_{pq} and A^S_{qp}, where $|A^S_{pq}| = |A^S_{qp}| = M$. This two sets will form a complete bipartite graph called $B^S_{pq} = (A^S_{pq}, A^S_{qp})$.
- For each set $S \in \mathcal{S}$ such that $S = \{x_p, x_q, x_r\}$, we create six disjoint sets A^S_{pq}, A^S_{qp}, A^S_{pr}, A^S_{rp}, A^S_{qr} and A^S_{rq}, where each one has M vertices. From them we will obtain three complete bipartite graphs called $B^S_{pq} = (A^S_{pq}, A^S_{qp})$, $B^S_{pr} = (A^S_{pr}, A^S_{rp})$ and $B^S_{qr} = (A^S_{qr}, A^S_{rq})$.
- For each $x_p \in X$ and for each $S \in \mathcal{S}$:
  · If $S = \{x_p, x_q\}$ then we create the vertex w^S_{pq}.
  · If $S = \{x_p, x_q, x_r\}$ then we create the vertices w^S_{pq} and w^S_{pr}.
- For each $x_p \in X$, we create the Z_p with $2M$ vertices.

We define the partial order relation $\prec$ over V as follows:

- For each $B^S_{pq} = (A^S_{pq}, A^S_{qp})$, and for all $u \in A^S_{pq}$ and $v \in A^S_{qp}$, we define $u \prec v$ if $p < q$ and $v \prec u$ if $q < p$.
- For each w^S_{pq} and $u \in A^S_{pq}$, we define $u \prec w^S_{pq}$ if $p < q$ and $w^S_{pq} \prec u$ if $q < p$.
- For all Z_p, $z \in Z_p$ and w^S_{pq}, we define $z \prec w^S_{pq}$ if $p < q$ and $w^S_{pq} \prec z$ if $q < p$.

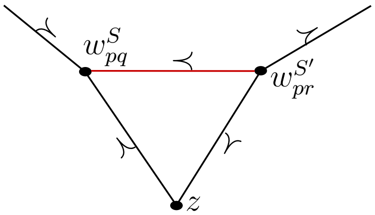


Fig. 1. The red edge was induced by transitivity through vertex z.

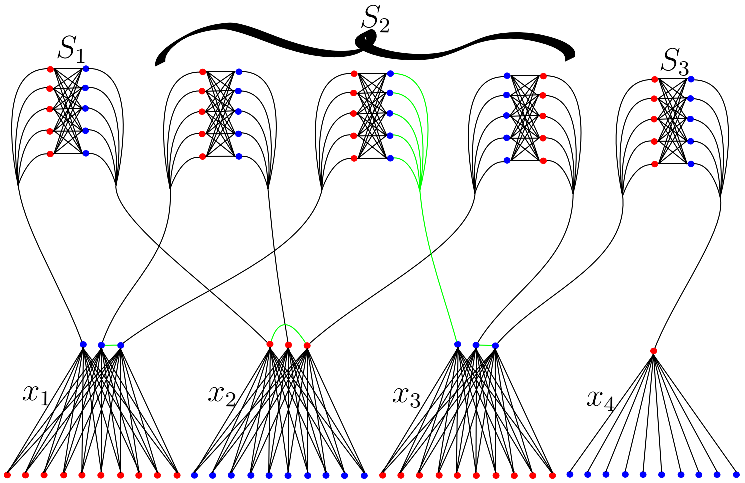


Fig. 2. An example of G constructed from the instance $X = \{x_1, x_2, x_3, x_4\}$ and $\mathcal{S} = \{S_1 = \{x_1, x_2\}, S_2 = \{x_1, x_2, x_3\}, S_3 = \{x_3, x_4\}\}$. The figure also shows the kind of cut presented in the proof of Lemma 2.1, where the green edges are the only not in it.

The graph $G = (V, E)$ is the comparability graph induced by $\prec$. The edges listed above are called *explicit*, and the edges induced by the transitive property are called *implicit*. By construction, the only implicit edges induced by $\prec$ connect vertices w related to the same $x_p \in X$, by transitivity through some vertex $z \in Z_p$, as illustrated by Figure 1. These edges sum at most $\binom{2n_2+6n_3}{2} = M - 1$. Figure 2 shows an example of the construction.

Let $K = M(n_3+1)-1$ and $\text{mc} = \max\{|\delta(U)| : U \subset V\}$. The next lemmas will connect the results of SIMPLE MAX-CUT on G and SET SPLITTING on $(X, \mathcal{S})$.

Lemma 2.1 *If $(X, \mathcal{S})$ is an affirmative instance of SET SPLITTING, then $mc \geq |E| - K$.*

Proof. Let X_1, X_2 be a partition of X that split all the sets in $\mathcal{S}$. Lets construct a set of vertices U such that $|\delta(U)| \geq |E| - K$.

- We start with $U = \emptyset$.
- For each $x_p \in X_1$, we put all w^S_{pq} in U.
- For each $x_p \in X_2$, we put all $z \in Z_p$ in U.
- For each $S = \{x_p, x_q\}$ such that $x_p \in X_1$, we put all elements of A^S_{qp} in U.
- For each $S = \{x_p, x_q, x_r\}$ such that $x_p \in X_1$ and $x_q, x_r \in X_2$, we put all elements of A^S_{qp}, A^S_{qr} and A^S_{rp} in U.
- For each $S = \{x_p, x_q, x_r\}$ such that $x_p, x_q \in X_1$ and $x_r \in X_2$, we put all elements of A^S_{pq}, A^S_{rp} and A^S_{rq} in U.

Figure 2 illustrates the construction. All explicit edges are in the cut, except M edges for each set of three elements in $\mathcal{S}$. Moreover, the implicit edges are also out of the cut. Hence:

$$\mathrm{mc} \geq |\delta(U)| \geq |E| - Mn_3 - \binom{2n_2 + 6n_3}{2} = |E| - M(n_3 + 1) + 1 = |E| - K.$$

□

The next lemma claims the reciprocal, but its proof will be omitted here.

Lemma 2.2 *If $mc \geq |E| - K$, then $(X, \mathcal{S})$ is an affirmative instance of SET SPLITTING.* □

Since the graph G may be constructed in polynomial time, the theorem follows:

Theorem 2.3 *SIMPLE MAX-CUT is NP-hard on comparability graphs.* □

References

[1] Buchheim, C. and L. Zheng, *Fixed linear crossing minimization by reduction to the maximum cut problem*, Lecture Notes in Computer Science **4112** (2006), pp. 507–516.

[2] Garey, M. R. and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness," W. H. Freeman, 1979.

[3] Masuda, S., K. Nakajima, T. Kashiwabara and T. Fujisawa, *Crossing minimization in linear embeddings of graphs*, IEEE Transactions on Computers **39** (1990), pp. 124–127.

Hardness results for stable exchange problems

Zsuzsa Karkus [1,2]

Department of Operations Research
Eötvös Loránd University
Budapest, Hungary

Abstract

In this paper, we study variants of the stable exchange problem which can be viewed as a model for kidney exchange. The b-way stable l-way exchange problem is a generalization of the stable roommates problem. For $b = l = 3$, Biró and McDermid proved that the problem is NP-complete and asked whether a polynomial time algorithm exists for $b = 2,\ l = 3$. We prove that the problem is NP-complete and it is W[1]-hard with the number of 3-cycles in the exchange as a parameter. We answer a question of Biró by proving that it is NP-hard to maximize the number of covered nodes in a stable exchange. We also prove some related results on strong stability, approximation and variants of the problem where we allow chains.

Keywords: stable exchange, kidney exchange, computational complexity

1 Introduction

Given a simple digraph $D = (V, A)$, a set of node-disjoint directed cycles is called an **exchange**. In an instance of a stable exchange problem, every $v \in V$

[1] Research was supported by the Hungarian Scientific Research Fund (OTKA, grant number K109240).

[2] Email: karkuszsuzsi@gmail.com

has a strictly ordered preference list containing the nodes to which there is an arc from v. We say that u gets v in the exchange if uv is an arc of one of the directed cycles in the exchange. We say that $v \in V$ is **covered** by the exchange E if v belongs to a cycle in E. An exchange is called **stable** if there is no directed cycle C such that for each arc $e = uv$ of C, u is not covered by the exchange or u prefers v over what he got in the exchange. An exchange is called **strongly stable** if there is no directed cycle C not in the exchange such that for each arc $e = uv$ of C, u is not covered by the exchange or e is in the exchange or u prefers v over what he got in the exchange. In both cases, the node set of a violating cycle C is called a **blocking coalition**.

An important motivation of this model is kidney exchange. (This was first described in [5].) Currently the best known treatment for kidney failure is transplantation. Since there are a large number of people on the deceased donor waiting list, the more efficient solution is living donation. However, a kidney of a willing living donor is often not suitable for the patient for immunological reasons. Therefore incompatible patient-donor pairs might want to exchange kidneys with other pairs in the same situation.

In the model described above, the nodes of the digraph correspond to the incompatible patient-donor pairs and $uv \in A$ if and only if the kidney of the donor corresponding to v is suitable for the patient corresponding to u. Each patient has a strict preference order over the kidneys suitable for him. In an exchange, the patient-donor pairs exchange kidneys backwards along the cycles.

Shapley and Scarf [7] showed that the stable exchange problem (SE) is always solvable, and a stable exchange can be found by the Top Trading Cycles (TTC) algorithm proposed by Gale.

In case of kidney exchanges, the cycles in the exchange should be short, since all operations along a cycle have to be carried out at the same time (to avoid someone backing out). If all the cycles in the exchange have length at most l, we call it an **l-way exchange**. An exchange is called **b-way stable** if there is no blocking coalition of size at most b. The definition is analogous for strong stability. The 2-way stable 2-way exchange problem is equivalent to the stable roommates with incomplete preference lists problem and hence solvable in polynomial time [3]. Biró and McDermid [1] proved that the problem of deciding whether a 3-way stable 3-way exchange exists is NP-complete, and asked whether a polynomial time algorithm exists for the problem of deciding whether a 2-way stable 3-way exchange exists. In this paper we settle this and other related questions. The proofs are omitted for lack of space, they can be found in [4].

2 Stable exchanges with restrictions and maximizing the number of covered nodes

Theorem 2.1 *The problem of deciding whether a stable l-way exchange exists is NP-complete for any $l \geq 3$. The same holds for b-way (strongly) stable l-way exchanges for any $b \geq 2$.*

Theorem 2.2 *For parameter k, the problem of deciding whether a 2-way stable 3-way exchange with at most k 3-cycles exists is W[1] -hard even in complete digraphs.*

An instance might admit more than one stable exchanges; therefore, it is a natural goal to maximize the number of covered nodes in the exchange. The complexity of this problem was mentioned as an open problem in [2] as well as the same question for 2-way stable exchanges.

Theorem 2.3 *It is NP-complete to decide if an instance of the stable exchange problem admits a complete stable exchange.*

Roth and Postlewaite [6] proved that the exchange found by the TTC algorithm is the only strongly stable solution. However, there might be more then one b-way strongly stable exchanges.

Theorem 2.4 *It is NP-complete to decide if an instance of the stable exchange problem admits a complete b-way (strongly) stable exchange for any $b \geq 2$.*

Theorem 2.5 *If the digraph is symmetric, then TTC is a $\frac{1}{2}$-approximation algorithm, while the stable partition algorithm [8] is a $\frac{2}{3}$-approximation algorithm for maximizing the number of covered nodes in a 2-way (strongly) stable exchange.*

3 Allowing chains

A recent innovation in kidney exchange is allowing chains. There are altruists who are willing to donate one of their kidneys to any patient who needs it. In an exchange with chains we allow chains ending in an altruist besides cycles. The cycles should be short in practice, however, the chains might be longer since the operations along a chain do not necessarily have to be carried out simultaneously (although it is desirable).

In this section we study the problem of deciding whether a 2-way stable pairwise exchange with chains exists. Pairwise means that besides the chains only 2-cycles are allowed. (The 2-way stable l-way exchange with chains problem is NP-hard since its special case, where there are no altruist nodes is NP-complete.)

Theorem 3.1 *The problem of deciding whether a 2-way stable pairwise exchange with chains exists is NP-complete, even if the number of altruists is restricted to 1.*

Theorem 3.2 *The problem of deciding whether a 2-way stable pairwise exchange with chains exists, where the lengths of the chains are at most l is NP-complete for any given $l \geq 1$.*

Theorem 3.3 *The 2-way stable pairwise exchange with chains problem is $W[1]$-hard even if the number of altruists is restricted to 1, if the parameter is the length of the longest chain in the exchange.*

Acknowledgement. The author would like to thank Tamás Király for his helpful suggestions and comments.

References

[1] P. Biró and E. McDermid, *Three-Sided Stable Matchings with Cyclic Preferences*, Algorithmica, **58(1)** (2010), 5–18.

[2] P. Biró, "The stable matching problem and its generalizations: an algorithmic and game theoretical approach,“ Ph.D. thesis, Budapest University of Technology and Economics, 2007.

[3] R. W. Irving and D.F. Manlove, *The stable roommates problem with ties*, Journal of Algorithms, **43(1)** (2002), 85–105.

[4] Zs. Karkus, "Stable exchanges,“ M.Sc. thesis, Eötvös Loránd University Budapest, 2015.

[5] A. E. Roth, T. Sönmez and U. M. Ünver, *Kidney exchange*, Quarterly Journal of Economics, **119** (2004), 457-488.

[6] A. Roth and A. Postlewaite, *Weak versus strong domination in a market with indivisible goods*, Journal of Mathematical Economics **4(2)** (1977), 131–137.

[7] L. S. Shapley and H. E. Scarf, *On cores and indivisibility*, Journal of mathematical economics, **1(1)** (1974), 23-37.

[8] J.J.M. Tan, *A necessary and sufficient condition for the existence of a complete stable matching*, Journal of Algorithms, **12** (1991), 154–178.

On the Complexity of the P_3-Hull Number of the Cartesian Product of Graphs [3]

Julliano R. Nascimento [a,1], Erika M. M. Coelho [a,1], Hebert Coelho [a,1], Jayme L. Szwarcfiter [b,c,2]

[a] *Instituto de Informática, Universidade Federal de Goiás, GO, Brazil*

[b] *IM, COPPE, and NCE, Universidade Federal do Rio de Janeiro, RJ, Brazil*

[c] *Instituto Nacional de Metrologia, Qualidade e Tecnologia, RJ, Brazil*

Abstract

Let G be a finite, simple, and undirected graph and let S be a set of vertices of G. If no vertex of G that does not belong to S has two neighbors in S, then S is P_3-convex. The P_3-convex hull $H(S)$ of S is the smallest P_3-convex set containing S. If $H(S) = V(G)$ we say that S is a P_3-hull set of G. The cardinality $h(G)$ of a minimum P_3-hull set in G is called the P_3-hull number of G. In this paper we extend the result of Centeno et al. [2] showing that, given a graph G and an integer k, deciding whether $h(G) \leq k$ remains NP-complete for the Cartesian product of graphs.

