Lecture Notes in Computer Science 7801

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Michel Goemans José Correa (Eds.)

Integer Programming and Combinatorial Optimization

16th International Conference, IPCO 2013
Valparaíso, Chile, March 18-20, 2013
Proceedings

Springer

Volume Editors

Michel Goemans
Massachusetts Institute of Technology
Department of Mathematics
77 Massachusetts Ave.
Cambridge, MA 02139, USA
E-mail: goemans@math.mit.edu

José Correa
Universidad de Chile
Department of Industrial Engineering
Republica 701
Santiago, Chile
E-mail: correa@uchile.cl

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-36693-2 e-ISBN 978-3-642-36694-9
DOI 10.1007/978-3-642-36694-9
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2013931229

CR Subject Classification (1998): G.1.6, F.2.2, G.2.1-3

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This volume contains the 33 extended abstracts presented at IPCO 2013, the 16th Conference on Integer Programming and Combinatorial Optimization, held during March 18–20, 2013, in Valparaíso, Chile. IPCO conferences are sponsored by the Mathematical Optimization Society. The first IPCO conference took place at the University of Waterloo in May 1990. It is held every year, except for those in which the International Symposium on Mathematical Programming is held.

The conference had a Program Committee consisting of 14 members and was chaired by Michel Goemans. In response to the Call for Papers, we received 98 extended abstracts, of which three got withdrawn prior to the decision progress. Each submission was reviewed by at least three Program Committee members. Once the reviews were available, the decisions were made in late November and early December through conference calls and electronic discussions using the EasyChair conference management system. We had many high-quality submissions and ended up selecting 33 extended abstracts. This number is dictated by the fact that the conference has a single-stream of non-parallel sessions, as is the tradition at IPCO. We expect the full versions of the papers contained in this volume to be submitted for publication in refereed journals.

This year, IPCO was followed by a Summer School during March 21–23, 2013, with lectures by Samuel Fiorini on extended formulations in combinatorial optimization, and by François Margot on recent developments in cutting planes for mixed integer programming. For the first time, there was also a Poster Session held on the first evening of the conference.

We would like to thank:

- All authors who submitted extended abstracts of their research to IPCO
- The members of the Program Committee, who graciously gave plenty of their time and energy to select the accepted extended abstracts
- The reviewers whose expertise was instrumental in guiding our decisions
- The members of the Local Organizing Committee (chaired by José Correa), who made this conference possible.

January 2013

Michel Goemans
José Correa

Organization

Program Committee

Chandra Chekuri	University of Illinois, USA
Bill Cook	University of Pittsburgh, USA
José Correa	Universidad de Chile, Chile
Jesús De Loera	University of California, Davis, USA
Michel Goemans	MIT, USA
Volker Kaibel	University of Magdeburg, Germany
Jon Lee	University of Michigan, USA
François Margot	CMU, USA
Thomas McCormick	UBC, Canada
Andreas Schulz	MIT, USA
David Shmoys	Cornell University, USA
Zoltán Szigeti	Grenoble INP, France
Robert Weismantel	ETH, Switzerland
Giacomo Zambelli	London School of Economics and Political Science, UK

Additional Reviewers

Aissi, Hassene
Andrews, Matthew
Averkov, Gennadiy
Baes, Michel
Bansal, Nikhil
Bley, Andreas
Bonami, Pierre
Burer, Samuel
Büsing, Christina
Cheriyan, Joseph
Cheung, Wang Chi
Chudnovsky, Maria
Dey, Santanu
Ene, Alina
Felsner, Stefan
Fischetti, Matteo
Fukunaga, Takuro
Gaspers, Serge
Geelen, Jim
Goundan, Pranava
Goyal, Vineet
Guiñez, Flavio
Gunluk, Oktay
Gusfield, Dan
Harvey, Nick
Hemmecke, Raymond
Homen-De-Mello, Tito
Hosten, Serkan
Huh, Tim
Husfeldt, Thore
Im, Sungjin
Iwata, Satoru
Kiraly, Tamas
Kiraly, Zoltan
Kleinberg, Bobby
Kobayashi, Yusuke
Koenemann, Jochen
Kolliopoulos, Stavro
Kucukyavuz, Simge
Laraki, Rida

Luebbecke, Marco
Luedtke, James
Mastrolilli, Monaldo
Mehta, Aranyak
Mittal, Shashi
Mueller, Dirk
Nagarajan, Viswanath
Natarajan, Karthik
Olver, Neil
Ordonez, Fernando
Orlin, James
Pap, Gyula
Peis, Britta
Pferschy, Ulrich
Pfetsch, Marco
Pruhs, Kirk
Pêcher, Arnaud
Ravi, R.
Richard, Jean-Philippe
Romeijn, Zedwin
Rothvoss, Thomas
Segev, Danny
Sidiropoulos, Anastasios
Singh, Mohit
Soto, Jose
Stier-Moses, Nicolas
Sviridenko, Maxim
Telha, Claudio
Tulsiani, Madhur
Van Vyve, Mathieu
van Zuylen, Anke
Vegh, Laci
Vegh, Laszlo
Verschae, Jose
Vygen, Jens
Weismantel, Robert
Woeginger, Gerhard J.
Zenklusen, Rico
Zwick, Uri

Table of Contents

On the Structure of Reduced Kernel Lattice Bases

Karen Aardal[1,2] and Frederik von Heymann[1]

[1] Delft Institute of Applied Mathematics, TU Delft, The Netherlands
{k.i.aardal,f.j.vonheymann}@tudelft.nl
[2] Centrum Wiskunde en Informatica, Amsterdam, The Netherlands

Abstract. Lattice-based reformulation techniques have been used successfully both theoretically and computationally. One such reformulation is obtained from the lattice $\ker_{\mathbb{Z}}(\boldsymbol{A}) = \{\boldsymbol{x} \in \mathbb{Z}^n \mid \boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}\}$. Some of the hard instances in the literature that have been successfully tackled by lattice-based techniques, such as market split and certain classes of knapsack instances, have randomly generated input $\boldsymbol{A}$. These instances have been posed to stimulate algorithmic research. Since the considered instances are very hard even in low dimension, less experience is available for larger instances. Recently we have studied larger instances and observed that the LLL-reduced basis of $\ker_{\mathbb{Z}}(\boldsymbol{A})$ has a specific sparse structure. In particular, this translates into a map in which some of the original variables get a "rich" translation into a new variable space, whereas some variables are only substituted in the new space. If an original variable is important in the sense of branching or cutting planes, this variable should be translated in a non-trivial way. In this paper we partially explain the obtained structure of the LLL-reduced basis in the case that the input matrix $\boldsymbol{A}$ consists of one row $\boldsymbol{a}$. Since the input is randomly generated our analysis will be probabilistic. The key ingredient is a bound on the probability that the LLL algorithm will interchange two subsequent basis vectors. It is worth noticing that computational experiments indicate that the results of this analysis seem to apply in the same way also in the general case that $\boldsymbol{A}$ consists of multiple rows. Our analysis has yet to be extended to this general case. Along with our analysis we also present some computational indications that illustrate that the probabilistic analysis conforms well with the practical behavior.

1 Introduction

Consider the following integer program:

$$\max\{\boldsymbol{c}\boldsymbol{x} \mid \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0}\}, \tag{1}$$

where $\boldsymbol{A}$ is an integer $m \times n$ matrix of full row rank and $\boldsymbol{b}$ an integer m-vector. Starting with the well-known algorithm of Lenstra [13], several lattice-based approaches to reformulate the feasible region have been proposed, see, e.g., [1, 3, 5, 11, 16–18]. Here we will consider the reformulation as in [1]:

$$\boldsymbol{x} := \boldsymbol{x}^0 + \boldsymbol{Q}\boldsymbol{\lambda}\,, \tag{2}$$

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 1–12, 2013.

where $\boldsymbol{x}^0 \in \mathbb{Z}^n$ satisfies $\boldsymbol{A}\boldsymbol{x}^0 = \boldsymbol{b}$, $\boldsymbol{\lambda} \in \mathbb{Z}^{n-m}$, and $\boldsymbol{Q}$ is a basis for the lattice $\ker_{\mathbb{Z}}(\boldsymbol{A}) = \{\boldsymbol{x} \in \mathbb{Z}^n \mid \boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}\}$. Due to the nonnegativity requirements on the $\boldsymbol{x}$-variables, one now obtains an equivalent formulation of the integer program (1):

$$\max\{\boldsymbol{c}(\boldsymbol{x}^0 + \boldsymbol{Q}\boldsymbol{\lambda}) \mid Q\boldsymbol{\lambda} \geq -\boldsymbol{x}^0\}\,. \tag{3}$$

This reformulation has been shown to be of particular computational interest in the case where $\boldsymbol{Q}$ is reduced in the sense of Lovász [12].

Several authors have studied knapsack instances that have a particular structure that makes them particularly difficult to solve by "standard" methods such as branch-and-bound. Examples of such instances can be found in [2, 7, 11]. Common for these instances is that the input is generated in such a way that the resulting lattice $\ker_{\mathbb{Z}}(\boldsymbol{A})$ has a very particular structure that makes the reformulated instances almost trivial to solve. Other instances that are randomly generated without any particular structure of the $\boldsymbol{A}$-matrix, such as the market split instances [6] and knapsack instances studied in [2, 3], have no particular lattice structure. Yet they are practically unsolvable by branch-and-bound in the original $\boldsymbol{x}$-variable space, whereas their lattice reformulation solves rather easily, at least up to a certain dimension. It is still to be understood why the lattice reformulation for these instances is computationally more effective.

If we consider the randomly generated instance without any particular lattice structure and solve small instances, such as $n - m \leq 25$, one typically observes that the number of zeros in the basis $\boldsymbol{Q}$ is small. In higher dimension, and here "high" is depending on the input, a certain sparser structure will start to appear.

More specifically, we observe computationally that $\boldsymbol{Q}$ has a certain number of rows with rich interaction between the variables $\boldsymbol{x}$ and $\boldsymbol{\lambda}$, but from some point on this interaction breaks down almost instantly and we get one '1' per row, i.e., $\boldsymbol{Q}$ yields variable substitutions. To be able to better understand the relative effectiveness of the lattice reformulation, and in order to be able to apply the lattice reformulation in a (more) useful way in higher dimension, it is important to identify the variables that have a nontrivial translation into the new $\boldsymbol{\lambda}$-variable space.

In this paper we partially explain the phenomenon described above for the case that $m = 1$, that is, $\boldsymbol{A}$ consists of a single row $\boldsymbol{a} = (a_1, \ldots, a_n)$. As the exact structure of $\boldsymbol{Q}$ depends on the choice of $\boldsymbol{a}$, our analysis will be probabilistic. To this end, we assume that the entries of our input vector $\boldsymbol{a}$ are drawn independently and uniformly at random from an interval $[l, \ldots, u] := [l, u] \cap \mathbb{Z}$, where $0 < l < u$. We notice that explaining the phenomenon is related to the analysis of the probability that the LLL-algorithm performs a basis vector interchange after a basis vector with a certain index k has been considered by the algorithm.

Let $\boldsymbol{Q} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{n-1}]$ be an LLL y-reduced basis (see Section 2 for more details) of $\ker_{\mathbb{Z}}(\boldsymbol{a})$, and let $\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_{n-1}^*$ be the Gram-Schmidt vectors corresponding to $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{n-1}$. If $\|\boldsymbol{b}_{i+1}^*\|^2 \geq y\|\boldsymbol{b}_i^*\|^2$, then basis vectors $i+1$ and i will not be interchanged. We will show that, starting with a basis $\bar{\boldsymbol{Q}}$ of $\ker_{\mathbb{Z}}(\boldsymbol{a})$ of a certain structure, the probability that the LLL-algorithm [12] performs basis vector interchanges becomes increasingly small the higher the index of the basis

vector. In particular, for given l, u, and reduction factor y, we derive a constant c and a k_0, such that for $k \geq k_0$ we have

$$\Pr\left(\|\boldsymbol{b}_{k+1}^*\|^2 < y\|\boldsymbol{b}_k^*\|^2\right) \quad \leq \quad e^{-c(k+1)^2} + 2^{-(k+1)/2}. \tag{4}$$

Note that stated in this form it is an asymptotic result, but we will see that the values of k_0 are very similar to the ones observed in the experiments.

To derive a bound on $\Pr\left(\|\boldsymbol{b}_{k+1}^*\|^2 < y\|\boldsymbol{b}_k^*\|^2\right)$ we first need to be able to express the length of the Gram-Schmidt vectors $\boldsymbol{b}_j^*$ in terms of the input vector $\boldsymbol{a}$. This is done in Section 2 and results in Expression (18). The bound on $\Pr\left(\|\boldsymbol{b}_{k+1}^*\|^2 < y\|\boldsymbol{b}_k^*\|^2\right)$ is derived through several steps in Section 3. In this derivation, the challenge is that $\|\boldsymbol{b}_{k+1}^*\|^2$ and $\|\boldsymbol{b}_k^*\|^2$ are not independent. To estimate the mean of the ratio $\|\boldsymbol{b}_{k+1}^*\|^2/\|\boldsymbol{b}_k^*\|^2$, we use a result by Pittenger [19], and to estimate how much this ratio deviates from the mean we use the Azuma-Hoeffding inequality [4, 8]. Some computational indications and further discussion are provided in Section 4. We notice that the computational results corresponds well to the observed practical behavior of the LLL algorithm on the considered class of input.

2 Notation and Preliminaries

We first repeat some known facts about lattices and bases of lattices, as well as a high-level description of the LLL-algorithm. Then we give some properties of the kernel lattice of $\boldsymbol{a}$.

2.1 Basic Results on Lattices

Let L be a lattice in $\mathbb{R}^n$, i.e., a discrete additive subgroup of $\mathbb{R}^n$. Furthermore, let $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m,\ m \leq n$, be a basis of L, and let $\boldsymbol{x}^T$ denote the transpose of vector $\boldsymbol{x}$. The Gram-Schmidt vectors are defined as follows:

$$\begin{aligned}
\boldsymbol{b}_1^* &= \boldsymbol{b}_1, \\
\boldsymbol{b}_i^* &= \boldsymbol{b}_i - \sum_{j=1}^{i-1} \mu_{ij}\boldsymbol{b}_j^*, \quad 2 \leq i \leq m, \quad \text{where} \\
\mu_{ij} &= \frac{\boldsymbol{b}_i^T\boldsymbol{b}_j^*}{\|\boldsymbol{b}_j^*\|^2}, \quad 1 \leq j < i \leq m.
\end{aligned}$$

For fixed $y \in (\frac{1}{4}, 1)$ we call $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m\}$ *y-reduced*, if

$$|\mu_{ij}| \leq \frac{1}{2}, \quad \text{for } 1 \leq j < i \leq m-1, \text{ and} \tag{5}$$

$$\|\boldsymbol{b}_i^* + \mu_{i,i-1}\boldsymbol{b}_{i-1}^*\|^2 \geq y\,\|\boldsymbol{b}_{i-1}^*\|^2, \quad \text{for } 1 < i \leq m-1\,. \tag{6}$$

Notice that, as $\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_m^*$ are pairwise orthogonal, Inequality (6) is satisfied if

$$\|\boldsymbol{b}_i^*\|^2 \geq y\|\boldsymbol{b}_{i-1}^*\|^2, \quad \text{for } 1 < i \leq m-1\,. \tag{7}$$

We will not describe the LLL-algorithm in detail, but just mention the two operations that are carried out by the algorithm. For $x \in \mathbb{R}^1$, let $\lfloor x \rceil$ denote the nearest integer to x. If Condition (5) is violated, i.e., $|\mu_{kj}| > 1/2$ for some $j < k$, then a *size reduction* is carried out by setting $\boldsymbol{b}_k := \boldsymbol{b}_k - \lfloor \mu_{kj} \rceil \boldsymbol{b}_j$. Notice that this operation will not change the Gram-Schmidt vector $\boldsymbol{b}_k^*$. If Condition (6) is violated for $i = j$, then vectors $\boldsymbol{b}_{j-1}$ and $\boldsymbol{b}_j$ are *interchanged*. This operation does affect several of the μ-values. Moreover, the new vector $\boldsymbol{b}_{j-1}^*$ will be the old vector $\boldsymbol{b}_j^* + \mu_{j,j-1}\boldsymbol{b}_{j-1}^*$.

For a given basis $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m\}$ of the lattice $L \subset \mathbb{R}^n$, define the matrix $\boldsymbol{B} = [\boldsymbol{b}_1 \cdots \boldsymbol{b}_m]$, such that the columns of $\boldsymbol{B}$ are given by the basis-vectors. Then $\boldsymbol{B}^T\boldsymbol{B}$ is an $m \times m$-matrix of full rank, and we can define the *determinant* of the lattice L as

$$d(L) = (\det(\boldsymbol{B}^T\boldsymbol{B}))^{1/2}\,. \tag{8}$$

It can be shown that this value is independent of the basis we choose for the lattice. Furthermore, we derive an expression of $d(L)$ in terms of the associated Gram-Schmidt orthogonalization.

Observation 1. *Given a basis $\boldsymbol{B} = [\boldsymbol{b}_1 \cdots \boldsymbol{b}_m]$ of a lattice $L \subset \mathbb{R}^n$ of rank m, and the associated Gram-Schmidt orthogonalization $\boldsymbol{B}^* = [\boldsymbol{b}_1^* \cdots \boldsymbol{b}_m^*]$, we have*

$$d(L) = \prod_{i=1}^{m} \|\boldsymbol{b}_i^*\|\,. \tag{9}$$

An explanation of how to derive Expression (9) can for instance be found in [10].

To every lattice L we can associate the *dual lattice*

$$L^\dagger = \{\boldsymbol{x} \in \text{lin. span}(L) \mid \boldsymbol{x}^T\boldsymbol{y} \in \mathbb{Z} \text{ for all } \boldsymbol{y} \in L\}.$$

Notice that $L^{\dagger\dagger} = L$, and that

$$d(L^\dagger) = \frac{1}{d(L)}\,. \tag{10}$$

A subset $K \subseteq L$ is called a *pure sublattice* of L if $K = \text{lin. span}(K) \cap L$. Let $K^\perp$ be the sublattice of $L^\dagger$ orthogonal to K, i.e., $K^\perp = \{\boldsymbol{x} \in L^\dagger \mid \boldsymbol{x}^T\boldsymbol{y} = 0 \text{ for all } \boldsymbol{y} \in K\}$.

Observation 2. *If K is a pure sublattice of L then $K^\perp$ is a pure sublattice of $L^\dagger$ and we have*

$$K^\perp = (L/K)^\dagger \tag{11}$$

and

$$d(L) = d(L/K) \cdot d(K)\,. \tag{12}$$

Suppose $L = \mathbb{Z}^n$. Then, by combining (12), (10), and (11) we obtain

$$d(K) = \frac{d(L)}{d(L/K)} = \frac{1}{d(L/K)} = d((L/K)^\dagger) = d(K^\perp)\,. \tag{13}$$

A more detailed account on this and much more can be found in, e.g., [14] and [15].

2.2 Some Results for the Kernel Lattice of $\boldsymbol{a}$

In this subsection we consider a vector $\boldsymbol{a} \in \mathbb{Z}^n$ such that $\gcd(a_1, \ldots, a_n) = 1$.

The kernel lattice of $\boldsymbol{a}$ is the set $\ker_{\mathbb{Z}}(\boldsymbol{a}) := \{\boldsymbol{x} \in \mathbb{Z} \mid \boldsymbol{a}\boldsymbol{x} = 0\}$. The lattice $\ker_{\mathbb{Z}}(\boldsymbol{a})$ is a pure sublattice of $\mathbb{Z}^n$.

We first show in Lemma 1 that the lattice $\ker_{\mathbb{Z}}(\boldsymbol{a})$ has a basis of the following form:

$$\boldsymbol{Q} = \begin{pmatrix} x & x & \cdots & x \\ x & x & \cdots & x \\ 0 & x & \cdots & x \\ \vdots & 0 & \ddots & x \\ 0 & \cdots & 0 & x \end{pmatrix}, \tag{14}$$

where each 'x' denotes some integer number that may be different from zero.

Lemma 1. *The lattice* $\ker_{\mathbb{Z}}(\boldsymbol{a})$ *has a basis* $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{n-1}$ *of the following form:*

$$\mathbb{Z}\boldsymbol{b}_1 + \ldots + \mathbb{Z}\boldsymbol{b}_k = \ker_{\mathbb{Z}}(\boldsymbol{a}) \cap (\mathbb{Z}^{k+1} \times 0^{n-k-1}) \tag{15}$$

for any $1 \leq k \leq n-1$.

Proof. Write $c_i = \min\{|y_i| > 0 \mid \boldsymbol{y} \in \ker_{\mathbb{Z}}(\boldsymbol{a}), y_j = 0 \text{ for } j > i\}$, where $2 \leq i \leq n$. Note that the set we minimize over is not empty, because the vector $(-a_i, 0, \ldots, 0, a_1, 0, \ldots, 0)^T$, where a_1 appears in the ith position, is in $\ker_{\mathbb{Z}}(\boldsymbol{a})$ for any $i \in \{2, \ldots, n\}$. Now choose

$$\boldsymbol{b}_i \in \{\boldsymbol{x} \in \ker_{\mathbb{Z}}(\boldsymbol{a}) \mid x_{i+1} = c_{i+1}, x_j = 0 \text{ for } j > i+1\}. \tag{16}$$

To see that this is indeed a lattice-basis, let $\boldsymbol{z} \in \ker_{\mathbb{Z}}(\boldsymbol{a})$ and let k be the largest index of a non-zero coordinate of $\boldsymbol{z}$. Let $\boldsymbol{Q} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{n-1}]$, where $\boldsymbol{b}_i$ satisfies (16).

We want to find $\boldsymbol{\lambda} \in \mathbb{Z}^{n-1}$ such that $\boldsymbol{z} = \boldsymbol{Q}\boldsymbol{\lambda}$. Observe that $\frac{z_k}{c_k}$ must be integer, because otherwise there is a $c' \in \mathbb{Z}$ such that $0 < |z_k - c'c_k| < c_k$, which contradicts the minimality of c_k. Therefore we may define $\lambda_{k-1} := \frac{z_k}{c_k}$.

Setting $\boldsymbol{z} = \boldsymbol{z} - \lambda_{k-1}\boldsymbol{b}_{k-1}$, this gives us a recursive construction for the integer coefficients $\lambda_1, \ldots, \lambda_{n-1}$ to express $\boldsymbol{z}$ in terms of our basis. □

One can additionally observe that if $\gcd(a_1, \ldots, a_i) = 1$ for some $1 \leq i \leq n$ then the last non-zero element of the basis vectors $\boldsymbol{b}_i, \ldots, \boldsymbol{b}_{n-1}$ is equal to ± 1.

We will follow up on this idea in Section 4.

Let L_k be the sublattice given by the basis $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_k$ as described in Lemma 1, for $1 \leq k \leq m$. Then we have $L_1 \subseteq L_2 \subseteq \cdots \subseteq L_{n-1} = \ker_{\mathbb{Z}}(\boldsymbol{a})$ and $d(L_k) = \prod_{i=1}^{k} \|\boldsymbol{b}_i^*\|$. Also, because of the specific structure of the basis, we can express L_k as

$$L_k = \{\boldsymbol{x} \in \mathbb{Z}^n \mid (a_1, \ldots, a_{k+1}, 0, \ldots, 0)\boldsymbol{x} = 0, x_j = 0, k+2 \leq j \leq n\}.$$

We can extend the above observations to conclude the following:

Lemma 2. *Let $L_1, \ldots, L_{n-1}$ be given as above and let $k \in \{1, \ldots, n-1\}$. If $\gcd(a_1, \ldots, a_{k+1}) = 1$, then*

$$d(L_k) = \sqrt{\sum_{i=1}^{k+1} a_i^2}, \tag{17}$$

and thus we get in particular

$$\|\boldsymbol{b}_k^*\|^2 = \frac{\sum_{i=1}^{k+1} a_i^2}{\sum_{i=1}^{k} a_i^2}. \tag{18}$$

Proof. Observe that $(a_1, \ldots, a_{k+1}, 0, \ldots, 0)^T$ and the unit vectors $\boldsymbol{e}_j$, with $k+2 \leq j \leq n$, are an orthogonal basis of $L_k^{\perp}$. Using (9) and the fact that $d(K) = d(K^{\perp})$ for pure sublattices of $\mathbb{Z}^n$ (see (13)), we get (17).

Equation (18) follows from (9) in combination with (17) for L_k and L_{k-1}. □

3 Probabilistic Analysis

Here we present the main result of the paper, namely a bound on the probability that the LLL-algorithm will perform a basis vector interchange after basis vector $\boldsymbol{b}_k$ is considered. We assume that the elements a_i of the vector $\boldsymbol{a}$ are drawn independently and uniformly at random from an interval $[l, \ldots, u] := [l, u] \cap \mathbb{Z}$, where $0 < l < u$, and that the starting basis of $\ker_{\mathbb{Z}}(\boldsymbol{a})$ is a basis of the structure given in Lemma 1. Recall from Subsection 2.1 that if, for given reduction factor $y \in (\frac{1}{4}, 1)$,

$$\|\boldsymbol{b}_{i+1}^*\|^2 \geq y\|\boldsymbol{b}_i^*\|^2, \quad \text{for } 1 \leq i < n-1\,,$$

then the LLL-algorithm will not interchange basis vectors $\boldsymbol{b}_i$ and $\boldsymbol{b}_{i+1}$.

We will prove the following result:

Theorem 1. *Let $y \in (\frac{1}{4}, 1)$ be fixed. Then, for k large enough, we get*

$$\Pr\left(\frac{\|\boldsymbol{b}_{k+1}^*\|^2}{\|\boldsymbol{b}_k^*\|^2} \leq y\right) \quad \leq \quad e^{-c(k+1)^2} + 2^{-(k+1)/2}\,, \tag{19}$$

where $c > 0$ depends on u, l, and y.

We provide explicit bounds on c and when k is large enough. To increase accessibility to the proof, we build our result from several lemmas. We start by noticing that for any $1 \leq k < n-1$

$$\begin{aligned}\Pr\left(\|\boldsymbol{b}_{k+1}^*\|^2 < y\|\boldsymbol{b}_k^*\|^2\right) \leq\ & \Pr\left(\|\boldsymbol{b}_{k+1}^*\|^2 < y\|\boldsymbol{b}_k^*\|^2 \mid \gcd(a_1, \ldots, a_{k+1}) = 1\right) \\ & + \Pr(\gcd(a_1, \ldots, a_{k+1}) > 1),\end{aligned} \tag{20}$$

and hence we can bound the two terms separately. The last one can be bounded in the following way:

Lemma 3. *Let $a_1, \ldots, a_n$ be chosen independently and uniformly at random from $[l, \ldots, u]$ for some integers $0 < l < u$, and let l and u be fixed. Then*

$$\Pr(\gcd(a_1, \ldots, a_{k+1}) > 1) \leq \left(\frac{1}{2}\right)^{(k+1)/2}$$

for any $k \geq \frac{\log_2(\lfloor \frac{u}{2} \rfloor + 1)}{\log_2(\frac{u-l+1}{u-l+2}) + \frac{1}{2}}$.

Next, for given reduction factor y, we want to derive a bound on the first term of Expression (20), i.e.:

$$\Pr\left(\frac{\|\boldsymbol{b}_{k+1}^*\|^2}{\|\boldsymbol{b}_k^*\|^2} < y \mid \gcd(a_1, \ldots, a_{k+1}) = 1\right).$$

Showing that the ratio between $\|\boldsymbol{b}_{k+1}^*\|^2$ and $\|\boldsymbol{b}_k^*\|^2$ behaves the way we suspect is not straightforward as the two quantities are not independent. To estimate the mean of this ratio we use a result by Pittenger [19], which we state below in a form that is adapted to our situation.

Theorem 2 ([19], adapted). *Let X be a random variable on some positive domain. Choose $c > 0$ such that $X - c \geq 0$ and define $\mu = \mathbb{E}[X]$ and $\sigma^2 = Var(X)$. Then*

$$\begin{aligned} \frac{1}{\mu} &\leq \mathbb{E}\left[\frac{1}{X}\right] \\ &\leq \frac{\mu^3 c - 3\mu^2 c^2 + 3\mu c^3 - c^4 + \sigma^2\mu^2 - \sigma^2\mu c + \sigma^4}{\mu^4 c - 3\mu^3 c^2 + 3\mu^2 c^3 - \mu c^4 + 2\sigma^2\mu^2 c - 3\sigma^2\mu c^2 + \sigma^2 c^3 + \sigma^4 c}. \end{aligned} \tag{21}$$

For convenience of notation we define $X_k := \sum_{i=1}^{k} a_i^2$. We first estimate the following mean.

Lemma 4. *Let $a_1, \ldots, a_n$ be chosen independently and uniformly at random from $[l, \ldots, u]$ for some integers $0 < l < u$, let $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{n-1}$ be given as in Lemma 1, and let $1 < k < n$.*

If $\gcd(a_1, \ldots, a_{k+1}) = 1$, there exists a function $f(k) \in \Theta(\frac{1}{k^2})$ such that

$$1 + \frac{1}{k} \leq \mathbb{E}\left[\|\boldsymbol{b}_k^*\|^2\right] \leq 1 + \frac{1}{k} + f(k), \tag{22}$$

and we can give an explicit expression for $f(k)$.

Note that using Theorem 2, we can compute an explicit upper bound in (22). We present this upper bound in the complete version of our paper.

Lemma 5. *Let $a_1, \ldots, a_n$ be chosen independently and uniformly at random from $[l, \ldots, u]$ for some integers $0 < l < u$. Then for any $1 \leq k < n - 1$ with $\gcd(a_1, \ldots, a_{k+1}) = 1$ we get*

$$\left|1 - \mathbb{E}\left[\|\boldsymbol{b}_{k+1}^*\|^2 / \|\boldsymbol{b}_k^*\|^2\right]\right| = O\left(\frac{1}{k}\right). \tag{23}$$

As with Lemma 4, we give explicit upper and lower bounds in the complete version of our paper.

Returning to Inequality (20), we will in fact only need the lower bound for $\mathbb{E}\left[\|\boldsymbol{b}^*_{k+1}\|^2/\|\boldsymbol{b}^*_k\|^2\right]$, to see that for any given reduction factor y we can find a $k(y)$ such that the mean is larger than y for any $k \geq k(y)$. More precisely:

Corollary 1. *Let $a_1, \ldots, a_n$ be chosen independently and uniformly at random from $[l, \ldots, u]$ for some integers $0 < l < u$, and let $y \in (1/4, 1)$ be fixed. Define $\hat{\mu} := \mathbb{E}[a_i^2]$ and $\hat{\sigma}^2 := Var(a_i^2)$.*

Suppose $k \leq n$ is given, and $\gcd(a_1, \ldots, a_{k+1}) = 1$. If k satisfies

$$1 - \frac{u^2 - \hat{\mu}}{(k+1)\hat{\mu}} - \frac{u^2\hat{\mu}}{(k+1)^2\hat{\mu}^2 + (k+1)\hat{\sigma}^2} > y, \tag{24}$$

then $\mathbb{E}\left[\frac{\|\boldsymbol{b}^*_{k+1}\|^2}{\|\boldsymbol{b}^*_k\|^2}\right] > y$.

Note that (24) can be solved explicitly for $k+1$, giving us a lower bound on k. We omit this calculation here as the solution is long and does not seem illuminating as to what size is sufficient for k. We will give some examples for given l, u, and y in Section 4.

If we can now also control the probability of $\|\boldsymbol{b}^*_{k+1}\|/\|\boldsymbol{b}^*_k\|$ deviating by more than a small amount from its mean for given $\boldsymbol{a}$, we have found a bound on the first term on the right in (20). For this we apply the inequality of Azuma-Hoeffding (cf. [4, 8]):

Let $Z_1, \ldots, Z_N$ be independent random variables, where Z_i takes values in the space Λ_i, and let $f : \prod_{i=1}^N \Lambda_i \to \mathbb{R}$. Define the following Lipschitz condition for the numbers $c_1, \ldots, c_N$:

(L) If the vectors $\boldsymbol{z}, \boldsymbol{z}' \in \prod_{i=1}^N \Lambda_i$ differ only in the jth coordinate, then $|f(\boldsymbol{z}) - f(\boldsymbol{z}')| \leq c_j$, for $j = 1, \ldots, N$.

Theorem 3 (see [9]). *If f is measurable and satisfies (L), then the random variable $X = f(Z_1, \ldots, Z_N)$ satisfies, for any $t \geq 0$,*

$$\begin{aligned} \Pr\left(X \geq \mathbb{E}[X] + t\right) &\leq e^{\frac{-2t^2}{\sum_{i=1}^N c_i^2}} \text{ and} \\ \Pr\left(X \leq \mathbb{E}[X] - t\right) &\leq e^{\frac{-2t^2}{\sum_{i=1}^N c_i^2}}. \end{aligned} \tag{25}$$

Thus, we indeed have a bound on the probability that a random variable satisfying (L) will deviate more than a little bit from its mean. Note that the bound gets stronger if we find small c_i and choose t large.

As with Lemma 5, we will ultimately just need one of the bounds, in this case (25).

Applied to our situation, we obtain the following result.

Corollary 2. *Let $a_1, \ldots, a_n$ be chosen independently and uniformly at random from $[l, \ldots, u]$ for some integers $0 < l < u$, and let $y \in (1/4, 1)$ be fixed.*

Suppose $k < n$ is given, and $\gcd(a_1, \ldots, a_{k+1}) = 1$. *If k satisfies* (24), *then*

$$\Pr\left(\frac{\|\boldsymbol{b}_{k+1}^*\|^2}{\|\boldsymbol{b}_k^*\|^2} \leq y\right) \quad \leq \quad e^{-t^2(k+1)^2\hat{c}}, \tag{26}$$

where $\hat{c} > 0$ depends on u and l, and $t > 0$ depends on u, l, and y.

To summarize, we proved in Lemma 3 and in Corollary 2 that for fixed reduction factor $y \in (1/4,\ 1)$, and for fixed l, u the following holds:

$$\Pr(\gcd(a_1, \ldots, a_{k+1}) > 1) \leq \left(\frac{1}{2}\right)^{(k+1)/2} \text{ for any } k \geq \frac{\log_2\left(\lfloor \frac{u}{2} \rfloor + 1\right)}{\log_2\left(\frac{u-l+1}{u-l+2}\right) + \frac{1}{2}} \tag{27}$$

and,

$$\Pr\left(\|\boldsymbol{b}_{k+1}^*\|^2 < y\|\boldsymbol{b}_k^*\|^2 \mid \gcd(a_1, \ldots, a_{k+1}) = 1\right) \quad \leq \quad e^{-t^2(k+1)^2\hat{c}}, \tag{28}$$

where $\hat{c} > 0$ depends on u and l, and $t > 0$ depends on u, l, and y. Adding the right-hand sides of Inequalities (27) and (28) yields the upper bound on $\Pr\left(\|\boldsymbol{b}_{k+1}^*\|^2/\|\boldsymbol{b}_k^*\|^2 \leq y\right)$ as stated in Theorem 1.

4 Discussion and Computations

If we again look at a basis $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_k$ that is obtained by applying the LLL reduction algorithm to an input basis of the format described in Lemma 1 in Subsection 2.2, we showed that for not too small k it will most likely have the following structure:

$$\left(\begin{array}{c|c} X_1 & X_2 \\ \hline \mathbf{0} & X_3 \end{array}\right).$$

The dimension of the submatrices X_1, X_2 and X_3 are $(k+1) \times k$, $(k+1) \times (n-(k+1))$, and $(n-(k+1)) \times (n-(k+1))$ respectively. All the elements of X_1 and X_2 may be non-zero, and X_3 is upper triangular.

In our computations, however, we see even more structure in the reduced basis, as discussed in the introduction. More precisely, we observe a reduced basis of the following form:

$$\left(\begin{array}{c|c} X_1 & \bar{X}_2 \\ \hline \mathbf{0} & I \end{array}\right), \tag{29}$$

that is, $X_3 = I$. So, a remaining question to address is why this is the case. We pointed out in Subsection 2.2 that if $\gcd(a_1, \ldots, a_{k+1}) = 1$, then it follows from the proof of Lemma 1 that the last nonzero element in each of the columns $\boldsymbol{b}_{k+1}, \ldots, \boldsymbol{b}_{n-1}$ must be ± 1. Therefore we know that the first column of X_3 is $(1, 0, \ldots, 0)^T$. The second column of X_3 is $(x, 1, 0, \ldots, 0)^T$, and so on. Here, again, x just denotes that the element may be non-zero. So, by subtracting x times vector $\boldsymbol{b}_{k+1}$ from vector $\boldsymbol{b}_{k+2}$ yields a unit column $(0, 1, 0, \ldots, 0)^T$ as the second column of X_3. This procedure can now be repeated for the remaining basis vectors to produce $X_3 = I$. Notice that these operations are elementary column operations.

Table 1. Column two gives an upper bound on $\Pr(\gcd(a_1, \ldots, a_{k+1}) > 1)$ for k greater than or equal to the value given in column 3, cf. Lemma 3. In the fourth column we give the value of $k(y)$ for reduction factor $y = 95/100$, such that $\mathbb{E}\left[\|\boldsymbol{b}^*_{k+1}\|^2/\|\boldsymbol{b}^*_k\|^2\right] > y$ for all $k \geq k(y)$.

Interval	Probability $\leq$	$k \geq$	$k(y)$
$[100, \ldots, 1,000]$	0.0014	19	36
$[15,000, \ldots, 150,000]$	0.000008	34	36

Observation 3. *If we apply the above column operations to the basis given in Lemma 1, then every part of the analysis where we assumed the basis to be given as in Lemma 1 also works for this new lattice basis.*

So, indeed, $\ker_{\mathbb{Z}}(\boldsymbol{a})$ has a basis of the structure given in (29), and we observe in our computational experiments that such a basis is y-reduced if the input vector $\boldsymbol{a}$ satisfies the assumptions given in the beginning of Section 3. Here we give qualitative arguments for why this is the case.

Suppose that the elementary column operations performed to obtain $X_3 = I$ yields a basis that is not size reduced. Then we can add any linear integer combination of the first k basis vectors to any of the last $n - (k + 1)$ vectors without destroying the identity matrix structure of submatrix X_3, since the first k vectors have zeros as the last $n - (k + 1)$ elements. These elementary column operations can be viewed as size reductions. If we consider the first k basis vectors we empirically observe that the absolute values of the non-zero elements (i.e., elements in submatrix X_1) are small, and that the vectors are almost orthogonal since they are reduced. Since all a_i-elements are positive, each basis vector has a mixture of positive, negative and zero elements. Apparently, once these size reductions are done, the basis is reduced, i.e., no further swaps are needed. This is in line with the results presented in Subsection 3 that the expected length of the Gram-Schmidt vectors $\boldsymbol{b}^*_k$ becomes arbitrarily close to one with increasing values of k, see also reduction Condition (7).

In Table 1 we give an upper bound on $\Pr(\gcd(a_1, \ldots, a_{k+1}) > 1)$ for k greater than or equal to the value given in the table. This probability is computed according to Lemma 3 for the intervals $[l, \ldots, u] = [100, \ldots, 1,000]$ and $[l, \ldots, u] = [15,000, \ldots, 150,000]$. That is, for the interval $[l, \ldots, u] = [100, \ldots, 1,000]$, the probability that $\gcd(a_1, \ldots, a_{k+1}) > 1$ is less than or equal to 0.0014 for $k \geq 19$. Notice that this value of k is only depending on l and u, and not on n. In the table we also give the value of $k(y)$ for reduction factor $y = 95/100$ such that $\mathbb{E}\left[\|\boldsymbol{b}^*_{k+1}\|^2/\|\boldsymbol{b}^*_k\|^2\right] > y$ for all $k \geq k(y)$. The values given in the table are very close to the values we observe empirically.

A comprehensive computational study for single- and multi-row instances is presented in the complete version of our paper.

To summarize, we have observed empirically that for larger instances, only relatively few of the $\boldsymbol{x}$-variables have a non-trivial translation into $\boldsymbol{\lambda}$-variables. This is well in line with the theoretical result reported in Table 1 that the expected value of $\|\boldsymbol{b}^*_{k+1}\|^2/\|\boldsymbol{b}^*_k\|^2$ is greater than the reduction factor for all

$k \geq 36$ for both of the considered intervals. Yet, we observe that if we solve the instances using Reformulation (3) rather than the original formulation (1), the number of branch-and-bound nodes needed in $\boldsymbol{\lambda}$-space could be one to two orders of magnitude smaller than in the original space. Thus, there is a computationally important structure in the $\boldsymbol{\lambda}$-space. But this structure is not arbitrarily "spread", but contained in a limited subset of the variables.

Suppose now that a row $\boldsymbol{ax} = b$ is part of a larger problem formulation, and that we expect this row to be important in the formulation in the sense of obtaining a good branching direction or a useful cut. If we wish to obtain this information through the lattice reformulation, then we need to be careful in indexing the $\boldsymbol{x}$-variables appropriately.

Acknowledgement. We wish to acknowledge the fruitful discussion with Andrea Lodi and Laurence Wolsey that lead to the question addressed in this contribution. We also want to thank Hendrik Lenstra for his helpful suggestions.

References

1. Aardal, K., Hurkens, C.A.J., Lenstra, A.K.: Solving a system of linear Diophantine equations with lower and upper bounds on the variables. Mathematics of Operations Research 25(3), 427–442 (2000)
2. Aardal, K., Lenstra, A.K.: Hard equality constrained integer knapsacks. Mathematics of Operations Research 29(3), 724–738 (2004); Erratum: Mathematics of Operations Research 31(4), 846 (2006)
3. Aardal, K., Wolsey, L.A.: Lattice based extended formulations for integer linear equality systems. Mathematical Programming 121, 337–352 (2010)
4. Azuma, K.: Weighted sums of certain dependent random variables. Tôhoku Mathematical Journal 19(3), 357–367 (1967)
5. Cook, W., Rutherford, T., Scarf, H.E., Shallcross, D.: An implementation of the generalized basis reduction algorithm for integer programming. ORSA Journal on Computing 5, 206–212 (1993)
6. Cornuéjols, G., Dawande, M.: A class of hard small 0-1 programs. INFORMS Journal on Computing 11, 205–210 (1999)
7. Cornuéjols, G., Urbaniak, R., Weismantel, R., Wolsey, L.A.: Decomposition of Integer Programs and of Generating Sets. In: Burkard, R.E., Woeginger, G.J. (eds.) ESA 1997. LNCS, vol. 1284, pp. 92–103. Springer, Heidelberg (1997)
8. Hoeffding, W.: Probability inequalities for sums of bounded random variables. Journal of the American Statistical Association 58, 13–30 (1963)
9. Janson, S.: On concentration of probability. In: Contemporary Combinatorics. Bolyai Soc. Math. Stud, vol. 10, pp. 289–301. János Bolyai Math. Soc., Budapest (2002)
10. Kannan, R.: Algorithmic geometry of numbers. In: Annual Review of Computer Science, vol. 2, pp. 231–267. Annual Reviews, Palo Alto (1987)
11. Krishnamoorthy, B., Pataki, G.: Column basis reduction and decomposable knapsack problems. Discrete Optimization 6(3), 242–270 (2009)
12. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261(4), 515–534 (1982)

13. Lenstra Jr., H.W.: Integer programming with a fixed number of variables. Mathematics of Operations Research 8(4), 538–548 (1983)
14. Lenstra Jr., H.W.: Flags and lattice basis reduction. In: Casacuberta, C., Miro-Roig, R.M., Verdera, J., Xambo-Descamps, S. (eds.) European Congress of Mathematics, Barcelona, July 10-14, 2000, Volume I. Progress in Mathematics, vol. 202, pp. 37–51. Birkhäuser, Basel (2001)
15. Lenstra Jr., H.W.: Lattices. In: Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography. Mathematical Science Research Institute Publications, vol. 44, pp. 127–181. Cambridge University Press, Cambridge (2008)
16. Louveaux, Q., Wolsey, L.A.: Combining problem structure with basis reduction to solve a class of hard integer programs. Mathematics of Operations Research 27(3), 470–484 (2002)
17. Lovász, L., Scarf, H.E.: The generalized basis reduction algorithm. Mathematics of Operations Research 17, 751–764 (1992)
18. Mehrotra, S., Li, Z.: Branching on hyperplane methods for mixed integer linear and convex programming using adjoint lattices. Journal of Global Optimization 49(4), 623–649 (2011)
19. Pittenger, A.O.: Sharp mean-variance bounds for Jensen-type inequalities. Statistics & Probabability Letters 10(2), 91–94 (1990)

All-or-Nothing Generalized Assignment with Application to Scheduling Advertising Campaigns

Ron Adany[1], Moran Feldman[2], Elad Haramaty[2], Rohit Khandekar[3], Baruch Schieber[4], Roy Schwartz[5], Hadas Shachnai[2], and Tami Tamir[6]

[1] Computer Science Department, Bar-Ilan University, Ramat-Gan 52900, Israel
adanyr@cs.biu.ac.il
[2] Computer Science Department, Technion, Haifa 32000, Israel
{moranfe,eladh,hadas}@cs.technion.ac.il
[3] Knight Capital Group, Jersey City, NJ 07310
rkhandekar@gmail.com
[4] IBM T.J. Watson Research Center, Yorktown Heights, NY 10598
sbar@us.ibm.com
[5] Microsoft Research, One Microsoft Way, Redmond, WA 98052
roysch@microsoft.com
[6] School of Computer science, The Interdisciplinary Center, Herzliya, Israel
tami@idc.ac.il

Abstract. We study a variant of the *generalized assignment problem* (GAP) which we label *all-or-nothing* GAP *(*AGAP*)*. We are given a set of items, partitioned into n groups, and a set of m bins. Each item ℓ has size $s_\ell > 0$, and utility $a_{\ell j} \geq 0$ if packed in bin j. Each bin can accommodate at most one item from each group, and the total size of the items in a bin cannot exceed its capacity. A group of items is *satisfied* if all of its items are packed. The goal is to find a feasible packing of a subset of the items in the bins such that the total utility from satisfied groups is maximized. We motivate the study of AGAP by pointing out a central application in scheduling advertising campaigns.

Our main result is an $O(1)$-approximation algorithm for AGAP instances arising in practice, where each group consists of at most $m/2$ items. Our algorithm uses a novel reduction of AGAP to maximizing submodular function subject to a matroid constraint. For AGAP instances with fixed number of bins, we develop a randomized *polynomial time approximation scheme (PTAS)*, relying on a non-trivial LP relaxation of the problem.

We present a $(3+\varepsilon)$-approximation as well as PTASs for other special cases of AGAP, where the utility of any item does not depend on the bin in which it is packed. Finally, we derive hardness results for the different variants of AGAP studied in the paper.

1 Introduction

Personalization of advertisements (ads) allows commercial entities to aim their ads at specific audiences, thus ensuring that each target audience receives its

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 13–24, 2013.

specialized content in the desired format. Recent media research reports [14,13] show that global spending on TV ads exceeded $323B in 2011, and an average viewer watched TV for 153 hours per month, with the average viewing time consistently increasing. Based on these trends and on advances in cable TV technology, personalized TV ads are expected to increase revenues for TV media companies and for mobile operators [4,7,17]. The proliferation of alternative media screens, such as cell-phones and tablets, generate new venues for personalized campaigns targeted to specific viewers, based on their interests, affinity to the advertised content, and location. In fact, ads personalization is already extensively used on the Internet, e.g., in Google AdWords [6]. Our study is motivated by a central application in personalized ad campaigns scheduling, introduced to us by SintecMedia [19].

An *advertising campaign* is a series of advertisement messages that share a single idea and theme which make up an integrated marketing communication. Given a large set of campaigns that can be potentially delivered to the media audience, a service provider attempts to fully deliver a subset of campaigns that maximizes the total revenue, while satisfying constraints on the placement of ads that belong to the same campaign, as well as possible placement constraints among conflicting campaigns. In particular, to increase the number of viewers exposed to an ad campaign, one constraint is that each commercial break contains no more than a single ad from this campaign.[1] Also, each ad has a given length (=size), which remains the same, regardless of the commercial break in which it is placed. This generic assignment problem defines a family of all-or-nothing variants of the *generalized assignment problem* (GAP).

Let $[k]$ denote $\{1, \ldots, k\}$ for an integer k. In *all-or-nothing* GAP (or AGAP), we are given a set of m bins, where bin $j \in [m]$ has capacity c_j, and a set of N items partitioned into n groups $G_1, \ldots, G_n$. Each group $i \in [n]$, consists of k_i items, for some $k_i \geq 1$, such that $\sum_i k_i = N$. Each item $\ell \in [N]$ has a size $s_\ell > 0$ and a non-negative utility $a_{\ell j}$ if packed in bin $j \in [m]$. An item can be packed in at most one bin, and each bin can accommodate at most one item from each group. Given a packing of a subset of items, we say that a group G_i is *satisfied* if all items in G_i are packed. The goal is to pack a subset of items in the bins so that the total utility of satisfied groups is maximized. Formally, we define a packing to be a function $p : [N] \to [m] \cup \{\perp\}$. If $p(\ell) = j \in [m]$ for $\ell \in [N]$, we say that item ℓ is packed in bin j. If $p(\ell) = \perp$, we say that item ℓ is not packed. A packing is *admissible* if $\sum_{\ell \in p^{-1}(j)} s_\ell \leq c_j$ for all $j \in [m]$, and $|p^{-1}(j) \cap G_i| \leq 1$ for all $j \in [m]$ and $i \in [n]$. Given a packing p, let $S_p = \left\{ i \in [n] \mid G_i \subseteq \cup_{j \in [m]} \{p^{-1}(j)\} \right\}$ denote the set of groups satisfied by p. The goal in AGAP is to find an admissible packing p that maximizes the utility: $\sum_{i \in S_p} \sum_{\ell \in G_i} a_{\ell p(\ell)}$.

We note that AGAP is NP-hard already when the number of bins is fixed. Such instances capture campaign scheduling in a given time interval (of a few hours) during the day. We further consider the following special cases of AGAP, which are of practical interest. In *all-or-nothing group packing*, each group G_i has a

[1] Indeed, overexposure of ads belonging to the same campaign in one break may cause lack of interest, thus harming the success of the campaign.

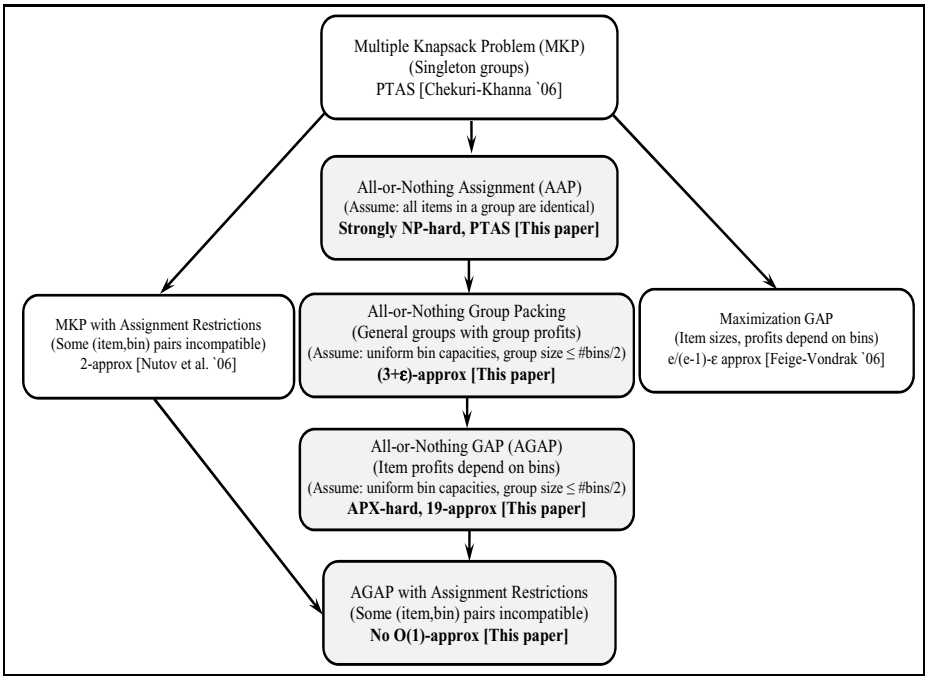


Fig. 1. Summary of our approximation and hardness results and comparison with related problems. An arrow from problem A to B indicates that A is a special case of B.

profit $P_i > 0$ if all items are packed, and 0 otherwise. Thus, item utilities do not depend on the bins. In the *all-or-nothing assignment problem* (AAP), all items in G_i have the same size, $s_i > 0$, and same utility $a_i \geq 0$, across all bins.

Note that the special case of AGAP where all groups consist of a *single* item yields an instance of classic GAP, where each item has the same size across the bins. The special case of AAP where all groups consist of a *single* item yields an instance of the multiple knapsack problem. Clearly, AGAP is harder to solve than these two problems. One reason is that, due to the *all-or-nothing* requirement, we cannot eliminate large items of small utilities, since these items may be essential for satisfying a set of most profitable groups. Moreover, even if the satisfied groups are known *a-priori*, since items of the same group cannot be placed in one bin, common techniques for classical packing, such as rounding and enumeration, cannot be applied.

1.1 Our Results

Figure 1 summarizes our contributions for different variants of AGAP and their relations to each other. Even relatively special instances of AAP are NP-hard. Furthermore, in the full paper [1], we show that with slight extensions, AGAP becomes hard to approximate within any bounded ratio. Thus, we focus in this paper on deriving approximation algorithms for AGAP and the above special cases.

Given an algorithm $\mathcal{A}$, let $\mathcal{A}(I), OPT(I)$ denote the utility of $\mathcal{A}$ and an optimal solution for a problem instance I, respectively. For $\rho \geq 1$, we say that $\mathcal{A}$ is a ρ-approximation algorithm if, for any instance I, $\frac{OPT(I)}{\mathcal{A}(I)} \leq \rho$.

In [1] we show that AGAP with non-identical bins is hard to approximate within any constant ratio, even if the utility of an item is identical across the bins. Thus, in deriving our results for AGAP, we assume the bins are of uniform capacities. Our main result (in Section 2) is a $(19 + \varepsilon)$-approximation algorithm for AGAP instances arising in practice, where each group consists of at most $m/2$ items.

Interestingly, AGAP with a fixed number of bins admits a randomized PTAS (see Section 3.1). In Section 3.2 we show that, for the special case where all items have unit sizes, an $\frac{e}{e-1}$-approximation can be obtained by reduction to submodular maximization with knapsack constraint. In Section 3.3 we give a $(3 + \varepsilon)$-approximation algorithm for All-or-Nothing Group Packing. This ratio can be improved to $(2+\varepsilon)$ if group sizes are relatively small. The details of these results are omitted due to lack of space and are given in the full paper [1].

In [1] we also present PTASs for two subclasses of instances of AAP. The first is the subclass of instances with unit-sized items, the second is the subclass of instances in which item sizes are drawn from a divisible sequence,[2] and group cardinalities can take the values $k_1, \ldots, k_r$, for some constant $r \geq 1$. Such instances arise in our campaign scheduling application. Indeed, the most common lengths for TV ads are $15, 30$ and 60 seconds [21,12]. Also, there are standard sizes of $150, 300$ and 600 pixels for web-banners on the Internet [20].

Finally, hardness results for the different all-or-nothing variants of GAP studied are also given in [1].

Technical Contribution: Our approximation algorithm for AGAP (in Section 2) uses a novel reduction of AGAP to maximizing submodular function subject to matroid constraint. At the heart of our reduction lies the fact that the sequence of sizes of large groups can be discretized to yield a logarithmic number of size categories. Thus, we can guarantee that the set of fractionally packed groups, in the initial Maximization Phase of the algorithm, has a total size at most m. Our reduction to submodular maximization encodes this knapsack constraint as a matroid constraint, by considering feasible vectors $(n_1, \ldots, n_H)$, where n_h gives the number of groups taken from size category h, for $1 \leq h \leq H$. These vectors (which are *implicitly* enumerated in polynomial time) are used for defining the matroid constraint.

Our definition of the submodular set function, $f(S)$ (see Section 2), which finds *fractional* packing of items, in fact guarantees that the rounding that we use for group sizes (to integral powers of $1+\varepsilon$, for some $\varepsilon > 0$), causes only small harm to the approximation ratio. This allows also to define a non-standard polynomial time implementation of an algorithm of [2], for maximizing a submodular function under matroid constraint. More precisely, while the universe for our submodular function f is of exponential size, we show that f can be computed in polynomial time.

[2] A sequence $d_1 < d_2 < \cdots < d_z$ is a *divisible* if d_{i-1} divides d_i for all $1 < i \leq z$.

Our randomized approximation scheme for AGAP instances with constant number of bins (in Section 3.1) is based on a non-trivial LP relaxation of the problem. While the resulting LP has polynomial size when the number of bins is fixed, solving it in polynomial time for general instances (where the number of variables is exponentially large) requires sophisticated use of separation oracles, which is of independent interest. The fractional solution obtained for the LP is rounded by using an approximation technique of [9,8] for maximizing a submodular function subject to fixed number of knapsack constraints.

1.2 Related Work

All-or-nothing GAP generalizes several classical problems, including GAP (with same sizes across the bins), the *multiple knapsack problem* (MKP), *multiple knapsack with assignment restrictions* (MKAR) [15], and the *generalized multi assignment problem*. In this section we briefly summarize the state of the art for these problems.

As mentioned above, the special case where all groups consist of a *single* item yields an instance of GAP, where each item takes a single size over all bins. GAP is known to be APX-hard already in this case, even if there are only two possible item sizes, each item can take two possible profits, and all bin capacities are identical [3]. The best approximation ratio obtained for GAP is $\frac{e}{e-1} - \varepsilon$ [5].

In minimum GAP (see, e.g., [10]), there are m machines and n jobs. Each machine i is available for T_i time units, and each job has a processing time (size), and a cost of being assigned to a machine. The goal is to schedule all the jobs at minimum total cost, where each job needs to be assigned to a single machine. The paper [18] gives an algorithm which minimizes the total cost, using a schedule where each machine i completes within $2T_i$ time units, $1 \leq i \leq m$.

The generalized multi-assignment problem extends minimum GAP to include multiple assignment constraints. Job processing times and the costs depend on the machine to which they are assigned, the objective is to minimize the costs, and all the jobs must be assigned. This problem was discussed in [16], where Lagrangian dual-based branch-and-bound algorithms were used for obtaining an exact solution for the problem.[3]

We are not aware of earlier work on AGAP or *all-or-nothing* variants of other packing problems.

2 Approximation Algorithm for AGAP

In this section we consider general AGAP instances, where each item ℓ has a size $s_\ell \in (0, 1]$ and arbitrary utilities across the bins. We assume throughout this section that all bins are of the same (unit) capacity. Our approach is based on a version of AGAP, called RELAXED-AGAP, obtained by relaxing the constraint that the total size of items packed in a bin must be at most 1, and by defining the *utility* of a solution to RELAXED-AGAP slightly differently. We prove that the

[3] The running time of this algorithm is not guaranteed to be polynomial.

maximum utility of a solution to RELAXED-AGAP upper bounds the objective value of the optimal AGAP solution. Our algorithm proceeds in two phases.

Maximization Phase: The algorithm approximates the optimal utility of RELAXED-AGAP in polynomial time, by applying a novel reduction to submodular function maximization under matroid constraints. Let S denote the subset of groups assigned by this RELAXED-AGAP solution.

Filling Phase: The algorithm next chooses a subset $S' \subseteq S$ whose utility is at least a constant fraction of the utility of S. Then, the algorithm constructs a feasible solution for AGAP that assigns the groups in S' (not necessarily to the same bins as the RELAXED-AGAP solution) and achieves AGAP value that is at least half of the utility of S', thereby obtaining $O(1)$-approximation for AGAP.

2.1 Maximization Phase

RELAXED-AGAP**:** The input for RELAXED-AGAP is the same as that for AGAP. A feasible RELAXED-AGAP solution is a subset S of the groups whose total size is no more than m (the total size of the bins) and a *valid* assignment p of the items in groups in S to bins; a valid assignment is defined as one in which no two items from the same group are assigned to the same bin. In RELAXED-AGAP, we do not have a constraint regarding the total size of the items assigned to a single bin. Given a solution (S, p) and a bin $j \in [m]$, let $p^{-1}(j) \subseteq [N]$ be the set of items assigned by p to bin j. The utility of a solution (S, p) is the sum of the utility contributions of the bins. The utility contribution of a bin $j \in [m]$ is the maximum value from (fractionally) assigning items in $p^{-1}(j)$ to j satisfying its unit capacity. In other words, we solve for bin j the *fractional knapsack* problem. To define this more formally, we introduce some notation.

Definition 1. *Given a subset $I \subseteq [N]$ of items and a bin j, define $\pi(j, I) = \max_{\boldsymbol{w}} \sum_{\ell \in I} w_\ell a_{\ell j}$, where the maximum is taken over all weight vectors $\boldsymbol{w} \in \Re_+^{|I|}$ that assign weights $w_\ell \in [0, 1]$ to $\ell \in I$, satisfying $\sum_{\ell \in I} w_\ell s_\ell \leq 1$.*

The utility of a solution (S, p) is given by $\sum_{j \in [m]} \pi(j, p^{-1}(j))$. The RELAXED-AGAP is to find a solution with maximum utility.

We can extend Definition 1 to *multisets* as follows.

Definition 2. *A multiset I of $[N]$ can be viewed as a function $I : [N] \to \mathbb{Z}_+$ that maps each $\ell \in [N]$ to a non-negative integer equal to the number of copies of ℓ present in I. Define $\pi(j, I) = \max_{\boldsymbol{w}} \sum_{\ell \in [N]} w_\ell a_{\ell j}$, where the maximum is taken over all weight vectors $\boldsymbol{w} \in \Re_+^N$ that assign weights $w_\ell \in [0, I(\ell)]$ to $\ell \in [N]$ satisfying $\sum_{\ell \in [N]} w_\ell s_\ell \leq 1$.*

It is easy to determine $\boldsymbol{w}$ that maximizes the utility contribution of bin j. Order the items in I as $\ell_1, \ldots, \ell_b$ in a non-increasing order of their ratio of utility to size, i.e., $a_{\ell_1 j}/s_{\ell_1} \geq a_{\ell_2 j}/s_{\ell_2} \geq \cdots \geq a_{\ell_b j}/s_{\ell_b}$. Let d be the maximum index such that $s = \sum_{i=1}^{d} s_{\ell_i} \leq 1$. Set $w_1 = \cdots = w_d = 1$. If $s < 1$ and $d < b$, set $w_{d+1} = (1 - s)/s_{\ell_{d+1}}$. Set the other weights $w_{d+2} = \cdots = w_b = 0$.

Solving RELAXED-AGAP **Near-Optimally:** Recall that a valid assignment of a subset of items in $[N]$ to bins is one in which no two items from a group get assigned to the same bin. Now define a universe U as follows:

$$U = \{(G, L) \mid L \text{ is a valid assignment of } \textit{all} \text{ items in group } G \text{ to bins } [m]\}$$

A subset $S \subseteq U$ defines a multiset of groups that appear as the first component of the pairs in S. Below, we use $G(S)$ to denote the multiset of such groups. For a subset $S \subseteq U$ and a bin $j \in [m]$, let $I_j = \uplus_{(G,L)\in S} L^{-1}(j)$ be the *multiset* union of sets of items mapped to j over all elements $(G, L) \in S$. Note that I_j can indeed be a multiset since S may contain two elements (G_1, L_1) and (G_2, L_2) with $G_1 = G_2$. Now define $f(S) = \sum_{j\in[m]} \pi(j, I_j)$. The following important but simple claim is proved in [1].

Claim 1. *The function $f(S)$ is non-decreasing and submodular.*

To identify subsets $S \subset U$ that define feasible RELAXED-AGAP solutions, we need two constraints.

Constraint 1. The subset S does not contain two elements (G_1, L_1) and (G_2, L_2) such that $G_1 = G_2$.

Constraint 2. The total size of the groups in $G(S)$, counted with multiplicities, is at most m, i.e., $\sum_{(G,L)\in S} \sum_{\ell\in G} s_\ell \leq m$.

Constraint 1 is easy to handle since it is simply the independence constraint in a partition matroid. Unfortunately, Constraint 2, which is essentially a knapsack constraint, is not easy to handle over the exponential-sized universe U.

Handling Constraint 2 Approximately in Polynomial Time: To this end, we split the groups into a logarithmic number of classes. Fix $\epsilon > 0$. Class 0 contains all groups G such that $s(G) := \sum_{\ell\in G} s_\ell \leq \varepsilon m/n$. For $h \geq 1$, class h contains all groups G with $s(G) \in (\varepsilon m/n \cdot (1+\varepsilon)^{h-1}, \varepsilon m/n \cdot (1+\varepsilon)^h]$. We use $\mathcal{C}_h$ to denote class h. Since $s(G) \leq m$ for all groups G, there are only $H = O(1/\varepsilon \cdot \log(n/\varepsilon))$ non-empty classes. We enforce an upper bound of m on the total size of groups in $G(S)$ by enforcing an upper bound on the total size of groups in $G(S)$ from each class separately. We call a vector $(y_1, \ldots, y_H) \in \mathbb{Z}_+^H$ of non-negative integers *legal* if $\sum_{h=1}^H y_h \leq H(1+1/\varepsilon)$. Note that the number of legal vectors is $O(\binom{H(1+1/\varepsilon)}{H}) = O(2^{H(1+1/\varepsilon)})$, which is polynomial in m and n.

Lemma 2. *For any $S \subseteq U$ satisfying Constraint 2, there exists a legal vector $(y_1, \ldots, y_H)$ such that for all $h \in [H]$, the number of groups in $G(S)$, counted with multiplicities, that are in $\mathcal{C}_h$ is at most $\hat{y}_h := \lfloor y_h n/(H(1+\varepsilon)^{h-1})\rfloor$.*

This lemma implies, in particular, that the optimum solution to AGAP satisfies the above property as well. With this motivation, we define $U_h = \{(G, L) \in U \mid G \in \mathcal{C}_h\}$ and define a new constraint as follows.

Constraint 2′ for a Fixed Legal Vector $(y_1, \ldots, y_H)$**.** For each $1 \leq h \leq H$, the number of groups in $G(S)$, counted with multiplicities, that are in $\mathcal{C}_h$ is at most $\hat{y}_h$ as defined in Lemma 2.

Lemma 3. *Fix a legal vector* $(y_1, \ldots, y_H)$*. The collection of all* $S \subseteq U$ *satisfying Constraint 1 and Constraint* 2′ *for this vector defines a* laminar *matroid* $M(y_1, \ldots, y_H)$ *over* U*. Furthermore, any independent set* $S \subseteq U$ *in this matroid satisfies* $\sum_{(G,L)\in S} \sum_{\ell \in G} s_\ell \leq m((1+\varepsilon)^2 + \varepsilon)$.

Given a legal vector $(y_1, \ldots, y_H)$, consider the SUBMOD-MATROID problem of maximizing the non-decreasing submodular function $f(S)$ over all independent sets in the matroid $M(y_1, \ldots, y_H)$. Recall that Nemhauser et al. [11] proved that a greedy algorithm that starts with an empty set and iteratively adds "most profitable" element to it while maintaining independence, as long as possible, is a 1/2-approximation. Each iteration can be implemented in polynomial time as follows. Given a current solution S and a group G, the problem of finding the assignment L that increases the utility f relative to S by the maximum amount can be cast as a bipartite matching problem. To see this, create a bipartite graph with elements in G as vertices on the left-hand-side and bins as vertices on the right-hand-side. For $\ell \in G$ and a bin j, add an edge (ℓ, j) with weight equal to the amount by which contribution of bin j would increase if ℓ is added to bin j. This quantity, in turn, can be computed by solving a fractional knapsack problem on bin j. The maximum weight assignment corresponds to the maximum-weight matching in this graph.

In the maximization phase, we enumerate over all (polynomially many) legal vectors and compute a 1/2-approximate solution to the corresponding SUBMOD-MATROID problem. In the end, we pick the maximum valued solution over all such solutions.

Improving the Approximation to $(e-1)/e$**:** Instead of the greedy algorithm of Nemhauser et al. [11], we can also use the $\frac{e}{e-1}$-approximate Continuous Greedy Algorithm of Calinescu et al. [2]. Some care is needed to show that this algorithm can indeed be implemented in polynomial time in our setting. We omit the details due to lack of space.

In summary, we find a set $S^* \subseteq U$ such that (1) each group appears at most once in $G(S^*)$, (2) the total size of the groups in $G(S^*)$ is at most $m((1+\varepsilon)^2+\varepsilon) \leq m(1+4\varepsilon)$ (if $\varepsilon \leq 1$), and (3) $f(S^*)$ is at least 1/2 (or $(e-1)/e$ if we use the algorithm of Calinescu et al. [2]) of the maximum value achieved by such sets.

2.2 Filling Phase

We show how to choose a subset of the groups in $G(S^*)$ and a feasible assignment of the items in these groups such that the utility of these assignments is a constant fraction of $f(S^*)$. In the description we use parameters $u, v > 0$, whose value will be optimized later.

Lemma 4. *Assume $v \geq 4$, $v(1+4\varepsilon) < u$ and $k_{\max} := \max_i k_i \leq m/2$. In polynomial time, we can compute a subset of groups $F \subseteq G(S^*)$ and a feasible assignment of their items to the bins, forming a feasible solution to* AGAP *with value at least $f(S^*) \cdot \min\{1/u, \frac{1}{2}(1/v(1+4\varepsilon) - 1/u)\}$.*

Recall that $f(S^*) = \sum_j \pi(j, I_j)$, where I_j is a set of items mapped to bin j over all $(G, L) \in S^*$. Since S^* satisfies Constraint 1, we do not have two elements $(G, L_1), (G, L_2) \in S^*$ for any G. We now subdivide the value $f(S^*)$ into the groups $G \in G(S^*)$, naturally, as follows. Suppose that $\pi(j, I_j)$ is achieved by a weight-vector $\boldsymbol{w}(j)$. Fix any such optimum weight vector $\boldsymbol{w}^*(j)$ for each j. These vectors, when combined, give a weight vector $\boldsymbol{w}^* \in \Re_+^N$, assigning a unique weight w^*_ℓ to each $\ell \in [N]$. We define the *contribution* of a group $G \in G(S^*)$ to $f(S^*)$ as $\sigma^*(G) = \sum_{\ell \in G} w^*_\ell a_{\ell L(\ell)}$ where $(G, L) \in S^*$.

Proof of Lemma 4. If there is a group $G \in G(S^*)$ with $\sigma^*(G) \geq f(S^*)/u$, we output $F = \{G\}$ with the best assignment of items in G to bins (computed using maximum matching, as described in the previous section) as solution. Clearly, the utility of this solution is at least $f(S^*)/u$.

Suppose that no such group exists. In this case we consider the groups $G \in G(S^*)$ in non-increasing order of $\sigma^*(G)/s(G)$. Choose the longest prefix in this order whose total size is at most m/v. Let $T \subset S^*$ be the solution induced by these groups. We first argue that $T \neq \emptyset$. Note that T can be empty only if the first group G in the above order has size more than m/v. Thus $\sigma^*(G)/(m/v) > \sigma^*(G)/s(G) \geq f^*(S)/(m(1+4\varepsilon))$. The second inequality holds since the total size of groups in $G(S^*)$ is at most $m(1+4\varepsilon)$ and the "density" $\sigma^*(G)/s(G)$ of G is at least the overall density of $G(S)$, which in turn is at least $f^*(S)/(m(1+4\varepsilon))$. This implies that $\sigma^*(G) > f^*(S)/(v(1+4\varepsilon)) > f^*(S)/u$, a contradiction.

The following three steps find a feasible solution to AGAP that consists of the groups in $G(T)$ and whose value is at least $f(T)/2$.

1. Eliminate all zero weights: Let $\boldsymbol{w} \in \Re_+^N$ be the weight vector that determines the value $f(T)$. Note that the weight w_ℓ assigned to some of the items ℓ in groups in $G(T)$ may be zero. We modify the assignment of items in the solution T so that no item would have zero weight. Note that if an item ℓ assigned to bin j in solution S has $w_\ell = 0$, the total size of the items assigned to bin j in S is at least 1. It follows that there are at most $\lfloor m/v \rfloor$ bins that may contain items of zero weight, since the total size of all items assigned in T is no more than m/v.

For each item with zero weight that belongs to a group G_i, there is at least one bin j such that the total size of the items assigned to bin j is less than 1 and no items from group G_i are assigned to bin j. This follows since $|G_i| + \lfloor m/v \rfloor \leq m/2 + \lfloor m/v \rfloor < m$. It follows that this item can be assigned to bin j and be assigned non-zero weight. We can continue this process as long as there are items with zero weight, thereby, eliminating all zero weights.

2. Evicting overflowed items: Suppose there are a (respectively, b) bins that are assigned items of total size more than 1 (respectively, more than $1/2$ and at most 1). Call these bins 'full' (respectively, 'half full'). Since the total volume of

packed items is at most m/v, we have $a + b/2 \leq m/v$. Next, we remove some items from these a full bins to make the assignment feasible. Consider such a bin. We keep in this bin either all the items assigned to it that have weight equal to 1, or the unique item that has weight strictly between 0 and 1, whichever contributes more to $f(T)$. In this step, we lose at most half of the contribution of the full bins to $f(T)$. We further evict all items assigned to the least profitable $\lfloor (m-a)/2 \rfloor$ non-full bins. In this step, we lose at most half of the contribution of the non-full bins to $f(T)$.

3. Repacking evicted items: We now repack all the evicted items to maintain feasibility of the solution. We first repack evicted items of size at least half. Note that are at most a such items from full bins, and at most b such items from half full bins. These $a + b$ items can be packed into evicted $\lfloor (m-a)/2 \rfloor$ bins by ensuring $a + b \leq \lfloor (m-a)/2 \rfloor$, i.e., $3a + 2b < m$. This is indeed true since $v \geq 4$ together with $a + b/2 \leq m/v$ implies $4a + 2b \leq m$.

We are now left only with items whose size is less than half to repack. For each such item from group i, we find a bin that does not contain another item from group i and whose total size is less than half, and insert the item to this bin. Note that, since the size of the item is less than half, the solution remains feasible. Since the total size of the items to be packed is at most m/v, there are at most $\lfloor 2m/v \rfloor$ bins of size at least half. Thus, we are guaranteed to find such a bin in case $m - \lfloor 2m/v \rfloor - k_i \geq 0$, i.e., $k_i \leq \lceil m(1 - 2/v) \rceil$.

We now bound $f(T)$. Since the contribution of any group to $f(S^*)$ is no more than $f(S^*)/u$, the contribution of the groups in T is at least $f(S^*) \cdot (1/v(1+4\varepsilon) - 1/u)$. Recall that the reduction in $f(T)$ due to the eviction of items is at most half of $f(T)$. Thus the value of the final solution is at least $f(S^*) \cdot \frac{1}{2}(1/v(1+4\varepsilon) - 1/u)$. This completes the proof of the lemma. ∎

Now, to bound the overall approximation ratio, we seek the values of u and v satisfying $1/u = \frac{1}{2}(1/(v(1+4\varepsilon)) - 1/u)$. Thus, we set $u = 3v(1+4\varepsilon)$. For $v = 4$ and $u = 12(1+4\varepsilon)$, we get a ratio of $\frac{1}{12(1+3\varepsilon)}$. Since we lost a factor of $1/2$ (or $(e-1)/e$) in the maximization phase, we get an overall $24(1+4\varepsilon)$-approximation (or $12(1+\varepsilon)\frac{e}{e-1}$-approximation).

This proves the following theorem.

Theorem 1. AGAP *admits a polynomial-time* $12(1+\varepsilon)\frac{e}{e-1}$*-approximation for any* $0 < \epsilon < 1$*, provided any group has at most* $k_{\max} \leq m/2$ *items.*

3 Approximating Special Cases of AGAP

In this section we consider several special cases of AGAP. We assume throughout the discussion that the bins have uniform (unit) capacities.

3.1 Approximation Scheme for Constant Number of Bins

We formulate the following LP relaxation for AGAP. For every group G_i, we define $\mathcal{P}_i$ to be the collection of admissible packings of elements of group G_i alone.

The relaxation has an indicator variable $x_{i,p}$ for every group G_i and admissible assignment $p \in \mathcal{P}_i$. Beside the constraints of AGAP, we further require the total size of the elements in the fractional solution to be at most $M \in [0, m]$. Note that this LP is a relaxation of AGAP only for $M = m$.

$$\begin{array}{lll}
\text{(AGAP-LP)} \max & \sum_{i\in[n]} \sum_{p\in\mathcal{P}_i} \left(x_{i,p} \cdot \sum_{\ell\in G_i} a_{\ell p(\ell)}\right) & \\
\text{s.t.} & \sum_{p\in\mathcal{P}_i} x_{i,p} \leq 1 \quad \forall\, i \in [n] & (1)\\
& \sum_{i\in[n]} \sum_{p\in\mathcal{P}_i \mid \exists\, \ell : p(\ell)=j} x_{i,p} \cdot s_\ell \leq c_j \quad \forall\, j \in [m] & (2)\\
& \sum_{i\in[n]} \sum_{p\in\mathcal{P}_i} \left(x_{i,p} \cdot \sum_{\ell\in G_i} s_\ell\right) \leq M & (3)\\
& x_{i,p} \geq 0 \quad \forall\, i \in [n], p \in \mathcal{P}_i &
\end{array}$$

Constraint (1) requires every group to have at most one assignment. Constraint (2) guarantees that no bin is over-packed. Finally, constraint (3) enforces that the total size of the packed elements does not exceed M.

Lemma 5. *AGAP-LP can be solved in polynomial time.*

The proof of Lemma 5 is based on finding a separation oracle for the dual LP. We give the details in [1].

We present an approximation scheme for the case where the number of bins is a constant. The algorithm uses AGAP-LP, which in this case is of polynomial size and thus can be solved in polynomial time using standard techniques. The rounding procedure we apply draws many ideas from the rounding procedure suggested in [9,8] for the problem of maximizing a submodular function subject to a constant number of knapsack constraints. The idea of the rounding procedure is to guess the most valuable groups of the optimal solution and their corresponding assignment in this solution. Note that this can be done efficiently because the number of bins is constant. None of the remaining groups can be valuable on their own, and therefore, we can safely dismiss all such groups containing a large element. This allows us to show, via concentration bounds, that a randomized rounding satisfies the capacity constraints of all bins with high enough probability (recall that all remaining elements are small).

Theorem 2. *There is a randomized polynomial time approximation scheme for* AGAP *with fixed number of bins.*

3.2 Approximation Algorithm for Unit Size Items

In the special case where all items have unit sizes, we give the best possible approximation ratio.

Theorem 3. AGAP *with unit item sizes admits an* $\frac{e}{e-1}$*-approximation.*

3.3 The All-or-Nothing Group Packing Problem

For AGAP instances where each group G_i has a utility $P_i > 0$ if all of its items are packed, and 0 otherwise, we show that AGAP can be approximated within a small constant $\rho \in (2, 3 + \varepsilon]$, for some $\varepsilon > 0$. Specifically,

Theorem 4. *There is a* $(\frac{2(\gamma+1)}{\gamma} + \varepsilon)$*-approximation for all-or-nothing group packing, where* $\gamma = \lfloor \frac{m}{k_{\max}} \rfloor$.

References

1. Adany, R., Feldman, M., Haramaty, E., Khandekar, R., Schieber, B., Schwartz, R., Shachnai, H., Tamir, T.: All-or-nothing generalized assignment with application to scheduling advertising campaigns (2012) Full version, `http://www.cs.technion.ac.il/~hadas/PUB/AGAP_full.pdf`
2. Calinescu, G., Chekuri, C., Pál, M., Vondrák, J.: Maximizing a submodular set function subject to a matroid constraint. SIAM J. on Computing 40(6), 1740–1766 (2011)
3. Chekuri, C., Khanna, S.: A PTAS for the multiple knapsack problem. SIAM J. on Computing 35(3), 713–728 (2006)
4. Dureau, V.: Addressable advertising on digital television. In: Proceedings of the 2nd European Conference on Interactive Television: Enhancing the Experience, Brighton, UK (March-April 2004)
5. Feige, U., Vondrák, J.: Approximation algorithms for allocation problems: Improving the factor of 1-1/e. In: FOCS, pp. 667–676 (2006)
6. Google AdWords, `http://adwords.google.com`
7. Kim, E.M., Wildman, S.S.: A deeper look at the economics of advertiser support for television: the implications of consumption-differentiated viewers and ad addressability. J. of Media Economics 19, 55–79 (2006)
8. Kulik, A., Shachnai, H., Tamir, T.: Maximizing submodular set functions subject to multiple linear constraints. To appear in Mathematics of Operations Research
9. Kulik, A., Shachnai, H., Tamir, T.: Maximizing submodular set functions subject to multiple linear constraints. In: SODA, pp. 545–554 (2009)
10. Kundakcioglu, O.E., Alizamir, S.: Generalized assignment problem. In: Floudas, C.A., Pardalos, P.M. (eds.) Encyclopedia of Optimization, pp. 1153–1162. Springer (2009)
11. Nemhauser, G., Wolsey, L., Fisher, M.: An analysis of the approximations for maximizing submodular set functions. Math. Programming 14, 265–294 (1978)
12. Nielsen Media Research. Advertising fact sheet. blog.nielsen.com (September 2010)
13. Nielsen Media Research. The cross-platform report, quarter 1, 2012 – US. blog.nielsen.com (May 2012)
14. Nielsen Media Research. Nielsen's quarterly global adview pulse report. blog.nielsen.com (April 2012)
15. Nutov, Z., Beniaminy, I., Yuster, R.: A (1-1/e)-approximation algorithm for the generalized assignment problem. Operations Research Letters 34(3), 283–288 (2006)
16. Park, J., Lim, B., Lee, Y.: A Lagrangian dual-based branch-and-bound algorithm for the generalized multi-assignment problem. Management Science 44, 271–282 (1998)
17. Pramataris, K., Papakyriakopoulos, D., Lekakos, G., Mulonopoulos, N.: Personalized Interactive TV Advertising: The iMEDIA Business Model. Electronic Markets 11, 1–9 (2001)
18. Shmoys, D., Tardos, É.: An approximation algorithm for the generalized assignment problem. Mathematical Programming 62(1), 461–474 (1993)
19. SintecMedia - On Air, `http://www.sintecmedia.com/OnAir.html`
20. The Interactive Advertising Bureau (IAB), `http://iab.net`
21. Young, C.: Why TV spot length matters. Admap (497), 45–48 (2008)

Constant Integrality Gap LP Formulations of Unsplittable Flow on a Path*

Aris Anagnostopoulos[1], Fabrizio Grandoni[2], Stefano Leonardi[1], and Andreas Wiese[3]

[1] Sapienza University of Rome, Italy
{aris,leon}@dis.uniroma1.it
[2] University of Lugano, Switzerland
fabrizio@idsia.ch
[3] Max-Planck-Institut für Informatik, Germany
awiese@mpi-inf.mpg.de

Abstract. The *Unsplittable Flow Problem on a Path* (UFPP) is a core problem in many important settings such as network flows, bandwidth allocation, resource constraint scheduling, and interval packing. We are given a path with capacities on the edges and a set of tasks, each task having a demand, a profit, a source and a destination vertex on the path. The goal is to compute a subset of tasks of maximum profit that does not violate the edge capacities.

In practical applications generic approaches such as integer programming (IP) methods are desirable. Unfortunately, no IP-formulation is known for the problem whose LP-relaxation has an integrality gap that is provably constant. For the unweighted case, we show that adding a few constraints to the standard LP of the problem is sufficient to make the integrality gap drop from $\Omega(n)$ to $O(1)$. This positively answers an open question in [Chekuri et al., APPROX 2009].

For the general (weighted) case, we present an extended formulation with integrality gap bounded by $7 + \varepsilon$. This matches the best known approximation factor for the problem [Bonsma et al., FOCS 2011]. This result exploits crucially a technique for embedding dynamic programs into linear programs. We believe that this method could be useful to strengthen LP-formulations for other problems as well and might eventually speed up computations due to stronger problem formulations.