Keywords: P_3-convexity, P_3-hull number, Cartesian product.

[1] Emails: {erikamorais,hebert,jullianorosanascimento}@inf.ufg.br

[2] Email: jayme@nce.ufrj.br

[3] The authors are partially supported by CAPES, CNPq, and FAPERJ.

1 Introduction

The spread disease on a square grid [1] is an example of problem in which the P_3-hull number of Cartesian product of graphs can be applied. In a square grid some cells are infected. Iteratively, an uninfected cell becomes infected if at least two of its neighbors are so. What is the minimum number of originally infected cells to guarantee that all cells of the grid become eventually infected?

In this paper we consider finite, simple and undirected graphs and we use the standard terminology. Let G be a graph with vertex set $V(G)$. A *graph convexity* on $V(G)$ is a collection $\mathcal{C}$ of subsets of $V(G)$ such that $\emptyset, V(G) \in \mathcal{C}$ and $\mathcal{C}$ is closed under intersections. The sets in $\mathcal{C}$ are called *convex sets*.

Many convexities in graphs are defined by a set $\mathcal{P}$ of paths in G, such that a set S of vertices of G is convex if and only if for every path $P : v_0 v_1 \ldots v_l$ in $\mathcal{P}$ such that v_0 and v_l belong to S, all vertices of P belong to S. If we define $\mathcal{P}$ as the set of, all shortest paths in G, all induced paths of G and all paths of G with 3 vertices, we have the *geodetic convexity* [5], the *monophonic convexity* [4], and the P_3-*convexity* [3], respectively.

In this paper we study the P_3-convexity $\mathcal{C}$ on a graph G. Given a set $S \subseteq V(G)$, the P_3-*interval* $I[S]$ of S is formed by S with every vertex outside S with at least two neighbors in S. If $I[S] = S$, then the set S is P_3-*convex*. The P_3-*convex hull* $H(S)$ of S is the smallest P_3-convex set containing S. If $H(S) = V(G)$ we say that S is a P_3-*hull set* of G. The cardinality $h(G)$ of a minimum P_3-hull set in G is called the P_3-*hull number* of G. Some results on the P_3-hull number follow in [2,7].

Our motivation comes from the infection problem presented at the first paragraph and from a work of Centeno et al. [2]. They proved that, given a graph G and an integer k, to decide whether $h(G) \leq k$ is NP-complete. We extend their result showing that the same problem remains NP-complete for the Cartesian product of graphs. The Section 2 contains our result.

2 Results

The *Cartesian product* of two graphs G_1 and G_2, denoted by $G_1 \square G_2$, is the graph with vertex set $V(G_1 \square G_2) = V(G_1) \times V(G_2)$ and edge set $E(G_1 \square G_2)$ satisfying the following condition: $(u, u')(v, v') \in E(G_1 \square G_2)$ if and only if either $u = v$ and $u'v' \in E(G_2)$ or $u' = v'$ and $uv \in E(G_1)$.

Let G_1 and G_2 be two graphs with vertex sets $V(G_1) = \{u_1, \ldots, u_m\}$ and $V(G_2) = \{v_1, \ldots, v_n\}$, respectively. We refer to *column* C_j, for $1 \leq j \leq n$, the subset of vertices $\{(u_1, v_j), \ldots, (u_m, v_j)\}$ of $V(G_1 \square G_2)$.

Problem 2.1 SAT-AM3 *[6]*
Instance: *A set $\mathcal{F} = \{F_1, F_2, \ldots, F_m\}$ of clauses, built on a finite set $\mathcal{U} = \{x_1, x_2, \ldots, x_n\}$ of boolean variables, such that each clause contains at most three literals and each variable appears at most three times.*
Question: *Is there a truth assignment to the variables in $\mathcal{U}$ satisfying $\mathcal{F}$?*

Problem 2.2 P_3-HULL NUMBER *[2]*
Instance: *A graph G and an integer $k \geq 0$.*
Question: *Does G have a P_3-hull set S, such that $|S| \leq k$?*

Theorem 2.3 *The P_3-*HULL NUMBER *remains NP-complete on Cartesian product of graphs.*

Proof (Sketch) According to Centeno et al. [2] the problem for general graphs is in NP, then the same problem with the restriction of the graph considered be the result of a Cartesian product is also in NP.

The problem SAT-AM3 is NP-complete [6]. Centeno et al. [2] perform a reduction from SAT-AM3, assuming that, for every $1 \leq i \leq n$, each of the two literals x_i and $\overline{x}_i$ is contained in exactly one or two of the clauses of $\mathcal{F}$. We proceed in the same way. From an instance ϕ of SAT-AM3 we first construct a graph G such that $h(G) = 7n$.

For every $1 \leq i \leq n$, we add to G a variable gadget $G(x_i)$, as in Figure 1. For every $1 \leq j \leq m$, we add to G a clause gadget $G(F_j)$, which is a complete bipartite graph. The first partite set of $G(F_j)$ consists of the vertices $F_j(1)$ and $F_j(2)$. For every literal x that occurs in F_j and is contained in exactly one clause of ϕ, the second partite set of $G(F_j)$ contains two vertices $F_j(x,1)$ and $F_j(x,2)$, and we add to G the six edges $x(1)F_j(1)$, $x(2)F_j(1)$, $x(3)F_j(2)$, $x(4)F_j(2)$, $x(5)F_j(x,1)$, $x(5)F_j(x,2)$. Similarly, for every literal x that occurs in F_j and is contained in exactly two clauses of ϕ, the second partite set of $G(F_j)$ contains one vertex $F_j(x,1)$, and we add to G the five edges $x(1)F_j(1)$, $x(2)F_j(1)$, $x(3)F_j(2)$, $x(4)F_j(2)$, $x(5)F_j(x,1)$.

To complete our construction, we use the graph G, to create a graph $G\Box K_2$, where $V(K_2) = \{v_1, v_2\}$, such that an instance ϕ of SAT-AM3 is satisfiable if and only if $h(G\Box K_2) \leq k$, where $k = 7n$. We denote by $X_i^{v_j}$ ($\overline{X}_i^{v_j}$) the set X_i ($\overline{X}_i$) of the variable gadget, for $1 \leq i \leq n$, that belongs to the column C_j, $j \in \{1, 2\}$.

Suppose that ϕ is satisfiable and let $\mathcal{A}$ be an assignment that turns $\mathcal{F}$ true. Let $S' = \{(x_i(p), v_1), (\overline{x}_i(p), v_1) : 7 \leq p \leq 9, 1 \leq i \leq n\}$. Consider $S \subseteq V(G\Box K_2)$ such that S contains the set $(S' \setminus \{(x_1(7), v_1)\}) \cup \{(x_1(7), v_2)\}$. For every literal x that is true in $\mathcal{A}$, add the vertex $(x(0), v_1)$ in S. Note that

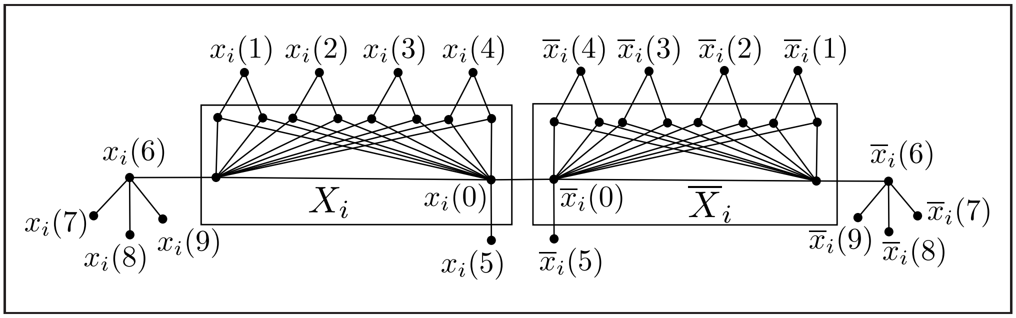


Fig. 1. The variable gadget $G(x_i)$.

$|S| = 7n$. It is possible to prove that the set S is a P_3-hull set of $G\Box K_2$.

For the converse, suppose that $G\Box K_2$ has a P_3-hull set S with $7n$ vertices. Since the set $\mathcal{X}_i = X_i^{v_1} \cup X_i^{v_2} \cup \overline{X}_i^{v_1} \cup \overline{X}_i^{v_2}$, for $1 \leq i \leq n$, is P_3-convex, then S must contain at least one vertex of each $\mathcal{X}_i$, $1 \leq i \leq n$. Defining a truth assignment $\mathcal{A}$ where x_i is true if and only if S has a vertex of $X_i^{v_1} \cup X_i^{v_2}$, for every $1 \leq i \leq n$, it is possible to show that $\mathcal{A}$ is a valid truth assignment.

□

References

[1] Bollobás, B., "The art of mathematics: Coffee time in Memphis," Cambridge University Press, 2006.

[2] Centeno, C. C., M. C. Dourado, L. D. Penso, D. Rautenbach and J. L. Szwarcfiter, *Irreversible conversion of graphs*, Theoretical Computer Science **412** (2011), pp. 3693–3700.

[3] Centeno, C. C., M. C. Dourado and J. L. Szwarcfiter, *On the convexity of paths of length two in undirected graphs*, Electronic Notes in Discrete Mathematics **32** (2009), pp. 11–18.

[4] Duchet, P., *Convex sets in graphs, ii. minimal path convexity*, Journal of Combinatorial Theory, Series B **44** (1988), pp. 307–316.

[5] Everett, M. G. and S. B. Seidman, *The hull number of a graph*, Discrete Mathematics **57** (1985), pp. 217–223.

[6] Garey, M. R. and D. S. Johnson, "Computers and intractability: a guide to NP-completeness," W.H. Freeman and Company, 1979.

[7] Penso, L. D., F. Protti, D. Rautenbach and U. dos Santos Souza, *Complexity analysis of p3-convexity problems on bounded-degree and planar graphs*, Theoretical Computer Science **607** (2015), pp. 83–95.

A lower bound for the sum of the two largest signless Laplacian eigenvalues

Carla Silva Oliveira[1]

Department of Mathematical and Statistics
National School of Statistics
Rio de Janeiro, Brasil

Leonardo de Lima[2]

Department of Production Engineering
Federal Center of Technological Education Celso Suckow da Fonseca
Rio de Janeiro, Brasil

Abstract

Let G be a connected graph of order $n \geq 3$ and let $Q(G) = D(G) + A(G)$ be the signless Laplacian of G, where $A(G)$ is the adjacency matrix and $D(G)$ is the diagonal matrix of the row-sums of $A(G)$. Write $q_1(G)$ and $q_2(G)$ for the two largest eigenvalues of $Q(G)$. In this paper, we obtain a lower bound to the sum of the two $Q-$largest eigenvalues, that is, $q_1(G) + q_2(G) \geq d_1(G) + d_2(G) + 1$ with equality if and only if G is the star S_n or the complete graph K_3, where d_i is the $i-$largest degree of a vertex of G.

Keywords: Signless Laplacian, two largest eigenvalues, sequence degree, lower bound

[1] Email: `carla.oliveira@ibge.gov.br`

[2] Email: `leonardo.lima@cefet-rj.br`

1 Introduction

Let $G(V,E)$ be a simple graph on n vertices. Define $N(u)$ as the set of neighbors of a vertex $u \in V$ and $|N(u)|$ its cardinality. The sequence degree of G is denoted by $d(G) = (d_1(G), d_2(G), \ldots, d_n(G))$, such that $d_i(G) = |N(v_i)|$ is the degree of the vertex $v_i \in V$ and $d_1(G) \geq d_2(G) \geq \ldots \geq d_n(G)$. Write A for the adjacency matrix of G and let D be the diagonal matrix of the row-sums of A, i.e., the degrees of G. The matrix $Q(G) = A + D$ is called the *signless Laplacian* or the Q-matrix of G. As usual, we shall index the eigenvalues of $Q(G)$ in non-increasing order and denote them as $q_1(G) \geq q_2(G) \geq \ldots \geq q_n(G)$, respectively. In [1], Cvetkovic, Rowlinson and Simic proved that $q_1(G) \geq d_1(G) + 1$ and Das in [3] proved that $q_2(G) \geq d_2(G) - 1$. From these two previous results, an obvious lower bound to the sum of the two largest $Q-$eigenvalues of G is obtained. In this paper, we improve that bound to $q_1(G) + q_2(G) \geq d_1(G) + d_2(G) + 1$, which is proved in Section 3. Some useful results to our purposes are presented in Section 2. We denote the following graphs on n vertices: the complete graph K_n; the star S_n and the complete bipartite graph K_{n_1,n_2}, such that $n_1 \geq n_2$ and $n = n_1 + n_2$.

2 Preliminary results

First, we recall the Interlacing Theorem that is a classical result in matrix theory and can be found at Horn and Johnson [4] (in Theorem 4.3.8). The Interlacing Theorem version to the signless Laplacian matrix holds in a specify way which is proved in [5] and stated here in Theorem 2.1. In [2], the authors proved an edge removal version of Interlacing Theorem for the Q-eigenvalues by using line graphs. Notice that the usage of both Interlacing Theorems play an important role in our proofs.