1 Introduction

In the *Unsplittable Flow Problem on a Path* (UFPP) we are given a set of n tasks $\mathcal{T}$ and a path $G = (V, E)$ on m edges. For each edge e denote by u_e its capacity. Each task $T_i \in \mathcal{T}$ is specified by a start vertex $s_i \in V$, a destination

* Partially supported by EU FP7 Project N. 255403 SNAPS, by the ERC Starting Grant NEWNET 279352, by the ERC Starting Grant PAAl 259515, and by a fellowship within the Postdoc-Programme of the German Academic Exchange Service (DAAD).

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 25–36, 2013.

vertex $t_i \in V$, a demand d_i and a weight (or profit) w_i. For each edge $e \in E$ denote by $\mathcal{T}_e$ all tasks T_i such that the (unique) path from s_i to t_i uses e. Also, we abuse slightly notation and we denote by T_i the set of edges in the path from s_i to t_i. For each task T_i we define its *bottleneck capacity* $b_i := \min\{u_e : e \in T_i\}$. For a value $\delta \in (0, 1)$ we say that a task T_i is δ-large if $d_i > \delta \cdot b_i$ and δ-*small* otherwise. The goal is to select a subset of the tasks $\mathcal{T}' \subseteq \mathcal{T}$ with maximum total weight $w(\mathcal{T}') := \sum_{T_i \in \mathcal{T}'} w_i$ such that $\sum_{T_i \in \mathcal{T}_e \cap \mathcal{T}'} d_i \leq u_e$ for all edges e. In the *unweighted* case, all weights are 1.

This problem occurs in various settings and important applications. As the name suggests, it is a special case of multi-commodity demand flow, with one task associated to each commodity. This problem clearly generalizes well known problems such as knapsack and maximum independent set in interval graphs. It can be used to model the availability over time of a resource of varying capacity, with each task demanding a specific amount of the resource within a fixed time interval. Despite their fundamental nature, the combinatorial structure and the polynomial-time approximability of this problem are not yet well understood. UFPP is strongly NP-hard [5,12] and the best known approximation results are a quasi-PTAS [1] and a polynomial time $(7 + \epsilon)$-approximation algorithm [5].

When solving optimization problems in practice, a common method is to formulate the problem as an integer linear program (ILP) and use an Integer Programming (IP) solver such as CPLEX or Gurobi. However, for many problems there are several possible ILP formulations which perform very differently in practice. One desired property of a good ILP formulation is that the resulting LP relaxation has a small integrality gap. This is helpful since in branch-and-cut algorithms LP relaxations are used to derive good lower bounds, which allow one to neglect certain subtrees and thus speed up the computation. Most of previous LP based approaches for UFPP refer to the following natural LP formulation:

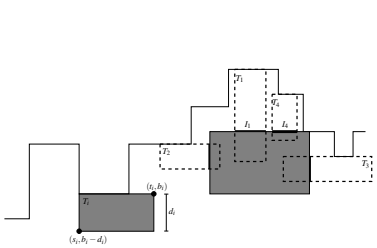
$$\mathrm{LP}_{\mathrm{UFPP}} = \{\max \sum_{i=1}^{n} w_i \cdot x_i \ : \ \sum_{T_i \in \mathcal{T}_e} d_i \cdot x_i \leq u_e, \forall e \in E; 0 \leq x_i \leq 1, i = 1, \ldots, n.\}$$

Unfortunately, $\mathrm{LP}_{\mathrm{UFPP}}$ suffers from an integrality gap $\Omega(n)$ [7]. Chekuri et al. [10] presented an LP formulation with integrality gap of at most $O(\log^2 n)$ obtained by adding an exponential number of constraints to the above LP, which can be approximately separated in polynomial time. (Recently they showed how to obtain a polynomial-size formulation and improved the integrality gap to $O(\log n)$ [9].)

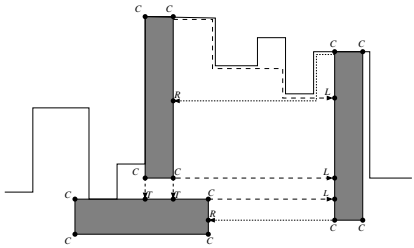
1.1 Our Contribution

In this paper we address the open problem of designing LP relaxations for UFPP with small (namely constant) integrality gap. Our main contributions are as follows:

Unweighted UFPP. We present the first LP relaxation for unweighted UFPP with provably constant integrality gap (see Section 2). Even though the canonical



(a) The shaded and dashed rectangles are in $\mathcal{T}_k$ and $\mathcal{T} \setminus \mathcal{T}_{\text{points}}$, respectively. Top ($T_1$ and T_4), left (T_2), and right (T_3) intersecting rectangles.



(b) Set of points Q. For each rectangle in $\mathcal{T}_k$ there are the 4 corner points (C), (at most) 2 points downwards (T), 2 points rightwards (L), and 2 points leftwards (R).

Fig. 1. Examples of some of the notions

LP-relaxation LP_{UFPP} has a very large integrality gap of $\Omega(n)$, we show that by adding only $O(n^2)$ constraints it drops to $O(1)$, independently of the input size. We show that up to constant factors our integrality gap is bounded by the worst-case integrality gap of the canonical LP for the *Maximum Independent Set of Rectangles* problem (MISR) for instances that stem from 1/2-large tasks in the sense described in [5]. Intuitively, we construct a rectangle with base along the subpath of T_i and height equal to its demand d_i, and then push it as high as possible while remaining below the curve induced by the capacities (see Figure 1(a)).

Bounding the integrality gap of the canonical MISR formulation has been a challenging open problem for a long time. We show that for unweighted instances stemming from UFPP the worst-case integrality gap is $O(1)$. Even more, we provide a purely combinatorial algorithm whose profit is by at most a constant factor smaller than the optimal LP value on those rectangles. Given that the general case of that problem is hard to tackle (no constant factor approximation algorithms are known, whereas the best known lower bound is just NP-hardness) we hope that our result helps understanding this important problem better.

The authors of [10] consider a relaxation of UFPP with an exponential number of additional constraints, and show that it has an $O(\log^2 n)$ integrality gap. Our reasoning implies that in the unweighted case already a polynomial size subset of those constraints yields a formulation with $O(1)$-integrality gap, thus answering an open question posed in [10].

Weighted UFPP. In [16] Martin et al. show a generic method to formulate dynamic programs as linear programs. Roughly speaking, they show that for any (well-behaved) DP one can construct a linear program whose extreme points correspond to the possible outputs of the DP, given suitable weights to the items (jobs). in the input. In the course of our research, and unaware of the result in [16], we developed a slightly different approach for embedding a DP into an LP. It turns out that our approach is somewhat simpler and marginally more general (their approach requires $\alpha_i^C = 1$—see Section 3, whereas a simple

extension gives a pseudopolynomial number of variables), so we include it for completeness.

We combine this DP-embedding theorem with dynamic programs [5] for subcases of UFPP for which the canonical LP has a large integrality gap. By embedding the DP for 1/2-large tasks from [5] we obtain an extended LP relaxation for weighted UFPP with a constant integrality gap (see Section 4). No relaxation with a constant integrality gap was known before. By embedding additionally the DPs for the tasks that are δ-large and 1/2-small, we obtain a formulation whose integrality gap is bounded by $7+\epsilon$, together with a matching polynomial-time rounding procedure. This improves slightly the result in [5], where the same approximation factor is proved w.r.t. the integral optimal profit only.

To the best of our knowledge, this is the first time that the structural insight of [16] is used to strengthen a linear programming formulation, especially to this magnitude (from $\Omega(n)$ to $7+\epsilon$). We believe that this technique could be useful for improving the IP-formulations of other problems as well. Eventually, this might lead to better running times of IP-solvers in practice due to stronger formulations.

1.2 Preliminaries and Related Work

UFPP is weakly NP-hard for the special case of a single edge since then it is equivalent to the knapsack problem. It admits a PTAS for constant number of edges since it reduces to multi-dimensional knapsack [14]. For an arbitrary number of edges the problem is strongly NP-hard [5,12], thus excluding an FPTAS if P $\neq$ NP. In terms of approximation algorithms, the first nontrivial result for UFPP was given by Bansal et al. [2] who gave an $O(\log n)$ approximation algorithm. In a previous paper, Bansal et al. [1] gave a QPTAS for the problem, which requires a quasi-polynomial bound on the edge-capacities and demands. Motivated by the work of [2], Chekuri et al. [10] presented an LP formulation with integrality gap $O(\log^2 n)$, obtained by adding a super-polynomial number of constraints to the natural LP formulation given above. The separation routine given in [10] loses an $O(1)$ factor. The algorithms for UFPP usually distinguish between small and large tasks. In a recent result, Bonsma et al. [5] gave a first $O(1)$ approximation $(7+\epsilon)$ for general UFPP, by designing a dynamic-programming algorithm for MISR and using the solution for UFPP.

A well-studied special case of UFPP is given by the *no-bottleneck assumption* (NBA), which requires that $\max_i\{d_i\} \leq \min_e\{u_e\}$. For UFPP-NBA, dynamic programming exploits the fact that on each edge any solution can have at most $2\lfloor 1/\delta^2 \rfloor$ tasks that are δ-large [7]. (Unfortunately, this property does not hold in the general case). Together with an LP rounding procedure for the remaining tasks this yields the best known approximation algorithm for UFPP-NBA, having a ratio of $2+\epsilon$ [11]. Previously, a similar approximation result was obtained, after a sequence of improvements, for the special case of the *resource-allocation problem* (RAP), which is given by the constraint that all the edges have equal capacity [3,6,13]. The natural LP formulation of UFPP has an $O(1)$ integrality gap for UFPP-NBA [7,11]. The integrality gap is however not less than 2.5 [11],

that is, larger than the best known approximation ratio. In [7] it is left open to find an LP relaxation with constant integrality gap for general UFPP on large tasks even for the unweighted case where all tasks have equal profit.

As we mentioned, the authors of [5] established a connection between UFPP and MISR. For the latter problem, the best known approximation ratio is $O(\log \log n)$ [8] (and $O(\log n)$ in the weighted case, see for example, [4,15]). It is still open to find an $O(1)$ approximation algorithm for MISR, even only for the unweighted case. In particular, the exact integrality gap of the standard LP formulation for the problem is not known. The best known lower and upper bounds for it (in the unweighted case) are 3/2 and $O(\log \log n)$ [8], respectively.

2 Constant Integrality Gap for Unweighted UFPP

In this section we show how to strengthen the canonical linear program $\mathrm{LP}_{\mathrm{UFPP}}$ for unsplittable flow to make its integrality gap drop from $\Omega(n)$ to $O(1)$. Let us focus for a moment on 1/2-large tasks $\mathcal{T}_{\mathrm{large}}$ only (which we next call *large* for brevity). For those, in [5] the following geometrical interpretation was introduced: for each task T_i draw a rectangle T_i specified by the upper left point (s_i, b_i) and the lower right point (t_i, ℓ_i) where $\ell_i := b_i - d_i$, where we interpret the vertices of the path as integers. In [5] the authors show that any feasible integral solution $\mathcal{T}'$ consisting of only large tasks has the property that any point in the plane can be covered by (i.e., is contained in the interior of) at most four rectangles in $\mathcal{T}'$ [5, Lemma 13]. Due to the geometry of the rectangles, to check the above property it is sufficient to consider a proper subset P of only $O(n^2)$ points.

Our main idea is to add the corresponding set of feasible constraints to the standard LP for unweighted UFPP, therefore obtaining the following refined LP:

$$\mathrm{LP}^{+}_{\mathrm{UFPP}} := \{\max \sum_{T_i \in T} x_i \text{ s.t. } \sum_{T_i \in T_e} x_i \cdot d_i \leq u_e, \forall e \in E; \sum_{T_i \in \mathcal{T}_{\mathrm{large}}:\, p \in T_i} x_i \leq 4, \forall p \in P;\ \ x_i \geq 0, \forall T_i \in \mathcal{T}\}$$

By the above reasoning any integral solution satisfies the added constraints.

We recall that for any $\delta \in (0,1)$ (hence, in particular, for $\delta = 1/2$), the canonical LP has already a constant integrality gap for instances with only $(1-\delta)$-small tasks. Therefore, it is sufficient to bound the integrality gap for large tasks only. Observe that, given a feasible solution x to $\mathrm{LP}^{+}_{\mathrm{UFPP}}$, the vector $y_i := x_i/4$, $T_i \in \mathcal{T}_{\mathrm{large}}$, yields a feasible solution for the following linear program:

$$\mathrm{LP}_{\mathrm{MISR}} := \{\max \sum_{T_i \in \mathcal{T}_{\mathrm{large}}} y_i \text{ s.t. } \sum_{T_i \in \mathcal{T}_{\mathrm{large}}:\, p \in T_i} y_i \leq 1, \forall p \in P;\ y_i \geq 0, \forall T_i \in \mathcal{T}_{\mathrm{large}}\}$$

The above LP is the canonical LP for MISR on rectangles $\mathcal{T}_{\mathrm{large}}$. By the definition of the heights of the rectangles, it is easy to see that any independent set of rectangles $\mathcal{T}' \subseteq \mathcal{T}_{\mathrm{large}}$ induces a feasible UFPP-solution. This yields the following lemma.

Lemma 1. *Assume that* $\mathrm{LP}_{\mathrm{MISR}}$ *has an integrality gap of* α *for instances that stem from* UFPP *instances and that* $\mathrm{LP}_{\mathrm{UFPP}}$ *has an integrality gap of* β *for instances with only* 1/2*-small tasks. Then* $\mathrm{LP}^{+}_{\mathrm{UFPP}}$ *has an integrality gap of at most* $4\alpha + \beta$*.*

2.1 A Combinatorial Algorithm for Large Tasks

It remains to show that $\mathrm{LP}_{\mathrm{MISR}}$ has an integrality gap of $\alpha = O(1)$. To this aim, we describe a combinatorial algorithm that computes an independent set of rectangles whose cardinality is at most by a constant factor smaller than the value of the optimal LP solution. Suppose we are given a set of rectangles $\mathcal{T}$ stemming from (the large tasks of) a UFPP instance. Our algorithm runs in phases where in each phase k we either compute a maximal set $\mathcal{T}_{k+1}$ based on the set $\mathcal{T}_k$ computed in the previous iteration such that $|\mathcal{T}_{k+1}| > |\mathcal{T}_k|$ or assert that $|\mathcal{T}_k|$ is large in comparison with the LP optimum. Because the optimal solution can contain at most n rectangles, there can be at most n phases. We start with any maximal independent set of rectangles $\mathcal{T}_0 \subseteq \mathcal{T}$, which can be trivially computed (say, using a greedy algorithm). Now suppose that we have computed a set $\mathcal{T}_k$. For each rectangle $T_i \in \mathcal{T}_k$ we identify at most ten points Q_i in the plane (see Figure 1(b)). The first four points in Q_i are the four corners of T_i (points C in Figure 1(b)). The other six points are obtained as follows:

- Take the bottom-left (the bottom-right) corner and move down until you hit the boundary of another rectangle in $\mathcal{T}_k$ (points T in Figure 1(b)), if any.
- The points (points L in Figure 1(b)) which are defined by the following process: Start from the top-right (also bottom-right) corner; call this point (x, y). Iteratively execute the following step until you hit the boundary of another rectangle in $\mathcal{T}_k$, if any: If $y \leq u_{\{x,x+1\}}$ then set $(x, y) \leftarrow (x + 1, y)$. Otherwise, set $(x, y) \leftarrow (x, y - 1)$.
- Similarly, going leftwards starting from the top-left and bottom-left points (points R in Figure 1(b)).

We denote by $\mathcal{T}_{\mathrm{points}} \subseteq \mathcal{T}$ all rectangles in the instance that overlap some point in $Q := \bigcup_{T_i \in \mathcal{T}_k} Q_i$. We show later in Lemma 4 that those tasks have bounded LP-weight. Consider the remaining rectangles $\mathcal{T} \setminus \mathcal{T}_{\mathrm{points}}$. First observe that because $\mathcal{T}_k$ is maximal, every rectangle in $\mathcal{T} \setminus \mathcal{T}_{\mathrm{points}}$ must intersect a rectangle of $\mathcal{T}_k$. We classify $\mathcal{T} \setminus \mathcal{T}_{\mathrm{points}}$ as *top-intersecting* rectangles $\mathcal{T}_{\mathrm{top}}$, *left-intersecting* rectangles $\mathcal{T}_{\mathrm{left}}$, and *right-intersecting* rectangles $\mathcal{T}_{\mathrm{right}}$. We call a rectangle T_i top intersecting if there exists a rectangle $T_j \in \mathcal{T}_k$ such that $s_j < s_i < t_i < t_j$, and $\ell_i < b_j < b_i$. We call a rectangle T_i left (resp., right) intersecting if there exists a rectangle $T_j \in \mathcal{T}_k$ such that $\ell_j < \ell_i < b_i < b_j$, and $s_i < s_j < t_i$ (resp., $s_i < t_j < t_i$) (see Figure 1(a)). We can then prove the following lemma:

Lemma 2. *All the rectangles in* $\mathcal{T} \setminus \mathcal{T}_{\mathrm{points}}$ *are either top-intersecting, left-intersecting, or right-intersecting.*

In our algorithm we now take each set $\mathcal{T}_{\mathrm{top}}$, $\mathcal{T}_{\mathrm{left}}$, and $\mathcal{T}_{\mathrm{right}}$ separately and compute an optimal solution for it. The crucial observation is now that this problem

is equivalent to the maximum independent set problem in interval graphs. To this end, construct a graph $G_{\text{top}} = (V_{\text{top}}, E_{\text{top}})$ where V_{top} consists of one vertex v_j for each rectangle $T_j \in \mathcal{T}_{\text{top}}$ and an edge $\{v_j, v'_j\}$ exists if and only if T_j and $T_{j'}$ overlap. Define the graphs G_{left} and G_{right} similarly.

Lemma 3. *The graphs G_{top}, G_{left}, and G_{right} are interval graphs.*

The proof is very technical, so we only provide an intuition for the top-intersecting rectangles—the argument for the other two cases is similar albeit somewhat more complicated. It relies highly on the definition of the points Q. Consider two top-intersecting rectangles T_1 and T_4 (see Figure 1(a)). To each of them corresponds an interval I_1 and I_4, defined by the intersection of the rectangle with the rectangle to which they intersect. Note that I_1 and I_4 intersect if and only if T_1 and T_4 overlap. To compute a maximum independent set, it suffices to preorder the rectangles according to their values t_i and at iteration k, for each interval, select rectangles greedily by increasing values of t_i [17].

Denote by OPT_{top}, OPT_{left}, and $\text{OPT}_{\text{right}}$ the optimal independent sets for G_{top}, G_{left}, and G_{right}, respectively. Now there are two cases. If $|\text{OPT}_{\text{top}}| \leq |\mathcal{T}_k|$ and $|\text{OPT}_{\text{left}}| \leq |\mathcal{T}_k|$ and $|\text{OPT}_{\text{right}}| \leq |\mathcal{T}_k|$, then we output $\mathcal{T}_k$ and halt. Otherwise we define $\mathcal{T}_{k+1}$ to be the set of maximum cardinality among OPT_{top}, OPT_{left}, and $\text{OPT}_{\text{right}}$, and we proceed with the next iteration.

Suppose now that our algorithm runs for k iterations and finally outputs the set $\mathcal{T}_k$. We want to bound its cardinality in comparison with the optimal fractional solution to LP_{MISR}. By Lemma 2 the union of the sets $\mathcal{T}_{\text{points}}, \mathcal{T}_{\text{top}}, \mathcal{T}_{\text{left}}, \mathcal{T}_{\text{right}}$ equals $\mathcal{T}$. We bound the LP profit for each of these sets separately. Note that the next lemma is the only part of our reasoning where we use that the rectangles are unweighted.

Lemma 4. *In any feasible solution y to LP_{MISR} the profit of the rectangles in $\mathcal{T}_{\text{points}}$ is bounded by $\sum_{p \in Q} \sum_{T_i:\, p \in T_i} y_i \leq 10 \cdot |\mathcal{T}_k|$.*

Proof (sketch). Let $Q_{\text{TL}} \subseteq Q$ be the set of top-left corners of all the rectangles in $\mathcal{T}_k$. Notice that $|Q_{\text{TL}}| = |\mathcal{T}_k|$. We have $\sum_{p \in Q_{\text{TL}}} \sum_{T_i:\, p \in T_i} y_i \leq \sum_{p \in Q_{\text{TL}}} 1 \leq |\mathcal{T}_k|$, where the first inequality follows from the constraints of LP_{MISR}. By performing the same approach for all the 10 families of points that constitute the set Q, we obtain the lemma.

Lemma 5. *For any feasible solution y to LP_{MISR} and any set $\mathcal{T}' \in \{\mathcal{T}_{\text{top}}, \mathcal{T}_{\text{left}}, \mathcal{T}_{\text{right}}\}$ it holds that $\sum_{T_i:\, \mathcal{T}'} y_i \leq OPT(\mathcal{T}')$, where $OPT(\mathcal{T}')$ stands for OPT_{top}, OPT_{left}, or $\text{OPT}_{\text{right}}$.*

Proof. By Lemma 3 the graphs G_{top}, G_{left}, and G_{right} are interval graphs and hence in particular perfect graphs. Therefore, for the maximum independent set problem the following LP formulation is exact: introduce a variable $x_v \geq 0$ for every vertex $v \in V$ and the clique inequality $\sum_{v \in C} x_v \leq 1$ for all maximal cliques $C \subseteq V$ (see, for example, [17]). In MISR, for every maximal clique $C \subseteq \mathcal{T}$ we can find a point in the plane that is covered by all the rectangles in C. Hence, LP_{MISR} contains a clique inequality for each maximal clique in the graphs G_{top},

G_{left}, and G_{right}. Thus, LP_{MISR} cannot gain more profit than the respective optimal integral solution for these subproblems.

Theorem 1. *Consider the set of rectangles $\mathcal{T}$ in the plane that stem from a* UFPP*-instance. There is a polynomial-time algorithm that computes a set $\mathcal{T}' \subseteq \mathcal{T}$ such that $\sum_{T_i \in \mathcal{T}} y_i \leq 13 \cdot |\mathcal{T}'|$ for any feasible solution y of* LP_{MISR}.

Proof. By Lemma 4 and 5 we have $\sum_{T_i \in \mathcal{T}} y_i = \sum_{T_i \in \mathcal{T}_{\text{points}}} y_i + \sum_{T_i \in \mathcal{T}_{\text{top}}} y_i + \sum_{T_i \in \mathcal{T}_{\text{left}}} y_i + \sum_{T_i \in \mathcal{T}_{\text{right}}} y_i \leq 10 \cdot |\mathcal{T}_k| + |\text{OPT}_{\text{top}}| + |\text{OPT}_{\text{left}}| + |\text{OPT}_{\text{right}}| \leq 13 \cdot |\mathcal{T}_k|$.

Combining this theorem with Lemma 1, we obtain the following theorem.

Theorem 2. *The integrality gap of* $\text{LP}^{+}_{\text{UFPP}}$ *for unweighted* UFPP *is constant.*

Finally, observe that if we define a task to be in $\mathcal{T}_{\text{large}}$ if it is 3/4-large, then the resulting LP still has constant integrality gap by the same reasoning. In particular, then our added constraints would be a proper (polynomial size) subset of the (exponentially many) *rank constraints* introduced in [10]: for each edge e and for each subset $\mathcal{T}'$ of large tasks using e, there is a rank constraint bounding the maximum number of tasks in $\mathcal{T}'$ which can be in a feasible solution. In [10] it was left as an open question whether the integrality gap of the standard LP together with these constraints is $O(1)$, and an upper bound of $O(\log^2 n)$ was shown. Hence, we answered this question affirmatively for the unweighted case.

3 Embedding Dynamic Programs into Linear Programs

Let us start with a formal definition of a *standard* dynamic program $\mathcal{DP}$, which seems to capture most natural dynamic programs. For the sake of simplicity, let us focus on maximization problems, the case of minimization problems being symmetric. Consider some instance of the problem. This instance induces a polynomial-size set of possible *states* $\mathcal{S}$. The dynamic program fills in a table $t(\cdot)$, indexed by the states. There is a collection $\mathcal{S}_{\text{base}} \subseteq \mathcal{S}$ of *base states*, whose profits can be computed with some trivial procedure (for example, they have profit zero). The remaining table entries $t(S)$, $S \notin \mathcal{S}_{\text{base}}$, are filled in as follows. There is a set $\mathcal{C}_S$ of possible choices associated to S. We let $\mathcal{C} := \cup_{S \in \mathcal{S}, C \in \mathcal{C}_S} \{C\}$ be the set of all the possible choices. For notational convenience, we assume that choices of distinct states are distinct, and we let S^C denote the (only) state associated to C. Each choice $C \in \mathcal{C}_S$ is characterized by a profit w^C and by a collection of distinct states $S^C_1, \ldots, S^C_{k^C}$. If $\mathcal{C}_S = \emptyset$, we set $t(S) = -\infty$. Otherwise $t(S)$ is computed by exploiting the following type of recurrence, for proper coefficients $\alpha^C_i > 0$ (which can be assumed to be integral w.l.o.g.)[1]:

$$t(S) := \max_{C \in \mathcal{C}_S} \{w^C + \sum_{i=1}^{k^C} \alpha^C_i \cdot t(S^C_i)\}.$$

[1] Often in practice all α^C_i are 1 and k^C is a small number like 1 or 2.

Each state S_i^C must be a predecessor of S according to a proper partial order defined on the states (where the minimal states are the base ones). This partial order guarantees that $t(\cdot)$ can be filled in a bottom up fashion (without cycling). At the end of the process $t(S_{\text{start}})$ contains the value of the desired solution, for a proper special state S_{start}. By keeping track of the choices C which give the maximum in each recurrence, one also obtains the corresponding solution. For notational convenience we next assume that $t(S) = 0$ for every $S \in \mathcal{S}_{\text{base}}$. This can be enforced by introducing a dummy choice node C' with weight $t(S)$, and an associated dummy child state node S' with $t(S') = 0$. This way the profit can be expressed as the sum of the weights of the selected choices. We will use $\mathcal{I}$ to denote the input instance, excluding the part which is needed to define the weights w^C in the recurrences. In particular, $\mathcal{I}$ defines the states, the feasible choices for each state, the states associated to each choice, and the corresponding coefficients.

Next we describe an LP whose basic solutions describe the execution of $\mathcal{DP}$ on a given $\mathcal{I}$ for all the possible weights w^C. In particular, the weights will not appear in the set of constraints. Let us define a digraph $G = (V, E)$, with *state nodes* $\mathcal{S}$ and *choice nodes* $\mathcal{C}$. For every $C \in \mathcal{C}$, we add edges (S^C, C) and (C, S_i^C) for all $1 \leq i \leq k^C$. Observe that G is a DAG (i.e., there are no directed cycles) due to the partial order on the states. W.lo.g. we can assume that S_{start} has no ancestors. We let $\delta^{in}(v)$ and $\delta^{out}(v)$ denote the set of edges ending and starting at v, respectively. We associate a variable y_e to each edge e. The value of y_e in a fractional solution will be interpreted as a directed flow crossing e. For each state node $S \in S_{int} := \mathcal{S} - (\mathcal{S}_{\text{base}} \cup \{S_{\text{start}}\})$, we introduce a *flow conservation* constraint: $\sum_{\delta^{in}(S)} y_e = \sum_{\delta^{out}(S)} y_e$. We remark that it might be $\delta^{out}(S) = \emptyset$, in which case we assume that the corresponding sum has value zero. Recall that, in this case, $t(S) = -\infty$. We also force S_{start} to be the source of one unit of flow: $\sum_{\delta^{out}(S_{\text{start}})} y_e = 1$. This flow will end in nodes of $\mathcal{S}_{\text{base}}$. For each choice node $C \in \mathcal{C}$, we add a *flow duplication* constraint which guarantees that the flow entering C from its only ingoing edge $e = (S^C, C)$ is duplicated on all its outgoing edges according to the integral coefficients α_i^C: $x_{(C,S_i^C)} = \alpha_i^C \cdot x_{(S^C,C)}$ for all $1 \leq i \leq k^C$. We remark that, due to flow duplication, the flow entering a given node might be larger than 1. For a state node S this means that $t(S)$ contributes multiple times to the objective function. Altogether, the LP is defined as follows:

$$LP_{\mathcal{DP},\mathcal{I}} = \{\max \sum_{C \in \mathcal{C}} w^C \cdot y_{(S^C,C)} \text{ s.t. } \sum_{\delta^{in}(S)} y_e = \sum_{\delta^{out}(S)} y_e, \forall S \in \mathcal{S}_{int};$$

$$\sum_{\delta^{out}(S_{\text{start}})} y_e = 1;$$

$$y_{(C,S_i^C)} = \alpha_i^C \cdot y_{(S^C,C)}, \forall C \in \mathcal{C}, 1 \leq i \leq k^C;$$

$$y_e \geq 0, \forall e \in E\}$$

Let $CLP_{\mathcal{DP},\mathcal{I}} = CLP_{\mathcal{DP},\mathcal{I}}(y)$ be the set of constraints of $LP_{\mathcal{DP},\mathcal{I}}$. Let also $CH_{\mathcal{DP},\mathcal{I}}$ denote the collection of set of choices made by $\mathcal{DP}$ on any feasible input $\mathcal{I}$ for any possible choice of the weights w^C.

Theorem 3. (DP-embedding) *The vertices of $CLP_{\mathcal{DP},\mathcal{I}}$ are integral and in one to one correspondence with $CH_{\mathcal{DP},\mathcal{I}}$. Furthermore, $t(S_{start})$ is $-\infty$ iff $CLP_{\mathcal{DP},\mathcal{I}}$ is infeasible, and in all the other cases $t(S_{start})$ equals the optimal value of $LP_{\mathcal{DP},\mathcal{I}}$ (for a given choice of the weights).*

4 Constant Integrality Gap for Weighted UFPP

In this section we present an extended LP formulations for UFPP with constant integrality gap. For reasons of space, we present here a weaker LP with $O(1)$ integrality gap. For the claimed LP with integrality gap $7+\varepsilon$ we refer to the full version of this paper.

Recall that T_i denotes either a task or the corresponding rectangle. For brevity we call *large* (resp., *small*) the tasks that are 1/2-large (resp., 1/2-small), and denote the corresponding set by $\mathcal{T}_{\text{large}}$ (resp., $\mathcal{T}_{\text{small}}$). W.l.o.g., we can assume that $\mathcal{T}_{\text{large}}$ is given by the first n' tasks. We will crucially need the following two lemmas.

Lemma 6 ([5]). *Let $\mathcal{T}' \subseteq \mathcal{T}_{\text{large}}$ be a feasible solution to* UFPP. *There exists a partition of $\mathcal{T}'$ into 4 (disjoint) subsets, where each subset is an independent set of rectangles.*

Lemma 7 ([5]). *There is a dynamic program $\mathcal{DP}'$ which computes a maximum weight independent set of rectangles in $\mathcal{T}_{\text{large}}$.*

Maximum weight independent set of rectangles is a *subset problem*, where we are given a collection of n items $\{1,\ldots,n\}$ where item i has profit w_i, and we need to select a maximum profit subset of items satisfying given constraints. A solution to these problems can be defined as a binary vector $z=(z_1,\ldots,z_n)\in\{0,1\}^n$, where $z_i=1$ iff item i (task T_i in our case) is selected. We remark that each choice of the dynamic program $\mathcal{DP}'$ from the previous lemma corresponds to selecting one or more items, and the structure of $\mathcal{DP}'$ guarantees that no item is selected more than once. Let $\mathcal{C}_i$ denote the choices of $\mathcal{DP}'$ that select item i. Consider the following LP:

$$EXT_{\mathcal{DP}',\mathcal{I}} := \{\max \sum_{i=1}^{n} w_i \cdot z_i \text{ s.t. } CLP_{\mathcal{DP}',\mathcal{I}}(y); z_i = \sum_{C\in\mathcal{C}_i} y_{(S^C,C)}, i=1,\ldots,n\}.$$

By Theorem 3, if we project the basic solutions of $EXT_{\mathcal{DP}',\mathcal{I}}$ on variables z_i we obtain the set of solutions that $\mathcal{DP}'$ might compute on instance $\mathcal{I}$ for some choice of the weights w_i. In other terms, $EXT_{\mathcal{DP}',\mathcal{I}}$ is an integral extended LP formulation of the problem solved by $\mathcal{DP}'$ on instance $\mathcal{I}'$ for given weights. Also in this case we let $CEXT_{\mathcal{DP}',\mathcal{I}} = CEXT_{\mathcal{DP}',\mathcal{I}}(z)$ denote the set of constraints of $EXT_{\mathcal{DP}',\mathcal{I}}$.

We remark that there exists a choice of (possibly negative[2]) weights of the tasks that forces $\mathcal{DP}'$ to compute any given feasible solution $\mathcal{T}'$: we need this property for technical reasons.

[2] Non-negative weights are sufficient, provided that ties in the computation of the maxima are broken in a proper way.

We consider the following LP formulation for UFPP:

$$\begin{aligned}\mathrm{LP}^{+}_{\mathrm{UFPP}} := \{\max \sum_{i=1}^{n} w_i \cdot x_i \text{ s.t. } & CEXT_{\mathcal{DP}',\mathcal{T}_{\mathrm{large}}}(z^j), j = 1,2,3,4;\\ & x_i \leq z_i^1 + z_i^2 + z_i^3 + z_i^4, i = 1,\ldots,n';\\ & \sum_{T_i \in \mathcal{T}_e} d_i \cdot x_i \leq u_e, \forall e \in E;\\ & 0 \leq x_i \leq 1, i = 1,\ldots,n\}\end{aligned}$$

Let us argue that every feasible solution $\mathcal{T}'$ induces a feasible integral solution $(\tilde{x}, \tilde{z})$ (of the same profit) to $\mathrm{LP}^{+}_{\mathrm{UFPP}}$. Set $\tilde{x}_i = 1$ if $T_i \in \mathcal{T}'$, and $\tilde{x}_i = 0$ otherwise. Let $\mathcal{T}'_{\mathrm{large}} := \mathcal{T}' \cap \mathcal{T}_{\mathrm{large}}$, and $(\mathcal{T}^1, \mathcal{T}^2, \mathcal{T}^3, \mathcal{T}^4)$ be the partition of $\mathcal{T}'_{\mathrm{large}}$ given by Lemma 6. Fix $\tilde{z}_i^j = 1$ if $T_i \in \mathcal{T}^j$ and $\tilde{z}_i^j = 0$ otherwise. The resulting integral solution trivially satisfies the last three constraints. For the first constraint, we observe that $\mathcal{T}^j$ is a feasible independent set of rectangles: consequently, there is a choice of the weights that forces $\mathcal{DP}'$ to compute that solution. Thus $\tilde{z}^j$ must be a feasible (indeed basic) solution of $EXT_{\mathcal{DP}',\mathcal{T}_{\mathrm{large}}}(z^j)$.

Consider the standard linear program $\mathrm{LP}_{\mathrm{UFPP}}$. Even though $\mathrm{LP}_{\mathrm{UFPP}}$ has unbounded integrality gap in general, its integrality gap is bounded when there are only small tasks.

Lemma 8 ([10]). *Let $\delta > 0$. For instances of* UFPP *with only $(1-\delta)$-small tasks, the integrality gap of $\mathrm{LP}_{\mathrm{UFPP}}$ is bounded by $O(\log(1/\delta)/\delta^3)$.*

We are ready to bound the integrality gap of $\mathrm{LP}^{+}_{\mathrm{UFPP}}$.

Theorem 4. *The integrality gap of $\mathrm{LP}^{+}_{\mathrm{UFPP}}$ is in $O(1)$.*

We can strengthen the linear program presented here even further such that its integrality gap is bounded by $7+\epsilon$, matching the ratio of the best known approximation algorithm for UFPP [5]. The latter algorithm works as follows: for the 1/2-large tasks it uses the DP described in the previous section. For δ-small tasks (for a sufficiently small value of δ depending on ϵ) it uses LP-based methods together with a framework to combine solutions for suitable subproblems. (In fact, already in [11, Corollary 3.4] it was shown that in that setting $\mathrm{LP}_{\mathrm{UFPP}}$ has an integrality gap of only $1+\epsilon$, if δ is sufficiently small.) For the remaining tasks, that is, tasks that are δ-large but 1/2-small, the algorithm in [5] employs $O(n)$ dynamic programs for suitably chosen subproblems. We can embed these dynamic programs into $\mathrm{LP}^{+}_{\mathrm{UFPP}}$. Using similar ideas as for the $(7+\epsilon)$-approximation algorithm in [5] one can show that the resulting LP has an integrality gap of at most $7+\epsilon$. We leave the details to the full version of this paper.

Theorem 5. *For every $\epsilon > 0$ there is a linear programming formulation of UFPP with an integrality gap of at most $7+\epsilon$ whose complexity is bounded by a polynomial in the input.*

References

1. Bansal, N., Chakrabarti, A., Epstein, A., Schieber, B.: A quasi-PTAS for unsplittable flow on line graphs. In: STOC, pp. 721–729 (2006)
2. Bansal, N., Friggstad, Z., Khandekar, R., Salavatipour, R.: A logarithmic approximation for unsplittable flow on line graphs. In: SODA, pp. 702–709 (2009)
3. Bar-Noy, A., Bar-Yehuda, R., Freund, A., Naor, J., Schieber, B.: A unified approach to approximating resource allocation and scheduling. In: STOC, pp. 735–744 (2000)
4. Berman, P., DasGupta, B., Muthukrishnan, S., Ramaswami, S.: Improved approximation algorithms for rectangle tiling and packing. In: SODA, pp. 427–436 (2001)
5. Bonsma, P., Schulz, J., Wiese, A.: A constant factor approximation algorithm for unsplittable flow on paths. In: FOCS, pp. 47–56 (2011)
6. Calinescu, G., Chakrabarti, A., Karloff, H., Rabani, Y.: Improved Approximation Algorithms for Resource Allocation. In: Cook, W.J., Schulz, A.S. (eds.) IPCO 2002. LNCS, vol. 2337, pp. 401–414. Springer, Heidelberg (2002)
7. Chakrabarti, A., Chekuri, C., Gupta, A., Kumar, A.: Approximation algorithms for the unsplittable flow problem. Algorithmica, 53–78 (2007)
8. Chalermsook, P., Chuzhoy, J.: Maximum independent set of rectangles. In: SODA, pp. 892–901 (2009)
9. Chekuri, C., Ene, A., Korula, N.: Unsplittable flow in paths and trees and column-restricted packing integer programs, unpublished, `http://web.engr.illinois.edu/~ene1/papers/ufp-full.pdf`
10. Chekuri, C., Ene, A., Korula, N.: Unsplittable Flow in Paths and Trees and Column-Restricted Packing Integer Programs. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX 2009. LNCS, vol. 5687, pp. 42–55. Springer, Heidelberg (2009)
11. Chekuri, C., Mydlarz, M., Shepherd, F.: Multicommodity demand flow in a tree and packing integer programs. ACM Trans. on Algorithms 3 (2007)
12. Chrobak, M., Woeginger, G.J., Makino, K., Xu, H.: Caching Is Hard – Even in the Fault Model. In: de Berg, M., Meyer, U. (eds.) ESA 2010, Part I. LNCS, vol. 6346, pp. 195–206. Springer, Heidelberg (2010)
13. Darmann, A., Pferschy, U., Schauer, J.: Resource allocation with time intervals. Theor. Comp. Sc. 411, 4217–4234 (2010)
14. Frieze, A.M., Clarke, M.R.B.: Approximation algorithms for the m-dimensional $0-1$ knapsack problem: worst-case and probabilistic analyses. European J. Oper. Res. 15, 100–109 (1984)
15. Khanna, S., Muthukrishnan, S., Paterson, M.: On approximating rectangle tiling and packing. In: SODA, pp. 384–393 (1998)
16. Kipp Martin, R., Rardin, R.L., Campbell, B.A.: Polyhedral characterization of discrete dynamic programming. Oper. Res. 38(1), 127–138 (1990)
17. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency. Springer, Berlin (2003)

Intersection Cuts for Mixed Integer Conic Quadratic Sets

Kent Andersen and Anders Nedergaard Jensen

Department of Mathematics, University of Aarhus, Denmark
{kent,jensen}@imf.au.dk

Abstract. Balas introduced intersection cuts for mixed integer linear sets. Intersection cuts are given by closed form formulas and form an important class of cuts for solving mixed integer linear programs. In this paper we introduce an extension of intersection cuts to mixed integer conic quadratic sets. We identify the formula for the conic quadratic intersection cut by formulating a system of polynomial equations with additional variables that are satisfied by points on a certain piece of the boundary defined by the intersection cut. Using a software package from algebraic geometry we then eliminate variables from the system and get a formula for the intersection cut in dimension three. This formula is finally generalized and proved for any dimension. The intersection cut we present generalizes a conic quadratic cut introduced by Modaresi, Kilinc and Vielma.

1 Introduction

In this paper we study a mixed integer set obtained from a single conic quadratic inequality defined from rational data $A \in \mathbb{Q}^{m\times n}$ and $d \in \mathbb{Q}^m$:

$$Q_I := \{x \in \mathbb{R}^n : Ax - d \in L^m \text{ and } x_j \in \mathbb{Z} \text{ for } j \in I\}, \tag{1}$$

where L^m is the m-dimensional Lorentz cone $L^m := \{y \in \mathbb{R}^m : y_m \geq \sqrt{\sum_{j=1}^{m-1} y_j^2}\}$ and I is an index set for the integer constrained variables. The continuous relaxation of Q_I is given by $Q := \{x \in \mathbb{R}^n : Ax - d \in L^m\}$. A mixed integer conic quadratic set of the form Q_I can be obtained from a single constraint of the continuous relaxation of a Mixed Integer Conic Quadratic Optimisation (MICQO) problem. Valid inequalities for Q_I (linear or non-linear) can therefore be used as cuts for solving MICQO problems.

The present paper gives an extension of the *intersection cut* of Balas [4] from Mixed Integer Linear Optimisation (MILO) problems to MICQO problems. Several previous papers have aimed at extending cuts from MILO to MICQO. An extension of the mixed integer rounding cuts of Nemhauser and Wolsey [13] to MICQO was given by Atamtürk and Narayanan [2,3]. Çezik and Iyengar [7] studied the extension of the Chvátal-Gomory procedure from MILO to MICQO. The Lift-and-Project algorithm of Balas et al. [5] developed for MILO was generalized by Stubbs and Mehrotra [15] to MICQO.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 37–48, 2013.

Intersection cuts form a very important class of cutting planes for solving MILO problems [6]. Mixed Integer Rounding (MIR) cuts [13], Mixed Integer Gomory (MIG) cuts [9], Lift-and-Project cuts [5] and Split Cuts [8] are all intersection cuts. Since the intersection cuts we propose for MICQO are also given by a closed form formula and derived using similar principles as for MILO problems, we hope they can be equally useful for solving MICQO problems.

As our study is inspired by the intersection cut introduced by Balas [4] for a mixed integer linear set $P_I := \{x \in P : x_j \in \mathbb{Z} \text{ for } j \in I\}$ with a polyhedral relaxation $P := \{x \in \mathbb{R}^n : Ax - d \in \mathbb{R}^m_+\}$, we first review the derivation of intersection cuts in a linear setting.

Intersection cuts for P_I are derived from a maximal choice B of linearly independent rows $\{a_i\}_{i \in B}$ of the matrix A. A given choice B gives a relaxation:

$$P^B := \{x \in \mathbb{R}^n : a_i^T x - d_i \geq 0 \text{ for } i \in B\} \tag{2}$$

of P obtained by removing constraints *not* indexed by B. An intersection cut is then obtained from P^B and a choice of a *split disjunction*. A split disjunction is a disjunction of the form $\pi^T x \leq \pi_0 \vee \pi^T x \geq \pi_0 + 1$, with $(\pi, \pi_0) \in \mathbb{R}^{n+1}$ chosen so that there are no mixed integer points strictly between the hyperplanes $\pi^T x = \pi_0$ and $\pi^T x = \pi_0 + 1$. The simple geometry of P^B gives that the convex hull $\text{conv}(P_1^B \cup P_B^2)$ of the sets:

$$P_1^B := \{x \in P^B : \pi^T x \leq \pi_0\} \text{ and } P_2^B := \{x \in P^B : \pi^T x \geq \pi_0 + 1\} \tag{3}$$

can be described with at most one additional linear inequality, and such an inequality is called the intersection cut obtained from B and (π, π_0).

Our proposal for an intersection cut for the mixed integer conic quadratic set Q_I is now the following. Again we consider a maximal choice B of linearly independent rows $\{a_i\}_{i \in B}$ of the matrix A. We require $m \in B$ since it is necessary to include the m^{th} row of $Ax - d$ in B in a conic quadratic setting for natural reasons. The choice B leads to the relaxation Q^B of Q:

$$Q^B := \{x \in \mathbb{R}^n : A_B \cdot x - d_B \in L^{|B|}\}, \tag{4}$$

where (A_B, d_B) is obtained from (A, d) by deleting rows *not* indexed by B. Given a choice of split disjunction $\pi^T x \leq \pi_0 \vee \pi^T x \geq \pi_0 + 1$, we will show that the convex hull $\text{conv}(Q_1^B \cup Q_2^B)$ of the sets:

$$Q_1^B := \{x \in Q^B : \pi^T x \leq \pi_0\} \text{ and } Q_2^B := \{x \in Q^B : \pi^T x \geq \pi_0 + 1\} \tag{5}$$

can be described with at most one additional inequality given by a closed form formula, and we call such an inequality a *conic quadratic intersection cut*.

We now present our main result: The inequality description of $\text{conv}(Q_1^B \cup Q_2^B)$. For simplicity assume in the following that there is only one choice of constraint set B, *i.e.*, assume the matrix A has full row rank. Let the sets $Q_1 := \{x \in Q : \pi^T x \leq \pi_0\}$ and $Q_2 := \{x \in Q : \pi^T x \geq \pi_0 + 1\}$ be the points in Q satisfying $\pi^T x \leq \pi_0 \vee \pi^T x \geq \pi_0 + 1$. We now give a characterization of $\text{conv}(Q_1 \cup Q_2)$. There are three cases that we need to consider.

We must first answer the question: When does $\pi^T x \leq \pi_0 \vee \pi^T x \geq \pi_0 + 1$ give an intersection cut for Q_I, *i.e.*, when do we have $\text{conv}(Q_1 \cup Q_2) \neq Q$? The answer depends on the geometry of the null space $\mathcal{L} := \{x \in \mathbb{R}^n : Ax = 0_n\}$ of A and the affine set $\mathcal{A} := \{x \in \mathbb{R}^n : Ax = d\}$, and is as follows (Lemma 1).

(1) **No intersection cut**: We have $\text{conv}(Q_1 \cup Q_2) = Q$ if and only if either $\pi \notin \mathcal{L}^\perp$ or $\mathcal{A}$ is *not* strictly between the hyperplanes $\pi^T x = \pi_0$ and $\pi^T x = \pi_0 + 1$.

To obtain a description of $\text{conv}(Q_1 \cup Q_2)$ when $\text{conv}(Q_1 \cup Q_2) \neq Q$ we apply an affine mapping from $\mathbb{R}^n$ to $\mathbb{R}^m$ to reduce the problem to the following question: Given a disjunction $\delta^T y \leq r_1 \vee \delta^T y \geq r_2$ on $\mathbb{R}^m$ such that the apex 0_m of the Lorentz cone L^m lies strictly between the hyperplanes $\delta^T y = r_1$ and $\delta^T y = r_2$, what is the inequality description of the convex hull $\text{conv}(S_1 \cup S_2)$ of the sets

$$S_1 := \{y \in L^m : \delta^T y \leq r_1\} \text{ and } S_2 := \{y \in L^m : \delta^T y \geq r_2\} \tag{6}$$

of points in the Lorentz cone L^m satisfying the disjunction $\delta^T y \leq r_1 \vee \delta^T y \geq r_2$? The precise formula for $\delta^T y \leq r_1 \vee \delta^T y \geq r_2$ is given in Definition 1 in Sect. 2.

There are two types of intersection cuts that may be needed to describe $\text{conv}(Q_1 \cup Q_2)$: A linear inequality or a conic quadratic inequality. A linear inequality is needed when either $\delta \in L^m$ or $-\delta \in L^m$ as follows (Corollary 1).

(2) **Linear intersection cut**: Suppose $\text{conv}(Q_1 \cup Q_2) \neq Q$. If $\delta \in L^m$, then we have $\text{conv}(Q_1 \cup Q_2) = \{x \in Q : \pi^T x \geq \pi_0 + 1\}$, and if $-\delta \in L^m$, then $\text{conv}(Q_1 \cup Q_2) = \{x \in Q : \pi^T x \leq \pi_0\}$.

The most interesting case is the following situation where a conic quadratic inequality is needed to describe $\text{conv}(Q_1 \cup Q_2)$ (Theorem 3).

(3) **Conic quadratic intersection cut**: If $\text{conv}(Q_1 \cup Q_2) \neq Q$ and $\pm\delta \notin L^m$, then $\text{conv}(Q_1 \cup Q_2)$ is the set of $x \in Q$ such that $y := Ax - d$ satisfies

$$4 \cdot r_1 \cdot r_2 \cdot (\delta^T y - r_1)(\delta^T y - r_2) + (r_1 - r_2)^2 (\sum_{i=1}^{m} y_i^2 - y_m^2) \cdot (\sum_{i=1}^{m} \delta_i^2 - \delta_m^2) \leq 0 \tag{7}$$

Observe that (7) is not in conic quadratic form. We present conic quadratic forms of (7) in Sect. 5. The validity of (7) for $S_1 \cup S_2$ is easy to see: The constant $4r_1 r_2$ is negative since $r_1 < 0 < r_2$, and for any $y \in \mathbb{R}^m$ satisfying $\delta^T y \leq r_1 \vee \delta^T y \geq r_2$ we have $(\delta^T y - r_1)(\delta^T y - r_2) \geq 0$. Furthermore, the condition $\pm\delta \notin L^m$ gives $\sum_{i=1}^m \delta_i^2 - \delta_m^2 > 0$, and finally any $y \in L^m$ satisfies $\sum_{i=1}^m y_i^2 - y_m^2 \leq 0$.

Intersection cut (7) was also obtained independently by Modaresi et al. [12] for the special case when: (a) the matrix A is non-singular, (b) the n^{th} row and column of A are both the n^{th} unit vector, and (c) the split disjunction $\pi^T x \leq \pi_0 \vee \pi^T x \geq \pi_0 + 1$ has $\pi_n = 0$. In this case the (transformed) disjunction $\delta^T y \leq r_1 \vee \delta^T y \geq r_2$ always has $\delta_m = 0$, and the hyperplanes $\delta^T y = r_1$ and $\delta^T y = r_2$ are therefore always parallel to the coordinate axis associated with the last variable y_m. Since the last coordinate is very different from the other coordinates for points in a Lorentz cone, the geometry becomes substantially

more complex when one allows $\delta_m \neq 0$, and it is not clear which inequality is needed. One of the main challenges in the general conic quadratic setting is to identify the missing inequality. For this purpose we decided to consider Gröbner bases from algebraic geometry, and this allowed us to identify (7). Our approach is inspired by a paper of Ranestad and Sturmfels [14] on determining the boundary of the convex hull of a variety.

For mixed integer linear sets intersection cuts and split cuts are equivalent [1]. We give a counterexample in Sect. 6 which shows that this is no longer the case in a conic quadratic setting.

The remainder of the paper is organized as follows. In Sect. 2 we reduce the problem of characterizing the set $\text{conv}(Q_1 \cup Q_2)$ in $\mathbb{R}^n$ to the problem of characterizing the set $\text{conv}(S_1 \cup S_2)$ in $\mathbb{R}^m$. In Sect. 3 we identify inequality (7) by characterizing the boundary of $\text{conv}(S_1 \cup S_2)$ by using the algebraic geometry software called Singular. We prove our main result in Sect. 4. In Sect. 5 we discuss conic quadratic forms of inequality (7), and finally in Sect. 6 we give an example to show that conic quadratic split cuts and intersection cuts are not equivalent.

2 Reduction to the Main Case

We continue studying a mixed integer conic quadratic set $Q_I = \{x \in Q : x_j \in \mathbb{Z} \text{ for } i \in I\}$ with relaxation $Q := \{x \in \mathbb{R}^n : Ax - d \in L^m\}$, where $A \in \mathbb{Q}^{m \times n}$ has $\text{rank}(A) = m$. The split disjunction $\pi^T x \leq \pi_0 \vee \pi^T x \geq \pi_0 + 1$ is arbitrary and gives two sets $Q_1 := \{x \in Q : \pi^T x \leq \pi_0\}$ and $Q_2 := \{x \in Q : \pi^T x \geq \pi_0 + 1\}$.

The main purpose of this section is to reduce the problem of characterizing $\text{conv}(Q_1 \cup Q_2)$ to the problem of characterizing $\text{conv}(S_1 \cup S_2)$, where $S_1 := \{y \in L^m : \delta^T y \leq r_1\}$ and $S_2 := \{y \in L^m : \delta^T y \geq r_2\}$ for some disjunction $\delta^T y \leq r_1 \vee \delta^T y \geq r_2$ on $\mathbb{R}^m$ which will be defined below.

We first characterize when no further inequalities are needed to describe $\text{conv}(Q_1 \cup Q_2)$. The set $\mathcal{L}$ denotes the nullspace of A, and $\mathcal{A}$ denotes the affine set $\mathcal{A} := \{x \in \mathbb{R}^n : Ax = d\} = \bar{x} + \mathcal{L}$, where $\bar{x}$ solves $Ax = d$.

Lemma 1. *We have* $\text{conv}(Q_1 \cup Q_2) \neq Q$ *if and only if*

(i) π *is orthogonal to* $\mathcal{L}$*, and*
(ii) $\mathcal{A}$ *lies strictly between the hyperplanes* $\pi^T x = \pi_0$ *and* $\pi^T x = \pi_0 + 1$*.*

Proof. Suppose (i) is *not* satisfied, *i.e.*, $\pi \notin \mathcal{L}^\perp$. Hence there exists $l \in \mathcal{L}$ such that $\pi^T l < 0$. Clearly $\text{conv}(Q_1 \cup Q_2) \subseteq Q$. Let $z \in Q$ be arbitrary. If $\pi^T z \notin]\pi_0, \pi_0 + 1[$ then $z \in \text{conv}(Q_1 \cup Q_2)$, so we assume $\pi^T z \in]\pi_0, \pi_0 + 1[$. Now $\pi^T l < 0$ implies we can choose $\mu^1, \mu^2 > 0$ such that $z^1 := z + \mu^1 l \in Q_1$ and $z^2 := z - \mu^2 l \in Q_2$. Since z is on the line between z^1 and z^2, we get $z \in \text{conv}(Q_1 \cup Q_2)$.

Next suppose (ii) is *not* satisfied. Wlog let $z \in \mathcal{A}$ satisfy $\pi^T z \leq \pi_0$. Clearly $\text{conv}(Q_1 \cup Q_2) \subseteq Q$. Let $w \in Q$ be arbitrary. We can assume $\pi^T w \in]\pi_0, \pi_0 + 1[$, since otherwise $w \in \text{conv}(Q_1 \cup Q_2)$. Define $r := w - z$. We have $\pi^T r > 0$. Furthermore, since $Az = d$ we have $Ar \in L^m$, and since L^m is a cone this gives $\{z + \alpha \cdot r : \alpha \geq 0\} \subseteq Q$. Also, $z \in Q^1$ and $\pi^T r > 0$ implies $\{z + \alpha \cdot r : \alpha \geq 0\} \subseteq \text{conv}(Q_1 \cup Q_2)$. Since w is on this halfline, we get $w \in \text{conv}(Q_1 \cup Q_2)$.

Finally suppose (i) and (ii) are satisfied. We claim $\bar{x} \notin \text{conv}(Q_1 \cup Q_2)$. Suppose, for a contradiction, that $\bar{x} \in \text{conv}(Q_1 \cup Q_2)$. We do not have $\bar{x} \in Q_1 \cup Q_2$ since $\pi^T \bar{x} \in]\pi_0, \pi_0 + 1[$ by (ii). Hence there exists $\lambda \in]0, 1[$, $x^1 \in Q_1$ and $x^2 \in Q_2$ so that $\bar{x} = \lambda x^1 + (1 - \lambda)x^2$. Let $y^1 := Ax^1 - d$ and $y^2 := Ax^2 - d$. We have $0_m = A\bar{x} - d = \lambda y^1 + (1 - \lambda)y^2$, so $-y^1 = \frac{(1-\lambda)}{\lambda} y^2$. Since $y^2 \in L^m$, $\frac{(1-\lambda)}{\lambda} > 0$ and L^m is a cone this gives $-y^1 \in L^m$. We now have $\pm y^1 \in L^m$, and therefore the line $\{\alpha \cdot y^1 : \alpha \in \mathbb{R}\}$ is contained in L^m. Since L^m is pointed, this implies $y^1 = y^2 = 0_m$. However, then $x^1, x^2 \in \mathcal{A}$, and since $\pi^T z = \pi^T \bar{x} \in]\pi_0, \pi_0 + 1[$ for all $z \in \mathcal{A}$ from (i) and (ii), this is a contradiction. ■

We now present the reduction. Define a disjunction $\delta^T y \leq r_1 \vee \delta^T y \geq r_2$ on $\mathbb{R}^m$ as follows.

Definition 1. *(Definition of the disjunction $\delta^T y \leq r_1 \vee \delta^T y \geq r_2$)*
The vector $\delta \in \mathbb{R}^m$ is the projection of π onto $\mathcal{L}^\perp$, i.e., we define $\delta = (AA^T)^{-1}A\pi$. The numbers $r_1, r_2 \in \mathbb{R}$ are given by $r_1 := \pi_0 - \delta^T d$ and $r_2 := r_1 + 1$.

Given a disjunction $\pi^T x \leq \pi_0 \vee \pi^T x \geq \pi_0 + 1$, we now argue that a description of $\text{conv}(S^1 \cup S^2)$ gives a description of $\text{conv}(Q^1 \cup Q^2)$(Lemma 2.(iii)). This argument is standard and therefore omitted.

Lemma 2. *Suppose $(\pi, \pi_0) \in \mathbb{R}^{n+1}$ satisfies (i) and (ii) of Lemma 1. Then*

(i) $0 \in]r_1, r_2[$ *and* $\text{conv}(S_1 \cup S_2) \neq L^m$,
(ii) $Q_k = \{x \in \mathbb{R}^n : Ax - d \in S_k\}$ *for* $k = 1, 2$, *and*
(iii) $\text{conv}(Q_1 \cup Q_2) = \{x \in \mathbb{R}^m : Ax - d \in \text{conv}(S_1 \cup S_2)\}$.

Observe that Lemma 2.(ii) gives a condition for when a linear inequality suffices to describe $\text{conv}(Q_1 \cup Q_2)$. Indeed, since L^m is a self-dual cone, $\delta \in L^m$ implies $\delta^T z \geq 0$ for all $z \in L^m$. Since $r_1 < 0 < r_2$, this gives $\text{conv}(S_1 \cup S_2) = S_2$ when $\delta \in L^m$. Symmetrically $\text{conv}(S_1 \cup S_2) = S_1$ when $-\delta \in L^m$.

Corollary 1. *Suppose $(\pi, \pi_0) \in \mathbb{Z}^{n+1}$ satisfies (i)-(ii) of Lemma 1.*

(i) If $\delta \in L^m$, then $\text{conv}(S_1 \cup S_2) = S_2$ and $\text{conv}(Q_1 \cup Q_2) = Q_2$.
(ii) If $-\delta \in L^m$, then $\text{conv}(S_1 \cup S_2) = S_1$ and $\text{conv}(Q_1 \cup Q_2) = Q_1$.

3 Describing a Piece of the Boundary of the Convex Hull

We now describe a part of the boundary of $\text{conv}(S_1 \cup S_2)$, where S_1 and S_2 are as defined in Sect. 2 from a disjunction $\delta^T x \leq r_1 \vee \delta^T x \geq r_2$ with $(\delta, r_1, r_2) \in \mathbb{R}^{m+2}$ and $r_1 r_2 < 0$. This will give the inequality needed to describe $\text{conv}(S_1 \cup S_2)$. We assume $\pm\delta \notin L^m$. For simplicity let $C := \text{conv}(S_1 \cup S_2)$. We consider points $x \in \partial C$ each being a convex combination of points $a \in S_1$ and $b \in S_2$ maximizing a linear form $h \in \mathbb{R}^m \setminus \{0_m\}$ over C. These points belong to the set B below.

Definition 2. *Let $F_k := \{x \in \mathbb{R}^m : \sum_{i=1}^{m-1} x_i^2 = x_m^2 \wedge \delta^T x = r_k\}$ for $k = 1, 2$, and let ∇L be the gradient of $x \mapsto x_m^2 - \sum_{i=1}^{m-1} x_i^2$. The set B is defined as:*

$$B := \{x \in \mathbb{R}^m : \exists (h,a,b,t) \in (\mathbb{R}^m \setminus \{0_m\}) \times F_1 \times F_2 \times \mathbb{R} :$$
$$x = ta + (1-t)b \quad \text{and} \quad h^T a = h^T b,$$
$$\dim(\text{span}(h, \nabla L(a), \delta)) \leq 2 \quad \text{and} \quad \dim(\text{span}(h, \nabla L(b), \delta)) \leq 2 \ \}.$$

Theorem 1. *Let $x \in C$ with $r_1 < \delta^T x < r_2$. If $x \in \partial C$ then $x \in B$.*

Proof. Since $r_1 < \delta^T x < r_2$ x must be a convex combination of a point $a \in S_1$ and a point $b \in S_2$. By convexity of L^m we may assume that $a \in H_1$ and $b \in H_2$, where $H_k := \{x \in \mathbb{R}^m : \delta^T x = r_k\}$ for $k = 1, 2$.

Since x is in ∂C, we have $a \in \partial(L^m \cap H_1)$ and $b \in \partial(L^m \cap H_2)$ in the affine spaces H_1 and H_2 respectively. This proves $a_m^2 = \sum_{i=1}^{m-1} a_i^2$ and $b_m^2 = \sum_{i=1}^{m-1} b_i^2$.

Since $x \in \partial C$ and C is convex, there exists an $h \in \mathbb{R}^m \setminus \{0\}$ which as a linear form attains its maximum over C in x. Moving along a line from x towards a or b we stay in C. Therefore, by colinearity of a, b and x, we get $h^T(a-b) = 0$.

Finally, consider the projection $\tilde{h}$ of h to the linear space parallel to H_1. Since h attains its maximum over $C \cap H_1$ at a, the gradient of $a_m^2 - \sum_{i=1}^{m-1} a_i^2$ in the subspace and $\tilde{h}$ are dependent. Hence $\dim(\text{span}(h, \nabla L(a), \delta)) \leq 2$. A similar argument for b shows that $\dim(\text{span}(h, \nabla L(b), \delta)) \leq 2$. ∎

Theorem 2. *Any point $x \in B$ must satisfy the equation $\sum_{i=1}^{m-1} x_i^2 = x_m^2$ or*

$$4r_1 r_2(\delta^T x - r_1)(\delta^T x - r_2) + (r_1 - r_2)^2 (\sum_{i=1}^{m-1} \delta_i^2 - \delta_m^2)(\sum_{i=1}^{m-1} x_i^2 - x_m^2) = 0. \quad (8)$$

These are polynomial equations in x with coefficients involving δ, r_1 and r_2.

Before proving Theorem 2, we show how to deduce the above equations for $m = 3$ in the computer algebra system Singular [10]. In Singular we type:

```
ring r=0,(delta1,delta2,delta3,r1,r2,x1,x2,x3,a1,
                a2,a3,b1,b2,b3,t,h1,h2,h3,A1,A2,B1,B2),dp;
poly f1=a1^2+a2^2-a3^2;
poly f2=a1*delta1+a2*delta2+a3*delta3-r1;
poly g1=b1^2+b2^2-b3^2;
poly g2=b1*delta1+b2*delta2+b3*delta3-r2;
poly K1=-h1+A1*diff(f1,a1)+B1*delta1;
poly K2=-h2+A1*diff(f1,a2)+B1*delta2;
poly K3=-h3+A1*diff(f1,a3)+B1*delta3;
poly L1=-h1+A2*diff(g1,b1)+B2*delta1;
poly L2=-h2+A2*diff(g1,b2)+B2*delta2;
poly L3=-h3+A2*diff(g1,b3)+B2*delta3;
poly R=h1*(a1-b1)+h2*(a2-b2)+h3*(a3-b3);
poly X1=x1-t*a1-(1-t)*b1;
poly X2=x2-t*a2-(1-t)*b2;
poly X3=x3-t*a3-(1-t)*b3;
ideal I=f1,f2,g1,g2,K1,K2,K3,L1,L2,L3,R,X1,X2,X3;
```

```
option(prot);
LIB "elim.lib";
ideal J=h1,h2,h3;
ideal K=eliminate(sat(I,J)[1],a1*a2*a3*b1*b2*b3*t*h1*h2*h3*A1*A2*B1*B2);
LIB "primdec.lib";
primdecGTZ(K);
```

This script produces the desired polynomials. We now explain how it works. Each polynomial in the list

```
ideal I=f1,f2,g1,g2,K1,K2,K3,L1,L2,L3,R,X1,X2,X3;
```

encodes a polynomial equation by setting the polynomial equal to zero. These equations arise from Definition 2. For simplicity we strengthen the dimension conditions to $h \in \text{span}(\nabla L(a), \delta)$ and $h \in \text{span}(\nabla L(b), \delta)\}$, respectively, and express these by introducing the three unknown coefficients of the two linear combinations as variables A_1, B_1, A_2, B_2. In total we form 14 equations.

The ideal I is the infinite set of polynomial consequences of the 14 polynomials obtained by forming linear combinations of these with polynomials as coefficients. As a subset of I we find the ideal $K \subseteq \mathbb{R}[\delta_1, \ldots, \delta_m, x_1, \ldots, x_m, r_1, r_2]$ containing consequences only involving δ, x, r_1, and r_2. Our computation shows that the ideal K is a principal ideal, *i.e.*, all its elements are polynomial multiples of a single polynomial P. The `eliminate` command computes this polynomial P. The last line of the script factors P into $x_m^2 - \sum_{i=1}^{m-1} x_i^2$ and a 37 term polynomial. It is not obvious that this polynomial gives the formula in Theorem 2, but for $m = 3$ the formula can easily be expanded and checked in Singular.

We still need to explain the operation `sat(I,J)` in the script. A priori, the vector h can always be chosen to be 0_3, and hence there would be no consequence for x in terms of δ, r_1 and r_2. To exclude this we *saturate* I wrt. the ideal $J = \langle h_1, h_2, h_3 \rangle$. We refer the reader to [11] for an introduction to elimination and saturation of polynomial ideals.

Proof (Proof of Theorem 2). We substitute $\delta^T a$ for r_1 and $\delta^T b$ for r_2. It remains to prove (under the assumption $h \neq 0_m$) that $\sum_{i=1}^{m-1} x_i^2 = x_m^2$ or

$$4(\delta^T a)(\delta^T b)(\delta^T x - \delta^T a)(\delta^T x - \delta^T b) + (\delta^T (a-b))^2 (\sum_{i=1}^{m-1} \delta_i^2 - \delta_n^2)(\sum_{i=1}^{m-1} x_i^2 - x_m^2) = 0$$

is a consequence of $x = ta + (1-t)b$, $h^T a = h^T b$, $\dim(\text{span}(h, \nabla L(a), \delta)) \leq 2$, $\dim(\text{span}(h, \nabla L(b), \delta)) \leq 2$, $h \neq 0_m$, $\sum_{i=1}^{m-1} a_i^2 - a_m^2 = 0$ and $\sum_{i=1}^{m-1} b_i^2 - b_m^2 = 0$.

To simplify we make substitutions and work over the complex numbers $\mathbb{C}$. In δ and h we multiply the last coordinate by i (where $i^2 = -1$), and in x, a, b we multiply the last coordinate by $-i$. With these changes our assumptions become

- $h \neq 0_m$ and $h \cdot a = h \cdot b$,
- $\{\delta, h, a\}$ and $\{\delta, h, b\}$ are both linearly dependent sets,
- $x = ta + (1-t)b$,
- $a \cdot a = 0$ and $b \cdot b = 0$.

where for $x, y \in \mathbb{C}^m$ we let $x \cdot y := \sum_{i=1}^{m} x_i y_i$. With this notation we must prove

$$4(\delta \cdot a)(\delta \cdot b)(\delta \cdot x - \delta \cdot a)(\delta \cdot x - \delta \cdot b) + (\delta \cdot (a-b))^2(\delta \cdot \delta)(x \cdot x) = 0. \quad (9)$$

First assume $\mathrm{span}(h, \delta, a) \neq \mathrm{span}(h, \delta, b)$. In this case h and δ are proportional, implying $\delta \cdot a = \delta \cdot b$, which equals $\delta \cdot x$. Equation (9) follows easily.

Suppose now $\mathrm{span}(h, \delta, a) = \mathrm{span}(h, \delta, b)$. Then a, b and δ are in the same 2-dimensional plane. Suppose that $\delta = ka + lb$ with $k, l \in \mathbb{C}$. We compute the left hand side of (9):

$$\begin{aligned}
&4((ka+lb)\cdot a)((ka+lb)\cdot b)((ka+lb)\cdot(x-a))((ka+lb)\cdot(x-b))+\\
&((ka+lb)\cdot(a-b))^2((ka+lb)\cdot(ka+lb))(x\cdot x)\\
=&4((ka+lb)\cdot a)((ka+lb)\cdot b)((ka+lb)\cdot((t-1)(a-b)))((ka+lb)\cdot(t(a-b)))+\\
&((ka+lb)\cdot(a-b))^2((ka+lb)\cdot(ka+lb))(x\cdot x)\\
=&((ka+lb)\cdot(a-b))^2(4((ka+lb)\cdot a)((ka+lb)\cdot b)(t-1)t+((ka+lb)\cdot(ka+lb))(x\cdot x))\\
=&((ka+lb)\cdot(a-b))^2(4(lb\cdot a)(ka\cdot b)(t-1)t+((2kla\cdot b)(2(ta)\cdot(1-t)b)))=0.
\end{aligned}$$

In the case $\delta \notin \mathrm{span}(a, b)$ we have that a and b are dependent. Wlog $x = c \cdot a$ for some $c \in \mathbb{C}$. Now $x \cdot x = (ca) \cdot (ca) = c^2(a \cdot a) = c^2 0 = 0$. Translated to our original coordinates, we are in the case where $\sum_{i=1}^{m-1} x_i^2 = x_m^2$. ∎

When $x \in B$ is between the hyperplanes $\delta^T x = r_1$ and $\delta^T x = r_2$ we can exclude one of the cases of Theorem 2.

Lemma 3. *Suppose $x \in B$, with a and b in Definition 2 chosen such that $a_m \geq 0$ and $b_m \geq 0$. Furthermore suppose $r_1 < \delta^T x < r_2$. Then $\sum_{i=1}^{m-1} x_i^2 \neq x_m^2$.*

Proof. Consider the degree two polynomial we get by restricting $\sum_{i=1}^{m-1} x_i^2 - x_m^2$ to the line from a through x to b. This polynomial evaluates to zero in a and b. Since the degree is two it is either the zero polynomial or non-zero between a and b. In the second case we conclude that $\sum_{i=1}^{m-1} x_i^2 \neq x_m^2$. In the first case, every point y on the line passing through a and b satisfies $\sum_{i=1}^{m-1} y_i^2 = y_m^2$. The hypersurface defined by $\sum_{i=1}^{m-1} y_i^2 = y_m^2$ contains only lines passing through the origin. From the inequalities $a_m \geq 0$ and $b_m \geq 0$ it follows that a and b are on the same side of the origin, contradicting $r_1 r_2 < 0, \delta^T a = r_1$ and $\delta^T b = r_2$. ∎

4 Characterization of the Convex Hull

In this section we prove Theorem 3 below. We will need a technical lemma.

Lemma 4. *Let $f \in \mathbb{R}[x_1, \ldots, x_m]$ be the left hand side of (8). Let $t \mapsto a + bt$ be a parametrization of a line. If $\delta^T b \neq 0$ and $b_m^2 > \sum_{i=1}^{m-1} b_i^2$ then $f(a+tb) \to -\infty$ as $t \to \pm\infty$.*

Proof. The summand $4r_1 r_2(\delta^T x - r_1)(\delta^T x - r_2)$ goes to $-\infty$. The second summand of f is either zero or has the sign of $\sum_{i=1}^{m-1} b_i^2 - b_m^2$ since it eventually will be dominated by $(r_1 - r_2)^2(\sum_{i=1}^{m-1} \delta_i^2 - \delta_m^2)t^2(\sum_{i=1}^{m-1} b_i^2 - b_m^2)$. ∎

Lemma 5. *If $x \in C$ and $r_1 < \delta^T x < r_2$ then*

$$4r_1r_2(\delta^T x - r_1)(\delta^T x - r_2) + (r_1 - r_2)^2(\sum_{i=1}^{m-1} \delta_i^2 - \delta_m^2)(\sum_{i=1}^{m-1} x_i^2 - x_m^2) \leq 0. \quad (10)$$

Proof. Let $f \in \mathbb{R}[x_1, \ldots, x_m]$ be the polynomial on the left hands side of (10). First observe that $f(0_m) > 0$. Suppose (10) did not hold for x. Then $f(x) > 0$. Consider the line starting at 0_m passing through x. On this line we find a point on the boundary of C. By Theorem 1, Theorem 2 and Lemma 3 f has value zero here. Furthermore, by Lemma 4 the values of f far from zero are negative. For a degree-two polynomial this is a contradiction. (Note that to apply Lemma 4 we need $\delta^T x \neq 0$. If it was not true, perturb x and f cannot be positive there). ■

Lemma 6. *Let x satisfy (10) and $r_1 < \delta^T x < r_2$. If $x_m > 0$ then $x \in C$.*

Proof. The assumptions imply that the first term of (10) is positive. Since $\pm\delta \notin L^m$ (10) gives $x \in L^m$. Choose $\varepsilon > 0$ such that f is positive on an ε-ball around the origin. In an ε-ball around x we choose a point b such that b is in the interior of L^m and $\delta^T b \neq 0$. Consider the line $x + tb$ and the values that f attains on this line. For $t = -1$ we are in the ε-ball where f is positive. For $t \to \pm\infty$ the function goes to $-\infty$ by Lemma 4. For some $t_0 > 0$ we get $\delta^T(x + t_0 b) = r_i$ for $i = 1$ or $i = 2$. Furthermore, the Lorentz inequality is satisfied, so that $x + t_0 b$ is in C. As we move towards $t = 0$, f will attain value 0 as we pass the boundary of C. After this f stays positive at least until the $x_m = 0$ hyperplane is reached, where f attains a positive value on the line. We conclude $x \in \text{closure}(C)$.

To prove that $x \in C$, suppose this is not the case. Intersect C with $\{y \in \mathbb{R}^m : y_m - 1 \leq x_m\}$. This intersection is compact because the convex hull of a compact set is compact. We conclude $x \in C$. ■

By combining Lemma 5 and Lemma 6 we obtain our main theorem.

Theorem 3. *Assume $\pm\delta \notin L^m$. Then $x \in C$ if and only if $x \in L^m$ and*

$$4r_1r_2(\delta^T x - r_1)(\delta^T x - r_2) + (r_1 - r_2)^2(\sum_{i=1}^{m-1} \delta_i^2 - \delta_m^2)(\sum_{i=1}^{m-1} x_i^2 - x_m^2) \leq 0. \quad (11)$$

Proof. Suppose $x \in C$. Clearly $x \in L^m$ since $C \subseteq L^m$. If $r_1 < \delta^T x < r_2$ then (11) follows from Lemma 5. If $r_1 \geq \delta^T x$ or $\delta^T x \geq r_2$, then the first term on the left hand side of (11) is ≤ 0. The second term is ≤ 0 since $x \in C \subseteq L^m$.

Conversely, suppose $x \in L^m$ and (11) is satisfied. If $r_1 < \delta^T x < r_2$, then $x \in C$ by Lemma 6. If not then $x \in C$ by the definition of C. ■

5 Conic Quadratic Representations

We have identified (11) for describing $C = \text{conv}(S_1 \cup S_2)$. However, this inequality is not in conic quadratic form. In this section we give conic quadratic representations of (11). We first consider the special case when $\delta_2 = \ldots = \delta_{m-1} = 0$.

Proposition 1. *If we assume that* $\delta_2 = \delta_3 = \cdots = \delta_{m-1} = 0$ *then*

$$4r_1r_2(\delta^T x - r_1)(\delta^T x - r_2) + (r_1 - r_2)^2(\sum_{i=1}^{m-1} \delta_i^2 - \delta_m^2)(\sum_{i=1}^{m-1} x_i^2 - x_m^2) =$$

$$((r_1+r_2)(\delta_1 x_1+\delta_m x_m)-2r_1r_2)^2+(r_1-r_2)^2(\delta_1^2-\delta_m^2)(\sum_{i=2}^{m-1} x_i^2)-(r_1-r_2)^2(\delta_m x_1+\delta_1 x_m)^2.$$

In particular, if $\pm\delta \notin L^m$ *the polynomial is a conic quadratic form with* m *terms.*

The proof of Proposition 1 is simply a sequence of equalities and therefore omitted. We now give an interpretation of the coefficients in the expression of Proposition 1. For simplicity suppose furthermore that $\delta_1 > 0$. Then

- $\delta_1 x_1 + \delta_m x_m = \delta^T x$
- $\delta_1^2 - \delta_m^2 = \sum_{i=1}^{m-1} \delta_i^2 - \delta_m^2$
- $\delta_m x_1 + \delta_1 x_m = \frac{\delta_m}{\sqrt{\sum_{i=1}^{m-1} \delta_i^2}} \begin{pmatrix} x_1 \\ \vdots \\ x_{m-1} \end{pmatrix} \cdot \begin{pmatrix} \delta_1 \\ \vdots \\ \delta_{m-1} \end{pmatrix} + x_m \sqrt{\sum_{i=1}^{m-1} \delta_i^2}$
- $\sum_{i=2}^{m-1} x_i^2$ is the squared norm of the projection of x to $\text{span}(\delta, e_m)^\perp$

Except for the third item, these quantities have geometric meaning. All are invariant under orthonormal linear transformation fixing the last coordinate. Since such transformations preserve the Lorentz cone, the assumption $\delta_2 = \cdots = \delta_{m-1} = 0$ was made without loss of generality, and in general the coefficients of our quadratic equation can be obtained from the right hand sides above.

The sum $\sum_{i=2}^{m-1} x_i^2$ remains a sum of squares after a linear transformation of coordinates. If $\delta_2, \ldots, \delta_{m-1}$ are not all zero, we still want a closed form formula. Let $b_2, \ldots, b_{m-1}$ be an orthogonal basis for $\text{span}(\delta, e_m)^\perp$. Then the squared length of the projection of x to this subspace is given by

$$\frac{(x \cdot b_2)^2}{b_2 \cdot b_2} + \cdots + \frac{(x \cdot b_{m-1})^2}{b_{m-1} \cdot b_{m-1}}.$$

We have reached the following generalization of Proposition 1

Proposition 2. *Let* $b_2, \ldots, b_{m-1}$ *be an orthogonal basis for* $\text{span}(\delta, e_m)^\perp$. *Then*

$$4r_1r_2(\delta^T x - r_1)(\delta^T x - r_2) + (r_1 - r_2)^2(\sum_{i=1}^{m-1} \delta_i^2 - \delta_m^2)(\sum_{i=1}^{m-1} x_i^2 - x_m^2) =$$

$$((r_1 + r_2)\delta^T x - 2r_1r_2)^2 + (r_1 - r_2)^2(\delta_1^2 - \delta_m^2)\left(\frac{(x \cdot b_2)^2}{b_2 \cdot b_2} + \cdots + \frac{(x \cdot b_{m-1})^2}{b_{m-1} \cdot b_{m-1}}\right)$$

$$-(r_1 - r_2)^2 \left(\frac{\delta_m}{\sqrt{\sum_{i=1}^{m-1} \delta_i^2}} \begin{pmatrix} x_1 \\ \vdots \\ x_{m-1} \end{pmatrix} \cdot \begin{pmatrix} \delta_1 \\ \vdots \\ \delta_{m-1} \end{pmatrix} + x_m \sqrt{\sum_{i=1}^{m-1} \delta_i^2}\right)^2.$$

Proposition 2 gives a general scheme for obtaining conic quadratic forms of inequality (11). We now give a concrete conic quadratic form of (11), which can be directly computed from the data $(\delta, r_1, r_2) \in \mathbb{R}^{m+2}$. Let $x^k := (x_1, \ldots, x_k)$ and $\delta^k = (\delta_1, \ldots, \delta_k)$ be vectors of the first k coordinates of x and δ. Inequality (11) can be written as:

$$((r_1 + r_2)\delta^T x - 2r_1 r_2)^2 - (r_1 - r_2)^2 ||\delta^{m-1}||^2 (x_m + \frac{(\delta^{m-1})^T x^{m-1}}{||\delta^{m-1}||^2}\delta_m)^2$$
$$+ (r_1 - r_2)^2 (||\delta^{n-1}||^2 - \delta_m^2)(\sum_{k=2}^{m-1} \frac{||\delta^{k-1}||^2}{||\delta^k||^2}(x_k - \frac{(\delta^{k-1})^T x^{k-1}}{||\delta^{k-1}||^2}\delta_k)^2) \leq 0. \quad (12)$$

6 Conic Quadratic Intersection Cuts and Split Cuts

In a linear setting split cuts and intersection cuts are equivalent [1]. We now give an example showing that this is not true in a conic quadratic setting. Consider a mixed integer conic quadratic set $Q_I := \{x \in Q : x_j \in \mathbb{Z} \text{ for } j \in I\}$ with continuous relaxation $Q := \{x \in \mathbb{R}^n : Ax - d \in L^m\}$, where the rows of A are *not* linearly independent. For a split disjunction $\pi^T x \leq \pi_0 \vee \pi^T x \geq \pi_0 + 1$, a *split cut* for Q_I is a valid inequality for $\text{conv}(Q_1 \cup Q_2)$ with $Q_1 = \{x \in Q : \pi^T x \leq \pi_0\}$ and $Q_2 = \{x \in Q : \pi^T x \geq \pi_0 + 1\}$ that is *not* valid for Q.

Example 1. The conic quadratic set

$$Q := \{(x, y) \in \mathbb{R}^2 : \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \in L^3\} \quad (13)$$

equals $\{(x, y) \in \mathbb{R}^2 : 1 \leq 2xy \wedge y > 0\}$ and is shown in Figure 1. Consider the relaxation Q^B of Q obtained from the first and last row of A:

$$Q^B := \{(x, y) \in \mathbb{R}^2 : (x - 1)^2 \leq (x + y - 1)^2 \wedge x + y - 1 \geq 0\}.$$

The set Q^B is polyhedral since Q^B the preimage of the 2-dimensional Lorentz cone under a linear map. We may think of Q^B as a relaxation obtained by

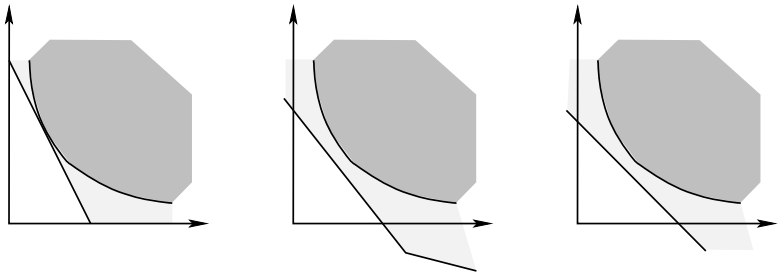


Fig. 1. The conic quadratic set Q of (13) and relaxations for $v = (1, 0), (\sqrt{3/4}, 1/2)$ and $(\sqrt{1/2}, \sqrt{1/2})$ respectively

substituting L^3 with $L^3 + \mathbb{R} \cdot e_2$ in (13). By instead choosing $L^3 + \mathbb{R} \cdot e_1$ we get the relaxation $Q^{\tilde{B}}$ of Q obtained from the last two rows of the matrix defining Q. In general, adding any line generated by some $v \in \mathbb{R}^2 \times \{0\}$ to L^3 gives a relaxation of Q. Relaxations for three choices of v are shown in Figure 1. The important observation is that the boundary of such a relaxation is tangent to the boundary of Q *in at most one point.*

Now consider any split disjunction (π, π_0), and suppose the intersection cut is a secant line between two points a, b on the curve $1 = 2xy$. For an intersection cut from a relaxation of the above type to give the same cut, the relaxation must contain *both* a and b in the boundary, which as argued above is impossible. Conic quadratic intersection cuts are therefore not always split cuts.

References

1. Andersen, K., Cornuéjols, G., Li, Y.: Split closure and intersection cuts. Mathematical Programming A 102, 457–493 (2005)
2. Atamtürk, A., Narayanan, V.: Conic mixed-integer rounding cuts. Mathematical Programming 122(1), 1–20 (2010)
3. Atamtürk, A., Narayanan, V.: Lifting for conic mixed-integer programming. Mathematical Programming A 126, 351–363 (2011)
4. Balas, E.: Intersection cuts - a new type of cutting planes for integer programming. Oper. Res. 19, 19–39 (1971)
5. Balas, E., Ceria, S., Cornuéjols, G.: A lift-and-project cutting plane algorithm for mixed 0-1 programs. Mathematical Programming 58, 295–324 (1993)
6. Bixby, R.E., Gu, Z., Rothberg, E., Wunderling, R.: Mixed integer programming: A progress report. In: Grötschel, M. (ed.) The Sharpest Cut: The Impact of Manfred Padberg and his Work. MPS/SIAM Ser. Optim., pp. 309–326 (2004)
7. Çezik, M., Iyengar, G.: Cuts for mixed 0-1 conic programming. Mathematical Programming 104(1), 179–202 (2005)
8. Cook, W.J., Kannan, R., Schrijver, A.: Chvátal closures for mixed integer programming problems. Mathematical Programming 47, 155–174 (1990)
9. Gomory, R.: An algorithm for the mixed integer problem. Technical Report RM-2597, The Rand Coporation (1960)
10. Decker, W., Greuel, G.-M., Pfister, G., Schönemann, H.: SINGULAR 3-1-3 — A computer algebra system for polynomial computations, University of Kaiserslautern (2011), `http://www.singular.uni-kl.de`
11. Greuel, G.-M., Pfister, G.: A Singular Introduction to Commutative Algebra, 2nd edn. Springer Publishing Company, Incorporated (2007)
12. Modaresi, S., Kilinc, M., Vielma, J.P.: Split Cuts for Conic Programming. Poster presented at the MIP 2012 Workshop at UC Davis
13. Nemhauser, G., Wolsey, L.: A recursive procedure to generate all cuts for 0-1 mixed integer programs. Mathematical Programming 46, 379–390 (1990)
14. Ranestad, K., Sturmfels, B.: The convex hull of a variety. In: Branden, P., Passare, M., Putinar, M. (eds.) Notions of Positivity and the Geometry of Polynomials. Trends in Mathematics, pp. 331–344. Springer, Basel (2011)
15. Stubbs, R.A., Mehrotra, S.: A branch-and-cut method for 0-1 mixed convex programming. Mathematical Programming 86(3), 515–532 (1999)

Content Placement via the Exponential Potential Function Method

David Applegate, Aaron Archer, Vijay Gopalakrishnan,
Seungjoon Lee, and K.K. Ramakrishnan

AT&T Shannon Research Laboratory, 180 Park Avenue, Florham Park, NJ 07932
{david,aarcher,gvijay,slee,kkrama}@research.att.com

Abstract. We empirically study the exponential potential function (EPF) approach to linear programming (LP), as applied to optimizing content placement in a video-on-demand (VoD) system. Even instances of modest size (e.g., 50 servers and 20k videos) stretch the capabilities of LP solvers such as CPLEX. These are packing LPs with block-diagonal structure, where the blocks are fractional uncapacitated facility location (UFL) problems. Our implementation of the EPF framework allows us to solve large instances to 1% accuracy 2000x faster than CPLEX, and scale to instances much larger than CPLEX can handle on our hardware.

Starting from the packing LP code described by Bienstock [4], we add many innovations. Our most interesting one uses priority sampling to shortcut lower bound computations, leveraging fast block heuristics to magnify these benefits. Other impactful changes include smoothing the duals to obtain effective Lagrangian lower bounds, shuffling the blocks after every round-robin pass, and better ways of searching for OPT and adjusting a critical scale parameter. By documenting these innovations and their practical impact on our testbed of synthetic VoD instances designed to mimic the proprietary instances that motivated this work, we aim to give a head-start to researchers wishing to apply the EPF framework in other practical domains.

Keywords: exponential potential function, approximate linear programming, Dantzig-Wolfe decomposition, priority sampling, content placement, video-on-demand.

1 Introduction

Exponential potential function (EPF) methods for approximately solving linear programs (LPs) have a long history of both theory and implementation. These methods are most attractive when the constraint matrix has a large block-diagonal piece plus some coupling constraints, particularly if there is a fast oracle for optimizing linear functions over each block (called a *block optimization*). While the main focus has been on multicommodity flow (MCF) problems, Bienstock's implementation for generic packing LPs [4, Ch. 4] has been effective also for some network design problems, and Müller, Radke and Vygen [23] have produced an effective code for VLSI design.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 49–61, 2013.

Using Bienstock's book as a starting point, we adapted the EPF method to solve very large LPs arising from the content placement problem in a video-on-demand (VoD) system. After several major algorithmic improvements and some careful engineering, our code is able to solve instances with 2.5 billion variables and constraints to 1% accuracy in about 2.5 minutes. The largest instance CPLEX can solve on the same hardware is 1/50th as big, taking 3 to 4 hours. Most of our algorithmic insights apply to the EPF method in general, although some are specific to our VoD problem. Our goals are two-fold: to describe our innovations and their practical impact, and to give empirical insight into the workings of the EPF method, as a guidepost to future implementers.

Our orientation is thoroughly empirical: we explore and report what works well in practice, and do not aim to improve theoretical running times. Our implementation is based on the FPTASes in the literature, but we view broken proofs as an acceptable price to pay for substantial improvements in the performance of our code on our testbed. Interior-point LP codes take a similar tack, using parameter settings with better empirical performance than the ones that lead to provable polynomial-time convergence [15].

Related Work. Building on the *flow deviation* method of Fratta, Gerla and Kleinrock [12], Shahrokhi and Matula [28] used an EPF to establish the first combinatorial FPTAS for the min-congestion MCF (a.k.a. *max concurrent flow*) problem. The myriad ensuing advances fall mainly into three lines. Fleischer [11], Garg and Könemann [13], Mądry [22], and the references therein contain improvements and extensions to other versions of MCF. Plotkin, Shmoys and Tardos [25] extended the MCF ideas to general packing and covering LPs; Koufogiannakis and Young [20] and the references therein contain further generalizations in this direction. Grigoriadis and Khachiyan [16] generalized in the direction of convex optimization; Müller, Radke, and Vygen [23] and the references therein extend this line of work. The survey by Arora, Hazan and Kale [2] draws connections between these algorithms and multiplicative weight update methods in other areas such as machine learning. All of these running times depend on the approximation error ϵ as ϵ^{-2}, and Klein and Young [19] give strong evidence that this is a lower bound for Dantzig-Wolfe methods like these. Building on work of Nesterov [24], Bienstock and Iyengar [6] obtained an algorithm with ϵ^{-1} dependence. However, their block optimizations solve separable quadratic programs rather than LPs. In practice, the improved ϵ dependence may or may not outweigh the increased iteration complexity.

In the literature, most implementations of the EPF framework are for MCF [3,8,14,17,18,21,27]. They show that the EPF method can outperform general-purpose LP solvers such as CPLEX by 2 or 3 orders of magnitude, but good performance requires careful implementation. Here, the blocks are ordinary flow problems, often shortest paths. Müller, Radke and Vygen [23] address a very different problem domain: VLSI design. Their blocks are fractional Steiner tree problems. Bienstock's code solves mixed packing and covering problems [5], although his published description covers only packing problems [4, Ch. 4].

Focusing on generality and minimizing iterations, he solves blocks as generic LPs using CPLEX. He reports success with both MCF and network design problems. His code departs strongly from the theory by relying heavily on a *bootstrap* method (a.k.a. *procedure NX*), and using the block iterations mainly as an elaborate form of column generation for the bootstrap. Extending this idea, Bienstock and Zuckerberg [7] solve very large open-pit mining LPs (millions of variables and constraints) *to optimality* using *only* the bootstrap.

Our Contributions. Bienstock's primary argument for the bootstrap is the strong lower bounds it yields. Our initial implementation replicated Bienstock, and although the bootstrap produced excellent lower bounds as advertised, we found that its time and memory requirements posed our primary barrier to scalability, so we got rid of it. We found that smoothing the ordinary EPF duals to use as Lagrangian multipliers gave good lower bounds. Each Lagrangian lower bound is expensive, as it requires a full pass of block optimizations. Our most interesting contribution is a method for shortcutting these lower bound passes, by extending the priority sampling techniques of [10,31], combined with cached block solutions and judicious use of block heuristics. Other empirically important contributions include a better way of adjusting an important scale parameter δ over the course of the run, replacing the usual binary search with a radically different method for searching for OPT, and smart chunking strategies to exploit parallelism. While round-robin [4,11,26] and uniform random [16,25] block selection strategies have been proposed in the past, we found round-robin with a random shuffle after each pass to be dramatically more powerful. All of these techniques apply to the EPF framework in general.

In our VoD application, the blocks are fractional uncapacitated facility location (UFL) problems. We use a greedy dual heuristic and a primal heuristic based on the local search algorithm of Charikar and Guha [9], for the *integer* UFL problem. This is insane for two reasons: we're using an approximation algorithm for an NP-hard problem as a heuristic for its LP relaxation, and the integrality gap means we may get worse solutions. Regardless, it is incredibly effective. Our primal and dual heuristics prove each other optimal most of the time, with only small gaps otherwise. Our primal heuristic is 30x to 70x faster than CPLEX, and our dual heuristic is 10x to 30x faster, where the speedup grows with network size. Our heuristic block solves are so fast that their running time is on par with the mundane data manipulation that surrounds them.

Section 2 describes the EPF framework. Section 3 describes our VoD model and our testbed of LPs. Section 4 compares our running times to CPLEX, and breaks them down by major components. Section 5 describes our key improvements to the basic framework, and demonstrates their practical impact on our testbed. Our end result is a code achieving a 2000x speedup over CPLEX, solving to 1% accuracy for the largest instances that CPLEX fits into memory on our system. The gap grows with problem size, and we easily solve instances that are 50x larger.

2 EPF Framework

We first discuss the general principles of the EPF framework, and then describe our algorithm as one instantiation. The EPF framework is a Dantzig-Wolfe decomposition method that uses exponential penalty functions in two capacities: first to define a potential function that encodes relaxed feasibility problems, and second to define Lagrange multipliers for computing lower bounds. Consider this LP:

$$\min cz \text{ s.t. } Az \leq b, z \in F = F^1 \times \cdots \times F^K \subseteq \mathbb{R}^n, \tag{1}$$

where each F^k is a polytope, $A = (a_{ij})$ is an $m \times n$ matrix, $c \in \mathbb{R}_n$ and $b \in \mathbb{R}^m$. Let OPT denote its optimum. Solution $z \in F$ is ϵ-*feasible* if $Az \leq (1+\epsilon)b$ (i.e., it violates each coupling constraint by at most $1+\epsilon$), and it is ϵ-*optimal* if $cz \leq (1+\epsilon)\text{OPT}$. Given constant ϵ, we aim for an ϵ-feasible, ϵ-optimal solution.

Dantzig-Wolfe decomposition takes advantage of fast algorithms for optimizing linear objective functions over each F^k. In this paper, we assume that the coupling constraints constitute a *packing* problem, i.e., $a_{ij} \geq 0$ and $b_i > 0$. Let the set R index the rows of A, let $a_i \in \mathbb{R}_n$ denote row i of A, let index 0 refer to the objective, and define $R^* = R \cup \{0\}$. Given a row vector of Lagrange multipliers $\lambda \in \mathbb{R}_{R^*}$ with $\lambda \geq 0$ and $\lambda_0 > 0$, define $c(\lambda) = c + \frac{1}{\lambda_0}\lambda_R A$. Whenever $k \in \mathcal{B} := \{1, \ldots, K\}$, a superscript k denotes the portion of an object corresponding to block k, e.g., z^k, A^k, F^k, or $c^k(\cdot)$. Define

$$LR^k(\lambda) = \min_{z^k \in F^k} c^k(\lambda) z^k, \tag{2}$$

and $LR(\lambda) = \sum_{k \in \mathcal{B}} LR^k(\lambda) - \frac{1}{\lambda_0}\lambda_R b$, where the notation λ_R means to restrict vector $\lambda \in \mathbb{R}_{R^*}$ to its R components (i.e., exclude λ_0). Standard duality arguments show that $LR(\lambda) \leq \text{OPT}$. All of our lower bounds derive from this fact.

The heart of the EPF method addresses feasibility problems, so we will guess a value B for OPT and consider the problem FEAS(B), wherein we replace the objective in (1) with the constraint $cz \leq B$. A solution is ϵ-feasible for FEAS(OPT) iff it is ϵ-feasible, ϵ-optimal for (1). With OPT unknown, we must search on B.

Define $\alpha(\delta) = \frac{\gamma \log(m+1)}{\delta}$, where $\gamma \geq 1$ and δ is a scale factor that evolves over the course of the algorithm. Let $r_i(z) = \frac{1}{b_i} a_i z - 1$ be the relative infeasibility of constraint i, and define aliases $a_0 := c$ and $b_0 := B$ so that $r_0(z) = \frac{1}{B} cz - 1$. Define $\delta_c(z) = \max_{i \in R} r_i(z)$ as the max relative infeasibility over the coupling constraints, and $\delta(z) = \max(\delta_c(z), r_0(z))$. Let $\Phi_i^\delta(z) = \exp(\alpha(\delta) r_i(z))$ define the potential due to constraint i, and $\Phi^\delta(z) = \sum_{i \in R^*} \Phi_i^\delta(z)$ define the overall potential function we aim to minimize (for fixed δ). If z is feasible for (1) then each $r_i(z) \leq 0$ so $\Phi^\delta(z) \leq m+1$, whereas if even one constraint i has $r_i(z) \geq \delta$, then $\Phi^\delta(z) > \Phi_i^\delta(z) \geq (m+1)^\gamma \geq m+1$. Thus, minimizing $\Phi^\delta(z)$ either finds a δ-feasible z, or proves that no feasible z exists.

Algorithm 1. EPF framework

Parameters: approximation tolerance $\epsilon > 0$, exponent factor $\gamma \approx 1$, smoothing parameter $\rho \in [0, 1)$, chunk size $s \in \mathbb{N}$
Initialize: solution $z \in F$, LB = valid lower bound on OPT, UB $\leftarrow \infty$, objective target $B \leftarrow$ LB, smoothed duals $\bar{\pi} = \pi^\delta(z)$, scale parameter $\delta = \delta(z)$, number of chunks $N_{\text{ch}} = \lceil K/s \rceil$
for Pass $= 1, 2, \ldots$ **do**
  Select a permutation σ of the blocks $\mathcal{B}$ uniformly at random, and partition $\mathcal{B}$ into chunks $C_1, \ldots, C_{N_{\text{ch}}}$, each of size s, according to σ.
  for chunk $C = C_1, \ldots, C_{N_{\text{ch}}}$ **do**
    for each block $k \in C$ **do**
      optimize block: $\hat{z}^k \leftarrow \arg\min_{z^k \in F^k} c^k(\pi^\delta(z))z^k$
      compute step size: $\tau^k \leftarrow \arg\min_{\tau \in [0,1]} \Phi^\delta(z + \tau(\hat{z}^k - z^k))$
      take step in this block: $z^k \leftarrow z^k + \tau^k(\hat{z}^k - z^k)$
      Save $\hat{z}^k$ for possible use in shortcutting step 15 later.
    shrink scale if appropriate: $\delta = \min(\delta, \delta(z))$
    if $\delta_c(z) \leq \epsilon$ (i.e., z is ϵ-feasible) and $cz <$ UB **then** UB $\leftarrow cz$, $z^* \leftarrow z$
    if UB $\leq (1+\epsilon)$LB **then** return z^*
    $\bar{\pi} \leftarrow \rho\bar{\pi} + (1-\rho)\pi^\delta(z)$
  lower bound pass: LB $\leftarrow \max(\text{LB}, LR(\bar{\pi}))$, $B \leftarrow$ LB
  if UB $\leq (1+\epsilon)$LB **then** return z^*

The plan is to minimize $\Phi^\delta(z)$ via gradient descent. Let $\pi_i^\delta(z) = \Phi_i^\delta(z)/b_i$ for $i \in R^*$, and $g(z) = \pi_0^\delta(z)c + \pi_R^\delta(z)A$. The gradient of the potential function is

$$\nabla\Phi^\delta(z) = \alpha(\delta)g(z) = \alpha(\delta)\pi_0^\delta(z)c\left(\pi^\delta(z)\right), \tag{3}$$

a positive scalar times $c\left(\pi^\delta(z)\right)$. By gradient descent, we mean to move z along some segment so as to decrease $\Phi^\delta(z)$ at maximum initial rate. More precisely, defining $z(\tau) = (1-\tau)z + \tau\hat{z}$, we choose $\hat{z} \in F$ to minimize the directional derivative $\frac{d}{d\tau}\Phi^\delta(z(\tau))|_{\tau=0} = \nabla\Phi^\delta(z)(\hat{z} - z) = \alpha(\delta)\pi_0^\delta(z)c\left(\pi^\delta(z)\right)(\hat{z} - z)$. This is equivalent to solving the optimization problem (2) with $\lambda = \pi^\delta(z)$, once for each block $k \in \mathcal{B}$. Thus, solving the Lagrangian relaxation of (1) with this choice of multipliers serves twin purposes: giving a primal search direction and a lower bound on OPT. If we require only a search direction, we can optimize just a single block k and step in that block, leaving the others fixed. This block iteration is the fundamental operation of the EPF method.

Our full algorithm is extremely intricate. For clarity of exposition, we begin our description with a high-level overview in pseudocode as Algorithm 1. Our approach departs from that of both the theory and previous experimental work in several key ways. Some of these departures are evident in the pseudocode, but most are embedded in how we implement key steps. We now outline them briefly, deferring details to Section 5.

Instead of locating OPT via binary search on B, we use Lagrangian lower bounds directly and employ an *optimistic-B* strategy, setting $B \leftarrow$ LB. This departs strongly from Bienstock [4], whose lower bounds come from his bootstrap procedure. The scale parameter δ is critical because it appears in the

denominator of the exponentials, and strongly affects the step sizes τ^k. Earlier work changed δ by discrete factors of 2, which we found to be quite disruptive to convergence. Instead, we lower δ gradually as $\delta(z)$ falls. Continuous δ works harmoniously with optimistic-B, since it avoids the spikes in $\delta(z)$ associated with decreases in B during binary search.

The theory suggests using the dual weights $\pi^\delta(z)$ in step 15, but we discovered that the smoothed duals $\bar{\pi}$ yield stronger and more consistent lower bounds. Although we cannot justify it, replacing $\pi^\delta(z)$ with $\bar{\pi}$ in the block solves (step 7) improves iteration performance, even though $\hat{z} - z$ is no longer a gradient direction. Fast block heuristics dramatically speed up steps 7 and 15, and we can partially parallelize the chunk iteration (steps 6–10), both with minimal impact on iteration performance. The simple idea of shuffling the round-robin order for each pass (step 4) has a surprisingly dramatic impact on iteration performance. We solve some knapsack problems to initialize z and LB in step 2 (details omitted for space). Finally, we use $\gamma \leftarrow 1$, $s \leftarrow 120$, and $\rho \leftarrow 0.001^{1/N_{\text{ch}}}$.

3 VoD Model, Testbed and Machine Environment

Our VoD model, introduced in [1], begins with a set of *video hub offices* (VHOs), each serving video requests from users in a metropolitan area. High-speed links connect the VHOs, allowing them to satisfy a local request by streaming the video from a remote VHO. Given a demand forecast, we desire an assignment of video content to VHOs that minimizes total network traffic while respecting link bandwidths and VHO disk capacities. Variable y_i^k indicates whether to place video k at VHO i, and x_{ij}^k denotes what fraction of requests for video k at VHO j should be served from VHO i. The blocks are fractional UFL problems. In reality, the y_i^k variables should be binary, and we have an effective heuristic that rounds an LP solution, one video at a time. In practice, this tends to blow up the disk and link utilizations by about 1%, and the objective by about 1% to 2%, relative to the LP solution. Therefore, we focus on finding an ϵ-feasible, ϵ-optimal solution to the LP, with $\epsilon = 1\%$. Since the content placement would be re-optimized weekly, we can afford to let the optimization run for several hours. Disk and link capacity are treated as fixed at this time scale. Optimizing those over a longer planning horizon is another interesting problem, but not our present focus.

We conducted experiments on a testbed of 36 synthetic instances, consisting of all combinations of 3 real network topologies (Tiscali, Sprint, and Ebone, taken from Rocketfuel [29]), 6 library sizes (ranging from 5k to 200k videos), and 2 disk size scenarios (small disk/large bandwidth, and vice versa). Our testbed is available at `http://www.research.att.com/~vodopt`. These non-proprietary instances were designed to mimic the salient features of the proprietary instances that motivated this work. In each instance, we set the uniform link bandwidth just slightly larger than the minimum value that makes the instance feasible. We ran our experiments on a system with two 6-core 2.67GHz Intel Xeon X5650

Table 1. Running time, memory usage , and number of passes. Each row aggregates 6 instances (i.e., 3 networks and 2 disk types).

library	CPLEX		Block (100 seeds)			
size	time (s)	mem (GB)	time (s)	# passes	mem (GB)	speedup
5,000	894.47	10.15	1.39	15.37	0.11	644x
10,000	2062.10	19.36	1.77	9.29	0.18	1168x
20,000	5419.57	37.63	2.62	6.85	0.33	2071x
50,000			5.44	5.94	0.77	
100,000			10.45	5.57	1.52	
200,000			20.03	5.25	3.02	
1,000,000			98.61	5.07	15.00	

CPUs (12 cores total) and 48GB of memory. Our code is written in C++, compiled with the GNU g++ compiler v4.4.6. For our exact (not heuristic) block optimizations, we use CPLEX version 12.3.

4 Top-Line Results

Table 1 compares the running time and memory usage of our code with $\epsilon = 1\%$ to the CPLEX parallel barrier code for solving the same instances to optimality. The results reported for CPLEX reflect a single run. The results for our code use 100 different random seeds, since there is some variability across seeds, owing primarily to the block shuffling (Section 5.3). For this experiment only, we included instances with one million videos to emphasize scalability. In 38 instances out of 42, the running time's standard deviation is within 10% of its mean. In 33 instances, the number of passes has standard deviation within 5% of its mean. We take the arithmetic mean over the 100 random seeds for each instance, then take the geometric mean over the 6 instances of each library size, and report these numbers. The memory footprint and running time of our code both scale about linearly with library size (modulo a constant offset). The CPLEX memory footprint also scales linearly, but its running time is decidedly superlinear. For the largest instances that CPLEX could solve, our code is 2000x faster.

Both our code and CPLEX ran on all 12 cores. Our code achieves an 8x parallel speedup on its block solves, but only 4x overall. CPLEX achieves a 3x parallel speedup.

Our code's total running time over these 4200 runs breaks down as follows. Block solves (step 7) account for 24.2%, line search (step 8) for 3.4%, and the remainder of block iterations (steps 9-14) for 22.7%. In the LB pass (step 15, see Section 5.1), heuristic LB passes account for 17.5% and exact CPLEX passes for 0.7%. Initializing z and LB (step 2) accounts for 17.0%, and various overheads (e.g., reading instance data, data structure setup, etc.) for the remaining 14.4%.

5 Key Algorithmic Improvements

We now describe each of our key algorithmic ideas in some detail, and report experiments quantifying their impact. Unless specified otherwise, each experiment

involves running with 10 different random seeds on each of the 36 instances. In reporting our data, we took an arithmetic mean over the 10 runs on each instance. When aggregating further, we then use a geometric mean across instances, to cope with the differing scales.

5.1 Shortcutting the Lower Bound Passes

Since step 15 is executed only once per primal pass, these LB passes account for at most half of the block solves. We can cut this further, by using the statistical technique of *priority sampling* to abort an LB pass early if we have high confidence that finishing it will not yield a useful bound.

Priority Sampling. Duffield, Lund and Thorup's priority sampling is a non-uniform sampling procedure that yields low-variance unbiased estimates for weighted queries, where the query need not be known at the time the sample is computed [10]. We describe it in the abstract before explaining how to apply it in our context. Given a set of items $i \in I$, non-negative item weights w_i, and a sample size N, priority sampling selects a random sample $S_N \subseteq I$ with $|S_N| = N$ and weight estimators $\hat{w}_i$ for $i \in S_N$ with the following properties:

- **The estimator is unbiased [10]:** for all query vectors $f \in [0,1]^I$,

$$\sum_{i \in I} f_i w_i = E\left[\sum_{i \in S_N} f_i \hat{w}_i\right] . \tag{4}$$

- **The estimator is nearly optimal:** in a sense formalized by Szegedy [30], it has lower variance than the best unbiased estimator using $N-1$ samples.
- **The estimator comes with confidence bounds [31]:** Given error tolerance $\xi > 0$, there are readily computable lower and upper confidence bounds $LC(\xi)$ and $UC(\xi)$ (depending also on w, f, and the random draw) such that $\Pr[LC(\xi) > \sum_{i \in I} f_i w_i] \leq \xi$ and $\Pr[UC(\xi) < \sum_{i \in I} f_i w_i] \leq \xi$. The confidence bounds assume adversarial w and f; the probability is over the random draw.
- **The estimator (4) evaluates f_i and $\hat{w}_i$ only for $i \in S_N$:** In the applications that originally motivated priority sampling, I and w_i were presented in a stream and priority sampling limited the memory required for $\hat{w}$; f was supplied afterwards, but given explicitly. In our application, we can easily store all of w, but f is given implicitly and is expensive to compute. Priority sampling allows us to compute f_i only for $i \in S_N$.
- **The samples can be nested:** The random draw establishes a permutation of the elements such that S_N consists of the first N, regardless of N.

The cited papers [10,30,31] consider only the binary case $f \in \{0,1\}^I$. However, all properties except the confidence bounds generalize to $f \in [0,1]^I$. We use the confidence bounds even though they are only conjectured.

Combining Priority Sampling with Solution Pools. We estimate $LR(\bar{\pi}) = \sum_{k\in\mathcal{B}} LR^k(\bar{\pi}) - \frac{1}{\bar{\pi}_0}\bar{\pi}_R b$ by sampling the blocks, so $I = \mathcal{B}$. Given any pool of solutions P^k for block k, $\overline{LR}^k(\bar{\pi}) := \min_{z^k\in P^k} c^k(\bar{\pi})z^k$ is an upper bound on $LR^k(\bar{\pi})$. Two solutions are readily available: z^k and the $\hat{z}^k$ we saved in step 10. Let $w_k := \overline{LR}^k(\bar{\pi})$, $f_k := LR^k(\bar{\pi})/\overline{LR}^k(\bar{\pi})$, and $\xi = 5\%$, so the quantity to estimate is

$$\sum_{k\in I} f_k w_k = \sum_{k\in\mathcal{B}} \frac{LR^k(\bar{\pi})}{\overline{LR}^k(\bar{\pi})}\overline{LR}^k(\bar{\pi}) = \sum_{k\in\mathcal{B}} LR^k(\bar{\pi}), \tag{5}$$

We always feed the priority sampling routine a target lower bound T. For $\ell = 0, s, 2s, \ldots$ we compute $LR^k(\bar{\pi})$ for the s new blocks in S_ℓ and check whether $UC(\xi) < T + \frac{1}{\bar{\pi}_0}\bar{\pi}_R b$. If so, we predict that $LR(\bar{\pi}) < T$, and exit without a bound. Otherwise we continue with the next chunk of s blocks. Eventually, we either exit with no bound or compute $LR(\bar{\pi}) \geq T$.

Supposing $LR(\bar{\pi}) \geq T$, then each computation of $UC(\xi)$ nominally causes us to mistakenly terminate early (a false-positive error) with a distinct probability of ξ. This suggests that the overall false-positive rate for step 15 could be higher than ξ. However, these errors are highly correlated, and the empirical false positive rate is precisely zero!

We also leverage our block heuristics in step 15. When UB $= \infty$, we call routine WeakLB, in which we solve blocks using the dual heuristic to get a lower bound, possibly weaker than $LR(\bar{\pi})$, and use an aggressive target $T >$ LB to encourage either an early exit or a substantial increment to LB. Once UB $< \infty$, we instead call StrongLB with $T = \text{UB}/(1+\epsilon)$, hoping to terminate. In this case, we use our block heuristics to reduce the number of (very expensive) exact block solves required. We first use the primal heuristic to generate a tighter upper bound on $LR(\bar{\pi})$, exiting early if $UC(\xi) < T$. Then we run the dual heuristic for all blocks; whenever it matches the primal heuristic, they both equal $LR^k(\bar{\pi})$. Then we exactly solve the remaining blocks to close their gaps, using the confidence bounds to exit early as appropriate. The dual heuristic already gives a (weak) bound, and the exact block solves strengthen it as we go along. In this case, StrongLB returns a bound even when it exits early.

Computational Results for Priority Sampling. To measure how effectively priority sampling terminates LB passes, each time we ran a sampling LB pass, we also ran the LB pass to completion to determine the true result (but didn't use that value in the run). As a result, we could divide the LB passes into two cases, *useless* LB passes in which the true result was $< T$, and *useful* LB passes, in which the true result was $\geq T$. Among the 360 runs in this experiment, 264 terminated as a result of a useful StrongLB pass; the others were able to terminate based on a LB obtained from a previous useful WeakLB pass or useless StrongLB pass that returned a bound.

For a useless pass, we measure the effectiveness by how many block solves were avoided by terminating the pass early. Priority sampling avoided 87.4% of the block solves in WeakLB, 69.8% in the primal heuristic portion of StrongLB, and 94.8% in the (very expensive) exact portion of StrongLB.

Table 2. Performance of block heuristics. "Average error" measures relative error between the heuristic solution and the optimum. "Fraction opt" and "Fraction within 1%" are the fraction of the solutions with relative error at most 10^{-6} and 1%, respectively.

	primal heuristic			dual heuristic		
	Tiscali	Sprint	Ebone	Tiscali	Sprint	Ebone
average error	0.19%	0.30%	0.36%	0.12%	0.24%	0.13%
fraction opt	82.3%	76.8%	77.8%	78.8%	68.2%	77.0%
fraction within 1%	93.5%	90.3%	88.6%	96.3%	91.6%	95.8%
heuristic time/block	49 μs	31 μs	19 μs	85 μs	57 μs	35 μs
CPLEX time/block	3384 μs	1305 μs	610 μs	2499 μs	930 μs	405 μs

For useful LB passes, the danger is incorrectly terminating the pass. Surprisingly, for the 1915 useful LB passes in this experiment, *none* were incorrectly shortcut. This is strong empirical evidence that the worst-case error bounds above are extremely pessimistic. Even using $\xi := 50\%$ would have caused only 18 passes to be incorrectly shortcut!

5.2 Block Heuristics

For block iterations, we need not actually compute a gradient direction; it suffices to compute a search direction that improves the potential Φ^δ. Thus, a fast primal heuristic can create significant savings. The block solves in LB passes need not be exact either, so we use a dual heuristic. In addition, when the primal heuristic fails to find an improving direction, we make a second attempt, using the greedy dual heuristic to provide an alternate warm start for the primal heuristic.

Table 2 illustrates the performance of the block heuristics, split by network since that determines the size of the UFL instances. The results are for 1 random seed, and are arithmetic means for the instances on that network in the testbed. Our heuristics are 11x to 70x faster than CPLEX and find optimal solutions in the majority of cases, with small error otherwise. While the detail is not shown here, the small errors by the heuristic solutions cause a modest increase in the number of passes. Specifically, when compared to the number of passes using exact block solutions, our primal and dual heuristics respectively lead to 16% and 1% more passes on average.

5.3 Block Shuffling

Bienstock's code uses round-robin passes [4]. We found that shuffling the blocks randomly after each pass improves iteration performance sharply. We compare our strategy with the following three static block ordering strategies: sorting videos by increasing or decreasing total request volume, or shuffling them randomly into a fixed order. All three require >40x more passes (45.2, 54.9, and 46.5, respectively) than our main code. Moreover, the demand-sorted versions failed to converge in some instances. We do not understand why the effect of shuffling is so dramatic.

Table 3. Varying chunk size and line search method

	bundled τ search			sequential τ search					
chunk size	12	24	48	12	48	96	120	192	240
thread loading	0.64	0.72	0.79	0.63	0.78	0.84	0.85	0.87	0.88
average τ	0.53	0.49	0.46	0.49	0.49	0.48	0.48	0.46	0.45
no. passes	9.25	10.28	11.52	7.26	7.29	7.37	7.45	7.67	7.97

5.4 Chunking

Our code groups s blocks into each chunk. Instead of solving the blocks sequentially as written in Algorithm 1, we solve them in parallel. Larger chunks enable better parallelism. The tradeoff is that block k cannot react to the change in $c^k(\pi^\delta(z))$ as other blocks in the same chunk update z in step 9. Another relevant aspect is how to perform line search for a given chunk. One way is to bundle all blocks in the chunk, select a single step size τ, and move them all by τ. Another way is to compute τ^k and update z^k sequentially for each block k, so that each step size adapts to the ones taken before, although the step directions do not. Step 9 actually updates two sets of values: $r_i(z)$ and z. Our code updates the former sequentially, so the next τ^k can benefit, but does the latter in parallel after computing all τ^k. The blocks within the chunk are visited in random order, according to σ from step 4.

Table 3 shows that bundled τ search achieves better thread-loading as s increases (defined as average utilization of each thread during parallel block solves), but the average step size drops noticeably, leading to more passes and poorer overall performance, even for $s = 48$. Sequential τ search shows the same trend for thread-loading, with the benefit tailing off by $s = 120$, but the decrease in step size and increase in passes are modest through $s = 120$, the value used in our main code. Empirically, the slightly outdated search direction does not significantly worsen the iteration counts until s is quite large.

5.5 Smoothing the Duals

Empirically, using the smoothed duals $\bar{\pi}$ instead of $\pi^\delta(z)$ causes the sequence of LBs computed to be both stronger and nearly monotone. This lessens the penalty for incorrectly aborting an LB pass, freeing us to be more aggressive about our shortcuts. For 85.2% of the passes, the LB we would obtain by disabling shortcutting and calling WeakLB exceeds the LB from the previous pass, and for 82.0% of the passes the LB is the largest we have seen so far. Compared to using $\pi^\delta(z)$ for the LB pass, using $\bar{\pi}$ results in 19% fewer passes on average. Using $\bar{\pi}$ for primal block iterations results in an average 19% further reduction in passes. We discovered this curious last item by mistake and cannot motivate or explain it.

6 Summary and Future Work

We implemented the EPF framework for approximate linear programming, and applied it to LPs that optimize content placement for a VoD service. We describe design choices and innovations that led to significant, sometimes dramatic, improvements in the performance of the code on our testbed. It would be interesting to learn whether these techniques are equally effective when applying the EPF method in other domains.

Other experimenters [4,8,14,21] have reported a better experimental iteration dependence on ϵ than the $O(\epsilon^{-2})$ predicted by theory. It would be interesting to see how our code compares.

We have discovered a method for solving our UFL block LPs via the simplex method, while handling all but $O(n)$ of the $O(n^2)$ variables and constraints combinatorially. Our method bears a resemblance to Wunderling's kernel simplex method [32]. In future work, we intend to implement this method and use it as a replacement for CPLEX in the few cases where we require an exact block solver.

Acknowledgments. We thank Dan Bienstock, Cliff Stein, David Shmoys, Jochen Könemann and David J. Phillips for useful discussions, and especially Bienstock for providing us with his EPF code for comparison. We also thank Mikkel Thorup for showing us how to generalize priority sampling to make it apply in our scenario.

References

1. Applegate, D., Archer, A., Gopalakrishnan, V., Lee, S., Ramakrishnan, K.K.: Optimal content placement for a large-scale VoD system. In: CoNEXT, pp. 4:1–4:12 (2010)
2. Arora, S., Hazan, E., Kale, S.: The multiplicative weights update method: a meta-algorithm and applications. Theor. Comput. 8, 121–164 (2012)
3. Batra, G., Garg, N., Gupta, G.: Heuristic Improvements for Computing Maximum Multicommodity Flow and Minimum Multicut. In: Brodal, G.S., Leonardi, S. (eds.) ESA 2005. LNCS, vol. 3669, pp. 35–46. Springer, Heidelberg (2005)
4. Bienstock, D.: Potential Function Methods for Approximately Solving Linear Programming Problems: Theory and Practice. Kluwer, Boston (2002)
5. Bienstock, D.: Personal communication (2011)
6. Bienstock, D., Iyengar, G.: Approximating fractional packings and coverings in $O(1/\epsilon)$ iterations. SIAM J. Comput. 35(4), 825–854 (2006)
7. Bienstock, D., Zuckerberg, M.: Solving LP Relaxations of Large-Scale Precedence Constrained Problems. In: Eisenbrand, F., Shepherd, F.B. (eds.) IPCO 2010. LNCS, vol. 6080, pp. 1–14. Springer, Heidelberg (2010)
8. Borger, J.M., Kang, T.S., Klein, P.N.: Approximating concurrent flow with unit demands and capacities: an implementation. In: Johnson, D.S., McGeoch, C.C. (eds.) Network Flows and Matching: First DIMACS Implementation Challenge, pp. 371–386. American Mathematical Society (1993)
9. Charikar, M., Guha, S.: Improved combinatorial algorithms for facility location problems. SIAM J. Comput. 34(4), 803–824 (2005)
10. Duffield, N.G., Lund, C., Thorup, M.: Priority sampling for estimation of arbitrary subset sums. J. ACM 54(6) (2007)

11. Fleischer, L.: Approximating fractional multicommodity flow independent of the number of commodities. SIAM J. Discrete Math. 13(4), 505–520 (2000)
12. Fratta, L., Gerla, M., Kleinrock, L.: The flow deviation method: An approach to store-and-forward communication network design. Networks 3(2), 97–133 (1973)
13. Garg, N., Könemann, J.: Faster and simpler algorithms for multicommodity flow and other fractional packing problems. SIAM J. Comput. 37(2), 630–652 (2007)
14. Goldberg, A.V., Oldham, J.D., Plotkin, S., Stein, C.: An Implementation of a Combinatorial Approximation Algorithm for Minimum-Cost Multicommodity Flow. In: Bixby, R.E., Boyd, E.A., Ríos-Mercado, R.Z. (eds.) IPCO 1998. LNCS, vol. 1412, pp. 338–352. Springer, Heidelberg (1998)
15. Gondzio, J.: Interior point methods 25 years later. Eur. J. Oper. Res. 218, 587–601 (2012)
16. Grigoriadis, M.D., Khachiyan, L.G.: Fast approximation schemes for convex programs with many blocks and coupling constraints. SIAM J. Optimiz. 4(1), 86–107 (1994)
17. Grigoriadis, M.D., Khachiyan, L.G.: An exponential-function reduction method for block-angular convex programs. Networks 26, 59–68 (1995)
18. Jang, Y.: Development and implementation of heuristic algorithms for multicommodity flow problems. PhD thesis, Columbia University (1996)
19. Klein, P.N., Young, N.: On the Number of Iterations for Dantzig-Wolfe Optimization and Packing-Covering Approximation Algorithms. In: Cornuéjols, G., Burkard, R.E., Woeginger, G.J. (eds.) IPCO 1999. LNCS, vol. 1610, pp. 320–327. Springer, Heidelberg (1999)
20. Koufogiannakis, C., Young, N.E.: Beating simplex for fractional packing and covering linear programs. In: FOCS, pp. 494–504 (2007)
21. Leong, T., Shor, P., Stein, C.: Implementation of a combinatorial multicommodity flow algorithm. In: Johnson, D.S., McGeoch, C.C. (eds.) Network Flows and Matching: First DIMACS Implementation Challenge, pp. 387–406. American Mathematical Society (1993)
22. Mądry, A.: Faster approximation schemes for fractional multicommodity flow problems via dynamic graph algorithms. In: STOC, pp. 121–130 (2010)
23. Müller, D., Radke, K., Vygen, J.: Faster min-max resource sharing in theory and practice. Math. Program. Comput. 3, 1–35 (2011)
24. Nesterov, Y.: Smooth minimization of non-smooth functions. Math. Program. Ser. A 103, 127–152 (2005)
25. Plotkin, S.A., Shmoys, D.B., Tardos, É.: Fast approximation algorithms for fractional packing and covering problems. Math. Oper. Res. 20, 257–301 (1995)
26. Radzik, T.: Fast deterministic approximation for the multicommodity flow problem. Math. Program. 77, 43–58 (1997)
27. Radzik, T.: Experimental study of a solution method for the multicommodity flow problem. In: ALENEX 2000, pp. 79–102 (2000)
28. Shahrokhi, F., Matula, D.W.: The maximum concurrent flow problem. J. ACM 37(2), 318–334 (1990)
29. Spring, N., Mahajan, R., Wetherall, D.: Measuring ISP topologies with Rocketfuel. In: SIGCOMM, pp. 133–145 (2002)
30. Szegedy, M.: The DLT priority sampling is essentially optimal. In: STOC, pp. 150–158 (2006)
31. Thorup, M.: Confidence intervals for priority sampling. In: SIGMETRICS, pp. 252–263 (2006)
32. Wunderling, R.: The kernel simplex method. Talk at ISMP (August 2012)

Equivariant Perturbation in Gomory and Johnson's Infinite Group Problem: II. The Unimodular Two-Dimensional Case

Amitabh Basu, Robert Hildebrand, and Matthias Köppe

Dept. of Mathematics, University of California, Davis
{abasu,rhildebrand,mkoeppe}@math.ucdavis.edu

Abstract. We give an algorithm for testing the extremality of a large class of minimal valid functions for the two-dimensional infinite group problem.

1 Introduction

1.1 The Group Problem

Gomory's *group problem* [7] is a central object in the study of strong cutting planes for integer linear optimization problems. One considers an abelian (not necessarily finite) group G, written additively, and studies the set of functions $s\colon G \to \mathbb{R}$ satisfying the following constraints:

$$\sum_{\mathbf{r}\in G} \mathbf{r}\, s(\mathbf{r}) \in \mathbf{f} + S \tag{IR}$$
$$s(\mathbf{r}) \in \mathbb{Z}_+ \quad \text{for all } \mathbf{r} \in G$$
$$s \text{ has finite support,}$$

where $\mathbf{f}$ is a given element in G, and S is a subgroup of G; so $\mathbf{f} + S$ is the coset containing the element $\mathbf{f}$. We will be concerned with the so-called *infinite group problem* [8,9], where $G = \mathbb{R}^k$ is taken to be the group of real k-vectors under addition, and $S = \mathbb{Z}^k$ is the subgroup of the integer vectors. We are interested in studying the convex hull $R_{\mathbf{f}}(G, S)$ of all functions satisfying the constraints in (IR). Observe that $R_{\mathbf{f}}(G, S)$ is a convex subset of the infinite-dimensional vector space $\mathcal{V}$ of functions $s\colon G \to \mathbb{R}$ with finite support.

Any linear inequality in $\mathcal{V}$ is given by $\sum_{\mathbf{r}\in G} \pi(\mathbf{r})s(\mathbf{r}) \geq \alpha$ where π is a function $\pi\colon G \to \mathbb{R}$ and $\alpha \in \mathbb{R}$. The left-hand side of the inequality is a finite sum because s has finite support. Such an inequality is called a *valid inequality* for $R_{\mathbf{f}}(G, S)$ if $\sum_{\mathbf{r}\in G} \pi(\mathbf{r})s(\mathbf{r}) \geq \alpha$ for all $s \in R_{\mathbf{f}}(G, S)$. It is customary to concentrate on valid inequalities with $\pi \geq 0$; then we can choose, after a scaling, $\alpha = 1$. Thus, we only focus on valid inequalities of the form $\sum_{\mathbf{r}\in G} \pi(\mathbf{r})s(\mathbf{r}) \geq 1$ with $\pi \geq 0$. Such functions π will be termed *valid functions* for $R_{\mathbf{f}}(G, S)$.

A valid function π for $R_{\mathbf{f}}(G, S)$ is said to be *minimal* for $R_{\mathbf{f}}(G, S)$ if there is no valid function $\pi' \neq \pi$ such that $\pi'(\mathbf{r}) \leq \pi(\mathbf{r})$ for all $\mathbf{r} \in G$. For every valid

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 62–73, 2013.

function π for $R_{\mathbf{f}}(G,S)$, there exists a minimal valid function π' such that $\pi' \leq \pi$ (cf. [2]), and thus non-minimal valid functions are redundant in the description of $R_{\mathbf{f}}(G,S)$. Minimal functions for $R_{\mathbf{f}}(G,S)$ were characterized by Gomory for finite groups G in [7], and later for $R_{\mathbf{f}}(\mathbb{R},\mathbb{Z})$ by Gomory and Johnson [8]. We state these results in a unified notation in the following theorem.

A function $\pi\colon G \to \mathbb{R}$ is *subadditive* if $\pi(\mathbf{x}+\mathbf{y}) \leq \pi(\mathbf{x})+\pi(\mathbf{y})$ for all $\mathbf{x},\mathbf{y} \in G$. We say that π is *symmetric* if $\pi(\mathbf{x}) + \pi(\mathbf{f}-\mathbf{x}) = 1$ for all $\mathbf{x} \in G$.

Theorem 1.1 (Gomory and Johnson [8]). *Let $\pi\colon G \to \mathbb{R}$ be a non-negative function. Then π is a minimal valid function for $R_{\mathbf{f}}(G,S)$ if and only if $\pi(\mathbf{z}) = 0$ for all $\mathbf{z} \in S$, π is subadditive, and π satisfies the symmetry condition. (The first two conditions imply that π periodic with respect to S, that is, $\pi(\mathbf{x}) = \pi(\mathbf{x}+\mathbf{z})$ for all $\mathbf{z} \in S$.)*

Due to this periodicity, we will study functions on $\mathbb{R}^2/\mathbb{Z}^2$ when investigating the infinite group problem with $G = \mathbb{R}^2$ and $S = \mathbb{Z}^2$. We will use $\oplus$ and $\ominus$ to denote vector addition and subtraction modulo 1, respectively. We use the same notation for pointwise sums and differences of sets.

1.2 Characterization of Extreme Valid Functions

A stronger notion is that of an *extreme function*. A valid function π is *extreme* for $R_{\mathbf{f}}(G,S)$ if it cannot be written as a convex combination of two other valid functions for $R_{\mathbf{f}}(G,S)$, i.e., $\pi = \frac{1}{2}\pi_1 + \frac{1}{2}\pi_2$ implies $\pi = \pi_1 = \pi_2$. Extreme functions are minimal. A tight characterization of extreme functions for $R_{\mathbf{f}}(\mathbb{R}^k,\mathbb{Z}^k)$ has eluded researchers for the past four decades now, however, various specific sufficient conditions for guaranteeing extremality [2,3,6,5,4,10] have been proposed. The standard technique for showing extremality is as follows. Suppose that $\pi = \frac{1}{2}\pi^1 + \frac{1}{2}\pi^2$, where π^1, π^2 are other (minimal) valid functions. One then studies the set of additivity relations $E(\pi) = \{\, (\mathbf{x},\mathbf{y}) \mid \pi(\mathbf{x})+\pi(\mathbf{y}) = \pi(\mathbf{x}\oplus\mathbf{y}) \,\}$. By subadditivity, it follows that $E(\pi) \subseteq E(\pi^1), E(\pi^2)$. Then a lemma of *real analysis*, such as the so-called Interval Lemma of Gomory and Johnson (or one of its variants), is used to deduce affine linearity properties of π^1, π^2 using the additivity relations. This is followed by a *linear algebra* argument to show that π is the unique solution to a finite-dimensional system of equations, implying $\pi = \pi^1 = \pi^2$, and thus establishing the extremality of π.

Surprisingly, the *arithmetic* (number-theoretic) aspect of the problem has been largely overlooked, even though it is at the core of the theory of the closely related *finite* group problem. In [1], the authors showed that this aspect is the key for completing the classification of extreme functions:

Theorem 1.2 (Theorem 1.3 in [1]). *Consider the following problem.*

> *Given a minimal valid function π for $R_f(\mathbb{R},\mathbb{Z})$ that is piecewise linear with a set of rational breakpoints with the least common denominator q, decide if π is extreme or not.*

There exists an algorithm for this problem that takes a number of elementary operations over the reals that is bounded by a polynomial in q.

To capture the relevant arithmetics of the problem, the authors studied sets of additivity relations of the form $\pi(\mathbf{t}^i) + \pi(\mathbf{y}) = \pi(\mathbf{t}^i + \mathbf{y})$ and $\pi(\mathbf{x}) + \pi(\mathbf{r}^i - \mathbf{x}) = \pi(\mathbf{r}^i)$, where the points $\mathbf{t}^i$ and $\mathbf{r}^i$ are certain breakpoints of the function π. This is an important departure from the previous literature, which only uses additivity relations over non-degenerate intervals. The arithmetic nature of the problem comes into focus when one realizes that isolated additivity relations over single points are also important for studying extremality. These isolated additivity relations give rise to a subgroup of the group $\mathrm{Aff}(\mathbb{R}^k)$ of invertible affine linear transformations of $\mathbb{R}^k$ as follows.

For a point $\mathbf{r} \in \mathbb{R}^k$, define the *reflection* $\rho_{\mathbf{r}} \colon \mathbb{R}^k \to \mathbb{R}^k$, $\mathbf{x} \mapsto \mathbf{r} - \mathbf{x}$. For a vector $\mathbf{t} \in \mathbb{R}^k$, define the *translation* $\tau_{\mathbf{t}} \colon \mathbb{R}^k \to \mathbb{R}$, $\mathbf{x} \mapsto \mathbf{x} + \mathbf{t}$. Given a set R of points and a set U of vectors, we define the *reflection group* $\Gamma = \langle\, \rho_{\mathbf{r}}, \tau_{\mathbf{t}} \mid \mathbf{r} \in R,\, \mathbf{t} \in U \,\rangle$. If we assign a *character* $\chi(\rho_{\mathbf{r}}) = -1$ to every reflection and $\chi(\tau_{\mathbf{t}}) = +1$ to every translation, then it can be shown that this extends to a *group character* of Γ, that is, a group homomorphism $\chi \colon \Gamma \to \mathbb{C}^\times$.

Definition 1.3. *A function $\psi \colon \mathbb{R}^k \to \mathbb{R}$ is called* Γ-equivariant *if it satisfies the* equivariance formula

$$\psi(\gamma(\mathbf{x})) = \chi(\gamma)\psi(\mathbf{x}) \quad \textit{for } \mathbf{x} \in \mathbb{R} \textit{ and } \gamma \in \Gamma. \tag{1}$$

For $k = 1$, the natural action of the reflection group Γ on the set of intervals delimited by the elements of $\frac{1}{q}\mathbb{Z}$ transfers the affine linearity established by the Interval Lemma on some interval I to all intervals in the orbit $\Gamma(I)$. When this establishes affine linearity of π^1, π^2 on all intervals where π is affinely linear, one proceeds with finite-dimensional linear algebra to decide extremality of π. Otherwise, there is a way to perturb π to construct distinct minimal valid functions $\pi^1 = \pi + \bar{\pi}$ and $\pi^2 = \pi - \bar{\pi}$, using any sufficiently small Γ-equivariant perturbation function $\bar{\pi}$ modified by restriction to a certain family of intervals. This is the main idea in [1] for proving Theorem 1.2.

1.3 Contributions of the Paper

We continue the program of [1]. We study a class of minimal functions π of the two-dimensional infinite group problem ($k = 2$). Let q be a positive integer. Consider the arrangement $\mathcal{H}_q$ of all hyperplanes (lines) of the form $(0,1) \cdot \mathbf{x} = b$, $(1,0) \cdot \mathbf{x} = b$, and $(1,1) \cdot \mathbf{x} = b$, where $b \in \frac{1}{q}\mathbb{Z}$. The complement of the arrangement $\mathcal{H}_q$ consists of two-dimensional cells, whose closures are the triangles $T_0 = \frac{1}{q}\,\mathrm{conv}(\{\binom{0}{0}, \binom{1}{0}, \binom{0}{1}\})$ and $T_1 = \frac{1}{q}\,\mathrm{conv}(\{\binom{1}{0}, \binom{0}{1}, \binom{1}{1}\})$ and their translates by elements of the lattice $\frac{1}{q}\mathbb{Z}^2$. We denote by $\mathcal{P}_q$ the collection of these triangles and the vertices and edges that arise as intersections of the triangles. Thus $\mathcal{P}_q$ is a polyhedral complex that is a triangulation of the space $\mathbb{R}^2$. Within the polyhedral complex $\mathcal{P}_q$, let $\mathcal{P}_{q,0}$ be the set of 0-faces (vertices), $\mathcal{P}_{q,1}$ be the set

of 1-faces (edges), and $\mathcal{P}_{q,2}$ be the set of 2-faces (triangles). The sets of diagonal, vertical, and horizontal edges will be denoted by $\mathcal{P}_{q,\diagdown}$, $\mathcal{P}_{q,|}$, and $\mathcal{P}_{q,-}$, respectively. Observe that $\mathcal{P}_q$ is periodic with respect to $\mathbb{Z}^2$, and so we will restrict our attention to the corresponding triangulation of $\mathbb{R}^2/\mathbb{Z}^2$; we will continue to denote this by $\mathcal{P}_q$.

We call a function $\pi\colon \mathbb{R}^2/\mathbb{Z}^2 \to \mathbb{R}$ *continuous piecewise linear over* $\mathcal{P}_q$ if it is an affine linear function on each of the triangles of $\mathcal{P}_q$. We introduce the following notation. For every $I \in \mathcal{P}_q$, the restriction $\pi|_I$ is an affine function, that is $\pi|_I(\mathbf{x}) = \mathbf{m}_I \cdot \mathbf{x} + b_I$ for some $\mathbf{m}_I \in \mathbb{R}^2$, $b_I \in \mathbb{R}$. We abbreviate $\pi|_I$ as π_I. The construction of $\mathcal{P}_q$ has convenient properties such as the following (we omit the proof from this extended abstract, which relies on strong unimodularity properties of $\mathcal{P}_q$).

Lemma 1.4. *Let $I, J \in \mathcal{P}_q$. Then $I \oplus J$ and $I \ominus J$ are unions of faces in $\mathcal{P}_q$.*

This allows one to give a finite combinatorial representation of the set $E(\pi)$ using the faces of $\mathcal{P}_q$; this extends a technique in [1]. For faces $I, J, K \in \mathcal{P}_q$, let

$$F(I,J,K) = \{\, (\mathbf{x},\mathbf{y}) \in \mathbb{R}^2 \times \mathbb{R}^2 \mid \mathbf{x} \in I,\, \mathbf{y} \in J,\, \mathbf{x} \oplus \mathbf{y} \in K \,\}.$$

Definition 1.5. *For $I, J, K \in \mathcal{P}_q \backslash \{\emptyset\}$, we say (I, J, K) is a* valid triple *provided that the following occur: $K \subseteq I \oplus J$, $I \subseteq K \oplus -J$, $J \subseteq K \oplus -I$.*

Let $E(\pi, \mathcal{P}_q)$ denote the set of valid triples (I, J, K) such that $\pi(\mathbf{x}) + \pi(\mathbf{y}) = \pi(\mathbf{x} \oplus \mathbf{y})$ for all $(\mathbf{x}, \mathbf{y}) \in F(I, J, K)$. $E(\pi, \mathcal{P}_q)$ is partially ordered by letting $(I, J, K) \leq (I', J', K')$ if and only if $I \subseteq I'$, $J \subseteq J'$, and $K \subseteq K'$. Let $E_{\max}(\pi, \mathcal{P}_q)$ be the set of all maximal valid triples of the poset $E(\pi, \mathcal{P}_q)$. Then it can be shown that $E(\pi)$ is exactly covered by the sets $F(I, J, K)$ for the maximal valid triples $(I, J, K) \in E_{\max}(\pi, \mathcal{P}_q)$ (we omit the details).

We will restrict ourselves to a setting without maximal valid triples that include horizontal or vertical edges.

Definition 1.6. *A continuous piecewise linear function π on $\mathcal{P}_q$ is called* diagonally constrained *if whenever $(I, J, K) \in E_{\max}(\pi, \mathcal{P}_q)$, then $I, J, K \in \mathcal{P}_{q,0} \cup \mathcal{P}_{q,\diagdown} \cup \mathcal{P}_{q,2}$.*

Due to the strong unimodularity properties of $\mathcal{P}_q$, we can easily check if $(I, J, K) \in E(\pi, \mathcal{P}_q)$ is a valid triple, for $I, J, K \in \mathcal{P}_q$, by just using the vertices of I, J, K. Using this test, by enumeration on triples from $\mathcal{P}_q$, we can determine if a function is diagonally constrained.

Remark 1.7. Given a piecewise linear continuous valid function $\zeta\colon \mathbb{R} \to \mathbb{R}$ for the one-dimensional infinite group problem, Dey–Richard [4, Construction 6.1] consider the function $\kappa\colon \mathbb{R}^2 \to \mathbb{R}$, $\kappa(\mathbf{x}) = \zeta(\mathbf{1}\cdot\mathbf{x})$, where $\mathbf{1} = (1, 1)$, and show that κ is minimal and extreme if and only if ζ is minimal and extreme, respectively. If ζ has rational breakpoints in $\frac{1}{q}\mathbb{Z}$, then κ belongs to our class of diagonally constrained continuous piecewise linear functions over $\mathcal{P}_q$. However there do exist diagonally constrained functions that cannot be obtained in this manner. See Figure 1 for an example.

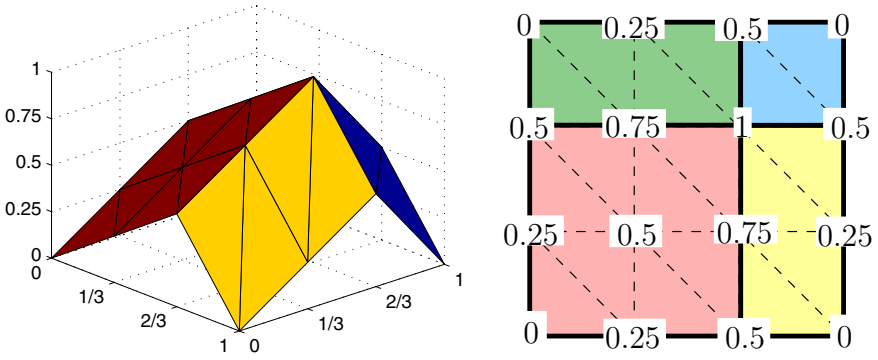


Fig. 1. This minimal, continuous, piecewise linear function over $\mathcal{P}_3$ is diagonally constrained, which was confirmed computationally. On the left is the 3-dimensional plot of the function. On the right is the complex $\mathcal{P}_3$, colored according to slopes to match the 3-dimensional plot, and decorated with function values at each vertex of $\mathcal{P}_3$.

We prove the following main theorem.

Theorem 1.8. *Consider the following problem.*

> *Given a minimal valid function π for $R_{\mathbf{f}}(\mathbb{R}^2, \mathbb{Z}^2)$ that is piecewise linear continuous on $\mathcal{P}_q$ and diagonally constrained, decide if π is extreme.*

There exists an algorithm for this problem that takes a number of elementary operations over the reals that is bounded by a polynomial in q.

We require the input function to the above algorithm to be a minimal function. It is a straightforward matter to check the minimality of a piecewise linear function in the light of Theorem 1.1.

Theorem 1.9 (Minimality test). *A function $\pi\colon \mathbb{R}^2/\mathbb{Z}^2 \to \mathbb{R}$ that is continuous piecewise linear over $\mathcal{P}_q$ is minimal if and only if*

(i) $\pi(\mathbf{0}) = 0$,
(ii) $\pi(\mathbf{x}) + \pi(\mathbf{y}) \geq \pi(\mathbf{x} \oplus \mathbf{y})$ *for all* $\mathbf{x}, \mathbf{y} \in \frac{1}{q}\mathbb{Z}^2 \cap [0,1)^2$,
(iii) $\pi(\mathbf{x}) + \pi(\mathbf{f} \ominus \mathbf{x}) = 1$ *for all* $\mathbf{x} \in \frac{1}{q}\mathbb{Z}^2 \cap [0,1)^2$.

As a direct corollary of the proof of Theorem 1.8, we obtain the following result relating the finite and infinite group problems.

Theorem 1.10. *Let π be a minimal continuous piecewise linear function over $\mathcal{P}_q$ that is diagonally constrained. Then π is extreme for $R_{\mathbf{f}}(\mathbb{R}^2, \mathbb{Z}^2)$ if and only if the restriction $\pi\big|_{\frac{1}{4q}\mathbb{Z}^2}$ is extreme for $R_{\mathbf{f}}(\frac{1}{4q}\mathbb{Z}^2, \mathbb{Z}^2)$.*

We conjecture that the hypothesis on π being diagonally constrained can be removed.

2 Real Analysis Lemmas

For any element $\mathbf{x} \in \mathbb{R}^k$, $k \geq 1$, $|\mathbf{x}|$ will denote the standard Euclidean norm. The proofs of the following results are omitted from this extended abstract.

Theorem 2.1. *If $\pi\colon \mathbb{R}^k \to \mathbb{R}$ is a minimal valid function, and $\pi = \frac{1}{2}\pi^1 + \frac{1}{2}\pi^2$, where π^1, π^2 are valid functions, then π^1, π^2 are both minimal. Moreover, if $\limsup_{\mathbf{h}\to 0} \frac{|\pi(\mathbf{h})|}{|\mathbf{h}|} < \infty$, then this condition also holds for π^1 and π^2. This implies that π, π^1 and π^2 are all Lipschitz continuous.*

Lemma 2.2. *Suppose π is a continuous function and let $(I, J, K) \in E(\pi, \mathcal{P}_q)$ be a valid triple of triangles, i.e., $I, J, K \in \mathcal{P}_{q,2}$. Then π is affine in I, J, K with the same gradient.*

Lemma 2.3. *Suppose π is a continuous function and let $(I, J, K) \in E(\pi, \mathcal{P}_q)$ where $I, J, K \in \mathcal{P}_{q,\diagdown} \cup \mathcal{P}_{q,2}$. Then π is affine in the diagonal direction in I, J, K, i.e., there exists $c \in \mathbb{R}$ such that such that $\pi(\mathbf{x} + \lambda \left(\begin{smallmatrix}-1\\1\end{smallmatrix}\right)) = \pi(\mathbf{x}) + c \cdot \lambda$ for all $\mathbf{x} \in I$ (resp., $\mathbf{x} \in J$, $\mathbf{x} \in K$) and $\lambda \in \mathbb{R}$ such that $\mathbf{x} + \lambda \left(\begin{smallmatrix}-1\\1\end{smallmatrix}\right) \in I$ (resp., $\mathbf{x} + \lambda \left(\begin{smallmatrix}-1\\1\end{smallmatrix}\right) \in J$, $\mathbf{x} + \lambda \left(\begin{smallmatrix}-1\\1\end{smallmatrix}\right) \in K$).*

Lemma 2.4. *Let $I, J \in \mathcal{P}_{q,2}$ be triangles such that $I \cap J \in \mathcal{P}_{q,|} \cup \mathcal{P}_{q,-}$. Let π be a continuous function defined on $I \cup J$ satisfying the following properties:*

(i) π is affine on I.
(ii) There exists $c \in \mathbb{R}$ such that $\pi(\mathbf{x} + \lambda \left(\begin{smallmatrix}-1\\1\end{smallmatrix}\right)) = \pi(\mathbf{x}) + c \cdot \lambda$ for all $\mathbf{x} \in J$ and $\lambda \in \mathbb{R}$ such that $\mathbf{x} + \lambda \left(\begin{smallmatrix}-1\\1\end{smallmatrix}\right) \in J$.

Then π is affine on J.

3 Proof of the Main Results

Let $\partial_{\mathbf{v}}$ denote the directional derivative in the direction of $\mathbf{v}$.

Definition 3.1. *Let π be a minimal valid function.*

(a) *For any $I \in \mathcal{P}_q$, if π is affine in I and if for all valid functions π^1, π^2 such that $\pi = \frac{1}{2}\pi^1 + \frac{1}{2}\pi^2$ we have that π^1, π^2 are affine in I, then we say that π is* affine imposing *in I.*
(b) *For any $I \in \mathcal{P}_q$, if $\partial_{(-1,1)}\pi$ is constant in I and if for all valid functions π^1, π^2 such that $\pi = \frac{1}{2}\pi^1 + \frac{1}{2}\pi^2$ we have that $\partial_{(-1,1)}\pi^1$, $\partial_{(-1,1)}\pi^2$ are constant in I, then we say that π is* diagonally affine imposing *in I.*
(c) *For a collection $\mathcal{P} \subseteq \mathcal{P}_q$, if for all $I \in \mathcal{P}$, π is affine imposing (or diagonally affine imposing) in I, then we say that π is* affine imposing (diagonally affine imposing) *in $\mathcal{P}$.*

We either show that π is affine imposing in $\mathcal{P}_q$ (subsection 3.1) or construct a continuous piecewise linear Γ-equivariant perturbation over $\mathcal{P}_{4q}$ that proves π is not extreme (subsections 3.2 and 3.3), where Γ is an appropriately defined reflection group. If π is affine imposing in $\mathcal{P}_q$, we set up a system of linear equations to decide if π is extreme or not (subsection 3.4). This implies Theorem 1.8 stated in the introduction.

3.1 Imposing Affine Linearity on Faces of $\mathcal{P}_q$

For the remainder of this paper, we will use reflections and translations modulo 1 to compensate for the fact that minimal functions are periodic with period 1. Working modulo 1 is accounted for by applying the translations $\tau_{(1,0)}$ and $\tau_{(0,1)}$ whenever needed. Hence, we define the reflection $\bar{\rho}_{\mathbf{v}}(\mathbf{x}) = \mathbf{v} \ominus \mathbf{x}$ and the translation $\bar{\tau}_{\mathbf{v}}(\mathbf{x}) = \mathbf{v} \oplus \mathbf{x}$. The reflections and translations arise from certain valid triples as follows.

Lemma 3.2. *Suppose (I, J, K) is a valid triple.*

(a) *If $K = \{\mathbf{a}\} \in \mathcal{P}_{q,0}$, then $J = \bar{\rho}_{\mathbf{a}}(I)$.*
(b) *If $J = \{\mathbf{a}\} \in \mathcal{P}_{q,0}$, then $K = \bar{\tau}_{\mathbf{a}}(I)$.*

Proof. Part a. Since $(I, J, \{\mathbf{a}\})$ is a valid triple, for all $\mathbf{x} \in I$, there exists a $\mathbf{y} \in J$ such that $\mathbf{x} \oplus \mathbf{y} = \mathbf{a}$, i.e., $\mathbf{y} = \mathbf{a} \ominus \mathbf{x} \in J$, and therefore $J \supseteq \bar{\rho}_{\mathbf{a}}(I)$. Also, for all $\mathbf{y} \in J$, there exists a $\mathbf{x} \in I$ such that $\mathbf{x} \oplus \mathbf{y} = \mathbf{a}$. Again, $\mathbf{y} = \mathbf{a} \ominus \mathbf{x}$, i.e., $J \subseteq \bar{\rho}_{\mathbf{a}}(I)$. Hence, $J = \bar{\rho}_{\mathbf{a}}(I)$.

Part b. Since $(I, \{\mathbf{a}\}, K)$ is a valid triple and J is a singleton, then for all $\mathbf{x} \in I$, we have $\mathbf{x} \oplus \mathbf{a} \in K$, i.e., $K \supseteq \bar{\tau}_{\mathbf{a}}(I)$. Also, for all $\mathbf{z} \in K$, there exists a $\mathbf{x} \in I$ such that $\mathbf{x} \oplus \mathbf{a} = \mathbf{z}$, i.e., $K \subseteq \bar{\tau}_{\mathbf{a}}(I)$. Hence, $K = \bar{\tau}_{\mathbf{a}}(I)$. □

Let $\mathcal{G} = \mathcal{G}(\mathcal{P}_{q,2}, \mathcal{E})$ be an undirected graph with node set $\mathcal{P}_{q,2}$ and edge set $\mathcal{E} = \mathcal{E}_0 \cup \mathcal{E}_{\diagdown}$ where $\{I, J\} \in \mathcal{E}_0$ (resp., $\{I, J\} \in \mathcal{E}_{\diagdown}$) if and only if for some $K \in \mathcal{P}_{q,0}$ (resp., $K \in \mathcal{P}_{q,\diagdown}$), we have $(I, J, K) \in E(\pi, \mathcal{P}_q)$ or $(I, K, J) \in E(\pi, \mathcal{P}_q)$. For each $I \in \mathcal{P}_{q,2}$, let $\mathcal{G}_I$ be the connected component of $\mathcal{G}$ containing I.

We now consider faces of $\mathcal{P}_{q,2}$ on which we will apply lemmas from section 2.

$$\mathcal{P}^1_{q,2} = \{\, I, J \in \mathcal{P}_{q,2} \mid \exists K \in \mathcal{P}_{q,\diagdown} \text{ with } (I, J, K) \in E(\pi, \mathcal{P}_q) \text{ or } (I, K, J) \in E(\pi, \mathcal{P}_q) \,\},$$

$$\mathcal{P}^2_{q,2} = \{\, I, J, K \in \mathcal{P}_{q,2} \mid (I, J, K) \in E(\pi, \mathcal{P}_q) \,\}.$$

It follows from Lemma 2.2 that π is affine imposing in $\mathcal{P}^2_{q,2}$ and from Lemma 2.3 that π is diagonally affine imposing in $\mathcal{P}^1_{q,2}$.

Faces connected in the graph have related slopes.

Lemma 3.3. *Let $\mathbf{v} \in \mathbb{R}^2$. For $\theta = \pi, \pi^1$, or π^2, if θ is affine in the $\mathbf{v}$ direction in I, i.e., there exists $c \in \mathbb{R}$ such that such that $\pi(\mathbf{x} + \lambda\mathbf{v}) = \pi(\mathbf{x}) + c \cdot \lambda$ for all $\mathbf{x} \in I$ and $\lambda \in \mathbb{R}$ such that $\mathbf{x} + \lambda\mathbf{v} \in I$, and $\{I, J\} \in \mathcal{E}$, then θ is affine in the $\mathbf{v}$ direction in J as well.*

The proof is omitted from this extended abstract.

With this in mind, we define the two sets of faces and any faces connected to them in the graph $\mathcal{G}$,

$$\mathcal{S}^1_{q,2} = \{\, J \in \mathcal{P}_{q,2} \mid J \in \mathcal{G}_I \text{ for some } I \in \mathcal{P}^1_{q,2} \,\},$$

$$\mathcal{S}^2_{q,2} = \{\, J \in \mathcal{P}_{q,2} \mid J \in \mathcal{G}_I \text{ for some } I \in \mathcal{P}^2_{q,2} \,\}.$$

It follows from Lemma 3.3 that π is affine imposing in $\mathcal{S}^2_{q,2}$ and diagonally affine imposing in $\mathcal{S}^1_{q,2}$.

From Lemma 2.4, it follows that if $I \in \mathcal{S}^2_{q,2}$, $J \in \mathcal{S}^1_{q,2}$ and $I \cap J \in \mathcal{P}_{q,|} \cup \mathcal{P}_{q,\diagdown}$, then π is affine imposing in J. Let

$$\bar{\mathcal{S}}_{q,2} = \{K \in \mathcal{G}_I \mid I \in \mathcal{S}^1_{q,2} \text{ and there exists a } J \in \mathcal{S}^2_{q,2} \text{ s.t. } I \cap J \in \mathcal{P}_{q,|} \cup \mathcal{P}_{q,-}\}.$$

Now set $\bar{\mathcal{S}}^2_{q,2} = \mathcal{S}^2_{q,2} \cup \bar{\mathcal{S}}_{q,2}$ and $\bar{\mathcal{S}}^1_{q,2} = \mathcal{S}^1_{q,2} \setminus \bar{\mathcal{S}}_{q,2}$. The following theorem is a consequence of Lemmas 2.2, 2.4, and 3.3.

Theorem 3.4. *If $\bar{\mathcal{S}}^2_{q,2} = \mathcal{P}_{q,2}$, then π is affine imposing in $\mathcal{P}_{q,2}$, and therefore θ is continuous piecewise linear over $\mathcal{P}_q$ for $\theta = \pi^1, \pi^2$.*

3.2 Non-extremality by Two-Dimensional Equivariant Perturbation

In this and the following subsection, we will prove the following result.

Lemma 3.5. *Let π be a minimal, continuous piecewise linear function over $\mathcal{P}_q$ that is diagonally constrained. If $\bar{\mathcal{S}}^2_{q,2} \neq \mathcal{P}_{q,2}$, then π is not extreme.*

In the proof, we will need two different equivariant perturbations that we construct as follows. Let $\Gamma_0 = \langle\, \rho_{\mathbf{g}}, \tau_{\mathbf{g}} \mid \mathbf{g} \in \frac{1}{q}\mathbb{Z}^2 \,\rangle$ be the group generated by reflections and translations corresponding to all possible vertices of $\mathcal{P}_q$. We define the function $\psi\colon \mathbb{R}^2 \to \mathbb{R}$ as a continuous piecewise linear function over $\mathcal{P}_{4q}$ in the following way: let $T_0 = \frac{1}{q}\operatorname{conv}(\{\binom{0}{0}, \binom{1}{0}, \binom{0}{1}\})$, and at all vertices of $\mathcal{P}_{4q}$ that lie in T_0, let ψ take the value 0, except at the interior vertices $\frac{1}{4q}\binom{1}{1}, \frac{1}{4q}\binom{2}{1}, \frac{1}{4q}\binom{1}{2}$, where we assign ψ to have the value 1. Interpolate these values over the restriction of $\mathcal{P}_{4q}$ to T_0, to define ψ on T_0. Since T_0 is a fundamental domain for Γ_0, we can extend ψ to all of $\mathbb{R}^2$ using the equivariance formula (1).

Lemma 3.6. *The function $\psi\colon \mathbb{R}^2 \to \mathbb{R}$ constructed above is well-defined and has the following properties:*

(i) $\psi(\mathbf{g}) = 0$ *for all* $\mathbf{g} \in \frac{1}{q}\mathbb{Z}^2$,
(ii) $\psi(\mathbf{x}) = -\psi(\rho_{\mathbf{g}}(\mathbf{x})) = -\psi(\mathbf{g} - \mathbf{x})$ *for all* $\mathbf{g} \in \frac{1}{q}\mathbb{Z}^2, \mathbf{x} \in [0,1]^2$,
(iii) $\psi(\mathbf{x}) = \psi(\tau_{\mathbf{g}}(\mathbf{x})) = \psi(\mathbf{g} + \mathbf{x})$ *for all* $\mathbf{g} \in \frac{1}{q}\mathbb{Z}^2, \mathbf{x} \in [0,1]^2$,
(iv) ψ *is continuous piecewise linear over* $\mathcal{P}_{4q}$.

Proof. The properties follow directly from the equivariance formula (1). □

It is now convenient to introduce the function $\Delta\pi(\mathbf{x}, \mathbf{y}) = \pi(\mathbf{x}) + \pi(\mathbf{y}) - \pi(\mathbf{x} \oplus \mathbf{y})$, which measures the slack in the subadditivity constraints. Let $\Delta\mathcal{P}_q$ be the polyhedral complex containing all polytopes $F = F(I, J, K)$ where $I, J, K \in \mathcal{P}_q$. Observe that $\Delta\pi|_F$ is affine; if we introduce the function $\Delta\pi_F(\mathbf{x}, \mathbf{y}) = \pi_I(\mathbf{x}) + \pi_J(\mathbf{y}) - \pi_K(\mathbf{x} \oplus \mathbf{y})$ for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^2$, then $\Delta\pi(\mathbf{x}, \mathbf{y}) = \Delta\pi_F(\mathbf{x}, \mathbf{y})$ for all $(\mathbf{x}, \mathbf{y}) \in F$. Furthermore, if (I, J, K) is a valid triple, then $(I, J, K) \in E(\pi, \mathcal{P}_q)$ if and only if $\Delta\pi|_{F(I,J,K)} = 0$. We will use $\operatorname{vert}(F)$ to denote the set of vertices of the polytope F.

Lemma 3.7. *Let $F \in \Delta\mathcal{P}_q$ and let $(\mathbf{x}, \mathbf{y})$ be a vertex of F. Then $\mathbf{x}, \mathbf{y}$ are vertices of the complex $\mathcal{P}_q$, i.e., $\mathbf{x}, \mathbf{y} \in \frac{1}{q}\mathbb{Z}^2$.*

The proof again uses the strong unimodularity properties of $\mathcal{P}_q$ and is omitted from this extended abstract.

Lemma 3.8. *Let π be a minimal, continuous piecewise linear function over $\mathcal{P}_q$ that is diagonally constrained. Suppose there exists $I^* \in \mathcal{P}_{q,2} \setminus (\bar{\mathcal{S}}^2_{q,2} \cup \bar{\mathcal{S}}^1_{q,2})$. Then π is not extreme.*

Proof. Let $R = \bigcup_{J \in \mathcal{G}_{I^*}} \operatorname{int}(J) \subseteq [0,1]^2$. Since R is a union of interiors, it does not contain any points in $\frac{1}{2q}\mathbb{Z}^2$. Let ψ be the Γ_0-equivariant function of Lemma 3.6. Let

$$\epsilon = \min\{\, \Delta\pi_{\hat{F}}(\mathbf{x}, \mathbf{y}) \neq 0 \mid \hat{F} \in \Delta\mathcal{P}_{4q},\ (\mathbf{x}, \mathbf{y}) \in \operatorname{vert}(\hat{F}) \,\},$$

and let $\bar{\pi} = \delta_R \cdot \psi$ where δ_R is the indicator function for the set R. We will show that for

$$\pi^1 = \pi + \tfrac{\epsilon}{3}\bar{\pi}, \quad \pi^2 = \pi - \tfrac{\epsilon}{3}\bar{\pi},$$

that π^1, π^2 are minimal, and therefore valid functions, and hence π is not extreme. We will show this just for π^1 as the proof for π^2 is the same.

Since $\psi(\mathbf{0}) = 0$ and $\psi(\mathbf{f}) = 0$, we see that $\pi^1(\mathbf{0}) = 0$ and $\pi^1(\mathbf{f}) = 1$.

We want to show that π^1 is symmetric and subadditive. In fact, it suffices to prove that π^1 is subadditive (symmetry of π^1 will then follow from the symmetry of π and the fact that $\pi = \frac{1}{2}\pi^1 + \frac{1}{2}\pi^2$). We will do this by analyzing the function $\Delta\pi^1(\mathbf{x}, \mathbf{y}) = \pi^1(\mathbf{x}) + \pi^1(\mathbf{y}) - \pi^1(\mathbf{x} \oplus \mathbf{y})$. Since ψ is piecewise linear over $\mathcal{P}_{4q}$, π^1 is also piecewise linear over $\mathcal{P}_{4q}$, and thus we only need to focus on vertices of $\Delta\mathcal{P}_{4q}$, which are contained in $\frac{1}{4q}\mathbb{Z}^2$ by Lemma 3.7.

Let $\mathbf{u}, \mathbf{v} \in \frac{1}{4q}\mathbb{Z}^2$. First, if $\Delta\pi(\mathbf{u}, \mathbf{v}) > 0$, then

$$\Delta\pi^1(\mathbf{u}, \mathbf{v}) \geq \pi(\mathbf{u}) - \epsilon/3 + \pi(\mathbf{v}) - \epsilon/3 - \pi(\mathbf{u} \oplus \mathbf{v}) - \epsilon/3 = \Delta\pi(\mathbf{u}, \mathbf{v}) - \epsilon \geq 0.$$

The next step is to show that if $\Delta\pi(\mathbf{u}, \mathbf{v}) = 0$, then $\Delta\pi^1(\mathbf{u}, \mathbf{v}) = 0$. This will show that $\Delta\pi^1(\mathbf{x}, \mathbf{y}) \geq 0$ for all $\mathbf{x}, \mathbf{y} \in [0,1]^2$, and therefore π^1 is subadditive. The proof of the fact that $\Delta\pi(\mathbf{u}, \mathbf{v}) = 0$ implies $\Delta\pi^1(\mathbf{u}, \mathbf{v}) = 0$ is similar to the proof of Lemma 3.5 in [1], and is omitted from this extended abstract. □

3.3 Non-extremality by Diagonal Equivariant Perturbation

We next construct a different equivariant perturbation function. Let $\Gamma_\diagdown = \langle\, \rho_{\mathbf{g}}, \tau_{\mathbf{g}} \mid \mathbf{1} \cdot \mathbf{g} \equiv 0 \pmod{\tfrac{1}{q}} \,\rangle$, where $\mathbf{1} = (1,1)$, be the group generated by reflections and translations corresponding to all points on diagonal edges of $\mathcal{P}_q$. We define the function $\varphi\colon \mathbb{R}^2 \to \mathbb{R}$ as a continuous piecewise linear function over $\mathcal{P}_{4q}$ in the following way:

$$\varphi(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{1} \cdot \mathbf{x} \equiv \frac{1}{4q} \pmod{\frac{1}{q}}, \\ -1 & \text{if } \mathbf{1} \cdot \mathbf{x} \equiv \frac{3}{4q} \pmod{\frac{1}{q}}, \\ 0 & \text{if } \mathbf{1} \cdot \mathbf{x} \equiv 0 \text{ or } \frac{2}{4q} \pmod{\frac{1}{q}}. \end{cases}$$

This function satisfies all properties of Lemma 3.6, but is also $\Gamma_\diagdown$-equivariant.

Lemma 3.9. *Suppose there exists $I^* \in \bar{\mathcal{S}}^1_{q,2}$ and π is diagonally constrained. Then π is not extreme.*

Proof. Let $R = (\bigcup_{J \in \mathcal{G}_{I^*}} J) \setminus \{ \mathbf{x} \mid \mathbf{1} \cdot \mathbf{x} \equiv 0 \text{ or } \frac{2}{4q} \pmod{\frac{1}{q}} \}$.

Let

$$\epsilon = \min\{ \Delta\pi_F(\mathbf{x}, \mathbf{y}) \neq 0 \mid F \in \Delta\mathcal{P}_{4q},\ (\mathbf{x}, \mathbf{y}) \in \text{vert}(F) \},$$

and let $\bar{\pi}$ be the unique continuous piecewise linear function over $\mathcal{P}_{4q}$ such that for any vertex $\mathbf{x}$ of $\mathcal{P}_{4q}$, we have $\bar{\pi}(\mathbf{x}) = \delta_R(\mathbf{x}) \cdot \varphi(\mathbf{x})$ where δ_R is the indicator function for the set R. By construction, $\bar{\pi}$ is a continuous function that vanishes on all diagonal hyperplanes in the complex $\mathcal{P}_q$. We will show that for

$$\pi^1 = \pi + \tfrac{\epsilon}{3}\bar{\pi}, \quad \pi^2 = \pi - \tfrac{\epsilon}{3}\bar{\pi},$$

that π^1, π^2 are minimal, and therefore valid functions, and hence π is not extreme. We will show this just for π^1 as the proof for π^2 is the same. Since, $\varphi(\mathbf{0}) = 0$ and $\varphi(\mathbf{f}) = 0$, we see that $\pi^1(\mathbf{0}) = 0$ and $\pi^1(\mathbf{f}) = 1$. Just like in the proof of Lemma 3.8, it suffices to show that π^1 is subadditive, and we analyze the function $\Delta\pi^1(\mathbf{x}, \mathbf{y}) = \pi^1(\mathbf{x}) + \pi^1(\mathbf{y}) - \pi^1(\mathbf{x} \oplus \mathbf{y})$ over the vertices of $\Delta\mathcal{P}_{4q}$. Let $\mathbf{u}, \mathbf{v} \in \frac{1}{4q}\mathbb{Z}^2$.