Theorem 2.1 **([5])** *Let G be a graph of order n and $v \in V$. Then for $i = 1, \ldots, n-1$, $q_{i+1}(G) - 1 \leq q_i(G - v) \leq q_i(G)$, where the right inequality holds if and only if v is an isolated vertex.*

Let u and v be the vertices with the two largest degrees of a graph G, that is, $|N(u)| = d_1(G)$ and $|N(v)| = d_2(G)$. A subgraph $H(V_H, E_H)$ of G can be obtained by taking the vertex set as $V_H = \{v_i \in V \,|\, v_i \in N(u) \cup N(v) \cup \{u\} \cup \{v\}\}$ and the edge set as $E_H = \{(v_i, v_j) \in E \,|\, v_i \in \{u, v\}$ and $v_j \in N(u) \cup N(v)\}$. The proof of our main result in Theorem 3.1 follows from the fact that $q_1(G) + q_2(G) \geq q_1(H) + q_2(H)$ by the Interlacing Theorem 2.1 and its edge version, and also that $d_1(G) + d_2(G) = d_1(H) + d_2(H)$ since we did not remove any vertex from $N(u)$ and $N(v)$ of G to build the graph H. In fact, if we prove that $q_1(H)+q_2(H) \geq d_1(H)+d_2(H)+1$, we are done. Before proving that, we need to introduce some notation and two types of graphs obtained by the definition of $H(V_H, E_H)$. Let $S_1 = N(u) \setminus (N(v) \cup v)$, $S_2 = N(u) \cap N(v)$ and $S_3 = N(v) \setminus (N(u) \cup u)$, such that $|S_1| = r$, $|S_2| = p$ and $|S_3| = s$. If u and v are not adjacent, H is denoted by $\mathcal{H}(p, r, s)$ with $d_1(G) = d_1(H) = p+r$ and $d_2(G) = d_2(H) = p+s$. If u and v are adjacent, H is denoted by $\mathcal{G}(p, r, s)$ with $d_1(G) = d_1(H) = p+r+1$ and $d_2(G) = d_2(H) = p+s+1$. Now, consider Lemmas 2.2 and 2.3 which establish lower bounds to $q_1(G)$ and $q_2(G)$ in terms of $d_1(G)$ and $d_2(G)$.

Lemma 2.2 **([1])** *Let G be a connected graph on $n \geq 4$ vertices. Then, $q_1(G) \geq d_1(G) + 1$ with equality if and only if G is the star S_n.*

Lemma 2.3 **([3])** *Let G be a graph. Then $q_2(G) \geq d_2(G) - 1$.*

We improve the lower bounds of the previous lemmas for the graphs $\mathcal{H}(p, r, s)$ and $\mathcal{G}(p, r, s)$ in Propositions 2.4 and 2.5. The proofs are based on the quotient matrix according to the partition $\pi = (u, v, S_2, S_1, S_3)$ of the matrix $Q(G)$. Such results are crucial to proof Theorem 3.1.

Proposition 2.4 *Let $G \in \mathcal{H}(p, r, s)$ be a graph on $n \geq 3$ vertices with $p \geq 1$ and $r \geq 1$. If $r \geq s \geq 1$, then $q_2(G) > d_2(G)$; otherwise, if $s = 0$, then $q_2(G) \geq d_2(G)$ with equality if and only if $G = P_4$.*

Proposition 2.5 *Let $G \in \mathcal{G}(p, r, s)$ be a graph on $n \geq 3$.*

(i) *If $r = p = 1$ and $s = 0$, then $q_2(G) = d_2(G)$;*

(ii) *If $p = 0, r, s \geq 1$, then $q_1(G) + q_2(G) > d_1(G) + d_2(G) + 1$;*

(iii) *If $p = 1$ and $r = s$, then $q_1(G) > d_1(G) + \frac{3}{2}$ and $q_2(G) > d_2(G) - \frac{1}{2}$;*

(iv) *If $p \geq 2$ and $r = s$, then $q_1(G) > d_1(G) + 2$;*

(v) *If $p \geq 1$ and $r \geq s + 3$, then $q_2(G) > d_2(G)$;*

(vi) *If $p \geq 1$ and $r \in \{s + 1, s + 2\}$, then $q_1(G) > d_1(G) + 1 + \frac{p}{n}$ and $q_2(G) > d_2(G) - \frac{p}{n}$.*

3 Main result

In this section, we prove the main result of the paper.

Theorem 3.1 *Let G be a simple connected graph on $n \geq 3$ vertices. Then $q_1(G) + q_2(G) \geq d_1(G) + d_2(G) + 1$ with equality if and only if G is one of the following graphs: the complete graph K_3 or a star S_n.*

Proof. Let G be a simple connected graph on $n \geq 3$ vertices. Assume that u and v are the vertices with largest and second largest degrees of G, i.e., $d(u) = d_1(G)$ and $d(v) = d_2(G)$. Take H as a subgraph of G containing u and v such that H is isomorphic to $\mathcal{H}(p, r, s)$ or $\mathcal{G}(p, r, s)$. Note that $d_1(G)+d_2(G) = d_1(H)+d_2(H)$ and from Interlacing Theorem 2.1 and its edge version, $q_1(G)+q_2(G) \geq q_1(H) + q_2(H)$. Firstly, suppose that H is isomorphic to $\mathcal{H}(p, r, s)$. Since G is connected, the cases $p = 0$ with any r and s are not possible. If $p = 1$ and $r = s = 0$, then $H = \mathcal{H}(1, 0, 0) = S_3$ and $q_1(H) + q_2(H) = 4 = d_1(H) + d_2(H) + 1$. If $p \geq 2$ and $r = s = 0$, then $H = \mathcal{H}(p, 0, 0) = K_{2,p}$ and $q_1(H)+q_2(H) = 2p+2 > d_1(H)+d_2(H)+1 = 2p+1$. If $p, r \geq 1$ and $s = 0$, from Proposition 2.4 and Lemma 2.2, we get $q_1(H) + q_2(H) > d_1(H) + d_2(H) + 1$. Now, if $p \geq 1$ and $r \geq s \geq 1$, from Proposition 2.4 and Lemma 2.2, follows that $q_1(H)+q_2(H) > d_1(H)+d_2(H)+1$. Now, suppose that H is isomorphic to $\mathcal{G}(p, r, s)$. If $p = s = 0$ and $r \geq 1$, $H = \mathcal{G}(0, r, 0) = S_{r+2}$ and $q_1(H) + q_2(H) = r + 3 = d_1(H) + d_2(H) + 1$. If $p = 0$ and $r, s \geq 1$, the result follows from Proposition 2.5. If $p = 1$ and $r = s = 0$, then H is the complete graph K_3 and $q_1(H) + q_2(H) = 5 = d_1(H) + d_2(H) + 1$. If $p \geq 2$ and $r = s = 0$, then $H = \mathcal{G}(p, 0, 0) = K_2 \vee \overline{K_p}$, i.e., the complete split graph, and it is well-known that $q_1(H) = (n+2+\sqrt{n^2 + 4n - 12})/2$ and $q_2(H) = n-2$. It is easy to check that for $p \geq 2$, we have $q_1(H) + q_2(H) > d_1(H) + d_2(H) + 1$. If $p \geq 1, r \geq 1$ and $s \geq 0$, the result of the theorem follows from Proposition 2.5 and Lemmas 2.2 and 2.3. From the cases above, the equality conditions are restricted to the graphs K_3 and S_n and the result follows.

□

References

[1] Cvetković D., Rowlinson P. and Simic S., *Eigenvalue bounds for the signless Laplacian*, Publ. Inst. Math. Beograd 81(95), (2007), 11–27.

[2] D. Cvetković, P. Rowlinson, S.K. Simic, *Signless Laplacians of finite graphs*, *Linear Algebra Appl.*, 423 (1) (2007), 155–171.

[3] Das K. C., *On conjectures involving second Largest signless Laplacian eigenvalue of graphs*, Linear Algebra Appl. 432, (2010), 3018–3029.

[4] Horn R. and Johnson C., Matrix Analysis, Cambridge University Press, Cambridge, 1985, xiii+561 pp.

[5] Wang J. and Belardo F., *A note on the signless Laplacian eigenvalues of graphs*, Linear Algebra Appl. 435 (10), (2011), 2585–2590.

On Three Extensions of Equimatchable Graphs

Zakir Deniz[a], Tınaz Ekim[b], Tatiana Romina Hartinger[c],
Martin Milanič[c], Mordechai Shalom[d]

[a] *Department of Mathematics, Süleyman Demirel University, Isparta, Turkey*
`zakirdeniz@sdu.edu.tr`

[b] *Department of Industrial Engineering, Boğaziçi University, Istanbul, Turkey*
`tinaz.ekim@boun.edu.tr`

[c] *University of Primorska, UP IAM and UP FAMNIT, Koper, Slovenia*
`tatiana.hartinger@iam.upr.si`, `martin.milanic@upr.si`

[d] *TelHai College, Upper Galilee, 12210, Israel*
`cmshalom@telhai.ac.il`

Abstract

A graph is said to be equimatchable if all its maximal matchings are of the same size. In this work we introduce three extensions of the property of equimatchability and present some initial structural and algorithmic insights about them.

Keywords: matching, equimatchable graph, greedy algorithm, equimatchable set, matching gap

1 Introduction

A *matching* is a set of pairwise disjoint edges in a graph. A matching is said to be *maximal* if it is not contained in any other matching and *maximum* if it is of maximum size. Given a graph G, we denote by $\nu(G)$ and $\beta(G)$ the sizes of a maximum matching and of a minimum maximal matching of G, respectively.

A graph G is *equimatchable* if all its maximal matchings are of the same size, that is, if $\beta(G) = \nu(G)$. Equimatchable graphs were first introduced and characterized in [3] and it has been shown in [2] that they are polynomially recognizable. In this work we generalize the property of equimatchability of graphs in three different ways: to a weighted case related to a greedy algorithm and by introducing two graph parameters measuring how far a graph is from being equimatchable.

2 Greedy equimatchable graphs

Our first generalization is related to a greedy algorithm and motivated by applications in routing in wireless networks. Consider an edge-weighted graph (G, w) (with $w : E(G) \to \mathbb{R}_+$) and the greedy algorithm that forms a maximal matching by starting with the empty matching and iteratively adding to it an edge of maximum possible weight. A maximal matching M of G is said to be a *greedy matching* of (G, w) if it can be chosen by the greedy algorithm. An edge-weighted graph is *t-greedy equimatchable* if the weight of every greedy matching of it is t, *greedy equimatchable* if it is t-greedy equimatchable for some t, and *strongly greedy equimatchable* if it is $\nu(G, w)$-greedy equimatchable, where $\nu(G, w)$ denotes the maximum w-weight of a matching in G. Note that t, if exists, is uniquely determined by (G, w). Given a weighted graph (G, w) and a real number ρ, we denote by G_ρ the subgraph of G consisting of all the edges of G of weight ρ and their endpoints.

In the study of greedy equimatchable and strongly greedy equimatchable graphs we may, without loss of generality, restrict our attention to weight functions with the smallest weight equal to 1. The special case when all the weights are equal to 1 coincides with the notion of equimatchability. While not every graph is an induced subgraph of an equimatchable graph [1], every graph G can be turned into a strongly greedy equimatchable graph, simply by assigning weight 2 to every edge of a fixed maximum matching, and weight 1 to every other edge. On the other hand, using the following sufficient condition for strong greedy equimatchability with weights 1 and 2, we can show that every graph H equals G_ρ for some strongly greedy equimatchable graph (G, w) with $w : E(G) \to \{1, 2\}$ and $\rho = 2$. A set S of vertices in a graph G is said to be *covered by* a matching M if $S \subseteq V(M)$, and *strongly matching-covered* if it is covered by every maximal matching of G.

Proposition 2.1 *Let $G = (V, E)$ be a graph that has a vertex cover S and a set $F \subseteq E(G[S])$ such that for every maximal matching M in the graph (S, F),* *the set $S - V(M)$ is a strongly matching-covered independent set of $G - V(M)$. Let $w : E \to \{1,2\}$ be the weight function such that $w(e) = 2$ if and only if $e \in F$. Then (G,w) is strongly greedy equimatchable, with $\nu(G,w) = |S|$.*

A *randomly matchable* graph is a graph G in which every maximal matching is perfect (that is, covers $V(G)$). Given a graph H, let G be any graph such that $V(G) = S \cup T$, $S \cap T = \emptyset$, $G[S] = H$, $|S| = |T|$, T is independent, and the bipartite graph defined by the edges between S and T is randomly matchable. Then, conditions in Proposition 2.1 apply with G, S, and $F = E(G[S])$.

The general case of greedy equimatchable graphs with two distinct weights is considered in the following theorem.

Theorem 2.2 *Let $\rho > 1$ and let (G,w) be a weighted graph with $w : E(G) \to \{1,\rho\}$. For every $t \geq 0$, the following two statements are equivalent:*
(i) *(G,w) is t-greedy equimatchable.*
(ii) *For every maximal matching M of G_ρ we have $\nu(G - V(M)) = \beta(G - V(M)) = t - \rho \cdot |M|$, and if G_ρ is not equimatchable, then $\rho = 2$.*
Moreover, if $\rho \geq 2$, then (G,w) is greedy equimatchable if and only if it is strongly greedy equimatchable.

By the last part of the above theorem, if G is a $\{1,\rho\}$-edge-weighted graph that is greedy equimatchable but not strongly greedy equimatchable, then $\rho < 2$. Three small examples of such weighted graphs can be obtained as follows. Let $1 < \rho < 2$, let G be a graph with $V(G) = \{a,b,c,d\}$, $\{ab,bc,cd\} \subseteq E(G)$, $ad \notin E(G)$, and let $w(bc) = \rho$ and $w(e) = 1$ for all $e \in E(G) \setminus \{bc\}$. Each of the so-obtained graphs G contains a *semi-induced* P_4, that is, a 4-vertex path the endpoints of which are non-adjacent in G. This is not a coincidence:

Theorem 2.3 *Let G be a graph with no semi-induced P_4 (equivalently: every component of G is either complete or complete bipartite) and let $w : E(G) \to \{1,\rho\}$ with $\rho > 1$. Then (G,w) is greedy equimatchable, if and only if (G,w) is strongly greedy equimatchable, if and only if G_ρ is equimatchable.*

3 Equimatchable sets

Given a graph G, we say that a set $S \subseteq V(G)$ is *equimatchable (in G)* if all maximal matchings of G that cover S are of the same size. We denote by $\eta(G)$ the minimum size of an equimatchable set in G. Clearly, a graph G is equimatchable if and only if $\eta(G) = 0$. Below we give a hitting set formulation of this parameter and some complexity results related to its computation. A *second best* matching in a graph G is a maximal matching of size $\nu(G) - 1$. For a graph G, let $\mathrm{Exp}_2(G)$ be the hypergraph with vertex set $V(G)$ and hyperedge set $\{\mathrm{Exp}(M) : M \text{ is a second best matching of } G\}$ where $\mathrm{Exp}(M) = V(G) \setminus V(M)$.