First, if $\Delta\pi(\mathbf{u}, \mathbf{v}) > 0$, then $\Delta\pi(\mathbf{u}, \mathbf{v}) \geq \epsilon$ and therefore

$$\Delta\pi^1(\mathbf{u}, \mathbf{v}) \geq \pi(\mathbf{u}) - \epsilon/3 + \pi(\mathbf{v}) - \epsilon/3 - \pi(\mathbf{u} \oplus \mathbf{v}) - \epsilon/3 = \Delta\pi(\mathbf{u}, \mathbf{v}) - \epsilon \geq 0.$$

Next, we will show that if $\Delta\pi(\mathbf{u}, \mathbf{v}) = 0$, then $\Delta\pi^1(\mathbf{u}, \mathbf{v}) = 0$. This will show that $\Delta\pi^1(\mathbf{x}, \mathbf{y}) \geq 0$ for all $\mathbf{x}, \mathbf{y} \in [0, 1]^2$, and therefore π^1 is subadditive. We will proceed by cases.

Case 1. Suppose $\mathbf{u}, \mathbf{v}, \mathbf{u} \oplus \mathbf{v} \notin R$. Then $\delta_R(\mathbf{u}) = \delta_R(\mathbf{v}) = \delta_R(\mathbf{u} \oplus \mathbf{v}) = 0$, and $\Delta\pi^1(\mathbf{u}, \mathbf{v}) = \Delta\pi(\mathbf{u}, \mathbf{v}) \geq 0$.

Case 2. Suppose $\mathbf{u}, \mathbf{v} \in \frac{1}{2q}\mathbb{Z}^2$. Then $\mathbf{1} \cdot (\mathbf{u} \oplus \mathbf{v}) \equiv 0 \pmod{\frac{1}{q}}$ and, by definition of R, $\mathbf{u}, \mathbf{v}, \mathbf{u} \oplus \mathbf{v} \notin R$, and we are actually in Case 1.

Case 3. Suppose we are not in Cases 1 or 2. That is, suppose $\Delta\pi(\mathbf{u}, \mathbf{v}) = 0$, not both $\mathbf{u}, \mathbf{v}$ are in $\frac{1}{2q}\mathbb{Z}^2$, and at least one of $\mathbf{u}, \mathbf{v}, \mathbf{u} \oplus \mathbf{v}$ is in R. Since $\Delta\pi^1(\mathbf{x}, \mathbf{y})$ is symmetric in $\mathbf{x}$ and $\mathbf{y}$, without loss of generality, since not both $\mathbf{u}, \mathbf{v}$ are in $\frac{1}{2q}\mathbb{Z}^2$, we will assume that $\mathbf{u} \notin \frac{1}{2q}\mathbb{Z}^2$.

Since $\mathbf{u} \notin \frac{1}{2q}\mathbb{Z}^2$, $(\mathbf{u}, \mathbf{v}) \notin \text{vert}(\Delta\mathcal{P}_q)$. Therefore, there exists a face $F \in \Delta\mathcal{P}_q$ such that $(\mathbf{u}, \mathbf{v}) \in \text{rel int}(F)$. Since $\Delta\pi_F \geq 0$ (π is subadditive) and $\Delta\pi_F(\mathbf{u}, \mathbf{v}) = 0$, it follows that $\Delta\pi_F = 0$. Now let $(I, J, K) \in E_{\max}(\pi, \mathcal{P}_q)$ such that $F(I, J, K) \supseteq F$. Since π is diagonally constrained, by definition, I, J, K are each either a vertex, diagonal edge, or triangle in $\mathcal{P}_q$. One can show that only the following cases need be considered (we omit a proof of this fact).

1. If $I, J, K \notin \mathcal{P}_{q,2}$, then I, J, K are all vertices or diagonal edges of $\mathcal{P}_q$, which are all not contained in R since all vertices and diagonal edges are subsets of $\{ \mathbf{x} \mid \mathbf{1} \cdot \mathbf{x} \equiv 0 \pmod{\frac{1}{q}} \}$. Therefore, $\mathbf{u}, \mathbf{v}, \mathbf{u} \oplus \mathbf{v} \notin R$, which means we are in Case 1.

2. If $I, J, K \in \mathcal{P}_{q,2}$, then $I, J, K \in \mathcal{S}^2_{q,2}$. By definition of $\bar{\mathcal{S}}^1_{q,2}$, for any $I' \in \mathcal{S}^2_{q,2}$ and $J' \in \bar{\mathcal{S}}^1_{q,2}$, either $I' \cap J' = \emptyset$, or $I' \cap J' \in \mathcal{P}_{q,\diagdown}$. Therefore, $\mathbf{u}, \mathbf{v}, \mathbf{u} \oplus \mathbf{v} \notin R$, which means we are in Case 1.
3. If two of I, J, K are in $\mathcal{P}_{q,2}$ and the third is a vertex, i.e., is in $\mathcal{P}_{q,0}$. Since $\mathbf{u} \notin \frac{1}{q}\mathbb{Z}^2$, I cannot be a vertex. Therefore, $I \in \mathcal{P}_{q,2}$. This case is similar to Lemma 3.5 in [1], and is omitted from this extended abstract.
4. If one of I, J, K is in $\mathcal{P}_{q,\diagdown}$, call it I', and the other two are in $\mathcal{P}_{q,2}$, call them J',K', then $J', K' \in \mathcal{S}^1_{q,2}$ and $\{J', K'\} \in \mathcal{E}_{\diagdown}$. Since $I' \in \mathcal{P}_{q,\diagdown}$, $I' \cap R = \emptyset$. Recall that $\mathcal{S}^1_{q,2} \subseteq \bar{\mathcal{S}}^1_{q,2} \cup \bar{\mathcal{S}}^2_{q,2}$. If either J' or K' is in $\bar{\mathcal{S}}^2_{q,2}$, then they both are in $\bar{\mathcal{S}}^2_{q,2}$, i.e., $J' \cup K' \cap R = \emptyset$ and therefore $\mathbf{u}, \mathbf{v}, \mathbf{u} \oplus \mathbf{v} \notin R$, which is Case 1. We proceed to consider the case where $I' \in \mathcal{P}_{q,\diagdown}$ and $J', K' \in \bar{\mathcal{S}}^1_{q,2}$ with $\{J', K'\} \in \mathcal{E}_{\diagdown}$ of which there are three possible cases.

Case 3a. $I \in \mathcal{P}_{q,\diagdown}$, $J, K \in \mathcal{P}_{q,2}$. Since $\{J, K\} \in \mathcal{E}_{\diagdown}$, $\delta_R(\mathbf{v}) = \delta_R(\mathbf{u} \oplus \mathbf{v})$. Since $I \in \mathcal{P}_{q,\diagdown}$ and $\mathbf{u} \in I$, $\mathbf{1} \cdot \mathbf{u} \equiv 0 \pmod{\frac{1}{q}}$. It follows that $\varphi(\mathbf{u}) = 0$ and $\mathbf{1} \cdot \mathbf{v} \equiv \mathbf{1} \cdot (\mathbf{u} \oplus \mathbf{v}) \pmod{\frac{1}{q}}$. Therefore, $\varphi(\mathbf{v}) = \varphi(\mathbf{u} \oplus \mathbf{v})$. Combining these, we have $\bar{\pi}(\mathbf{u}) + \bar{\pi}(\mathbf{v}) - \bar{\pi}(\mathbf{u} \oplus \mathbf{v}) = 0$, and therefore $\Delta\pi^1(\mathbf{u}, \mathbf{v}) = \Delta\pi(\mathbf{u}, \mathbf{v}) = 0$.

Case 3b. $J \in \mathcal{P}_{q,\diagdown}$, $I, K \in \mathcal{P}_{q,2}$. This is similar to Case 3a and the proof need not be repeated.

Case 3c. $I, J \in \mathcal{P}_{q,2}$, $K \in \mathcal{P}_{q,\diagdown}$ and hence $\mathbf{1} \cdot (\mathbf{u} \oplus \mathbf{v}) \equiv 0 \pmod{\frac{1}{q}}$. Since $\{I, J\} \in \mathcal{E}_{\diagdown}$, we have $\delta_R(\mathbf{u}) = \delta_R(\mathbf{v})$. Since $\mathbf{1} \cdot (\mathbf{u} \oplus \mathbf{v}) \equiv 0 \pmod{\frac{1}{q}}$, we have $\mathbf{1} \cdot \mathbf{u} \equiv -\mathbf{1} \cdot \mathbf{v} \pmod{\frac{1}{q}}$, and hence $\varphi(\mathbf{u}) = -\varphi(\mathbf{v})$. It follows that $\bar{\pi}(\mathbf{u}) + \bar{\pi}(\mathbf{v}) - \bar{\pi}(\mathbf{u} \oplus \mathbf{v}) = 0$, and therefore $\Delta\pi^1(\mathbf{u}, \mathbf{v}) = \Delta\pi(\mathbf{u}, \mathbf{v}) = 0$. □

Proof (of Lemma 3.5). This follows directly from Lemmas 3.8 and 3.9. □

The specific form of our perturbations as continuous piecewise linear function over $\mathcal{P}_{4q}$ implies the following corollary.

Corollary 3.10. *Suppose π is a continuous piecewise linear function over $\mathcal{P}_q$ and is diagonally constrained. If π is not affine imposing over $\mathcal{P}_{q,2}$, then there exist distinct minimal π^1, π^2 that are continuous piecewise linear over $\mathcal{P}_{4q}$ such that $\pi = \frac{1}{2}\pi^1 + \frac{1}{2}\pi^2$.*

3.4 Extremality and Non-extremality by Linear Algebra

In this section we suppose π is a minimal continuous piecewise linear function over $\mathcal{P}_q$ that is affine imposing in $\mathcal{P}_{q,2}$. Therefore, π^1 and π^2 must also be continuous piecewise linear functions over $\mathcal{P}_q$. Recall $E(\pi) \subseteq E(\pi^1), E(\pi^2)$.

We now set up a system of linear equations that π satisfies and that π_1 and π_2 must also satisfy. Let $\varphi\colon \frac{1}{q}\mathbb{Z}^2 \to \mathbb{R}$. Suppose φ satisfies the following system of linear equations:

$$\begin{cases} \varphi(\mathbf{0}) = 0,\ \varphi(\mathbf{f}) = 1,\ \varphi(\binom{0}{1}) = 0,\ \varphi(\binom{0}{1}) = 0,\ \varphi(\binom{1}{1})) = 0, \\ \varphi(\mathbf{u}) + \varphi(\mathbf{v}) = \varphi(\mathbf{u} \oplus \mathbf{v}) \text{ if } \mathbf{u}, \mathbf{v} \in \frac{1}{q}\mathbb{Z}^2,\ \pi(\mathbf{u}) + \pi(\mathbf{v}) = \pi(\mathbf{u} \oplus \mathbf{v}) \end{cases} \tag{2}$$

Since π exists and satisfies (2), we know that the system has a solution.

Theorem 3.11. *Let $\pi\colon \mathbb{R}^2 \to \mathbb{R}$ be a continuous piecewise linear valid function over $\mathcal{P}_q$.*

(i) *If the system* (2) *does not have a unique solution, then π is not extreme.*
(ii) *Suppose π is minimal and affine imposing in $\mathcal{P}_{q,2}$. Then π is extreme if and only if the system of equations* (2) *has a unique solution.*

The proof is similar to one in [1] and is omitted from this extended abstract.

3.5 Connection to a Finite Group Problem

Theorem 3.12. *Let π be a minimal continuous piecewise linear function over $\mathcal{P}_q$ that is diagonally constrained. Then π is extreme if and only if the system of equations* (2) *with $\frac{1}{4q}\mathbb{Z}^2$ has a unique solution.*

Proof. Since π is piecewise linear over $\mathcal{P}_q$, it is also piecewise linear over $\mathcal{P}_{4q}$. The forward direction is the contrapositive of Theorem 3.11(i), applied when we view π piecewise linear over $\mathcal{P}_{4q}$. For the reverse direction, observe that if the system of equations (2) with $\frac{1}{4q}\mathbb{Z}^2$ has a unique solution, then there cannot exist distinct minimal π^1, π^2 that are continuous piecewise linear over $\mathcal{P}_{4q}$ such that $\pi = \frac{1}{2}\pi^1 + \frac{1}{2}\pi^2$. By the contrapositive of Corollary 3.10, π is affine imposing in $\mathcal{P}_{q,2}$. Then π is also affine imposing on $\mathcal{P}_{4q,2}$ since it is a finer set. By Theorem 3.11 (ii), since π is affine imposing in $\mathcal{P}_{4q,2}$ and the system of equations (2) on $\mathcal{P}_{4q}$ has a unique solution, π is extreme. □

Theorem 1.8 and Theorem 1.10 are direct consequences of Theorem 3.12.

References

1. Basu, A., Hildebrand, R., Köppe, M.: Equivariant perturbation in Gomory and Johnson's infinite group problem. eprint arXiv:1206.2079 (math.OC) (2012)
2. Basu, A., Hildebrand, R., Köppe, M., Molinaro, M.: A $(k+1)$-slope theorem for the k-dimensional infinite group relaxation. eprint arXiv:1109.4184 (math.OC) (2011)
3. Cornuéjols, G., Molinaro, M.: A 3-slope theorem for the infinite relaxation in the plane. Mathematical Programming, 1–23 (2012), doi:10.1007/s10107-012-0562-7
4. Dey, S.S., Richard, J.-P.P.: Facets of two-dimensional infinite group problems. Mathematics of Operations Research 33(1), 140–166 (2008)
5. Dey, S.S., Richard, J.-P.P.: Relations between facets of low- and high-dimensional group problems. Mathematical Programming 123(2), 285–313 (2010)
6. Dey, S.S., Richard, J.-P.P., Li, Y., Miller, L.A.: On the extreme inequalities of infinite group problems. Mathematical Programming 121(1), 145–170 (2009)
7. Gomory, R.E.: Some polyhedra related to combinatorial problems. Linear Algebra and its Applications 2(4), 451–558 (1969)
8. Gomory, R.E., Johnson, E.L.: Some continuous functions related to corner polyhedra, I. Mathematical Programming 3, 23–85 (1972), doi:10.1007/BF01585008
9. Gomory, R.E., Johnson, E.L.: Some continuous functions related to corner polyhedra, II. Mathematical Programming 3, 359–389 (1972), doi:10.1007/BF01585008
10. Gomory, R.E., Johnson, E.L.: T-space and cutting planes. Mathematical Programming 96, 341–375 (2003), doi:10.1007/s10107-003-0389-3

Blocking Optimal Arborescences

Attila Bernáth[1] and Gyula Pap[2]

[1] Warsaw University, Institute of Informatics, ul. Banacha 2, 02-097 Warsaw, Poland
athos@cs.elte.hu
[2] MTA-ELTE Egerváry Research Group, Department of Operations Research, Eötvös University, Pázmány Péter sétány 1/C, Budapest, Hungary, H-1117
gyuszko@cs.elte.hu

Abstract. The problem of covering minimum cost common bases of two matroids is NP-complete, even if the two matroids coincide, and the costs are all equal to 1. In this paper we show that the following special case is solvable in polynomial time: given a digraph $D = (V, A)$ with a designated root node $r \in V$ and arc-costs $c : A \to \mathbb{R}$, find a minimum cardinality subset H of the arc set A such that H intersects every minimum c-cost r-arborescence. The algorithm we give solves a weighted version as well, in which a nonnegative weight function $w : A \to \mathbb{R}_+$ is also given, and we want to find a subset H of the arc set such that H intersects every minimum c-cost r-arborescence, and $w(H)$ is minimum. The running time of the algorithm is $O(n^3 T(n, m))$, where n and m denote the number of nodes and arcs of the input digraph, and $T(n, m)$ is the time needed for a minimum $s - t$ cut computation in this digraph. A polyhedral description is not given, and seems rather challenging.

Keywords: arborescences, covering, polynomial algorithm.

1 Introduction

Let $D = (V, A)$ be a digraph with vertex set V and arc set A. A **spanning arborescence** is a subset $B \subseteq A$ that is a spanning tree in the undirected sense, and every node has in-degree at most one. Thus there is exactly one node, the root node, with in-degree zero. Equivalently, a spanning arborescence is a subset $B \subseteq A$ with the property that there is a root node $r \in V$ such that $\varrho_B(r) = 0$, and $\varrho_B(v) = 1$ for $v \in V - r$, and B contains no cycle. An **arborescence** will mean a spanning arborescence, unless stated otherwise. If $r \in V$ is the root of the spanning arborescence B then we will say that B is an **r-arborescence**.

The Minimum Cost Arborescence Problem is the following: given a digraph $D = (V, A)$, a designated root node $r \in V$ and a cost function $c : A \to \mathbb{R}$, find an r-arborescence $B \subseteq A$ such that the cost $c(B) = \sum_{b \in B} c(b)$ of B is smallest possible. Fulkerson [2] has given a two-phase algorithm for solving this problem, and he also characterized minimum cost arborescences. Kamiyama in [4] raised the following question.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 74–85, 2013.

Problem 1. Given a digraph $D = (V, A)$, a designated root node $r \in V$ and a cost function $c : A \to \mathbb{R}$, find a subset H of the arc set such that H intersects every minimum cost r-arborescence, and $|H|$ is minimum.

The minimum in Problem 1 measures the robustness of the minimum cost arborescences, since it asks to delete a minimum cardinality set of arcs in order to destroy all minimum cost r-arborescences. One might ask why we fix the root of the arborescences that we want to cover. The problem of finding the minimum number of arcs that intersect every (globally) minimum cost arborescence can be reduced to Problem 1: add a new node r' to the digraph and connect r' with every old node by high cost and high multiplicity arcs. Then minimum cost arborescences in this new instance will be necessarily rooted at r', they will only contain one arc leaving r' by the high cost of these arcs, and an optimal arc set intersecting these will not use these new arcs because of their high multiplicity.

Kamiyama [4] solved special cases of this problem and he investigated some necessary and sufficient conditions for the minimum in this problem. In this paper we give a polynomial time algorithm solving Problem 1. In fact, our algorithm will solve the following, more general problem, too.

Problem 2. Given a digraph $D = (V, A)$, a designated root node $r \in V$, a cost function $c : A \to \mathbb{R}$ and a nonnegative weight function $w : A \to \mathbb{R}_+$, find a subset H of the arc set such that H intersects every minimum cost r-arborescence, and $w(H)$ is minimum.

The rest of this paper is organized as follows. In Section 2 we give variants of the problem based on Fulkerson's characterization of minimum cost arborescences. These variants are all equivalent with Problem 1 as simple reductions show, so we deal with the variant that can be handled most conveniently. In Section 3 we solve the special case of covering all arborescences: this is indeed a very special case, but the answer is very useful in the solution of the general case. Section 4 contains our main result broken down into two steps: in Section 4.1 we prove a min-min formula that gives a useful reformulation of our problem, and –after introducing some essential results and techniques in Section 4.2– we finally give a polynomial time algorithm solving Problems 1 and 2 in Section 4.3. We give the running time of this algorithm in Section 4.4.

2 The Problem and Its Variants

In this paper we investigate Problem 1 and the more general Problem 2. One interpretation of these problems is that we want to *cover the minimum cost common bases of two matroids*: one matroid being the graphic matroid of D (in the undirected sense), the other being a partition matroid with partition classes $\delta^{in}(v)$ for every $v \in V - r$. A related problem for matroids, the problem of *covering all minimum cost bases of a matroid* is solved in [3]. For sake of simplicity we will mostly speak about Problem 1, and in Section 4.3 we sketch the necessary modifications of our algorithm needed to solve Problem 2. Note

that Problem 2 with an integer weight function w can be reduced to Problem 1 by replacing an arc $a \in A$ (of weight $w(a)$) with $w(a)$ parallel copies (each of weight 1): this reduction is however not polynomial. On the other hand, the algorithm we give for Problem 1 can be simply modified to solve Problem 2 in strongly polynomial time.

Let us give some more definitions. The arc set of the digraph D will also be denoted by $A(D)$. Given a digraph $D = (V, A)$ and a node set $Z \subseteq V$, let $D[Z]$ be the digraph obtained from D by deleting the nodes of $V - Z$ (and all the arcs incident with them). If $B \subseteq A$ is a subset of the arc set, then we will sometimes abuse the notation by identifying B and the graph (V, B): thus $B[Z]$ is obtained from (V, B) by deleting the nodes of $V - Z$ (and the arcs of B incident with them). The set of arcs of D entering Z is denoted $\delta^{in}_D(Z)$, the number of these arcs is $\varrho_D(Z) = |\delta^{in}_D(Z)|$.

The following theorem of Fulkerson characterizes the minimum cost arborescences and leads us to a more convenient, but equivalent problem.

Theorem 1 (Fulkerson, [2]). *There exists a subset $A' \subseteq A$ of arcs (called* tight arcs*) and a laminar family $\mathcal{L} \subseteq 2^{V-r}$ such that an r-arborescence is of minimum cost if and only if it uses only tight arcs and it enters every member of $\mathcal{L}$ exactly once. The set A' and the family $\mathcal{L}$ can be found in polynomial time.*

Since non-tight arcs do not play a role in our problems, we can forget about them, so we assume that $A' = A$ from now on.

Let $\mathcal{L}$ be a laminar family of subsets of V. A spanning arborescence $B \subseteq A$ in D is called a *$\mathcal{L}$-tight arborescence* if both of the following hold.

1. $|\delta^{in}_B(F)| \leq 1$ for all $F \in \mathcal{L}$, and
2. $|\delta^{in}_B(F)| = 0$ for all $F \in \mathcal{L}$ containing the root r of B.

We point out that the second condition in the above definition is needed because we don't want to fix the root of the arborescences: this will be natural in the solution we give for Problem 1. The result of Fulkerson leads us to the following problem.

Problem 3. Given a digraph $D = (V, A)$, a designated root node $r \in V$ and a laminar family $\mathcal{L} \subseteq 2^V$, find a subset H of the arc set such that H intersects every r-rooted $\mathcal{L}$-tight arborescence and $|H|$ is minimum.

Note that in this problem we allow that $r \in F$ for some members $F \in \mathcal{L}$. By Fulkerson's Theorem above, if we have a polynomial algorithm for Problem 3 then we can also solve Problem 1 in polynomial time with this algorithm. However, this can be reversed by the next claim. (Proof is left to the reader.)

Claim 1. *If we have a polynomial algorithm solving Problem 1 then we can also solve Problem 3 in polynomial time.*

We point out that the construction in the above proof also shows how to find an $\mathcal{L}$-tight arborescence, if it exists at all. So we can turn our attention to Problem 3. However, in order to have a more compact answer, it is more convenient

to consider the following, equivalent problem instead, in which the root is not designated. (Proof of equivalence is left to the reader.)

Problem 4. Given a digraph $D = (V, A)$ and a laminar family $\mathcal{L} \subseteq 2^V$, find a subset H of the arc set such that H intersects every $\mathcal{L}$-tight arborescence and $|H|$ is minimum.

Claim 2. *There exists a polynomial algorithm solving Problem 3 if and only if there exists a polynomial algorithm solving Problem 4.*

The main result of this paper is a polynomial algorithm solving Problem 4, and thus, by Claims 1 and 2, for Problems 1 and 3. For a digraph $D = (V, A)$, and a laminar family $\mathcal{L}$ of subsets of V, let $\gamma(D, \mathcal{L})$ denote the minimum number of arcs deleted from D to obtain a digraph that does not contain an $\mathcal{L}$-tight arborescence, that is,

$$\gamma(D, \mathcal{L}) := \min\{|H| : H \subseteq A \text{ such that } D - H \text{ contains no } \mathcal{L}\text{-tight arborescence }\}. \quad (1)$$

3 Covering All Arborescences – A Special Case

In the proof of our main result below, we will use its special case when the laminar family $\mathcal{L}$ is empty. This special case amounts to the following well-known characterization of the existence of a spanning arborescence.

Lemma 1. *For any digraph $D = (V, A)$ exactly one of the following two alternatives holds:*

1. *there exists a spanning arborescence,*
2. *there exist two disjoint non-empty subsets $Z_1, Z_2 \subset V$ such that $\varrho_D(Z_1) = \varrho_D(Z_2) = 0$.*

This characterization also implies a formula to determine the minimum number of edges to be deleted to destroy all arborescences. The characterization is based on double cuts.

Definition 1. *For a digraph $D = (V, A)$, a* double cut *$\delta^{in}(Z_1) \cup \delta^{in}(Z_2)$ is determined by a pair of non-empty disjoint node subsets $Z_1, Z_2 \subseteq V$. The minimum cardinality of a double cut is denoted by $\mu(D)$, that is*

$$\mu(D) := \min\{|\delta^{in}(Z_1)| + |\delta^{in}(Z_2)| : Z_1 \cap Z_2 = \emptyset \neq Z_1, Z_2 \subsetneq V\}. \quad (2)$$

Corollary 1. *For any digraph $D = (V, A)$ the following equation holds: $\gamma(D, \emptyset) = \mu(D)$.*

We point out that a minimum double cut can be found in polynomial time by a simple reduction to minimum cut. Furthermore we will need the following observation (the proof is left to the reader).

Lemma 2. *Given a digraph $D = (V, A)$, let $R = \{r \in V :$ there exists an r-rooted spanning arborescence in $D\}$. Then $D[R]$ is a strongly connected digraph, and $\varrho_D(R) = 0$.*

4 Covering Tight Arborescences

Given a laminar family $\mathcal{L} \subseteq 2^V$, for $F \in \mathcal{L} \cup \{V\}$, let $\mathcal{L}_F := \{F' \in \mathcal{L}, F' \subseteq F\}$. A simple, albeit quite important observation is that a member F of the laminar family also induces a tight arborescence with respect to the laminar family $\mathcal{L}_F$, thus we obtain the following Claim.

Claim 3. *For any $\mathcal{L}$-tight arborescence B, and any $F \in \mathcal{L} \cup \{V\}$, $B[F]$ is an $\mathcal{L}_F$-tight arborescence in $D[F]$.*

The following observation is crucial in our proofs. Given a digraph $D = (V, A)$ and a laminar family $\mathcal{L} \subseteq 2^V$, for an arbitrary member $F \in \mathcal{L}$ and arc $a = xy \in A$ *leaving* F, let $\tilde{D}$ be the graph obtained from D by changing the tail of a for an arbitrary other node $x' \in F$, that is $\tilde{D} = D - xy + x'y$ (where $x, x' \in F$ and $y \notin F$). This operation will be called a **tail-relocation**. Then clearly there is a natural bijection between the arcs of D and those of $\tilde{D}$, but even more importantly, this bijection also induces a bijection between the $\mathcal{L}$-tight arborescences in D and those in $\tilde{D}$. This is formulated in the following claim.

Claim 4. *Let $B \subseteq A$ and $xy \in B$. Then $B - xy + x'y$ is an $\mathcal{L}$-tight arborescence in $\tilde{D}$ if and only if B is an $\mathcal{L}$-tight arborescence in D.*

The claim also implies that $\gamma(D, \mathcal{L}) = \gamma(\tilde{D}, \mathcal{L})$.

4.1 A "Min-Min" Formula

Our approach to determine $\gamma(D, \mathcal{L})$ is broken down into two steps. First, we prove a "min-min" formula, that is, we show that a set H that attains the minimum in (1) is equal to a special arc subset called an *$\mathcal{L}$-double-cut*. The second step will be the construction of an algorithm to find a minimum cardinality $\mathcal{L}$-double-cut.

So what is this first step – the min-min formula all about? It expresses that in order to cover optimally the $\mathcal{L}$-tight arborescences we need to consider the problem of covering the $\mathcal{L}_F$-tight arborescences for every $F \in \mathcal{L} \cup \{V\}$.

Definition 2. *For a set $Z \subseteq V$, let $\mathcal{L}_Z$ denote the family of sets in $\mathcal{L}$ not disjoint from Z, that is, let*

$$\mathcal{L}_Z := \{F \in \mathcal{L} : F \cap Z \neq \emptyset\}. \tag{3}$$

*Then an $\mathcal{L}$-*cut *$M(Z)$ is defined as the set of arcs entering Z, but not leaving any set in $\mathcal{L}_Z$, that is, let*

$$M(Z) := M_{D,\mathcal{L}}(Z) := \delta_D^{in}(Z) - \bigcup_{F \in \mathcal{L}_Z} \left(\delta_D^{out}(F)\right). \tag{4}$$

Note that for a set $F \in \mathcal{L} \cup \{V\}$ this definition of $\mathcal{L}_F$ does not contradict with the definition given in the beginning of Section 4. Thus $M(Z)$ consists of those arcs entering Z, but not leaving any of those sets in $\mathcal{L}$ that have non-empty

intersection with Z. A set function f is given by the cardinality of an $\mathcal{L}$-cut, that is, we define

$$f(Z) := f_D(Z) := f_{D,\mathcal{L}}(Z) := |M_{D,\mathcal{L}}(Z)|. \tag{5}$$

It is useful to observe that

$$f_{D,\mathcal{L}}(Z) \geq f_{D[F],\mathcal{L}_F}(Z \cap F) \text{ for any } F \in \mathcal{L}. \tag{6}$$

The motivation for f and $M(Z)$ is that $H = M(Z)$ is a set of arcs the deletion of which destroys all tight arborescences rooted outside of Z, as claimed by the following lemma.

Lemma 3. *For any $\emptyset \neq Z \subsetneq V$, there is no $\mathcal{L}$-tight arborescence in $D - M(Z)$ rooted in a node $s \in V - Z$.*

Proof. Let $\bar{D} = D - M(Z)$. We prove the lemma by induction on $|\mathcal{L}|$: the base case when $\mathcal{L} = \emptyset$ is obvious. So let $|\mathcal{L}| > 0$ and assume that $P \subseteq A - M(Z)$ is an s-rooted $\mathcal{L}$-tight arborescence in $\bar{D}$ (where $s \in V - Z$). First observe that if $F \in \mathcal{L}_Z$ is arbitrary then, by the induction hypothesis, the root of the $\mathcal{L}_F$-tight arborescence $P[F]$ must be in $F \cap Z$ (apply induction for $\mathcal{L}_F - \{F\}$). Let $v \in Z$ be arbitrary and consider the unique path in P from s to v: assume that $a \in A(\bar{D})$ is the first arc on this path that enters Z. Then there must exist a set $F \in \mathcal{L}_Z$ such that a leaves F. But the root of $P[F]$ must precede a on this path, and it lies in Z, a contradiction.

For any $F \in \mathcal{L} \cup \{V\}$ and nonempty disjoint subsets $Z_1, Z_2 \subseteq F$ the set of arcs in $M_{D[F],\mathcal{L}_F}(Z_1) \cup M_{D[F],\mathcal{L}_F}(Z_2)$ will be called an *$\mathcal{L}$-double cut*, and we introduce the following notation for the minimum cardinality of an $\mathcal{L}$-double cut:

$$\Theta_F := \Theta_{F,D} := \Theta_{F,D,\mathcal{L}} := \min\{f_{D[F],\mathcal{L}_F}(Z_1) + f_{D[F],\mathcal{L}_F}(Z_2) : \\ \emptyset \neq Z_1, Z_2 \subseteq F, Z_1 \cap Z_2 = \emptyset\}.$$

The following simple observation is worth mentioning.

Claim 5. *Given a digraph $D = (V, A)$ and a laminar family $\mathcal{L} \subseteq 2^V$, then $f_{D,\mathcal{L}}(Z) \leq \varrho_D(Z)$ holds for every $Z \subseteq V$. Consequently, $\Theta_{F,D,\mathcal{L}} \leq \mu(D[F])$ holds for any $F \in \mathcal{L} \cup \{V\}$.*

Note that the tail-relocation operation introduced above does not change the f-value of any set $Z \subseteq V$, that is $f_{D,\mathcal{L}}(Z) = f_{D',\mathcal{L}}(Z)$, if D' is obtained from D by (one or several) tail-relocation. Consequently, this operation does not modify the Θ value, either, that is $\Theta_{F,D,\mathcal{L}} = \Theta_{F,D',\mathcal{L}}$ for any $F \in \mathcal{L} \cup \{V\}$. The following "min-min" theorem motivates the definition of Θ.

Theorem 2. *For a digraph $D = (V, A)$, and a laminar family $\mathcal{L}$ of subsets of V, the minimum number of arcs to be deleted from D to obtain a digraph that does not contain an $\mathcal{L}$-tight arborescence is attained on an $\mathcal{L}$-double cut, that is*

$$\gamma(D, \mathcal{L}) = \min_{F \in \mathcal{L} \cup \{V\}} \Theta_{F,D,\mathcal{L}}.$$

Proof. By Lemma 3, $\gamma(D,\mathcal{L}) \leq \min_{F\in\mathcal{L}\cup\{V\}} \Theta_F$, since if we delete an arc set $M_{D[F]}(Z_1)\cup M_{D[F]}(Z_2)$ for some $F \in \mathcal{L}\cup\{V\}$ and non-empty disjoint $Z_1, Z_2 \subseteq F$, then no $\mathcal{L}_F$-tight arborescence survives in $D[F]$ (since its root can neither be in $F - Z_1$, nor in $F - Z_2$, by Lemma 3, and $F = (F - Z_1) \cup (F - Z_2)$).

Assume that $H \subseteq A$ is such that $|H| < \min_{F\in\mathcal{L}\cup\{V\}} \Theta_F$: we will show that there exists an $\mathcal{L}$-tight arborescence in $\bar{D} = D - H$, proving the theorem. It suffices to show the following lemma.

Lemma 4. *If $f_{\bar{D}[F]}(Z_1) + f_{\bar{D}[F]}(Z_2) > 0$ for any $F \in \mathcal{L}\cup\{V\}$ and non-empty disjoint sets $Z_1, Z_2 \subseteq F$, then there exists a $\mathcal{L}$-tight arborescence in $\bar{D}$.*

Proof. We will use induction on $|\mathcal{L}| + |V| + |A(\bar{D})|$. If $\mathcal{L} = \emptyset$ then the lemma is true by Lemma 1. Otherwise let $F \in \mathcal{L}$ be an inclusionwise minimal member of $\mathcal{L}$: again by Lemma 1, there exists a spanning arborescence in $\bar{D}[F]$. Let R be the subset of nodes of F that can be the root of a spanning arborescence in $\bar{D}[F]$, i.e. $R = \{r \in F :$ there exists an r-rooted arborescence (spanning F) in $\bar{D}[F]\}$.

1. Assume first that $|R| \geq 2$ and let $\bar{D}_1 = \bar{D}/R$ obtained by contracting R. For any set $Z \subseteq V$ which is either disjoint form R, or contains R, let Z/R be its (well-defined) image after the contraction and let $\mathcal{L}_1 = \{X/R : X \in \mathcal{L}\}$. By induction, there exists an $\mathcal{L}_1$-tight arborescence P in $\bar{D}_1$, since $f_{\bar{D}[X/R]}(Z/R) = f_{\bar{D}[X]}(Z)$ for any $X/R \in \mathcal{L}_1$ and $Z/R \subseteq X/R$. It is clear that we can create an $\mathcal{L}$-tight arborescence in $\bar{D}$ from P: we describe one possible way. Consider the unique arc in P that enters F and assume that the pre-image of this arc has head $r \in R$. Delete every arc from P induced by F/R and substitute them with an arbitrary r-rooted arborescence (spanning F) of $\bar{D}[F]$. This clearly gives an $\mathcal{L}$-tight arborescence.
2. So we can assume that $R = \{r\}$. Next assume that there exists an arc $uv \in A(\bar{D})$ entering F with $r \neq v$. Let $\bar{D}_2 = \bar{D} - uv$: we claim that there exists an $\mathcal{L}$-tight arborescence in $\bar{D}_2$ (which is clearly an $\mathcal{L}$-tight arborescence in $\bar{D}$, too). If this does not hold then by the induction there must exist a set $F' \in \mathcal{L}$ and non-empty disjoint subsets $Z_1, Z_2 \subseteq F'$ with $\sum_{i=1,2} f_{\bar{D}_2[F']}(Z_i) = 0$. Since $\sum_{i=1,2} f_{\bar{D}[F']}(Z_i) > 0$, the arc uv must be equal to (say) $M_{\bar{D}[F'],\mathcal{L}}(Z_1)$ (while $M_{\bar{D}[F'],\mathcal{L}}(Z_2) = \emptyset$). This implies that uv enters Z_1, while $r \in Z_1$ must also hold, otherwise $f_{\bar{D}[F']}(Z_1) \geq 2$ would hold, since v is reachable from r in $\bar{D}[F']$. Let $Z_1' = Z_1 - (F - r)$ and observe that $f_{\bar{D}[F']}(Z_1') = 0$: this is because the arcs in $\delta^{in}_{\bar{D}[F']}(Z_1') - \delta^{in}_{\bar{D}[F']}(Z_1)$ all leave F, since $\varrho_{\bar{D}[F]}(r) = 0$ by Lemma 2. Thus $f_{\bar{D}[F']}(Z_1') + f_{\bar{D}[F']}(Z_2) = 0$, a contradiction.
3. So we can also assume that the arcs of $\bar{D}$ entering F all enter r. Let $\mathcal{L}_2 = \mathcal{L} - \{F\}$: then clearly $f_{\bar{D}[F'],\mathcal{L}_2}(Z) \geq f_{\bar{D}[F'],\mathcal{L}}(Z)$ for any $F' \in \mathcal{L}_2$ and $Z \subseteq F'$, so by induction there exists an $\mathcal{L}_2$-tight arborescence in $\bar{D}$: by our assumptions this is also $\mathcal{L}$-tight, so the theorem is proved.

4.2 In-Solid Sets and Anchor Nodes

In this section we give some important results for the main theorem.

Definition 3. *A family of sets $\mathcal{F} \subseteq 2^V$ of a finite ground set V is said to satisfy the Helly-property, if any sub-family $\mathcal{X}$ of pairwise intersecting members of $\mathcal{F}$ has a non-empty intersection, i.e. $\mathcal{X} \subseteq \mathcal{F}$ and $X \cap X' \neq \emptyset$ for every $X, X' \in \mathcal{X}$ implies that $\cap\mathcal{X} \neq \emptyset$.*

The following definition is taken from [1].

Definition 4. *Given a digraph $G = (V, A)$, a non-empty subset of nodes $X \subseteq V$ is called* in-solid*, if $\varrho(Y) > \varrho(X)$ holds for every nonempty $Y \subsetneq X$.*

Theorem 3 (Bárász, Becker, Frank [1]). *The family of in-solid sets of a digraph satisfies the Helly-property.*

The authors of [1] prove in fact more: they show that the family of in-solid sets is a *subtree-hypergraph*, but we will only use the Helly property here. The following theorem formulates the key observation for the main result.

Theorem 4. *In a digraph $G = (V, A)$ there exists a node $t \in V$ such that $\varrho(Z) \geq \frac{\mu(G)}{2}$ for every non-empty $Z \subseteq V - t$.*

Proof. Consider the family $\mathcal{X} = \{X \subseteq V : X \text{ is in-solid and } \varrho(X) < \frac{\mu(G)}{2}\}$. If there were two disjoint members $X, X' \in \mathcal{X}$ then $\varrho(X) + \varrho(X') < \mu(G)$ would a contradict the definition of $\mu(G)$. Therefore, by the Helly-property of the in-solid sets, there exists a node $t \in \cap\mathcal{X}$. This node satisfies the requirements of the theorem, since if there was a non-empty $Z \subseteq V - t$ with $\varrho(Z) < \frac{\mu(G)}{2}$, then Z would necessarily contain an in-solid set $Z' \subseteq Z$ with $\varrho(Z') \leq \varrho(Z)$ (this follows from the definition of in-solid sets), contradicting the choice of t.

In a digraph $G = (V, A)$, a node $t \in V$ with the property $\varrho(Z) \geq \frac{\mu(G)}{2}$ for every non-empty $Z \subseteq V - t$ will be called an *anchor node* of G.

4.3 A Polynomial-Time Algorithm

In this section we present a polynomial time algorithm to determine the robustness of tight arborescences, which also implies a polynomial time algorithm to determine the robustness of minimum cost arborescences. A sketch of the algorithm goes as follows. We maintain a subset $\mathcal{L}'$ of $\mathcal{L}$, which is initiated with $\mathcal{L}' := \mathcal{L}$. For a minimal member F of $\mathcal{L}'$, we apply Theorem 4, and find its anchor node a_F. We replace the tail of every arc leaving F by a_F, remove F from $\mathcal{L}'$, and repeat until $\mathcal{L}'$ goes empty. This way we construct a sequence of digraphs on the same node set: let D' be the last member of this sequence. Then for any $a \in \cup\mathcal{L}'$ we construct another digraph D_a from D': for every $F \in \mathcal{L}'$ with $a \in F$ and every arc of D' leaving F we replace the tail of this arc with a. Finally, we determine minimum double cuts in $D'[F]$ for every $F \in \mathcal{L} \cup \{V\}$, and we also determine minimum double cuts in $D_a[F]$ for every $F \in \mathcal{L} \cup \{V\}$ with $a \in F$: this way we have determined $O(n^2)$ double cuts altogether. Each of these double cuts also determines an $\mathcal{L}$-double cut in D, and we pick the one with the smallest cardinality, to claim that it actually is optimal.

Algorithm COVERING_TIGHT_ARBORESCENCES
begin
INPUT A digraph $D = (V, A)$ and a laminar family $\mathcal{L} \subseteq 2^V$
OUTPUT $\gamma(D, \mathcal{L})$
/* First Phase: creating graphs D' and D_a */
1.1. Let $D' = D$ and $\mathcal{L}' = \mathcal{L}$.
1.2. While $\mathcal{L}' \neq \emptyset$ do
1.3. Choose an inclusionwise minimal set $F \in \mathcal{L}'$
1.4. Let $a_F \in F$ be an anchor node of $D'[F]$ (apply Theorem 4 to $G = D'[F]$)
1.5. Modify D': relocate the tail of all arcs leaving F to a_F
1.6. Let $\mathcal{L}' = \mathcal{L}' - F$
1.7. For every $a \in \cup\mathcal{L}$
1.8. Let D_a be obtained from D' by relocating the tail of every arc leaving a set $F \in \mathcal{L}$ with $a \in F$ to a
/*Second Phase: finding the optimum*/
1.9. Let $best = +\infty$ and $\mathcal{L}' = \mathcal{L}$.
1.10. While $\mathcal{L}' \neq \emptyset$ do
1.11. Choose an inclusionwise minimal set $F \in \mathcal{L}'$
1.12. If $best > \mu(D'[F])$ then $best := \mu(D'[F])$
1.13. For each $a \in F$ do
1.14. If $best > \mu(D_a[F])$ then $best := \mu(D_a[F])$
1.15. Let $\mathcal{L}' = \mathcal{L}' - F$
1.16. Return $best$.
end

The algorithm above is formulated in a way that it returns the optimum $\gamma(D, \mathcal{L})$ in question, but by the correspondance between the arc set of D and that of D' and D_a in the algorithm, clearly we can also return the optimal arc set, too. It is also clear that the algorithm can be formulated to run in strongly polynomial time for Problem 2, too: we only need to modify the definition of the in-degree function ϱ_D, so that the weights are taken into account.

Theorem 5. *The Algorithm COVERING_ TIGHT_ ARBORESCENCES returns a correct answer.*

Proof. First of all, since $\Theta_{F,D} = \Theta_{F,D'} = \Theta_{F,D_a}$ for any $F \in \mathcal{L} \cup \{V\}$ and $a \in F \cap \cup\mathcal{L}$, and $\Theta_{F,D'} \leq \mu(D'[F])$ and $\Theta_{F,D_a} \leq \mu(D_a[F])$, the algorithm returns an upper bound for the optimum $\gamma(D, \mathcal{L})$ in question by Theorem 2.

On the other hand, assume that F is an inclusionwise minimal member of $\mathcal{L} \cup \{V\}$ such that the optimum $\gamma(D, \mathcal{L}) = \Theta_{F,D}$ (such a set exists again by Theorem 2). Assume furthermore that the non-empty disjoint sets $Z_1, Z_2 \subseteq F$ are such that $\Theta_{F,D} = f_{D[F]}(Z_1) + f_{D[F]}(Z_2)$. The following sequence of observations proves the theorem.

1. First observe, that any member $F' \in \mathcal{L}$ which is a proper subset of F can intersect at most one of Z_1 and Z_2. Assume the contrary, and note that $f_{D[F]}(Z_i) \geq f_{D[F']}(Z_i \cap F')$ holds for $i = 1, 2$, contradicting the minimal choice of F.

2. Next observe that there do not exist two disjoint members $F', F'' \in \mathcal{L}_{Z_1 \cup Z_2}$ that are proper subsets of F such that $a_{F'}$ and $a_{F''}$ are both outside $Z_1 \cup Z_2$. To see this assume again the contrary and let F', F'' be two inclusionwise minimal such sets. By exchanging the roles of Z_1 and Z_2 or the roles of F' and F'' we arrive at the following two cases: either both F' and F'' intersect Z_1, or F' intersects Z_1 and F'' intersects Z_2. The proof is analogous for both cases. Assume first that both F' and F'' intersect Z_1. Then we have

$$\begin{aligned}\gamma(D,\mathcal{L}) = \Theta_{F,D} = \Theta_{F,D'} = f_{D'[F]}(Z_1) + f_{D'[F]}(Z_2) \geq \\ \geq f_{D'[F]}(Z_1) \geq f_{D'[F']}(Z_1 \cap F') + f_{D'[F'']}(Z_1 \cap F'') = \\ = \varrho_{D'[F']}(Z_1 \cap F') + \varrho_{D'[F'']}(Z_1 \cap F'') \geq \\ \geq \frac{\mu(D'[F'])}{2} + \frac{\mu(D'[F''])}{2} > \gamma(D,\mathcal{L}), \quad (7)\end{aligned}$$

a contradiction. Here the second inequality follows from the definition of the function f, the equality following it is because $a_{F'''} \in Z_1$ if $F''' \in \mathcal{L}_{Z_1}$ is a proper subset of F' or F''. The next inequality follows from the definition of $a_{F'}$ and $a_{F''}$, and the last (strict) inequality is by the minimal choice of F. In the other case, when F' intersects Z_1 and F'' intersects Z_2, we get the contradiction in a similar way:

$$\begin{aligned}\gamma(D,\mathcal{L}) = \Theta_{F,D} = \Theta_{F,D'} = f_{D'[F]}(Z_1) + f_{D'[F]}(Z_2) \geq \\ \geq f_{D'[F']}(Z_1 \cap F') + f_{D'[F'']}(Z_2 \cap F'') = \varrho_{D'[F']}(Z_1 \cap F') + \varrho_{D'[F'']}(Z_2 \cap F'') \geq \\ \geq \frac{\mu(D'[F'])}{2} + \frac{\mu(D'[F''])}{2} > \gamma(D,\mathcal{L}). \quad (8)\end{aligned}$$

3. Therefore we are left with two cases. In the first case assume that $a_{F'} \in Z_1 \cup Z_2$ for any $F' \in \mathcal{L}_{Z_1 \cup Z_2}$ that is proper subsets of F. In that case we have that $f_{D'[F]}(Z_i) = \varrho_{D'[F]}(Z_i)$ for both $i = 1, 2$, and thus $\gamma(D,\mathcal{L}) = \Theta_{F,D'} = \sum_{i=1,2} \varrho_{D'[F]}(Z_i) \geq \mu(D'[F]) \geq \Theta_{F,D'}$.
4. In our last case there exists a unique inclusionwise minimal $F' \in \mathcal{L}_{Z_1 \cup Z_2}$ such that F' is proper subsets of F and $a_{F'} \notin Z_1 \cup Z_2$. Assume without loss of generality that F' intersects Z_1 and choose an arbitrary $a \in F' \cap Z_1$. Then $f_{D_a[F]}(Z_i) = \varrho_{D_a[F]}(Z_i)$ for both $i = 1, 2$, and thus $\gamma(D,\mathcal{L}) = \Theta_{F,D_a} = \sum_{i=1,2} \varrho_{D_a[F]}(Z_i) \geq \mu(D_a[F]) \geq \Theta_{F,D_a}$.

4.4 Running Time

Let $T(N, M)$ be the time needed to find a minimum $s-t$ cut in an edge-weighted digraph having N nodes and M arcs (that is, $M \leq N^2$ here).

The natural weighted version of Problem 4 is the following.

Problem 5. Given a digraph $D = (V, A)$, and a nonnegative weight function $w : A \to \mathbb{R}_+$, and a laminar family $\mathcal{L} \subseteq 2^V$, find a subset H of the arc set such that H intersects every $\mathcal{L}$-tight arborescence and $w(H)$ is minimum.

As mentioned above, if we want to solve the weighted Problem 5, the only thing to be changed is that the in-degree $\varrho(X)$ of a set should mean the weighted in-degree. We will analyze the algorithm in this sense, so we assume that the input digraph does not contain parallel arcs, but weighted ones.

In order to analyze the performance of Algorithm COVERING_TIGHT_-ARBORESCENCES, let n and m denote the number of nodes and arcs in its input (so $m \leq n^2$).

To implement the algorithm above we need 2 subroutines. The first subroutine finds an anchor node in an edge-weighted digraph. This subroutine will be used $|\mathcal{L}| \leq n$ times in Step 1.4 for digraphs having at most n nodes and at most m arcs. By the definition of anchor nodes, any node r maximizing $\min\{\varrho_G(X) : \emptyset \neq X \subseteq V - r\}$ can serve as an anchor node. Therefore, finding an anchor can be done in $n^2T(n,m)$.

The second subroutine determines $\mu(G)$ for a given edge-weighted digraph G. This subroutine is used at most n times in Step 1.12 and n^2 times in Step 1.14 for digraphs having at most n nodes and at most m arcs. Note however that these suboutine calls are not independent from each other, and we will make use of this fact later.

We can determine $\mu(G)$ for a given edge-weighted digraph G the following way: take two disjoint copies of G, and reverse all arcs in the first copy (and denote this modified first copy by G^1). Let the second copy be denoted by G^2, and for each $v \in V(G)$ let the corresponding node in $V(G^i)$ be v^i for $i = 1, 2$. For each $v^1 \in V(G^1)$ add an arc v^1v^2 of infinite capacity from v^1 to its corresponding copy $v^2 \in V(G^2)$. This way we define an auxiliary graph $\hat{G}$. It is easy to see that for some $s \neq t$ nodes in $V(G)$ we have $\min\{\delta_{\hat{G}}(Z) : s^1 \in Z \subseteq V(\hat{G}) - t^2\} = \min\{\varrho_G(X) + \varrho_G(Y) : s \in X \subsetneq V(G), t \in Y \subsetneq V(G), X \cap Y = \emptyset\}$. Thus, by trying every possible pair s, t, we can calculate $\mu(G)$ with n^2 minimum s^1-t^2-cut computations in $\hat{G}$ in time $n^2T(n,m)$.

We will calculate $\mu(G)$ for $O(n^2)$ graphs G (each having at most n nodes and m arcs): n times in Step 1.12 for the graphs $D'[F]$, and n^2 times in Step 1.14 for the graphs $D_a[F]$. On the other hand, as mentioned earlier, these calls are not independent from each other, since if $\mu(D_a[F]) < \mu(D'[F])$ for some $F \in \mathcal{L}$ and $a \in F$ then $a \in X \cup Y$ has to hold for the (optimal disjoint non-empty) sets $X, Y \subseteq F$ giving $\mu(D_a[F]) = \varrho_{D_a[F]}(X) + \varrho_{D_a[F]}(Y)$. Therefore checking whether $\mu(D_a[F]) < best$ or not in Step 1.14, we only need to calculate minimum $a^1 - t^2$-cuts in $\widehat{D_a[F]}$ (for the node $a^1 \in V((D_a[F])^1)$ corresponding to a and every $t^2 \in V((D_a[F])^2)$ corresponding to nodes $t \in F - a$), needing only n minimum cut computations.

Putting everything together we get that the Algorithm COVERING_TIGHT _ARBORESCENCES can be implemented to run in $n^3T(n,m)$ time. It seems possible to further reduce the complexity of Steps 1.4 and 1.12, however we cannot do this for Step 1.14.

4.5 Remarks on the Polyhedral Approach

The tractability of the weighted Problem 5 is equivalent with optimization over the following polyhedron:

$$\mathcal{P} := conv(\{\chi_H : H \text{ a } \mathcal{L}\text{-double cut}\}) + \mathbb{R}^A_+.$$

A completely different approach to the problem would be to directly show that this polyhedron $\mathcal{P}$ is tractable, which hinges upon finding a nice polyhedral description of the given polyhedron. Firstly, the polyhedron has facets with large coefficients, which rules out a rank-inequality type description. Secondly, the polyhedron seems to be of a composite nature in the following sense. For $\mathcal{L} = \emptyset$,

$$\mathcal{P} = conv\left(\bigcup_{s\neq t, s,t\in V} conv(\{\chi_H : H \text{ a double cut separating } s,t\}) + \mathbb{R}^A_+\right),$$

where a double cut is said to separate s,t if $s \in Z_1, t \in Z_2$. Thus optimization over $\mathcal{P}$ reduces to optimization over $\binom{n}{2}$ polyhedra of double cuts separating a given pair s,t. For any given pair s,t, this polyhedron has a nice description, and also nice combinatorial algorithm for optimization. When we apply this approach to a general $\mathcal{L}$, then we need to consider the union of an exponential number of polyhedra: one for every possible choice of an anchor node in every set of $\mathcal{L}$. Thus the proposed approach only results in an efficient algorithm for the special case $\mathcal{L} = \emptyset$, and leaves the general case without a polyhedral description.

Acknowledgements. We thank Naoyuki Kamiyama for calling our attention to this problem at the 7th Hungarian-Japanese Symposium on Discrete Mathematics and Its Applications in Kyoto. We would like to thank Kristóf Bérczi, András Frank, Erika Kovács, Tamás Király and Zoltán Király of the Egerváry Research Group for useful discussions and remarks. The authors received a grant (no. CK 80124) from the National Development Agency of Hungary, based on a source from the Research and Technology Innovation Fund. The work of the first author was partially supported by the ERC StG project PAAl no. 259515. A preliminary version of this paper was presented at the 21st International Symposium on Mathematical Programming (ISMP 2012).

References

1. Bárász, M., Becker, J., Frank, A.: An algorithm for source location in directed graphs. Oper. Res. Lett. 33(3), 221–230 (2005)
2. Fulkerson, D.R.: Packing rooted directed cuts in a weighted directed graph. Mathematical Programming 6, 1–13 (1974), doi:10.1007/BF01580218
3. The EGRES Group: Covering minimum cost spanning trees, EGRES QP-2011-08, `www.cs.elte.hu/egres`
4. Kamiyama, N.: Robustness of Minimum Cost Arborescences. In: Asano, T., Nakano, S.-I., Okamoto, Y., Watanabe, O. (eds.) ISAAC 2011. LNCS, vol. 7074, pp. 130–139. Springer, Heidelberg (2011)

Minimum Clique Cover in Claw-Free Perfect Graphs and the Weak Edmonds-Johnson Property

Flavia Bonomo[1], Gianpaolo Oriolo[2], Claudia Snels[2], and Gautier Stauffer[3]

[1] IMAS-CONICET and Departamento de Computación, FCEN,
Universidad de Buenos Aires, Argentina
fbonomo@dc.uba.ar
[2] Dipartimento di Ingegneria Civile e Ingegneria Informatica,
Università Tor Vergata, Roma, Italy
{oriolo,snels}@disp.uniroma2.it
[3] Bordeaux Institute of Mathematics, France
gautier.stauffer@math.u-bordeaux1.fr

Abstract. We give new algorithms for the minimum (weighted) clique cover in a claw-free perfect graph G, improving the complexity from $O(|V(G)|^5)$ to $O(|V(G)|^3)$. The new algorithms build upon neat reformulations of the problem: it basically reduces either to solving a 2-SAT instance (in the unweighted case) or to testing if a polyhedra associated with the edge-vertex incidence matrix of a bidirected graph has an integer solution (in the weighted case). The latter question was elegantly answered using neat polyhedral arguments by Schrijver in 1994. We give an alternative approach to this question combining pure combinatorial arguments (using techniques from 2-SAT and shortest paths) with polyhedral ones. Our approach is inspired by an algorithm from the Constraint Logic Programming community and we give as a side benefit a formal proof that the corresponding algorithm is correct (apparently answering an open question in this community). Interestingly, the systems we study have properties closely connected with the so-called Edmonds-Johnson property and we study some interesting related questions.

Keywords: clique cover, claw-free perfect graphs, bidirected graphs, Edmonds-Johnson property.

1 Introduction

Given a graph G, a *clique cover* is a collection $\mathcal{K}$ of cliques covering all the vertices of G. Given a weight function $w : V(G) \mapsto \mathbb{Q}$ defined on the vertices of G, a *weighted clique cover* of G is a collection of cliques $\mathcal{K}$, with a positive weight y_K assigned to each clique K in the collection, such that, for each vertex v of G, $\sum_{K \in \mathcal{K}: v \in K} y_K \geq w(v)$. A *minimum clique cover* of G (MCC) is a clique cover of minimum cardinality, while a *minimum weighted clique cover* of G (MWCC) is a weighted clique cover minimizing $\sum_{K \in \mathcal{K}} y_K$.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 86–97, 2013.

For perfect graphs, it is well-known [5,20] that the convex hull of the incidence vectors of all stable sets is described by clique inequalities and non-negativity constraints. It follows that the maximum weighted stable set (MWSS) problem (the left program) and the MWCC problem (the right program) form a primal-dual pair:

$$\max \sum_{v \in V} w(v)x_v \qquad\qquad \min \sum_{C \in \mathcal{K}(G)} y_C$$

$$\sum_{v \in C} x_v \leq 1 \quad \forall C \in \mathcal{K}(G) \qquad\qquad \sum_{C \in \mathcal{K}(G): v \in C} y_C \geq w(v) \quad \forall v \in V$$

$$x_v \geq 0 \quad \forall v \in V \qquad\qquad y_C \geq 0 \quad \forall C \in \mathcal{K}(G)$$

Moreover, when w is integral, there always exists an integer solution to the MWCC problem, as it was originally shown by Fulkerson [10].

In 1988, Grötschel, Lovász and Schrijver [12] gave a (non-combinatorial) polynomial time algorithm, building upon Lovász's theta function, to compute solutions to the MWSS problem and the MWCC problem in perfect graphs. It is a major open problem in combinatorial optimization whether there exist polynomial time combinatorial algorithms for those two problems.

For particular classes of perfect graphs, such algorithms exist. This is the case, for instance, for claw-free perfect graphs: a graph is *claw-free* if none of its vertices has a stable set of size three in its neighborhood. Claw-free graphs are a superclass of line graphs, and the MWSS problem in claw-free graphs is a generalization of the matching problem, and in fact there are several polynomial time combinatorial algorithms for solving the former problem (see [23]) and the fastest algorithm [9] runs in time $O(|V(G)|^2 \log |V(G)| + |V(G)||E(G)|)$. Conversely, to the best of our knowledge, the only combinatorial algorithm for the MWCC problem in the (entire) class of claw-free perfect graphs is due to Hsu and Nemhauser [15] in 1984 and runs in $O(|V(G)|^5)$. The algorithm is based on a clever use of complementary slackness in linear programming, combined with the resolution of several MWSS problems. Hsu and Nemhauser also designed a more efficient algorithm for the unweighted case [14], that runs in $O(|V(G)|^4)$. However, building *non-trivially* upon the clique cutset decomposition theorems for claw-free perfect graphs by Chvátal and Sbihi [6] and Maffray and Reed [19] and the algorithmic approach by Whitesides [27], one may design an $O(|V(G)|^3 \log |V(G)|)$-time algorithm for the MCC problem and a more involved $O(|V(G)|^4)$-time algorithm for the MWCC problem (for the latter result, one needs to use some ideas from [4], where an $O(|V(G)|^3)$-time algorithm for solving the MWCC problem on the subclass of *strip-composed* claw-free perfect graphs is given). [We defer the (long) details for this approach to the journal version of this paper.]

Our new approach to the problem relies on testing and building integer solution to systems of inequalities with at most 2 non-zero coefficients per row,

both of them in $\{-1, +1\}$. We study in a slightly more general problem: given an $m \times n$ matrix A satisfying

$$\sum_{j=1}^{n} |a_{ij}| \leq 2, \text{ for all } i = 1, .., m, \text{ with } a_{ij} \in \mathbb{Z} \text{ for all i,j} \quad (1)$$

(i.e., A is the vertex-edge incidence matrix of a bidirected graph, see Chapter 36 in [23] for more properties of those systems) and an integer vector b, can one determine in polynomial time if the system $Ax \leq b$ has an integer solution (and build one if any)? So we are interested in the polyhedron $P_b(A) := \{x \in \mathbb{R}^n : Ax \leq b\}$, and in particular in knowing if the integer hull of $P_b(A)$, that we denote by $Int(P_b(A))$, is empty or not: we sometimes refer to this question as the *integer feasibility* for $P_b(A)$. (When A is clear from the context, we abuse notation and denote $P_b(A)$ by P_b.) Note that all inequalities in $P_b(A)$ are of the type $x_i + x_j \leq b_{ij}$, $-x_i - x_j \leq b_{ij}$, $x_i - x_j \leq b_{ij}$, $x_i \leq b_i$, $-x_i \leq b_i$, $2x_i \leq b_i$, $-2x_i \leq b_i$.

We just pointed out that addressing efficiently the is question of integer feasibility for $P_b(A)$ leads to improved algorithm for MWCC in claw-free perfect graphs. However those systems are interesting for their own sake, as they also appear in other contexts, like for instance hardware and software verification [2], and, they received considerable attention from the Constraint Logic Programming community, as we recall later.

Schrijver [22] was, to the best of our knowledge, the first to consider this question (and he was motivated by some path problem in planar graphs!), and he gave an $O(n^3)$-time algorithm based on the Fourier-Motzkin elimination scheme that also produces a feasible integer solution when it exists.

An alternative to Schrijver's approach is that of Peis [21]. She reduces the problem of checking whether $Int(P_b) = \emptyset$ to, first, testing for fractional feasibility, i.e. if $P_b = \emptyset$, through shortest paths techniques. If P_b is non-empty, she gets a half integral solution certificate as a side benefit of the shortest path calculation. Then she tests if the fractional components of this half integral solution can be "rounded" up or down to an integer solution (solving a suitable 2-SAT problem). She proves that there always exists such a rounding procedure when a feasible integer solution exists. Like Schrijver's approach, her method is constructive, i.e. she builds a feasible integer solution when the system is integer non-empty. Her algorithm can be implemented to run in time $O(nm)$.

The result of Schrijver, and the more recent work of Peis, do not seem to be very well known, as several people in the Constraint Logic Programming community developed alternative algorithms and arguments for the problem (see e.g. [16,13,18,25,2,24]), apparently ignoring the (previous) result in [22]. Interestingly though, the focus of this community is slightly different. They are not only interested in the integer feasibility, but they want to build efficiently what they call the *tight closure* of the system to possibly derive additional structural properties.

The best algorithm [24] to derive the tight closure runs in time $O(n^2 \log n + mn)$ (note that this is better than $O(n^3)$ as $m = O(n^2)$ when A satisfies (1)),

while the best algorithm for testing integer feasibility runs in $O(nm)$ [18]. It seems however that all those results were pretty controversial in this community as they all rely on a fundamental theorem claimed in [16] that was never proved formally, as pointed out by [2] who declare in their paper "to present and, for the first time, fully justify an $O(n^3)$ algorithm to compute the tight closure of a set of UTVPI integer constraints".

We outline now the main contributions of each section. In Section 2, we discuss a new $O(|V(G)|^3)$-time, very simple, algorithm for the minimum (cardinality) clique cover in claw-free perfect graphs. We then extend our finding and devise a $O(V(G)|^3)$ algorithm for the weighted case thanks to Schrijver's result for matrices satisfying (1). In Section 3, we revisit from a polyhedral perspective the algorithm proposed in [24] for the integer feasibility and tight closure of systems $Ax \leq b$, with A satisfying (1), and offer a self-contained proof for its correctness and running time. We believe that this contribution is important as it bridges the gap between the CP community and the integer programming one, and also yields the tight closure (this is not possible neither with the approach of Schrijver, nor with that of Peis), and therefore addresses the different focus of the CP community. In Section 3.1, we introduce and study properties of those system that are closely related to the so-called Edmonds-Johnson property, and in Section 3.2 we identify a class of them with the following nice property: if the system has a fractional solution, then it has an integral one, and we show that this class includes the systems arising from the MWCC problem. [*For the sake of shortness some proofs will be postponed to the full version of this paper.*]

2 Clique Covers in Claw-Free Perfect Graphs

We focus here on claw-free perfect graphs. We will give new $O(|V(G)|^3)$-time algorithms for the MCC and the MWCC problem. In particular, we will show how to "reduce" the latter problem to testing the existence of integer solution in polyhedra associated with the edge-incidence matrix of bidirected graphs. We start with the unweighted case.

2.1 A New Algorithm for MCC in Claw-Free Perfect Graphs

Suppose that we are given a stable set S of a claw-free perfect graph $G = (V, E)$. We want to check if S is a maximum stable set of G. In the case that it is, we want to build a suitable clique cover of G of size $|S|$; in case it is not, we want to find an augmenting path (given a stable set S of a graph G, a path P is *S–alternating* if $(V(P) \setminus S) \cup (S \setminus V(P))$ is a stable set of G; *S–augmenting*, if in addition this stable set has size $|S| + 1$. Berge [3] proved that a stable set S is maximum for a claw-free graph G if and only if there are no paths that are S–augmenting). Without loss of generality we assume that S is maximal; therefore a vertex $v \in V \setminus S$ is either *bound*, i.e., it is adjacent to two vertices $s_1(v)$ and $s_2(v)$ of S, or is *free*, i.e., it is adjacent to one vertex $s(v)$ of S.

We will achieve our target by solving a suitable instance of the 2-SAT problem. The rationale is the following. By complementary slackness, in a perfect graph, every clique of a MCC intersects every MSS. Therefore, given S, in order to build a MCC we must "assign" each vertex of $v \in V \setminus S$ to a vertex in $N(v) \cap S$, in such a way that the set of vertices of $V \setminus S$ assigned to a same $s \in S$ will form a clique. As we show in the following, this can be easily expressed as a 2-SAT formula.

For every bound (resp. free) vertex $v \in V \setminus S$, we define two (resp. one) variables, or terms, $x_{vs_1(v)}$ and $x_{vs_2(v)}$ (resp. $x_{vs(v)}$) that will specify the above assignment. We also introduce an auxiliary boolean variable y to express that, for a free vertex v, $x_{vs(v)}$ has to be *true*. We consider three classes of clauses (we again denote by $\neg x_{vs}$ the negation of a term x_{vs}):

(c1) for each $v \in V \setminus S$ that is bound, $x_{vs_1(v)} \vee x_{vs_2(v)}$ must be true;
(c2) for each $s \in S$ and each $u, v \in N(s)$ that are non-adjacent, $\neg x_{us} \vee \neg x_{vs}$ must be true;
(c3) for each $v \in V \setminus S$ that is free, both $x_{vs(v)} \vee y$ and $x_{vs(v)} \vee \neg y$ must be true (i.e., $x_{vs(v)}$ must be true).

Consider the 2-SAT instance made of the conjunction of all the above clauses, which we denote in the following by the pair (G, S). It is straightforward to check that a clique cover of size $|S|$ induces a solution (i.e. a satisfying truth assignment) to (G, S). Vice versa, from a solution to (G, S) we can easily build a clique cover of size $|S|$ of G. In fact, for each vertex $s \in S$, let $X(s) := \{s\} \cup \{v \in N(s) : x_{vs} \text{ true}\}$. Note that for each free vertex u, following (c3), $u \in X(s(u))$. Moreover, for each $s \in S$, $X(s)$ is a clique, following (c2). Finally, following the clauses (c1), each bound vertex u belongs to either $X(s_1(u))$ or to $X(s_2(u))$. The family $\{X(s), s \in S\}$ is then a clique cover of size $|S|$. Therefore, a maximal stable set S is a maximum stable set of G if and only if there exists a solution to the 2-SAT instance (G, S). Moreover, from a solution to (G, S) we can easily build a MCC of G.

Following the above discussion, in order to design an algorithm for the MCC problem of a claw-free perfect graph G, we are left with the following question: what if S is *not* a maximum stable set of G, i.e. there is no solution to the 2-SAT instance (G, S)? In this case, in time $O|V(G)^2|$ we can find a path that is augmenting with respect to S. While we postpone the proof of this argument, that is rather standard, to the full version of the paper, we point that the search for this augmenting path is *not* technical, as we simply get it from a careful analysis of the implication graph of the unsatisfiable 2-SAT instance.

One therefore gets a simple algorithm that produces both a MCC and a MSS of a claw-free perfect graph G in time $O(|V(G)|^3)$, in the spirit of the augmenting path algorithm for maximum bipartite matching and minimum vertex cover.

2.2 A New Algorithm for MWCC in Claw-Free Perfect Graphs

We are now given a claw-free perfect graph $G = (V, E)$ and also a weight function $w : V(G) \mapsto \mathbb{N} \setminus \{0\}$. Since w is strictly positive, every MWSS is maximal. We want to check if a given maximal stable set S of G is also a MWSS.

We will follow an approach inspired by the unweighted case. In that case, as in a perfect graph every clique of a MCC intersects every MSS, we tried to "assign" each vertex $v \in V \setminus S$ to a vertex in $N(v) \cap S$, so that the vertices of $V \setminus S$ assigned to a same $s \in S$ form a clique. In the weighted case, the assignment is no longer possible, as some vertices might have to be covered by several cliques in a MWCC. However, for each $v \in V \setminus S$ and $s \in N(v) \cap S$, we will *compute how much of $w(v)$ is covered by cliques that contain both s and v*. Therefore, for every bound (resp. free) vertex $v \in V \setminus S$, we define two (resp. one) non-negative integer variables $x_{vs_1(v)}$ and $x_{vs_2(v)}$ (resp. $x_{vs(v)}$), that will provide that information. We then consider the following constraints (note that, for $s \in S$ and $v \in N(s)$, x_{vs} is equivalent to either $x_{vs(v)}$, or $x_{vs_1(v)}$, or $x_{vs_2(v)}$):

(d1) for each $v \in V \setminus S$ that is bound, $x_{vs_1(v)} + x_{vs_2(v)} \geq w(v)$;
(d2) for each $v \in V \setminus S$ that is free: $x_{vs(v)} \geq w(v)$.
(d3) for each $s \in S$ and each $u, v \in N(s)$ that are non-adjacent, $x_{us} + x_{vs} \leq w(s)$;
(d4) for each $s \in S$ and each $u \in N(s)$, $x_{us} \leq w(s)$.

Consider the integer program P_b defined by the previous constraints, together with non-negativity and integrality for each variable. We claim that P_b has a (integer) solution if and only if there exists for G a (integer) weighted clique cover $(\mathcal{K}, y)$ with weight $w(S)$, i.e. if and only if S is a MWSS of G. Suppose there exists a weighted clique cover $(\mathcal{K}, y)$ of G with weight $w(S)$. Then S is a MWSS of G. It is straightforward to check that y induces a solution to P_b by letting, for each $s \in S$ and $v \in N(s)$, $x_{vs} = \sum_{K \in \mathcal{K}: s,v \in K} y_K$. Vice versa, let x be a (integer) solution to P. We want to "translate" x into a weighted clique cover $(\mathcal{K}, y)$ of weight $w(S)$. Let $s \in S$: we first take care of the weights of the cliques in the cover that contain s. So consider the graph $G^s = G[N[s]]$, with a weight function w^s defined as follows: for each vertex $v \in N(s)$, $w^s(v) = x_{vs}$; $w^s(s) = w(s)$. Trivially, $\{s\}$ is a MWSS of G^s, with respect to the weight function w^s, following constraints (d3)-(d4). Moreover, as every clique of G^s is a clique of G too, and $w^s(s) = w(s)$, if we compute a MWCC $(\mathcal{K}^s, y^s)$ of G^s (with respect to w^s), then the following holds: (j) for each vertex $v \in N(s)$, $\sum_{K \in \mathcal{K}^s: s,v \in K} y^s_K \geq x_{vs}$; (jj) $\sum_{K \in \mathcal{K}^s} y^s_K = w(s)$. Following constraints (d1)-(d2), if we compute, for *each* $s \in S$, a MWCC $(\mathcal{K}^s, y^s)$ of G^s, with respect to w^s, and we take $\mathcal{K} = \bigcup_{s \in S} \mathcal{K}^s$ and juxtapose the different y^s, $s \in S$, we then get a weighted clique cover $(\mathcal{K}, y)$ of G of weight $w(S)$.

We are left with two questions. The first, and more challenging one, is that of showing how it is possible to find an integral solution x to the system P_b defined by constraints (d1)-(d4). Observe that any inequality in (d1)-(d-4) involves at most two non-zero coefficients in $\{-1, +1\}$. Building integer solution to such systems can be done in $O(n^3)$ by an algorithm of Schrijver [22]. The second one is that of finding a MWCC of G^s with respect to the weight function w^s; we postpone to the full version of the paper the details, but this can be done in $O(|V(G^s)|^2)$-time. The overall complexity of this "translation" step is then $O(\sum_{s \in S} |V(G^s)|^2)$. By simple algebra, $\sum_{s \in S} |V(G^s)|^2 \leq (\sum_{s \in S} |V(G^s)|)^2$. But each $v \in V(G)$ belongs to at most two different graphs G^s, so $\sum_{s \in S} |V(G^s)| \leq 2|V(G)|$.

Our algorithm for the MWCC problem is summarized in the following: first compute a MWSS S of G, and then build a MWCC, as to run in $O(|V(G)|^3)$-time. The MWSS S can be computed in $O(|V(G)|^3)$-time (cfr. [9]). A non-negative, integer solution x to P_b defined by constraints (d1)-(d4) can be found in $O(|V(G)|^3)$-time, see the next section and Section 3.3. Note also that, differently from the unweighted case, this algorithm does not use augmenting paths techniques to build *concurrently* a MWSS and a MWCC: we do not push this augmenting paths approach, as it would result in a $O(|V(G)|^4)$-time algorithm (we defer the details to the journal version).

3 $Ax \leq b$ When A Satisfies (1), and b Is Integer

We are interested in the following problem: given an $m \times n$ matrix A satisfying (1) and an integer vector b, can one determine in polynomial time if the system $Ax \leq b$ has an integer solution (and build one if any)?

We associate to P_b (recall $P_b := \{x \in \mathbb{R}^n : Ax \leq b\}$) another polyhedron $Q_b \subseteq \mathbb{R}^{2n} := \{A' \begin{pmatrix} y \\ \bar{y} \end{pmatrix} \leq b'\}$ by associating inequalities to each inequality in the system $Ax \leq b$ as follows:

$$x_i + x_j \leq b_{ij} \rightarrow \begin{cases} y_i - \bar{y}_j \leq b_{ij} \\ -\bar{y}_i + y_j \leq b_{ij} \end{cases} \quad x_i \leq b_i \rightarrow y_i - \bar{y}_i \leq 2b_i$$

$$-x_i - x_j \leq b_{ij} \rightarrow \begin{cases} \bar{y}_i - y_j \leq b_{ij} \\ -y_i + \bar{y}_j \leq b_{ij} \end{cases} \quad 2x_i \leq b_i \rightarrow y_i - \bar{y}_i \leq b_i \quad -x_i \leq b_i \rightarrow -y_i + \bar{y}_i \leq 2b_i$$

$$x_i - x_j \leq b_{ij} \rightarrow \begin{cases} y_i - y_j \leq b_{ij} \\ -\bar{y}_i + \bar{y}_j \leq b_{ij} \end{cases} \quad -2x_i \leq b_i \rightarrow -y_i + \bar{y}_i \leq b_i$$

Lemma 1. *P_b has a solution if and only if Q_b has a solution.*

Proof. Necessity. Given a feasible solution $x^* \in P_b$, $(y^*, \bar{y}^*) : y_i^* = x_i^*, \bar{y}_i^* = -x_i^*$ for all i, is a solution to Q_b. *Sufficiency.* Given a feasible solution $(y^*, \bar{y}^*) \in Q_b$, $x^* : x_i^* = \frac{1}{2}y_i^* - \frac{1}{2}\bar{y}_i^*$ is a solution to P_b. We check it for the first type of inequality (i.e. to prove $x_i^* + x_j^* \leq b_{ij}$) but the method is the same for all 5 cases. If the inequality $x_i + x_j \leq b_{ij}$ is in the system $Ax \leq b$, by definition we have $y_i - \bar{y}_j \leq b_{ij}$ and $-\bar{y}_i + y_j \leq b_{ij}$ in the system defining Q_b. Taking the combination of those last two inequalities with multipliers $\frac{1}{2}$, $\frac{1}{2}$ yields $x_i^* + x_j^* \leq b_{ij}$. □

Observe that $(A')^t$ is a network matrix; we call D the corresponding (directed) graph, with cost b on its arcs. Any solution to Q_b defines what is usually called a feasible *potential* for D, and it follows from standard LP duality arguments that there is such a solution if and only if there are no negative cost cycles in D. In fact, given D, we can find in $O(nm)$-time either a feasible potential (integer potential as b is integer) or a negative cost cycle (see e.g. Theorem 7.7 in [17]).

For what follows, suppose therefore that Q_b has a feasible potential, i.e. $P_b \neq \emptyset$. The following lemma links the projection of P_b on each variable x_i, that we denote by $Proj_{x_i}(P_b)$, to the length of some suitable shortest paths in D, that e.g. can be computed in $O(mn + n^2 \log n)$-time by the algorithm of Moore-Bellman-Ford (see e.g. [17]).

Lemma 2. *If $P_b \neq \emptyset$, then $Proj_{x_i}(P_b) = [\frac{p_i}{2}, \frac{q_i}{2}]$, with q_i being the length of a shortest path from $\bar{y_i}$ to y_i in D (if any, else $q_i = \infty$), and $-p_i$ that of one from y_i to $\bar{y_i}$ (if any, else $-p_i = \infty$).*

Observe that if $\frac{p_i}{2}$ or $\frac{q_i}{2}$ are not integer values, then $x_i \geq \lceil \frac{p_i}{2} \rceil$ and $x_i \leq \lfloor \frac{q_i}{2} \rfloor$ are valid inequalities for the integer hull of P_b (recall that it is denoted by $Int(P_b)$). Therefore, if we are interested in the integer feasibility of P_b, i.e. if $Int(P_b)$ is empty or not, we can add those inequalities and define a new polyhedron $\overline{P}_b := P_b \cap \{x \in \mathbb{R}^n : \lceil \frac{p_i}{2} \rceil \leq x_i \leq \lfloor \frac{q_i}{2} \rfloor, i = 1, ..., n\}$ that is a tighter formulation for $Int(P_b)$.

Lemma 3. *Suppose that $P_b \neq \emptyset$. If, for each i, $\lceil \frac{p_i}{2} \rceil \leq \lfloor \frac{q_i}{2} \rfloor$, then $Int(\overline{P}_b) = Int(P_b) \neq \emptyset$ and we may find an integer solution to P_b in time $O(n^3)$.*

We would like to point out here that a slightly weaker result is implicit in Schrijver's approach (we defer the proof to the journal version of the paper).

Lemma 4. *$Int(P_b) \neq \emptyset$ if and only if, for each i, $Proj_{x_i}(P_b)$ has an integer point.*

The result is weaker than Lemma 3 in the sense that it does not tell us that the projection can be computed efficiently through shortest path (and actually Schrijver's approach does not even compute the projections). The next corollary follows from both lemmas (if we define $\overline{P}_b := P_b \cap \{x : \lceil \min_{x \in P_b} x_i \rceil \leq x_i \leq \lfloor \max_{x \in P_b} x_i \rfloor, \forall i = 1, ...n\}$ when using Lemma 4).

Corollary 1. *$\overline{P}_b = \emptyset$ if and only if $Int(\overline{P}_b) = Int(P_b) \neq \emptyset$.*

We close this section by linking with the results from the Constraint Logic Programming community. Because we can compute the transitive closure by shortest path calculation in D (this is immediate by definition of the transitive closure), our result also shows that we can compute the tight closure in time $O(n^2 log n + nm)$ (we apply the shortest path calculation twice). This is essentially the approach proposed in [24].

3.1 A Weak Edmonds-Johnson Property for Matrices A Satisfying (1)

Given a polyhedron $P = \{x \in \mathbb{R}^n : Ax \leq b\}$, we denote by P' its Chvátal-Gomory closure (or CG-closure), that is, the polytope obtained by adding to the system $Ax \leq b$ all its Chvatál-Gomory cuts (i.e., inequalities of the form $cx \leq \lfloor \delta \rfloor$, where c is an integer vector and $cx \leq \delta$ holds for each point in P.

A rational matrix A has the *Edmonds-Johnson property* if, for all d_1, d_2, b_1, b_2 integer vectors, the integer hull of

$$P = \{x \in \mathbb{R}^n : d_1 \leq x \leq d_2, b_1 \leq Ax \leq b_2\} \tag{2}$$

is given by P'. Edmonds and Johnson [7,8] proved that if $A = (a_{ij})$ is an integral $m \times n$-matrix such that $\sum_{i=1}^{m} |a_{ij}| \leq 2$ for all $j = 1, .., n$, then A has

the Edmonds-Johnson property. As shown by Gerards and Schrijver [11], the property does not hold when passing to transpose i.e. when A satisfies (1) as illustrated by taking A to be the edge-vertex incidence matrix of K_4 and then considering the system $0 \leq x \leq 1, 0 \leq Ax \leq 1$ (note that this is the linear relaxation of the edge formulation of the stable set polytope of K_4). Indeed it is easily proved that two rounds of Chvátal-Gomory cuts are needed in this case (one to produce all triangle inequalities, and one to produce the facet $x(V(K_4)) \leq 1$). In some sense, Gerards and Schrijver [11] prove that the converse holds i.e. matrix A satisfying (1) has the Edmonds-Johnson property if and only if it is the edge-vertex incidence matrix of a bidirected graph with no odd K_4-subdivision (see [11] for a proper definition). Moreover, in this case, optimizing over the integer hull of system (2) is easy, by the ellipsoid method, see [11] for more details; note that, if we only assume condition (1), there is no hope (unless $P = NP$) to optimize in polynomial time over the integer hull of (2), as one may encode the stable set problem.

We here define a *weaker* notion of Edmonds-Jonson property, that is mainly concerned with integer feasibility (recall that P' denotes the CG-closure of P):

Definition 1. *A rational matrix A has the* weak *Edmonds-Johnson property if, for all integer vectors d_1, d_2, b_1, b_2, the polyhedron $P = \{x \in \mathbb{R}^n : d_1 \leq x \leq d_2, b_1 \leq Ax \leq b_2\}$ has an integer solution if and only if P' is non-empty.*

By definition, the Edmonds-Johnson property implies the *weak* Edmonds-Johnson one, but the converse is not true. For instance, the edge-vertex incidence matrix of K_4 does not have the Edmonds-Johnson property but it has the weak Edmonds-Johnson one. In fact, we show in the following, *every* matrix A satisfying (1) has the property.