Proposition 3.1 *For every graph G, a set $S \subseteq V(G)$ is equimatchable if and only if S is a hitting set of the hypergraph $\mathrm{Exp}_2(G)$. In particular, $\eta(G)$ equals the minimum size of a hitting set of $\mathrm{Exp}_2(G)$.*

Theorem 3.2 *Computing $\eta(G)$ for a given graph G is APX-hard (and consequently also NP-hard). For each constant k, testing if $\eta(G) \leq k$ is polynomial.*

4 The matching gap of a graph

For a graph G, we define the *matching gap* of G as the quantity $\nu(G) - \beta(G)$ and denote it by $\mu(G)$. Clearly, for every graph G we have $\mu(G) \geq 0$, with equality if and only if G is equimatchable. In the following two theorems, we characterize graphs with $\mu(G) = 1$ and give bounds relating parameters η and μ to each other and to the matching number. For an (independent) set S, we say that a matching M *isolates* S if every vertex of S is a component of $G - V(M)$.

Theorem 4.1 *A graph G has $\mu(G) = 1$ if and only if G has a semi-induced $P_4 := u, w, y, v$ and a matching M isolating $\{u, v\}$ and containing wy, and there exists an integer m such that for each semi-induced $P_4 := u, w, y, v$ of G, every maximal matching of $G - V(P_4)$ saturating $N(\{u, v\}) - V(P_4)$ is of size m.*

Theorem 4.2 *For every graph G with at least one edge, we have $\mu(G) \leq \eta(G) \leq 2\nu(G) - 2$. Each of the two inequalities is achieved with equality by graphs with arbitrarily large values of $\eta(G)$.*

References

[1] Dibek, C., T. Ekim, D. Gozupek, and M. Shalom, *Equimatchable graphs are C_{2k+1}-free for $k \geq 4$*, submitted to Discrete Mathematics.

[2] Lesk, M., M.D. Plummer, and W.R. Pulleyblank, *Equi-matchable graphs*, Graph theory and combinatorics (Cambridge, 1983), 239–254, Academic Press, London, 1984.

[3] Lewin, M., *Matching-perfect and cover-perfect graphs*, Israel J. Math., **18(4)** (1974), 345–347.

Diameter, minimum degree and hyperbolicity constant in graphs

Verónica Hernández [1,2]

Department of Mathematics
Universidad Carlos III de Madrid
Madrid, Spain

Domingo Pestana [1,3]

Department of Mathematics
Universidad Carlos III de Madrid
Madrid, Spain

José Manuel Rodríguez [1,4]

Department of Mathematics
Universidad Carlos III de Madrid
Madrid, Spain

Abstract

In this work, we obtain good upper bounds for the diameter of any graph in terms of its minimum degree and its order, improving a classical theorem due to Erdös, Pach, Pollack and Tuza. We use these bounds in order to study hyperbolic graphs (in the Gromov sense). Since computing the hyperbolicity constant is an almost intractable problem, it is natural to try to bound it in terms of some parameters of the graph. Let $\mathcal{H}(n, \delta_0)$ be the set of graphs G with n vertices and minimum degree δ_0. We study $a(n, \delta_0) := \min\{\delta(G) \mid G \in \mathcal{H}(n, \delta_0)\}$ and $b(n, \delta_0) := \max\{\delta(G) \mid G \in$

$\mathcal{H}(n,\delta_0)\}$. In particular, we obtain bounds for $b(n,\delta_0)$ and we compute the precise value of $a(n,\delta_0)$ for all values of n and δ_0.

Keywords: Diameter, minimum degree, finite graphs, Gromov hyperbolicity, hyperbolicity constant.

1 Introduction

All graphs considered in this paper are undirected, connected and simple. Let us denote by $G = (V,E)$ a graph such that every edge has length equal to 1. Here $V = V(G)$ denotes the set of vertices of G and $E = E(G)$ the set of edges of G. The degree of $v \in V(G)$ is the number of edges incident to the vertex and is denoted $\deg(v)$. The *diameter* of a graph is defined as $\operatorname{diam}(G) := \max\{d(x,y) \mid (x,y) \in G\}$, while the *diameter of the vertices* of a graph is defined as $\operatorname{diam} V(G) := \max\{d(x,y) \mid (x,y) \in V(G)\}$. The maximum and minimum degree of a graph G are $\Delta := \max\{\deg(v) \mid v \in V(G)\}$, $\delta_0 := \min\{\deg(v) \mid v \in V(G)\}$.

In the design of communication networks, it is common to take into account limitations on the vertex degrees and the diameter. Throughout the years, problems related with the diameter and degree have attracted the attention of many researchers and they have numerous applications (see [5] for an overview on results related to this topic).

In this paper, we focus on improving a result due to Erdös, Pach, Pollack and Tuza (see [2]), which gives an asymptotically sharp upper bound for the diameter of a connected graph in terms of its minimum degree and its order.

On the other hand, on the second part of this work, we deal with hyperbolic graphs in the Gromov sense.

Gromov hyperbolicity was introduced by Mikhail Leonidovich Gromov in the setting of geometric group theory, but has played an increasing role in analysis on general metric, with applications to the Martin boundary, invariant metrics in several complex variables and extendability of Lipschitz mappings. The concept of hyperbolicity appears also in discrete mathematics, algorithms and networking. Another important application of these spaces is the secure

[1] Supported in part by a grant from Ministerio de Economía y Competitividad (MTM 2013-46374-P), Spain.

[2] Email: `vehernan@math.uc3m.es`
[3] Email: `dompes@math.uc3m.es`
[4] Email: `jomaro@math.uc3m.es`

transmission of information by internet. For detailed expositions about Gromov hyperbolicity, see e.g. [7], [3].

The study of mathematical properties of Gromov hyperbolic spaces and its applications is a topic of recent and increasing interest in graph theory. Now, let us introduce the main concepts and results concerning this theory.

If X is a metric space we say that the curve $\gamma : [a, b] \longrightarrow X$ is a *geodesic* if we have $L(\gamma|_{[t,s]}) = d(\gamma(t), \gamma(s)) = |t - s|$ for every $s, t \in [a, b]$. The metric space X is said *geodesic* if for every couple of points in X there exists a geodesic joining them. We can consider a graph G as a geodesic metric space.

If X is a geodesic metric space and $x_1, x_2, x_3 \in X$, a *geodesic triangle* $T = \{x_1, x_2, x_3\}$ is the union of the three geodesics $[x_1x_2]$, $[x_2x_3]$ and $[x_3x_1]$. We say that T is δ-*thin* if each of its sides is contained in the δ-neighborhood of the union of the other sides. We denote by $\delta(T)$ the sharp thin constant of T, i.e., $\delta(T) = \inf\{\delta \geq 0 |\, T \text{ is } \delta\text{-thin}\,\}$. The space X is δ-*hyperbolic* if every geodesic triangle in X is δ-thin. We denote by $\delta(X)$ the sharp hyperbolicity constant of X, i.e., $\mathrm{d}(X) := \sup\{\delta(T) |\, T \text{ is a geodesic triangle in } X\,\}$. We say that X is *hyperbolic* if X is δ-hyperbolic for some $\delta \geq 0$.

For a general graph deciding whether or not the space is hyperbolic seems an intractable problem. Thus, it is interesting to study the hyperbolicity of particular classes of graphs; see, for instance,[1], [6].

Let $\mathcal{H}(n, \delta_0)$ be the set of graphs G with n vertices and minimum degree δ_0. Let us define $a(n, \delta_0) := \min\{\delta(G) \mid G \in \mathcal{H}(n, \delta_0)\}$, $b(n, \delta_0) := \max\{\delta(G) \mid G \in \mathcal{H}(n, \delta_0)\}$.

The structure of this paper is as follows. In Section 2 we improve the upper bound for the diameter of a graph given in [2] (see Theorem 2.2). In Section 3 we prove upper and lower bounds for $b(n, \delta_0)$. Moreover, we compute the precise value of $b(n, \delta_0)$ for many values of n and δ_0 (see Theorem 3.1). In Section 4 we compute the precise values of $a(n, \delta_0)$ (see Theorem 4.1). See [4] for the detailed proofs.

2 Minimum degree and hyperbolicity constant

The following result gives an asymptotically sharp upper bound for the diameter of a connected graph (see [2, Theorem 1]).

Theorem 2.1 (Erdös, P., Pach, J., Pollack, R. and Tuza) *Let* $G \in \mathcal{H}(n, \delta_0)$ *with* $\delta_0 \geq 2$. *Then diam*$V(G) \leq \lfloor 3n/(\delta_0 + 1) \rfloor - 1$.

The next result provides better estimations of $\mathrm{diam}V(G)$.

Theorem 2.2 *If $G \in \mathcal{H}(n, \delta_0)$, then diam$V(G) \le n-1$ if $\delta_0 = 1$, and diam$V(G) \le \max\{2, \lfloor (3n-4)/(\delta_0+1) \rfloor - 1\}$ for every $\delta_0 \ge 2$.*

3 Bounds and some precise values for $b(n, \delta_0)$

We use Theorem 2.2, along with further results, to prove the theorem below, which gives good bounds for $b(n, \delta_0)$ when δ_0 is big enough, and provides the precise value of $b(n, \delta_0)$ in many cases.

Theorem 3.1 *Consider any $n \ge 4$ and $3 \le \delta_0 \le n-1$. If $\delta_0 \ge n-2$, then $b(n, \delta_0) = 1$. If $\delta_0 = n-3$, then $b(n, \delta_0) = 5/4$. If $(n-2)/2 \le \delta_0 \le n-4$, then $b(n, \delta_0) = 3/2$. If $\delta_0 = (n-3)/2$, then $b(n, \delta_0) = 7/4$. If $\delta_0 = (n-4)/2$, then $b(n, \delta_0) = 2$. If $\delta_0 = (n-5)/2$, then $b(n, \delta_0) = 9/4$. If $(n-2)/3 \le \delta_0 < (n-5)/2$, then $b(n, \delta_0) = 5/2$. If $\delta_0 = (n-3)/3$, then $5/2 \le b(n, \delta_0) \le 3$.*

4 Computation of $a(n, \delta_0)$

Theorem 4.1 *Consider $G \in \mathcal{H}(n, \delta_0)$ with $1 \le \delta_0 \le n-1$. If $\delta_0 = 1$, then $a(n, \delta_0) = 0$ for all values of n. If $\delta_0 = 2$, then $a(n, \delta_0) = 1$ if $n = 4$ and $a(n, \delta_0) = 3/4$ if $n \ne 4$. If $\delta_0 \ge 3$, then $a(n, \delta_0) = 1$.*

References

[1] Brinkmann, G., Koolen J. and Moulton, V., On the hyperbolicity of chordal graphs, *Ann. Comb.* **5** (2001), 61-69.

[2] Erdös, P., Pach, J., Pollack, R. and Tuza, Z., Radius, Diameter and Minimum Degree, *Journal of Combinatorial Theory* **47** (1989), 73-79.

[3] Ghys, E. and de la Harpe, P. *Sur les Groupes Hyperboliques d'après Mikhael Gromov*, Progress in Mathematics 83, Birkhäuser Boston Inc., MA, 1990.

[4] Hernández, V., Pestana, D. and Rodríguez, J. M., On a classical theorem on the diameter and minimum degree of a graph, *Submitted.*

[5] Miller, M., Sirán, J. *Moore graphs and beyond: A survey of the degree/diameter problem*, Electr. J. Comb. **20(2)** (2013), DS14v2.

[6] Rodríguez, J. M., Characterization of Gromov hyperbolic short graphs, *Acta Math. Sinica* **30** (2014), 197-212.

[7] Väisälä, J. *Gromov hyperbolic spaces*, Expo. Math. **23** (2005), 187-231.

Weighted Graphs as Dynamical Interaction Systems

Ulrich Faigle

Mathematisches Institut
Universität zu Köln
Cologne, Germany
faigle@zpr.uni-koeln.de

Abstract

This is an **extended** abstract.

Keywords: Graph, interaction system, Markov evolution, Schrödinger evolution

1 Symmetric interaction systems

A *(weighted) graph* on $N = \{1, \ldots, n\}$ is a map $A : N \times N \to \mathbb{R}$ and thus represented by a real $n \times n$ matrix $A = [A_{ij}]$. If N is a set of agents, A corresponds to a situation of pairwise interaction in N with structural coefficients A_{ij}. The vector space $\mathbb{R}^{n \times n}$ of all real $n \times n$ matrices is n^2-dimensional is a (real) Hilbert space with the inner product

$$\langle A|B\rangle = \sum_{ij} A_{ij}B_{ij} = \text{tr}(B^T A) \quad \text{and norm} \quad \|A\| = \sqrt{\langle A|A\rangle}.$$

A *simple* (real) interaction A is described by a vector $a \in \mathbb{R}^n$ of normed euclidian length $\|a\| = 1$ and an *amplitude* $\lambda \in \mathbb{R}$ such that

$$A_{ij} = \lambda a_i a_j \ \forall i, j \in N \ , \quad \text{i.e., } \ A = \lambda \cdot aa^T.$$

NOTE: The n diagonal elements a_i^2 of aa^T define a probability distribution on N.

Theorem 1.1 (Spectral decomposition) *$A \in \mathbb{R}^{n\times n}$ is symmetric if and only if A is the superposition of n pairwise orthogonal simple interactions $\lambda_i P_i$:*

$$A = \sum_{i\in N} \lambda_i P_i .$$

The amplitudes λ_i are the eigenvalues of A. ◇

The (well-known) spectral decomposition (Theorem 1.1) shows that the dynamics of symmetric interaction systems are determined by the dynamics of simple interactions. Note furthermore,

$$A^t = \sum_{i\in N} \lambda_i^t P_i \quad (t = 1, 2, \ldots)$$

2 Markov and Schrödinger evolution

Let $A = \lambda P$ with $P = aa^T$ be a simple interaction with amplitude λ. The *Markov evolution* is based on the evolution of the amplitude, *i.e.*, on the repeated application of A and thus the *Markov chain*

$$A^2 = \lambda^2 P, A^3 = \lambda^3 P, \ldots, A^t = \lambda^t P, \ldots$$

So the evolution converges to a limit if and only if the amplitudes λ^t converge to a limit.

In the *Schrödinger picture* the evolution of $A = \lambda P$ is described by an orthogonal transformation $a \mapsto Ua$ of $\mathbb{R}^n$ (with $U^{-1} = U^T$) and hence the chain

$$\lambda(Ua)(Ua)^T = \lambda UPU^T, \ldots, \lambda(U^t a)(U^t a)^T = \lambda(UPU^T)^t, \ldots .$$

of simple interactions with identical amplitude λ.

NOTE: In both evolution models, the evolution is derived from a linear operator on the space of all graphs (interactions).

3 Average convergence and sampling

Let μ be a linear operator on the Hilbert space of all interactions (graphs) A and define the averages

$$\overline{\mu}_t(A) = \frac{1}{t}\sum_{k=1}^{t} \mu^k(A) \quad (t = 1, 2, \ldots).$$

Theorem 3.1 (Sampling Theorem) *For any $A \in \mathbb{R}^{n\times n}$, the following statements are equivalent:*

(i) *The averages $\overline{\mu}_t(A)$ converge to a well-defined limit $\overline{\mu}(A)$ as $t \to \infty$.*

(ii) *The evolution chain $\{\mu^k(A) \mid k = 1, 2, \ldots\}$ is bounded in the norm.*

$\diamond$

For example, consider the simple interaction $A = \lambda P$. The Markov evolution of A converges on the average if and only if $|\lambda| \leq 1$. Observe

$$|\lambda| < 1 \quad \Longrightarrow \quad \lim_{t\to\infty} \lambda^t P = 0 \quad \text{and} \quad \lim_{t\to\infty} \frac{1}{t}\sum_{k=1}^{t} \lambda^k P = 0.$$

In the case $\lambda = -1$, the graphs (interactions) $(-1)^t P$ do not converge. Yet, their averages do:

$$\lim_{t\to\infty} \frac{1}{t}\sum_{k=1}^{t} (-1)^k P = 0.$$

In contrast, average convergence is guaranteed for any λ in a Schrödinger evolution. The graphs $\lambda(UPU^T)^t$, of course, do not need to converge themselves.

A *sampling function* is a linear functional $f : \mathbb{R}^{n\times n} \to \mathbb{R}$ and hence of the form

$$f(A) = \langle F|A\rangle \quad \text{for some } F \in \mathbb{R}^{n\times n}.$$

Theorem 3.1 says that the sampling averages

$$\overline{f}_t(A) = \frac{1}{t}\sum_{k=1}^{t} \langle F|\mu^k(A)\rangle$$

converge to a well-defined limit $\overline{f}_\infty(A)$ on bounded evolution chains $(\mu^k(A))$. Classical examples arise from the observation of random walks (Markov chains) on graphs.

4 General interaction systems

A non-symmetric matrix $A \in \mathbb{R}^{n \times n}$ does not admit a spectral representation as in Theorem 1.1. Considering A as an interaction representative, however, we may choose another representation. Recall that A admits a uniquely determined symmetric matrix A^+ and a skew-symmetric matrix A^- such that

$$A = A^+ + A^-.$$

Where $\mathrm{i}^2 = -1$, define $\hat{A} = A^+ + \mathrm{i}A^-$ and consider the real vector space

$$\mathcal{H}_n = \{\hat{A} \mid A \in \mathbb{R}^{n \times n}\} \subseteq \mathbb{C}^{n \times n}$$

Proposition 4.1 *$\mathcal{H}_n$ is the (real) Hilbert space of self-adjoint complex $n \times n$ matrices under the hermitian inner product*

$$\langle C|D\rangle = \mathrm{tr}(D^*C) \quad \textit{where } D^* = \overline{D}^T \textit{ is the adjoint of } D.$$

Moreover, $A \mapsto \hat{A}$ is an isometry *between $\mathbb{R}^{n \times n}$ and $\mathcal{H}_n$.*

◇

A *pure* matrix is a matrix of the form $P = vv^*$ with $v \in \mathbb{C}^n$, $\|v\| = 1$.

Theorem 4.2 (Spectral Representation) *For every $A \in \mathbb{R}^{n \times n}$, there are pairwise orthogonal pure matrices $P_1, \ldots, P_n$ and real amplitudes λ_i such that*

$$\hat{A} = \sum_{i=1}^{n} \lambda_i P_i.$$

◇

5 Binary interaction and quantum spins

Let $n = 2$. The space of symmetric matrices in $\mathbb{R}^{2 \times 2}$ is generated by

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \sigma_1 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \sigma_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

The skew symmetric subspace is generated by the matrix $\sigma_3 = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$.

In this case, one has $\hat{I} = I$, $\hat{\sigma}_1 = \sigma_1$, $\hat{\sigma}_2 = \sigma_2$ and

$$\hat{\sigma}_3 = \mathrm{i}\sigma_3 = \begin{pmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{pmatrix}.$$

I, σ_1, σ_2 and $\hat{\sigma}_3$ are the *Pauli spin matrices* in quantum mechanics.

NOTICE: $\sigma_3^2 = -I$ (*i.e.*, "$\sigma_3 = \sqrt{-I}$") but $\hat{\sigma}_3^2 = I$.

6 Complex numbers and quaternions

The set

$$\mathcal{C} = \left\{ aI + b\sigma_3 = \begin{pmatrix} a & -b \\ b & a \end{pmatrix} \mid a, b \in \mathbb{R} \right\} \subseteq \mathbb{R}^{2\times 2}$$

is closed under matrix addition and multiplication and, in fact, isomorphic to the field $\mathbb{C}$ of complex numbers.

For the use in mechanics, Hamilton introduced *quaternions* as formal linear combinations of the type

$$q = a + bi + cj + dk \quad (a, b, c, d \in \mathbb{R})$$

where

$$i^2 = j^2 = k^2 = -1, ij = -ji = k, jk = -kj = 1.ki = -ik = j.$$

Quaternions form a non-commutative field $\mathbb{H}$ and can be represented *via* matrices

$$Q = aI + b\sigma_3 + c(\mathrm{i}\sigma_1) + d(\mathrm{i}\sigma_2) \quad (a, b, c, d \in \mathbb{R}).$$

Quaternions endow $\mathbb{R}^{2\times 2}$ with the algebraic structure of a non-commutative field.

Some positive results on the complexity of the chip-firing reachability problem

Bálint Hujter [1,4]

Department of Operations Research
Eötvös Loránd University, Budapest, Hungary

Viktor Kiss [2,5]

Department of Analysis
Eötvös Loránd University, Budapest, Hungary

Lilla Tóthmérész [1,3]

Department of Computer Science
Eötvös Loránd University, Budapest, Hungary

Abstract

The chip-firing reachability problem asks whether for two given chip-distributions x and y, y can be reached from x by playing a legal chip-firing game. Previously Björner and Lovász gave an algorithm that decides this problem and runs in polynomial time for simple Eulerian digraphs. The complexity of the problem was left open for multigraphs and for non-Eulerian digraphs. We show that for Eulerian digraphs, the reachability problem can be decided in polynomial time even if the digraph has multiple edges. As a further positive result, we show that the chip-firing reachability problem is in $\mathbf{co-NP}$ for general digraphs.

Keywords: chip-firing game, computational complexity, algorithms

1 Introduction

Chip-firing is a solitary game on a directed graph, defined by Björner, Lovász and Shor [2]. Each vertex contains a non-negative integer number of chips. A legal move is to choose a vertex with at least as many chips as its out-degree and let it send a chip along each outgoing edge. The motivation for studying this game is that it gives a simple diffusion process on graphs, that still has interesting properties. We analyze the complexity of the following reachability question: given two chip-distributions x and y, decide whether y can be reached from x by playing a legal game. Let us denote by $x \rightsquigarrow y$ if such a legal game exists. This question is a special case of the reachability problem for integral vector addition systems [1]. It was first considered by Björner and Lovász, who gave an algorithm that decides the reachability question and runs in polynomial time for simple Eulerian digraphs [1]. The complexity of the reachability problem was left open for Eulerian digraphs with multiple edges, and for non-Eulerian digraphs. The question whether the reachability problem is in **NP** or in $\mathbf{co}-\mathbf{NP}$ was also left open. In this paper, we show that the chip-firing reachability problem can be decided in polynomial time for Eulerian digraphs with multiple edges. For general digraphs, we show that the reachability problem is in $\mathbf{co}-\mathbf{NP}$.

Let us sum up our notations. By *digraph*, we mean a weakly connected directed graph $G = (V, E)$, that can have multiple edges but no loops. For a vertex v, the in-degree and the out-degree of v are denoted by $d^-(v)$ and $d^+(v)$, respectively. We denote a directed edge leading from vertex u to vertex v by $\overrightarrow{uv}$. The multiplicity of $\overrightarrow{uv}$ is denoted by $\overrightarrow{d}(u, v)$. A digraph is *simple*, if $\overrightarrow{d}(u, v) \leq 1$ and $\overrightarrow{d}(v, u) \leq 1$ for each pair of vertices $u, v \in V$. A digraph is *Eulerian*, if $d^+(v) = d^-(v)$ for each $v \in V$. If we give a digraph as an input to an algorithm, we always encode it by its adjacency matrix. Hence a digraph might have exponentially many edges in terms of the size of the input.

In the chip-firing game, we call the assignment of a non-negative integer to each vertex a *chip-distribution*. We denote by $\mathbb{Z}_+^V$ the set of non-negative integer vectors indexed by the vertices of a digraph G. Hence, a chip-distribution is a vector $x \in \mathbb{Z}_+^V$. The *Laplacian* of a digraph G is the following matrix $L \in \mathbb{Z}^{V \times V}$: $L(v, v) = -d^+(v)$ for each v, and $L(u, v) = \overrightarrow{d}(v, u)$ for each

[1] Supported by the Hungarian Scientific Research Fund - OTKA K109240.

[2] Supported by the Hungarian Scientific Research Fund - OTKA 104178, 113047.

[3] Email: tmlilla@cs.elte.hu

[4] Email: hujterb@cs.elte.hu

[5] Email: kivi@cs.elte.hu

$u \neq v$. Note that firing a vertex v transforms the chip-distribution x to $x + L\mathbf{1}_v$, where $\mathbf{1}_v$ denotes the characteristic vector of vertex v.

Let us point out a simple necessary condition for $x \rightsquigarrow y$: If $x \rightsquigarrow y$, then for the vector $f \in \mathbb{Z}_+^V$ encoding for each vertex how many times it has been fired during the legal game transforming x to y, $y = x + Lf$. Hence the existence of a vector $f \in \mathbb{Z}_+^V$ such that $y = x + Lf$ is a necessary condition for $x \rightsquigarrow y$. By results of Björner and Lovász [1], if such a vector exists, then there is a unique minimal one among such nonzero vectors, that they call *reduced*. Whether such a vector exists can be decided in polynomial time by using Gaussian elimination and then solving a system of linear congruence equations. Also, if there exists a vector $f \in \mathbb{Z}_+^V$ such that $y = x + Lf$, then the reduced f can be computed in polynomial time by this procedure. By Björner and Lovász, if $x \rightsquigarrow y$, and f is the reduced vector such that $y = x + Lf$, then there exists a legal game transforming x to y, where each vertex v is fired exactly $f(v)$ times. Also by Björner and Lovász, whether for given x and f, there exists such a legal game, can be checked "greedily" [1, Lemma 1.4]. This gives an algorithm for deciding the reachability problem, but unfortunately the reduced f can have exponentially large coordinates. Björner and Lovász employ a scaling-like technique to speed up this greedy algorithm, which gives a polynomial algorithm for simple Eulerian digraphs. We use a different grouping of the firings, that enables us to create a polynomial algorithm also for Eulerian digraphs with multiple edges.

2 Results

Theorem 2.1 *There is a polynomial algorithm that decides whether $x \rightsquigarrow y$ for two chip-distributions x and y on an Eulerian digraph G (with possibly multiple edges).*

The heart of our algorithm is the following lemma. Informally, it says, that if one chip-distribution is reachable from another, then it can be reached so that we fire an ascending chain of subsets of vertices.

Lemma 2.2 *Let G be an Eulerian digraph. Suppose that $x, y \in \mathbb{Z}_+^V$ such that $x \rightsquigarrow y$. Then there exists a sequence of legal firings $(v_1, v_2, \dots, v_s)$ that transforms x to y, and there exist indices $i_0 = 0, i_1, i_2, \dots i_t = s$ such that for each $j = 1, \dots, t$, no vertex appears twice in the sequence $v_{i_{j-1}+1}, \dots, v_{i_j}$, and by setting $S_j = \{v_{i_{j-1}+1}, \dots, v_{i_j}\}$, we have $S_1 \subseteq S_2 \subseteq \dots \subseteq S_t \subsetneq V$.*

Remark 2.3 'Ascending chains' also play a role in the related field of graph divisor theory, see for example [3, Lemma 1.3.] or [4].

The idea of the algorithm is the following: We decide if there exists $f \in \mathbb{Z}_+^V$ such that $y = x + Lf$, and if yes, we compute the reduced f. From f, we can determine the sets $S_1, \dots S_t$. As the chain of sets $S_1 \subseteq S_2 \subseteq \dots \subseteq S_t \subsetneq V$ is ascending, at most polynomially many different sets appear in it. The key observation is that if we can fire a set at its last occurrence, we can fire it in all of its previous occurrences. We can compute the current chip-distribution at the beginning of the last firing of a set, and we can check greedily whether the set can be fired from that chip-distribution. We need to do this for polynomially many sets, hence we have a polynomial algorithm.

For general digraphs Björner and Lovász conjectures that the reachability problem is **NP**-hard [1], but previously, the question of whether the reachability problem is in **NP** or in $\mathbf{co}-\mathbf{NP}$ was also open. We show the following:

Theorem 2.4 *Let G be a digraph (with possibly multiple edges) and $x, y \in \mathbb{Z}_+^V$. Then deciding whether $x \rightsquigarrow y$ is in $\mathbf{co}-\mathbf{NP}$.*

3 Open questions

The most intriguing open question in the area is the complexity of the reachability problem on general digraphs.

Problem 3.1 *Let G be a digraph and $x, y \in \mathbb{Z}_+^V$. What is the complexity of deciding whether $x \rightsquigarrow y$?*

Acknowledgements

We would like to thank András Frank and Ágoston Weisz for calling our attention to reachability questions.

References

[1] A. Björner and L. Lovász. Chip-firing games on directed graphs. *J. Algebraic Combin.*, 1(4):305–328, 1992.

[2] A. Björner, L. Lovász, and P. W. Shor. Chip-firing games on graphs. *European J. Combin.*, 12(4):283–291, 1991.

[3] J. van Dobben de Bruyn and D. Gijswijt. Treewidth is a lower bound on graph gonality. *Preprint*, arXiv:1407.7055, 2014.

[4] J. van Dobben de Bruyne. Reduced divisors and gonality in finite graphs, Bachelor's thesis, Mathematisch Instituut, Universiteit Leiden, 2012.