Theorem 1. *Every integral matrix B such that, for each i, $\sum_j |\overline{b}_{ij}| \leq 2$ has the weak Edmonds-Johnson property.*

Proof. Let A be a matrix satisfying (1). For each integer vector b, consider the polyhedron $P_b = \{x \in \mathbb{R}^n : Ax \leq b\}$. Without loss of generality, assume that $P_b \neq \emptyset$. We know from Corollary 1 that $Int(P_b) \neq \emptyset$ if and only if $\overline{P}_b \neq \emptyset$, where $\overline{P}_b := P_b \cap \{x : \left\lceil \min_{x \in P_b} x_i \right\rceil \leq x_i \leq \left\lfloor \max_{x \in P_b} x_i \right\rfloor, \forall i = 1, ...n\}$. Observe that $(P_b)' \subseteq \overline{P}_b$, as the inequalities $\lceil \min_{x \in P_b} x_i \rceil \leq x_i \leq \lfloor \max_{x \in P_b} x_i \rfloor$, are special CG-cuts for P_b. Therefore, $IP_b \neq \emptyset$ if and only if $(P_b)' \neq \emptyset$. The statement follows by observing that $\{x \in \mathbb{R}^n : d_1 \leq x \leq d_2, b_1 \leq Bx \leq b_2\}$ can be rewritten as $\{x \in \mathbb{R}^n : Ax \leq b\}$, with $b = (b_2, -b_1, d_2, -d_1)^t$, and $A = (B, -B, I, -I)^t$ satisfying (1). □

3.2 When CG-Cuts Are Not Needed

We would like to understand now under which conditions we do not need to add Chvátal-Gomory inequalities to P_b to ensure that fractional feasibility implies integer feasibility.

Observe that in the proof of Lemma 1, we retrieve a solution x of P_b from a solution $(y, \bar{y}) \in Q_b$ by taking a simple convex combination of the values y_i and $-\bar{y}_i$ (with multipliers $\frac{1}{2}$). We could try to see if other "convex combinations" yield valid solutions. For this purpose, we define $\Pi_A = \{\lambda \in [0,1]^q : A_2\lambda \leq \frac{A_2\mathbf{1}}{2}\}$, where A_2 is the submatrix of A made of those rows with $\sum_j |a_{ij}| = 2$. Observe that $\frac{A_2\mathbf{1}}{2} \in \{0, 1, -1\}^q$ and by definition $\lambda = \frac{\mathbf{1}}{2}$ is a feasible solution to Π_A. The system Π_A is made of inequalities of the type $\lambda_i - \lambda_j \leq 0, \lambda_i + \lambda_j \leq 1$, $-\lambda_i - \lambda_j \leq -1$, $2\lambda_i \leq 1$ and $-2\lambda_i \leq -1$. If we are interested in integer solution of Π_A, the last two restrictions impose $\lambda_i = 0$ and $\lambda_i = 1$ respectively. We call $\overline{\Pi_A}$ the polyhedra obtained from Π_A by substituting the restrictions $2\lambda_i \leq 1$ and $-2\lambda_i \leq -1$ with $\lambda_i = 0$ and $\lambda_i = 1$ respectively. All inequalities in $\overline{\Pi_A}$ can be rewritten under the form $\lambda_i + (1 - \lambda_j) \leq 1, \lambda_i + \lambda_j \leq 1, (1 - \lambda_i) + (1 - \lambda_j) \leq 1$, $\lambda_i \leq 0$ or $-\lambda_i \leq -1$ and thus $\overline{\Pi_A}$ can be trivially identified with the linear relaxation associated with the standard integer programming formulation of a 2-SAT instance. We have therefore:

Lemma 5. *Π_A has an integer solution if and only if the corresponding 2-SAT instance is satisfiable.*

The latter claim has the following nice consequence.

Lemma 6. *If Π_A has an integer solution, then P_b has an integer solution if and only if P_b is non-empty.*

The proof of Lemma 6 (that we postpone to the full version of the paper) shows that, when Q_b is non-empty, and we are given an integer solution λ to Π_A, one may always build an integer solution to P_b by (essentially) solving a single shortest path calculation. We sum-up the results obtained in the following:

Theorem 2. *If one knows a priori that Π_A has an integer solution, one can build an integer solution to P_b by solving a single shortest path problem and a single 2-SAT instance.*

Observe that any matrix A which is TU has the property that Π_A has an integer solution. This follows from the fact that A_2 is a submatrix of A and it is thus also TU, and that Π_A has the fractional solution $\frac{\mathbf{1}}{2}$. Interestingly, there are other 0,+/-1 matrices, that are not TU, that satisfy this property. For instance the matrix $A = \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}$. In general though, the fact that P_b has a integer solution does not imply that Π_A has one (consider for instance the system defined by the relations $x_1 + x_2 = 2, x_2 + x_3 = 2, x_3 + x_1 = 2$). However if we ask the property for all vector b and all subsystems (in the spirit of the definition of TU matrices), the converse holds, i.e. π_A has an integer solution, as Π_A is a special subsystem of P_b with $b = \frac{A\mathbf{1}}{2}$. We are currently investigating a proper definition of this kind to extend total unimodularity to the integer feasibility question, as we did for the weak Edmonds-Johnson property. We defer this to the journal version of the paper.

3.3 Back to Minimum Weighted Clique Cover

In the previous section we identified a class of systems that have a fractional solution if and only if they have an integral one. We now show that this class includes the systems arising from the MWCC problem.

We therefore go back to the algorithm in Section 2.2. So let S be a MWSS of a claw-free perfect graph G. We want to compute a non-negative, integer solution x to the system P_b defined by constraints $(d1)$-$(d4)$. Now let us give a look at the corresponding Π_A. Because we only keep those rows with two non-zero elements per row, Π_A reads:

$$\begin{aligned} \lambda_{us} + \lambda_{vs} &\leq 1, \forall s \in S, u, v \in N(s), uv \notin E \\ \lambda_{vs} + \lambda_{vs'} &\geq 1, \forall v \text{ bound, where } s, s' \text{ are the vertices in } S \cap N(v) \\ \lambda &\in [0,1]^q \end{aligned}$$

Now if there exists an integer solution to this system, there exists one with $\lambda_{vs} = 0$ for all v free (those vertices are only involved in the first type of constraints). Thus, integer feasibility for Π_A reduces to the existence of integer solutions to:

$$\begin{aligned} \lambda_{us} + \lambda_{vs} &\leq 1, \forall s \in S, u, v \in N(s), uv \notin E, u, v \text{ bound} \\ \lambda_{vs} + \lambda_{vs'} &\geq 1, \forall v \text{ bound, where } s, s' \text{ are the vertices in } S \cap N(v) \\ \lambda &\in [0,1]^n \end{aligned}$$

Note that this latter system has an integral solution if and only if there exists a clique cover of size $|S|$ in the graph $G[V \setminus F]$, where F is the set of the vertices that are free with respect to S. But this is trivially the case, as in $G[V \setminus F]$ there are no free vertices, and therefore no augmenting paths.

References

1. Aspvall, B., Plass, M., Tarjan, R.: A linear-time algorithm for testing the truth of certain quantified boolean formulas. Inf. Process. Lett. 8(3), 121–123 (1979)
2. Bagnara, R., Hill, P.M., Zaffanella, E.: An Improved Tight Closure Algorithm for Integer Octagonal Constraints. In: Logozzo, F., Peled, D.A., Zuck, L.D. (eds.) VMCAI 2008. LNCS, vol. 4905, pp. 8–21. Springer, Heidelberg (2008)
3. Berge, C.: Graphs and Hypergraphs. Dunod, Paris (1973)
4. Bonomo, F., Oriolo, G., Snels, C.: Minimum Weighted Clique Cover on Strip-Composed Perfect Graphs. In: Golumbic, M.C., Stern, M., Levy, A., Morgenstern, G. (eds.) WG 2012. LNCS, vol. 7551, pp. 22–33. Springer, Heidelberg (2012)
5. Chvátal, V.: On certain polytopes associated with graphs. J. Combin. Theory, Ser. B 18(2), 138–154 (1975)
6. Chvátal, V., Sbihi, N.: Recognizing claw-free perfect graphs. J. Combin. Theory, Ser. B 44, 154–176 (1988)
7. Edmonds, J., Johnson, E.: Matching: a well-solved class of integer linear programs. In: Guy, R., Hanani, H., Sauer, N., Schönheim, J. (eds.) Combinatorial Structures and Their Applications, pp. 89–92. Gordon and Breach, New York (1970)
8. Edmonds, J., Johnson, E.: Matching, Euler tours and the Chinese postman. Math. Program. 5, 88–124 (1973)

9. Faenza, Y., Oriolo, G., Stauffer, G.: An algorithmic decomposition of claw-free graphs leading to an $O(n^3)$-algorithm for the weighted stable set problem. In: Randall, D. (ed.) Proc. 22nd SODA, San Francisco, CA, pp. 630–646 (2011)
10. Fulkerson, D.: On the perfect graph theorem. In: Hu, T., Robinson, S. (eds.) Mathematical Programming, pp. 69–76. Academic Press, New York (1973)
11. Gerards, A., Schrijver, A.: Matrices with the Edmonds-Johnson property. Combinatorica 6(4), 365–379 (1986)
12. Grötschel, M., Lovász, L., Schrijver, A.: Geometric Algorithms and Combinatorial Optimization. Springer, Berlin (1988)
13. Harvey, W., Stuckey, P.: A unit two variable per inequality integer constraint solver for constraint logic programming. In: The 20th Australasian Computer Science Conference, Sydney, Australia. Australian Computer Science Communications, pp. 102–111 (1997)
14. Hsu, W., Nemhauser, G.: Algorithms for minimum covering by cliques and maximum clique in claw-free perfect graphs. Discrete Math. 37, 181–191 (1981)
15. Hsu, W., Nemhauser, G.: A polynomial algorithm for the minimum weighted clique cover problem on claw-free perfect graphs. Discrete Math. 38(1), 65–71 (1982)
16. Jaffar, J., Maher, M.J., Stuckey, P.J., Yap, R.H.C.: Beyond Finite Domains. In: PPCP 1994. LNCS, vol. 874, pp. 86–94. Springer, Heidelberg (1994)
17. Korte, B., Vygen, J.: Combinatorial Optimization: Theory and Algorithms. Springer, Berlin (2011)
18. Lahiri, S.K., Musuvathi, M.: An Efficient Decision Procedure for UTVPI Constraints. In: Gramlich, B. (ed.) FroCos 2005. LNCS (LNAI), vol. 3717, pp. 168–183. Springer, Heidelberg (2005)
19. Maffray, F., Reed, B.: A description of claw-free perfect graphs. J. Combin. Theory, Ser. B 75(1), 134–156 (1999)
20. Padberg, M.: Perfect zero-one matrices. Math. Program. 6, 180–196 (1974)
21. Peis, B.: Structure Analysis of Some Generalizations of Matchings and Matroids under Algorithmic Aspects. PhD thesis, Universität zu Köln (2007)
22. Schrijver, A.: Disjoint homotopic paths and trees in a planar graph. Discrete Comput. Geom. 6(1), 527–574 (1991)
23. Schrijver, A.: Combinatorial Optimization. Polyhedra and Efficiency (3 volumes). Algorithms and Combinatorics, vol. 24. Springer, Berlin (2003)
24. Schutt, A., Stuckey, P.: Incremental satisfiability and implication for UTVPI constraints. INFORMS J. Comput. 22(4), 514–527 (2010)
25. Seshia, S., Subramani, K., Bryant, R.: On solving Boolean combinations of UTVPI constraints. J. Satisf. Boolean Model. Comput. 3, 67–90 (2007)
26. Tarjan, R.: Depth-first search and linear graph algorithms. SIAM J. Comput. 1(2), 146–160 (1972)
27. Whitesides, S.: A method for solving certain graph recognition and optimization problems, with applications to perfect graphs. In: Berge, C., Chvátal, V. (eds.) Topics on Perfect Graphs, North-Holland. North-Holland Mathematics Studies, vol. 88, pp. 281–297 (1984)

A Complexity and Approximability Study of the Bilevel Knapsack Problem

Alberto Caprara[1], Margarida Carvalho[2], Andrea Lodi[1], and Gerhard J. Woeginger[3]

[1] DEI, University of Bologna, Italy
[2] Departamento de Ciências de Computadores, Universidade do Porto, Portugal
[3] Department of Mathematics, TU Eindhoven, Netherlands

Abstract. We analyze three fundamental variants of the bilevel knapsack problem, which all are complete for the second level of the polynomial hierarchy. If the weight and profit coefficients in the knapsack problem are encoded in unary, then two of the bilevel variants are solvable in polynomial time, whereas the third is NP-complete. Furthermore we design a polynomial time approximation scheme for this third variant, whereas the other two variants cannot be approximated in polynomial time within any constant factor (assuming P≠NP).

Bilevel and Multilevel Optimization. In bilevel optimization the decision variables are split into two groups that are controlled by two decision makers called *leader* (on the upper level) and *follower* (on the lower level). Both decision makers have an objective function of their own and a set of constraints on their variables. Furthermore there are coupling constraints that connect the decision variables of leader and follower. The decision making process is as follows. First the leader makes his decision and fixes the values of his variables, and afterwards the follower reacts by setting his variables. The leader has perfect knowledge of the follower's scenario (objective function and constraints) and also of the follower's behavior. The follower observes the leader's action, and then optimizes his own objective function subject to the decisions made by the leader (and subject to the imposed constraints). As the leader's objective function does depend on the follower's decision, the leader must take the follower's reaction into account.

Bilevel and multilevel optimization have received much interest in the literature over the last decades; see for instance the books by Migdalas, Pardalos & Värbrand [15] and Dempe [3]. Multilevel optimization problems are extremely difficult from the computational point of view and cannot be expressed in terms of classical integer programs (which can only handle a single level of optimization). A ground-breaking paper by Jeroslow [11] established that various multilevel problems are complete for various levels of the polynomial hierarchy in computational complexity theory; see Papadimitriou [16] for more information. Further hardness results for broad families of multilevel optimization problems are due to Deng [6] and Dudás, Klinz & Woeginger [7].

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 98–109, 2013.

Standard Knapsack Problems and Bilevel Knapsack Problems. An instance of the knapsack problem consists of a set of items with given weights and profits together with a knapsack with a given weight capacity. The objective is to select a subset of the items with maximum total profit, subject to the constraint that the overall selected item weight must fit into the knapsack. The knapsack problem is well-known to be NP-complete [10].

Over the last few years, a variety of authors has studied certain bilevel variants of the knapsack problem. Dempe & Richter [4] considered the variant where the leader controls the weight capacity of the knapsack, and where the follower decides which items are packed into the knapsack. Mansi, Alves, de Carvalho & Hanafi [14] consider a bilevel knapsack variant where the item set is split into two parts, one of which is controlled by the leader and one controlled by the follower. DeNegre [5] suggests yet another variant, where both players have a knapsack on their own; the follower can only choose from those items that the leader did not pack. Section 1 gives precise definitions of these three variants and provides further information on them.

Our Contributions. We pinpoint the computational complexity of the three bilevel knapsack variants mentioned above: they are complete for the complexity class Σ_2^p and hence located at the second level of the polynomial hierarchy. If a problem is Σ_2^p-complete, there is no way of formulating it as a single-level integer program of polynomial size *unless the polynomial hierarchy collapses* (a highly unlikely event which would cause a revolution in complexity theory). The complexity class Σ_2^p is the natural hotbed for bilevel problems that are built on top of NP-complete single-level problems; as a rule of thumb, the bilevel version of an NP-complete problem should always be expected to be Σ_2^p-complete.

In a second line of investigation, we study these bilevel problems under *unary* encodings. The classical knapsack problem becomes polynomially solvable if the input is encoded in unary, and it is only natural to expect a similar behavior from our bilevel knapsack problems. Indeed, two of our three bilevel variants become polynomially solvable if the input is encoded in unary, and thus show exactly the type of behavior that one would expect from a knapsack variant. The third variant behaves differently and stubbornly becomes NP-complete.

Our third line of results studies the approximability of the three bilevel knapsack variants. As a rule of thumb Σ_2^p-hard problems do not allow good approximation algorithms. Indeed, the literature only contains negative results in this direction that establish the inapproximability of various Σ_2^p-hard optimization problems; see Ko & Lin [12] and Umans [18]. Two of our bilevel knapsack variants (actually the same ones that are easy under unary encodings) behave exactly as expected and do not allow polynomial time approximation algorithms with finite worst case guarantee, assuming P$\neq$NP. For the third variant, however, we derive a polynomial time approximation scheme. This is the first approximation scheme for a Σ_2^p-hard optimization problem in the history of approximation algorithms, and from the technical point of view it is the most sophisticated result in this paper.

Our investigations provide a complete and clean picture of the complexity landscape of the considered bilevel knapsack problems. We expect that our results will also be useful in classifying and understanding other bilevel problems, and that our hardness proofs will serve as stepping stones for future results.

Organization of the Paper. Section 1 defines the three bilevel knapsack variants and summarizes the literature on them. Section 2 presents the Σ_2^p-completeness results for these problems (under the standard binary encoding) and also discusses their behavior under unary encodings. Section 3 discusses the approximability and inapproximability behavior of the considered bilevel problems.

1 Definitions and Preliminaries

In bilevel optimization the follower observes the leader's action, and then optimizes his own objective function value subject to the decisions made by the leader and subject to the imposed constraints. This statement does not fully determine the follower's behavior: there might be many feasible solutions that all are optimal for the follower but yield different objective values for the leader. Which one will the follower choose? In the *optimistic* scenario the follower always picks the optimal solution that yields the *best* objective value for the leader, and in the *pessimistic* scenario he picks the solution that yields the *worst* objective value for the leader. All our negative (hardness) results and all our positive (polynomial time) results hold for the optimistic scenario as well as for the pessimistic scenario.

In the following subsections, we use x and $x_1, \ldots, x_m$ to denote the variables controlled by the leader, and $y_1, \ldots, y_n$ to denote the variables controlled by the follower. Furthermore we use a_i, b_i, c_i and A, B, C, C' to denote item profits, item weights, cost coefficients, upper bounds, and lower bounds; all these numbers are non-negative integers (or rationals). As usual, we use the notation $a(I) = \sum_{i \in I} a_i$ for an index set I, and $a(x) = \sum_i a_i x_i$ for a 0-1 vector x.

1.1 The Dempe-Richter (DR) Variant

The first occurrence of a bilevel knapsack problem in the optimization literature seems to be due to Dempe & Richter [4]. In their problem variant DR as depicted in Figure 1, the leader controls the capacity x of the knapsack while the follower controls all items and decides which of them are packed into the knapsack. The objective function of the leader depends on the knapsack capacity x as well as on the packed items, whereas the objective function of the follower solely depends on the packed items.

All decision variables in this bilevel program are integers; the knapsack capacity satisfies $x \in \mathbb{Z}$ and the variables $y_1, \ldots, y_n \in \{0, 1\}$ encode whether item i is packed into the knapsack ($y_i = 1$) or not ($y_i = 0$). We note that in the original model in [4] the knapsack capacity x is continuous; one nasty consequence

$$\text{Maximize} \quad f_1(x,y) \;=\; Tx + \sum_{i=1}^{n} a_i y_i \tag{1a}$$

$$\text{subject to} \quad C \;\le\; x \;\le\; C' \tag{1b}$$

$$\text{where } y_1,\ldots,y_n \text{ solves the follower's problem}$$

$$\max \sum_{i=1}^{n} b_i y_i \quad \text{s.t.} \quad \sum_{i=1}^{n} b_i y_i \le x \tag{1c}$$

Fig. 1. The bilevel knapsack problem DR

of this continuous knapsack capacity is that the problem (1a)–(1c) may fail to have an optimal solution. The computational complexity of the problem remains the same, no matter whether x is integral or continuous.

Dempe & Richter [4] discuss approximation algorithms for DR, and furthermore design a dynamic programming algorithm that solves variant DR in pseudo-polynomial time. Brotcorne, Hanafi & Mansi [1] derive another (simpler) dynamic program with a much better running time.

1.2 The Mansi-Alves-de-Carvalho-Hanafi (MACH) Variant

Mansi, Alves, de Carvalho & Hanafi [14] consider a bilevel knapsack variant where both players pack items into the knapsack. There is a single common knapsack for both players with a prespecified capacity of C. The item set is split into two parts, which are respectively controlled by the leader and the follower. The leader starts the game by packing some of his items into the knapsack, and then the follower adds some further items from his set. Figure 2 specifies the bilevel problem MACH. The 0-1 variables $x_1,\ldots,x_m$ (for the leader) and $y_1,\ldots,y_n$ (for the follower) encode whether item i is packed into the knapsack.

Mansi, Alves, de Carvalho & Hanafi [14] describe several applications of their problem in revenue management, telecommunication, capacity allocation, and

$$\text{Maximize} \quad f_2(x,y) \;=\; \sum_{j=1}^{m} a_j x_j + \sum_{i=1}^{n} a'_i y_i \tag{2a}$$

$$\text{subject to} \quad y_1,\ldots,y_n \text{ solves the follower's problem}$$

$$\max \sum_{i=1}^{n} b'_i y_i \quad \text{s.t.} \quad \sum_{i=1}^{n} c'_i y_i \le C - \sum_{j=1}^{m} c_j x_j \tag{2b}$$

Fig. 2. The bilevel knapsack problem MACH

transportation. Variant MACH has also been studied in a more general form by Brotcorne, Hanafi & Mansi [2], who reduced the model to one-level in pseudo-polynomial time.

1.3 The DeNegre (DN) Variant

DeNegre [5] proposes another bilevel knapsack variant where both players hold their own private knapsacks and choose items from a common item set. First the leader packs some of the items into his private knapsack, and then the follower picks some of the remaining items and packs them into his private knapsack. The objective of the follower is to maximize the profit of the items in his knapsack, and the objective of the hostile leader is to minimize this profit.

$$\text{Minimize} \quad f_3(x,y) = \sum_{i=1}^{n} b_i y_i \tag{3a}$$

$$\text{subject to} \quad \sum_{i=1}^{n} a_i x_i \le A \tag{3b}$$

$$\text{where } y_1,\dots,y_n \text{ solves the follower's problem}$$

$$\max \sum_{i=1}^{n} b_i y_i \quad \text{s.t.} \sum_{i=1}^{n} b_i y_i \le B \quad \text{and} \tag{3c}$$

$$y_i \le 1 - x_i \quad \text{for } 1 \le i \le n \tag{3d}$$

Fig. 3. The bilevel knapsack problem DN

Figure 3 depicts the bilevel problem DN. The 0-1 variables $x_1,\dots,x_n$ (for the leader) and $y_1,\dots,y_n$ (for the follower) encode whether the corresponding item is packed into the knapsack. The interdiction constraint $y_i \le 1 - x_i$ in (3d) enforces that the follower cannot take item i once the leader has picked it. Note that leader and follower have exactly opposing objectives.

2 Hardness Results

As usual, we consider the decision versions corresponding of our optimization problems: "Does there exist an action of the leader that makes his objective value at least as good as some given bound?" Theorem 1 summarizes the results under the standard binary encoding; its proof follows from the fact that all decision problems are in the class Σ_2^p (see Chapter 17 in Papadimitriou's book [16]) and by reductions from the decision problem Subset-Sum-Interval, which has been proved to be Σ_2^p-complete by Eggermont & Woeginger [8].

Theorem 1. *The decision versions of (a) DR, (b) MACH, and (c) DN in binary encoding are Σ_2^p-complete.*

If the input data is encoded in unary, the corresponding problem variants unary-DR and unary-MACH are solvable in polynomial time by dynamic programming. These results are routine and perfectly expected, and their proofs use as main tool the polynomial time algorithm for the standard knapsack problem under unary encodings (see Garey & Johnson [10]). The third variant unary-DN is much more interesting, as it turns out to be NP-complete. Our reduction is from the VERTEX-COVER problem in undirected graphs; see [10].

Problem: VERTEX-COVER

Instance: An undirected graph $G = (V, E)$; an integer bound t.

Question: Does G possess a vertex cover of size t, that is, a subset $T \subseteq V$ such that every edge in E has at least one of its vertices in T?

A *Sidon sequence* is a sequence $s_1 < s_2 < \cdots < s_n$ of positive numbers in which all pairwise sums $s_i + s_j$ with $i < j$ are different. Erdős & Turán [9] showed that for any odd prime p, there exists a Sidon sequence of p integers that all are below $2p^2$. The argument in [9] is constructive and yields a simple polynomial time algorithm for finding Sidon sequences of length n whose elements are bounded by $O(n^2)$.

We start our polynomial time reduction from an arbitrary instance $G = (V, E)$ and k of VERTEX-COVER. Let $n = |V| \geq 10$, and let $v_1, \ldots, v_n$ be an enumeration of the vertices in V. We construct a Sidon sequence $s_1 < s_2 < \cdots < s_n$ whose elements are polynomially bounded in n. We define $S = \sum_{i=1}^{n} s_i$ as the sum of all numbers in the Sidon sequence, and we construct the following instance of DN as specified in (3a)–(3d).

- For every vertex v_i, we create a corresponding vertex-item with leader's weight $a(v_i) = 1$ and follower's weight $b(v_i) = S + s_i$.
- For every edge $e = [v_i, v_j]$, we create a corresponding edge-item with leader's weight $a(e) = t + 1$ and follower's weight $b(e) = 5S - s_i - s_j$.
- The capacity of the leader's knapsack is $A = t$, and the capacity of the follower's knapsack is $B = 7S$.

We claim that in the DN instance the leader can make his objective value $\leq 7S-1$ if and only if the VERTEX-COVER instance has answer YES.

(Proof of if). Assume that there exists a vertex cover T of size $|T| = t$. Then a good strategy for the leader is to put the t vertex-items that correspond to vertices in T into his knapsack, which fills his knapsack of capacity $A = t$ to the limit. Suppose for the sake of contradiction that afterwards the follower can still fill his knapsack with total weight $7S$. Then the follower must pick at least one edge-item (he can pack at most six vertex-items, and their weight would stay strictly below $7S$). Furthermore the follower cannot pick two edge-items (since every edge-item has weight greater than $4S$). Consequently the follower must pick exactly one edge-item that corresponds to some edge $e = [v_i, v_j]$.

The remaining space in the follower's knapsack is $2S + s_i + s_j$ and must be filled by two vertex-items. By the definition of a Sidon sequence, the only way of doing this would be by picking the two vertex-items corresponding to v_i and v_j. But that's impossible, as at least one of the vertices v_i and v_j is in the cover T so that the item has already been picked by the leader. This contradiction shows that the follower cannot reach an objective value of $7S$.

(Proof of only if). Now let us assume that the graph G does not possess any vertex cover of size t, and let us consider the game right after the move of the leader. Since the leader can pack at most t vertex-items, there must exist some edge $e = [v_i, v_j]$ in E for which the leader has neither picked the item corresponding to v_i nor the item corresponding to v_j. Then the follower may pick the vertex-item v_i, the vertex-item v_j, and the edge-item e, which brings him a total weight of $7S$.

Theorem 2. *The decision version of the bilevel problem DN in unary encoding is NP-complete, both for the optimistic scenario and the pessimistic scenario.*

Proof. The above construction can be performed in polynomial time. As the elements in the Sidon sequence are polynomially bounded in $|V|$, also their sum S and all the integers in our construction are polynomially bounded in $|V|$. In particular, this yields that the unary encoding length of the constructed DN instance is polynomially bounded in $|V|$. Together with the above arguments, this implies that DN in unary encoding is NP-hard.

To show containment of DN under unary encoding in class NP, we use the optimal move of the leader as NP-certificate. The certificate is short, as it just specifies a subset of the items. To verify the certificate, we have to check that the follower cannot pick any item set of high weight. Since all weights are encoded in unary, this checking amounts to solving a standard knapsack problem in unary encoding, which can be done in polynomial time. □

3 Approximability and Inapproximability

The Σ_2^p-completeness proofs for DR and MACH have devastating consequences in terms of existence of a polynomial time approximation for them: it is Σ_2^p-hard to distinguish the DR instances in which the leader can reach an objective value of 1 from those DR instances in which the leader can only reach objective value 0. An analogous statement holds for problem MACH. As a polynomial time approximation algorithm with finite worst case guarantee would be able to distinguish between these two instance types, we get the following result.

Corollary 1. *Problems DR and MACH do not possess a polynomial time approximation algorithm with finite worst case guarantee, unless P=Σ_2^p and therefore P=NP holds.* □

The statement in Corollary 1 is not surprising, as the literature on the approximability of Σ_2^p-hard optimization problems entirely consists of such negative

statements that show the inapproximability of various problems; see Ko & Lin [12] and Umans [18]. The following theorem breaks with this old tradition, and presents the first approximation scheme for a Σ_2^p-hard optimization problem.

Theorem 3. *Problem DN has a polynomial time approximation scheme.*

The rest of this section is dedicated to the proof of Theorem 3. We apply and extend a number of rounding tricks from the seminal paper [13] by Lawler, we use approximation schemes from the literature as a black box, and we also add a number of new ingredients and rounding tricks.

Throughout the proof we will consider a fixed instance of problem DN. Without loss of generality we assume that no item i in the instance satisfies $b_i > B$: such items could never be used by the follower, and hence are irrelevant and may as well be ignored. Let ε with $0 < \varepsilon < 1/3$ be a small positive real number; for the sake of simplicity we will assume that the reciprocal value $1/\varepsilon$ is integer.

Our global goal is to determine in polynomial time a feasible solution for the leader that yields an objective value of at most $(1+\varepsilon)^4$ times the optimum. This will be done by a binary search over the range $0, 1, \ldots, B$ that (approximately) sandwiches the optimal objective value between a lower and an upper bound. Whenever we bisect the search interval between these bounds at some value U, we have to decide whether the optimal objective value lies below or above U. If the optimal objective value lies below U, then Lemma 5 (derived in Section 3.1) and Lemma 6 (derived in Section 3.2) show how to find and how to verify in polynomial time an approximate solution for the leader whose objective value is bounded by $(1+\varepsilon)^3\, U$. If these lemmas succeed then we make U the new upper bound. If the lemmas fail to produce an approximate objective value of at most $(1+\varepsilon)^3\, U$, then we make U the new lower bound. The binary search process terminates as soon as the upper bound comes within a factor of $1+\varepsilon$ of the lower bound. Note that we then lose a factor of $1+\varepsilon$ between upper and lower bound, and that we lose a factor of at most $(1+\varepsilon)^3$ by applying the lemmas. All in all, this yields the desired approximation guarantee of $(1+\varepsilon)^4$ and completes the proof of Theorem 3.

3.1 How to Handle the Central Cases

Throughout this section, we assume that U is an upper bound on the optimal objective value of the considered instance with

$$B/2 \;\le\; U \;\le\; B/(1+\varepsilon). \tag{4}$$

The items $i = 1, \ldots, n$ are partitioned according to their b-values into so-called *large* items that satisfy $U < b_i$, into *medium* items that satisfy $\varepsilon U < b_i \le U$, and into *small* items that satisfy $b_i \le \varepsilon U$. We denote by L, M, S respectively the set of large, medium, small items. Furthermore a medium item i belongs to class $\mathcal{C}_k$, if it satisfies

$$k\varepsilon^2 U \;\le\; b_i \;<\; (k+1)\varepsilon^2 U.$$

Note that only classes $\mathcal{C}_k$ with $1/\varepsilon \leq k \leq 1/\varepsilon^2$ play a role in this classification. By (4) the overall size of $2/\varepsilon$ medium items exceeds the capacity of the follower's knapsack, so that the follower uses at most $2/\varepsilon$ medium items in his solution.

In the following we analyze two scenarios. In the first scenario, the solution x^* for the leader and the solution y^* for the follower both will carry a superscript*. The sets of large, medium, small items packed by x^* into the leader's knapsack will be denoted respectively by L_x^*, M_x^*, S_x^*, and the corresponding sets for y^* and the follower are denoted L_y^*, M_y^*, S_y^*. In the second scenario we use analogous notations with the superscript$^\#$. The first scenario is centered around an optimal solution x^* for the leader. The second scenario considers another feasible solution $x^\#$ for the leader that we call the *aligned* version of x^*.

- Solution $x^\#$ packs all large items into the knapsack; hence $L_x^\# = L$.
- Solution $x^\#$ selects the following items from class $\mathcal{C}_k$: it picks an item $i \in M_x^* \cap \mathcal{C}_k$ if and only if $\mathcal{C}_k - M_x^*$ contains at most $2/\varepsilon$ items j with $b_j \leq b_i$. (By this choice, the $2/\varepsilon$ items with smallest b-value in $\mathcal{C}_k - M_x^*$ coincide with the $2/\varepsilon$ items with smallest b-value in $\mathcal{C}_k - M_x^\#$.) Note that $M_x^\# \subseteq M_x^*$.
- For the small items we first determine a $(1+\varepsilon)$-approximate solution to the following auxiliary problem (Aux): find a subset $Z \subseteq S$ of the small items that minimizes $b(Z)$, subject to the covering constraint $a(Z) \geq a(L_x^\# \cup M_x^\#) + a(S) - A$. Solution $x^\#$ then packs the complementary set $S_x^\# = S - Z$.

This completes the description of $x^\#$, which is easily seen to be a feasible action for the leader. Note that also the optimal solution x^* packs all the large items, as otherwise the follower could pack a large item and thereby push the objective value above the bound U. Then $L_x^\# = L_x^*$ and $M_x^\# \subseteq M_x^*$ imply $a(L_x^* \cup M_x^*) \geq a(L_x^\# \cup M_x^\#)$, which yields

$$A \;\geq\; a(L_x^* \cup M_x^* \cup S_x^*) \;\geq\; a(L_x^\# \cup M_x^\#) + a(S_x^*). \tag{5}$$

As $a(S_x^*) = a(S) - a(S - S_x^*)$, we conclude from (5) that the set $S - S_x^*$ satisfies the covering constraint in the auxiliary problem (Aux). Hence the optimal objective value of (Aux) is upper bounded by $b(S - S_x^*)$, and any $(1+\varepsilon)$-approximate solution Z to (Aux) must satisfy $b(Z) \leq (1+\varepsilon)\, b(S - S_x^*)$, which is equivalent to

$$b(S - S_x^\#) \;\leq\; (1+\varepsilon)\, b(S - S_x^*). \tag{6}$$

The following lemma demonstrates that the aligned solution $x^\#$ is almost as good for the leader as the underlying optimal solution x^*.

Lemma 4. *If the leader uses the aligned solution $x^\#$, then every feasible reaction $y^\#$ for the follower yields an objective value $f_3(x^\#, y^\#) \leq (1+2\varepsilon)\, U$.*

Proof. Suppose for the sake of contradiction that there exists a reaction $y^\#$ for the follower that yields an objective value of $f_3(x^\#, y^\#) > (1+2\varepsilon)\, U$. Based on $y^\#$ we will construct another solution y^* for the follower in the first scenario:

- Solution y^* does not use any large item; hence $L_y^* = \emptyset$.
- Solution y^* picks the same number of items from every class $\mathcal{C}_k$ as $y^\#$ does. It avoids items in x^* and selects the $|\mathcal{C}_k \cap M_y^\#|$ items in $\mathcal{C}_k - M_x^*$ that have the smallest b-values.

– Finally we add small items from $S - S_x^*$ to the follower's knapsack, until no further item fits or until we run out of items.

Solution $y^\#$ packs at most $2/\varepsilon$ medium items, and hence uses at most $2/\varepsilon$ items from $\mathcal{C}_k$. By our choice of medium items for $x^\#$ we derive $b(\mathcal{C}_k \cap M_y^*) \leq b(\mathcal{C}_k \cap M_y^\#)$ for every k, which implies

$$b(M_y^*) \;\leq\; b(M_y^\#) \;\leq\; B. \tag{7}$$

Solution y^* only selects items that are not used by x^*, and inequality (7) implies that all the selected items indeed fit into the follower's knapsack. Hence y^* constitutes a feasible reaction of the follower if the leader chooses x^*.

Next, let us quickly go through the item types. First of all neither solution y^* nor solution $y^\#$ can use any large item, so that we have

$$b(L_y^*) \;=\; b(L_y^\#) \;=\; 0. \tag{8}$$

For the medium items, the ratio between the smallest b-value and the largest b-value in class $\mathcal{C}_k$ is at least $k/(k+1) \geq 1-\varepsilon$. Hence we certainly have $b(\mathcal{C}_k \cap M_y^*) \geq (1-\varepsilon)\, b(\mathcal{C}_k \cap M_y^\#)$, which implies

$$b(M_y^*) \;\geq\; (1-\varepsilon)\, b(M_y^\#). \tag{9}$$

Let us turn to the small items. Suppose that y^* cannot accommodate all small items from $S - S_x^*$ in the follower's knapsack. Then some small item i with $b_i < \varepsilon U$ does not fit, which with (4) leads to $b(y^*) > B - \varepsilon\, U \geq U$. As this violates our upper bound U on the optimal objective value, we conclude that y^* accommodates all such items and satisfies $S_y^* = S - S_x^*$. This relation together with (6) and the disjointness of the sets $S_x^\#$ and $S_y^\#$ yields

$$b(S_y^*) \;=\; b(S - S_x^*) \;\geq\; \frac{b(S - S_x^\#)}{1+\varepsilon} \;\geq\; \frac{b(S_y^\#)}{1+\varepsilon} \;>\; (1-\varepsilon)\, b(S_y^\#). \tag{10}$$

Now let us wrap things up. If the leader chooses x^*, the follower may react with the feasible solution y^* and get an objective value

$$\begin{aligned} f_3(x^*, y^*) &= b(L_y^*) + b(M_y^*) + b(S_y^*) \\ &> (1-\varepsilon)\, b(L_y^\#) + (1-\varepsilon)\, b(M_y^\#) + (1-\varepsilon)\, b(S_y^\#) \\ &= (1-\varepsilon)\, f_3(x^\#, y^\#) \;>\; (1-\varepsilon)(1+2\varepsilon)\, U \;>\; U. \end{aligned}$$

Here we used the estimates in (8), (9), and (10). As this objective value violates the upper bound U, we have reached the desired contradiction. □

Lemma 5. *Given an upper bound U on the objective value that satisfies (4), one can compute in polynomial time a feasible solution x for the leader, such that every reaction y of the follower has $f_3(x, y) \leq (1+\varepsilon)^3\, U$.*

Proof. If we did not only know the bound U but also an optimal solution x^*, then we could simply determine the corresponding aligned solution $x^\#$ and apply Lemma 4. We will bypass this lack of knowledge by checking many candidates for the set $M_x^\#$. Let us recall how the aligned solution $x^\#$ picks medium items from class $\mathcal{C}_k$.

- If $|\mathcal{C}_k - M_x^*| \leq 2/\varepsilon$ then $M_x^\# \cap \mathcal{C}_k = M_x^* \cap \mathcal{C}_k$. Note that there are only $O(|\mathcal{C}_k|^{2/\varepsilon})$ different candidates for $M_x^\# \cap \mathcal{C}_k$.
- If $|\mathcal{C}_k - M_x^*| > 2/\varepsilon$ then $M_x^\# \cap \mathcal{C}_k$ is a subset of M_x^*; an item i from $M_x^* \cap \mathcal{C}_k$ enters $M_x^\#$ if there are at most $2/\varepsilon$ items $j \in \mathcal{C}_k - M_x^*$ with $b_j \leq b_i$. Note that $M_x^\# \cap \mathcal{C}_k$ is fully determined by the $2/\varepsilon$ items with smallest b-value in $\mathcal{C}_k - M_x^*$. As there are only $O(|\mathcal{C}_k|^{2/\varepsilon})$ ways for choosing these $2/\varepsilon$ items, there are only $O(|\mathcal{C}_k|^{2/\varepsilon})$ different candidates for $M_x^\# \cap \mathcal{C}_k$.

Altogether there are only $O(|\mathcal{C}_k|^{2/\varepsilon})$ ways of picking the medium items from class $\mathcal{C}_k$. As every class satisfies $|\mathcal{C}_k| \leq n$ and as there are only $1/\varepsilon^2$ classes to consider, we get a polynomial number $O(n^{2/\varepsilon^3})$ of possibilities for choosing the set $M_x^\#$ in the aligned solution. Summarizing, we only need to check a polynomial number of candidates for set $M_x^\#$.

How do we check such a candidate $M_x^\#$? The aligned solution always uses $L_x^\# = L$, and the auxiliary problem (Aux) is fully determined once $M_x^\#$ and $L_x^\#$ have been fixed. We approximate the auxiliary problem by standard methods (see for instance Pruhs & Woeginger [17]), and thus also find the set $S_x^\#$ in polynomial time. This yields the full corresponding aligned solution $x^\#$. It remains to verify the quality of this aligned solution for the leader, which amounts to analyzing the resulting knapsack problem at the follower's level. We use one of the standard approximation schemes for knapsack as for instance described by Lawler [13], and thereby get a $(1+\varepsilon)$-approximate solution for the follower's problem.

While checking and scanning through the candidates, we eventually must hit a good candidate $M_x^\#$ that yields the correct aligned version x of an optimal solution. By Lemma 4 the corresponding objective value $f_3(x, y)$ is bounded by $(1+2\varepsilon)\,U$. Then the approximation scheme finds an objective value of at most $(1+\varepsilon)(1+2\varepsilon)\,U \leq (1+\varepsilon)^3 U$. This completes the proof of the lemma. □

3.2 How to Handle the Boundary Cases

Finally let us discuss the remaining cases where U does not satisfy the bounds in (4). The first case $U > B/(1+\varepsilon)$ is trivial, as the objective value never exceeds the follower's knapsack capacity B; hence in this case the objective value will always stay below $(1+\varepsilon)\,U$. The second case $U < B/2$ is settled by the following lemma; its proof is based on the framework of Pruhs & Woeginger [17] and can be found in the long version of this paper.

Lemma 6. *Given an upper bound $U < B/2$ on the objective value, one can compute in polynomial time a feasible solution x for the leader, such that every reaction y of the follower has $f_3(x, y) \leq (1+\varepsilon)\,U$.* □

References

1. Brotcorne, L., Hanafi, S., Mansi, R.: A dynamic programming algorithm for the bilevel knapsack problem. Operations Research Letters 37, 215–218 (2009)
2. Brotcorne, L., Hanafi, S., Mansi, R.: One-level reformulation of the bilevel knapsack problem using dynamic programming. Technical Report, Université de Valenciennes et du Hainaut-Cambrésis, France (2011)
3. Dempe, S.: Foundations of Bilevel Programming. Kluwer Academic Publishers, Dordrecht (2002)
4. Dempe, S., Richter, K.: Bilevel programming with Knapsack constraint. Central European Journal of Operations Research 8, 93–107 (2000)
5. DeNegre, S.: Interdiction and discrete bilevel linear programming. Ph.D. dissertation, Lehigh University (2011)
6. Deng, X.: Complexity issues in bilevel linear programming. In: Migdalas, A., Pardalos, P.M., Värbrand, P. (eds.) Multilevel Optimization: Algorithms and Applications, pp. 149–164. Kluwer Academic Publishers, Dordrecht (1998)
7. Dudás, T., Klinz, B., Woeginger, G.J.: The computational complexity of multi-level bottleneck programming problems. In: Migdalas, A., Pardalos, P.M., Värbrand, P. (eds.) Multilevel Optimization: Algorithms and Applications, pp. 165–179. Kluwer Academic Publishers, Dordrecht (1998)
8. Eggermont, C., Woeginger, G.J.: Motion planning with pulley, rope, and baskets. In: Proceedings of the 29th International Symposium on Theoretical Aspects of Computer Science, STACS 2012. Leibniz International Proceedings in Informatics, vol. 14, pp. 374–383 (2012)
9. Erdős, P., Turán, P.: On a problem of Sidon in additive number theory, and on some related problems. Journal of the London Mathematical Society 16, 212–215 (1941)
10. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, San Francisco (1979)
11. Jeroslow, R.: The polynomial hierarchy and a simple model for competitive analysis. Mathematical Programming 32, 146–164 (1985)
12. Ko, K., Lin, C.-L.: On the complexity of min-max optimization problems and their approximation. In: Du, D.-Z., Pardalos, P.M. (eds.) Minimax and Applications, pp. 219–239. Kluwer Academic Publishers, Dordrecht (1995)
13. Lawler, E.L.: Fast approximation algorithms for knapsack problems. Mathematics of Operations Research 4, 339–356 (1979)
14. Mansi, R., Alves, C., de Carvalho, J.M.V., Hanafi, S.: An exact algorithm for bilevel 0-1 knapsack problems. Mathematical Problems in Engineering, Article ID 504713 (2012)
15. Migdalas, A., Pardalos, P.M., Värbrand, P.: Multilevel Optimization: Algorithms and Applications. Kluwer Academic Publishers, Dordrecht (1998)
16. Papadimitriou, C.H.: Computational Complexity. Addison-Wesley (1994)
17. Pruhs, K., Woeginger, G.J.: Approximation schemes for a class of subset selection problems. Theoretical Computer Science 382, 151–156 (2007)
18. Umans, C.: Hardness of approximating $\sum_2^{\mathrm{p}}$ minimization problems. In: Proceedings of the 40th Annual Symposium on Foundations of Computer Science, FOCS 1999, pp. 465–474 (1999)

Matroid and Knapsack Center Problems*

Danny Z. Chen[1], Jian Li[2], Hongyu Liang[2], and Haitao Wang[3]

[1] Department of Computer Science and Engineering
University of Notre Dame, Notre Dame, IN 46556, USA
dchen@cse.nd.edu
[2] Institute for Interdisciplinary Information Sciences (IIIS)
Tsinghua University, Beijing 100084, China
{lijian83@mail,lianghy08@mails}.tsinghua.edu.cn
[3] Department of Computer Science
Utah State University, Logan, UT 84322, USA
haitao.wang@usu.edu

Abstract. In the classic k-center problem, we are given a metric graph, and the objective is to open k nodes as centers such that the maximum distance from any vertex to its closest center is minimized. In this paper, we consider two important generalizations of k-center, the matroid center problem and the knapsack center problem. Both problems are motivated by recent content distribution network applications. Our contributions can be summarized as follows:

1. We consider the matroid center problem in which the centers are required to form an independent set of a given matroid. We show this problem is NP-hard even on a line. We present a 3-approximation algorithm for the problem on general metrics. We also consider the outlier version of the problem where a given number of vertices can be excluded as the outliers from the solution. We present a 7-approximation for the outlier version.
2. We consider the (multi-)knapsack center problem in which the centers are required to satisfy one (or more) knapsack constraint(s). It is known that the knapsack center problem with a single knapsack constraint admits a 3-approximation. However, when there are at least two knapsack constraints, we show this problem is not approximable at all. To complement the hardness result, we present a polynomial time algorithm that gives a 3-approximate solution such that one knapsack constraint is satisfied and the others may be violated by at most a factor of $1 + \epsilon$. We also obtain a 3-approximation for the outlier version that may violate the knapsack constraint by $1 + \epsilon$.

1 Introduction

The k-center problem is a fundamental facility location problem. In the basic version, we are given a metric space (V, d) and are asked to locate a set $\mathcal{S} \subseteq V$ of at most k vertices as centers and to assign the other vertices to the centers, so as to minimize the maximum distance from any vertex to its assigned center, or more formally, to minimize $\max_{v \in V} \min_{u \in \mathcal{S}} d(v, u)$. In the *demand* version of the k-center problem, each vertex v

* This work was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61061130540, 61073174, 61202009. The research of D.Z. Chen was supported in part by NSF under Grants CCF-0916606 and CCF-1217906.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 110–122, 2013.

has a positive demand $r(v)$, and our goal is to minimize the maximum weighted distance from any vertex to the centers, i.e., $\max_{v \in V} \min_{u \in S} r(v)d(v,u)$. It is well known that the k-center problem is NP-hard and admits a polynomial time 2-approximation even for the demand version [13,16], and that no polynomial time $(2-\epsilon)$-approximation algorithm exists unless $\mathrm{P} = \mathrm{NP}$ [13].

In this paper, we initiate a systematic study on two generalizations of the k-center problem and their variants. The first one is the *matroid center* problem, denoted by MatCenter, which is almost the same as the k-center problem except that, instead of the cardinality constraint on the set of centers, now the centers are required to form an independent set of a given matroid. A finite matroid $\mathcal{M}$ is a pair $(V, \mathcal{I})$, where V is a finite set (called the *ground set*) and $\mathcal{I}$ is a collection of subsets of V. Each element in $\mathcal{I}$ is called an *independent set*. Moreover, $\mathcal{M} = (V, \mathcal{I})$ satisfies the following three properties: (1) $\emptyset \in \mathcal{I}$; (2) if $A \subseteq B$ and $B \in \mathcal{I}$, then $A \in \mathcal{I}$; and (3) for all $A, B \in \mathcal{I}$ with $|A| > |B|$ there exists an element $e \in A \setminus B$ such that $B \cup \{e\} \in \mathcal{I}$. Following conventions in the literature, we assume the matroid $\mathcal{M}$ is given by an independence oracle which, given a subset $S \subseteq V$, decides whether $S \in \mathcal{I}$. For more information about the theory of matroids, see, e.g., [25].

The second problem we study is the *knapsack center* problem (denoted as KnapCenter), another generalization of k-center in which the chosen centers are subject to (one or more) knapsack constraints. More formally, in KnapCenter, there are m non-negative weight functions $w_1, \ldots, w_m$ on V, and m weight budgets $\mathcal{B}_1, \ldots, \mathcal{B}_m$. Let $w_i(V') := \sum_{v \in V'} w_i(v)$ for all $V' \subseteq V$. A solution opens a set of vertices $\mathcal{S} \subseteq V$ as centers such that $w_i(\mathcal{S}) \leq \mathcal{B}_i$ for all $1 \leq i \leq m$. The objective is still to minimize the maximum service cost of any vertex in V (the service cost of v equals $\min_{c \in \mathcal{S}} d(v,c)$, or $\min_{c \in \mathcal{S}} r(v)d(v,c)$ in the demand version). In this paper, we are only interested in the case where the number m of knapsack constraints is a constant. We note that the special case with only one knapsack constraint was studied in [17] under the name of weighted k-center, which already generalizes the basic k-center problem.

Both MatCenter and KnapCenter are motivated by important applications in content distribution networks [15,20]. In a content distribution network, there are several types of servers and a set of clients to be connected to the servers. Often there is a budget constraint on the number of deployed servers of each type [15]. We would like to deploy a set of servers subject to these budget constraints in order to minimize the maximum service cost of any client. The budget constraints correspond to finding an independent set in a partition matroid.[1] We can also use a set of knapsack constraints to capture the budget constraints for all types (we need one knapsack constraint for each type). Motivated by such applications, Hajiaghayi et al. [15] first studied the red-blue median problem in which there are two types (red and blue) of facilities, and the goal is to deploy at most k_r red facilities and k_b blue facilities so as to minimize the sum of service costs. Subsequently, Krishnaswamy et al. [20] introduced a more general *matroid median* problem which seeks to open a set of facilities that is an independent set in the given matroid and the *knapsack median* problem in which the set of facilities

[1] Let $B_1, B_2, \ldots, B_b$ be a collection of disjoint subsets of V and d_i be integers such that $1 \leq d_i \leq |B_i|$ for all $1 \leq i \leq b$. We say a set $I \subseteq V$ is independent if $|I \cap B_i| \leq d_i$ for $1 \leq i \leq b$. All such independent sets form a partition matroid.

must satisfy a knapsack constraint. The work mentioned above uses the sum of service costs as the objective (the k-median objective), while our work tries to minimize the maximum services cost (the k-center objective), which is another popular objective in the clustering and network design literature.

1.1 Our Results

For MatCenter, we show the problem is NP-hard to approximate within a factor of $2 - \epsilon$ for any constant $\epsilon > 0$, even on a line. Note that the k-center problem on a line can be solved exactly in polynomial time [5]. We present a 3-approximation algorithm for MatCenter on general metrics, which improves the constant factors implied by the approximation algorithms for matroid median [20,3].

Next, we consider the outlier version of MatCenter, denoted as Robust-MatCenter, where one can exclude at most $n - p$ nodes as outliers. We obtain a 7-approximation for Robust-MatCenter. Our algorithm is a nontrivial generalization of the greedy algorithm due to Charikar et al. [2], which only works for the outlier version of basic k-center. However, their algorithm and analysis do not extend to our problem. In their analysis, if at least p nodes are covered by k disks (with radius $3 \cdot \mathsf{OPT}$), they have found the set of k centers and obtained a 3-approximation. However, in our case, we may not be able to open enough centers in the covered region, due to the matroid constraint. Therefore, we need to search the centers globally. To this end, we carefully construct two matroids and argue their intersection provides a desirable answer (the construction is similar to that for the non-outlier version, but more involved).

We next deal with the KnapCenter problem. We show that for any $f > 0$, the existence of an f-approximation algorithm for KnapCenter with more than one knapsack constraint implies $\mathrm{P} = \mathrm{NP}$. This is a sharp contrast to the case with only one knapsack constraint, for which a 3-approximation exists [17] and is known to be optimal [8]. Given this strong inapproximability result, it is then natural to ask whether efficient approximation algorithms exist if we are allowed to slightly violate the constraints. We answer this question affirmatively. We provide a polynomial time algorithm that, given an instance of KnapCenter with a constant number of knapsack constraints, returns a 3-approximate solution that is guaranteed to satisfy one constraint and violate each of the others by at most a factor of $1 + \epsilon$ for any fixed $\epsilon > 0$. This generalizes the result of [17] to the multi-constraint case. Our algorithm also works for the demand version.

We then consider the outlier version of KnapCenter, which we denote by Robust-KnapCenter. We obtain a 3-approximation for Robust-KnapCenter that violates the knapsack constraint by a factor of $1 + \epsilon$ for any fixed $\epsilon > 0$. Our algorithm can be regarded as a "weighted" version of the greedy algorithm due to Charikar et al. [2] which only works for the unit-weight case. Since their charging argument does not apply to the weighted case, we instead adopt a more involved algebraic approach. We translate our algorithm into inequalities involving point sets, and then directly manipulate the inequalities to establish the approximation ratio. The total weight of our chosen centers may exceed the budget by the maximum weight of any client, which can be turned into a $1 + \epsilon$ multiplicative factor by the partial enumeration technique.

Due to space limitations, some proofs and details are omitted and can be found in the full version of this paper [6].

1.2 Related Work

For the basic k-center problem, Hochbaum and Shmoys [16,17] and Gonzalez [13] developed 2-approximation algorithms, which are the best possible if P $\neq$ NP [13]. The former algorithms are based on the idea of the threshold method, which originates from [11]. On some special metrics like the shortest path metrics on trees, k-center (with or without demands) can typically be solved in polynomial time by dynamic programming. By exploring additional structures of the metrics, even linear or quasi-linear time algorithms can be obtained; see e.g. [5,9,12] and the references therein. Several generalizations and variations of k-center have also been studied in a variety of application contexts; see, e.g. [1,23,18,4,10,19].

A problem closely related to k-center is the well-known k-median problem, whose objective is to minimize the sum of service costs of all nodes instead of the maximum one. Hajiaghayi et al. [15] introduced the red-blue median problem that generalizes k-median, and presented a constant factor approximation base on local search. Krishnaswamy et al. [20] introduced the more general matroid median problem and presented a 16-approximation algorithm based on LP rounding, whose ratio was improved to 9 by Charikar and Li [3] using a more careful rounding scheme. Another generalization of k-median is the knapsack median problem studied by Kumar [21], which requires to open a set of centers with total weight at most a specified value. Kumar gave a (large) constant factor approximation for knapsack median, which was improved by Charikar and Li [3] to a 34-approximation. Several other classical problems have also been investigated recently under matroid or knapsack constraints, such as minimum spanning tree [28], maximum matching [14], and submodular maximization [22,26].

For the k-center formulation, it is well known that a few distant vertices (outliers) can disproportionately affect the final solution. Such outliers may significantly increase the cost of the solution, without improving the level of service to the majority of clients. To deal with outliers, Charikar et al. [2] initiated the study of the robust versions of k-center and other related problems, in which a certain number of points can be excluded as outliers. They gave a 3-approximation for robust k-center, and showed that the problem with forbidden centers (i.e., some points cannot be centers) is inapproximable within $3 - \epsilon$ unless P = NP. For robust k-median they presented a bicriteria approximation algorithm that returns a $4(1 + 1/\epsilon)$-approximate solution in which the number of excluded outliers may violate the upper bound by a factor of $1 + \epsilon$. Later, Chen [7] gave a truly constant factor approximation (with a very large constant) for the robust k-median problem. McCutchen and Khuller [24] and Zarrabi-Zadeh and Mukhopadhyay [27] considered the robust k-center problem in a streaming context.

2 The Matroid Center Problem

In this section we consider the matroid center problem and its outlier version. A useful ingredient of our algorithms is the *(weighted) matroid intersection* problem defined as follows. We are given two matroids $\mathcal{M}_1(V, \mathcal{I}_1)$ and $\mathcal{M}_2(V, \mathcal{I}_2)$ defined on the same ground set V. Each element $v \in V$ has a weight $w(v) \geq 0$. The goal is to find a common independent set S in the two matroids, i.e., $S \in \mathcal{I}_1 \cap \mathcal{I}_2$, such that the total weight $w(S) = \sum_{v \in S} w(v)$ is maximized. It is well known that this problem can be solved in polynomial time (e.g., see [25]).

Algorithm 1. Algorithm for MatCenter on G_i

1 Initially, $C \leftarrow \emptyset$, and mark all vertices in V as uncovered.
2 **while** V *contains uncovered vertices* **do**
3 Pick an uncovered vertex v. Set $\mathsf{B}(v) \leftarrow \mathsf{B}(v, d(e_i))$ and $C \leftarrow C \cup \{v\}$.
4 Mark all vertices in $\mathsf{B}(v, 2d(e_i))$ as covered.
5 **end**
6 Define a partition matroid $\mathcal{M}_{\mathsf{B}} = (V, \mathcal{I})$ with partition $\{\{\mathsf{B}(v)\}_{v \in C}, V \setminus \cup_{v \in C} \mathsf{B}(v)\}$ (note that $\{\mathsf{B}(v)\}_{v \in C}$ are disjoint sets), where $\mathcal{I}$ is the set of subsets of V that contains at most 1 element from every $\mathsf{B}(v)$ and 0 element from $V \setminus \cup_{v \in C} \mathsf{B}(v)$.
7 Solve the unweighted (or, unit-weight) matroid intersection problem between $\mathcal{M}_{\mathsf{B}}$ and $\mathcal{M}$ to get an optimal intersection $\mathcal{S}$. If $|\mathcal{S}| < |C|$, then we declare a failure and try the next G_i. Otherwise, we succeed and return $\mathcal{S}$ as the set of centers.

2.1 NP-Hardness of Matroid Center on a Line

In contrast to the basic k-center problem on a line which can be solved in near-linear time [5], we show the following theorem whose proof can be found in the full paper [6].

Theorem 1. *It is* NP*-hard to approximate* MatCenter *on a line within a factor strictly better than 2, even when the given matroid is a partition matroid.*

2.2 A 3-Approximation for MatCenter

In fact, we can obtain a constant approximation for MatCenter by using the constant approximation for the matroid median problem [20,3], which roughly gives a 9-approximation for MatCenter. The details will appear in the full paper [6].

We next present a 3-approximation for MatCenter, thus improving the ratio derived from the matroid median algorithms [20,3]. Also, compared to their LP-based algorithms, ours is simpler, purely combinatorial, and very easy to implement. We begin with the description of our algorithm. Regard the metric space as a (complete) graph $G = (V, E)$ where each edge $\{u, v\}$ has length $d(u, v)$. Let $\mathsf{B}(v, r)$ be the set of vertices that are at most r unit distance away from v (it depends on the underlying graph). Let $e_1, e_2, \ldots, e_{|E|}$ be the edges in a non-decreasing order of their lengths. We consider each spanning subgraph G_i of G that contains only the first i edges. We run Algorithm 1 on each G_i and take the best solution. It is easy to see that $\mathsf{B}(u) \cap \mathsf{B}(v) = \emptyset$ for any distinct $u, v \in C$.

Theorem 2. *Algorithm 1 produces a* 3*-approximation for* MatCenter.

Proof. Suppose the maximum radius of any cluster in an optimal solution is r^* and a set of optimal centers is C^*. Consider the algorithm on G_i with $d(e_i) = r^*$ (r^* must be the length of some edge). First we claim that there exists an intersection of $\mathcal{M}$ and $\mathcal{M}_{\mathsf{B}}$ of size $|C|$. In fact, we show there is a subset of C^* that is such an intersection. For each node u, let $a(u)$ be an optimal center in C^* that is at most $d(e_i)$ away from u. Consider the set $\mathcal{S}^* = \{a(u)\}_{u \in C}$. Since $\mathcal{S}^*$ is a subset of C^*, it is an independent set of $\mathcal{M}$ by the definition of matroid. It is also easy to see that $a(u) \in \mathsf{B}(u)$ for each $u \in C$. Therefore, $\mathcal{S}^*$ is also independent in $\mathcal{M}_{\mathsf{B}}$, which proves our claim. Thus, the

Algorithm 2. Algorithm for Robust-MatCenter on G_i

1 Initially, set $C \leftarrow \emptyset$ and mark all vertices in V as uncovered.
2 **while** V *contains uncovered vertices* **do**
3 Pick an uncovered vertex v such that $\mathsf{B}(v, d(e_i))$ covers the most number of uncovered elements.
4 $\mathsf{B}(v) \leftarrow \mathsf{B}(v, d(e_i))$. ($\mathsf{B}(v)$ is called the disk of v.)
5 $\mathsf{E}(v) \leftarrow \mathsf{B}(v, 3d(e_i)) \setminus \cup_{u \in C} \mathsf{E}(u)$. ($\mathsf{E}(v)$ is called the expanded disk of v. This definition ensures that all expanded disks in $\{\mathsf{E}(u)\}_{u \in C}$ are pairwise disjoint.)
6 $C \leftarrow C \cup \{v\}$. Mark all vertices in $\mathsf{E}(v)$ as covered.
7 **end**
8 Create a set U of (vertex, expanded disk) pairs, as follows: For each $v \in V$ and $u \in C$, if $\mathsf{B}(v, d(e_i)) \cap \mathsf{B}(u, 3d(e_i)) \neq \emptyset$, we add $(v, \mathsf{E}(u))$ to U. The weight $w((v, \mathsf{E}(u)))$ of the pair $(v, \mathsf{E}(u))$ is $|\mathsf{E}(u)|$.
9 Define two matroids $\mathcal{M}_1$ and $\mathcal{M}_2$ over U as follows:
– A subset $\{(v_i, \mathsf{E}(u_i))\}$ is independent in $\mathcal{M}_1$ if all v_i's in the subset are distinct and form an independent set in $\mathcal{M}$.
– A subset $\{(v_i, \mathsf{E}(u_i))\}$ is independent in $\mathcal{M}_2$ if all $\mathsf{E}(u_i)$'s in the subset are distinct. (It is easy to see $\mathcal{M}_2$ is a partition matroid.)
10 Solve the matroid intersection problem between $\mathcal{M}_1$ and $\mathcal{M}_2$ optimally (note that the independence oracles for $\mathcal{M}_1$ and $\mathcal{M}_2$ can be easily simulated in polynomial time). Let $\mathcal{S}$ be an optimal intersection. If $w(\mathcal{S}) < p$, then we declare a failure and try the next G_i. Otherwise, we succeed and return $V(\mathcal{S})$ as the set of centers, where $V(\mathcal{S}) = \{v \mid (v, \mathsf{E}(u)) \in \mathcal{S} \text{ for some } u \in C\}$.

algorithm returns a set $\mathcal{S}$ that contains exactly 1 element from each $\mathsf{B}(v)$ with $v \in C$. According to the algorithm, for each $v \in V$ there exists $u \in C$ that is at most $2d(e_i)$ away, and this u is within distance $d(e_i)$ from the (unique) element in $\mathsf{B}(u) \cap \mathcal{S}$. Thus every node of V is within a distance $3d(e_i) = 3r^*$ from some center in $\mathcal{S}$. □

2.3 Dealing with Outliers: Robust-MatCenter

We now consider the outlier version of MatCenter, denoted as Robust-MatCenter, in which an additional parameter p is given and the goal is to place centers (which must form an independent set) such that after excluding at most $|V| - p$ nodes as outliers, the maximum service cost of any node is minimized. For $p = |V|$, we have the standard MatCenter. In this section, we present a 7-approximation for Robust-MatCenter.

Our algorithm bears some similarity to the 3-approximation algorithm for robust k-center by Charikar et al. [2], who also showed that robust k-center with forbidden centers cannot be approximated within $3-\epsilon$ unless P = NP. However, their algorithm for robust k-center does not directly yield any approximation ratio for the forbidden center version. In fact, robust k-center with forbidden centers is a special case of Robust-MatCenter since forbidden centers can be easily captured by a partition matroid. We briefly describe the algorithm in [2]. Assume we have guessed the right optimal radius r. For each $v \in V$, call $\mathsf{B}(v, r)$ the *disk* of v and $\mathsf{B}(v, 3r)$ the *expanded disk* of v. Repeat the following step k times: Pick an uncovered vertex as a center such that its disk covers the most number of uncovered nodes, then mark all nodes in the corresponding

expanded disk as covered. Using a clever charging argument they showed that at least p nodes can be covered, which gives a 3-approximation. However, their algorithm and analysis do not extend to our problem in a straightforward manner. The reason is that even if at least p nodes are covered, we may not be able to open enough centers in the covered region due to the matroid constraint. In order to remedy this issue, we need to search for centers in the entire graph, which also necessitates a more careful charging argument to show that we can cover at least p nodes.

Now we describe our algorithm and prove its performance guarantee. For each $1 \leq i \leq \binom{|V|}{2}$, we run Algorithm 2 on the graph G_i defined as before. It is easy to prove that $\mathcal{M}_1$ is a matroid (details will be given in the full version of this paper [6]).

Theorem 3. *Algorithm 2 produces a* 7*-approximation for* Robust-MatCenter.

Proof. Assume the maximum radius of any cluster in an optimal solution is r^* and the set of optimal centers is C^*. For each $v \in C^*$, let $\mathsf{O}(v)$ denote the optimal disk $\mathsf{B}(v, r^*)$. As before, we claim that our algorithm succeeds if $d(e_i) = r^*$. It suffices to show the existence of an intersection of $\mathcal{M}_1$ and $\mathcal{M}_2$ with weight at least p. We next construct such an intersection $\mathcal{S}'$ from the optimal center set C^*. The high level idea is as follows. Let the disk centers in C be $v_1, v_2, \ldots, v_k$ (according to the order that our algorithm chooses them). (Note that these are the centers chosen by the greedy procedure in the first part of the algorithm, but not those returned at last.) We process these centers one by one. Initially, $\mathcal{S}'$ is empty. As we process a new center v_j, we may add $(v, \mathsf{E}(v_j))$ for some $v \in C^*$ to $\mathcal{S}'$. Moreover, we charge each newly covered node in any optimal disk to some nearby node in the expanded disk $\mathsf{E}(v_j)$. (Note that this is the key difference between our charging argument and that of [2]; in [2], a node may be charged to some node far away.) We maintain that all nodes in $\cup_{v \in C^*} \mathsf{O}(v)$ covered by $\cup_{j'=1}^{j} \mathsf{E}(v_{j'})$ are charged after processing v_j. Thus, eventually, all nodes in $\cup_{v \in C^*} \mathsf{O}(v)$ are charged. We also make sure that each node in any expanded disk in $\mathcal{S}'$ is charged to at most once. Therefore, the weight of $\mathcal{S}'$ is at least $|\cup_{v \in C^*} \mathsf{O}(v)| \geq p$.

Now, we present the details of the construction of $\mathcal{S}'$. If every node in $\mathsf{O}(v)$ for some $v \in C^*$ is charged, we say $\mathsf{O}(v)$ is *entirely charged*. Consider the step when we process $v_j \in C$. We distinguish the following cases.

Case 1: Suppose there is a node $v \in C^*$ such that $\mathsf{O}(v)$ is not entirely charged and $\mathsf{O}(v)$ intersects $\mathsf{B}(v_j)$. Then add $(v, \mathsf{E}(v_j))$ to $\mathcal{S}'$ (if there are multiple such v's, we only add one of them). We charge the newly covered nodes in $\cup_{v \in C^*} \mathsf{O}(v)$ (i.e., the nodes in $(\cup_{v \in C^*} \mathsf{O}(v)) \cap \mathsf{E}(v_j)$) to themselves (we call this charging rule I). Note that $\mathsf{O}(v)$ is entirely charged after this step since $\mathsf{O}(v) \subseteq \mathsf{B}(v_j, 3r^*)$.

Case 2: Suppose $\mathsf{B}(v_j)$ does not intersect $\mathsf{O}(v)$ for any $v \in C^*$, but there is some node $v \in C^*$ such that $\mathsf{O}(v)$ is not entirely charged and $\mathsf{O}(v) \cap \mathsf{E}(v_j) \neq \emptyset$. Then we add $(v, \mathsf{E}(v_j))$ to S' and charge all newly covered nodes in $\mathsf{O}(v)$ (i.e., the node in $\mathsf{O}(v) \cap \mathsf{E}(v_j)$) to $\mathsf{B}(v_j)$ (we call this charging rule II). Since $\mathsf{B}(v_j)$ covers the most number of uncovered elements when v_j is added, there are enough vertices in $\mathsf{B}(v_j)$ to charge. Obviously, $\mathsf{O}(v)$ is entirely charged after this step. If there is some other node $u \in C^*$ such that $\mathsf{O}(u)$ is not entirely charged and $\mathsf{O}(u) \cap \mathsf{E}(v_j) \neq \emptyset$, then we charge each newly covered node (i.e., nodes in $\mathsf{O}(u) \cap \mathsf{E}(v_j)$) in $\mathsf{O}(u)$ to itself using rule I.

Case 3: If $\mathsf{E}(v_j)$ does not intersect with any optimal disk $\mathsf{O}(v)$ that is not entirely charged, then we simply skip this iteration and continue to the next v_j.

It is easy to see that all covered nodes in $\cup_{v\in C^*}\mathsf{O}(v)$ are charged in the process and each node is charged to at most once. Indeed, consider a node u in $\mathsf{B}(v_j)$. If $\mathsf{B}(v_j)$ intersects some $\mathsf{O}(v)$, then u may be charged by rule I and, in this case, no further node can be charged to u again. If $\mathsf{B}(v_j)$ does not intersect any $\mathsf{O}(v)$, then u may be charged by rule II. This also happens at most once. It is obvious that in this case, no node can be charged to u using rule I. For a node $u \in \mathsf{E}(v_j) \setminus \mathsf{B}(v_j)$, it can be charged at most once using rule I. Moreover, by the charging process, all nodes in $\cup_{v\in C^*}\mathsf{O}(v)$ are charged to the nodes in some expanded disks that appear in $\mathcal{S}'$. Therefore, the total weight of $\mathcal{S}$ is at least p. We can see that each vertex in $V(\mathcal{S}')$ is also in C^* and appears at most one. Therefore, $\mathcal{S}'$ is independent in $\mathcal{M}_1$. Clearly, each $\mathsf{E}(u)$ appears in $\mathcal{S}'$ at most once. Hence, $\mathcal{S}'$ is also independent in $\mathcal{M}_2$, which proves our claim.

Since $\mathcal{S}$ is an optimal intersection, we know the expanded disks in $\mathcal{S}$ contain at least p nodes. By the requirement of $\mathcal{M}_1$, we can guarantee that the set of centers form an independent set in $\mathcal{M}$. For each $(v, \mathsf{E}(u))$ in $\mathcal{S}$, we can see that every node v' in $\mathsf{E}(u)$ is within a distance $7d(e_i)$ from v as follows. Suppose $u' \in \mathsf{B}(v, d(e_i)) \cup \mathsf{B}(u, 3d(e_i))$ (because $\mathsf{B}(v, d(e_i)) \cup \mathsf{B}(u, 3d(e_i)) \neq \emptyset$ for any pair $(v, \mathsf{E}(u)) \in U$). By triangle inequality, $d(v', v) \leq d(v', u)+d(u, u')+d(u', v) \leq 3d(e_i)+3d(e_i)+d(e_i) = 7d(e_i)$. This completes the proof of the theorem. □

3 The Knapsack Center Problem

In this section we study the KnapCenter problem and its outlier version. Recall that an input of KnapCenter consists of a metric space (V, d), m nonnegative weight functions $w_1, \ldots, w_m$ on V, and m budgets $\mathcal{B}_1, \ldots, \mathcal{B}_m$. The goal is to open a set of centers $\mathcal{S} \subseteq V$ with $w_i(\mathcal{S}) \leq \mathcal{B}_i$ for all $1 \leq i \leq m$, so as to minimize the maximum service cost of any vertex in V. In the outlier version of KnapCenter, we are given an additional parameter $p \leq |V|$, and the objective is to minimize $cost_p(\mathcal{S}) := \min_{V'\subseteq V:|V'|\geq p} \max_{v\in V'} \min_{i\in\mathcal{S}} d(v, i)$, i.e., the maximum service cost of any non-outlier node after excluding at most $|V| - p$ nodes as outliers.

3.1 Approximability of KnapCenter

When there is only one knapsack constraint (i.e., $m = 1$), the problem degenerates to the weighted k-center problem for which a 3-approximation algorithm exists [17]. However, as we show in Theorem 4, the situation changes dramatically even if there are only two knapsack constraints. The proof will be given in the full paper [6].

Theorem 4. *For any $f > 0$, if there is an f-approximation algorithm for* KnapCenter *with two knapsack constraints, then* $\mathrm{P} = \mathrm{NP}$.

It is then natural to ask whether constant factor approximation can be obtained if the constraints can be relaxed slightly. We show in Theorem 5 that this is achievable (even for the demand version). The proof of Theorem 5 can be found in the full paper [6].

Algorithm 3. Algorithm for Robust-KnapCenter

1 Guess the optimal objective value OPT.
2 For each $v \in V$, let $\mathsf{B}(v) \leftarrow \mathsf{B}(v, \mathsf{OPT})$ and $\mathsf{E}(v) \leftarrow \mathsf{B}(v, 3\mathsf{OPT})$.
3 $\mathcal{S} \leftarrow \emptyset; \mathcal{C} \leftarrow \emptyset$ (the points in $\mathcal{C}$ are *covered* and those in $V \setminus \mathcal{C}$ are *uncovered*).
4 **while** $w(\mathcal{S}) < \mathcal{B}$ *and* $V \setminus \mathcal{C} \neq \emptyset$ **do**
5 Choose $i \in V \setminus \mathcal{S}$ that maximizes $\frac{|\mathsf{B}(i) \setminus \mathcal{C}|}{w(i)}$.
6 $\mathcal{S} \leftarrow \mathcal{S} \cup \{i\}; \mathcal{C} \leftarrow \mathcal{C} \cup \mathsf{E}(i)$ (i.e., mark all uncovered points in $\mathsf{E}(i)$ as covered).
7 **end**
8 **return** $\mathcal{S}$

Theorem 5. *For any fixed $\epsilon > 0$, there is a 3-approximation algorithm for* Knap-Center *with a constant number of knapsack constraints, that is guaranteed to satisfy one constraint and violate each of the others by at most a factor of $1 + \epsilon$.*

3.2 Dealing with Outliers: Robust-KnapCenter

We now study Robust-KnapCenter, the outlier version of KnapCenter. Here we consider the case with one knapsack constraint (with weight function w and budget $\mathcal{B}$) and unit demand. Our main theorem is as follows.

Theorem 6. *There is a 3-approximation algorithm for* Robust-KnapCenter *that violates the knapsack constraint by at most a factor of $1 + \epsilon$ for any fixed $\epsilon > 0$.*

We present our algorithm for Robust-KnapCenter as Algorithm 3. We assume that $\mathcal{B} < w(V)$, since otherwise the problem is trivial. We also set $A/0 := \infty$ for $A > 0$ and $0/0 := 0$, which makes line 5 work even if $w(i) = 0$. Our algorithm can be regarded as a "weighted" version of that of Charikar et al. [2], but the analysis is much more involved. We next prove the following theorem, which can be used together with the partial enumeration technique to yield Theorem 6 (see [6] for details). Note that, if all clients have unit weight, Theorem 7 will guarantee a 3-approximate solution $\mathcal{S}$ with $w(\mathcal{S}) < \mathcal{B} + 1$, which implies $w(\mathcal{S}) \leq \mathcal{B}$. So it actually gives a 3-approximation without violating the constraint. Thus, our result generalizes that of Charikar et al. [2].

Theorem 7. *Given an input of the* Robust-KnapCenter *problem, Algorithm 3 returns a set $\mathcal{S}$ with $w(\mathcal{S}) < \mathcal{B} + \max_{v \in V} w(v)$ such that $cost_p(\mathcal{S}) \leq 3\mathsf{OPT}$.*

Proof. We call $\mathsf{B}(v)$ the disk of v and $\mathsf{E}(v)$ the expanded disk of v. Assume w.l.o.g. that the algorithm returns $\mathcal{S} = \{1, 2, \ldots, q\}$ where $q = |\mathcal{S}|$, and that the centers are chosen in the order $1, 2, \ldots, q$. It is easy to prove that $\mathsf{B}(1), \ldots, \mathsf{B}(q)$ are pairwise disjoint. For ease of notation, let $\mathsf{B}(V') := \bigcup_{v \in V'} \mathsf{B}(v)$ and $\mathsf{E}(V') := \bigcup_{v \in V'} \mathsf{E}(v)$ for $V' \subseteq V$. By the condition of the WHILE loop, $w(\{1, \ldots, q-1\}) < \mathcal{B}$, and thus $w(\mathcal{S}) < \mathcal{B} + w(q) \leq \mathcal{B} + \max_{v \in V} w(v)$. It remains to prove $cost_p(\mathcal{S}) \leq 3\mathsf{OPT}$. Note that this clearly holds if the expanded disks $\mathsf{E}(1), \ldots, \mathsf{E}(q)$ together cover at least p points. Thus, it suffices to show that $|\mathsf{E}(\mathcal{S})| \geq p$. If $w(S) < \mathcal{B}$, then all points in V are covered by $\mathsf{E}(\mathcal{S})$ due to the termination condition of the WHILE loop, and thus $|\mathsf{E}(\mathcal{S})| = |V| \geq p$. In the rest of the proof, we deal with the case $w(\mathcal{S}) \geq \mathcal{B}$.

For each $v \in V$, let $f(v)$ be the minimum $i \in \mathcal{S}$ such that $\mathsf{B}(v) \cap \mathsf{B}(i) \neq \emptyset$; let $f(v) = \infty$ if no such i exists (i.e., if disk $\mathsf{B}(v)$ is disjoint from all disks centered in $\mathcal{S}$). Suppose $\mathcal{O} = \{o_1, o_2, \ldots, o_m\}$ is an optimal solution, in which the centers are ordered such that $f(o_1) \leq \ldots \leq f(o_m)$. Clearly $|\mathsf{B}(\mathcal{O})| \geq p$. Hence we only need to show $|\mathsf{E}(\mathcal{S})| \geq |\mathsf{B}(\mathcal{O})|$. For any sets A, B we have $|A| = |A \setminus B| + |A \cap B|$. Therefore, $|\mathsf{E}(\mathcal{S})| - |\mathsf{B}(\mathcal{O})| = (|\mathsf{E}(\mathcal{S}) \setminus \mathsf{B}(\mathcal{O})| + |\mathsf{E}(\mathcal{S}) \cap \mathsf{B}(\mathcal{O})|) - (|\mathsf{B}(\mathcal{O}) \setminus \mathsf{E}(\mathcal{S})| + |\mathsf{E}(\mathcal{S}) \cap \mathsf{B}(\mathcal{O})|) = |\mathsf{E}(\mathcal{S}) \setminus \mathsf{B}(\mathcal{O})| - |\mathsf{B}(\mathcal{O}) \setminus \mathsf{E}(\mathcal{S})| \geq |\mathsf{B}(\mathcal{S}) \setminus \mathsf{B}(\mathcal{O})| - |\mathsf{B}(\mathcal{O}) \setminus \mathsf{E}(\mathcal{S})|$. As $\mathsf{B}(1), \ldots, \mathsf{B}(q)$ are pairwise disjoint, $|\mathsf{B}(\mathcal{S}) \setminus \mathsf{B}(\mathcal{O})| = |\cup_{i \in \mathcal{S}} (\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O}))| = \sum_{i \in \mathcal{S}} |\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O})|$, and $|\mathsf{B}(\mathcal{O}) \setminus \mathsf{E}(\mathcal{S})| = |\cup_{j=1}^{m} (\mathsf{B}(o_j) \setminus \mathsf{E}(\mathcal{S}))| \leq \sum_{j=1}^{m} |\mathsf{B}(o_j) \setminus \mathsf{E}(\mathcal{S})|$. Thus,

$$|\mathsf{E}(\mathcal{S})| - |\mathsf{B}(\mathcal{O})| \geq \sum_{i \in \mathcal{S}} |\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O})| - \sum_{j=1}^{m} |\mathsf{B}(o_j) \setminus \mathsf{E}(\mathcal{S})|. \tag{1}$$

Let t be the unique integer in $\{1, \ldots, m+1\}$ such that $f(o_j) \leq |\mathcal{S}|$ for all $1 \leq j \leq t-1$ and $f(o_j) = \infty$ for all $t \leq j \leq m$. Then, for all $1 \leq j \leq t-1$, we have $\mathsf{B}(o_j) \cap \mathsf{B}(f(o_j)) \neq \emptyset$, and thus $\mathsf{B}(o_j) \subseteq \mathsf{E}(f(o_j)) \subseteq \mathsf{E}(\mathcal{S})$, implying that $|\mathsf{B}(o_j) \setminus \mathsf{E}(\mathcal{S})| = 0$ for all $1 \leq j \leq t-1$. Combining with the inequality (1), we have $|\mathsf{E}(\mathcal{S})| - |\mathsf{B}(\mathcal{O})| \geq \sum_{i \in \mathcal{S}} |\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O})| - \sum_{j=t}^{m} |\mathsf{B}(o_j) \setminus \mathsf{E}(\mathcal{S})|$. Hence, it suffices to prove that

$$\sum_{i \in \mathcal{S}} |\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O})| - \sum_{j=t}^{m} |\mathsf{B}(o_j) \setminus \mathsf{E}(\mathcal{S})| \geq 0. \tag{2}$$

The inequality is trivial when $t = m+1$. Thus, we assume in what follows that $t \leq m$. For each $i \in \mathcal{S}$, define $\mathsf{R}(i) := \{j \mid 1 \leq j \leq m; f(o_j) = i\}$, and let $l(i) := \min\{j \mid j \in \mathsf{R}(i)\}$ and $q(i) := \max\{j \mid j \in \mathsf{R}(i)\}$ (let $l(i) = q(i) = \infty$ if $\mathsf{R}(i) = \emptyset$). By the definitions of $f(\cdot)$ and t, each $\mathsf{R}(i)$ is a set of consecutive integers (or empty), and $\{\mathsf{R}(i)\}_{i \in \mathcal{S}}$ forms a partition of $\{1, 2, \ldots, t-1\}$. Also, $q(i) = l(i+1)-1$ if $l(i+1) \neq \infty$.