Vertices, edges, distances and metric dimension in graphs

Ismael González Yero [1]

Departamento de Matemáticas, Escuela Politécnica Superior de Algeciras
Universidad de Cádiz
Av. Ramón Puyol s/n, 11202 Algeciras, Spain

Abstract

Given a connected graph $G = (V, E)$, a set of vertices $S \subset V$ is an edge metric generator for G, if any two edges of G are identified by S by mean of distances to the vertices in S. Moreover, in a natural way, S is a mixed metric generator, if any two elements of G (vertices or edges) are identified by S by mean of distances. In this work we study the (edge and mixed) metric dimension of graphs.

Keywords: mixed metric dimension, edge metric dimension, metric dimension.

Parameters related to distances in graphs have attracted the attention of several researchers since several years, and recently, one of them has centered several investigations, namely, the metric dimension. A vertex v of a connected graph G *distinguishes* two vertices u, w if $d(u, v) \neq d(w, v)$, where $d(x, y)$ represents the length of a shortest $x - y$ path in G. A subset of vertices S of G is a *metric generator* for G, if any pair of vertices of G is distinguished by at least one vertex of S. The minimum cardinality of any metric generator for G is the *metric dimension* of G. This concept was introduced by Slater in [5] in connection with some location problems in graphs. On the other hand,

[1] Email: ismael.gonzalez@uca.es

the concept of metric dimension was independently introduced by Harary and Melter in [2].

One can now consider the following situation. A metric generator uniquely recognizes the vertices of a graph in order to look out how they "behave". However, what does it happen if there are anomalous situations occurring in some connections (edges) between some vertices? Is it possible that metric generators would properly identify the edges in order to also see their behaving? The answer to this question is negative. In connection with this, the following concepts deserve to be considered.

Given a connected graph $G = (V, E)$, a vertex $v \in V$ and an edge $e = uw \in E$, the distance between the vertex v and the edge e is defined as $d_G(e, v) = \min\{d_G(u, v), d_G(w, v)\}$. A vertex $w \in V$ *distinguishes* two edges $e_1, e_2 \in E$ if $d_G(w, e_1) \neq d_G(w, e_2)$. A set $S \subset V$ is an *edge metric generator* for G if any two edges of G are distinguished by some vertex of S. The smallest cardinality of an edge metric generator for G is the *edge metric dimension* and is denoted by $edim(G)$ [3]. Moreover, a kind of mixed version of these two parameters described above is of interest. That is, a vertex v of G *distinguishes* two elements (vertices or edges) x, y of G if $d_G(x, v) \neq d_G(y, v)$. Now, a set $S \subset V$ is a *mixed metric generator* if any two elements of G are distinguished by some vertex of S. The smallest cardinality of a mixed metric generator for G is the *mixed metric dimension* and is denoted by $mdim(G)$ [4].

1 Results

As stated, there are several graphs in which no metric generator is also an edge metric generator. In this sense, one could think that probably any edge metric generator S is also a standard metric generator. Nevertheless, this is again further away from the reality, although there are several graph families in which such facts occur. In [3], among other results, some comparison between these two parameters above were discussed. In contrast with this, for the case of mixed metric dimension, it clearly follows that that any mixed metric generator is also a metric generator and an edge metric generator. In this sense, the following relationship immediately follows. For any graph G, $mdim(G) \geq \max\{dim(G), edim(G)\}$. From now on, we present several results concerning the (edge, mixed) metric dimension of graphs. First of all, we remark the next complexity result.

Theorem 1.1 [3] *Computing the edge metric dimension of graphs is NP-hard.*

The result above was proved by using a reduction from the 3-SAT problem.

Now, for the mixed metric dimension, nothing similar is known yet, although it is relatively natural to think that computing the mixed metric dimension is NP-hard, since also a similar fact occurs for the standard metric dimension. Based on these results, we next present some bounds or closed formulae for the (edge, mixed) metric dimension of several families of graphs.

Proposition 1.2 [3]

(i) *For any integer $n \geq 2$, $edim(P_n) = 1$, $edim(C_n) = 2$ and $edim(K_n) = n - 1$. Moreover, $edim(G) = 1$ if and only if G is a path P_n.*

(ii) *For any complete bipartite graph $K_{r,t}$ different from $K_{1,1}$, $edim(K_{r,t}) = r + t - 2$.*

Proposition 1.3 [4] *Let G be any graph of order n. Then*

(i) *$mdim(G) = 2$ if and only if G is a path.*

(ii) *If at least one of the next situations happens, then $mdim(G) = n$.*
- *Every vertex of G is a true twin vertex or an extreme vertex.*
- *There are at least two vertices of degree $n - 1$.*

Proposition 1.4 [4]

(i) *For any integer $n \geq 4$, $mdim(C_n) = 3$.*

(ii) *For any integers $r, t \geq 2$,* $mdim(K_{r,t}) = \begin{cases} r + t - 1, & \textit{if } r = 2 \textit{ or } t = 2, \\ r + t - 2, & \textit{otherwise.} \end{cases}$

A vertex of degree at least 3 in a tree T is a *major vertex* of T. Any leaf u of T is said to be a *terminal vertex* of a major vertex v of T if $d(u, v) < d(u, w)$ for every other major vertex w of T. The *terminal degree* of a major vertex v is the number of terminal vertices of v. A major vertex v of T is an *exterior major vertex* of T if it has positive terminal degree. Let $n_1(T)$ denote the number of leaves of T, and let $ex(T)$ denote the number of exterior major vertices of T.

Proposition 1.5 [3] *If T is a tree which is not a path, then $edim(T) = n_1(T) - ex(T)$.*

Proposition 1.6 [4] *For any tree T, $mdim(T) = n_1(T)$.*

The Cartesian product of two graphs G and H is the graph $G \Box H$, such that $V(G \Box H) = \{(a, b) \ : \ a \in V(G), \ b \in V(H)\}$ and two vertices (a, b) and (c, d) are adjacent in $G \Box H$ if and only if, either ($a = c$ and $bd \in E(H)$), or ($b = d$ and $ac \in E(G)$).

Proposition 1.7 [3] *For any integers $r \geq t \geq 2$, $edim(P_r \square P_t) = 2$.*

Proposition 1.8 [4] *For any integers $r \geq t \geq 2$, $mdim(P_r \square P_t) = 3$.*

The next result shows an example where the edge metric dimension is strictly less than the standard metric dimension, since $edim(C_{4r} \square C_{4t}) = 4$.

Proposition 1.9 [3] *For any integers r, t, $edim(C_{4r} \square C_{4t}) = 3$.*

We close our exposition with a mathematical programming model for computing the mixed metric dimension of a graph G. A similar model for the metric dimension is known [1]. Let G be a graph of order n and size m with vertex set $V = \{v_1, \ldots, v_n\}$ and edge set $E = \{e_1, \ldots, e_m\}$. We consider the matrix $D = [d_{ij}]$ of order $(n+m) \times n$ such that $d_{ij} = d_G(x_i, x_j)$, $x_i \in V \cup E$ and $x_j \in V$. Now, given the variables $y_j \in \{0,1\}$ with $j \in \{1, 2, \ldots, n\}$ we define the following function:

$$\mathcal{F}(y_1, y_2, \ldots, y_n) = y_1 + y_2 + \ldots + y_n.$$

Clearly, minimizing the function $\mathcal{F}$ subject to the following constraints

$$\sum_{i=1}^{n} |d_{ji} - d_{li}| y_i \geq 1 \text{ for every } 1 \leq j < l \leq n+m,$$

is equivalent to finding a mixed metric basis of G, since the solution for $y_1, y_2, \ldots, y_n$ represents a set of values for which the function $\mathcal{F}$ achieves the minimum possible.

References

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

[2] Harary, F., and R. A. Melter, *On the metric dimension of a graph*, Ars Combin. **2** (1976), 191–195.

[3] Kelenc, A., N. Tratnik, and I. G. Yero, *Uniquely identifying the edges of a graph: the edge metric dimension*, Manuscript, (2015).

[4] Kelenc, A., D. Kuziak, A. Taranenko, and I. G. Yero, *On the mixed metric dimension of graphs*, Manuscript, (2016).

[5] Slater, P. J., *Leaves of trees*, Congr. Numer. **14** (1975) 549–559.

Squares of Low Clique Number

Petr Golovach[a,1] Dieter Kratsch[b,2] Daniël Paulusma[c,3,5]
Anthony Stewart[c,4]

[a] *University of Bergen, Bergen, Norway*

[b] *Université de Lorraine, Metz, France*

[c] *Durham University, Durham, UK*

Abstract

The Square Root problem is that of deciding whether a given graph admits a square root. This problem is only known to be NP-complete for chordal graphs and polynomial-time solvable for non-trivial minor-closed graph classes and a very limited number of other graph classes. By researching boundedness of the treewidth of a graph, we prove that Square Root is polynomial-time solvable on various graph classes of low clique number that are not minor-closed.

Keywords: graph classes, square roots, squares, treewidth

The *square* $G = H^2$ of a graph $H = (V_G, E_G)$ is the graph with vertex set V_H, such that any two distinct vertices $u, v \in V_H$ are adjacent in G if and only if u and v are of distance at most 2 in H. A graph H is a *square root* of G if $G = H^2$. There exist graphs with no square root, graphs with a unique square root as well as graphs with many square roots. The corresponding

[1] petr.golovach@ii.uib.no

[2] dieter.kratsch@univ-lorraine.fr

[3] daniel.paulusma@durham.ac.uk

[4] a.g.stewart@durham.ac.uk

[5] supported by EPSRC Grant EP/K025090/1.

recognition problem, which asks whether a given graph admits a square root, is called the Square Root problem and is known to be NP-complete [9]. As such, it is natural to restrict the input to special graph classes in order to obtain polynomial-time results. For many graph classes the complexity of Square Root is still unknown. For instance, Milanic and Schaudt [8] posed the complexity of Square Root restricted to split graphs and cographs as open problems. In Table 1 we survey the known results (note that the row for planar graphs could be absorbed by the row above of it). We explain this table in more detail below. In this paper we aim to identify new classes of squares of bounded treewidth. Our motivation for this question stems from the following result (obtained via applying Courcelle's meta-theorem).

Lemma 1 (**[2]**) *The* Square Root *problem can be solved in time* $O(f(t)n)$ *for* n*-vertex graphs of treewidth at most* t*.*

The unreferenced results in Table 1 correspond to our new results. The last column of this table indicates whether the squares of the graph class have bounded treewidth, where an $*$ means that these squares have bounded treewidth after some appropriate edge reduction (see [3] for further details). Note that the seven graph classes in the bottom seven rows not only have bounded treewidth but also have bounded clique number. We also observe that Nestoridis and Thilikos [10] proved that Square Root is polynomial-time solvable for non-trivial minor-closed graph classes by showing boundedness of carving width instead of treewidth. However, it is possible, by using the graph minor decomposition of Robertson and Seymour, to show that squares of graphs from such classes have in fact bounded treewidth as well.

We sketch the proof of one of our results from Table 1, namely the proof for 3-degenerate graphs (that is, graphs for which every subgraph has a vertex of degree at most 3.) We need one known and one new lemma (proof omitted).

Lemma 2 (**[1]**) *For any fixed constant* k*, it is possible to decide in linear time whether the treewidth of a graph is at most* k*.*

Lemma 3 *Let* H *be a square root of a graph* G*. Let* T *be the bipartite graph with* $V_T = \mathcal{C} \cup \mathcal{B}$*, where partition classes* $\mathcal{C}$ *and* $\mathcal{B}$ *are the set of cut vertices and blocks of* H*, respectively, such that* $u \in \mathcal{C}$ *and* $Q \in \mathcal{B}$ *are adjacent if and only if* Q *contains* u*. For* $u \in \mathcal{C}$*, let* X_u *consist of* u *and all neighbours of* u *in* H*. For* $Q \in \mathcal{B}$*, let* $X_Q = V_Q$*. Then* (T, X) *is a tree decomposition of* G*.*

We call the tree decomposition (T, X) the H*-tree decomposition* of G. We also need the following lemma.

Lemma 4 *If* G *is a* 3*-degenerate graph with a square root, then* $\mathbf{tw}(G) \leq 3$*.*

graph class	complexity	square bounded tw
trivially perfect graphs [8]	polynomial	no
threshold graphs [8]	polynomial	no
chordal graphs [4]	NP-complete	no
line graphs [7]	polynomial	no
non-trivial and minor-closed [10]	linear	yes
planar graphs [6]	linear	yes
K_4-free graphs	linear	yes
(K_r, P_t)-free graphs	linear	yes
3-degenerate graphs	linear	yes
graphs of maximum degree ≤ 5	linear	yes
graphs of maximum degree ≤ 6 [2]	polynomial	yes*
graphs of maximum average degree $< \frac{46}{11}$ [3]	polynomial	yes*

Table 1
The known results for Square Root restricted to some special graph class.

Proof. Without loss of generality we may assume that G is connected and has at least one edge. Let H be a square root of G; let $\mathcal{C}$ be the set of cut vertices of H and let $\mathcal{B}$ be the set of blocks of H. We construct the H-tree decomposition (T, X) of G (cf. Lemma 3) and show that (T, X) has width at most 3.

We start with two useful observations. If $v \in V_H$, then $N_H[v]$ is a clique in G. Because G is 3-degenerate, this means that $\Delta(H) \leq 3$. For the same reason H contains no cycles of length at least 5 as a subgraph, since a square of a cycle of length at least 5 has minimum degree 4.

We claim that X_Q has size at most 4 for every $Q \in \mathcal{B}$. In order to see this let Q be a block of H and let $u \in V_Q$. Suppose that Q has a vertex v at distance at least 3 from u. Since Q is 2-connected, Q has two internally vertex disjoint paths that join u and v. Therefore, Q (and thus H) contain a cycle of length at least 6 which, as we saw, is not possible. We find that each vertex $v \in V_Q$ is at distance at most 2 from u. Hence, u is adjacent to all other vertices of Q in G. Similarly, any two vertices in Q are of distance at most 2 from each other. Hence, Q is a clique in G. As G is 3-degenerate, this means that Q is a clique in G of size at most 4. Consequently, X_Q, has size at most 4. As $\Delta(H) \leq 3$, X_u has size at most 4 for every cut vertex u of H.$\square$

We can now prove the following result.

Theorem 1 Square Root *can be solved in* $O(n)$ *time for* 3*-degenerate graphs.*

Proof. Let G be an 3-degenerate graph on n vertices. By Lemma 2 we can check in $O(n)$ time whether $\mathbf{tw}(G) \leq 3$. If $\mathbf{tw}(G) > 3$, then G has no square root by Lemma 4. Otherwise, apply Lemma 1. $\square$

We cannot claim any upper bound for the treewidth of 4-degenerate graphs with a square root: take the square of a wall of arbitrarily large treewidth in which each edge is subdivided three times. We pose determining the complexity of the Square Root problem for 4-degenerate graphs as an open problem.