Consider an arbitrary $i \in \mathcal{S}$. For each j such that $l(i+1) \leq j \leq t-1$, we know that $j \in \mathsf{R}(i')$ for some $i' > i$, i.e., $f(o_j) = i' > i$, and thus $\mathsf{B}(o_j) \cap \mathsf{B}(i) = \emptyset$. By the definition of t, we also have $\mathsf{B}(o_j) \cap \mathsf{B}(i) = \emptyset$ for all $t \leq j \leq m$. Therefore,

$$\mathsf{B}(o_j) \cap \mathsf{B}(i) = \emptyset \text{ for all } j \text{ s.t. } \min\{t, l(i+1)\} \leq j \leq m. \tag{3}$$

We next try to lower-bound $|\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O})|$ in order to establish (2). Equality (3) tells us that $\mathsf{B}(o_j) \cap \mathsf{B}(i) \neq \emptyset$ implies $j \in \mathsf{R}(1) \cup \ldots \cup \mathsf{R}(i)$. In consequence,

$$\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O}) = \mathsf{B}(i) \setminus \cup_{j=1}^{m} \mathsf{B}(o_j) = \mathsf{B}(i) \setminus \cup_{j \in \mathsf{R}(1) \cup \ldots \cup \mathsf{R}(i)} \mathsf{B}(o_j). \tag{4}$$

For each $j \in \mathsf{R}(i')$ with $1 \leq i' \leq i-1$, $\mathsf{B}(o_j) \cap \mathsf{B}(i') \neq \emptyset$, and thus $\mathsf{B}(o_j) \subseteq \mathsf{E}(i') \subseteq \mathsf{E}(\{1, 2, \ldots, i-1\})$. For convenience, define $\mathsf{E}_{<i} := \mathsf{E}(\{1, 2, \ldots, i-1\})$. Then, from (4) we get $\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O}) \supseteq \mathsf{B}(i) \setminus (\mathsf{E}_{<i} \cup \bigcup_{j \in \mathsf{R}(i)} \mathsf{B}(o_j))$, and hence

$$|\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O})| \geq |\mathsf{B}(i) \setminus (\mathsf{E}_{<i} \cup \bigcup_{j \in \mathsf{R}(i)} \mathsf{B}(o_j))| = |\mathsf{B}(i) \setminus (\mathsf{E}_{<i} \cup \bigcup_{j \in \mathsf{R}(i)} (\mathsf{B}(o_j) \setminus \mathsf{E}_{<i}))|$$

$$= |(\mathsf{B}(i) \setminus \mathsf{E}_{<i}) \setminus \bigcup_{j \in \mathsf{R}(i)} (\mathsf{B}(o_j) \setminus \mathsf{E}_{<i})| \geq |\mathsf{B}(i) \setminus \mathsf{E}_{<i}| - \sum_{j \in \mathsf{R}(i)} |\mathsf{B}(o_j) \setminus \mathsf{E}_{<i}|. \tag{5}$$

Now consider the particular execution of line 5 in which i is chosen and added to $\mathcal{S}$. Note that (3) holds for all $i \in \mathcal{S}$. Thus, for all $1 \leq i' \leq i-1$ and $\min\{t, l(i'+1)\} \leq j \leq m$, $\mathsf{B}(o_j)$ is disjoint from $\mathsf{B}(i')$, which in particular implies $o_j \notin \mathsf{B}(i')$. By considering all $i' \in \{1, \ldots, i-1\}$ and noting that $l(i) \geq l(i'+1)$, we have $o_j \notin \mathsf{B}(\{1, 2, \ldots, i-1\})$ for all $\min\{t, l(i)\} \leq j \leq m$. This further indicates that $\{1, 2, \ldots, i-1\} \cap \{o_j \mid \min\{t, l(i)\} \leq j \leq m\} = \emptyset$. Recall that $1, 2, \ldots, i-1$ are all the points added to $\mathcal{S}$ before i. Therefore, no point in $\{o_j \mid \min\{t, l(i)\} \leq j \leq m\}$ was chosen before i. By our way of choosing centers (see line 5), we have

$$\frac{|\mathsf{B}(i) \setminus \mathsf{E}_{<i}|}{w(i)} \geq \frac{|\mathsf{B}(o_j) \setminus \mathsf{E}_{<i}|}{w(o_j)} \text{ for all } j \text{ s.t. } \min\{t, l(i)\} \leq j \leq m. \tag{6}$$

Hence, for all $j \in \mathsf{R}(i)$, $|\mathsf{B}(o_j) \setminus \mathsf{E}_{<i}| \leq \frac{w(o_j)}{w(i)} |\mathsf{B}(i) \setminus \mathsf{E}_{<i}|$. By (5) we have

$$|\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O})| \geq \left(1 - \sum_{j \in \mathsf{R}(i)} \frac{w(o_j)}{w(i)}\right) |\mathsf{B}(i) \setminus \mathsf{E}_{<i}|. \tag{7}$$

By (6) we also have $|\mathsf{B}(i) \setminus \mathsf{E}_{<i}| \geq w(i) \cdot \max_{t \leq j \leq m} \frac{|\mathsf{B}(o_j) \setminus \mathsf{E}_{<i}|}{w(o_j)} \geq w(i) \cdot \frac{\sum_{j=t}^{m} |\mathsf{B}(o_j) \setminus \mathsf{E}_{<i}|}{\sum_{j=t}^{m} w(o_j)}$, where we use the inequality $\max_j \frac{A_j}{B_j} \geq \frac{\sum_j A_j}{\sum_j B_j}$ when $B_j \geq 0$ for all j. Plugging this inequality into (7) and noting that $\mathsf{E}_{<i} \subseteq \mathsf{E}(\mathcal{S})$, we obtain:

$$|\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O})| \geq \frac{w(i) - \sum_{j \in \mathsf{R}(i)} w(o_j)}{\sum_{j=t}^{m} w(o_j)} \cdot \sum_{j=t}^{m} |\mathsf{B}(o_j) \setminus \mathsf{E}(\mathcal{S})|. \tag{8}$$

Applying (8) for all $i \in \mathcal{S}$ and summing the resulting inequalities up, we get

$$\sum_{i \in \mathcal{S}} |\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O})| \geq \frac{\sum_{i \in \mathcal{S}} w(i) - \sum_{i \in \mathcal{S}} \sum_{j \in \mathsf{R}(i)} w(o_j)}{\sum_{j=t}^{m} w(o_j)} \cdot \sum_{j=t}^{m} |\mathsf{B}(o_j) \setminus \mathsf{E}(\mathcal{S})|. \tag{9}$$

As $\{\mathsf{R}(i)\}_{i \in \mathcal{S}}$ is a partition of $\{1, \ldots, t-1\}$, we have $\sum_{i \in \mathcal{S}} \sum_{j \in \mathsf{R}(i)} w(o_j) = \sum_{j=1}^{t-1} w(o_j)$. Recall that we are dealing with the case $w(\mathcal{S}) \geq \mathcal{B}$. Since $\mathcal{O}$ satisfies the weight constraint, we have $w(\mathcal{O}) = \sum_{j=1}^{m} w(o_j) \leq \mathcal{B} \leq w(\mathcal{S})$. Therefore, by (9) we have

$$\sum_{i \in \mathcal{S}} |\mathsf{B}(i) \setminus \mathsf{B}(\mathcal{O})| \geq \frac{\sum_{j=1}^{m} w(o_j) - \sum_{j=1}^{t-1} w(o_j)}{\sum_{j=t}^{m} w(o_j)} \sum_{j=t}^{m} |\mathsf{B}(o_j) \setminus \mathsf{E}(\mathcal{S})| = \sum_{j=t}^{m} |\mathsf{B}(o_j) \setminus \mathsf{E}(\mathcal{S})|,$$

which immediately gives (2). The completes the proof of Theorem 7. □

4 Concluding Remarks and Open Problems

We gave a 3-approximation algorithm for MatCenter and the best known inapproximability bound is $2 - \epsilon$. For Robust-MatCenter, we give a 7-approximation while the current best known lower bound is $3 - \epsilon$ due to the hardness of robust k-center with

forbidden centers [2]. It would be interesting to close these gaps. (Note that MatCenter includes as a special case the k-center problem with forbidden centers, i.e., some points are not allowed to be chosen as centers. It is known that another generalization of the latter, namely the k-supplier problem, is NP-hard to approximate within $3 - \epsilon$ [17].) For Robust-KnapCenter, it is interesting to explore whether constant factor approximation exists while not violating the knapsack constraint. It is also open whether there is a constant factor approximation for the demand version (even for the unit-weight case). Finally, extending our results for Robust-KnapCenter to the multi-constraint case seems intriguing and may require essentially different ideas.

Acknowledgements. The authors are grateful to the referees for their helpful suggestions on improving the quality and presentation of this paper.

References

1. Aggarwal, G., Feder, T., Kenthapadi, K., Khuller, S., Panigrahy, R., Thomas, D., Zhu, A.: Achieving anonymity via clustering. In: PODS, pp. 153–162 (2006)
2. Charikar, M., Khuller, S., Mount, D., Narasimhan, G.: Algorithms for facility location problems with outliers. In: SODA, pp. 642–651 (2001)
3. Charikar, M., Li, S.: A Dependent LP-Rounding Approach for the k-Median Problem. In: Czumaj, A., Mehlhorn, K., Pitts, A., Wattenhofer, R. (eds.) ICALP 2012, Part I. LNCS, vol. 7391, pp. 194–205. Springer, Heidelberg (2012)
4. Chechik, S., Peleg, D.: The Fault Tolerant Capacitated k-Center Problem. In: Even, G., Halldórsson, M.M. (eds.) SIROCCO 2012. LNCS, vol. 7355, pp. 13–24. Springer, Heidelberg (2012)
5. Chen, D.Z., Wang, H.: Efficient Algorithms for the Weighted k-Center Problem on a Real Line. In: Asano, T., Nakano, S.-i., Okamoto, Y., Watanabe, O. (eds.) ISAAC 2011. LNCS, vol. 7074, pp. 584–593. Springer, Heidelberg (2011)
6. Chen, D.Z., Li, J., Liang, H., Wang, H.: Matroid and knapsack center problems. Technical report (2012), `http://arxiv.org/abs/1301.0745`
7. Chen, K.: A constant factor approximation algorithm for k-median clustering with outliers. In: SODA, pp. 826–835 (2008)
8. Chuzhoy, J., Guha, S., Halperin, E., Khanna, S., Kortsarz, G., Krauthgamer, R., Naor, J.: Asymmetric k-center is $\log^* n$-hard to approximate. J. ACM 52(4), 538–551 (2005)
9. Cole, R.: Slowing down sorting networks to obtain faster sorting algorithms. J. ACM 34(1), 200–208 (1987)
10. Cygan, M., Hajiaghayi, M., Khuller, S.: LP rounding for k-centers with non-uniform hard capacities. In: FOCS, pp. 273–282 (2012)
11. Edmonds, J., Fulkerson, D.: Bottleneck extrema. J. Combin. Theory 8(3), 299–306 (1970)
12. Frederickson, G.N.: Parametric Search and Locating Supply Centers in Trees. In: Dehne, F., Sack, J.-R., Santoro, N. (eds.) WADS 1991. LNCS, vol. 519, pp. 299–319. Springer, Heidelberg (1991)
13. Gonzalez, T.: Clustering to minimize the maximum intercluster distance. Theor. Comput. Sci. 38, 293–306 (1985)
14. Grandoni, F., Zenklusen, R.: Approximation Schemes for Multi-Budgeted Independence Systems. In: de Berg, M., Meyer, U. (eds.) ESA 2010, Part I. LNCS, vol. 6346, pp. 536–548. Springer, Heidelberg (2010)
15. Hajiaghayi, M., Khandekar, R., Kortsarz, G.: Budgeted Red-Blue Median and Its Generalizations. In: de Berg, M., Meyer, U. (eds.) ESA 2010, Part I. LNCS, vol. 6346, pp. 314–325. Springer, Heidelberg (2010)

16. Hochbaum, D., Shmoys, D.: A best possible heuristic for the k-center problem. Math. Oper. Res., 180–184 (1985)
17. Hochbaum, D., Shmoys, D.: A unified approach to approximation algorithms for bottleneck problems. J. ACM 33(3), 533–550 (1986)
18. Khuller, S., Pless, R., Sussmann, Y.: Fault tolerant k-center problems. Theor. Comput. Sci. 242(1-2), 237–245 (2000)
19. Khuller, S., Saha, B., Sarpatwar, K.K.: New Approximation Results for Resource Replication Problems. In: Gupta, A., Jansen, K., Rolim, J., Servedio, R. (eds.) APPROX/RANDOM 2012. LNCS, vol. 7408, pp. 218–230. Springer, Heidelberg (2012)
20. Krishnaswamy, R., Kumar, A., Nagarajan, V., Sabharwal, Y., Saha, B.: The matroid median problem. In: SODA (2011)
21. Kumar, A.: Constant factor approximation algorithm for the knapsack median problem. In: SODA, pp. 824–832 (2012)
22. Lee, J., Mirrokni, V.S., Nagarajan, V., Sviridenko, M.: Non-monotone submodular maximization under matroid and knapsack constraints. In: STOC, pp. 323–332 (2009)
23. Li, J., Yi, K., Zhang, Q.: Clustering with Diversity. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010, Part I. LNCS, vol. 6198, pp. 188–200. Springer, Heidelberg (2010)
24. Matthew McCutchen, R., Khuller, S.: Streaming Algorithms for k-Center Clustering with Outliers and with Anonymity. In: Goel, A., Jansen, K., Rolim, J.D.P., Rubinfeld, R. (eds.) APPROX and RANDOM 2008. LNCS, vol. 5171, pp. 165–178. Springer, Heidelberg (2008)
25. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency. Springer, Berlin (2003)
26. Vondrák, J., Chekuri, C., Zenklusen, R.: Submodular function maximization via the multilinear relaxation and contention resolution schemes. In: STOC, pp. 783–792 (2011)
27. Zarrabi-Zadeh, H., Mukhopadhyay, A.: Streaming 1-center with outliers in high dimensions. In: CCCG, pp. 83–86 (2009)
28. Zenklusen, R.: Matroidal degree-bounded minimum spanning trees. In: SODA, pp. 1512–1521 (2012)

Cut-Generating Functions

Michele Conforti[1], Gérard Cornuéjols[2], Aris Daniilidis[3], Claude Lemaréchal[4], and Jérôme Malick[5]

[1] University of Padova
[2] Carnegie Mellon University
[3] Autonomous University of Barcelona
[4] INRIA, Grenoble
[5] CNRS, Grenoble

conforti@math.unipd.it, gc0v@andrew.cmu.edu, arisd@mat.uab.cat, {claude.lemarechal,jerome.malick}@inria.fr

Abstract. In optimization problems such as integer programs or their relaxations, one encounters feasible regions of the form $\{x \in \mathbb{R}^n_+ : Rx \in S\}$ where R is a general real matrix and $S \subset \mathbb{R}^q$ is a specific closed set with $0 \notin S$. For example, in a relaxation of integer programs introduced in [ALWW2007], S is of the form $\mathbb{Z}^q - b$ where $b \notin \mathbb{Z}^q$. One would like to generate valid inequalities that cut off the infeasible solution $x = 0$. Formulas for such inequalities can be obtained through cut-generating functions. This paper presents a formal theory of minimal cut-generating functions and maximal S-free sets which is valid independently of the particular S. This theory relies on tools of convex analysis.

Keywords: Integer programming, Convex analysis, Separation, Generalized gauges, S-free sets.

1 Introduction

1.1 The Separation Problem, Examples

This paper deals with sets of the form

$$X = X(R, S) := \left\{x \in \mathbb{R}^n_+ \,:\, Rx \in S\right\}, \tag{1}$$

where

$$\begin{array}{l} R = [r_1, \ldots, r_n] \text{ is a real } q \times n \text{ matrix,} \\ S \subset \mathbb{R}^q \text{ is a closed set with } 0 \notin S\ . \end{array} \tag{2}$$

In other words, our set X is the intersection of a closed convex cone (the nonnegative orthant) with a reverse image by a linear mapping. Since $0 \notin S$, it is not difficult to show that 0 does not lie in the closed convex hull of X.

We are interested in *separating* 0 from X: we want to generate cuts, i.e. inequalities valid for X, which we write as

$$c^\top x \geqslant 1\,, \quad \text{for all } x \in X\,. \tag{3}$$

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 123–132, 2013.

Geometrically, we want to generate *half-spaces* $H^+ = \left\{x \in \mathbb{R}^n \,:\, c^\top x \geqslant 1\right\}$ (note: $0 \notin H^+$) satisfying $H^+ \supset X$. This paper presents an overview of a formal theory of the functions that generate the coefficients c_j of such cuts.

Let us first give some motivation for our model (1), (2), arising in mixed integer programming. Starting from a polyhedron

$$P = \left\{(x, y) \in \mathbb{R}^n_+ \times \mathbb{R}^m \,:\, Ax + y = b\right\}$$

(nonnegativity of the y-variables can also be imposed), assume that $b \notin \mathbb{Z}^m$. Several situations have been considered in the literature.

Example 1 (An integer linear program). Suppose first that all variables must be integers: the set of interest is $P \cap \{\mathbb{Z}^n \times \mathbb{Z}^m\}$, i.e. the set of points $(x, y = b - Ax)$ such that $x \in \mathbb{Z}^n_+$ and $b - Ax \in \mathbb{Z}^m$. Our problem has the form (1), (2) if we set

$$q = n + m\,, \quad R = \begin{bmatrix} I \\ -A \end{bmatrix}, \quad S = \mathbb{Z}^n \times \mathbb{Z}^m - \begin{bmatrix} 0 \\ b \end{bmatrix}. \tag{4}$$

Since $b \notin \mathbb{Z}^m$, the above S is a closed set not containing the origin; (4) is the model considered by Gomory [G1969]. □

Example 2 (A mixed integer linear program). Consider now $P \cap \{\mathbb{R}^n \times \mathbb{Z}^m\}$: the set of interest is the set of points $(x, y = b - Ax)$ such that $x \in \mathbb{R}^n_+$ and $b - Ax \in \mathbb{Z}^m$. Then (4) is replaced by

$$q = m\,, \quad R = -A\,, \quad S = \mathbb{Z}^m - b\,,$$

which is the model considered by Andersen, Louveaux, Weismantel and Wolsey [ALWW2007]. □

We will retain from the above two examples the asymmetry between S (a very particular and highly structured set) and R (an arbitrary matrix). Keeping this in mind, we will consider that (q, S) is given and fixed, while (n, R) is instance-dependent data: our cutting problem can be viewed as *parametrized* by (n, R). A number of papers have appeared in recent years, dealing with the above problem with various special forms for S, see [ALWW2007], [DW2010], [BCCZ2010] and references therein.

1.2 Cut-Generating Functions and S-Free Sets

Let (q, S) be given and fixed. To generate cuts in the present situation, it would be convenient to have a mapping, taking instances of (1), (2) as input, and producing cuts as output. What we need for this is a function

$$\mathbb{R}^q \ni r \mapsto \rho(r) \in \mathbb{R}\,.$$

We will apply the function ρ to the columns r_j of R (an arbitrary matrix, with an arbitrary number of columns) to produce the coefficients $c_j := \rho(r_j)$ of a cut (3). In summary, we require that our ρ satisfies, for any instance X of (1),

$$x \in X \quad \Longrightarrow \quad \sum_{j=1}^{n} \rho(r_j) x_j \geqslant 1\,. \tag{5}$$

Such a ρ can be called a *cut-generating function* (CGF). So far, a CGF is a rather abstract object; but the (vast!) class of functions from $\mathbb{R}^q$ to $\mathbb{R}$ can be drastically reduced from the following observations.

(i) First consider in (3) a vector c' with $c'_j \leqslant c_j$ for $j = 1, \ldots, n$; then ${c'}^\top x \leqslant c^\top x$ for any $x \geqslant 0$. If c' is a cut, it is tighter than c in the sense that it cuts a bigger portion of $\mathbb{R}^n_+$. We can impose some "minimal" character to a CGF, in order to reach some "tightness" of the resulting cuts.

(ii) Next observe that changing R to tR $(t > 0)$ divides X by t; the set of cuts is just multiplied by t. Since we seek a minimal ρ, we can impose without loss of generality $\rho(tr) = t\rho(r)$, for any $r \in \mathbb{R}^q$ and $t > 0$: only *positively homogeneous* CGF's are of interest.

(iii) It can be proved that the closed convex hull of a CGF ρ is again a CGF. Moreover, if ρ is positively homogeneous, then the closed convex hull of ρ is positively homogeneous as well.

A function is *sublinear* if it is convex and positively homogeneous. The above observations show that the class of sublinear functions suffices to generate all relevant cuts; a fairly narrow class indeed, which is fundamental in convex analysis. Sublinear functions are in correspondence with closed convex sets and in our context, such a correspondence is based on the mapping $\rho \mapsto V$ defined by

$$V = V(\rho) := \left\{ r \in \mathbb{R}^q \,:\, \rho(r) \leqslant 1 \right\}. \tag{6}$$

Sublinear functions $\rho : \mathbb{R}^q \mapsto \mathbb{R}$ are convex, continuous and satisfy $\rho(0) = 0$, which implies that $V(\rho)$ in (6) is a closed convex neighborhood of 0 in $\mathbb{R}^q$. The set V turns out to be a cornerstone: via Theorem 1 below, (6) establishes a correspondence between the (sublinear) CGF's and the so-called *S-free sets*.

Definition 1 (S-free set). *Given a closed set $S \subset \mathbb{R}^q$ not containing the origin, a closed convex neighborhood V of $0 \in \mathbb{R}^q$ is called S-free if its interior contains no point in S:* $\operatorname{int}(V) \cap S = \emptyset$. □

Theorem 1. *Let ρ be a sublinear function from $\mathbb{R}^q$ to $\mathbb{R}$ and $V(\rho)$ the closed convex neighborhood of $0 \in \mathbb{R}^q$ defined in* (6)*. Then ρ is a* CGF *for* (1), (2) *if and only if $V(\rho)$ is S-free.*

As a result, the cut generation problem for X can alternatively be studied from a geometric point of view, involving sets V instead of functions ρ. This situation, common in convex analysis, is often very fruitful.

Definition 2 (CGF as representation). *Let $V \subset \mathbb{R}^q$ be a closed convex neighborhood of the origin. A* representation *of V is a (finite-valued) sublinear function ρ satisfying* (6)*. We will say that ρ* represents *V. A (sublinear) cut-generating function for* (1), (2) *is a representation of an S-free set.* □

A sublinear ρ represents a unique $V = V(\rho)$, well-defined by (6). One easily checks

$$\rho \leqslant \rho' \quad \Longrightarrow \quad V(\rho) \supset V(\rho'). \tag{7}$$

Hence, minimality of ρ corresponds to maximality of V. By contrast, the mapping $\rho \mapsto V(\rho)$ in (6) is many-to-one and therefore has no inverse. There is a difficulty here: a given neighborhood V may have several representations, and we are interested in the small ones.

1.3 Goals and Outline of the Paper

The aim of this paper is to present the main points of a formal theory of minimal cut-generating functions and maximal S-free sets which is valid independently of the particular S. This theory of cut-generating functions gathers, generalizes and synthesizes some existing results (see [BCZ2011], [DW2010], [BCCZ2010] and references therein). The complete theory is presented in an extended version of this paper [CCDLM2013]; in particular, the proofs of the results are omitted here, so the reader is referred to [CCDLM2013] to see precisely how things combine.

The paper is organized as follows. We study the mapping (6) in Section 2. We show that the pre-images of a given V (the representations of V) have a unique maximal element γ_V and a unique minimal element μ_V; in view of (i) above, the latter is *the* relevant inverse of $\rho \mapsto V(\rho)$. Then we study in Section 3 the correspondence $V \leftrightarrow \mu_V$. We show that different concepts of minimality come into play for ρ in (i). Geometrically they correspond to different concepts of maximality for V. We also show that they coincide in a number of cases.

2 Largest and Smallest Representations

In this section, we study the representation operation introduced in Definition 2 and its geometric counterpart. We first recall some basic definitions of convex analysis; The monograph [HL2001] (especially its Chapter C) is suggested for an elementary introduction, while textbooks [HL1993, R1970] are more complete.

2.1 Basic Definitions of Convex Analysis

The *support function* of a set $G \subset \mathbb{R}^q$ is

$$\sigma_G(r) := \sup_{d \in G} d^\top r \,. \tag{8}$$

It is seen to be sublinear, to grow when G grows, but to remain unchanged if G is replaced by its closed convex hull: $\sigma_G = \sigma_{\overline{\text{conv}}(G)}$. Conversely, any sublinear function ρ is the support function of a closed convex set, unambiguously defined by

$$G = G_\rho := \left\{ d \in \mathbb{R}^n \,:\, d^\top r \leqslant \rho(r) \text{ for all } r \in \mathbb{R}^q \right\};$$

we say that ρ supports G. Note that a sublinear function ρ is finite valued if and only if ρ is the support function of a bounded closed convex set.

Another relevant object for our purpose is the *gauge*

$$\mathbb{R}^q \ni r \mapsto \gamma_V(r) := \inf\{\lambda > 0 \,:\, r \in \lambda V\} \tag{9}$$

of our neighborhood V. In fact, results in convex analysis [HL2001, Theorem C.1.2.5 and Proposition C.3.2.4] show that γ_V
– also appears as a representation of V
– is the support function of the polar set of V defined by

$$V^\circ := \left\{d \,:\, d^\top r \leqslant 1 \text{ for all } r \in V\right\} = \left\{d \,:\, \sigma_V(d) \leqslant 1\right\}. \tag{10}$$

2.2 Prepolars and Representations

From now on in this section, we are given a subset V of $\mathbb{R}^q$, which is a closed convex neighborhood of the origin. If G is such that $G^\circ = V$, we can say that G is a *prepolar* of V, i.e. that σ_G represents V in the sense of Definition 2. As already mentioned, V may have several representations, and there may be several G's such that $G^\circ = V$, that is, several G's may be prepolars of V. Because $(V^\circ)^\circ = V$, the standard polar V° is itself a prepolar – which is somewhat confusing – and turns out to be the largest one; or equivalently γ_V turns out to be the largest representation of V, as shown by Theorem 2 below. This theorem states furthermore that V has also a smallest prepolar, or equivalently a smallest representation; keeping (i) of Section 1 in mind, this is exactly what we want. This result is actually [BCZ2011, Theorem 1]; we give a different treatment here.

The following geometric objects turn out to be relevant:

$$\begin{cases} \widetilde{V}^\circ := \left\{d \in V^\circ \,:\, d^\top r = \sigma_V(d) = 1 \text{ for some } r \in V\right\}, \\ \widehat{V}^\circ := \left\{d \in V^\circ \,:\, \sigma_V(d) = 1\right\}. \end{cases} \tag{11}$$

For later use, we illustrate this construction with a simple example.

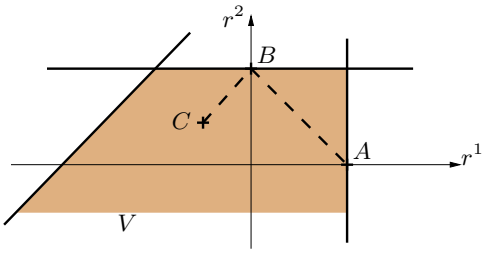


Fig. 1. Constructing $\widetilde{V}^\circ$ or $\widehat{V}^\circ$

Example 3. With $\begin{bmatrix} r^1 \\ r^2 \end{bmatrix} \in \mathbb{R}^2$, take for V the polyhedron given by the following three inequalities (see Figure 1):

$$r^1 \leqslant 1\,, \quad r^2 \leqslant 1\,, \quad r^2 \leqslant 2 + r^1\,.$$

Recalling that extreme points of V° correspond to facets of V, we see that V° has the three extreme points A, B, C defined by the equation $d^\top r = 1$, for r respectively on the three lines making up the boundary of V. We obtain $A = (1, 0)$, $B = (0, 1)$, $C = \frac{1}{2}(-1, 1)$.

In this example, $\widetilde{V}^\circ$ and $\widehat{V}^\circ$ are the same set, namely the union of the two segments $[A, B]$ and $[B, C]$. To obtain V°, convexify them with the fourth point 0; if V had a fourth constraint, say $r^2 \geqslant -1$, then this fourth point would be moved down to $D = (0, -1)$ – and would be part of the sets $\widetilde{V}^\circ$ and $\widehat{V}^\circ$. □

Because $0 \in \operatorname{int} V$, the definition (8) of a support function shows that σ_V is positive whenever it is finite: for some $\varepsilon = \varepsilon(V) > 0$,

$$\varepsilon\|d\| \leqslant \sigma_V(d) \leqslant +\infty \quad \text{for all } d \in \mathbb{R}^q\,. \tag{12}$$

The two sets in (11) are therefore bounded. Besides, the next proposition shows that they differ very little.

Proposition 1. *We have* $\widetilde{V}^\circ \subset \widehat{V}^\circ \subset \operatorname{cl}\big(\widetilde{V}^\circ\big)$. *It follows that* $\widehat{V}^\circ$ *and* $\widetilde{V}^\circ$ *have the same closed convex hull.*

The closed convex hull revealed by this proposition deserves a notation, as well as its support function: we set

$$V^\bullet := \overline{\operatorname{conv}}\big(\widetilde{V}^\circ\big) = \overline{\operatorname{conv}}\big(\widehat{V}^\circ\big) \quad \text{and} \quad \mu_V := \sigma_{V^\bullet} = \sigma_{\widetilde{V}^\circ} = \sigma_{\widehat{V}^\circ} \tag{13}$$

(in Figure 1, $V^\bullet$ is the triangle $\operatorname{conv}(A, B, C)$). In fact, the next result shows that μ_V is *the* smallest representation we are looking for. From now on, we assume $V \neq \mathbb{R}^q$ (otherwise $V^\bullet = \emptyset$, $\mu_V \equiv -\infty$, a degenerate situation which is trivial).

Proposition 2 (Smallest representation). *Any* ρ *representing* V *satisfies* $\rho \geqslant \mu_V$. *Geometrically,* $V^\bullet$ *is the smallest closed convex set whose support function represents* V.

Thus, V does have a smallest representation, whose supported set is $V^\bullet$. On the other hand, it is interesting to link it with V°. The intuition suggested by Figure 1 is confirmed by the following result.

Proposition 3. *Appending* 0 *to* $V^\bullet$ *gives the standard polar:*

$$\gamma_V = \max\big\{\mu_V, 0\big\} \quad \textit{i.e.} \quad V^\circ = \overline{\operatorname{conv}}\big(V^\bullet \cup \{0\}\big) = [0, 1]V^\bullet\,.$$

We actually have an equivalence.

Theorem 2 (Representations). *A sublinear function* ρ *represents* V *if and only if it satisfies*

$$\mu_V \leqslant \rho \leqslant \gamma_V\,. \tag{14}$$

Geometrically, the support function of a set G *represents* V *if and only if* G *is sandwiched between the two extreme prepolars of* V*:*

$$G^\circ = V \quad \Longleftrightarrow \quad V^\bullet \subset \overline{\operatorname{conv}}(G) \subset V^\circ\,.$$

3 Minimal CGF's and Maximal S-Free Sets

3.1 Minimal CGF's

In our quest for small CGF's, the following definition is natural.

Definition 3 (Minimality). *A* CGF ρ *is called minimal if any* CGF $\rho' \leqslant \rho$ *is* ρ *itself.* □

A minimal CGF is certainly a smallest representation:

$$\rho \text{ is a minimal CGF} \quad \Longrightarrow \quad \rho = \mu_{V(\rho)} = \sigma_{V(\rho)\bullet} \tag{15}$$

(indeed, Theorem 2 states that $\mu_{V(\rho)}$ represents the same set $V(\rho)$ as ρ – and is therefore a CGF if so is ρ).

If ρ is a minimal CGF, $V(\rho)$ must of course be a special S-free set. Take for example $S = \{1\} \subset \mathbb{R}$ and the S-free set $V = [-1, +1]$; $\rho(r) := |r|$ is the smallest (because unique) representation of V but ρ is not minimal: $\rho'(r) := \max\{0, r\}$ is also a CGF, representing $V' =]-\infty, +1]$. From (7), a smaller ρ describes a larger V; so Definition 3 has its geometrical counterpart:

Definition 4 (Maximality). *An S-free set V of Definition* 1 *is called* maximal *if any S-free set $V' \supset V$ is V itself.* □

Actually, this "duality" is deceiving, as the two definitions do not match: the set represented by a minimal CGF need not be maximal. Here is a trivial example.

Example 4. When ρ is linear, the property introduced in Definition 3 holds vacuously: no sublinear function can properly lie below a linear function. Thus, any linear CGF ρ is minimal; yet, a linear ρ represents a neighborhood $V(\rho)$ (a half-space) which is S-free but has not reason to be maximal. See Figure 2: with $n = 1$, the set $V =]-\infty, 1]$ (represented by $\rho(x) = x$) is $\{2\}$-free but is obviously not maximal. □

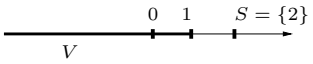


Fig. 2. A linear CGF is always maximal

Note that, if the half-space represented by a linear function is S-free, it actually separates S from 0. A simple assumption such as $0 \in \operatorname{conv} S$ will therefore rule out the above counterexample; but Example 5 below will reveal a more serious deficiency. So a subtlety is necessary, indeed the smallest representation of a maximal V enjoys a stronger property than minimality.

3.2 Strongly Minimal CGF's

Let ρ be a CGF, which represents via (6) the set $V = V(\rho)$. The gauge $\gamma_{V(\rho)}$ is then a function of ρ and here comes the correct substitute to Definition 3.

Definition 5 (Strongly minimal CGF). *A CGF ρ is called strongly minimal if any CGF $\rho' \leqslant \gamma_{V(\rho)}$ satisfies $\rho' \geqslant \rho$.*

Needless to say, the class of strong minimality CGF's is a subclass of the class of minimal CGF's. Example 5 below will complement Example 4, showing that the restriction is a real one. At any rate, strong minimality turns out to be *the* appropriate definition in general:

Theorem 3 (Strongly minimal $\Leftrightarrow$ maximal). *An S-free set V is maximal if and only if its smallest representation μ_V of* (13) *is a strongly minimal* CGF.

In fact, the concept of minimality involves two properties from a sublinear function:
– it must be the *smallest* representation of some V (recall (15)),
– the neighborhood V must enjoy some maximality property.
In view of the first property, a CGF can be imposed to be not only sublinear but also to support a set that is a *smallest* prepolar. Then Definition 3 has a geometric counterpart: minimality of $\rho = \mu_V = \sigma_{V^\bullet}$ means

$$\begin{array}{ccccc} G' \subset V^\bullet & \text{and } (G')^\circ \text{ is } S\text{-free} & \Longrightarrow & G' = V^\bullet, \text{ i.e. } (G')^\circ = V\,. \\ [\rho' = \sigma_{G'} \leqslant \mu_V] & [\rho' \text{ is a CGF}] & & [\rho' = \rho] \end{array}$$

Likewise for Definition 5: strong minimality of $\rho = \gamma_V = \sigma_{V^\circ}$ means

$$\begin{array}{ccccc} G' \subset V^\circ & \text{and } (G')^\circ \text{ is } S\text{-free} & \Longrightarrow & G' \supset V^\bullet, \text{ i.e. } (G')^\circ \subset V\,. \\ [\rho' = \sigma_{G'} \leqslant \gamma_V] & [\rho' \text{ is a CGF}] & & [\rho' \geqslant \rho] \end{array}$$

These observations allow some more insight into the $(\cdot)^\bullet$ operation:

Proposition 4. *Let $\rho = \mu_V = \sigma_{V^\bullet}$ be a minimal* CGF. *If an S-free neighborhood W satisfies $W^\bullet \subset V^\bullet$, then $W = V$.*

Thus, the trouble necessitating strong minimality lies in (7): even though the reverse implication holds when $\rho = \gamma_V$, it does not hold for $\rho = \mu_V$: the mapping $V \mapsto V^\bullet$ is not monotonic; and of course, this phenomenon is linked to the presence of the recession cone V_∞. The following example helps for a better understanding.

Example 5. In Example 3, take for S the union of the three lines given respectively by the three equations

$$r^1 = 1\,, \quad r^2 = 1\,, \quad r^2 = 2 + r^1\,,$$

so that V is clearly maximal S-free.

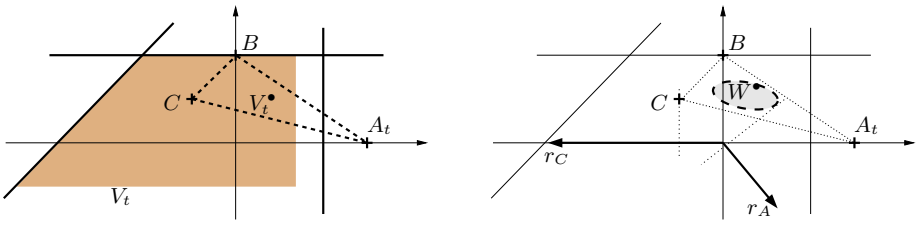


Fig. 3. The mapping $V \mapsto V^\bullet$ is not monotonic

Now shrink V to V_t (left part of Figure 3) by moving its right vertical boundary to $r^1 \leqslant 1 - t$. Then A is moved to $A_t = \left(\frac{1}{1-t}, 0\right)$; there is no inclusion between the new $V_t^\bullet = \mathrm{conv}\,(A_t, B, C)$ and the original $V^\bullet = \mathrm{conv}\,(A, B, C)$; this is the key to our example.

Let us show that μ_{V_t} is minimal, even though V_t is not maximal. Take for this a CGF $\rho \leqslant \mu_{V_t}$, which represents an S-free set W; by (7), $W \supset V_t$. With the notation (13), we therefore have

$$\sigma_{W^\bullet} = \mu_W \leqslant \rho \leqslant \mu_{V_t} = \sigma_{V_t^\bullet}\,, \quad \text{i.e.,} \quad W^\bullet \subset V_t^\bullet$$

and we proceed to show that equality does hold, i.e. the three extreme points of $V_t^\bullet$ do lie in $W^\bullet$.

– If $A_t \notin W^\bullet$, the right part of Figure 3 shows that $W^\bullet$ is included in the open upper half-space. Knowing that

$$W = \left(W^\bullet\right)^\circ = \left\{r \,:\, d^\top r \leqslant 1 \text{ for all } d \in W^\bullet\right\}$$

(see the end of Section 2), this implies that the recession cone W_∞ has a vector of the form $r_A = (\varepsilon, -1)$ $(\varepsilon > 0)$; W cannot be S-free.

– If $C \notin W^\bullet$, there is $r_C \in \mathbb{R}^2$ such that $C^\top r_C > \sigma_{W^\bullet}(r_C) = \mu_W(r_C)$ (we denote also by C the 2-vector representing C). For example $r_C = (-2, 0) \in \mathrm{bd}\,(V)$ (see Figure 3), so that

$$C^\top r_C = 1 > \sigma_{W^\bullet}(-2, 0) = \mu_W(-2, 0)\,.$$

By continuity, $\mu_W(-2 - \varepsilon, 0) \leqslant 1$ for $\varepsilon > 0$ small enough. Because μ_W represents W, this implies that $(-2 - \varepsilon, 0) \in W$; W (which contains V_t) is not S-free.

– By the same token, we prove that $B \in W^\bullet$ (the separator $r_B = (0, 1) \in \mathrm{bd}\,(V)$ does the job).

We have therefore proved that $W^\bullet = V_t^\bullet$, i.e $\mu_W = \mu_{V_t}$, i.e. μ_{V_t} is minimal. □

Examples 4 and 5 show that minimality does not imply strong minimality in general. On the other hand, the following theorem provides two favorable cases when this implication holds.

Theorem 4. *Suppose* $0 \in \hat{S} := \overline{\mathrm{conv}} S$ *and that* μ_V *is minimal. Then* μ_V *is strongly minimal under any of the following conditions:*

(i) $V_\infty \cap \hat{S}_\infty = \{0\}$ *(in particular S bounded),*
(ii) $V_\infty \cap \hat{S}_\infty = L \cap \hat{S}_\infty$ *where L stands for the lineality space of V, and $\hat{S} = G + \hat{S}_\infty$ where G in any nonempty bounded set.*

Theorem 4 generalizes several earlier results. The special case where S is a finite set of points in $\mathbb{Z}^q - b$ was first considered by Johnson [J1981] and more recently by Dey and Wolsey [DW2010]. Theorem 4(ii) was proven by [DW2010] and [BCCZ2010] in the special case where $S = P \cap (\mathbb{Z}^q - b)$ for some rational polyhedron P.

3.3 Asymptotically Maximal Sets

Finally a natural question arises: how far from being maximal are the S-free sets represented by minimal CGF's? For this, we introduce one more concept, which does not seem to have arisen in the literature on cut-generating functions.

Definition 6. *An S-free set V of Definition* 1 *is called* asymptotically maximal *if any S-free set $V' \supset V$ satisfies $V'_\infty = V_\infty$.*

Then we have a partial answer to the question about S-free sets represented by minimal CGF's.

Theorem 5 (Minimal ⇒ asymptotically maximal). *The S-free neighborhood represented by a minimal* CGF *is asymptotically maximal.*

References

[ALWW2007] Andersen, K., Louveaux, Q., Weismantel, R., Wolsey, L.: Cutting Planes from Two Rows of a Simplex Tableau. In: Proceedings of IPCO XII, Ithaca, New York, pp. 1–15 (2007)
[BCZ2011] Basu, A., Cornuéjols, G., Zambelli, G.: Convex Sets and Minimal Sublinear Functions. Journal of Convex Analysis 18, 427–432 (2011)
[BCCZ2010] Basu, A., Conforti, M., Cornuéjols, G., Zambelli, G.: Minimal Inequalities for an Infinite Relaxation of Integer Programs. SIAM Journal on Discrete Mathematics 24, 158–168 (2010)
[CCDLM2013] Conforti, M., Cornuéjols, G., Daniilidis, A., Lemaréchal, C., Malick, J.: Cut-Generating Functions and S-free Sets (submitted for publication, 2013)
[DW2010] Dey, S.S., Wolsey, L.A.: Constrained Infinite Group Relaxations of MIPs. SIAM Journal on Optimization 20, 2890–2912 (2010)
[G1969] Gomory, R.G.: Some Polyhedra Related to Combinatorial Problems. Linear Algebra and Applications 2, 451–558 (1969)
[J1981] Johnson, E.L.: Characterization of Facets for Multiple Right-Hand Side Choice Linear Programs. Mathematical Programming Study 14, 112–142 (1981)
[HL1993] Hiriart-Urruty, J.-B., Lemaréchal, C.: Convex Analysis and Minimization Algorithms. Springer (1993)
[HL2001] Hiriart-Urruty, J.-B., Lemaréchal, C.: Fundamentals of Convex Analysis. Springer (2001)
[R1970] Rockafellar, R.T.: Convex Analysis. Princeton University Press (1970)

Reverse Chvátal-Gomory Rank*

Michele Conforti[1], Alberto Del Pia[2], Marco Di Summa[1], Yuri Faenza[3], and Roland Grappe[4]

[1] Dipartimento di Matematica, Università degli Studi di Padova, Italy
[2] IFOR, Department of Mathematics, ETH Zürich, Switzerland
[3] DISOPT, Institut de mathématiques d'analyse et applications, EPFL, Switzerland
[4] Laboratoire d'Informatique de Paris-Nord, UMR CNRS 7030, Université Paris 13, France

Abstract. We introduce the *reverse Chvátal-Gomory rank* $r^*(P)$ of an integral polyhedron P, defined as the supremum of the Chvátal-Gomory ranks of all rational polyhedra whose integer hull is P. A well-known example in dimension two shows that there exist integral polytopes P with $r^*(P) = +\infty$. We provide a geometric characterization of polyhedra with this property in general dimension, and investigate upper bounds on $r^*(P)$ when this value is finite. We also sketch possible extensions, in particular to the reverse split rank.

1 Introduction

A polyhedron is *integral* if it is the convex hull of its integer points. Given an integral polyhedron $P \subseteq \mathbb{R}^n$, a *relaxation* of P is a rational polyhedron $Q \subseteq \mathbb{R}^n$ such that $Q \cap \mathbb{Z}^n = P \cap \mathbb{Z}^n$. Note that if Q is a relaxation of P, then $P = \text{conv}(Q \cap \mathbb{Z}^n)$, i.e., P is the *integer hull* of Q, where we denote the convex hull of a set S by $\text{conv}(S)$ (for the definition of convex hull and other standard preliminary notions not given in here, we refer the reader to textbooks, e.g. [9] and [17]). An inequality $cx \leq \lfloor \delta \rfloor$ is a *Chvátal–Gomory inequality* (*CG inequality* for short) for a polyhedron $Q \subseteq \mathbb{R}^n$ if c is an integer vector and $cx \leq \delta$ is valid for Q. Note that $cx \leq \lfloor \delta \rfloor$ is a valid inequality for $Q \cap \mathbb{Z}^n$. The *CG closure* Q' of Q is the set of points that satisfy all the CG inequalities for Q. If Q is a rational polyhedron, then Q' is again a rational polyhedron [16]. For $p \in \mathbb{N}$, the *p–th CG closure* $Q^{(p)}$ of Q is defined iteratively as $Q^{(p)} = (Q^{(p-1)})'$, with $Q^{(0)} = Q$. If Q is a rational polyhedron, then there exists some $p \in \mathbb{N}$ such that $Q^{(p)} = \text{conv}(Q \cap \mathbb{Z}^n)$ [16]. The minimum p for which this occurs is called the *CG rank* of Q and is denoted by $r(Q)$.

Cutting plane procedures in general and CG inequalities in particular are of crucial importance to the integer programming community, because of their convergence properties (see e.g. [8,17]) and relevance in practical applications (see e.g. [10]). Hence, a theoretical understanding of their features has been the

* This work was supported by the *Progetto di Eccellenza 2008–2009* of the *Fondazione Cassa di Risparmio di Padova e Rovigo*.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 133–144, 2013.

goal of several papers from the literature. Many of them aimed at giving upper or lower bounds on the CG rank for some families of polyhedra. For instance, Bockmayr et al. [4] proved that the CG rank of a polytope $Q \subseteq [0,1]^n$ is at most $O(n^3 \log n)$. The bound was later improved to $O(n^2 \log n)$ by Eisenbrand and Schulz [7]. Recently, Rothvoß and Sanità [15], improving over earlier results of Eisenbrand and Schulz [7] and Pokutta and Stauffer [14], showed that this bound is almost tight, as there are polytopes in the unit cube whose CG rank is at least $\Omega(n^2)$. An upper bound on the CG rank for polytopes contained in the cube $[0,\ell]^n$ for an arbitrary given ℓ was provided by Li [12]. Recently, Averkov et al. [1] studied the rate of convergence – in terms of number of CG closures – of the affine hull of a rational polyhedron to the affine hull of its integer hull.

Our Contribution. In this paper we investigate a question that is, in a sense, reverse to that of giving bounds on the CG rank for a fixed polyhedron Q. In fact, in most applications, even if we do not have a complete linear description of the integer hull P, we know many of its properties: for instance, the integer points of most polyhedra stemming from combinatorial optimization problems have 0-1 coordinates. Hence, for a fixed *integral* polyhedron P, we may want to know how "bad" a relaxation of P can be in terms of its CG rank. More formally, we want to answer the following question: given an integral polyhedron P, what is the supremum of $r(Q)$ over all rational polyhedra Q whose integer hull is P? We call this number the *reverse CG rank* of P and denote it by $r^*(P)$:

$$r^*(P) = \sup\{r(Q) : Q \text{ is a relaxation of } P\}.$$

Note that $r^*(P) < +\infty$ if and only if there exists $p \in \mathbb{N}$ such that $r(Q) \leq p$ for every relaxation Q of P. Our main result gives a geometric characterization of those integral polyhedra P for which $r^*(P) = +\infty$. Denoting by $\mathrm{rec}(P)$ the recession cone of P, by $\langle v \rangle$ the line generated by a non-zero vector v, and by $+$ the Minkowski sum of two sets, we prove the following:

Theorem 1. *Let $P \subseteq \mathbb{R}^n$ be an integral polyhedron. Then $r^*(P) = +\infty$ if and only if P is non-empty and there exists $v \in \mathbb{Z}^n \setminus \mathrm{rec}(P)$ such that $P + \langle v \rangle$ does not contain any integer point in its relative interior.*

Let us illustrate Theorem 1 with an example in dimension two. Let $P = \mathrm{conv}\{(0,0),(0,1)\}$, and consider the family $\{Q_t\}_{t\in\mathbb{N}}$ of relaxations of P, where we define $Q_t = \mathrm{conv}\{(0,0),(0,1),(t,1/2)\}$. It is folklore that the CG rank of Q_t increases linearly with t (see Figure 1). This implies that $r^*(P) = +\infty$. Note that if one chooses $v = (1,0)$, then $P + \langle v \rangle$ does not contain any integer point in its (relative) interior. A simple application of Theorem 1 shows that the previous example can be generalized to every dimension: any 0-1 polytope $P \subseteq \mathbb{R}^n$, $n \geq 2$, whose dimension is at least 1, has infinite reverse CG rank, since there always exists a vector v parallel to one of the axis such that $P + \langle v \rangle$ does not contain any integer point in its relative interior. On the other hand, every integral polyhedron containing an integer point in its relative interior has finite reverse CG rank,

as no vector v satisfying the condition of Theorem 1 exists in this case. However, there are also integral polyhedra with finite reverse CG rank that do not contain integer points in their interior, such as $\text{conv}\{(0,0),(2,0),(0,2)\} \subseteq \mathbb{R}^2$.

We then show that for a wide class of polyhedra with finite reverse CG rank, r^* can be upper bounded by functions depending only on parameters such as the dimension of the space and the number of the integer points in the relative interior of P. Moreover, we give examples showing that any upper bound on r^* for those polyhedra *must* depend on those parameters. We also investigate the extension of the concept of reverse rank to split cuts in dimension 2 and 3.

Results of this paper are proved combining classical tools from cutting plane theory (e.g. the lower bound on the CG rank of a polyhedron by Chvátal, Cook, and Hartmann [5], see Lemma 4) with geometric techniques that are not usually applied to the theory of CG cuts, mostly from geometry of numbers (such as the characterization of maximal lattice-free convex sets [3], or Minkowski's Convex Body Theorem).

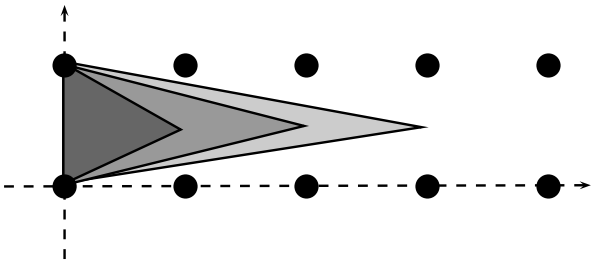


Fig. 1. In increasingly lighter shades of grey, polytopes Q_1, Q_2, and Q_3

Recall that a relaxation of an integral polyhedron is a *rational* polyhedron by definition. We remark that the rationality assumption is crucial in the statement of Theorem 1. As an example, consider the polytope $P \subseteq \mathbb{R}^2$ consisting only of the origin. Any line $Q \subseteq \mathbb{R}^2$ passing through the origin and having irrational slope is an (irrational) polyhedron whose integer hull is P. One readily verifies that the CG closure of Q is Q itself, showing that in this case the CG closures of Q do not converge to the integer hull P. However, no vector v satisfying the conditions of Theorem 1 exists.

The paper is organized as follows. In Section 2, we settle notation and definitions, and state some known and new auxiliary lemmas needed in the rest of the paper. In Section 3, we prove the main result of the paper, that is the geometric characterization of integral polyhedra with infinite reverse CG rank (Theorem 1). In Section 4, we focus on two classes of polyhedra with finite reverse CG rank and investigate upper bounds on r^* for those classes. We conclude in Section 5, where we point out open problems and derive first results on the reverse split rank.

2 Definitions and Tools

Throughout the paper, n will be a strictly positive integer denoting the dimension of the ambient space. Given a set $S \subseteq \mathbb{R}^n$, the *affine hull* of S, denoted $\text{aff}(S)$, is the smallest affine subspace containing S. The *dimension* of S is the dimension of $\text{aff}(S)$. S is *full-dimensional* if its dimension is n. We denote by $\text{int.cone}(S)$ the set of all linear combinations of vectors in S using nonnegative integer multipliers. Given a closed, convex set $C \subseteq \mathbb{R}^n$, we denote by C_I the integer hull of C, by $\text{int}(C)$ the interior of C, by $\text{relint}(C)$ the relative interior of C, and by $\text{bd}(C)$ the boundary of C. We say that C is *lattice-free* if $\text{int}(C) \cap \mathbb{Z}^n = \emptyset$, and *relatively lattice-free* if $\text{relint}(C) \cap \mathbb{Z}^n = \emptyset$. Note that the relative interior of a single point in $\mathbb{R}^n$ is the point itself. Hence, if it is integer, then it is not relatively lattice-free. Also, note that if C is not lattice-free, then it is full-dimensional. A *convex body* is a closed, convex, bounded set with non-empty interior. A set C is *centrally symmetric* with respect to a given point $x \in C$ (or centered at x) when, for every $y \in \mathbb{R}^n$, one has $x + y \in C$ if and only if $x - y \in C$.

By *distance* between two points $x, y \in \mathbb{R}^n$ (resp. a point $x \in \mathbb{R}^n$ and a set $S \subseteq \mathbb{R}^n$) we mean the Euclidean distance, which we denote by $d(x, y)$ (resp. $d(x, S)$). We use the standard notation $\|\cdot\|$ for the Euclidean norm. For $r \in \mathbb{Q}_+$, $x \in \mathbb{R}^n$ and an affine subspace $H \subseteq \mathbb{R}^n$ of dimension d, the *d–ball (of radius r lying on H and centered at x)* is the set of points lying on H whose distance from x is at most r. When referring to the volume of a d–dimensional convex set C, denoted $\text{vol}(C)$, we shall always mean its d-dimensional volume, that is, the Lebesgue measure with respect to the affine subspace $\text{aff}(C)$ of the Euclidean space $\mathbb{R}^n$.

Bounds on the CG Rank

We give here upper and lower bounds on the CG rank of polyhedra. The proof of the following two results can be found in [6] and [1] respectively.

Lemma 2. *Each rational polyhedron $Q \subseteq \mathbb{R}^n$ with $Q_I = \emptyset$ has CG rank at most $\varphi(n)$, where φ is a function depending on n only.*

Lemma 3. *For every polyhedron $Q \subseteq \mathbb{R}^n$ and for every $a \in \mathbb{Z}^n$ and $\delta, \delta' \in \mathbb{R}$ (with $\delta' \geq \delta$) such that $ax \leq \delta$ is valid for Q_I and $ax \leq \delta'$ is valid for Q, the inequality $ax \leq \delta$ is valid for $Q^{(p+1)}$, where $p = (\lfloor \delta' \rfloor - \lfloor \delta \rfloor) f(n)$ and f is a function depending on n only.*

In order to derive lower bounds, one can apply a result by Chvátal, Cook, and Hartmann [5] that gives sufficient conditions for a sequence of points to be in successive CG closures of a rational polyhedron. The one we provide next is a less general, albeit sufficient for our needs, version of their original lemma.

Lemma 4. *Let $Q \subseteq \mathbb{R}^n$ be a rational polyhedron, $x \in Q$, $v \in \mathbb{R}^n$, $p \in \mathbb{N}$ and, for $j \in \{1, \ldots, p\}$, let $x^j = x - j \cdot v$. Assume that, for all $j \in \{1, \ldots, p\}$ and every inequality $cx \leq \delta$ valid for Q_I with $c \in \mathbb{Z}^n$ and $cv < 1$, one has $cx^j \leq \delta$. Then $x^j \in Q^{(j)}$ for all $j \in \{1, \ldots, p\}$.*

As a corollary, we have the following result:

Lemma 5. *Let $Q \subseteq \mathbb{R}^n$ be a rational polyhedron, $x \in Q$, and $v \in \mathbb{Z}^n$ be such that $\{x - tv : t \geq 0\} \cap Q_I \neq \emptyset$. Let $\bar{t} = \min\{t \geq 0 : x - tv \in Q_I\}$. Then $r(Q) \geq \lceil \bar{t} \rceil$.*

Proof. By hypothesis, there exists a point $x' \in Q_I$ such that $x = x' + \bar{t}v$. We apply Lemma 4 with $p = \lceil \bar{t} \rceil - 1$. Let $cx \leq \delta$ be valid for Q_I, with c integer. If $cv < 1$, then $cv \leq 0$, since c and v are integer. Then for $j = 1, \ldots, \lceil \bar{t} \rceil - 1$, one has

$$cx^j = c(x - j \cdot v) = c(x' + (\bar{t} - j) \cdot v) = cx' + (\bar{t} - j)cv \leq \delta,$$

where the inequality follows from $x' \in Q_I$, $cv \leq 0$, $\bar{t} - j > 0$. Hence the hypothesis of Lemma 4 holds. We conclude $x^{\lceil \bar{t} \rceil - 1} \in Q^{(\lceil \bar{t} \rceil - 1)}$. Since by construction $x^{\lceil \bar{t} \rceil - 1} \notin Q_I$, the statement follows.

Unimodular Transformations

A *unimodular transformation* $u : \mathbb{R}^n \to \mathbb{R}^n$ maps a point $x \in \mathbb{R}^n$ to $u(x) = Ux + v$, where U is an $n \times n$ unimodular matrix (i.e. a square integer matrix with $|\det(U)| = 1$) and $v \in \mathbb{Z}^n$. It is well-known (see e.g. [17]) that a nonsingular matrix U is unimodular if and only if so is U^{-1}. Furthermore, a unimodular transformation is a bijection of both $\mathbb{R}^n$ and $\mathbb{Z}^n$ that preserves n–dimensional volumes. Moreover, the following holds ([7]).

Lemma 6. *Let $Q \subseteq \mathbb{R}^n$ be a polyhedron and $u : \mathbb{R}^n \to \mathbb{R}^n$, $u(x) = Ux + v$, be a unimodular transformation. Then for each $t \in \mathbb{N}$, an inequality $cx \leq \delta$ is valid for $Q^{(t)}$ if and only if the inequality $cU^{-1}x \leq \delta + cU^{-1}v$ is valid for $u(Q)^{(t)}$. Moreover, the CG rank of Q equals the CG rank of $u(Q)$.*

Thanks to the previous lemma, when investigating the CG rank of a d–dimensional rational polyhedron $Q \subseteq \mathbb{R}^n$ with $Q \cap \mathbb{Z}^n \neq \emptyset$, we can apply a suitable unimodular transformation and assume that the affine hull of Q is the rational subspace $\{x \in \mathbb{R}^n : x_{d+1} = x_{d+2} = \cdots = x_n = 0\}$.

3 Geometric Characterization of Integral Polyhedra with Infinite Reverse CG Rank

In this section we prove Theorem 1. Since it is already known that, when P is empty, $r^*(P) < +\infty$ (see Lemma 2), we assume $P \neq \emptyset$. We omit the easy proofs of Observation 7 and Observation 8.

Observation 7. *Let $P \subseteq \mathbb{R}^n$ be a polyhedron and $v \in \mathbb{R}^n$. Then* $\mathrm{relint}(P) + \langle v \rangle = \mathrm{relint}(P + \langle v \rangle)$.

Observation 8. *Let $C \subseteq \mathbb{R}^n$ be a convex set contained in a rational hyperplane such that $\mathrm{aff}(C) \cap \mathbb{Z}^n \neq \emptyset$. Then C is relatively lattice-free if and only if there exists $v \in \mathbb{Z}^n \setminus \mathrm{rec}(C)$ such that $C + \langle v \rangle$ is relatively lattice-free.*

3.1 Proof of Theorem 1: Sufficiency

Let $P \subseteq \mathbb{R}^n$ be a non-empty integral polyhedron and assume that $P + \langle v \rangle$ is relatively lattice-free for some $v \in \mathbb{Z}^n \setminus \mathrm{rec}(P)$: we prove that $r^*(P) = +\infty$. Let $\bar{x} \in \mathbb{R}^n$ be a point in the relative interior of P such that $\bar{x} + v \notin P$, and V be the set of vertices of P. For $\alpha \in \mathbb{Z}_+$, define $Q_\alpha = \mathrm{conv}(V, \bar{x} + \alpha v) + \mathrm{rec}(P)$. Q_α is a polyhedron and it strictly contains P. In order to prove that it is a relaxation of P, it suffices to show that $Q_\alpha \cap \mathbb{Z}^n = P \cap \mathbb{Z}^n$. $\bar{x} + \alpha v \in \mathrm{relint}(P) + \langle v \rangle$ hence, by Observation 7, $\bar{x} + \alpha v \in \mathrm{relint}(P + \langle v \rangle)$. Thus, for each $x \in Q_\alpha$, at least one of the following holds: x lies in P; x lies in the relative interior of $P + \langle v \rangle$, and since $P + \langle v \rangle$ is relatively lattice-free by hypothesis, x is not integer. This shows $Q_\alpha \cap \mathbb{Z}^n = P \cap \mathbb{Z}^n$. We now apply Lemma 5 with $Q = Q_\alpha$ and $x = \bar{x} + \alpha v$; note that $\lceil \bar{t} \rceil = \alpha$. Hence, we deduce that $r(Q_\alpha) \geq \alpha$. The thesis then follows from the fact that α was chosen arbitrarily in $\mathbb{Z}_+$.

3.2 Proof of Theorem 1: Necessity

First, we show that the non-full-dimensional case follows from the full-dimensional one. More precisely, assuming that the statement holds for any full-dimensional polyhedron, we let $P \subseteq \mathbb{R}^n$ be a non-empty integral polyhedron of dimension $d < n$ so that there is no $v \in \mathbb{Z}^n \setminus \mathrm{rec}(P)$ such that $P + \langle v \rangle$ is relatively lattice-free, and we show that $r^*(P) < +\infty$. Hence, let P be as above. Up to a unimodular transformation, we can assume that $\mathrm{aff}(P) = \{x \in \mathbb{R}^n : x_{d+1} = x_{d+2} = \cdots = x_n = 0\}$. Observation 8 implies that P is not relatively lattice-free. We then make use of the following fact [1, Theorem 1].

Theorem 9. *There exists a function $f : \mathbb{N} \to \mathbb{N}$ such that, for each integral polyhedron $P \subseteq \mathbb{R}^n$ that is not relatively lattice-free, and each relaxation Q of P, $Q^{(f(n))}$ is contained in* $\mathrm{aff}(P)$.

By Theorem 9, there is an integer p depending only on n such that, for each relaxation Q of P, $Q^{(p)} \subseteq \mathrm{aff}(P)$, i.e., modulo at most p iterations of the CG closure, we can assume that both P and Q are full-dimensional, and P is not lattice-free. Hence, $P + \langle v \rangle$ is not lattice-free for any $v \in \mathbb{Z}^d$, and $r^*(P) < +\infty$ follows from the full-dimensional case.

Therefore it suffices to show the statement for P full-dimensional: we assume that $P \subseteq \mathbb{R}^n$ is a non-empty integral polyhedron such that $r^*(P) = +\infty$, and prove that there exists $v \in \mathbb{Z}^n \setminus \mathrm{rec}(P)$ such that $P + \langle v \rangle$ is relatively lattice-free.

Let $Ax \leq b$ be an irredundant description of P, with $A \in \mathbb{Z}^{m \times n}$ and $b \in \mathbb{Z}^m$. For $k \in \mathbb{N}$, let $P_k = \{x \in \mathbb{R}^n : Ax \leq b + k \cdot \mathbf{1}\}$, where $\mathbf{1}$ denotes the m–dimensional all-one vector. The next claim is an application of Lemma 3 (we omit its easy proof).

Claim 1. *For each $k \in \mathbb{N}$, there exists a relaxation Q_k of P such that $Q_k \setminus P_k \neq \emptyset$.*

The rest of the proof is divided into the following steps: (a) We construct a candidate vector $v \notin \text{rec}(P)$; (b) We show that $P + \langle v \rangle$ is lattice-free; (c) We show that either v is integer or we can replace it with a suitable integer vector.

(a) Construction of $v \notin \text{rec}(P)$. By Claim 1, for every $k \in \mathbb{N}$ there exists a point $y^k \in Q_k \setminus P_k$. Let x^k be the point in P such that $d(y^k, x^k) = d(y^k, P)$ and define $v^k = y^k - x^k$.

Remark 1. *For every $k \in \mathbb{N}$, the hyperplane $H = \{x \in \mathbb{R}^n : v^k x = v^k x^k\}$ is a supporting hyperplane for P containing x^k.*

Consider the sequence of normalized vectors $\{\frac{v^k}{\|v^k\|}\}_{k\in\mathbb{N}}$. Since it is contained in the unit $(n-1)$–dimensional sphere S, which is a compact set, it has a subsequence that converges to an element of S, say v. We denote by $\mathcal{I}$ the set of indices of this subsequence. Remark 1 shows that every vector v^k belongs to the *optimality cone* of P, which is defined as the set of vectors c such that the problem $\max\{cx : x \in P\}$ has finite optimum. Since the optimality cone of a polyhedron is a polyhedral cone, in particular it is a closed set. Then v belongs to the optimality cone of P. This implies that $v \notin \text{rec}(P)$, as $\max\{cx : x \in P\}$ is never finite if c is a non-zero vector in $\text{rec}(P)$.

(b) $P + \langle v \rangle$ is lattice-free. Assume the existence of $\tilde{z} \in \mathbb{Z}^n$ for some $\tilde{z} \in \text{int}(P + \langle v \rangle)$. Observation 7 implies that there exist $\tilde{w} \in \text{int}(P)$ and $\alpha \in \mathbb{R}$ such that $\tilde{z} = \tilde{w} + \alpha v$. Since P is a rational polyhedron, $P = P^* + \text{int.cone}(R)$, where $R = \{r^1, \ldots, r^{|R|}\}$ is a set of integer generators of $\text{rec}(P)$ and P^* is a suitable rational polytope such that $\tilde{w} \in \text{int}(P^*)$ (for instance, if we let V be the vertex set of P, we can take $P^* = \text{conv}\{V \cup_{i=1}^{|R|} (\tilde{w} + r^i)\}$). We denote the *geometric diameter* of P^* (i.e. the maximum distance between two points of P^*) by δ (see Figure 2).

Claim 2. *There exist a number $\beta > 2\delta$ and points $w \in \text{int}(P^*)$ and $z \in \mathbb{Z}^n$, such that $z = w + \beta v$.*

Proof. We make use of the following fact, shown by Basu et al. [3, Lemma 13] as a consequence of the well-known Dirichlet's Lemma: *Given $u \in \mathbb{Z}^n$ and $r \in \mathbb{R}^n$, then for every $\varepsilon > 0$ and $\bar{\lambda} \geq 0$, there exists an integer point at distance less than ε from the halfline $\{u + \lambda r : \lambda \geq \bar{\lambda}\}$.* Apply this result with $u = \tilde{z}$, $r = v$, $0 < \varepsilon < d(\tilde{w}, \text{bd}(P^*))$, and $\bar{\lambda} = \max(0, 2\delta - \alpha + \varepsilon)$. It guarantees the existence of an integer point z at distance less than ε from the halfline $\{\tilde{z} + \lambda v : \lambda \geq 2\delta - \alpha + \varepsilon\} = \{\tilde{w} + \lambda v : \lambda \geq 2\delta + \varepsilon\}$. Then $z = w + \beta v$ for some point w at distance less than ε from $\tilde{w}$ and $\beta > 2\delta$. As $\varepsilon < d(\tilde{w}, \text{bd}(P^*))$, it follows that $w \in \text{int}(P^*)$. ◇

Let β, w, z be as in Claim 2. If for $a \in \mathbb{Z}_+^{|R|}$ we define $P^*(a) = P^* + \sum_{i=1,\ldots,|R|} a_i r^i$, then $P = \bigcup_{a \in \mathbb{Z}_+^{|R|}} P^*(a)$ (see again Figure 2). Recall that, for $k \in \mathbb{N}$, one has $y^k \in Q_k \setminus P_k$, $x^k \in P$, and $v^k = y^k - x^k$. For $k \in \mathbb{N}$, let $a^k \in \mathbb{Z}_+^{|R|}$ be such that $x^k \in P^*(a^k)$. Also, let $w^k = w + \sum_{i=1}^{|R|} a_i^k r^i$. Note that each w^k is a translation

of w by an integer combination of integer vectors $r^1, \ldots, r^{|R|}$, so that w^k lies in the same translation of P^* as x^k. This implies $d(w^k, x^k) \leq \delta$. For each $k \in \mathbb{N}$, we also define $z^k = w^k + \beta v$. One easily checks that $z^k = z + \sum_{i=1}^{|R|} a_i^k r^i$, that is, z^k is a translation of z by integer vectors $r^1, \ldots, r^{|R|}$ with the same multipliers as w^k, hence it is an integer vector. The proof of (b) is an immediate consequence of the following claim, which contradicts the fact that Q_k is a relaxation of P for every $k \in \mathbb{N}$.

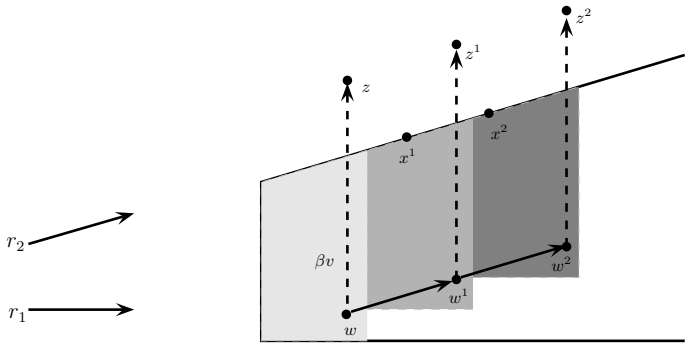


Fig. 2. Illustration from part (b) of the proof of Theorem 1. On the left: the vectors r^1 and r^2 from $\mathrm{rec}(P)$. On the right: polytope P, and its covering with polyhedra $P^*(a)$, $a \in \mathbb{Z}_+$. In increasingly darker shadows of grey: $P^* = P^*\binom{0}{0}$, $P^*\binom{0}{1}$, $P^*\binom{0}{2}$. If moreover $x^1 \in P^*\binom{0}{1}$ and $x^2 \in P^*\binom{0}{2}$, we obtain w^1, w^2, z^1, z^2 as in the picture.

Claim 3. *$z^k \in Q_k \setminus P$ for each $k \in \mathcal{I}$ large enough.*

Proof. We first show that $z^k \notin P$ for $k \in \mathcal{I}$ large enough. By Remark 1, the hyperplane $H = \{x \in \mathbb{R}^n : v^k x = v^k x^k\}$ is a supporting hyperplane of P containing x^k. Let $\gamma \in \mathbb{R}$ be such that $w^k + \gamma \frac{v^k}{\|v^k\|} \in H$. Note that γ is well-defined since v^k is normal to H, and moreover $\gamma \geq 0$, as $w^k \in P$. Since $\frac{v^k}{\|v^k\|}$ is a unit vector normal to H, one has

$$\gamma = d(w^k, H) \leq d(w^k, x^k) \leq \delta, \tag{1}$$

where the first inequality comes from the fact that $x^k \in H$.

Let now ϕ be the angle between v^k and v, and $k \in \mathcal{I}$ be large enough, so that $0 \leq \phi \leq \frac{\pi}{3}$. Let $\sigma \in \mathbb{R}$ be such that $w^k + \sigma v \in H$ (recall that $\|v\| = 1$). By simple trigonometric arguments and by (1), we obtain $\sigma = \frac{\gamma}{\cos \phi} \leq 2\delta$. Hence, points $w^k + \lambda v$ with $\lambda > 2\delta$ do not belong to P. In particular, $z^k \notin P$, since $z^k = w^k + \beta v$ with $\beta > 2\delta$ from Claim 2.

We now show that $z^k \in Q_k$ for $k \in \mathcal{I}$ large enough. Let ε be such that $0 < \varepsilon < d(w, \mathrm{bd}(P^*))$. Note that $\varepsilon < d(w^k, \mathrm{bd}(P))$ for all $k \in \mathbb{N}$. For each $k \in \mathbb{N}$,

let H^k be the hyperplane with normal v containing point w^k, i.e., $H^k = \{x : vx = vw^k\}$. Define B^k to be the $(n-1)$–ball of radius ε lying on H^k and centered at w^k. Note that $B^k \subseteq P$. Let C be the cone generated by the set of vectors $\{z^k - x : x \in B^k\}$: the definition of C does not depend on k, and C is indeed a *cone of revolution* defined by direction v and some angle $0 < 2\theta < \pi/2$ (see Figure 3), i.e. C is the set of vectors of $\mathbb{R}^n$ that form an angle of at most 2θ with v. Note that

$$z^k \in \text{conv}(x, B^k) \quad \text{for every } x \in z^k + C. \tag{2}$$

Now let D be the cone of revolution of direction v and angle θ. Note that D is strictly contained in cone C. Since $d(x^k, w^k) \leq \delta$ for all k, there exists a positive number τ such that $\{x \in x^k + D : d(x, x^k) \geq \tau\} \subseteq z^k + C$ for all $k \in \mathbb{N}$. Since $\lim_{k\to+\infty} d(y^k, P) = +\infty$, for $k \in \mathbb{N}$ large enough $d(y^k, x^k) \geq d(y^k, P) \geq \tau$. If moreover we take $k \in \mathcal{I}$ large enough so that the angle between v^k and v is at most θ, one has $y^k \in x^k + D$ and consequently $y^k \in z^k + C$. Because $y^k \in Q_k$ and (2), we conclude that $z^k \in Q_k$, as required. ◇

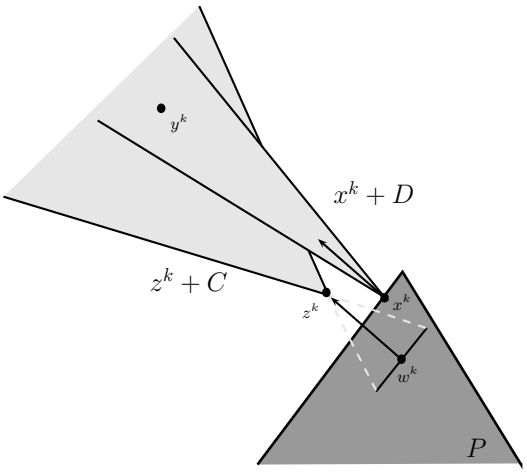


Fig. 3. Illustration from the proof of Claim 3. Both C and D are cones of revolution defined by direction v, with angles respectively 2θ and θ.

(c) Either v is integer, or we can replace it with a suitable integer vector. This is immediate if v is rational, so assume that it is not. In [3, Theorem 2] (see also [13]) it is proved that a maximal lattice-free convex set is either an irrational affine hyperplane of $\mathbb{R}^n$, or a polyhedron $Q + L$, where Q is a polytope and L is a rational linear space. $P + \langle v \rangle$ is lattice-free, thus it is contained in a maximal lattice-free convex set. Since it is full-dimensional, it is not contained in an irrational hyperplane. It follows that $P \subseteq Q + L$, with Q, L as above. Moreover, L has dimension at least 1, since it contains v. Pick a set $S \subseteq \mathbb{Z}^n$ of

generators of L such that $v \in \text{cone}(S)$. Since $v \notin \text{rec}(P)$, then $s \notin \text{rec}(P)$ for at least one $s \in S$. Moreover, $P + \langle s \rangle \subseteq Q + L$ and it is full-dimensional, hence it is lattice-free. We can then replace v by s. This concludes the proof of Theorem 1.

4 On Some Polyhedra with Finite Reverse CG Rank

In this section we investigate the behavior of the reverse CG rank for two classes of polyhedra. Namely, let $\mathcal{A}$ be the family of integral polyhedra P such that (i) no facet of P is relatively lattice-free and (ii) either P is not relatively lattice-free or P is full-dimensional; also, let $\mathcal{B}$ the family of integral polyhedra that are not relatively lattice-free. We show the following.

Theorem 10. (i) *For each $n \in \mathbb{N}$, $\sup\{r^*(P) : P \subseteq \mathbb{R}^n, P \in \mathcal{A}\} \leq \lambda(n)$, where λ is a function depending on n only.*
(ii) *For each $n, k \in \mathbb{N}$, $\sup\{r^*(P) : P \subseteq \mathbb{R}^n, P \in \mathcal{B}, |\operatorname{relint}(P) \cap \mathbb{Z}^n| \leq k\} \leq \mu(n,k)$, where μ is a function depending on n and k only.*

We build on the following result [1, Theorem 12].

Theorem 11. *There exists a function $\phi : \mathbb{N} \to]0, +\infty[$ such that every integral non-lattice-free polyhedron $P \subseteq \mathbb{R}^n$ contains a centrally symmetric polytope of volume $\phi(n)$, whose only integer point is its center.*

Lemma 12. *Let $P \subseteq \mathbb{R}^n$ be an integral polyhedron. Let $cx \leq \delta$ be a valid inequality for P inducing a facet of P that is not relatively lattice-free. Then, for every relaxation Q of P contained in $\text{aff}(P)$, $cx \leq \delta$ is valid for $Q^{(p)}$, where p depends on n only.*

Proof. Let P be d–dimensional and F be the facet of P induced by inequality $cx \leq \delta$. Modulo a unimodular transformation, we can assume that $\text{aff}(P) = \{x \in \mathbb{R}^n : x_{d+1} = \cdots = x_n = 0\}$ and $cx \leq \delta$ is the inequality $x_d \leq 0$. If $d = 0$, then there is nothing to prove, as P has no facet; and if $d = 1$, then $Q^{(1)} = P$ since Q is a relaxation of P contained in $\text{aff}(P)$. Thus we assume $d \geq 2$. By Theorem 11, F contains a $(d-1)$-dimensional centrally symmetric polytope E of volume $\phi(d-1)$, whose only integer point is its center. We assume wlog that this point is the origin. We now argue that the inequality $x_d \leq \frac{d \cdot 2^{d-1}}{\phi(d-1)}$ is valid for each relaxation Q of P contained in $\text{aff}(P)$. Assume that this is not true, i.e., there exists a point $\bar{x} \in Q$ with $\bar{x}_d > \frac{d \cdot 2^{d-1}}{\phi(d-1)}$. Define $C = \text{conv}(E, \bar{x}) \subseteq Q$. Since Q is a relaxation of P, C is a d–dimensional convex body whose only integer point is the origin, which lies on its boundary. Moreover,

$$\text{vol}(C) = \bar{x}_d \cdot \frac{\text{vol}(E)}{d} > \frac{d \cdot 2^{d-1}}{\phi(d-1)} \cdot \frac{\phi(d-1)}{d} = 2^{d-1}.$$

Let C' be the symmetrization of C w.r.t. the origin, i.e. $C' = C \cup -C$. Note that C' is a d–dimensional centrally symmetric polytope in the space of the

first d variables whose only integer point is the origin. Furthermore, $\mathrm{vol}(C') = 2\mathrm{vol}(C) > 2^d$. However, by Minkowski's Convex Body Theorem (see, e.g., [2]), every centrally symmetric convex body in $\mathbb{R}^d$ whose only integer point is the origin has volume at most 2^d. This is a contradiction. Therefore the inequality $x_d \leq \frac{d \cdot 2^{d-1}}{\phi(d-1)}$ is valid for each relaxation Q of P contained in $\mathrm{aff}(P)$. Lemma 3 then implies that $cx \leq \delta$ is valid for $Q^{(p)}$, where p depends on n and d. To eliminate the dependence on d, it is sufficient to replace, in the above argument, the right-hand side of inequality $x_d \leq \frac{d \cdot 2^{d-1}}{\phi(d-1)}$ with $\max\{\frac{d \cdot 2^{d-1}}{\phi(d-1)} : 2 \leq d \leq n\}$.

Proof of Theorem 10. (i). Let $P \in \mathcal{A}$. Then no facet of P is relatively lattice-free. Suppose first that P is full-dimensional. By Lemma 12, for each facet-defining inequality $cx \leq \delta$ of P, $cx \leq \delta$ is valid for $Q^{(p)}$, with p depending on n only. This implies that $Q^{(p)} = P$, concluding the proof. Now, assume that P is of dimension $d < n$. By definition of $\mathcal{A}$, P is not relatively lattice-free. Theorem 9 implies that there exists a number p depending only on n such that, for each relaxation Q of P, $Q^{(p)} \subseteq \mathrm{aff}(P)$. Thus in a number of iterations of the CG closure depending on n only we are back to the full-dimensional case. This proves (i).

(ii). Now fix $n, k \in \mathbb{N}$, $k \geq 1$, and consider the family of polyhedra $P \subseteq \mathbb{R}^n$, $P \in \mathcal{B}$, with $|\operatorname{relint}(P) \cap \mathbb{Z}^n| = k$. Actually, this family is only composed of polytopes, as every unbounded integral polyhedron with an integer point in its relative interior contains infinitely many of those. By Theorem 1, $r^*(P)$ is finite for each polytope from this family. Lagarias and Ziegler [11] showed that, up to unimodular transformations, for each d and $k \geq 1$ there is only a finite number of d–dimensional polytopes with k integer points in their relative interior. Hence there exists a number $t_{n,k}$ such that $r^*(P) \leq t_{n,k}$ for all polytopes $P \subseteq \mathbb{R}^n$ with $P \in \mathcal{B}$ and $|\operatorname{relint}(P) \cap \mathbb{Z}^n| = k$, concluding the proof of (ii).

It is not difficult to provide examples showing that, in general, bounds on $r^*(P)$ for the classes of polyhedra $\mathcal{A}$ and $\mathcal{B}$ *must* depend on the parameters considered in Theorem 10: we defer them to the journal version of the paper.

5 Concluding Remarks

Though Theorem 1 gives a characterization of the integral polyhedra having infinite reverse CG rank, it would be interesting to have an alternative (and perhaps more explicit) characterization of those integral polyhedra for which a vector v as in the statement of the theorem exists. The results in this paper give partial answer to this question. For instance, for a non-full-dimensional integral polyhedron P, such a v exists if and only if P is not relatively lattice-free (see Observation 8). For full-dimensional polyhedra, the situation is different. Theorem 10 shows that full-dimensional lattice-free integral polyhedra with an integer point in the relative interior of each facet have finite reverse CG rank. For these polyhedra, the non-existence of a direction v as in the statement of Theorem 1 is due to the fact that by applying any direction $v \notin \mathrm{rec}(P)$, one of the integer points in the facets of the polyhedron will fall in the interior of

$P+\langle v\rangle$. However, this is not a necessary condition for an integral polytope to have finite reverse CG rank. As an example, consider the polytope $P = \text{conv}\{(1,0,0), (0,2,0), (2,1,0), (2,1,2)\} \subseteq \mathbb{R}^3$. If we take, e.g., $v = (0,1,0)$, then $P + \langle v\rangle$ does not contain any integer point of P in its interior, but $P + \langle v\rangle$ is not lattice-free, as $(1,1,1)$ is in its interior.

Another interesting problem is the extension of the concept of reverse CG rank to the case of split inequalities. It can be proved that in dimension 2 the split rank of every rational polyhedron is at most 2. That is, the *reverse split rank* (defined in the obvious way) is bounded by a constant in dimension 2, while recall that this is not true for the reverse CG rank, see Section 1. However, already in dimension 3 a constant bound does not exists, as implied by the following lemma, whose proof we defer to the journal version of the paper.

Lemma 13. *Let* $T \subseteq \mathbb{R}^3$ *be the triangle* $\text{conv}\{(0,0,0), (0,2,0), (2,0,0)\}$. *The reverse split rank of* T *is* $+\infty$.

References

1. Averkov, G., Conforti, M., Del Pia, A., Di Summa, M., Faenza, Y.: On the convergence of the affine hull of the Chvátal-Gomory closures (2012) (submitted manuscript), `http://arxiv.org/abs/1210.6280`
2. Barvinok, A.: A course in Convexity. American Mathematical Society (2002)
3. Basu, A., Conforti, M., Cornuéjols, G., Zambelli, G.: Maximal lattice-free convex sets in linear subspaces. Math. Oper. Res. 35, 704–720 (2010)
4. Bockmayr, A., Eisenbrand, F., Hartmann, M., Schulz, A.S.: On the Chvátal rank of polytopes in the 0/1 cube. Discrete Appl. Math. 98, 21–27 (1999)
5. Chvátal, V., Cook, W., Hartmann, M.: On cutting-plane proofs in combinatorial optimization. Linear Algebra Appl. 114/115, 455–499 (1989)
6. Cook, W., Coullard, C.R., Túran, G.: On the complexity of cutting-plane proofs. Discrete Appl. Math. 18, 25–38 (1987)
7. Eisenbrand, F., Schulz, A.S.: Bounds on the Chvátal rank of polytopes in the 0/1 cube. Combinatorica 23, 245–261 (2003)
8. Gomory, R.E.: Outline of an algorithm for integer solutions of linear programs. Bull. Amer. Math. Soc. (N.S.) 64, 275–278 (1958)
9. Grünbaum, B.: Convex Polytopes. Springer (2003)
10. ILOG CPLEX 11 Documentation, `http://www-eio.upc.es/lceio/manuals/cplex-11/html/`
11. Lagarias, J., Ziegler, G.: Bounds for lattice polytopes containing a fixed number of interior points in a sublattice. Canadian J. Math. 43, 1022–1035 (1991)
12. Li, Y.: Personal Communication (2012)
13. Lovász, L.: Geometry of Numbers and Integer Programming. In: Iri, M., Tanabe, K. (eds.) Mathematical Programming: Recent Developements and Applications, pp. 177–210. Kluwer (1989)
14. Pokutta, S., Stauffer, G.: Lower bounds for the Chvátal-Gomory rank in the 0/1 cube. Oper. Res. Lett. 39, 200–203 (2011)
15. Rothvoß, T., Sanità, L.: 0/1 polytopes with quadratic Chvátal rank. In: Proceedings of IPCO 2013 (to appear, 2013)
16. Schrijver, A.: On cutting planes. Ann. Discrete Math. 9, 291–296 (1980)
17. Schrijver, A.: Theory of Linear and Integer Programming. John Wiley (1986)

On Some Generalizations of the Split Closure

Sanjeeb Dash[1], Oktay Günlük[1], and Diego Alejandro Morán Ramirez[2]

[1] IBM Research
[2] Georgia Institute of Technology

Abstract. Split cuts form a well-known class of valid inequalities for mixed-integer programming problems (MIP). Cook et al. (1990) showed that the split closure of a rational polyhedron P is again a polyhedron. In this paper, we extend this result from a single rational polyhedron to the union of a finite number of rational polyhedra. We also show how this result can be used to prove that some generalizations of split cuts, namely cross cuts, also yield closures that are rational polyhedra.