References

[1] H. L. Bodlaender, A linear-time algorithm for finding tree-decompositions of small treewidth, SIAM J. Comput. 25 (1996) 305–1317.

[2] M. Cochefert, J.-F. Couturier, P. A. Golovach, D. Kratsch, and D. Paulusma, Sparse square roots, Proc. WG 2013, LNCS 8165 (2013) 177–188.

[3] P.A. Golovach, D. Kratsch, D. Paulusma and A.G. Stewart, A linear kernel for finding square roots of almost planar graphs, Proc. SWAT 2016, Leibniz Int. Proc. Inform., to appear.

[4] L. C. Lau and D. G. Corneil, Recognizing powers of proper interval, split, and chordal graph, SIAM J. Discrete Math. 18 (2004) 83–102.

[5] V. B. Le and N. N. Tuy, A good characterization of squares of strongly chordal split graphs, Inf. Process. Lett. 111 (2011) 120–123.

[6] Y.-L. Lin and S. Skiena, Algorithms for square roots of graphs, SIAM J. Discret. Math. 8 (1995) 99–118.

[7] M. Milanic, A. Oversberg and O. Schaudt, A characterization of line graphs that are squares of graphs, Discret. Appl. Math. 173 (2014) 83–91.

[8] M. Milanic and O. Schaudt, Computing square roots of trivially perfect and threshold graphs, Discret. Appl. Math. 161 (2013) 1538–1545.

[9] R. Motwani and M. Sudan, Computing roots of graphs is hard, Discret. Appl. Math. 54 (1994) 81–88.

[10] N.V. Nestoridis and D.M. Thilikos, Square roots of minor closed graph classes, Discret. Appl. Math. 168 (2014) 34–39.

On the shelling antimatroids of split graphs

Keno Merckx [1]

Département d'Informatique
Université Libre de Bruxelles
Brussels, Belgium

Jean Cardinal

Département d'Informatique
Université Libre de Bruxelles
Brussels, Belgium

Jean-Paul Doignon

Département d'Informatique
Université Libre de Bruxelles
Brussels, Belgium

Abstract

Unlike poset antimatroids, chordal graph shelling antimatroids have received little attention as regard their structures, optimization properties and associated circuits. Here we consider a special case of those antimatroids, namely the split graph shelling antimatroids. We establish a connection between the structure of split graph shelling antimatroids and poset shelling antimatroids. We discuss some applications of this new connection, in particular, we give a simple polynomial time algorithm to find a maximum weight feasible set in split graph shelling antimatroids.

Keywords: Antimatroid, split graph, shelling, poset.

1 Introduction

Many classical problems in combinatorial optimization have the following form.

Problem 1.1 *For a set system $(V,\mathcal{F})$ and for a function $w : V \to \mathbb{R}$, find a set F of $\mathcal{F}$ maximizing the value of*

$$w(F) = \sum_{f \in F} w(f).$$

For instance, the problem is known to be efficiently solvable for the independent sets of matroids using the greedy algorithm. Since antimatroids capture a combinatorial abstraction of convexity in the same way as matroids capture linear dependence, we investigate the optimization of linear objective functions for antimatroids.

We recall that a set system $(V,\mathcal{F})$, where V is a finite set of elements and $\varnothing \neq \mathcal{F} \subseteq 2^V$, is an *antimatroid* when

$$V \in \mathcal{F}, \tag{AM0}$$

$$\forall F_1, F_2 \in \mathcal{F} \Rightarrow F_1 \cup F_2 \in \mathcal{F}, \tag{AM1}$$

$$\forall F \in \mathcal{F} \text{ and } F \neq \varnothing \Rightarrow \exists\, f \in F \text{ such that } F \setminus \{f\} \in \mathcal{F}. \tag{AM2}$$

The *feasible sets* of the antimatroid $(V,\mathcal{F})$ are the members of $\mathcal{F}$. We call *path* any feasible set that cannot be decomposed into the union of two other (non-empty) feasible sets.

Antimatroids arise naturally from various kinds of shellings and searches on combinatorial objects, and appear in various contexts in mathematics and computer science. Dilworth [4] first examined structures very close to antimatroids in terms of lattice theory. Later, Edelman [5] and Jamison [7] studied the convex aspects of antimatroids. Korte, Lovász and Schrader [8] considered antimatroids as a subclass of greedoids. Today, the concept of antimatroid appears in many fields of mathematics such as formal language theory (Boyd and Faigle [2]), choice theory (Koshevoy [9]), game theory (Algaba *et al.* [1]) and mathematical psychology (Falmagne and Doignon [6]) among others. The concept of a convex geometry is dual to the one of an antimatroid.

For instance, one particular class of antimatroids comes from shelling processes over posets by removing successively the maximum elements. Let $(V, \leq)$

[1] Email: `kmerckx@ulb.ac.be`

be a poset, then $(V, \mathrm{flt}(V, \leq))$ is a *poset (shelling) antimatroid* when $\mathrm{flt}(V, \leq)$ denotes the family of all filters of the poset.

It is not known whether a general efficient algorithm exists for Problem 1.1 in the case of antimatroids as shown in the following proposition.

Proposition 1.2 *The problem of finding a maximum weight feasible set in an antimatroid encoded in the form of its path is not approximable in polynomial time within a factor better than $O(N^{\frac{1}{2}-\varepsilon})$ for any $\varepsilon > 0$, where N is the number of paths, unless $P = NP$.*

2 Main results

Here, we gave a polynomial time algorithm for solving Problem 1.1 when set system $(V, \mathcal{F})$ is a "split graph shelling antimatroids" which are particular instances of "chordal graph shelling antimatroids". For any chordal graph $G = (V, E)$, we define an antimatroid $(V, \mathcal{F})$ in which $F \subseteq V$ is feasible if and only if there is some ordering $O = (f_1, \ldots, f_{|F|})$ of the elements of F such that for all j between 1 and $|F|$, f_j is simplicial in $G \setminus \{f_1, \ldots, f_{j-1}\}$. The antimatroid resulting from this construction is called a *chordal graph (vertex) shelling antimatroid*. Here we consider the special case of chordal graph shelling antimatroids where the graph is a split graph. These antimatroids will be called *split graph (vertex) shelling antimatroids*.

We have obtained a useful characterization of the feasible sets in a split graph shelling antimatroid.

Proposition 2.1 *Let $G = (K \cup I, E)$ be a split graph and $(V, \mathcal{F})$ be the split graph vertex shelling antimatroid defined on G. Then a subset F of vertices is feasible for the antimatroid if and only if $N(F)$ induces a clique.*

We use Proposition 2.1 to establish a connection between the structure of split graph shelling antimatroids and poset shelling antimatroids. This connection given by Proposition 2.1 helps us to solve optimization problems on split graph shelling antimatroid.

For an antimatroid, we call *path* any feasible set that cannot be decomposed into the union of two other (non-empty) feasible sets. The family of paths can be partially ordered by inclusion, forming the *path poset*. Antimatroids are completely determined by their path posets. The path poset is can be seen as a short way to encode all the information of the antimatroid.

Proposition 2.2 *Giving a split graph G (as a list of vertices and a list of edges), the problem of finding a maximum weight feasible set in the split graph* *shelling antimatroid defined on G can be done in polynomial time.*

We can also use Proposition 2.1 to characterize in simple terms the "circuits" and "free sets" of a split graph shelling antimatroid. Let $(V, \mathcal{F})$ be an antimatroid and $X \subseteq V$, we define the *trace* of $(V, \mathcal{F})$ on X by

$$\mathrm{tr}(\mathcal{F}, X) = \{F \cap X : F \in \mathcal{F}\}.$$

We say that a set $X \subseteq V$ is *free* if $\mathrm{tr}(\mathcal{F}, X) = 2^X$. A minimal nonfree set is called a *circuit*. Remark that if C is a circuit then $\mathrm{tr}(\mathcal{F}, X) = 2^C \setminus \{\{r\}\}$ for some r in C, see [8] for proof. The element r is the *root* of C, and we will call the pair $(C \setminus \{r\}, r)$ a *rooted circuit*. Dietrich [3] provides a characterization of antimatroids in terms of their circuits.

References

[1] E. A. Algaba, J. M. Bilbao, R. van den Brink, and A. Jimnez-Losada. Cooperative games on antimatroids. *Discrete Mathematics*, 282(1-3):1–15, 2004.

[2] E. A. Boyd and U. Faigle. An algorithmic characterization of antimatroids. *Discrete Appl. Math.*, 28(3):197–205, 1990.

[3] B. L. Dietrich. A circuit set characterization of antimatroids. *J. Combin. Theory Ser. B*, 43(3):314–321, 1987.

[4] R. P. Dilworth. Lattices with unique irreducible decompositions. *Ann. of Math. (2)*, 41:771–777, 1940.

[5] P. H. Edelman. Meet-distributive lattices and the anti-exchange closure. *Algebra Universalis*, 10(3):290–299, 1980.

[6] J.-C. Falmagne and J.-P. Doignon. *Learning Spaces.* Springer-Verlag, Berlin, 2011.

[7] R. E. Jamison-Waldner. A perspective on abstract convexity: classifying alignments by varieties. In *Convexity and related combinatorial geometry (Norman, Okla., 1980)*, volume 76 of *Lecture Notes in Pure and Appl. Math.*, pages 113–150. Dekker, New York, 1982.

[8] B. Korte, L. Lovász, and R. Schrader. *Greedoids*, volume 4 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 1991.

[9] G. Koshevoy. Choice functions and abstract convex geometries. *Math. Social Sci.*, 38(1):35–44, 1999.

A Polynomial Recognition of Unit Forms

Jesmmer Alves [1,2] Diane Castonguay [3]

Instituto de Informática - Universidade Federal de Goiás
Goiânia, Goiás, Brasil

Thomas Brüstle [4]

Faculté des Sciences - Université de Sherbrooke
Sherbrooke, Quebec, Canada

Abstract

In this paper we introduce a polynomial algorithm for the recognition of weakly nonnegative unit forms. The algorithm identify hypercritical restrictions testing every 9-point subset of the quadratic form associated graph. With Depth First Search strategy, we use a similar approach for the weakly positive recognition.

Keywords: unit form, polynomial algorithm, graph. MSC: 11E04, 16G60, 42A82.

1 Introduction

An integral quadratic form q is defined as $q(x) = \sum_{i \leq j} a_{ij} x_i x_j$, for $x \in \mathbb{Z}^n$. In this paper, we are interested in **unit form**, that is an integral quadratic form

[1] This author was supported by Capes, Fapeg, UFG and Instituto Federal Goiano.

[2] Email: `jesmmeralves@inf.ufg.br`

[3] Email: `diane@inf.ufg.br`

[4] Email: `thomas.brustle@usherbrooke.ca`

where $a_{ii} = 1$, for all i. Its corresponding symmetric bilinear form is such that $q(x) = \frac{1}{2}q(x, x)$. The recognition of weakly nonnegative and weakly positive unit forms have an important role in the representation theory of algebras, however, some concepts related to computational complexity theory have not been fully described.

Dean and De La Peña [1] developed an algorithm to decide whether a given unit form is weakly nonnegative. The strategy was to generate all positive roots and make some tests to identify whether the unit form is weakly nonnegative. Despite being a significant development, the weakly nonnegative unit forms can have an infinite number of positive roots, and this strategy become unfeasible.

A very interesting solution for weakly nonnegative recognition comes from the hypercritical unit forms, classified by Unger [5]. All unit forms in the Unger's list have 9 or less vertices. Therefore, we use an algorithm to test all 9-point subsets. That strategy give us a polynomial algorithm of complexity $O(n^9)$. By adding Depth First Search approach, we use a similar strategy in the weakly positive recognition.

2 Basic Concepts

A vector $x \in \mathbb{Z}^n$ is said to be positive, written $x > 0$, provided $x \neq 0$ and $x_i \geq 0$ for all i. A unit form q is **weakly positive** if $q(x) > 0$ for all positive $x \in \mathbb{Z}^n$, or **weakly nonnegative**, if $q(x) \geq 0$ for all positive $x \in \mathbb{Z}^n$.

Definition 2.1 [2] A unit form q is said to be critical, resp. hypercritical, if every proper restriction q' is weakly positive, resp. weakly nonnegative, but q itself is not.

All critical forms were classified by von Höhne [4] and all hypercritical by Unger [5]. A unit form is properly represented by quivers.[5] A **Quiver** $Q = (Q_0, Q_1)$ is a finite and connected graph with a set of vertices $Q_0 = \{1, \ldots, n\}$ and a set of edges Q_1, with possibly multiple edges but without loops.

3 A Polynomial Approach

The following corollary gives support to the polynomial algorithm for weakly nonnegative recognition.

[5] See more details about quiver representations in [2].

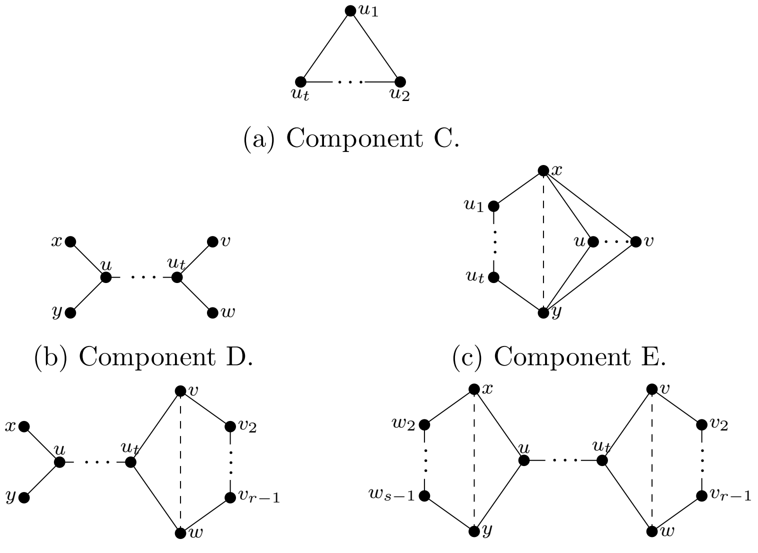
Corollary 3.1 *A unit form q is weakly nonnegative if and only if every restriction q' which correspond to a connected induced subquiver of 9 vertices is weakly nonnegative.*

Proof. Follows from De La Peña [2] and Unger [5]. □

For an algorithm that generates all connected induced subquivers see [3]. Testing[6] each subquiver is constant since we have a fixed number of points, and there are $\binom{n}{9}$ such subquivers - a polynomial of degree 9 in n.