Keywords: Cross cuts, closure, polyhedrality.

1 Introduction

Cutting planes (or *cuts*, for short) are crucial for solving mixed-integer programs (MIPs), and currently the most effective cuts for general MIPs are the *split cuts*. In their seminal paper Cook, Kannan and Schrijver [8] studied split cuts which can also be seen as a class of disjunctive cuts that generalize GMI cuts. In [8], Cook et al. showed that the split closure of a rational polyhedron P – that is, the set of points in P satisfying all split cuts for P – is again a polyhedron. This is not a trivial result as one has to consider infinitely many split cuts associated with P.

Recently there has been substantial work on generalizing split cuts in different ways to obtain new and more effective classes of cutting planes, and analogues of the polyhedrality of the split closure result have been obtained for some of these classes. Andersen et. al. [3] studied cuts obtained from two dimensional convex lattice-free sets, and Andersen, Louveaux and Weismantel [2] showed that the set of points in a rational polyhedron satisfying all cuts from lattice-free sets with bounded max-facet-width is a polyhedron. Averkov [4] recently gave a short proof of this latter result. Averkov, Wagner and Weismantel [5] showed that the closure with respect to integral lattice-free sets is a polyhedron. In another recent paper, Basu et. al. [7] show that the triangle closure (points satisfying cuts obtained from maximal lattice-free triangles) of a polyhedron in a specific family (the two-row continuous group relaxation) is a polyhedron. As a generalization of split cuts, recently Dash, Dey and Günlük [10] studied cuts which are obtained by considering two split sets simultaneously. These cuts are called *cross cuts* and are equivalent to the 2-branch split cuts of Li and Richard [14]. In this paper, we generalize Cook et al's result from a single rational polyhedron to the union of a finite number of rational polyhedra and use this result to show that the cross cut closure of a rational polyhedron is a polyhedron.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 145–156, 2013.

We next formally define split sets, split cuts for a given polyhedron (all polyhedra in this paper are assumed to be rational) and the split closure of a polyhedron. Let $(\pi, \pi_0) \in \mathbb{Z}^n \times \mathbb{Z}$, then the split set associated with (π, π_0) is defined to be

$$S(\pi, \pi_0) = \{x \in \mathbb{R}^n : \pi_0 < \pi^T x < \pi_0 + 1\}.$$

Clearly, $S(\pi, \pi_0) \cap \mathbb{Z}^n = \emptyset$ and consequently the integer points contained in a polyhedron $P \subset \mathbb{R}^n$ are the same as the ones contained in $\text{conv}(P \setminus S(\pi, \pi_0)) \subset \mathbb{R}^n$. Linear inequalities that are valid for $\text{conv}(P \setminus S(\pi, \pi_0))$ are called split cuts generated by the split set $S(\pi, \pi_0)$.

Let $\mathcal{S}^* = \{S(\pi, \pi_0) : (\pi, \pi_0) \in \mathbb{Z}^n \times \mathbb{Z}\}$ denote the collection of all split sets and let $\mathcal{S} \subseteq \mathcal{S}^*$ be given. The split closure of a set $A \subseteq \mathbb{R}^n$, with respect to $\mathcal{S}$ is defined as

$$\text{SC}(A, \mathcal{S}) = \bigcap_{S \in \mathcal{S}} \text{conv}\,(A \setminus S)\,,$$

where for a given set $X \subseteq \mathbb{R}^n$, we denote its convex hull by $\text{conv}(X)$. We refer to $\text{SC}(A, \mathcal{S}^*)$ as the split closure of A and denote it as $\text{SC}(A)$.

Cook, Kannan and Schrijver [8] showed that if P is a rational polyhedron, then $\text{SC}(P)$ is also a rational polyhedron. Several other proofs of this result can also be found in [1], [2], [11] and [17]. A crucial step in most of these proofs is to show that there exists a finite set $\widehat{\mathcal{S}} \subseteq \mathcal{S}^*$ such that $\text{SC}(P, \mathcal{S}) = \text{SC}(P, \widehat{\mathcal{S}})$. When such $\widehat{\mathcal{S}}$ exists, we say that the split closure is finitely generated. For a non-polyhedral set the split closure is not necessarily polyhedral. However, in some cases it can be finitely generated (see for example [9]). We show the following generalization to a finite union of rational polyhedra:

Theorem 1. *Let P_k be rational polyhedra for $k \in \mathcal{K}$ where $\mathcal{K}$ is a finite set and let $P = \bigcup_{k \in \mathcal{K}} P_k$. Then $\text{SC}\,(P, \mathcal{S})$ is finitely generated for any $\mathcal{S} \subseteq \mathcal{S}^*$.*

Note that Theorem 1 does not always implies that $\text{SC}\,(P, \mathcal{S})$ is polyhedral as it is easy to see that for $P_1 = \{(0, 0)\}$ and $P_2 = \{x \in \mathbb{R}^2 \ : \ x_2 = 1\}$ we have $\text{SC}\,(P_1 \cup P_2, \mathcal{S}^*) = \text{conv}(P_1 \cup P_2)$ which is not a polyhedron.

As a generalization of split cuts, recently Dash, Dey and Günlük [10] studied *cross cuts*. Let $\mathcal{S}_1, \mathcal{S}_2 \subseteq \mathcal{S}^*$ be two collections of split sets. A cross disjunction is a pair (S_1, S_2), where $S_1 \in \mathcal{S}_1, S_2 \in \mathcal{S}_2$. The cross closure of a set $A \subseteq \mathbb{R}^n$, with respect to $\mathcal{S}_1, \mathcal{S}_2$, is defined as

$$\text{CC}(A, \mathcal{S}_1, \mathcal{S}_2) = \bigcap_{S_1 \in \mathcal{S}_1, S_2 \in \mathcal{S}_2} \text{conv}\,(A \setminus (S_1 \cup S_2))\,,$$

and the cross closure of P is $\text{CC}(A, \mathcal{S}^*, \mathcal{S}^*)$, denoted simply by $\text{CC}(A)$. In Section 5 we show another generalization of Cook, Kannan and Schrijver's result, this time to cross cuts:

Theorem 2. *Let P be a rational polyhedron. Then $\text{CC}(P)$ is a polyhedron.*

2 Preliminaries

The two main ingredients that we use in this paper are the so-called Gordan-Dickson Lemma, and, the analysis of intersection points of (closed) split sets and half-lines that have their end point contained in the split set. In [1], Anderson, Cornuejols and Li give an alternate proof of the polyhedrality of the split closure of polyhedra using a new proof technique. We next summarize the relevant results from [1], and state the Gordan-Dickson Lemma.

We start with defining the point where a rational half-line $H = \{v + \lambda r : \lambda \geq 0\}$, where $v, r \in \mathbb{Q}^n$, intersects for the first time the complement of a split set $S \in \mathcal{S}^*$ that contains the end point v of H.

Definition 1 (Intersection point step size). *Let $v, r \in \mathbb{Q}^n$ and $S \in \mathcal{S}^*$ such that $v \in S$, then*

$$\lambda_{vr}(S) = \sup\{\lambda : v + \lambda r \in S\}.$$

Given a split set $S = S(\pi, \pi_0)$, the step size can be explicitly computed as follows:

$$\lambda_{vr}(S) = \begin{cases} \frac{\pi^T v - \pi_0}{-\pi^T r} & \pi^T r < 0 \\ \frac{\pi_0 + 1 - \pi^T v}{\pi^T r} & \pi^T r > 0 \\ +\infty & \pi^T r = 0 \end{cases}$$

Furthermore, notice that if $\pi^T r > 0$, then the point $p = v + \lambda_{vr}(S)\, r$ is the point where the half-line $H = \{v + \lambda r : \lambda \geq 0\}$ intersects the hyperplane $\{x \in \mathbb{R}^n : \pi^T x = \pi_0\}$. If, on the other hand, $\pi^T r < 0$ then p is the intersection point with the hyperplane $\{x \in \mathbb{R}^n : \pi^T x = \pi_0 + 1\}$. We next review some properties of $\lambda_{vr}(S)$ presented in [1] and [2].

Lemma 1 (Lemma 5 in [1]). *Let $S \in \mathcal{S}^*$ and $H = \{v + \lambda r : \lambda \geq 0\}$ where $v, r \in \mathbb{Q}^n$ and $v \in S$. If $\lambda_{v,r}(S) < +\infty$, then $\lambda_{vr}(S) < \min\{z \in \mathbb{Z}_+ : zr \in \mathbb{Z}^n\}$.*

Lemma 2 (Lemma 6 in [1]). *Let $\lambda^* > 0$ and $H = \{v + \lambda r : \lambda \geq 0\}$ where $v, r \in \mathbb{Q}^n$. Then there exists a finite set $\Lambda \in \mathbb{R}$ such that for all $S \in \mathcal{S}^*$, $\lambda_{vr}(S) \in \Lambda$ provided that $\infty > \lambda_{vr}(S) > \lambda^*$ and $v \in S$.*

For a rational polyhedron P, we denote by $V(P) \subseteq \mathbb{Q}^n$ its set of vertices and by $E(P) \subseteq \mathbb{Q}^n$ its set of extreme rays. When $V(P) \neq \emptyset$, we say that the polyhedron is pointed.

Definition 2 (Relevant directions). *For a vertex $v \in V(P)$, we define*

$$\begin{aligned} D_v(P) = &\big\{r \in E(P) : \{v + \lambda r : \lambda \in \mathbb{R}_+\} \text{ is a 1-dimensional face of } P\big\} \\ &\cup \big\{v' - v : v' \in V(P), \text{ and } \operatorname{conv}(v, v') \text{ is a 1-dimensional face of } P\big\} \end{aligned}$$

to denote the set of relevant directions for the vertex v.

Observe that for $v \in V(P)$, the relevant directions are the extreme rays of the radial cone at the vertex v in the polyhedron P.

The following result is originally presented in [1] for conic polyhedra and later generalized by Andersen, Louveaux, and Weismantel [2] to general polyhedra.

Lemma 3 (Lemmas 2.3, 2.4, 4.2 in [1,2]). *Let P be a pointed rational polyhedron and let $S \in \mathcal{S}$. If $P \setminus S \neq \emptyset$, then (1) $\operatorname{conv}(P \setminus S)$ is a rational polyhedron. (2) The extreme rays of $\operatorname{conv}(P \setminus S)$ are the same as the extreme rays of P. (3) If u is a vertex of of $\operatorname{conv}(P \setminus S)$, then either $u \in V(P) \setminus S$, or, $u = v + \lambda_{vr}(S)\, r$, where $v \in V(P) \cap S$ and $r \in D_v(P)$ satisfies one of the following: (i) $r \in E(P)$ and $\lambda_{vr}(S) < +\infty$, or, (ii) $r = v' - v$ for some $v' \in V(P) \setminus S$ such that $\operatorname{conv}(v, v')$ is an edge of P and $\lambda_{vr}(S) < 1$.*

Finally we state a very simple and useful lemma that shows that for any positive integer p, every set of p-tuples of natural numbers has finitely many minimal elements.

Lemma 4 (Gordan-Dickson Lemma). *Let $X \subseteq \mathbb{Z}^p_+$. Then there exists a finite set $Y \subseteq X$ such that for every $x \in X$ there exists $y \in Y$ satisfying $x \geq y$.*

3 Split Closure of a Finite Collection of Polyhedral Sets

In this section, we show that given a finite collection of rational polyhedra, there exists a finite set of splits that define the split closure. We start with a simple observation based on Lemma 3.

Corollary 1. *Let P be a rational pointed polyhedron and $S_1, S_2 \in \mathcal{S}^*$ such that $V(P) \cap S_1 = V(P) \cap S_2$. If*

$$\lambda_{vr}(S_1) \geq \lambda_{vr}(S_2), \text{ for all } v \in V(P) \cap S_1 \text{ and } r \in D_v(P), \tag{1}$$

then $\operatorname{conv}(P \setminus S_1) \subseteq \operatorname{conv}(P \setminus S_2)$.

Proof. The claim clearly holds when $\operatorname{conv}(P \setminus S_1) = \emptyset$ and therefore we assume that $\operatorname{conv}(P \setminus S_1) \neq \emptyset$. Notice that by Lemma 3 $\operatorname{conv}(P \setminus S_1)$ and $\operatorname{conv}(P \setminus S_2)$ are polyhedral and have the same recession cone. Moreover, (1) implies that the vertices of $\operatorname{conv}(P \setminus S_1)$ belong to $\operatorname{conv}(P \setminus S_2)$. Therefore, $\operatorname{conv}(P \setminus S_1) \subseteq \operatorname{conv}(P \setminus S_2)$. ∎

Using Lemmas 1 and 2 we obtain the following result.

Lemma 5. *Let $v, r \in \mathbb{Q}^n$. There exists a function $x_{vr} : \mathcal{S}^* \to \mathbb{Z}_+$ such that whenever $v \in S_1, S_2 \in \mathcal{S}^*$, we have,*

$$x_{vr}(S_1) \leq x_{vr}(S_2) \Leftrightarrow \lambda_{vr}(S_1) \geq \lambda_{vr}(S_2). \tag{2}$$

Proof. Let $\Lambda = \{\lambda_{vr}(S) : v \in S \text{ and } \lambda_{vr}(S) < +\infty\}$ and $M_r = \min\{z \in \mathbb{Z}_+ : zr \in \mathbb{Z}^n\}$. Define

$$x_{vr}(S) = \begin{cases} 0, & \lambda_{vr}(S) = +\infty \\ |\Lambda \cap [\lambda_{vr}(S), M_r]|, & \lambda_{vr}(S) < +\infty. \end{cases}$$

Notice that $x_{vr}(S)$ is well defined for all $S \in \mathcal{S}^*$ as $\lambda_{vr}(S) < M_r$ by Lemma 1 and $|\Lambda \cap [\lambda_{vr}(S), M_r]| < +\infty$ by Lemma 2.

When $\lambda_{vr}(S_1) = +\infty$ we have $x_{vr}(S_1) = 0$ and the equivalence (2) clearly holds. If, on the other hand, $\lambda_{vr}(S_1) < +\infty$, we obtain that $\lambda_{vr}(S_1) \geq \lambda_{vr}(S_2)$ is equivalent to $x_{vr}(S_1) \leq x_{vr}(S_2)$, since it is easy to see that the latter occurs if and only if $|\Lambda \cap [\lambda_{vr}(S_1), M_r]| \leq |\Lambda \cap [\lambda_{vr}(S_2), M_r]|$. ∎

This observation together with the Gordan-Dickson Lemma (Lemma 4) can be used to show that the split closure of a polyhedron is again a polyhedron. In [4], Averkov uses a similar argument to show the polyhedrality of more general closures that include the split closure. We next use Lemma 5 for split closures of unions of polyhedra.

Consider a finite collection of pointed rational polyhedra $P_k, k \in \mathcal{K}$, where $\mathcal{K}$ is a finite set. For $V' \subseteq \bigcup_{k\in\mathcal{K}} V(P_k)$ we denote $\mathcal{S}(V') = \{S \in \mathcal{S} : V' = \bigcup_{k\in\mathcal{K}} V(P_k) \cap S\}$.

Proposition 1. *Let $\mathcal{S} \subseteq \mathcal{S}^*$ and $\{P_k\}_{k\in\mathcal{K}}$ be a finite collection of pointed rational polyhedra. Then, there exists a finite set $\mathcal{S}_Y \subseteq \mathcal{S}$ such that for all $S_1 \in \mathcal{S}$ there exists $S_2 \in \mathcal{S}_Y$ such that*

$$\operatorname{conv}(P_k \setminus S_2) \subseteq \operatorname{conv}(P_k \setminus S_1) \quad \text{for all } k \in \mathcal{K}.$$

Proof. Notice that sets $\mathcal{S}(V')$ for $V' \subseteq \bigcup_{k\in\mathcal{K}} V(P_k)$ form a finite partition of $\mathcal{S}$, that is, $\mathcal{S}(V') \cap \mathcal{S}(V'') = \emptyset$ if $V' \neq V''$, and

$$\mathcal{S} = \bigcup_{V' \subseteq \bigcup_{k\in\mathcal{K}} V(P_k)} \mathcal{S}(V').$$

Consequently, it suffices to show the existence of finite sets $\mathcal{S}_Y(V') \subseteq \mathcal{S}(V')$ for each $V' \subseteq \bigcup_{k\in\mathcal{K}} V(P_k)$ that satisfy the claim when $S_1 \in \mathcal{S}(V')$.

We now consider an arbitrary set $V' \subseteq \bigcup_{k\in\mathcal{K}} V(P_k)$. If $V' = \emptyset$, then it is easy to see that for all $S \in \mathcal{S}(V')$ we have $\operatorname{conv}(P \setminus S) = P$. Thus, it is sufficient to take $S_Y = \{S\}$, where $S \in \mathcal{S}(V')$. If $V' \neq \emptyset$, let

$$p = \sum_{k\in\mathcal{K}} \sum_{v\in V'\cap V(P_k)} |D_v(P_k)|.$$

For each $S \in \mathcal{S}(V')$ we now define a p-tuple $t(S)$, where for each $k \in \mathcal{K}, v \in V' \cap V(P_k)$, and $r \in D_v(P_k)$, the tuple has a unique entry that equals $x_{vr}(S)$ (Lemma 5). Collection of these p-tuples gives the following set contained in $\mathbb{Z}^p_+$:

$$X = \Big\{ t(S) : S \in \mathcal{S}(V') \Big\}.$$

By Lemma 4, there exists a finite set $Y \subseteq X$ such that for every $x \in X$ there exists $y \in Y$ satisfying $x \geq y$. In particular, there exists a finite set $\mathcal{S}_Y(V') \subseteq \mathcal{S}(V')$ such that for any $S_1 \in \mathcal{S}(V')$ there exists $S_2 \in \mathcal{S}_Y(V')$ satisfying

$$x_{vr}(S_2) \leq x_{vr}(S_1) \text{ for all } k \in \mathcal{K}, v \in V' \cap V(P_k), r \in D_v(P_k). \tag{3}$$

By Lemma 5, the above inequality implies that

$$\lambda_{vr}(S_2) \geq \lambda_{vr}(S_1) \text{ for all } k \in \mathcal{K}, v \in V' \cap V(P_k), r \in D_v(P_k). \tag{4}$$

As both $S_1, S_2 \in \mathcal{S}(V')$, we have $V(P_k) \cap S_2 = V(P_k) \cap S_1$ for all $k \in \mathcal{K}$ and applying Corollary 1 we conclude that $\text{conv}\,(P_k \setminus S_2) \subseteq \text{conv}\,(P_k \setminus S_1)$ for all $k \in \mathcal{K}$. To conclude the proof it suffices to let

$$\mathcal{S}_Y = \bigcup_{V' \subseteq \bigcup_{k\in\mathcal{K}} V(P_k)} \mathcal{S}_Y(V') \qquad \blacksquare$$

Lemma 6. *Let P_k be a rational pointed polyhedron for $k \in \mathcal{K}$ where $\mathcal{K}$ is a finite set and let $P = \bigcup_{k\in\mathcal{K}} P_k$. Then $\text{SC}\,(P, \mathcal{S})$ is finitely generated for any $\mathcal{S} \subseteq \mathcal{S}^*$. More precisely,*

$$\text{SC}\,(P, \mathcal{S}) = \bigcap_{S\in\hat{\mathcal{S}}} \text{conv}\,(P \setminus S)$$

where $\hat{\mathcal{S}} \subset \mathcal{S}$ is a finite set.

Proof. Note that for any $S_2 \in \mathcal{S}$: $\text{conv}\big((\bigcup_{k\in\mathcal{K}} P_k)\setminus S_2\big) = \text{conv}\big(\bigcup_{k\in\mathcal{K}}(P_k \setminus S_2)\big) = \text{conv}\big(\bigcup_{k\in\mathcal{K}} \text{conv}(P_k \setminus S_2)\big)$. Furthermore, by Proposition 1, there is a finite set $\mathcal{S}_Y \subset \mathcal{S}$ such that for each $S_1 \in \mathcal{S}$ there exists $S_2 \in \mathcal{S}_Y$ that satisfies

$$\text{conv}\Big(\bigcup_{k\in\mathcal{K}} \text{conv}(P_k \setminus S_2)\Big) \subseteq \text{conv}\Big(\bigcup_{k\in\mathcal{K}} \text{conv}(P_k \setminus S_1)\Big) = \text{conv}\Big(\bigcup_{k\in\mathcal{K}} P_k \setminus S_1\Big).$$

As $\mathcal{S}_Y$ is finite, to complete the proof, it suffices to observe that

$$\text{SC}(P, \mathcal{S}) = \bigcap_{S\in\mathcal{S}} \text{conv}\Big(\bigcup_{k\in\mathcal{K}} P_k \setminus S\Big) = \bigcap_{S\in\mathcal{S}_Y} \text{conv}\Big(\bigcup_{k\in\mathcal{K}} P_k \setminus S\Big). \qquad \blacksquare$$

We next relax the assumption of pointedness in the previous result; due to space restrictions we only give a sketch of the proof below.

Theorem 1. *Let P_k be a rational polyhedron for $k \in \mathcal{K}$ where $\mathcal{K}$ is a finite set and let $P = \bigcup_{k\in\mathcal{K}} P_k$. Then $\text{SC}\,(P, \mathcal{S})$ is finitely generated for any $\mathcal{S} \subseteq \mathcal{S}^*$.*

Proof. (sketch) We first observe that $P_k = Q_k + L_k$ for each $k \in \mathcal{K}$, where L_k is a rational linear subspace and $Q_k \subseteq L_k^{\perp}$ is a rational pointed polyhedron. For any $S \in \mathcal{S}$ it is possible to show that if $\text{conv}(P_k \setminus S)$ is not equal to P_k then $P_k \setminus S = (Q_k \setminus S') + L_k$ where S' is a split set in $L_k^{\perp}$ obtained as a projection of S onto $L_k^{\perp}$. Based on this observation, we associate an integer p-tuple to any split set to apply Gordan-Dickson Lemma as in the proof of Proposition 1. $\blacksquare$

4 Split Closure of a Union of Mixed-Integer Sets

Consider a mixed-integer set defined by a polyhedron $P^{LP} \in \mathbb{R}^{n+l}$ and the mixed-integer lattice $\mathbb{Z}^n \times \mathbb{R}^l$ where n and l are positive integers:

$$P^I = P^{LP} \cap (\mathbb{Z}^n \times \mathbb{R}^l) \tag{5}$$

An inequality is called a split cut for P^{LP} with respect to the lattice $\mathbb{Z}^n \times \mathbb{R}^l$ if it is valid for $\text{conv}(P^{LP} \setminus S)$ for some $S \in \mathcal{S}^*_{n,l}$ where

$$\mathcal{S}^*_{n,l} = \{S(\pi, \pi_0) \in \mathcal{S}^* \ : \ \pi \in \mathbb{Z}^n \times \{0\}^l\}.$$

The split closure is then defined in the usual way as the intersection of all such split cuts. A straightforward extension of Theorem 1 is the following:

Corollary 2. *Let $P_k \in \mathbb{R}^{n+l}$ be a rational polyhedron for $k \in \mathcal{K}$ where $\mathcal{K}$ is a finite set and let $P = \bigcup_{k\in\mathcal{K}} P_k$. Then $\text{SC}(P, \mathcal{S})$ is finitely generated for any $\mathcal{S} \subseteq \mathcal{S}^*_{n,l}$.*

5 Cross Closure of a Polyhedral Set

In this section, we show that the cross closure of a rational polyhedron is again a polyhedron. We combine the proof technique of Cook, Kannan and Schrijver [8] for showing that the split closure of a polyhedron is polyhedral along with the results we derived in earlier sections based on proof techniques of Anderson, Cornuéjols, Li [1], and Averkov [4]. We need some definitions to discuss the overall techniques used. Lets denote by $\|\cdot\|$ the usual euclidean norm. Define the width of a split set $S(\pi, \pi_0)$ as $w(S(\pi, \pi_0)) = 1/\|\pi\|$ (this is the geometric distance between the parallel hyperplanes bounding the split set). Then $w(S(\pi, \pi_0)) > \eta$ for some $\eta > 0$ implies that $\|\pi\| < 1/\eta$. Therefore, for any fixed $\eta > 0$ and $\pi_0 \in \mathbb{Z}^n$, there are only a finite number of $\pi \in \mathbb{Z}^n$ such that $w(S(\pi, \pi_0)) > \eta$.

Cook, Kannan, Schrijver (roughly) prove their polyhedrality result using the following idea. Assume P is a polyhedron, $\mathcal{L}$ is a finite list of split sets and let $\text{SC}(P, \mathcal{L}) = \cap_{S\in\mathcal{L}}\text{conv}(P \setminus S)$. Suppose that for every face F of P, $\text{SC}(P, \mathcal{L}) \cap F \subseteq \text{SC}(F)$. Then (i) there are only finitely many split sets beyond the ones contained in $\mathcal{L}$ which yield split cuts cutting off points of $\text{SC}(P, \mathcal{L})$ (they show that if $S(\pi, \pi_0)$ is such a split set, then π must have bounded norm). Therefore, (ii) if one assumes (by induction on dimension) that the number of split sets needed to define the split closure of each face of a polyhedron is finite, then so is the number of split sets needed to define the split closure of the polyhedron.

Santanu Dey [12] observed that idea (i) in the Cook, Kannan, Schrijver proof technique can also be used in the case of some disjunctive cuts which generalize split cuts. We apply a modification of idea (i) to cross cuts; namely we show in Lemma 13 that if $\mathcal{L}$ is a finite list of cross disjunctions (represented as a pair of split sets) such that

$$\text{CC}(P, \mathcal{L}) = \cap_{(S_1,S_2)\in\mathcal{L}}\text{conv}(P \setminus (S_1 \cup S_2)) \tag{6}$$

intersected with each face of P is equal to the cross closure of each face, then cross disjunctions (S_1, S_2) where both $w(S_1)$ and $w(S_2)$ are at most some $\eta > 0$ can only yield cross cuts valid for $\mathrm{CC}(P, \mathcal{L})$, and are therefore not needed to define the cross closure of P. We then only need to consider cross disjunctions (S_1, S_2) where one of $w(S_1), w(S_2)$ is greater than η (such cross disjunctions are still infinitely many in number).

We first need a generalization of Lemma 3, property (2.). Let $\mathrm{rec}(P)$ denote the recession cone of P, $\mathrm{aff}(P)$ denote the affine hull of P, and P^I denote the integer hull of P.

Lemma 7. *Let P be a polyhedron in $\mathbb{R}^n$, and let $S_1, S_2 \in \mathcal{S}^*$ be any two split sets. If $\mathrm{conv}(P \setminus (S_1 \cup S_2))$ is nonempty, then its recession cone equals $\mathrm{rec}(P)$.*

Proof. Let $P' = \mathrm{conv}(P \setminus (S_1 \cup S_2))$. As $P' \subseteq P$ and P is closed, we obtain $\mathrm{rec}(P') \subseteq \mathrm{rec}(P)$. To prove the reverse inclusion, let v be a point in P', and let $r \in \mathrm{rec}(P) \neq \emptyset$. Let v_1, v_2 be two points in $P \setminus (S_1 \cup S_2)$ such that v is a convex combination of v_1, v_2 (we will choose $v_1 = v_2 = v$ if $v \notin S_1 \cup S_2$). Consider the half lines $H_1 = \{v_1 + \lambda r : \lambda \geq 0\}$ and $H_2 = \{v_2 + \lambda r : \lambda \geq 0\}$. As $v_1 \notin S_1 \cup S_2$, either H_1 does not intersect $S_1 \cup S_2$, or else $\sup\{\lambda \geq 0 : v_1 + \lambda r \in S_1 \cup S_2\}$ is finite. In either case, the half line $H_1 \subseteq P'$, and similarly $H_2 \subseteq P'$ and therefore $\mathrm{conv}(H_1 \cup H_2) \subseteq P'$. But then the half line $\{v + \lambda r : \lambda \geq 0\} \subseteq P'$ and therefore $r \in \mathrm{rec}(P') \implies \mathrm{rec}(P) \subseteq \mathrm{rec}(P')$. ∎

Now we generalize Lemma 3, property (1.). For a set $A \subseteq \mathbb{R}^n$, let $\overline{\mathrm{conv}}(A)$ denote the topological closure of $\mathrm{conv}(A)$. The following result is a direct consequence of Theorem 3.5 in [13].

Lemma 8 ([13]). *Let $A \subseteq \mathbb{R}^n$ be a nonempty closed set. Then every vertex of $\overline{\mathrm{conv}}(A)$ belongs to A.*

Lemma 9. *Let P be a pointed polyhedron, and let $S_1, S_2 \in \mathcal{S}^*$ be two split sets. Then $\mathrm{conv}\,(P \setminus (S_1 \cup S_2))$ is a polyhedron.*

Proof. The claim trivially holds if $\mathrm{conv}(P \setminus (S_1 \cup S_2))$ is empty. Therefore we assume that $\mathrm{conv}(P \setminus (S_1 \cup S_2))$ is nonempty. Since $\overline{\mathrm{conv}}\,(P \setminus (S_1 \cup S_2))$ is a closed convex set not containing a line, by a standard result in convex analysis (see for example Theorem 18.5 in [15]) we have that $\overline{\mathrm{conv}}\,(P \setminus (S_1 \cup S_2))$ can be written as the Minkowski sum of its set of vertices and its recession cone.

By Lemma 8 we have that the vertices of $\overline{\mathrm{conv}}\,(P \setminus (S_1 \cup S_2))$ belong to $P \setminus (S_1 \cup S_2)$. Thus, since $P \setminus (S_1 \cup S_2)$ is a finite union of polyhedra, we obtain that $\overline{\mathrm{conv}}\,(P \setminus (S_1 \cup S_2))$ has a finite number of vertices.

On the other hand, by Lemma 7 we obtain that

$$\mathrm{rec}(\mathrm{conv}\,(P \setminus (S_1 \cup S_2))) = \mathrm{rec}(\overline{\mathrm{conv}}\,(P \setminus (S_1 \cup S_2))) = \mathrm{rec}(P).$$

Therefore $\overline{\mathrm{conv}}\,(P \setminus (S_1 \cup S_2))$ is a polyhedron. As all its vertices belong to $P \setminus (S_1 \cup S_2) \subseteq \mathrm{conv}\,(P \setminus (S_1 \cup S_2))$, we conclude that $\mathrm{conv}\,(P \setminus (S_1 \cup S_2)) = \overline{\mathrm{conv}}\,(P \setminus (S_1 \cup S_2))$. Therefore, $\mathrm{conv}\,(P \setminus (S_1 \cup S_2))$ is a polyhedron.

∎

Observe that Lemma 7 and Lemma 9 implies that $\mathrm{CC}(P, \mathcal{L})$ is a polyhedron with $\mathrm{rec}(\mathrm{CC}(P, \mathcal{L})) = \mathrm{rec}(P)$ if $\mathrm{CC}(P, \mathcal{L})$ is defined as in (6).

The proof of the following lemma is ommited.

Lemma 10. *Let P be a polyhedron and let F be a face of P. For any set B, $\mathrm{conv}(P \setminus B) \cap F = \mathrm{conv}(F \setminus B)$.*

The next result is essentially contained in Cook, Kannan and Schrijver [8], though our statement and proof are slightly different.

Lemma 11. *Let P and P' be pointed, full-dimensional polyhedra in $\mathbb{R}^n$ with $P \subset P'$. Then there exists a number $r > 0$ such that for any $c \in \mathbb{R}^n$ satisfying (i) $\max\{c^T x : x \in P\} = d < \max\{c^T x : x \in P'\} < \infty$ and (ii) the first maximum is attained at a vertex of P contained in the interior of P', there exists a ball of radius r in P' with each point x in the ball satisfies $c^T x > d$.*

Proof. Let $P' = \{x : a_i^T x \leq b_i \text{ for } i = 1, \ldots, m\}$ where $a_i \in \mathbb{R}^n$ and let $\eta = \max_i\{||a_i||\}$. Let $V = \{v_1, \ldots, v_k\}$ be the set of vertices of P contained in the interior of P', and let $\epsilon = \min_{i,j}\{b_i - a_i^T v_j\} > 0$. Let c satisfy the conditions of the lemma and let $c^T v_j = d$ for some vertex $v_j \in V$. Further, let $\max\{c^T x : x \in P'\} = d' < \infty$ be attained at a vertex v' of P'. By LP duality, there exists multipliers $0 \leq \lambda = (\lambda_1, \ldots, \lambda_m)$ such that $c = \sum_{i=1}^m \lambda a_i$ and $d' = \sum_{i=1}^m \lambda_i b_i$ and $\tau = \sum_{i=1}^m \lambda_i > 0$. Let $(\bar{c}, \bar{d}, \bar{\lambda}) = (c, d', \lambda)/\tau$. Then $\bar{c} = \sum_{i=1}^m \bar{\lambda} a_i$ where $\sum_{i=1}^m \bar{\lambda}_i = 1$ and therefore $||\bar{c}|| \leq \eta$. Further

$$\bar{d} - \bar{c}^T v_j = \sum_{i=1}^m \bar{\lambda}_i (b_i - a_i^T v_j) \geq \epsilon.$$

By definition, $\max\{\bar{c}^T x : x \in P\}$ is attained at v_j, $\max\{\bar{c}^T x : x \in P'\}$ is attained at v', and the distance between the hyperplanes $\bar{c}^T x = \bar{c}^T v_j$ and $\bar{c}^T x = \bar{d}$ is at least $\epsilon/||\bar{c}|| > \epsilon/(\eta+1)$. Therefore, any point z in the ball $B(v', \epsilon/(\eta+1))$ satisfies $\bar{c}^T z > \bar{c}^T v_j$ (and also $c^T z > c^T v_j$). We can find an $r > 0$ such that $B(v, \epsilon/(\eta+1))$ contains a ball of radius r for each vertex v of P'. ∎

In the proof above, we can assume that we construct a fixed set $\mathcal{B}$ of balls, one per each vertex of P', such that one of these balls satisfies the desired property in Lemma 11.

A *strip* in $\mathbb{R}^n$ is the set of points between a pair of parallel hyperplanes (and including the hyperplanes) and the width of a strip is the distance between its bounding hyperplanes. Thus the topological closure of a split set is a strip. If the *minimum width* of a closed, compact, convex set A is defined as the minimum width of a strip containing A, then it is known (Bang [6]) that the sum of widths of a collection of strips containing A must exceed its minimum width. The following statement is a trivial consequence of Bang's result.

Lemma 12. *Let B be a ball of radius $r > 0$, and let S_1, S_2 be split sets such that $B \subseteq (S_1 \cup S_2)$. Then,*

$$w(S_1) + w(S_2) \geq 2r.$$

Lemma 13. *Let P be a pointed, full-dimensional polyhedron. Let $\mathcal{L} \subseteq \mathcal{S}^* \times \mathcal{S}^*$ be a finite list of cross disjunctions. Let $\mathrm{CC}(P, \mathcal{L})$ be defined as in (6). If $\mathrm{CC}(P, \mathcal{L}) \cap F \subseteq \mathrm{CC}(F)$ for all faces F of P, then there exists $\eta > 0$ such that all cross cuts obtained from cross disjunctions $(S_1, S_2) \in \mathcal{S}^* \times \mathcal{S}^*$ with $w(S_1), w(S_2) \leq \eta$ are valid for $\mathrm{CC}(P, \mathcal{L})$.*

Proof. As P is pointed, $\mathrm{CC}(P, \mathcal{L})$ is also pointed as it is contained in P.

Let $r > 0$ be the number given by Lemma 11 when applied to P and $\mathrm{CC}(P, \mathcal{L})$. Let $0 < \eta < r$. Let $c^T x \leq \gamma$ be a cross cut obtained from a cross disjunction (S_1, S_2) with $w(S_1), w(S_2) \leq \eta$. We will prove that $c^T x \leq \gamma$ is valid for $\mathrm{CC}(P, \mathcal{L})$. As $\gamma \geq \max\{c^T x : x \in \mathrm{conv}(P \setminus (S_1 \cup S_2))\} < \infty$ and since by Lemma 7 we have $\mathrm{rec}(\mathrm{CC}(P, \mathcal{L})) = \mathrm{rec}(P) = \mathrm{rec}(\mathrm{conv}(P \setminus (S_1 \cup S_2)))$, we obtain that $d := \max\{c^T x : x \in \mathrm{CC}(P, \mathcal{L})\} < \infty$. We have two cases.

Case 1: The maximum is attained in a face F of P. Then, since

$$\mathrm{CC}(P, \mathcal{L}) \cap F \subseteq \mathrm{CC}(F) \subseteq \{x \in \mathbb{R}^n : c^T x \leq \gamma\},$$

we infer that $d \leq \gamma$. Therefore $c^T x \leq \gamma$ is valid for $\mathrm{CC}(P, \mathcal{L})$.

Case 2: The maximum is attained in a vertex of $\mathrm{CC}(P, \mathcal{L})$ in the interior of P. This implies that $d < \max\{c^T x : x \in P\}$. By Lemma 11, there exists a ball B of radius r in P with all points in the ball satisying $c^T x > d$. Since $w(S_1) + w(S_2) < 2r$, Lemma 12 implies that $B \setminus (S_1 \cup S_2) \neq \emptyset$. Let $\bar{x} \in B \setminus (S_1 \cup S_2)$. Then $\bar{x} \in \mathrm{conv}(P \setminus (S_1 \cup S_2))$ with $c^T \bar{x} > d$. Since $c^T x \leq \gamma$ is valid for $\mathrm{conv}(P \setminus (S_1 \cup S_2))$, it follows that $d < \gamma$. Therefore $c^T x \leq \gamma$ is valid for $\mathrm{CC}(P, \mathcal{L})$. ∎

The discussion after Lemma 11 implies that the ball B in the proof above can be assumed to be a member of $\mathcal{B}$. Further, the proof implies that even if $w(S_1), w(S_2) \geq r$, if $B \setminus (S_1 \cup S_2) \neq \emptyset$, then (assuming $P, \mathcal{L}$ satisfy the conditions of the Lemma) cross cuts from the disjunction (S_1, S_2) are valid for $\mathrm{CC}(P, \mathcal{L})$.

We will need the following basic properties of unimodular matrices (matrices with determinant ± 1). If V is a rational affine subspace of $\mathbb{R}^n$ with dimension $k < n$ such that $V \cap \mathbb{Z}^n \neq \emptyset$, then there is a $n \times n$ integral unimodular matrix U and vector $v \in \mathbb{Z}^n$ such that the one-to-one mapping $\sigma(x) = Ux + v$ maps V to $\mathbb{R}^k \times \{0\}^{n-k}$; also split sets are mapped to split sets. Further, the intersection of any split set S in $\mathbb{R}^n$ with $\mathbb{R}^k \times \{0\}^{n-k}$ is either empty, $\mathbb{R}^k \times \{0\}^{n-k}$ or equals $S' \times \{0\}^{n-k}$ where S' is a split set in $\mathbb{R}^k$.

Lemma 14. *Let P be a rational pointed polyhedron. Then $\mathrm{CC}(P)$ is a polyhedron. More precisely,*

$$\mathrm{CC}(P) = \bigcap_{(S_1, S_2) \in \mathcal{L}} \mathrm{conv}\,(P \setminus (S_1 \cup S_2))$$

where $\mathcal{L} \subset \mathcal{S}^* \times \mathcal{S}^*$ is a finite set.

Proof. We use standard techniques to show that P can be assumed to be full-dimensional. Assume P has dimension $k < n$. As $\mathrm{aff}(P)$ is a rational, affine

subspace of $\mathbb{R}^n$, if $\text{aff}(P) \cap \mathbb{Z}^n = \emptyset$, then it is well-known (see [16, Corollary 4.1a]) that $P \subseteq \text{aff}(P) \subseteq S$ for some split set S. Then

$$\text{CC}(P) \subseteq \text{SC}(P) \subseteq \text{conv}(P \setminus S) = \emptyset = P^I.$$

Therefore we assume $\text{aff}(P) \cap \mathbb{Z}^n \neq \emptyset$. There is a function $\sigma(x)$ (as discussed before the theorem) that maps $\text{aff}(P)$ to a polyhedron in $\mathbb{R}^k \times \{0\}^{n-k}$, i.e., to $P' \times \{0\}^{n-k}$, where P' is a full-dimensional polyhedron. Let $\mathcal{L}'$ be a subset (not necessarily finite) of all cross disjunctions in $\mathbb{R}^k$ which yield the cross closure of P'; Then for each cross disjunction (or split set pair) (S'_1, S'_2) in $\mathcal{L}'$, if we define a cross set (S_1, S_2) in $\mathbb{R}^n$ as $(S'_1 \times \{0\}^{n-k}, S'_2 \times \{0\}^{n-k})$, and then apply the inverse function $\sigma^{-1}(x)$ (of $\sigma(x)$) to (S_1, S_2), we get a list of cross disjunctions in $\mathbb{R}^n$ which yield the cross closure of P. Therefore we can work on P'.

The proof is by induction on $\dim(P)$. The case $\dim(P) = 0$ is straightforward. Now, lets assume that for all polyhedra Q of dimension strictly less than $\dim(P)$, $\text{CC}(Q)$ is defined by a finite number of cross disjunctions. Let F be a face of P. Since $\dim(F) < \dim(P)$, by the induction hypothesis (and the argument in the previous paragraph) we infer that there exists a finite set of cross disjunctions $\mathcal{L}(F)$ in $\mathbb{R}^n$ such that $\text{CC}(F) = \text{CC}(F, \mathcal{L}(F))$.

Define $\mathcal{L} = \cup_{F \text{ is a face of } P} \mathcal{L}(F)$. By the induction hypothesis, $\mathcal{L}$ is a finite list of cross disjunctions. Also, for any face F of P, $\text{CC}(P, \mathcal{L}) \subseteq \text{CC}(P, \mathcal{L}(F))$ and therefore

$$\text{CC}(P, \mathcal{L}) \cap F = \text{CC}(F, \mathcal{L}) \subseteq \text{CC}(F, \mathcal{L}(F)) = \text{CC}(F),$$

where the first equality follows from Lemma 10. Therefore, Lemma 13 implies the existence of a number $\eta > 0$ such that all cross cuts obtained from cross disjunctions (S_1, S_2) with $w(S_1), w(S_2) \leq \eta$ are valid for $P_{\mathcal{L}}$.

This implies that to define the cross closure of P, it suffices to consider cross disjunctions with (S_1, S_2) with either $w(S_1) > \eta$ or $w(S_2) > \eta$. Further, the discussion after Lemma 13 implies that one of S_1, S_2 satisfies the property: $w(S) > \eta$ and $S \cap B \neq \emptyset$ for a ball $B \in \mathcal{B}$ defined after Lemma 11. Define $\mathcal{S}_\eta = \{S \in \mathcal{S}^* : w(S) > \eta \text{ and } S \cap (\cup_{\mathcal{B}} B) \neq \emptyset\}$ and observe that it is a finite set. For $S \in \mathcal{S}_\eta$ the set $P \setminus S$ is a union of two pointed rational polyhedra (possibly empty). Therefore, Theorem 1 implies that $\text{SC}(P \setminus S) = \text{SC}(P \setminus S, \mathcal{S}^*)$ is finitely generated. On the other hand, we have

$$\begin{aligned}
\text{CC}(P) &= \bigcap_{S_1 \in \mathcal{S}^*, S_2 \in \mathcal{S}^*} \text{conv}\left(P \setminus (S_1 \cup S_2)\right) \\
&= \bigcap_{(S_1,S_2) \in \mathcal{L}} \text{conv}\left(P \setminus (S_1 \cup S_2)\right) \ \cap \bigcap_{S_1 \in \mathcal{S}_\eta, S_2 \in \mathcal{S}^*} \text{conv}\left(P \setminus (S_1 \cup S_2)\right) \\
&= \text{CC}(P, \mathcal{L}) \ \cap \bigcap_{S \in \mathcal{S}_\eta} \text{SC}(P \setminus S, \mathcal{S}^*).
\end{aligned}$$

Therefore, we conclude that $\text{CC}(P)$ is finitely generated and, by Lemma 9, is a polyhedron. ∎

Lemma 14 can be extended to general polyhedra by using the same ideas used in the proof of Theorem 1.

Theorem 2. *Let P be a rational polyhedron. Then* $\mathrm{CC}(P)$ *is a polyhedron.*

References

1. Andersen, K., Cornuéjols, G., Li, Y.: Split closure and intersection cuts. Mathematical Programming 102(3), 457–493 (2005)
2. Andersen, K., Louveaux, Q., Weismantel, R.: An analysis of mixed integer linear sets based on lattice point free convex sets. Math. Oper. Res. 35(1), 233–256 (2010)
3. Andersen, K., Louveaux, Q., Weismantel, R., Wolsey, L.A.: Inequalities from Two Rows of a Simplex Tableau. In: Fischetti, M., Williamson, D.P. (eds.) IPCO 2007. LNCS, vol. 4513, pp. 1–15. Springer, Heidelberg (2007)
4. Averkov, G.: On finitely generated closures in the theory of cutting planes. Discrete Optimization 9(1), 209–215 (2012)
5. Averkov, G., Wagner, C., Weismantel, R.: Maximal lattice-free polyhedra: finiteness and an explicit description in dimension three. Mathematics of Operations Research 36(4), 721–742 (2011)
6. Bang, T.: A solution of the plank problem. Proc. American Mathematical Society 2(6), 990–993 (1951)
7. Basu, A., Hildebrand, R., Köppe, M.: The triangle closure is a polyhedron (2011) (manuscript)
8. Cook, W.J., Kannan, R., Schrijver, A.: Chvátal closures for mixed integer programming problems. Math. Program. 47, 155–174 (1990)
9. Dadush, D., Dey, S.S., Vielma, J.P.: The split closure of a strictly convex body. Oper. Res. Lett. 39(2), 121–126 (2011)
10. Dash, S., Dey, S., Günlük, O.: Two dimensional lattice-free cuts and asymmetric disjunctions for mixed-integer polyhedra. Math. Program. 135, 221–254 (2012)
11. Dash, S., Günlük, O., Lodi, A.: MIR closures of polyhedral sets. Math. Program. 121(1), 33–60 (2010)
12. Dey, S.S.: Personal Communication (2010)
13. Klee, V.L.: Extremal structure of convex sets. Archiv der Mathematik 8, 234–240 (1957)
14. Li, Y., Richard, J.P.P.: Cook, Kannan and Schrijver's example revisited. Discrete Optimization 5, 724–734 (2008)
15. Rockafeller, G.T.: Convex analysis. Princeton University Press, New Jersey (1970)
16. Schrijver, A.: Theory of linear and integer programming. John Wiley and Sons, New York (1986)
17. Vielma, J.P.: A constructive characterization of the split closure of a mixed integer linear program. Oper. Res. Lett. 35(1), 29–35 (2007)

Packing Interdiction and Partial Covering Problems

Michael Dinitz[1] and Anupam Gupta[2]

[1] Weizmann Institute of Science
[2] Carnegie Mellon University
{mdinitz,anupamg}@cs.cmu.edu

Abstract. In the *Packing Interdiction* problem we are given a packing LP together with a separate interdiction cost for each LP variable and a global interdiction budget. Our goal is to harm the LP: which variables should we forbid the LP from using (subject to forbidding variables of total interdiction cost at most the budget) in order to minimize the value of the resulting LP? Interdiction problems on graphs (interdicting the maximum flow, the shortest path, the minimum spanning tree, etc.) have been considered before; here we initiate a study of interdicting packing linear programs. Zenklusen showed that matching interdiction, a special case, is NP-hard and gave a 4-approximation for unit edge weights. We obtain an constant-factor approximation to the matching interdiction problem without the unit weight assumption. This is a corollary of our main result, an $O(\log q \cdot \min\{q, \log k\})$-approximation to Packing Interdiction where q is the row-sparsity of the packing LP and k is the column-sparsity.

1 Introduction

In an *interdiction* problem we are asked to play the role of an adversary: e.g., if a player is trying to maximize some function, how can we best restrict the player in order to minimize the value attained? One of the classic examples of this is the *Network Interdiction Problem* (also called *network inhibition*), in which the player is attempting to maximize the s-t flow in some graph G, and we (as the adversary) are trying to destroy part of the graph in order to minimize this maximum s-t flow. Our ability to destroy the graph is limited by a budget constraint: each edge, along with its capacity, has a cost for destroying it, and we are only allowed to destroy edges with a total cost of at most some value $B \geq 0$ (called the budget). This interdiction problem has been widely studied due to the many applications (see e.g. [1,2,3,4]). Obviously, if the cost of the minimum s-t cut (with respect to the destruction costs) is at most B, then we can simply disconnect s from t, but if this is not the case then the problem becomes NP-hard. Moreover, good approximation algorithms for this problem have been elusive. Similarly, a significant amount of work has been done on interdicting shortest paths (removing edges in order to maximize the shortest path) [5,6], interdicting minimum spanning trees [7], and interdicting maximum matchings [8].

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 157–168, 2013.

Our motivation is from the problem of interdicting the maximum matching. Zenklusen [8] defined both edge and vertex versions of this problem, but we will be concerned with the edge version. In this problem, the input is a graph $G = (V, E)$, a weight function $w : E \to \mathbb{R}^+$, a cost function $c : E \to \mathbb{R}^+$, and a budget $B \in \mathbb{R}^+$. The goal is to find a set $R \subseteq E$ with cost $c(R) := \sum_{e \in R} c(e)$ at most B that minimizes the weight of the maximum matching in $G \setminus R$. Zenklusen et al. [9] proved that this problem is NP-complete even when restricted to bipartite graphs with unit edge weights and unit interdiction costs. Subsequently, Zenklusen [8] gave a 4-approximation for the special case when all edge weights are unit (which is also a 2-approximation for the unit-weight bipartite graph case) and also an FPTAS for bounded treewidth graphs. These papers left open the question of giving a constant-factor approximation without the unit-weight assumption. This is a special case of the general problem we study, and indeed our algorithm resolves this question.

Maximum matching is a classic example of a packing problem. If we forget about the underlying graph and phrase the matching interdiction problem as an LP, we get the following problem: given a packing LP (i.e., an LP of the form $\max\{w^\intercal x \mid Ax \leq b, x \geq 0\}$, where A, b, w are all non-negative), in which every column j has an interdiction cost (separate from the weight w_j given to the column by the objective function), find a set of columns of total cost at most B that when removed minimizes the value of the resulting LP. This is the problem of *Packing Interdiction* , and is the focus of this paper. Interestingly, it appears to be one of the first versions of interdiction that is *not* directly about graphs: to the best of our knowledge, the only other is the *matrix interdiction problem* of Kasiviswanathan and Pan [10], in which we are given a matrix and are asked to remove columns in order to minimize the sum over rows of the largest element remaining in the row.

The Packing Interdiction problem is NP-hard, by the fact that bipartite matching interdiction is a special case due to the integrality of its standard LP relaxation, and the results of [9], and hence we consider approximation algorithms for it. Let (k, q)-packing interdiction, or (k, q)-PI for short, denote the Packing Interdiction problem in which the given non-negative matrix $A \in \mathbb{R}^{m \times n}$ has at most k nonzero entries in each row and at most q nonzero entries in each column. So, for example, bipartite matching interdiction is a special case of $(|V|, 2)$-PI, where $|V|$ is the number of nodes in the bipartite graph. Note that $k \leq n$ (where n is the number of variables in the LP) and $q \leq m$ (where m is the number of constraints). Our main result is the following.

Theorem 1. *There is a polynomial time $O(\log q \cdot \min\{q, \log k\})$-approximation algorithm for the (k, q)-Packing Interdiction problem.*

As a corollary, we get an $O(1)$-approximation for matching interdiction without assuming unit weights, since the natural LP relaxation has $q = 2$ and an integrality gap of 2. (See Lemma 1 for a formal proof.)

Corollary 1. *There is a polynomial-time $O(1)$-approximation for matching interdiction*

Packing Interdiction problems turn out to be closely related to the well-studied problems called *partial covering problems*; indeed, there is an algorithm for one if and only if there is an algorithm for the other (see Theorem 3). In a partial covering problem we are given a *covering* LP (i.e., a problem of the form $\min\{w^\intercal x \mid Ax \geq b, x \geq 0\}$, where A, b, w are again all non-negative, together with costs for each row (rather than for each column as in Packing Interdiction), and a budget B. We seek an vector $x \geq 0$ that minimizes the linear objective function $w^\intercal x$, subject to x being feasible for all the constraints except those in a subset of total cost at most B. In other words, rather than our (fractional) solution x being forced to satisfy all constraints as in a typical linear program, we are allowed to choose constraints with total cost at most B and violate them arbitrarily. When the matrix A defining the covering problem has at most k nonzero entries in each row and at most q nonzero entries in each column, we refer to this as the (k, q)-*partial covering* problem, or (k, q)-PC for short. We prove the following theorem about partial covering:

Theorem 2. *There is a polynomial time $O(\log k \cdot \min\{k, \log q\})$-approximation algorithm for the (k, q)-partial covering problem.*

Using the correspondence between Packing Interdiction and partial covering alluded to above, Theorem 1 follows from Theorem 2.

While many specific partial covering problems have been studied, the general partial covering problem we define above appears to be new. The closest work is by Könemann, Parekh, and Segev [11], who define the *generalized partial cover problem* to be the version in which the variables are required to be integral (i.e., even after choosing which rows to remove, they still have to solve an integer programming problem, whereas we have only a linear program which we want to solve fractionally); moreover, they consider the case where A is a $\{0, 1\}$ matrix. Their main result is a general reduction of these integer partial covering problems to certain types of algorithms for the related "prize-collecting" covering problems (where covering constraints may be violated by paying some additive penalty in the objective function). They use this reduction to prove upper bounds for many special cases of integer partial covering, such as the partial vertex cover, and partial set cover problem. Our approach to partial covering will, to a large extent, follow their framework with suitable modifications.

2 Packing Interdiction and Partial Covering

A packing LP consists of a matrix $A \in \mathbb{R}^{n \times m}$, a vector $c \in \mathbb{R}^m$, and a vector $b \in \mathbb{R}^n$, all of which have only nonnegative entries. The packing LP is defined as:

$$\max\{c^\intercal x \mid Ax \leq b, x \in \mathbb{R}^m_{\geq 0}\}$$

A packing LP is called q-*column-sparse* if every column of A has at most q nonzero entries, and k-*row-sparse* if every row of A has at most k nonzero entries. Note that $q \leq m$ and $k \leq n$. We consider the problem of Packing Interdiction (or PI), in which we are given a packing LP and are asked to play the role

of an adversary that is allowed to essentially "forbid" certain variables (which corresponds to setting their c_i multiplier to 0) in an attempt to force the optimum to be small. More formally, an instance of Packing Interdiction is a 5-tuple (c, A, b, r, B) where $c \in \mathbb{R}^m, A \in \mathbb{R}^{n \times m}, b \in \mathbb{R}^n, r \in \mathbb{R}^m$, and $B \in \mathbb{R}^+$ and all entries of c, A, B and r are nonnegative. Given such an instance and a vector $z \in \{0,1\}^m$, define

$$\Phi(z, c, A, b) := \begin{cases} \max & \sum_{i=1}^{m} c_i(1 - z_i)x_i \\ s.t. & Ax \leq b \\ & x_i \geq 0 \qquad \forall i \in [m] \end{cases}$$

to be the optimum value of the packing LP when we interdict the columns with $z_i = 1$. The Packing Interdiction problem on (c, A, b, r, B) is the following integer program:

$$\begin{aligned} \min \quad & \Phi(z, c, A, b) \\ s.t. \quad & \sum_{i=1}^{m} r_i z_i \leq B \\ & z_i \in \{0,1\} \qquad \forall i \in [m] \end{aligned} \tag{MIP_{PI}}$$

Observe that while we want the z variables to be $\{0,1\}$ (to denote which variables are zero to zero), the x variables are allowed to be fractional. When the matrix A is k-row-sparse and q-column-sparse we call this problem (k,q)-Packing Interdiction (or (k,q)-PI). We say that an algorithm is an α-approximation if it always returns a solution z to (MIP_{PI}) that is within α of optimal, i.e., the vector z is feasible for (MIP_{PI}) (satisfies the budget and integrality constraints), and $\Phi(z', c, A, b) \leq \alpha \cdot \Phi(z, c, A, b)$ for any other feasible vector z'.

Our main example, and the initial motivation for this work, is (integer) *matching interdiction*. In this problem we are given a graph $G = (V, E)$, where each edge e has a weight $w_e \geq 0$ and a cost $r_e \geq 0$, and budget B. We, as the interdictor, seek a set $E' \subseteq E$ with $\sum_{e \in E'} r_e \leq B$ that minimizes the maximum weight matching in $G \setminus E'$. We can relax this to the *fractional* matching interdiction problem, where instead of interdicting the maximum weight matching we interdict the maximum weight fractional matching, defined as the optimum solution to the following LP relaxation:

$$\begin{aligned} \max \quad & \sum_{e \in E} w_e x_e \\ s.t. \quad & \sum_{e \in \partial(v)} x_e \leq 1 \; \forall v \in V \\ & x_e \geq 0 \qquad \forall e \in E \end{aligned} \tag{2.1}$$

It is easy to see that fractional matching interdiction is a special case of $(n, 2)$-PI.

Lemma 1. *A ρ-approximation for fractional matching interdiction gives a 2ρ-approximation for the integer matching interdiction problem.*

Proof. Consider the optimal solution E' to integer matching interdiction, and say the weight of the max-weight matching in $G \setminus E'$ is W. It is well-known that dropping the odd-cycle constraints in the LP for non-bipartite matching results in the vertices being half-integral (and hence an integrality gap of at most 2). This means the value of the LP after interdicting E' is at most $2W$, which gives an upper bound on the optimal fractional interdiction solution. Now a ρ-approximation finds a set E'' such that the fractional solution on $G \setminus E''$ is at most $2\rho W$. Since (2.1) is a relaxation, the weight of the max-weight matching in $G \setminus E''$ is also at most $2\rho W$, giving the claimed approximation.

2.1 Partial Covering Problems

A dual problem which will play a crucial role for us when designing an algorithm for (k, q)-PI, is the problem of *Partial Covering*. Given a matrix $A \in \mathbb{R}^{m \times n}$, vectors $c \in \mathbb{R}^m$ and $b \in \mathbb{R}^n$, all having nonnegative entries, the covering LP is:

$$\min\{b^\intercal x \mid Ax \geq c, x \in \mathbb{R}^n_{\geq 0}\}$$

As before, we say that a covering LP is *q-column-sparse* if every column of A has at most q nonzeros and is *k-row-sparse* if every row of A has at most k nonzeros. For $j \in [m]$ and $i \in [n]$, let a_{ji} denote the entry of A in row j and column i. An instance of Partial Covering is a 5-tuple (b, A, c, r, B) in which $b \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}, c \in \mathbb{R}^m, r \in R^m, B \in \mathbb{R}^+$, and all entries of b, A, c, r are nonnegative. Given such an instance and a vector $z \in \{0, 1\}^m$, we define

$$\Psi(z, b, A, c) := \begin{cases} \min & b^\intercal x \\ s.t. & \sum_{i=1}^{n} a_{ji} x_i \geq c_j(1 - z_j) \ \forall j \in [m] \\ & x_i \geq 0 \qquad\qquad \forall i \in [n] \end{cases}$$

to be the value of the covering LP we get when we make the constraints j with $x_j = 1$ trivial by setting their right side to 0. Then the Partial Covering problem is the problem of computing

$$\begin{array}{lll} \min & \Psi(z, b, A, c) & \\ s.t. & \sum_{i=1}^{m} r_i z_i \leq B & (MIP_{PC}) \\ & z_i \in \{0, 1\} \qquad \forall i \in [m] & \end{array}$$

Analogously to Packing Interdiction, we say that an algorithm is an α-approximation to partial covering if on any instance it returns a vector z such that the value of (MIP_{PC}) is at most α times the optimal value. We let (k, q)-Partial Covering be partial covering restricted to covering LPs that are k-row-sparse and q-column-sparse.

2.2 Relating Packing Interdiction and Partial Covering

Theorem 3. *There is a polynomial time α-approximation algorithm for (q,k)-Packing Interdiction if and only if there is a polynomial time α-approximation algorithm for (k,q)-Partial Covering.*

Proof. Suppose that we have an α-approximation algorithm for (k,q)-partial covering. Let (c,A,b,r,B) be an instance of (q,k)-Packing Interdiction. Then $(b,A^\intercal,c,r,B)$ is an instance of (k,q)-Partial Covering. Note that for a fixed $z \in \{0,1\}^m$, linear programming strong duality implies that $\Phi(z,c,A,b) = \Psi(z,b,A^\intercal,c)$. Let $z^* \in \{0,1\}^m$ be the optimal solution to the Packing Interdiction problem, and let $\widehat{z} \in \{0,1\}^m$ be the solution to the partial covering problem computed by the algorithm. Then $\Phi(\widehat{z},c,A,b) = \Psi(\widehat{z},b,A^\intercal,c) \leq \alpha \cdot \Psi(z^*,b,A^\intercal,c) = \alpha \cdot \Phi(z^*,c,A,b)$, where the first and last step are by strong duality and the middle inequality is by the definition of an α-approximation algorithm. Thus $\widehat{z}$ is an α-approximation to the Packing Interdiction instance. The proof of the other direction is entirely symmetric.

2.3 Partial Fractional Set Cover

A useful case of (k,q)-Partial Covering is (k,q)-*Partial Fractional Set Cover* (or (k,q)-PFSC), in which the matrix A has all entries from $\{0,1\}$, and moreover where the covering requirement $c_i = 1$ for all rows $i \in [m]$. As the name suggests, we can interpret (k,q)-PFSC in terms of set systems as follows: The universe of elements U is $[m]$, the set of rows in A. For each column $j \in [n]$ of A we have a set $S_j := \{i \in [m] \mid a_{ij} = 1\}$ corresponding to the rows which have a 1 in the j^{th} column. The k-row-sparsity of A means that every element is in at most k sets, and the q-column-sparsity means that every set contains at most q elements. Then (k,q)-PFSC corresponds to choosing a set of elements E with $\sum_{i \in E} r_i \leq B$ to ignore the covering requirement for, and then constructing a minimum-cost *fractional* set cover for the remaining elements.

The version of this problem in which the x variables are forced to be integral is the *partial set cover* problem. For this problem, algorithms are known that achieve an approximation ratio of $O(\min\{k, H(q)\})$, where $H(q) = \sum_{i=1}^{q} 1/i = \Theta(\log q)$ is the harmonic number. (See, e.g., [11], which improves on [12,13,14,15].)

3 Algorithms for Partial Covering

Thanks to Theorem 3 we know that an algorithm for partial covering implies one for packing interdiction (i.e., Theorem 2 implies Theorem 1). We now prove Theorem 2 by designing an approximation algorithm for (k,q)-Partial Covering. The approach is to first reduce the general (k,q)-Partial Covering problem to the (k,q)-Partial Fractional Set Cover problem with a loss of $O(\log k)$ (i.e., with this logarithmic loss we can assume that A, c are not just non-negative, but have entries in $\{0,1\}$). We finally give an approximation for (k,q)-PFSC.

3.1 Reduction to Partial Fractional Set Cover

Let $\mathcal{I} = (b, A, c, r, B)$ be an instance of (k, q)-Partial Covering; the associated optimization problem is given by the following mixed-integer program obtained by expanding out (MIP_{PC}):

$$\begin{array}{lll} \min & b^\intercal x & \\ s.t. & \sum_{i=1}^{n} a_{ji} x_i \geq c_j(1 - z_j) & \forall j \in [m] \\ & \sum_{j=1}^{m} r_j z_j \leq B & \\ & x_i \geq 0 & \forall i \in [n] \\ & z_j \in \{0, 1\} & \forall j \in [m] \end{array} \tag{3.2}$$

We modify this mixed-integer program to be an instance of partial fractional set. If $a_{ji} \neq 0$, let $t(i, j) = \lceil \log_2(c_j/a_{ji}) \rceil$, and for each $j \in [m]$ let $S(j) = \{i \in [n] : a_{ji} \neq 0\}$. For every $i \in [n]$, we replace the variable x_i with a collection of variables $\{x_i^t\}_{t \in \mathbb{Z}}$, where the "intended" interpretation of $x_i^t = 1$ is that $x_i \geq 2^t$ as in the following mixed-integer program:

$$\begin{array}{lll} \min & \sum_{i=1}^{n} \sum_{t \in \mathbb{Z}} 2^t b_i x_i^t & \\ s.t. & \sum_{i \in S(j)} x_i^{t(i,j)} \geq 1 - z_j & \forall j \in [m] \\ & \sum_{j=1}^{m} r_j z_j \leq B & \\ & x_i^t \geq 0 & \forall i \in [n], t \in \mathbb{Z} \\ & z_j \in \{0, 1\} & \forall j \in [m] \end{array} \tag{3.3}$$

Although as stated there are an infinite number of variables, we only need the variables x_i^t where $t = t(i, j)$ for some j, and hence the number of x_i^t variables is at most nm. We will call this instance $\mathcal{I}' = (b', A', 1, r, B)$, where r and B are unchanged from the original instance.

Lemma 2. *$\Psi(z, b, A, c) \leq \Psi(z, b', A', 1) \leq O(\log k) \cdot \Psi(z, b, A, c)$ for any vector $z \in \{0, 1\}^m$.*

Proof. To show that $\Psi(z, b, A, c) \leq \Psi(z, b', A', 1)$, take $\{x_i^t\}_{i \in [n], t \in \mathbb{Z}}$ to be a solution to (3.3) for integral vector z, and construct a solution $\{x_i\}_{i \in [n]}$ to (3.2) on the same vector z as follows: define

$$x_i := \max_{j : i \in S(j)} 2^{t(i,j)} \, x_i^{t(i,j)}.$$

Every constraint $j \in [m]$ with $z_j = 1$ is trivially satisfied, so consider some j with $z_j = 0$. For such a constraint j,

$$\sum_{i=1}^{n} a_{ji} x_i = \sum_{i \in S(j)} a_{ji} \cdot \max_{j' : i \in S(j')} 2^{t(i,j')} \, x_i^{t(i,j')} \geq \sum_{i \in S(j)} a_{ji} \, 2^{t(i,j)} \, x_i^{t(i,j)}$$

$$\geq \sum_{i \in S(j)} a_{ji} \left(c_j/a_{ji}\right) x_i^{t(i,j)} = c_j \sum_{i \in S(j)} x_i^{t(i,j)} \geq c_j(1 - z_j),$$

where we use the definition of $2^{t(i,j)} \geq c_j/a_{ji}$, and that $x_i^{t(i,j)}$ is a feasible solution for (MIP_{PC}) on z. Thus $\{x_i\}_{i \in [n]}$ is a valid solution to (3.2) on z. Its cost is

$$\sum_{i=1}^{n} b_i x_i = \sum_{i=1}^{n} b_i \cdot \max_{j \in [m]} 2^{t(i,j)} x_i^{t(i,j)} \leq \sum_{i=1}^{n} b_i \sum_{t \in \mathbb{Z}} 2^t x_i^t,$$

which is exactly the value of $\{x_i^t\}_{i \in [n], t \in \mathcal{Z}}$ in (3.3) on z. Thus $\Psi(z, b, A, c) \leq \Psi(z, b', A', 1)$ for each binary vector z.

To prove that the second inequality of the lemma, consider $\{x_i\}_{i \in [n]}$, a solution to (3.2) on z, and construct a solution for (3.3) on z of cost at most $O(\log k) \cdot \sum_{i=1}^{n} b_i x_i$ as follows. For each $i \in [n]$, set $x_i^t = 1$ for all integers $t \leq \log_2 x_i$. For integers t that satisfy $\log_2 x_i < t \leq \log_2(4kx_i)$, set $x_i^t = x_i/2^t$, and for larger integers t, set $x_i^t = 0$.

Define $I_j = \{i \in [n] : a_{ji} x_i \geq c_j(1 - z_j)/2k\} \subseteq S(j)$. Since there are at most k values of i for which $a_{ji} \neq 0$, the value $\sum_{i \notin I_j} a_{ji} x_i < k \cdot c_j(1 - z_j)/2k = c_j(1 - z_j)/2$; hence $\sum_{i \in I_j} a_{ji} x_i \geq \frac{1}{2} c_j(1 - z_j)$.

If $z_j = 0$, we claim that for each $i \in I_j$, either $x_i^{t(i,j)} \in \{1, x_i/2^{t(i,j)}\}$. In other words, we need to show that $t(i,j) \leq \log_2(4kx_i)$ and hence is not set to zero. Indeed, by definition, $t(i,j) \leq \log_2(c_j/a_{ji}) + 1 = \log_2(2c_j/a_{ji})$. Moreover, by the definition of I_j, if $i \in I_j$ and $z_j = 0$ we know that $c_j/a_{ji} \leq 2k\, x_i$. Combining the two inequalities, $t(i,j) \leq \log_2(4k\, x_i)$ as claimed.

Consider some constraint $j \in [m]$ for (3.3); we will show that the solution $4x_i^t$ satisfies this constraint. If $z_j = 1$ then the constraint is trivially satisfied. Else $z_j = 0$; then for each $i \in I_j$, we have $x_i^{t(i,j)} \in \{1, x_i/2^{t(i,j)}\}$. If any of these $x_i^{t(i,j)}$ values are set to 1, the constraint j in (3.3) is satisfied by that variable alone. If not, $x_i^{t(i,j)} = x_i/2^{t(i,j)}$ for all $i \in I_j$, and

$$\begin{aligned} \sum_{i \in S(j)} x_i^{t(i,j)} &\geq \sum_{i \in I_j} x_i^{t(i,j)} = \sum_{i \in I_j} x_i/2^{t(i,j)} \geq \sum_{i \in I_j} (a_{ji}/2c_j) x_i \geq \frac{1}{2c_j} \frac{c_j}{2} (1 - z_j) \\ &= \frac{1}{4}(1 - z_j). \end{aligned}$$

Now by multiplying all of the x_i^t variables by 4 we have a valid solution to (3.3) on z. The cost of this solution is at most

$$4 \sum_{i=1}^{n} b_i \sum_{t \in \mathbb{Z}} 2^t x_i^t \leq 4 \sum_{i=1}^{n} b_i \left(\sum_{t \leq \log x_i} 2^t \cdot 1 + \sum_{t = \log x_i}^{\log x_i + \log(4k)} 2^t \cdot \frac{x_i}{2^t} \right) \leq O(\log k) \sum_{i=1}^{n} b_i x_i,$$

as desired.

Lemma 3. *An α-approximation algorithm for (k, q)-Partial Fractional Set Cover gives an $O(\alpha \log k)$-approximation for (k, q)-Partial Covering.*

Proof. The above reduction to Partial Fractional Set Cover loses a factor of $O(\log k)$ in the approximation due to Lemma 6, so it remains to show that the instance $(b', A', 1, r, B)$ is in fact a (k, q)-PFSC instance, i.e., the row and column sparsities of A' are the same as in A.

For each constraint $j \in [m]$, the number of non-zeroes in the partial covering constraint of (3.3) for j equals the number of nonzeros in the partial covering constraint of (3.2) for j: both have one nonzero for each $i \in S(j)$. Thus the row sparsity of our PFSC instance is at most k. Similarly, for $i \in [n]$ and value $t \in \mathbb{Z}$, the variable x_i^t has a nonzero coefficient in (3.3) only for constraints j in which $i \in S(j)$ and $t = t(i, j)$, which is at most the number of constraints j in which $i \in S(j)$. This is the number of constraints j for which $a_{ji} \neq 0$, which is at most q, and the column sparsity of our PFSC instance is at most q, as desired.

3.2 Approximating Partial Fractional Set Cover

We now give approximation algorithms for (k, q)-Partial Fractional Set Cover. Könemann, Parekh, and Segev [11] give good algorithms for the partial set cover problem, i.e., the variant in which the x variables are also required to be integral. We adapt their framework to our setting of partial *fractional* set cover, giving the desired approximation for (k, q)-PFSC, and thus for (k, q)-Partial Cover and (q, k)-Packing Interdiction.

Prize-Collecting Covering Problems. *Prize-collecting fractional set cover* can be interpreted as the Lagrangian relaxation of partial fractional set cover, and is defined thus: given a collection of sets $\mathcal{S}$ over a universe of elements U, cost function $c : \mathcal{S} \to \mathbb{R}$, and for each element $e \in U$ there is a penalty $p(e)$, every element needs to either be covered by a set or else we pay the penalty for that element, and the goal is to minimize the total cost. We are allowed to cover an element fractionally, i.e., by fractionally buying sets which in total cover the element; however, the decision of whether to cover the set or pay the penalty for it is an integral decision. This is formalized as the following mixed integer program.

$$\begin{aligned} \min \sum_{S \in \mathcal{S}} c(S) x_S + \sum_{e \in U} p(e) z_e & \\ s.t. \sum_{S \ni e} x_S + z_e \geq 1 & \qquad \forall e \in U \\ x_S \geq 0 & \qquad \forall S \in \mathcal{S} \\ z_e \in \{0, 1\} & \qquad \forall e \in U \end{aligned} \tag{3.4}$$

In prize-collecting set cover, we change the requirements that $x_S \geq 0$ to $x_S \in \{0, 1\}$. A *ρ-Lagrangian multiplier preserving (ρ-LMP) algorithm* for prize-collecting (integral) set cover, as defined by [11], is one which on any instance I of prize-collecting (integral) set cover returns a solution with $C + \rho \cdot \Pi \leq \rho \cdot OPT(I)$, where C is the cost of the sets chosen and Π is the sum of penalties of all uncovered elements. The modification for our context is natural: an algorithm is

ρ-LMP for prize-collecting *fractional* set cover if it always returns a solution to (3.4) with $C + \rho \cdot \Pi \leq \rho \cdot OPT_{MIP}$, where as before Π is the sum of the penalties of uncovered elements (elements where $z_e = 1$), C is the total cost of the fractional covering (i.e. $\sum_{S \in \mathcal{S}} c(S)x_S$), and OPT_{MIP} is value of the optimal solution to (3.4).

Könemann et al. [11, Theorem 1] show that a ρ-LMP algorithm for prize-collecting (integral) set cover gives a $(\frac{4}{3}+\epsilon)\rho$-approximation for partial set cover, for any constant $\epsilon > 0$. Theorem 4 below generalizes this to the fractional version in a natural way, but we need an additional property: even for the fractional prize-collecting problem, the algorithm returns a solution where both x, z variables are integral. (We defer the simple proof to the full version of the paper; the crucial idea is that for any PFSC instance I, if we ignore all sets of cost more than $2OPT_{LP}$, the optimal value of the resulting PFSC instance remains at most $2OPT_{LP}$. And once every set has small cost, one can follow the earlier analysis.)

Theorem 4. *If there is a ρ-LMP algorithm for the k-row-sparse, q-column-sparse prize-collecting fractional set cover problem which returns an integral solution, then there is an $O(\rho)$-approximation algorithm for (k, q)-PFSC.*

Using this theorem, it suffices to give algorithms for the prize-collecting fractional set cover problem. Könemann et al. [11, Section 4.1] show that a natural variant of the greedy algorithm is $H(q)$-LMP for prize-collecting (integer) set cover, where $H(q)$ is the q-th harmonic number. We show that their algorithm is, in fact, $H(q)$-LMP for prize-collecting *fractional* set cover (despite returning an integral solution) by analyzing their algorithm relative to an LP rather than relative to the optimal integer solution. The algorithm works as follows: given an instance of prize-collecting partial fractional set cover $(U, \mathcal{S}, c, p)$, we create a new collection of sets $\mathcal{S}'$ where, in addition to the sets in $\mathcal{S}$, we have a singleton set $\{e\}$ for every element $e \in U$. For every set $S \in \mathcal{S}$ we set $c'(S) = c(S)$, and for each element $e \in U$ we set $c'(\{e\}) = H(q) \cdot p(e)$. We now run the greedy algorithm on this collection of sets, where we iteratively buy the set in $\mathcal{S}'$ that maximizes the number of currently uncovered elements divided by the cost c' of the set.

Lemma 4. *This greedy algorithm is $H(q)$-LMP for prize-collecting fractional set cover.*

Proof. We prove this by a standard dual-fitting argument. Relaxing the integrality constraints on the z variables of (3.4) and taking the dual, we get:

$$\begin{aligned} \max & \sum_{e \in U} y_e \\ s.t. \quad & y_e \leq p(e) \qquad \forall e \in U \\ & \sum_{e \in S} y_e \leq c(S) \quad \forall S \in \mathcal{S} \\ & y_e \geq 0 \qquad \forall e \in U \end{aligned} \tag{3.5}$$

Let OPT denote the value of an optimal solution for (3.4). Let $\mathcal{S}_{gr}$ denote the sets in $\mathcal{S}$ bought by the greedy algorithm, and let $\mathcal{P}_{gr}$ denote the singleton sets it bought. Suppose at some point the greedy algorithm has covered elements in $Z \subset U$, and then picks a set A containing an element e. Then we know that $\frac{c'(A)}{|A \setminus Z|} \leq \frac{c'(B)}{|B \setminus Z|}$ for every set $B \in \mathcal{S}'$. Defining price(e) to be $\frac{c'(A)}{|A \setminus Z|}$, and recalling the definition of $c'(\cdot)$, we get

$$\sum_{e \in U} \text{price}(e) = \sum_{S \in \mathcal{S}_{gr}} c(S) + \sum_{\{e\} \in \mathcal{P}_{gr}} H(q)p(e) = \sum_{S \in \mathcal{S}_{gr}} c(S) + H(q) \sum_{\{e\} \in \mathcal{P}_{gr}} p(e).$$

To prove the greedy algorithm is $H(q)$-LMP, it suffices to show $\sum_{e \in U} \text{price}(e) \leq H(q) \cdot OPT$. Let LP denote the optimal fractional solution to (3.4) where the z variables are no longer constrained to be integral. Then $LP \leq OPT$, and by duality any solution to (3.5) is a lower bound on LP. We claim that $y_e = \text{price}(e)/H(q)$ is a valid dual solution; hence $\sum_{e \in U} \text{price}(e) = H(q) \sum_{e \in U} y_e \leq H(q) \times LP \leq H(q) \times OPT$, as required.

Finally, we show y_e is a valid solution to (3.5). Since at any point we could choose the singleton set $\{e\}$ to cover element e, we get $\text{price}(e) \leq c'(\{e\})/1 = H(q)p(e)$, and thus $y_e \leq p(e)$ for every element $e \in U$. Now let S be an arbitrary element of $\mathcal{S}$, and order the elements of $S = \{x_1, x_2, \cdots, x_{|S|}\}$ by the time that they are covered. Then since S could have been picked to cover x_i, we know that $\text{price}(x_i) \leq \frac{c'(S)}{|S|-i+1} = \frac{c(S)}{|S|-i+1}$. Thus $\sum_{e \in S} y_e = (1/H(q)) \sum_{i=1}^{n} \text{price}(x_i) \leq (1/H(q)) \cdot c(S) \cdot H(|S|) \leq c(S)$, and thus our choice of y variables form a valid dual solution.

A different approximation ratio for prize-collecting fractional set cover is in terms of k, i.e., the maximum number of sets that any element is contained in (a.k.a. its *frequency*). Könemann et al. [11, Lemma 15] showed that the primal-dual algorithm of Bar-Yehuda and Even [16] can be modified to give the following result.

Lemma 5. *There is a k-LMP algorithm for the prize-collecting fractional set cover problem.*

We can now combine these ingredients into an algorithm for (k, q)-PFSC.

Lemma 6. *There is an $O(\min\{k, H(q)\})$-approximation algorithm for (k, q)-PFSC.*

Proof. Lemmas 4 and 5 give algorithms that always return integral solutions. Thus combined with Theorem 4 they give the lemma.

3.3 Putting It Together

Having assembled all the necessary components, we can now state our main results. (Observe that all the above reductions run in polynomial time.) Combining Lemmas 3 and 6, we get.

Theorem 5. *There is a polynomial time $O(\log k \cdot \min\{k, \log q\})$-approximation algorithm for (k, q)-Partial Covering.*

Now combining Theorem 5 with Theorem 3, we get

Theorem 6. *There is a polynomial time $O(\log q \cdot \min\{q, \log k\})$-approximation algorithm for (k, q)-Packing Interdiction.*

Corollary 2. *There is a polynomial time $O(1)$-approximation algorithm for the Matching Interdiction problem.*

Proof. As mentioned in Section 2, matching interdiction is a special case of $(n, 2)$-Packing Interdiction, and using $q = 2$ in Theorem 6 gives us an $O(1)$-approximation for fractional matching interdiction. By Lemma 1, we lose another factor of 2 in going to integer matching interdiction.

References

1. Phillips, C.A.: The network inhibition problem. In: Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, STOC 1993, pp. 776–785 (1993)
2. Burch, C., Carr, R., Krumke, S., Marathe, M., Phillips, C., Sundberg, E.: A decomposition-based pseudoapproximation algorithm for network flow inhibition. In: Network Interdiction and Stochastic Integer Programming, pp. 51–68 (2003)
3. Wood, R.: Deterministic network interdiction. Mathematical and Computer Modelling 17, 1–18 (1993)
4. Zenklusen, R.: Network flow interdiction on planar graphs. Discrete Appl. Math. 158, 1441–1455 (2010)
5. Fulkerson, D.R., Harding, G.C.: Maximizing Minimum Source-Sink Path Subject To A Budget Constraint. Mathematical Programming 13, 116–118 (1977)
6. Israeli, E., Wood, R.K.: Shortest-path network interdiction. Networks 40, 2002 (2002)
7. Frederickson, G.N., Solis-Oba, R.: Increasing the weight of minimum spanning trees. In: SODA 1996, pp. 539–546 (1996)
8. Zenklusen, R.: Matching interdiction. Discrete Appl. Math. 158, 1676–1690 (2010)
9. Zenklusen, R., Ries, B., Picouleau, C., de Werra, D., Costa, M.C., Bentz, C.: Blockers and transversals. Discrete Mathematics 309, 4306–4314 (2009)
10. Kasiviswanathan, S.P., Pan, F.: Matrix Interdiction Problem. In: Lodi, A., Milano, M., Toth, P. (eds.) CPAIOR 2010. LNCS, vol. 6140, pp. 219–231. Springer, Heidelberg (2010)
11. Könemann, J., Parekh, O., Segev, D.: A unified approach to approximating partial covering problems. Algorithmica 59, 489–509 (2011)
12. Kearns, M.J.: The computational complexity of machine learning. PhD thesis, Harvard University, Cambridge, MA, USA (1990)
13. Slavík, P.: Improved performance of the greedy algorithm for partial cover. Inf. Process. Lett. 64, 251–254 (1997)
14. Bar-Yehuda, R.: Using homogeneous weights for approximating the partial cover problem. J. Algorithms 39, 137–144 (2001)
15. Fujito, T.: On approximation of the submodular set cover problem. Oper. Res. Lett. 25, 169–174 (1999)
16. Bar-Yehuda, R., Even, S.: A linear-time approximation algorithm for the weighted vertex cover problem. Journal of Algorithms 2, 198–203 (1981)

On Valid Inequalities for Quadratic Programming with Continuous Variables and Binary Indicators

Hongbo Dong and Jeff Linderoth

Wisconsin Institutes for Discovery
University of Wisconsin-Madison, USA
{hdong6,linderoth}@wisc.edu

Abstract. In this paper we study valid inequalities for a set that involves a continuous vector variable $x \in [0,1]^n$, its associated quadratic form xx^T, and binary indicators on whether or not $x > 0$. This structure appears when deriving strong relaxations for mixed integer quadratic programs (MIQPs). Valid inequalities for this set can be obtained by lifting inequalities for a related set without binary variables (**QPB**), that was studied by Burer and Letchford. After closing a theoretical gap about **QPB**, we characterize the strength of different classes of lifted **QPB** inequalities. We show that one class, *lifted-posdiag-QPB inequalities*, capture no new information from the binary indicators. However, we demonstrate the importance of the other class, called *lifted-concave-QPB inequalities*, in two ways. First, all lifted-concave-QPB inequalities define the relevant convex hull for the case of *convex* quadratic programming with indicators. Second, we show that all *perspective constraints* are a special case of lifted-concave-QPB inequalities, and we further show that adding the perspective constraints to a semidefinite programming relaxation of convex quadratic programs with binary indicators results in a problem whose bound is equivalent to the recent optimal diagonal splitting approach of Zheng *et al.*. Finally, we show the separation problem for lifted-concave-QPB inequalities is tractable if the number of binary variables involved in the inequality is small. Our study points out a direction to generalize perspective cuts to deal with non-separable nonconvex quadratic functions with indicators in global optimization. Several interesting questions arise from our results, which we detail in our concluding section.