On the other hand, deciding whether a unit form is weakly positive is equivalent to excluding all subgraphs that are critical restrictions. Except for the diagrams $\widetilde{\mathbb{A}}_n$ and $\widetilde{\mathbb{D}}_n$, all critical restrictions also have less than 10 vertices. Therefore, we use the DFS strategy to identify the restrictions in the form $\widetilde{\mathbb{A}}_n$ and $\widetilde{\mathbb{D}}_n$, and if it doesn't fail, we test all connected induced subquivers with 9 vertices of the quiver associated to q, by using De La Peña algorithm [2].

We represent the critical restrictions of type $\widetilde{\mathbb{A}}_n$ and $\widetilde{\mathbb{D}}_n$ by the components in the Figure 1. The component C represent the critical restrictions of type $\widetilde{\mathbb{A}}_n$ and, the components D, E, F and G, critical restrictions of type $\widetilde{\mathbb{D}}_n$.



(a) Component C.

(b) Component D.

(c) Component E.

(d) Component F.

(e) Component G.

Fig. 1. Components that represent critical restrictions of type $\widetilde{\mathbb{A}}_n$ and $\widetilde{\mathbb{D}}_n$.

[6] The test can be done by the Dean and De La Peña algorithm [1].

We highlight the ideas of our algorithms in five situation that characterize these five critical restrictions in a quiver Q.[7] Any vertex that are neighbors to the explored subquiver without the vertices that we are looking to extends, by positive or negative edges, are considered blocked.

When Q has a component C, the algorithm starts with some triplet $p = \langle x, u, y \rangle$ and increase p adding a new unblocked vertex at the end of p. Eventually it finds some vertex v that is adjacent to x and fails. If Q has a component D, it starts with $p = \langle u \rangle$ and increase p adding a new unblocked vertex v at the end of p. Eventually it finds two unblocked neighbors v and w of the end vertex u_t of p, that are not neighbors by positive edges and fails. In the case that Q has a component E, we start with the triplet $\langle x, u, y \rangle$ and will look for $\langle x, v, y \rangle$. Afterwards, it finds the path $\langle u_1, u_2, \ldots, u_t \rangle$ and fails. For the component F, the algorithm starts with the triplet $\langle x, u, y \rangle$. As for component D, the algorithm increase the path p (starting with p = $\langle u \rangle$) till it finds two unblocked neighbors v and w of the end vertex u_t of p. In this case, $(v, w) \in Q_{1\_p}$ and the algorithm will extend $p = \langle v, u_t, w \rangle$. It will fail after finding a path between v and w. Finally, when Q has a component G, the algorithm starts with the triplet $\langle x, u, y \rangle$. At this stage, the algorithm had verify that it has no Component E. As for component D, it increase the path p, starting with $(p = \langle u \rangle)$, till it finds two unblocked neighbors v and w of the end vertex u_t of p. In this case, $(v, w) \in Q_{1\_p}$, and the algorithm finds both paths $\langle v_1, v_2, \ldots, v_r \rangle$ and $\langle w_1, w_2, \ldots, w_s \rangle$ and fails.

References

[1] Dean, A. and De La Peña, J.A.: Algorithms for Weakly Nonnegative Quadratic Forms, Linear Algebra and its Applications, Elsevier, 235, 35–46, (1996).

[2] De La Peña, J.A.: Quadratic Forms and the Representation Type of an Algebra, Sonderforschungsbereich, Univ. Bielefeld, 343, ISSN 0936–7926, 90–003, (1990).

[3] Maxwell, S. and Chance, M.R. and Koyutürk, M.: Efficiently Enumerating All Connected Induced Subgraphs of a Large Molecular Network, Algorithms for Computational Biology, Springer, 171–182, (2014).

[4] von Höhne, H.: On Weakly Positive Unit Forms, Commentarii Mathematici Helvetici, Springer, 63, 1, 12–336, (1988).

[5] Unger, L.: The Concealed Algebras of the Minimal Wild, Hereditary Algebras, Bayreuth. Math. Schr., 99, 145–154, (1990).

[7] All paths and triplets cited are connected by negative edges.

Encoding Bigraphical Reactive Systems into Graph Transformation Systems

Amal Gassara [a,1], Ismael Bouassida Rodriguez [a,c], Mohamed Jmaiel [a,b] and Kalil Drira [c]

[a] *ReDCAD Laboratory, University of Sfax, B.P. 1173, 3038 Sfax, Tunisia*

[b] *Digital Research Center of Sfax, B.P. 275, Sakiet Ezzit, 3021 Sfax, Tunisia*

[c] *LAAS-CNRS, Université de Toulouse, CNRS, Toulouse, France*

Abstract

In this paper, we present a solution for executing bigraphical reactive systems based on an investigation on graph transformation systems. For this, we encode a bigraph into a ranked graph. This encoding is ensured, formally, by defining a faithful functor that allows to move from bigraph category to ranked graph category. Then, we show that reaction rules can be simulated with graph rules.

Keywords: Bigraphs, BRS, Graphs, Matching.

1 Introduction

The theory of Bigraphical Reactive Systems (BRSs) has been developed by Milner [5] as a formalism for describing and analyzing mobile computation and pervasive systems. A BRS is a graphical model in which bigraphs can be reconfigured using reaction rules. It is very important to have an implementation of the dynamic of a BRS to enable experimentations. The main challenge of this implementation is the matching problem. In fact, it is a computational task that determines for a given bigraph B and a reaction rule R whether and how the reaction rule can be applied to rewrite the bigraph B.

The theory of BRS is closely related to graph transformation system (GTS) [3,2]. Considering the exhaustiveness of studies on graph transformations, it is natural to ask whether we could apply graph matching algorithms on Bigraphs. As an alternative to implementing matching for bigraphs, we could try to formalize BRSs as GTSs. By this way, we can benefit from existing

[1] Email: amal.gassara@redcad.org

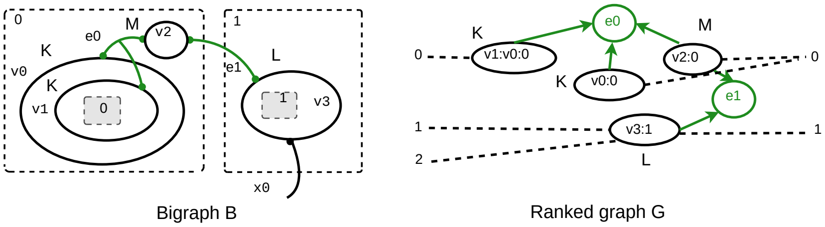


Fig. 1. Encoding a bigraph into a ranked graph

tools and techniques developed for graph transformations. Consequentially, we have initiated an investigation of how to simulate a BRS with a GTS.

In this paper, we propose a formal basis allowing such simulation. Indeed, we encode a bigraph into a graph by defining a function named F_{sim} that allows to move from bigraph category to graph category. We demonstrate that F_{sim} is a well defined and faithful functor. Then, we rely on the work of Ehrig [1] to show that reaction rules can be simulated with graph rules. As a result, we ensure the validity of simulating a BRS by a GTS.

2 Encoding a Bigraph into a Ranked Graph

In order to understand our contribution, the reader should understand bigraphs [5] and ranked graphs [4].

The main difference between bigraphs and graphs lies in the nesting and the linking structure of bigraphs. Hence, we define the nesting structure of bigraphs through the node identifiers of graphs. For instance, in Fig. 1, v_0 is nested in 0 (the parent of v_0 is 0). Its image in the graph G is a node having the identifier $v_0 : 0$. So, we encode the parent of a node through its identifier.

Furthermore, the linking structure of bigraphs is represented in graphs by defining two types of nodes: *place nodes* that represent bigraph places, and *link nodes* that represent bigraph hyperedges. For example, the hyperedge e_1 in the bigraph of Fig. 1, connecting v_2 and v_3, is represented in the graph with the green node e_1 to which are connected $v_2 : 0$ and $v_3 : 1$.

Categorically, bigraphs and their morphisms form a category $\mathcal{BG}$ which has as objects inner and outer interfaces, and as arrows bigraphs. Similar to bigraphs, ranked graphs are presented as morphisms between two interfaces i and j, forming a category denoted $\mathcal{DG}$.

Our main objective is to ensure the validity of encoding bigraphs into ranked graphs, preserving their structure. We shall achieve this by defining a functor [5] which allows to move from one category to another.

Hence, we define a functor, named $F_{sim} : \mathcal{BG} \to \mathcal{DG}$, which allows to move from $\mathcal{BG}$ to $\mathcal{DG}$. This functor associates to each morphism (Bigraph) $B : I \to J$ from $\mathcal{BG}$, a morphism (Graph) $G : i \to j$ from $\mathcal{DG}$.

2.1 Defining F_{sim} on objects.

We define an injective function F_{sim} between the objects (interfaces) of the two models. Given a bigraphical interface $\langle m, X\rangle$, F_{sim} associates a graph interface represented as a list of ordered numbers with exactly $m + |X| - 1$ elements, regarded as a discrete graph. Every $x < m$ is encoded by a *place node* and every name $\in X$ is encoded by a *link node*. For example in Fig. 1, the image of the inner interface $I = \langle 2, \{x_0\}\rangle$ is the interface $i = \{0, 1, 2\}$ of G where the nodes 0 and 1 are *place nodes* and the node 2 is a *link node*. The interface i is represented by the list of numbers on the left of the graph.

2.2 Defining F_{sim} on morphisms.

Consider a bigraph $B = (V_B, E_B, ctrl_B, prnt_B, link_B)$. $F_{sim}(B) = G = (V_G, VE_G, ctrl_G, prnt_G, link_G)$. F_{sim} is defined as a pair of functions (f_v, f_e) where:

- $V_G = f_v(V_B)$. f_v associates for each node $\in V_B$, a node $\in V_G$. The identifier of a node image is determined by concatenating the identifier of this node with the identifier of its parents. For example, in Fig. 1, $f_v(v_1) = v_1 : v_0 : 0$.
- $VE_G = f_e(E_B)$. f_e associates for each hyperedge $\in E_B$, a *link node* $\in VE_G$. For example, in Fig. 1, $f_e(e_0) = e_0$ (the image of the hyperedge e_0 in the bigraph B is the node e_0 in the graph G).

Proposition 2.1 *$F_{sim} = (f_v, f_e)$ respects the structure in the following sense:*

(1) F_{sim} preserves the controls

(2) F_{sim} preserves the structural mapping prnt

(3) F_{sim} preserves the structural mapping link

Proposition 2.2 *F_{sim} is a faithful functor between $\mathcal{BG}$ and $\mathcal{DG}$.*

Proof. We have demonstrated that F_{sim} is a well defined functor by demonstrating that it preserves functor properties (i.e., preserves identity and composition). So, F_{sim} is a faithful functor since the morphisms f_v and f_e are injective functions. For sake of shortness, we have not present the proof. □

Proposition 2.2 ensures the validity of moving from $\mathcal{BG}$ to $\mathcal{DG}$ by F_{sim}, ensuring in this way the validity of encoding a bigraph into a ranked graph.

3 Simulating BRSs with GTSs

Bigraphs are associated with reaction rules which can be applied to rewrite bigraphs. On the other hand, graphs are associated with rewrite rules or productions according to the double pushout approach, DPO approach [3].

We propose a correspondence between a reaction rule and a production. Let $(R, R' : \epsilon \to I)$ a reaction rule in $\mathcal{BG}$ and let a reaction relation $a \to a'$ via

(R, R') and $D : I \to J$. By definition, F_{sim} translates the rule $(R, R' : \epsilon \to I)$ and the context $D : I \to J$ from $\mathcal{BG}$ into $(F_{sim}(R), F_{sim}(R') : \epsilon \to F_{sim}(I))$ and $F_{sim}(D) : F_{sim}(I) \to F_{sim}(J)$ in $\mathcal{DG}$, respectively.

Since F_{sim} preserves composition, $F_{sim}(a) = F_{sim}(D \circ R) = F_{sim}(D) \circ F_{sim}(R)$ and $F_{sim}(a') = F_{sim}(D \circ R') = F_{sim}(D) \circ F_{sim}(R')$. So, F_{sim} translates each transformation $a \to a'$ into a transformation $F_{sim}(a) \to F_{sim}(a')$ into $\mathcal{DG}$.

Ehrig [1] showed that it is possible to use the cospan idea to construct from a reaction relation a corresponding DPO transformation $a \Rightarrow a'$ via (p, D) where p is constructed from the reaction rule (R, R').

So, we can obtain from the reaction relation $F_{sim}(a) \to F_{sim}(a')$ via $(F_{sim}(R), F_{sim}(R'))$ and $F_{sim}(D) : F_{sim}(I) \to F_{sim}(J)$ a DPO transformation $F_{sim}(a) \Rightarrow F_{sim}(a')$ via $(p, F_{sim}(D))$ where $p = (F_{sim}(R) \leftarrow F_{sim}(I) \to F_{sim}(R'))$. Hence, we can simulate the application of a reaction rule on a bigraph by applying its corresponding production on the encoded graph.

4 Conclusion

In this paper, we have presented a solution for executing BRSs that is based on an investigation of GTSs. In fact, we have encoded a bigraph into a ranked graph. This encoding is ensured, formally, by defining a faithful functor F_{sim} from bigraph categories to ranked graph categories. Then, we have referenced Ehrig et al. [1] to show that reaction rules can be simulated by graph rules. Hence, the behavior of bigraphs can be simulated by simulating their encoded graphs using the graph transformation tools and techniques.

References

[1] H. Ehrig. Bigraphs meet double pushouts. *Bulletin of the EATCS*, 78:72–85, 2002.

[2] H. Ehrig, K. Ehrig, U. Prange, and G. Taentzer. *Fundamentals of Algebraic Graph Transformation (Monographs in Theoretical Computer Science. An EATCS Series).* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[3] H. Ehrig, M. Pfender, and H.J. Schneider. Graph-grammars: An algebraic approach. In *IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 167–180, 1973.

[4] F. Gadducci and R. Heckel. An inductive view of graph transformation. In FrancescoParisi Presicce, editor, *Recent Trends in Algebraic Development Techniques*, volume 1376 of *Lecture Notes in Computer Science*, pages 223–237. Springer Berlin Heidelberg, 1998.

[5] R. Milner. *The Space and Motion of Communicating Agents.* Cambridge University Press, 2009.