Keywords: Mixed integer quadratic programming, Semidefinite programming, Valid inequalities, Perspective reformulation.

1 Introduction

Our primary goal in this work is to solve Mixed Integer Quadratic Programming (MIQP) problems with indicator variables of the form

$$\min_{x\in\mathbb{R}^n, z\in\{0,1\}^n} \{q^T x + c^T z + x^T Q x \mid Ax + Bz \le b,\ 0 \le x_i \le u_i z_i\ \forall i = 1,\dots n\}. \quad (1)$$

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 169–180, 2013.

In (1), the binary variable z_i is used to indicate the positivity of its associated continuous variable $x_i, \forall i = 1, \ldots, n$. Related problems of this type arise in many applications, including portfolio selection [4], sparse least-squares [21], optimal control [19], and unit-commitment for power generation [15]. The optimization problem (1) can be very difficult to solve to optimality. Computational experience presented in [3] shows that for problems of size $n = 100$, a branch-and-bound algorithm typically requires more than 10^6 nodes to solve the problem to optimality.

A standard technique for solving (1) is to linearize the objective by introducing a new variable for each product of variables $x_i x_j$, arranging these new variables into a matrix variable X. Problem (1) can then be written as

$$\min_{(x,z,X)\in T} \{q^T x + c^T z + Q \bullet X\}, \tag{2}$$

where

$$T := \left\{ (x, z, X) \in \mathbb{R}^{2n+\frac{n(n+1)}{2}} \;\middle|\; \begin{array}{l} z \in \{0,1\}^n,\ X = xx^T,\ Ax + Bz \le b \\ 0 \le x_i \le u_i z_i,\ i = 1, ..., n \end{array} \right\}.$$

All matrices considered in this paper are symmetric, so they can be represented as a vector in a linear space of dimension $\frac{n(n+1)}{2}$ by stacking columns of upper triangular part of the matrix. Given two $n \times n$ symmetric matrices X and Y, their inner product is defined as $X \bullet Y = \sum_{i=1}^{n} X_{ii} Y_{ii} + 2 \sum_{i<j} X_{ij} Y_{ij}$.

To solve Problem (2), it suffices to optimize the objective over $\mathbf{conv}(T)$, so it is natural to study T and closely-related sets. In this paper, we primarily study valid inequalities for the following set and its convex hull:

$$S := \left\{ (x, z, X) \in \mathbb{R}^{2n+\frac{n(n+1)}{2}},\ \begin{array}{l} x \in [0,1]^n, z \in \{0,1\}^n, \\ X = xx^T, x_i \le z_i, i = 1, ..., n \end{array} \right\}.$$

In S, the general bounds on the continuous variables in T have changed to $x \in [0,1]^n$. This change results in no loss of generality. However, the set S does not have the linear constraints $Ax + Bz \le b$ in the definition of T.

By moving the nonlinearity in (1) into the constraints, many of the results we obtain can be directly applied to create strong convex relaxations of problems that additionally have quadratic constraints and indicator variables. These problem arise in applications such as product pooling with network design [12,23] and digital filter design [25].

When the quadratic functions are convex, a more natural relaxation to study is the following "larger" set,

$$S^{\succeq} := \left\{ (x, z, X) \in \mathbb{R}^{2n+\frac{n(n+1)}{2}},\ \begin{array}{l} x \in [0,1]^n, z \in \{0,1\}^n, \\ X \succeq xx^T, x_i \le z_i, i = 1, ..., n \end{array} \right\},$$

where the notation $X \succeq xx^T$ means that the matrix $X - xx^T$ is positive semidefinite.

The remainder of the extended abstract is organized into five sections. Section 2, describes basic properties of the set S. The relationship between S, the

Boolean Quadric Polytope **BQP** [22], and the box-constrained QP set **QPB** [10] is shown, and we slightly strengthen an earlier result known about valid inequalities for **QPB**. We next discuss valid inequalities of S obtained by lifting certain inequalities for **QPB**. The inequalities are divided into two classes, called *lifted-posdiag-QPB* inequalities, and *lifted-concave-QPB* inequalities. Section 3 shows the negative results that *lifted-posdiag-QPB* inequalities contribute essentially no additional strength to the continuous relaxation. In Section 4, we establish the importance of lifted-concave-QPB inequalities for defining strong relaxations of S. We show that the "simplest" class of lifted-concave-QPB inequalities already contains all perspective cuts [14]. As a by-product, for convex quadratic programs with binary indicators, we propose a semidefinite programming (SDP) relaxation that is no worse than the relaxation obtained by *any* diagonal splitting and perspective reformulation scheme [16]. Further, the corresponding dual SDP provides the optimal diagonal splitting. A similar (but slightly weaker) result was previously obtained in [26]. In Section 4, we also show that every valid linear inequality for $\mathbf{conv}(S^{\succeq})$ is a lifted-concave-QPB inequality. Finally, in Section 5, we provide a tractability result on the separation of lifted-concave-QPB inequalities, establishing that the inequalities can be separated (in the weak sense) in time that is polynomial in n when the binary variables simultaneously lifted is bounded. Section 5 also contains an example of size $n = 3$ where the relaxation with lifted-concave-QPB inequalities dominates the doubly-nonnegative relaxation of [8]. We conclude in Section 6 with some natural directions for research that are motivated by this work.

2 Basic Properties

Proposition 1 establishes three fundamental properties of $\mathbf{conv}(S)$ and $\mathbf{conv}(S^{\succeq})$.

Proposition 1.

- *Both* $\mathbf{conv}(S)$ *and* $\mathbf{conv}(S^{\succeq})$ *are full-dimensional;*
- *The set of extreme points for* $\mathbf{conv}(S)$ *is* S*;*
- $\mathbf{conv}(S^{\succeq}) = \mathbf{conv}(S) + \left\{(0,0,X) \in \mathbb{R}^{2n+\frac{n(n+1)}{2}}, X \succeq 0\right\}$.

Proof. The straightforward proof is given in our extended version [13].

By projecting away z from $\mathbf{conv}(S)$, we obtain the set **QPB** studied in [10],

$$\mathbf{proj}_{(x,X)}\left(\mathbf{conv}(S)\right) = \mathbf{QPB} = \mathbf{conv}\{(x,X) \in \mathbb{R}^{n+\frac{n(n+1)}{2}} : x \in [0,1]^n, X_{ij} = x_i x_j, 1 \le i \le j \le n\}.$$

Furthermore, as proved by [10], projecting away the diagonal entries of X in **QPB** yields the well-known *Boolean Quadric Polytope* (**BQP**) [22]:

$$\mathbf{proj}_{(x,\mathbf{ADiag}(X))}\left(\mathbf{QPB}\right) = \mathbf{BQP} = \mathbf{conv}\{(x,y) \in \mathbb{R}^{n+\frac{n(n-1)}{2}} : x \in \{0,1\}^n, y_{ij} = x_i x_j, 1 \le i < j \le n\},$$

where $\mathbf{ADiag}(X)$ denotes a vector of dimension $n(n-1)/2$ obtained by stacking entries above (but not including) the diagonal of X. These two observations reveal the set $\mathbf{conv}(S)$ to contain interesting interactions between continuous and binary variables in the quadratic context.

Burer and Letchford [10] also classified linear inequalities valid for **QPB** according to the eigenvalues of the matrix of coefficients for X. Specifically, the inequality

$$B \bullet X + \alpha^T x + \gamma \leq 0 \tag{3}$$

is called *convex-QPB*, *concave-QPB*, or *indefinite-QPB*, if its associated quadratic form $x^T Bx + \alpha^T x + \gamma$ is convex, concave or indefinite, respectively. Burer and Letchford proved the following results for convex and concave-QPB inequalities.

Proposition 2 ([10],Proposition 8). *A point* $(\bar{x}, \bar{X}) \in \mathbb{R}^{n+\frac{n(n+1)}{2}}$ *satisfies all concave-QPB inequalities if and only if it is in the convex set*

$$\left\{(x, X) \middle| X \succeq xx^T, x \in [0,1]^n\right\}.$$

The original proposition in [10] does not demonstrate the "only if" part of Proposition 2, but the result easily follows from the fact that $X \succeq xx^T$ is equivalent to (x, X) satisfying the infinitely-many concave inequalities

$$-\begin{pmatrix} s \\ v \end{pmatrix}^T \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \begin{pmatrix} s \\ v \end{pmatrix} = -(vv^T) \bullet X - 2(sv)^T x - s^2 \leq 0, \forall s \in \mathbb{R}, v \in \mathbb{R}^{n-1}.$$

This observation also establishes that it suffices to consider concave-QPB inequalities with $\mathbf{rank}(B) \leq 1$.

For convex-QPB inequalities, Burer and Letchford provided the following partial characterization.

Proposition 3 ([10], Proposition 9). *If* $B \bullet X + \alpha^T x + \gamma \leq 0$ *is a valid inequality for* **QPB** *and* $B \succeq 0$*, then it is valid for the convex set*

$$\{(x, X) | (x, \mathbf{ADiag}(X)) \in \mathbf{BQP}, X_{ii} \leq x_i, \forall i = 1, \ldots, n\}.$$

Proposition 3 only establishes the necessity for (3) to be a convex-QPB inequality, not its sufficiency. We fill this gap in Proposition 4 by considering a larger class that includes the convex-QPB inequalities.

Proposition 4. *A point* $(\bar{x}, \bar{X})$ *satisfies all inequalities* $B \bullet X + \alpha^T x + \gamma \leq 0$ *with* $B_{ii} \geq 0, \forall i = 1, ..., n$ *valid for* **QPB** *if and only if it is in the convex set*

$$\{(x, X) | (x, \mathbf{ADiag}(X)) \in \mathbf{BQP}, X_{ii} \leq x_i, \forall i = 1, \ldots, n\}.$$

Proof. The proof is given in our extended version [13].

We call inequalities (3) with $B_{ii} \geq 0$ valid for **QPB** *posdiag-QPB* inequalities.

Let $\mathcal{Q}$ be the intersection of the two convex sets in Propositions 2 and 4, i.e., $\mathcal{Q}$ is the relaxation of **QPB** defined by all concave and posdiag-QPB inequalities.

Separating concave-QPB inequalities can be done in polynomial time, but separating convex, or posdiag-QPB inequalities is NP-Complete, as **BQP** is affinely equivalent to the cut polytope [22].

Burer and Letchford demonstrate that $\mathbf{QPB} \subsetneq \mathcal{Q}$, even for $n = 3$, although it follows from [2] that $\mathbf{QPB} = \mathcal{Q}$ for $n \leq 2$. On the other hand, $\mathcal{Q}$ empirically has been shown to be a very tight relaxation of **QPB**. Specifically, Anstreicher [1] shows that using a subset of all valid inequalities for $\mathcal{Q}$ suffices to solve 49 of 50 instances (up to size $n = 60$) of the BoxQP library [11] at the root node. The inequalities used in the study of Anstreicher are all concave-QPB inequalities and posdiag-QPB inequalities derived via the Reformulation-Linearization Technique [24] and the triangle inequalities for **BQP** introduced by [22].

In the remainder of the paper, we study valid inequalities for the case $\mathbf{conv}(S)$ (and $\mathbf{conv}(S^{\succeq})$), when the indicator variables z come into play. Note that by setting $z_i = 1\ \forall i$, $\mathbf{conv}(S)$ is easily mapped to **QPB**. Our hope is to capitalize on the strength of $\mathcal{Q}$ as a relaxation of **QPB** to generate strong relaxations for $\mathbf{conv}(S)$. More specifically, for any valid inequality for $\mathbf{conv}(S)$

$$B \bullet X + \alpha^T x + \gamma \leq \delta^T z, \tag{4}$$

the inequality $B \bullet X + \alpha^T x + (\gamma - \delta^T e) \leq 0$ is a valid inequality for **QPB**, where e is a vector of all ones with proper dimension. In this sense, valid inequalities for $\mathbf{conv}(S)$ can be obtained by lifting valid inequality for **QPB**, i.e., by determining δ and modifying the constant term appropriately. We analyze the strength of lifted-concave and lifted-posdiag-QPB inequalities separately in the following two sections.

3 Lifted-Posdiag-QPB Inequalities

In this section we characterize the set defined by all lifted-posdiag-QPB inequalities for $\mathbf{conv}(S)$. The analysis shows the "negative" result that lifted-posdiag-QPB inequalities provide no restriction on z_i other than that provided by the continuous relaxation: $x_i \leq z_i \leq 1$.

Theorem 1. *A point* $(\bar{x}, \bar{X}, \bar{z}) \in \mathbb{R}^{2n+\frac{n(n+1)}{2}}$ *satisfies all valid inequalities* $B \bullet X + \alpha^T x + \gamma \leq \delta^T z$ *for* $\mathbf{conv}(S)$*, with* $B_{ii} \geq 0, \forall i = 1, \ldots, n$*, if and only if it is in the following convex set:*

$$\{(x, X, z) | (x, \mathbf{ADiag}(X)) \in \mathbf{BQP}, X_{ii} \leq x_i \leq z_i \leq 1, \forall i = 1, \ldots .n\}. \tag{5}$$

Proof. We first show that if $(\bar{x}, \bar{X}, \bar{z})$ satisfies all valid inequalities for $\mathbf{conv}(S)$ with $B_{ii} \geq 0$, then the point is in the set defined in (5). Since **BQP** is a projection of **QPB**, any valid inequality for $(x, \mathbf{ADiag}(X)) \in \mathbf{BQP}$ is a lifted-posdiag-QPB inequality for $\mathbf{conv}(S)$, as the coefficients for X_{ii} are zeros. The inequalities $X_{ii} - x_i \leq 0$, $x_i \leq z_i$ and $-1 \leq -z_i$ are also lifted-posdiag-QPB inequalities.

To prove the other direction, let $(\bar{x}, \bar{X}, \bar{z})$ be such that $(\bar{x}, \mathbf{ADiag}(\bar{X})) \in \mathbf{BQP}, \bar{X}_{ii} \leq \bar{x}_i \leq \bar{z}_i \leq 1 \forall i = 1, \ldots, n$. We show this point satisfies all lifted-posdiag-QPB inequalities for $\mathbf{conv}(S)$. The first claim is that it suffice to show

this for all lifted-posdiag-QPB inequalities with $\delta_i \geq 0\ \forall i = 1, \ldots, n$. A proof of the claim is given in our extended version [13].

Claim. $B \bullet X + \alpha^T x + \gamma \leq \delta^T z$ is valid for $\mathbf{conv}(S)$ if and only if the tighter inequality

$$B \bullet X + \alpha^T x + \gamma \leq \sum_{i:\delta_i \geq 0} \delta_i z_i + \sum_{i:\delta_i < 0} \delta_i \tag{6}$$

is also valid for $\mathbf{conv}(S)$.

Next for any $B \bullet X + \alpha^T x + \gamma \leq \delta^T z$ valid for $\mathbf{conv}(S)$, if $x = z \in \{0,1\}^n$, we have that $x^T B x + (\alpha - \delta)^T x + \gamma \leq 0$ for all $x \in \{0,1\}^n$.

As we assumed $(\bar{x}, \mathbf{ADiag}(\bar{X})) \in \mathbf{BQP}$, there exists a set with at most $K = n + \frac{n(n+1)}{2} + 1$ binary vectors: $\{y_k\}_{k=1}^K$ such that $\bar{x} = \sum_{k=1}^K \lambda_k y_k$ and $\bar{X} - \mathbf{Diag}(\bar{X}) + \mathbf{Diag}(\bar{x}) = \sum_{k=1}^K \lambda_k y_k y_k^T$. Here $\lambda_k \geq 0$, $\sum_k \lambda_k = 1$, $\bar{X} - \mathbf{Diag}(\bar{X}) + \mathbf{Diag}(\bar{x})$ means replacing the diagonal of $\bar{X}$ with entries in $\bar{x}$, i.e., $\mathbf{Diag}(\bar{X})$ is a diagonal matrix with the diagonal entries of $\bar{X}$, and $\mathbf{Diag}(\bar{x})$ is a diagonal matrix with entries of vector $\bar{x}$. Then,

$$\begin{aligned}
&B \bullet \bar{X} + \alpha^T \bar{x} + \gamma - \delta^T \bar{z} \leq B \bullet \bar{X} + (\alpha - \delta)^T \bar{x} + \gamma \\
&= B \bullet (\bar{X} - \mathbf{Diag}(\bar{X}) + \mathbf{Diag}(\bar{x})) + (\alpha - \delta)^T \bar{x} + \gamma + \sum_{i=1}^n B_{ii}(\bar{X}_{ii} - \bar{x}_i) \\
&\leq B \bullet \left(\sum_k \lambda_k y_k y_k^T \right) + (\alpha - \delta)^T \left(\sum_k \lambda_k y_k \right) + \gamma \\
&= \sum_k \lambda_k \left(B \bullet y_k y_k^T + (\alpha - \delta)^T y_k + \gamma \right) \leq 0.
\end{aligned}$$

The first inequality follows because $\delta_i \geq 0$ and $\bar{x}_i \leq \bar{z}_i$. The second inequality is because $B_{ii} \geq 0$ and $\bar{X}_{ii} \leq \bar{x}_i$. The final inequality follows from the observation in the previous paragraph. This concludes our proof.

A similar negative result holds for $\mathbf{conv}(S^{\succeq})$.

Proposition 5. *An inequality $B \bullet X + \alpha^T x + \gamma \leq \delta^T z$ with $B_{ii} \geq 0$, $\forall i = 1, \ldots, n$ is valid for $\mathbf{conv}(S^{\succeq})$ if and only if $B = 0$ and $\alpha^T x + \gamma \leq \delta^T z$ is valid for the convex set $\{(x, z) \mid 0 \leq x \leq z \leq 1\}$.*

Proof. The proof is given in our extended version [13].

4 Lifted-Concave-QPB Inequalities

In this section, we consider the lifted-concave-QPB inequalities for $\mathbf{conv}(S)$ and show that the class defines $\mathbf{conv}(S^{\succeq})$.

Proposition 6. *A point $(\bar{x}, \bar{X}, \bar{z}) \in \mathbb{R}^{2n + \frac{n(n+1)}{2}}$ satisfies all valid inequalities $B \bullet X + \alpha^T x + \gamma \leq \delta^T z$ for $\mathbf{conv}(S)$, with $B \preceq 0$ if and only if $(\bar{x}, \bar{X}, \bar{z}) \in \mathbf{conv}(S^{\succeq})$.*

Proof. The proof uses the fact that

$$\mathbf{conv}(S^{\succeq}) = \mathbf{conv}(S) + \left\{(0,0,X) \in \mathbb{R}^{2n+\frac{n(n+1)}{2}}, X \succeq 0\right\}$$

and is given in our extended version [13].

Next we consider the special case where each of B, α, and δ have at most one nonzero entry. We show that this class of inequalities includes all perspective cuts that use diagonal entries of X. Further, we show that by adding this simple class of inequalities to the semidefinite programming (SDP) relaxation of (1) when $Q \succeq 0$ results in an relaxation equivalent to the recent optimal diagonal splitting approach of [26]. We first characterize all valid inequalities for $\mathbf{conv}(S)$ that involve only x, $\mathbf{diag}(X)$ and z.

Theorem 2. *A point $(\bar{x}, \bar{z}, \bar{X})$ satisfies all valid inequalities $\sum_{i=1}^n b_i X_{ii} + \alpha^T x + \gamma \leq \delta^T z$ for $\mathbf{conv}(S)$ if and only if it is in the convex set*

$$\mathbf{P} := \left\{(x,z,X) \middle| \begin{array}{l} 0 \leq X_{ii} \leq x_i \leq z_i \leq 1, \\ X_{ii} z_i \geq x_i^2, \forall i = 1, ..., n \end{array} \right\}.$$

Proof. Note that the definition of $\mathbf{P}$ involves only x, z and $\mathbf{diag}(X)$. For all $i = 1, \ldots, n$, since $X_{ii} \geq 0$ and $z_i \geq 0$, the second-order-cone representable constraints $X_{ii} z_i \geq x_i^2$ are can be replaced by their (infinite number of) linearized inequalities. At point $(\hat{x}_i, \hat{X}_{ii}, \hat{z}_i)$ such that $\hat{X}_{ii}\hat{z}_i = \hat{x}_i^2$ and $0 \leq \hat{x}_i \leq \hat{z}_i \leq 1$, the linearization is

$$-\hat{z}_i X_{ii} + 2\hat{x}_i x_i \leq \hat{X}_{ii} z_i. \tag{7}$$

So if $(\bar{x}, \bar{z}, \bar{X})$ satisfies all $\sum_{i=1}^n b_i X_{ii} + \alpha^T x + \gamma \leq \delta^T z$ that are valid for $\mathbf{conv}(S)$, it must be in $\mathbf{P}$.

On the other hand, if $\sum_{i=1}^n b_i X_{ii} + \alpha^T x + \gamma \leq \delta^T z$ is valid for $\mathbf{conv}(S)$, then $\gamma \leq \min\{\delta^T z - \sum_{i=1}^n b_i x_i^2 - \alpha^T x | 0 \leq x_i \leq z_i \in \{0,1\}, \forall i\}$. Define $\gamma_i = \min\{\delta_i z_i - b_i x_i^2 - \alpha_i x_i | 0 \leq x_i \leq z_i \in \{0,1\}\}$, we must have $\gamma \leq \sum_{i=1}^n \gamma_i$. Further, each disaggregated inequality $b_i X_{ii} + \alpha_i x_i + \gamma_i \leq \delta_i z_i$ is valid for $\{(x_i, z_i, x_i^2) | 0 \leq x_i \leq z_i \in \{0,1\}\}$. By the convex hull characterization of the latter set (for example [17]), such a disaggregated inequality is valid for $\mathbf{P}$. Therefore $\sum_{i=1}^n b_i X_{ii} + \alpha^T x + \gamma \leq \delta^T z$ is also valid for $\mathbf{P}$.

The inequalities $X_{ii} z_i \geq x_i^2$ are called *perspective constraints* in the literature [16,17,18]. In these works, the variables X_{ii} are introduced to represent x_i^2. For fixed i, in the space of (x_i, z_i, X_{ii}), the lower convex envelope of the feasible set $\{(0,0,0)\} \cup \{(x_i, 1, x_i^2) | 0 \leq x_i \leq 1\}$ is

$$\tilde{X}_{ii}(z_i, x_i) = \begin{cases} \frac{x_i^2}{z_i}, & 0 \leq x_i \leq z_i \leq 1, z_i \neq 0, \\ 0, & x_i = z_i = 0. \end{cases}$$

So we see that $X_{ii} \geq \tilde{X}_{ii}(z_i, x_i)$ is equivalent to $X_{ii} z_i \geq x_i^2$ with additional restriction $0 \leq X_{ii} \leq x_i \leq z_i \leq 1$.

It is shown, for example in [17], that if the nonlinear functions are appropriately separable (in our context, that there are no off-diagonal entries of X appearing in the objective or constraints), employing perspective constraints improves the solution time significantly for convex MINLPs. For the case of non-separable quadratic programs, one approach is to extract a separable part from the objective function, and apply the perspective constraints on this separable part. We briefly describe this procedure here and show how it is related with the simplest class of lifted-concave-QPB inequalities.

Let ζ denote the optimal value of (1) with $Q \succeq 0$. A method to strengthen the continuous relaxation of (1) proposed by [16] is to find a diagonal matrix D with $D_{ii} \geq 0\ \forall i$ and $Q - D \succeq 0$, and to solve the diagonally-split convex (perspective) relaxation

$$\zeta_{PR}(D) := \min_{p,x,z} \left\{ x^T(Q-D)x + \sum_{i=1}^{n} p_i + q^T x + c^T z \,\middle|\, \begin{array}{l} Ax + Bz \leq b, p_i z_i \geq D_{ii} x_i^2 \\ 0 \leq x_i \leq z_i \leq 1, \forall i \end{array} \right\}.$$

The constraints $p_i z_i \geq D_{ii} x_i^2$ come again from the fact that the function $f(x_i, z_i) = \frac{D_{ii} x_i^2}{z_i}$ (if we define $f(0,0) = 0$) is the lower convex envelope of set $\{(0,0)\} \cup \left\{(D_{ii} x_i^2, 1) \;\middle|\; 0 \leq x_i \leq 1\right\}$ in the space of (x_i, z_i). The matrix D can be chosen to be $\lambda_{\min} I$ if Q is positive definite with $\lambda_{\min}$ as its minimum eigenvalue, or D can be obtained from the solution of a semidefinite program that seeks to maximize its trace. The work [16] also illustrates that this approach improves the performance of standard commercial solvers by several orders of magnitude on some portfolio optimization problems. In [16], the second order cone constraints $p_i z_i \geq D_{ii} x_i^2$ are used to generate linear cutting planes (perspective cuts) like (7).

An alternative way of constructing a tight relaxation is to use SDP. The standard semidefinite relaxation for (1) is

$$\zeta_{SDP} := \min \left\{ Q \bullet X + q^T x + c^T z \,\middle|\, \begin{array}{l} X \succeq xx^T, Ax + Bz \leq b, \\ 0 \leq x_i \leq z_i \leq 1, \forall i \end{array} \right\}, \tag{8}$$

and it is easy to show that the bound obtained from (8) is equal to the bound obtained from the continuous relaxation of (1). However, if we strengthen (1) by adding the perspective constraints as in Theorem 2, we obtain a semidefinite relaxation which is no worse than $\zeta_{PR}(D)$ with **any** valid splitting $Q = D + (Q - D)$. Specifically, if we define

$$\zeta_{SDP/PR} := \min \left\{ Q \bullet X + q^T x + c^T z \,\middle|\, \begin{array}{l} X \succeq xx^T, Ax + Bz \leq b, \\ X_{ii} z_i \geq x_i^2, 0 \leq x_i \leq z_i \leq 1, \forall i \end{array} \right\}, \tag{9}$$

then we have the following proposition.

Proposition 7. *For all diagonal $D \succeq 0$ and $Q - D \succeq 0$, $\zeta \geq \zeta_{SDP/PR} \geq \zeta_{PR}(D)$.*

Proof. It is straightforward to see $\zeta \geq \zeta_{SDP/PR}$. Suppose $(\bar{x}, \bar{X}, \bar{z})$ is an optimal solution to (9), then for any nonnegative diagonal D such that $Q - D \succeq 0$,

$$\begin{aligned}\zeta_{SDP/PR} &= Q \bullet \bar{X} + q^T\bar{x} + c^T\bar{z} = D \bullet \bar{X} + (Q-D) \bullet \bar{X} + q^T\bar{x} + c^T\bar{z} \\ &\geq \sum_{i:\bar{z}_i>0} D_{ii}\frac{\bar{x}_i^2}{\bar{z}_i} + \bar{x}^T(Q-D)\bar{x} + q^T\bar{x} + c^T\bar{z} \;\geq\; \zeta_{PR}(D).\end{aligned}$$

The first inequality is due to the fact that $\bar{X}_{ii}\bar{z}_i \geq \bar{x}_i^2$ and $\bar{X} \succeq \bar{x}\bar{x}^T$, and last one is by definition of $\zeta_{PR}(D)$.

Further, if under some mild conditions, we can illustrate that there exists an "optimal" D^* such that $\zeta_{SDP/PR} = \zeta_{PR}(D^*)$. This result can be seen as a more natural derivation of the main result in [26], while our result deals with slightly more general linear constraints.

Proposition 8. *Suppose at least one of the following two conditions are satisfied,*

1. *$\exists \bar{x}, \bar{z}$ such that $A\bar{x} + B\bar{z} < b, 0 < \bar{x}_i < \bar{z}_i < 1, \forall i = 1, \ldots, n$ (Slater Condition);*
2. *Q is positive definite.*

Let $\left(\hat{y}, \hat{\alpha}, \hat{\beta}, \hat{\gamma}, \hat{s}, \hat{v}, \hat{W}, \hat{\lambda}, \hat{\mu}, \hat{\tau}\right)$ be an optimal solution to the following semidefinite optimization

$$\begin{aligned}\zeta^D_{SDP/PR} := \max\; & -b^T y - s - e^T\tau \\ s.t.\; & Q - \mathbf{Diag}(\alpha) = W \\ & q + A^T y = 2\gamma + 2v + \lambda - \mu \\ & c + B^T y = \beta + \mu - \tau \\ & \begin{pmatrix} s & v^T \\ v & W \end{pmatrix} \succeq 0, \begin{pmatrix} \alpha_i & \gamma_i \\ \gamma_i & \beta_i \end{pmatrix} \succeq 0, \forall i = 1, ..., n, \\ & y, \lambda, \mu, \tau \in \mathbb{R}^n_+,\end{aligned}$$

then $\zeta_{PR}(\mathbf{Diag}(\hat{\alpha})) = \zeta_{SDP/PR} = \zeta^D_{SDP/PR}$.

Proof. The proof is given in our extended version [13].

Two remarks are in order. First, Proposition 7 and 8 are relevant to results for the so called QCR method [5,6]. The QCR method aims to **convexify** nonconvex quadratic programs by adding terms which do not change the optimal value, for example by adding a constant multiple of $x_i^2 - x_i$ if x_i is binary, or $(a^T x - b)^2$ if $a^T x = b$ is a valid constraint. The diagonal splitting approach works in the opposite manner. One starts with a convex objective, extracts a separable part while maintaining the convexity, and strengthen the separable terms using perspective constraints. It is interesting that in both cases, the optimal reformulation parameters can be found by solving an SDP. Second, as suggested by Kurt Anstreicher (personal communication), the inequalities $X_{ii}z_i \geq x_i^2$ are implied by the standard doubly nonnegative (DNN) relaxations [8,9] for (1).

5 Separation of Lifted-Concave-QPB Inequalities via Simultaneous Lifting

In this section we show that if the number of binary variables appearing in the inequality ($\mathbf{Card}(\delta)$) is fixed, then separation for lifted-concave-QPB inequalities can be accomplished by solving a semidefinite programming problem of size polynomial in n. Key to showing this result is a "dual" result to Proposition 2, which gives a direct characterization of all concave-QPB inequalities.

Proposition 9. *An inequality $B \bullet X + \alpha^T x + \gamma \leq 0$ is a concave-*$\mathbf{QPB}$ *inequality if and only if (B, α, γ) is in the following set $\mathcal{V}_n$:*

$$\mathcal{V}_n := \left\{ (B,\alpha,\gamma) \middle| \begin{array}{l} \begin{pmatrix} s & v^T \\ v & -B \end{pmatrix} \succeq 0, \;\; \mu - 2v + \lambda = \alpha \\ -s - \mu^T e \geq \gamma, \, v \in \mathbb{R}^n, \lambda, \mu \in \mathbb{R}^n_+, s \geq 0 \end{array} \right\}.$$

Proof. The proposition is proved by noting that $B \bullet X + \alpha^T x + \gamma \leq 0$ is a concave-QPB inequality if and only if the following optimization (P) has non-positive optimal objective value, where (D) is the associated dual problem.

$$\begin{array}{ll} \max\limits_{0 \leq x \leq e} & B \bullet X + \alpha^T x + \gamma \\ s.t., & \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \succeq 0 \end{array} \qquad \text{(P)} \qquad \begin{array}{ll} \min\limits_{\lambda,\mu \in \mathbb{R}^n_+} & \gamma + s + \mu^T e, \\ s.t., & \alpha = \mu - 2v - \lambda \\ & \begin{pmatrix} s & v^T \\ v & -B \end{pmatrix} \succeq 0 \end{array} \qquad \text{(D)}$$

The primal problem satisfies the Slater condition, so by strong duality the conclusion easily follows.

We use Proposition 9 to create a separation problem for lifted-concave-QPB inequalities. Note that $B \bullet X + \alpha^T x + \gamma \leq \delta^T z$ is a valid lifted-concave-QPB inequality and $\mathbf{Card}(\delta) \leq k$ if and only if for all $I \subseteq \{1, ..., n\}$, $|I| \geq n - k$, $(B_{[I,I]}, \alpha_I, \gamma - \delta^T e_I) \in \mathcal{V}_{|I|}$, where $B_{[I,I]}$, α_I are the corresponding principal submatrix and subvector, and e_I is a vector with ones at indices in I and zeros elsewhere. Thus, for fixed k, the separation problem of all lifted-concave-QPB inequalities with $\mathbf{Card}(\delta) \leq k$ can be written as an SDP of polynomial size in n. (The number of choices of I increases at rate $O(n^k)$).

We conclude this extended abstract by providing a small computational example to illustrate using that lifted-QPB-inequalities can improve the DNN relaxation, even for $n = 3$. The example seems to also suggest the importance of lifted concave inequalities with $\mathbf{rank}(B)$ small.

Example 1 (Non-dominance by doubly non-negative relaxation). We consider the following convex quadratic program with binary indicators

$$\begin{array}{ll} \min\limits_{x \in [0,1]^3} & x^T Q x + c^T x + d^T z \\ s.t. & 0 \leq x_i \leq z_i, z_i \in \{0,1\}, i = 1,2,3, \end{array}$$

where

$$Q = \begin{pmatrix} 4.4 & 3.1 & -4.2 \\ 3.1 & 3.0 & -3.2 \\ -4.2 & -3.2 & 4.6 \end{pmatrix}, c = \begin{pmatrix} -1.4 \\ -1.4 \\ 0.1 \end{pmatrix}, d = \begin{pmatrix} 0.4 \\ 0.2 \\ 0.5 \end{pmatrix}.$$

The optimal value is 0 and the optimal solution is $x = z = 0$. The DNN relaxation [8] (solved by using Yalmip [20] with CSDP [7]) yields a lower bound that equals approximately -3.89×10^{-2}. Then we employ the SDP-based separation procedure based on Proposition 9 with $k = 3$ to generate a valid lifted concave inequality, and then resolve the strengthened DNN relaxation. The lower bound is improved to the exact optimal value 0 (with accuracy about 10^{-10}) after three rounds. This computationally verifies Proposition 6. It is worth noting that the eigenvalues of B matrices in three cuts are

$$[0.0000, 0.0000, -0.5492], [0.0000, -0.0469, -0.6526], [0.0000, 0.0000, -0.7511],$$

respectively, i.e., all of the B matrices are close to rank-one.

6 Discussion and Future Work

Results in this extended abstract leave interesting open questions that we hope to address in future work. First, note for the set **QPB**, we may assume that all concave inequalities have $\mathbf{rank}(B) \leq 1$. A natural question is the extent to which this result is true for $\mathbf{conv}(S)$. Example 1 suggests that lifted concave-QPB inequalities with low rank of B may be more important than those with high rank. Next, can we design effective separation heuristic algorithms for lifted concave-QPB inequalities, especially when B has low rank? Last but not least, does the lifted concave approach motivate "projected formulations" where one derives valid inequalities using only $O(n)$ number of variables?

References

1. Anstreicher, K.M.: On convex relaxations for quadratically constrained quadratic programming. Mathematical Programming, Series B (2012)
2. Anstreicher, K.M., Burer, S.: Computable representations for convex hulls of low-dimensional quadratic forms. Mathematical Programming 124(1-2), 33–43 (2010)
3. Bertsimas, D., Shioda, R.: Algorithm for cardinality-constrained quadratic optimization. Computational Optimization and Applications 43(1), 1–22 (2009)
4. Bienstock, D.: Computational study of a family of mixed-integer quadratic programming problems. Mathematical Programming, Series A 74(2), 121–140 (1996)
5. Billionnet, A., Elloumi, S., Lambert, A.: Extending the QCR method to general mixed-integer programs. Mathematical Programming, Series A 131, 381–401 (2012)
6. Billionnet, A., Elloumi, S., Plateau, M.-C.: Improving the performance of standard solvers for quadratic 0-1 programs by a tight convex reformulation: The QCR method. Discrete Applied Mathematics 157, 1185–1197 (2009)
7. Borchers, B.: CSDP, a C library for semidefinite programming. Optimization Methods and Software 11(1), 613–623 (1999)

8. Burer, S.: On the copositive representation of binary and continuous nonconvex quadratic programs. Mathematical Programming 120, 479–495 (2009)
9. Burer, S.: Optimizing a polyhedral-semidefinite relaxation of completely positive programs. Math. Prog. Comp. 2(1), 1–19 (2010)
10. Burer, S., Letchford, A.N.: On nonconvex quadratic programming with box constriants. SIAM J. Optim. 20(2), 1073–1089 (2009)
11. Burer, S., Vandenbussche, D.: A finite branch-and-bound algorithm for nonconvex quadratic programming via semidefinite relaxations. Mathematical Programming 113, 259–282 (2008)
12. D'Ambrosio, C., Linderoth, J., Luedtke, J.: Valid Inequalities for the Pooling Problem with Binary Variables. In: Günlük, O., Woeginger, G.J. (eds.) IPCO 2011. LNCS, vol. 6655, pp. 117–129. Springer, Heidelberg (2011)
13. Dong, H., Linderoth, J.: On valid inequalities for quadratic programming with continuous variables and binary indicators. Optimization Online (August 2012)
14. Frangioni, A., Gentile, C.: Perspective cuts for a class of convex 0-1 mixed integer programs. Mathematical Programming 106, 225–236 (2006)
15. Frangioni, A., Gentile, C.: Solving nonlinear single-unit commitment problems with ramping constraints. Operations Research 54(4), 767–775 (2006)
16. Frangioni, A., Gentile, C.: SDP diagonalizations and perspective cuts for a class of nonseparable MIQP. Operations Research Letters 35(2), 181–185 (2007)
17. Günlük, O., Linderoth, J.: Perspective reformulations of mixed integer nonlinear programming with indicator variables. Mathematical Programming, Series B 124(1-2), 183–205 (2010)
18. Günlük, O., Linderoth, J.: Perspective reformulation and applications. In: Lee, J., Leyffer, S. (eds.) IMA Volumes in Mathematics and its Applications, vol. 154, pp. 61–92. Springer (2012)
19. Gao, J., Li, D.: Cardinality constraint linear-quadratic optimal control. IEEE Transactions on Automatic Control 56(8), 1936–1941 (2011)
20. Löfberg, J.: Yalmip: A toolbox for modeling and optimization in matalb. In: Proceedings of the CACSD Conference, Taipei, Taiwan (2004)
21. Miller, A.: Subset Selection in Regression. Monographs in Statistics and Applied Probability, vol. 40. Chapman and Hall, London (1990)
22. Padberg, M.: The Boolean quadric polytope: some characteristics, facets and relatives. Math. Programming, Series B 45(1), 139–172 (1989)
23. Papageorgiou, D.J., Toriello, A., Nemhauser, G.L., Savelsbergh, M.W.P.: Fixed-charge transportation with product blending. Transportation Science 46(2), 281–295 (2012)
24. Sherali, H.D., Adams, W.P.: A reformulation-linearization technique for solving discrete and continuous nonconvex problems. Nonconvex Optimization and its Applications, vol. 31. Kluwer Academic Publishers, Dordrecht (1999)
25. Wei, D., Oppenheim, A.V.: A branch-and-bound algorithm for quadratically-constrained sparse filter design. IEEE Transactions on Signal Processing (2012) (to appear)
26. Zheng, X., Sun, X., Li, D.: Improving the performance of MIQP solvers for quadratic programs with cardinality and minimum threshold constraints: A semidefinite program approach (November 2010) (manuscript)

An Improved Integrality Gap for Asymmetric TSP Paths

Zachary Friggstad[1], Anupam Gupta[2,⋆], and Mohit Singh[3]

[1] Department of Combinatorics and Optimization, University of Waterloo
[2] Department of Computer Science, Carnegie Mellon University
[3] Microsoft Research, Redmond

Abstract. The Asymmetric Traveling Salesperson Path (ATSPP) problem is one where, given an *asymmetric* metric space (V, d) with specified vertices s and t, the goal is to find an s-t path of minimum length that visits all the vertices in V.

This problem is closely related to the Asymmetric TSP (ATSP) problem, which seeks to find a tour (instead of an s-t path) visiting all the nodes: for ATSP, a ρ-approximation guarantee implies an $O(\rho)$-approximation for ATSPP. However, no such connection is known for the *integrality gaps* of the linear programming relxations for these problems: the current-best approximation algorithm for ATSPP is $O(\log n/\log\log n)$, whereas the best bound on the integrality gap of the natural LP relaxation (the subtour elmination LP) for ATSPP is $O(\log n)$.

In this paper, we close this gap, and improve the current best bound on the integrality gap from $O(\log n)$ to $O(\log n/\log\log n)$. The resulting algorithm uses the structure of narrow s-t cuts in the LP solution to construct a (random) tree witnessing this integrality gap. We also give a simpler family of instances showing the integrality gap of this LP is at least 2.

1 Introduction

In the Asymmetric Traveling Salesperson Path (ATSPP) problem, we are given an *asymmetric* metric space (V, d) (i.e., one where the distances satisfy the triangle inequality, but potentially not the symmetry condition), and also specified source and sink vertices s and t, and the goal is to find an s-t Hamilton path of minimum length.

This ATSPP problem is a close relative of the Asymmetric TSP problem (ATSP), where the goal is to find a Hamilton tour instead of an s-t path. For this ATSP problem, the $\log_2 n$-approximation of Frieze, Galbiati, and Maffioli [9] from 1982 was improved by constant factors in [4,11,8]. A remarkable breakthrough on this problem was an $O(\frac{\log n}{\log\log n})$-approximation result due to Asadpour, Goemans, Mądry, Oveis Gharan, and Saberi [2] where they also bounded the integrality gap of the subtour elimination linear programming relaxation for ATSP by the same factor.

⋆ Research was partly supported by NSF awards CCF-0964474 and CCF-1016799.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 181–192, 2013.

Surprisingly the study of ATSPP has been of a more recent vintage: the first approximations appeared around 2005 [12,6,8]. It is easily seen that the ATSP reduces to ATSPP in an approximation preserving fashion (by guessing two consecutive nodes on the tour). In the other direction, [8] showed that a ρ-approximation to the ATSP problem implies an $O(\rho)$-approximation to the ATSPP problem. Using the above-mentioned $O(\frac{\log n}{\log \log n})$-approximation for ATSP [2], this implies an $O(\frac{\log n}{\log \log n})$-approximation for ATSPP as well.

The subtour elimination linear program generalizes simply to the ATSPP problem and is given in Section 2. However, the best previous integrality gap for this LP for ATSPP was $O(\log n)$ [10]. In this paper we show the following result.

Theorem 1. *The integrality gap of the subtour elimination linear program for the ATSPP problem is at most* $O(\frac{\log n}{\log \log n})$.

We also give a simple construction showing that the integrality gap of this LP is at least 2; this example is simpler than previous known integrality gap instance showing the same lower bound, due to Charikar, Goemans, and Karloff [5].

Given the central nature of linear programs in approximation algorithms, it is useful to understand the integrality gaps for linear programming relaxations of optimization problems. Not only does this study give us a deeper understanding into the underlying problems, but also upper bounds on the integrality gap of LPs are often required for some reductions to go through. For example, the polylogarithmic approximation guarantees in the work of Nagarajan and Ravi [13] for Directed Orienteering and Minimum Ratio Rooted Cycle, and those in the work of Bateni and Chuzhoy [3] for Directed k-Stroll and Directed k-Tour were all improved by a factor of $\log \log n$ following the improved bound of $O(\frac{\log n}{\log \log n})$ on the integrality gap of the subtour LP relaxation for ATSP. Note that these improvements do not follow merely from improved approximation guarantees.

1.1 Our Approach

Our approach to bound the integrality gap for ATSPP is similar to that for ATSP [2], but with some crucial differences. We sample a random spanning tree whose marginals are close to the optimal LP solution x^* and then augment the directed version of this tree to an integral circulation using Hoffman's circulation theorem while ensuring the t-s edge is only used once. Following the corresponding Eulerian circuit and deleting the t-s edge results in a spanning s-t walk.

However, the non-Eulerian nature of the ATSPP problem makes it difficult to satisfy the cut requirements in Hoffman's circulation theorem if we sample the spanning tree directly from the distribution given by the LP solution. It turns out that the problems come from the s-t cuts U that are nearly-tight: i.e., which satisfy $1 < x^*(\partial^+(U)) < 1 + \tau$ for some small constant τ — these give rise to problems when the sampled spanning tree includes more than one edge across this cut. Such problems also arise in the symmetric TSP paths case (studied in a recent paper of An, Kleinberg, and Shmoys [1]): their approach is again to take

a random tree directly from the distribution given by the optimal LP solution, but in some cases they need to boost the narrow cuts, and they show that the loss due to this boosting is small.

In our case, the asymmetry in the problem means that boosting the narrow cuts might be prohibitively expensive. Hence, our idea is to preprocess the distribution given by the LP solution to *tighten* the narrow cuts, so that we never pick two edges from a narrow cut. Since the original LP solution lies in the spanning tree polytope, lowering the solution on some edges means we need to raise the fractional value on other edges, which may cause the cost to increase, and technical heart of the paper is to ensure this can be done with little extra loss.

1.2 Other Related Work

The first non-trivial approximation for ATSPP was an $O(\sqrt{n})$-approximation by Lam and Newman [12]. This was improved to $O(\log n)$ by Chekuri and Pál [6], and the constant was further improved in [8]. The paper [8] also showed that ATSP and ATSPP had approximability within a constant factor of each other. All these results are combinatorial and do not bound integrality gap of ATSPP. A bound of $O(\sqrt{n})$ on the integrality gap of ATSPP was given by Nagarajan and Ravi [14], and was improved to $O(\log n)$ by Friggstad, Salavatipour and Svitkina [10]. Note that there is no known result relating the integrality gaps of the ATSP and ATSPP problems in a black-box fashion.

In the symmetric case (where the problems become TSPP and TSP respectively), constant factor approximations and integrality gaps have long been known. We do not survey the rich body of literature on TSP here, instead pointing the reader to, e.g., the recent paper on graphical TSP by Sebő and Vygen [17]. It is, however, important to mention the the recent 1.618-approximation for TSPP in a beautiful new result by An, Kleinberg, and Shmoys [1] which has recently improved to a 1.6-approximation by Sebő [16]. They proceed via bounding the integrality gap of the LP relaxation, and their algorithm also proceeds via studying the narrow s-t cuts; the connections to their work are discussed in Section 1.1.

1.3 Notation and Preliminaries

Given a directed graph $G=(V,A)$, and two disjoint sets $U, U' \subseteq V$, let $\partial(U;U') = A \cap (U \times U')$. We use the standard shorthand that $\partial^+(U) := \partial(U; V \setminus U)$, and $\partial^-(U) := \partial(V \setminus U; U)$. When the set U is a singleton (say $U = \{u\}$), we use $\partial^+(u)$ or $\partial^-(u)$ instead of $\partial^+(\{u\})$ or $\partial^-(\{u\})$. For undirected graph $H = (V, E)$, we use $\partial(U;U')$ to denote edges crossing between U and U', and $\partial(U)$ to denote the edges with exactly one endpoint in U (which is the same as $\partial(V \setminus U)$).

For a digraph $G = (V, A)$, a set of arcs $B \subseteq A$ is *weakly connected* if the undirected version of B forms a connected graph that spans all vertices in A.

For values $x_a \in \mathbb{R}$ for all $a \in A$, and a set of arcs $B \subseteq A$, we let $x(B)$ denote the sum $\sum_{a \in B} x_a$.

Given an undirected graph $H = (V, E)$, we let $\chi_T \in \{0,1\}^{|E|}$ denote the characteristic vector of a spanning tree T, then the spanning tree polytope is the convex hull of $\{\chi_T \mid T \text{ spanning tree of } H\}$. See, e.g., [15, Chapter 50] for several equivalent linear programming formulations of this polytope. We sometimes abuse notation and call a set of directed arcs T a tree if the undirected version of T is a tree in the usual sense.

2 The Rounding Algorithm

In this section, we give the linear programming relaxation for the Asymmetric TSP Path problem, and show how to round it to get a path of cost at most $O(\frac{\log n}{\log \log n})$ times the cost of the optimal LP solution. We then give the proof, with some of the details being deferred to the following sections.

Given a directed metric graph $G = (V, A)$ with arc costs $\{c_a\}_{a \in A}$, we use the following standard linear programming relaxation for ATSPP which is also known as the subtour elimination linear program.

$$\begin{aligned}
&\text{minimize} : \sum_{a \in E} c_a x_a && && (ATSPP)\\
\text{s.t.} : \quad & x(\partial^+(s)) = x(\partial^-(t)) = 1 && && (1)\\
& x(\partial^-(s)) = x(\partial^+(t)) = 0 && && (2)\\
& x(\partial^+(v)) = x(\partial^-(v)) = 1 && \forall\ v \in V \setminus \{s,t\} && (3)\\
& x(\partial^+(U)) \geq 1 && \forall\ \{s\} \subseteq U \subsetneq V && (4)\\
& x_a \geq 0 && \forall\ a \in E &&
\end{aligned}$$

We begin by solving the above LP to obtain an optimal solution x^*. Consider the undirected (multi)graph $H = (V, E)$ obtained by removing the orientation of the arcs of G. That is, create precisely two edges between every two nodes $u, v \in V$ in H, one having cost c_{uv} and the other having cost c_{vu}. (Hence, $|E| = |A|$.) For a point $w \in \mathbb{R}^A_+$, let $\kappa(w)$ denote the corresponding point in $\mathbb{R}^E_+$, and view $\kappa(w)$ as the "undirected" version of w.

We will use the following definition: An *s-t cut* is a subset $U \subset V$ such that $\{s\} \subseteq U \subseteq V \setminus \{t\}$. The LP constraints imply that $x^*(\partial^+(U)) - x^*(\partial^-(U)) = 1$ for every *s-t* cut U. Also, $x^*(\partial^+(U)) = x^*(\partial^-(U)) \geq 1$ for every nonempty $U \subseteq V \setminus \{s,t\}$.

Definition 1 (Narrow cuts). *Let $\tau \geq 0$. An s-t cut U is* τ-narrow *if $x^*(\partial^+(U)) < 1 + \tau$ (or equivalently, $x^*(\partial^-(U)) < \tau$).*

The main technical lemma is the following:

Lemma 1. *For any $\tau \in [0, 1/4]$, one can find, in polynomial-time, a vector $z \in [0,1]^A$ (over the directed arcs) such that:*

(a) its undirected version $\kappa(z)$ lies in the spanning tree polytope for H,
(b) $z \leq \frac{1}{1-3\tau} x^*$ (where the inequality denotes component-wise dominance), and
(c) $z(\partial^+(U)) = 1$ and $z(\partial^-(U)) = 0$ for every τ-narrow s-t cut U.

Before we prove the lemma (in Section 2.1), let us sketch how it will be useful to get a cheap solution to the ATSPP. Since z (or more correctly, its undirected version $\kappa(z)$) lies in the spanning tree polytope, it can be represented as a convex combination of spanning trees. Using some recently-developed algorithms (e.g., those due to [2,7]) one can choose a spanning tree that crosses each cut only $O(\frac{\log n}{\log \log n})$ times more than the LP solution. Finally, we can use $O(\frac{\log n}{\log \log n})$ times the LP solution to patch this tree to get an s-t path. Since the LP solution is "weak" on the narrow cuts and may contribute very little to this patching (at most τ), it is crucial that by property (c) above, this tree will cross the narrow cuts *only once*, and that too, it crosses in the "right" direction, so we never need to use the LP when verifying the cut conditions of Hoffman's circulation theorem on narrow cuts. The details of these operations appear in Section 3.

2.1 The Structure of Narrow Cuts

We now prove Lemma 1: it says that we can take the LP solution x^* and find another vector z such that if a s-t cut is narrow in x^* (i.e., the total x^* value crossing the cut lies in $[1, 1+\tau)$, then z crosses it to an extent precisely 1. Moreover, the undirected version of z can be written as a convex combination of spanning trees, and z_a is not much larger than x^*_a for any arc a.

Note that the undirected version of x^* itself can be written as a convex combination of spanning trees. Thus if we force z to cross the narrow cuts to an extent less than x^* (loosely, this reduces the connectivity), we must increase the fractional value on other arcs. To show we can perform this operation without changing any of the coordinates by very much, we need to study the structure of narrow cuts more closely. (Such a study is done in the *symmetric* TSP path paper of An et al. [1], but our goals and theorems are somewhat different.)

First, say two s-t cuts U and W *cross* if $U \setminus W$ and $W \setminus U$ are non-empty.

Lemma 2. *For $\tau \leq 1/4$, no two τ-narrow s-t cuts cross.*

Lemma 2 says that the τ-narrow cuts form a chain $\{s\} = U_1 \subset U_2 \subset \ldots \subset U_k = V \setminus \{t\}$ with $k \geq 2$. For $1 < i \leq k$. let $L_i := U_i \setminus U_{i-1}$. We also define $L_1 = \{s\}$ and $L_{k+1} = \{t\}$. Let $L_{\leq i} := \bigcup_{j=1}^{i} L_i$ and $L_{\geq i} := \bigcup_{j=i}^{k+1} L_i$. For the rest of this paper, we will use τ to denote a value in the range $[0, 1/4]$. Ultimately, we will set $\tau := 1/4$ for the final bound but we state the lemmas in their full generality for $\tau \leq 1/4$.

Next, we show that out of the (at most) $1+\tau$ mass of x^* across each τ-narrow cut U_i, most of it comes from the "local" arcs in $\partial(L_i; L_{i+1})$.

Lemma 3. *For each $1 \leq i \leq k$; $x^*(\partial(L_i, L_{i+1})) \geq 1 - 3\tau$.*

Now, recall that $\kappa(x^*)$ denotes the assignment of arc weights to the graph $H = (V, E)$ from the previous section obtained by "removing" the directions from arcs in A. We prove that the restriction of $\kappa(x^*)$ to any L_i almost satisfies the partition inequalities that characterize the convex hull of connected graphs. For a partition $\pi = \{W_1, \ldots, W_\ell\}$, we let $\partial(\pi)$ denote the set of edges whose endpoints lie in two different sets in the partition.

Lemma 4. *For any $1 \le i \le k+1$ and any partition $\pi = \{W_1, \ldots, W_\ell\}$ of L_i, we have $\kappa(x^*)(\partial(\pi)) \ge \ell - 1 - 2\tau$.*

The following corollary will be useful.

Corollary 1. *For any partition π of L_i, we have $\frac{\kappa(x^*)(\partial(\pi))}{1-2\tau} \ge |\pi| - 1$.*

Finally, to efficiently implement the arguments in the proof of Lemma 1, we need to be able to efficiently find all τ-narrow cuts U_i. This is done by a standard recursive algorithm that exploits the fact that the cuts are nested.

Lemma 5. *There is a polynomial-time algorithm to find all τ-narrow $s-t$ cuts.*

We are now in a position to prove Lemma 1, the main result of this section.

Proof (Proof of Lemma 1). The claimed vector z can be described by linear constraints: indeed, consider the following LP on the variables z where constraints (5) imply that $\kappa(z)$ is in the convex hull of spanning connected graphs [15, Corollary 50.8a].[1]

$$\kappa(z)(\partial(\pi)) \ge |\pi| - 1 \qquad \forall \text{ partitions } \pi \text{ of } V \tag{5}$$

$$z_a \le \tfrac{1}{1-3\tau}\, x^*_a \qquad \forall\ a \in A \tag{6}$$

$$z(\partial^+(U_i)) = 1 \qquad \forall\ \tau\text{-narrow } s\text{-}t \text{ cuts } U_i \tag{7}$$

$$z(\partial^-(U_i)) = 0 \qquad \forall\ \tau\text{-narrow } s\text{-}t \text{ cuts } U_i \tag{8}$$

$$z_a \ge 0 \qquad \forall\ a \in A \tag{9}$$

We demonstrate a feasible z as follows.

$$z_a = \begin{cases} \frac{x^*_a}{x^*(\partial(L_i; L_{i+1}))} & \text{if } a \in \partial(L_i; L_{i+1}) \text{ for some } i; \\ \frac{x^*_a}{1-2\tau} & \text{if } a \in E[L_i] \text{ for some } i; \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

We claim that this solution z satisfies the above constraints. Constraints (8) and (9) are satisfied by construction. Constraint (6) follows from Lemma 3 for edges in $\partial(L_i; L_{i+1})$ and by construction for rest of the edges. For constraint (7), note that

$$z(\partial^+(U_i)) = z(\partial(L_i; L_{i+1})) + z(\partial^+(U_i) \setminus \partial(L_i; L_{i+1})) = \frac{x^*(\partial(L_i; L_{i+1}))}{x^*(\partial(L_i; L_{i+1}))} + 0 = 1.$$

To complete the proof, we now show constraints (5) holds. It suffices to show that $\kappa(z)$ can be decomposed as a convex combination of characteristic vectors of

[1] The statement of Lemma 1 makes a claim about $\kappa(z)$ being in the convex hull of spanning *trees* and not spanning *connected graphs*. However, the equivalent statement for spanning trees will follow by dropping some edges from the connected subgraphs in the decomposition of z to get spanning trees. Constraints (7) and (8) will still be satisfied by y since we retain connectivity.

connected graphs. For $1 \leq i \leq k+1$, let z^i denote the restriction of $\kappa(z)$ to edges whose both endpoints are contained in L_i. Then Corollary 1, constraints (9), and [15, Corollary 50.8a] imply that z^i can be decomposed as a convex combination of integral vectors, each of which corresponds to an edge set that is connected on L_i. Next, let z' denote the restriction of $\kappa(z)$ to edges whose both endpoints are contained in some common L_i for some i. Since the sets $E(L_1), \ldots, E(L_{k+1})$ are disjoint, we have that $z' = \sum_i z^i$ (where the addition is component-wise). Furthermore, z', being the sum of the z^i vectors, can be decomposed as a convex combination of integral vectors corresponding to edge sets E' such that the connected components of the graph $H' = (V, E')$ are precisely the sets $\{L_i\}_{i=1}^{k+1}$.

Next, let z'' denote the restriction of $\kappa(z)$ to edges contained in one such $\partial(L_i; L_{i+1})$. We also note that the sets $\partial(L_1; L_2), \ldots, \partial(L_k; L_{k+1})$ are disjoint. By construction, we have $z''(\partial(L_i; L_{i+1})) = 1$ for each $1 \leq i \leq k$ so we may decompose z'' as a convex-combination of integral vectors, each of which includes precisely one edge across each $\partial(L_i; L_{i+1})$.

Now, adding any integral point y' in the decomposition of z' to any integral point y'' in the decomposition of z'' results in an integral vector that corresponds to a connected graph: each L_i is connected by y' and consecutive L_i are connected by y''. By construction of z, we have $\kappa(z) = z' + z''$ so we may write z as a convex combination of characteristic vectors of connected graphs, each of which satisfies constraints (5).

To see why z can be found efficiently, we first compute all τ-narrow cuts using Lemma 5. Then z is easy to compute in equation 10. Finally, [15, Corollary 51.6a] implies the decomposition of $\kappa(z)$ into a convex combination of connected graphs can be done efficiently. Thus the arguments in the footnote to reduce z such that $\kappa(z)$ is in the spanning tree polytope can be implemented efficiently.

3 Obtaining an *s-t* Path

Having transformed the optimal LP solution x^* into the new vector z (as in Lemma 1) without increasing it too much in any coordinate, we now sample a random tree such that it has a small total cost, and that the tree does not cross any cut much more than prescribed by x^*. Finally we add some arcs to this tree (without increasing its cost much) so that it is Eulerian at all nodes except $\{s, t\}$, and hence gives us an Eulerian *s-t* walk. By the triangle inequality, shortcutting this walk past repeated nodes yields a Hamiltonian $s-t$ path of no greater cost. While this general approach is similar to that used in [2], some new ideas are required because we are working with the LP for ATSPP—in particular, only one unit of flow is guaranteed to cross *s-t* cuts, which is why we needed to deal with narrow cuts in the first place. The details appear in the rest of this section.

3.1 Sampling a Tree

For a collection of arcs $\mathcal{A} \subset A$, we say $\mathcal{A}$ is *α-thin with respect to x^** if $|\mathcal{A} \cap \partial^+(U)| \leq \alpha x^*(\partial^+(U))$ for every $\emptyset \subsetneq U \subsetneq V$. The set $\mathcal{A}$ is also *β-approximate*

with respect to x^* if the total cost of all arcs in $\mathcal{A}$ is at most β times the cost of x^*—i.e., $\sum_{a\in\mathcal{A}} c_a \le \beta \sum_{a\in A} c_a x^*_a$. The reason we are deviating from the undirected to the directed setting is that the orientation of the arcs across each τ-narrow cut will be important when we sample a random "tree".

Lemma 6. *Let* $\tau \in [0, 1/4]$. *Let* $\beta = \frac{3}{1-3\tau}$ *and* $\alpha = \Theta(\frac{\log n}{\tau \log\log n})$. *There is a randomized, polynomial time algorithm that, with probability at least 1/2, finds an* α*-thin and* β*-approximate (with respect to* x^**) collection of arcs* $\mathcal{A}$ *that is weakly connected and satisfies* $|\mathcal{A}\cap(\partial^+(U))| = 1$ *and* $|\mathcal{A}\cap(\partial^-(U))| = 0$ *for each* τ*-narrow* s*-*t *cut* U.

Proof. Let z be a vector as promised by Lemma 1. From $\kappa(z)$, randomly sample a set of arcs $\mathcal{A}$ whose undirected version $\mathcal{T}$ is a spanning tree on V. This should be done from any distribution with the following two properties:

(i) *(Correct Marginals)* $\Pr[e \in \mathcal{T}] = \kappa(z)_e$
(ii) *(Negative Correlation)* For any subset of edges $F \subseteq E$, $\Pr[F \subseteq \mathcal{T}] \le \prod_{e\in F} \Pr[e \in \mathcal{T}]$

This can be obtained using, for example, the swap rounding approach for the spanning tree polytope given by Chekuri et al. [7]. As in [2], the negative correlation property implies the following theorem.

Theorem 2. *The tree* $\mathcal{T}$ *is* α*-thin with high probability.*

By Lemma 1(b), property (i) of the random sampling, and Markov's inequality, we get that $\mathcal{A}$ (from Lemma 6) is $\frac{3}{1-3\tau}$-approximate with respect to x^* with probability at least 2/3. By a trivial union bound, for large enough n we have with probability at least 1/2 that $\mathcal{A}$ is both α-thin and β-approximate with respect to x^*. It is also weakly connected—i.e., the undirected version of $\mathcal{A}$ (namely, $\mathcal{T}$) connects all vertices in V.

The statement for τ-narrow s-t cuts follows from the fact that z satisfies Lemma 1(c). That is, $\mathcal{A}$ contains no arcs of $\partial^-(U)$, since $z(\partial^-(U)) = 0$ (for U being a τ-narrow s-t cut). But since $\mathcal{T}$ is a spanning tree, $\mathcal{A}$ must contain at least one arc from $\partial^+(U)$. Finally, since $z(\partial^+(U))$ is exactly 1, then any set of arcs supported by this distribution we use must have precisely one arc from $\partial^+(U)$.

3.2 Augmenting to an Eulerian s-t Walk

Finally, we wrap up by augmenting the set of arcs $\mathcal{A}$ to an Eulerian s-t walk. For this, we use Hoffman's circulation theorem, as in [2], which we recall here for convenience (see, e.g, [15, Theorem 11.2]):

Theorem 3. *Given a directed flow network* $D = (V, A)$, *with each arc having a lower bound* ℓ_a *and an upper bound* u_a *(and* $0 \le \ell_a \le u_a$*), there exists a circulation* $f : A \to \mathbb{R}_+$ *satisfying* $\ell_a \le f(a) \le u_a$ *for all arcs* a *if and only if* $\ell(\partial^+(U)) \le u(\partial^-(U))$ *for all* $U \subseteq V$. *Moreover, if the* ℓ *and* u *are integral, then the circulation* f *can be taken to be integral.*

Set lower bounds $\ell : A \to \{0, 1\}$ on the arcs by:

$$\ell_a = \begin{cases} 1 \text{ if } a \in \mathcal{A} \text{ or } a = ts \\ 0 \text{ otherwise} \end{cases}$$

For now, we set an upper bound of 1 on arc ts and leave all other arc upper bounds at ∞. We compute the minimum cost circulation satisfying these bounds (we will soon see why one must exist). Since the bounds are integral and since $\mathcal{A}$ is weakly connected, this circulation gives us a directed Eulerian graph. Furthermore, since $u_{ta} = \ell_{ta} = 1$, the ts arc must appear exactly once in this Eulerian graph. Our final Hamiltonian s-t path is obtained by following an Eulerian circuit, removing the single ts arc from this circuit to get an Eulerian s-t walk, and finally shortcutting this walk past repeated nodes. The cost of this Hamiltonian path will be, by the triangle inequality, at most the cost of the circulation minus the cost of the ts arc.

Finally, we need to bound the cost of the circulation (and also to prove one exists). To this end, we will impose further upper bounds $u : A \to \mathbb{R}_{\geq 0}$ as follows:

$$u_a = \begin{cases} 1 & \text{if } a = ts \\ 1 + (1 + \tau^{-1})\alpha x^*_a & \text{if } a \in \mathcal{A} \\ (1 + \tau^{-1})\alpha x^*_a & \text{otherwise} \end{cases}$$

We use Hoffman's circulation theorem to show that a circulation f exists satisfying these bounds ℓ and u (The calculations appear in the next paragraph.) Since u is no longer integral, the circulation f might not be integral, but it does demonstrate that a circulation exists where each arc $a \neq ts$ is assigned at most $(1 + \tau^{-1})\alpha x^* a$ more flow in the circulation than the number of times it appears in $\mathcal{A}$. Consequently, it shows that the minimum cost circulation g in the setting where we only had a non-trivial upper bound of 1 on the arc ts can be no more expensive (since there are fewer constraints), and that circulation g can be chosen to be integral. The cost of circulation g is at most the cost of f, which is at most

$$\sum_{a \in A} c_a u_a = \sum_{a \in \mathcal{A}} c_a + (1 + \tau^{-1})\alpha \sum_{a \in A} c_a x^*_a + c_{ts}.$$

Subtracting the cost of the ts arc (since we drop it to get the Hamilton path) and recalling that $\mathcal{A}$ is $\frac{3}{1-3\tau}$-approximate with respect to x^* (and hence $\sum_{a \in \mathcal{A}} c_a \leq \frac{3}{1-3\tau} \sum_{a \in A} c_a x^*_a$, we get that the final Hamiltonian path has cost at most

$$\left(\frac{3}{1 - 3\tau} + (1 + \tau^{-1})\alpha \right) \sum_{a \in A} c_a x^*_a,$$

and hence $O(\frac{\log n}{\log \log n})$ times the cost of the LP relaxation for $\tau = 1/4$. This proves the claim that the cost of the s-t path we found is $O(\frac{\log n}{\log \log n})$ times the LP value, with constant probability, and completes the proof of Theorem 1.

One detail remains: we need to verify the conditions of Theorem 3 for the bounds ℓ and u. Firstly, it is clear by definition that $\ell_a \leq u_a$ for each arc a. Now we need to check $\ell(\partial^+(U)) \leq u(\partial^-(U))$ for each cut U. This is broken into four cases (where saying U is a u-v cut means $u \in U, v \notin U$).

1. U is a τ-narrow s-t cut. Then $\ell(\partial^+(U)) = 1$, since $\mathcal{A}$ contains only one arc in $\partial^+(U)$. But $1 = u_{ts} \leq u(\partial^-(U))$.
2. U is an s-t cut, but not τ-narrow. Then by the α-thinness of $\mathcal{A}$,

$$\ell(\partial^+(U)) \leq \alpha x^*(\partial^+(U)) = \alpha x^*(\partial^-(U)) + \alpha.$$

   On the other hand,

$$u(\partial^-(U)) \geq (1+\tau^{-1})\alpha x^*(\partial^-(U)) = \alpha x^*(\partial^-(U)) + \tau^{-1}\alpha x^*(\partial^-(U)) \geq \alpha x^*(\partial^-(U)) + \alpha$$

   where the last inequality used the fact that $x^*(\partial^-(U)) \geq \tau$.
3. U is a t-s cut. Then

$$\ell(\partial^+(U)) \leq 1 + \alpha x^*(\partial^+(U)) = 1 + \alpha x^*(\partial^-(U)) - \alpha \leq \alpha x^*(\partial^-(U)),$$

   the last inequality using that $\alpha \geq 1$. Moreover

$$u(\partial^-(U)) \geq (1+\tau^{-1})\alpha x^*(\partial^-(U)) \geq \alpha x^*(\partial^-(U)).$$

   Then $\ell(\partial^+(U)) \leq u(\partial^-(U))$.
4. U does not separate s from t. Then

$$\ell(\partial^+(U)) \leq \alpha x^*(\partial^+(U)) = \alpha x^*(\partial^-(U)) \leq (1+\tau^{-1})\alpha x^*(\partial^-(U)) \leq u(\partial^-(U))$$

4 A Simple Integrality Gap Example

In this section, we show that the integrality gap of the subtour elimination LP *ATSPP* is at least 2. This result can also be inferred from the integrality gap of 2 for the ATSP tour problem [5], but our construction is relatively simpler.

For a fixed integer $r \geq 1$, consider the directed graph G_r defined below (and illustrated in Figure 1). The vertices of G_r are $\{s,t\} \cup \{u_1,\ldots,u_r\} \cup \{v_1,\ldots,v_r\}$; the edges are as follows:

- $\{su_1, sv_1, u_rt, v_rt\}$, each with cost 1,
- $\{u_1v_r, v_1u_r\}$, each with cost 0,
- $\{u_{i+1}u_i \mid 1 \leq i < r\} \cup \{v_{i+1}v_i \mid 1 \leq i < r\}$, each with cost 1,
- and $\{u_iu_{i+1} \mid 1 \leq i < r\} \cup \{v_iv_{i+1} \mid 1 \leq i < r\}$, each with cost 0.

Let F_r denote the ATSPP instance obtained from the metric completion of G_r.

Lemma 7. *The integrality gap of the LP* ATSPP *on the instance F_r is at least $2 - o(1)$.*

Proof (sketch). The assignment $x^*_a = \frac{1}{2}$ for every edge a of F_r corresponding to an edge of G_r is feasible for LP *ATSPP* with cost $k+1$. However, any spanning $s-t$ walk in G_r has length at least $2k - O(1)$, so the optimum ATSPP solution in F_r also has cost at least $2k - O(1)$.

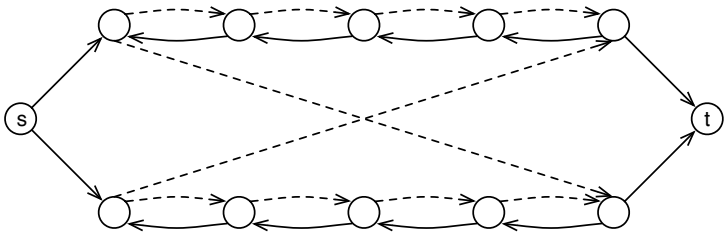


Fig. 1. The graph G_r with $r = 5$. The solid edges have cost 1 and the dashed edges have cost 0.

5 Conclusion

In this paper we showed that the integrality gap for the ATSPP problem is $O(\frac{\log n}{\log \log n})$. In fact, our proof also shows an integrality gap of α for ATSPP whenever we can construct a procedure which takes a point $y \in \mathbb{R}^{|E|}$ in the spanning tree polytope of an undirected (multi)graph $H = (V, E)$ and outputs a tree T that is (a) α-thin, and (b) also satisfies $|T \cap \partial(U)| = 1$ for any cut U where $y(\partial(U)) = 1$. We also showed a simpler construction achieving a lower bound of 2 for the subtour elimination LP.

Acknowledgments. We thank V. Nagarajan for enlightening discussions in the early stages of this project. Z.F. and A.G. also thank A. Vetta and M. Singh for their generous hospitality.

References

1. An, H.-C., Kleinberg, R.D., Shmoys, D.B.: Improving Christofides' algorithm for the s-t path TSP. In: Proc. 44th ACM Symp. on Theory of Computing (2012)
2. Asadpour, A., Goemans, M.X., Mądry, A., Oveis Gharan, S., Saberi. A.: An $O(\log n / \log \log n)$-approximation algorithm for the asymmetric traveling salesman problem. In: Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 379–389. SIAM, Philadelphia (2010)
3. Bateni, M., Chuzhoy, J.: Approximation Algorithms for the Directed k-Tour and k-Stroll Problems. In: Serna, M., Shaltiel, R., Jansen, K., Rolim, J. (eds.) APPROX and RANDOM 2010. LNCS, vol. 6302, pp. 25–38. Springer, Heidelberg (2010)
4. Bläser, M.: A new approximation algorithm for the asymmetric TSP with triangle inequality. ACM Trans. Algorithms 4(4), Art. 47, 15 (2008)
5. Charikar, M., Goemans, M.X., Karloff, H.: On the integrality ratio for the asymmetric traveling salesman problem. Math. Oper. Res. 31(2), 245–252 (2006)
6. Chekuri, C., Pál, M.: An $O(\log n)$ approximation ratio for the asymmetric traveling salesman path problem. Theory Comput. 3, 197–209 (2007)
7. Chekuri, C., Vondrák, J., Zenklusen, R.: Dependent randomized rounding via exchange properties of combinatorial structures. In: FOCS, pp. 575–584 (2010)

8. Feige, U., Singh, M.: Improved Approximation Ratios for Traveling Salesperson Tours and Paths in Directed Graphs. In: Charikar, M., Jansen, K., Reingold, O., Rolim, J.D.P. (eds.) APPROX and RANDOM 2007. LNCS, vol. 4627, pp. 104–118. Springer, Heidelberg (2007)
9. Frieze, A.M., Galbiati, G., Maffioli, F.: On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. Networks 12(1), 23–39 (1982)
10. Friggstad, Z., Salavatipour, M.R., Svitkina, Z.: Asymmetric traveling salesman path and directed latency problems. In: Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 419–428. SIAM, Philadelphia (2010)
11. Kaplan, H., Lewenstein, M., Shafrir, N., Sviridenko, M.: Approximation algorithms for asymmetric TSP by decomposing directed regular multigraphs. J. ACM 52(4), 602–626 (2005)
12. Lam, F., Newman, A.: Traveling salesman path problems. Math. Program. 113(1, Ser. A), 39–59 (2008)
13. Nagarajan, V., Ravi, R.: Poly-logarithmic Approximation Algorithms for Directed Vehicle Routing Problems. In: Charikar, M., Jansen, K., Reingold, O., Rolim, J.D.P. (eds.) APPROX and RANDOM 2007. LNCS, vol. 4627, pp. 257–270. Springer, Heidelberg (2007)
14. Nagarajan, V., Ravi, R.: The Directed Minimum Latency Problem. In: Goel, A., Jansen, K., Rolim, J.D.P., Rubinfeld, R. (eds.) APPROX and RANDOM 2008. LNCS, vol. 5171, pp. 193–206. Springer, Heidelberg (2008)
15. Schrijver, A.: Combinatorial optimization. Polyhedra and efficiency. Algorithms and Combinatorics, vol. 24. Springer, Berlin (2003)
16. Sebő, A.: Eight-Fifth Approximation for the Path TSP. In: Goemans, M., Correa, J. (eds.) IPCO 2013. LNCS, vol. 7801, pp. 362–374. Springer, Heidelberg (2013)
17. Sebő, A., Vygen, J.: Shorter tours by nicer ears: 7/5-approximation for graphic TSP, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs. CoRR, abs/1201.1870 (2012)

Single Commodity-Flow Algorithms for Lifts of Graphic and Co-graphic Matroids

Bertrand Guenin* and Leanne Stuive**

Dept. of Combinatorics & Optimization
Faculty of Mathematics
University of Waterloo, Canada

Abstract. Consider a binary matroid M given by its matrix representation. We show that if M is a lift of a graphic or a co-graphic matroid, then in polynomial time we can either solve the single commodity flow problem for M or find an obstruction for which the Max-Flow Min-Cut relation does not hold. The key tool is an algorithmic version of Lehman's Theorem for the set covering polyhedron.

1 Introduction

Let M be a binary matroid on ground set E. We follow [16] for matroid notation and terminology. Consider an element $e \in E$. An *e-path* is a set of the form $C - e$ where C is a circuit of M containing e. [1] An *e-cut* is a minimal subset B of $E - e$ that intersects every e-path; that is, $B \cap P \neq \emptyset$ for all e-paths P and if there exists $B' \subseteq B$ such that $B' \cap P \neq \emptyset$ for all e-paths P then $B' = B$.

Let $w : E \to \mathbb{Z}_+$ be a weight function on the ground set E of M. Given M, e and w, consider the following primal-dual pair of linear programs.

$$\begin{aligned} \min \quad & \sum (w_f x_f : f \in E - e) \\ \text{subject to} \quad & \sum (x_f : f \in P) \geq 1 \qquad \text{for all } e\text{-paths } P \\ & x \geq \mathbb{0} \ , \end{aligned} \tag{1}$$

$$\begin{aligned} \max \quad & \sum (y_P : P \text{ is an } e\text{-path}) \\ \text{subject to} \quad & \sum (y_P : f \in e\text{-path } P) \leq w_f \qquad \text{for all } f \in E - e \\ & y \geq \mathbb{0} \ . \end{aligned} \tag{2}$$

We say that the pair (M, e) has the *fractional MFMC* (Max-Flow Min-Cut) property if for every $w : E \to \mathbb{Z}_+$ there exists an integer solution to (1) and

* Supported by a Discovery Grant from NSERC and ONR grant N00014-12-1-0049.
** Supported by an NSERC graduate scholarship.
[1] Here $-$ indicates set difference and $A - a$ denotes $A - \{a\}$.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 193–204, 2013.

a solution to (2) that have the same objective value. Similarly, we say that (M, e) has the *integer MFMC* property if for every $w : E \to \mathbb{Z}_+$ there exists an integer solution to (1) and an integer solution to (2) that have the same objective value. (Note that to keep terminology consistent throughout this paper, our terminology differs from that introduced in [20].)

Consider a graph G with distinct vertices s, t and an edge $e = (s, t)$. Let H be obtained from G by deleting e. Suppose M is the graphic matroid corresponding to G; i.e., the circuits of M correspond to the circuits of G. Then e-paths and e-cuts of M correspond to respectively st-paths and st-cuts of H. An integer solution to (2) gives an integer st-flow, and an integer solution to (1) gives the characteristic vector of an st-cut. By the MFMC Theorem of Ford Fulkerson [4], the value of a maximum st-flow is equal to the size of a minimum st-cut; hence (M, e) has the integer MFMC property. Suppose M is the co-graphic matroid corresponding to G; i.e., the circuits of M correspond to the bonds (minimal cuts) of G. Then e-paths and e-cuts of M correspond to respectively st-cuts and st-paths of H. The Max Work Min Potential Theorem of Duffin [3] states that the size of a maximum packing of st-cuts is equal to the minimum length of an st-path. It readily follows that (M, e) has the integer MFMC property in this case as well.

1.1 From Matroids to clutters

In this section we express the MFMC property in the terminology of clutters. Given a ground set of elements $E = E(\mathcal{F})$, a clutter $\mathcal{F}$ is a finite family of sets of E such that no set in $\mathcal{F}$ contains or is equal to some other set in $\mathcal{F}$. A set $B \subseteq E$ is a *cover* of $\mathcal{F}$ if $B \cap S \neq \emptyset$ for all $S \in \mathcal{F}$. The set of all inclusion-wise minimal covers of $\mathcal{F}$ forms a clutter $b(\mathcal{F})$ called the *blocker* of $\mathcal{F}$. As $b(b(\mathcal{F})) = \mathcal{F}$ [2], we call a pair $\mathcal{F}, \mathcal{K}$ of clutters a *blocking pair* if $\mathcal{K} = b(\mathcal{F})$. A clutter $\mathcal{F}$ is *binary* if for all $S_1, S_2, S_3 \in \mathcal{F}$ there exists $S \in \mathcal{F}$ such that $S \subseteq S_1 \triangle S_2 \triangle S_3$ [2]. Equivalently, $\mathcal{F}$ is binary if for all $S \in \mathcal{F}$ and $T \in b(\mathcal{F})$, $|S \cap T|$ is odd [2].

The following result relates e-paths, e-cuts, and binary clutters [12].

Proposition 1. *Let M be a binary matroid and let $\in E(M)$.*

1. *The set of e-paths is a binary clutter.*
2. *The set of e-cuts is a binary clutter.*
3. *The clutter of e-paths and the clutter of e-cuts form a blocking pair.*

Moreover, every binary clutter is the set of e-paths for some binary matroid M and some $e \in E(M)$ [12]. Consider a clutter $\mathcal{F}$ and let $w : E \to \mathbb{Z}_+$ be a weight function on the ground set E of $\mathcal{F}$. Given $\mathcal{F}$ and w, consider the following primal-dual pair of linear programs.

[2] $A \triangle B = (A \cup B) - (A \cap B)$.

$$\begin{aligned} \min \quad & \sum(w_f x_f : f \in E) \\ \text{subject to} \quad & \sum(x_f : f \in S) \geq 1 \qquad \text{for all } S \in \mathcal{F} \\ & x \geq \mathbb{O} \end{aligned} \tag{3}$$

$$\begin{aligned} \max \quad & \sum(y_S : S \in \mathcal{F}) \\ \text{subject to} \quad & \sum(y_S : f \in S) \leq w_f \qquad \text{for all } f \in E \\ & y \geq \mathbb{O} \ . \end{aligned} \tag{4}$$

We say that $\mathcal{F}$ has the fractional MFMC property if for every $w : E \to \mathbb{Z}_+$ there exists an integer solution to (3) and a solution to (4) that have the same objective value. Similarly, we say that $\mathcal{F}$ has the integer MFMC property if for every $w : E \to \mathbb{Z}_+$ there exists an integer solution to (3) and an integer solution to (4) that have the same objective value. Observe that if $\mathcal{F}$ is the clutter of e-paths for some binary matroid M with $e \in E(M)$, then (1) is the same as (3) and (2) is the same as (4). In particular, (M, e) has the fractional (resp. integer) MFMC property whenever $\mathcal{F}$ does.

Considering a clutter $\mathcal{F}$ and $f \in E(\mathcal{F})$, we define $\mathcal{F} \setminus f$ as $\{S \in \mathcal{F} : f \notin S\}$, and $\mathcal{F}/f$ as the set of inclusion-wise minimal sets in $\{S - f : S \in \mathcal{F}\}$. We say that $\mathcal{F} \setminus f$ (resp. $\mathcal{F}/f$) is obtained from $\mathcal{F}$ by *deleting* (resp. *contracting* f). A minor of $\mathcal{F}$ is any clutter $\mathcal{F}'$ obtained by a sequence of deletions and contractions. As deletions and contractions associate, we can write $\mathcal{F}' = \mathcal{F}/I \setminus J$ to indicate that the elements of $I \subseteq E(\mathcal{F})$ were deleted and the elements of $J \subseteq E(\mathcal{F})$ were contracted to obtain $\mathcal{F}'$ from $\mathcal{F}$. It can be readily checked that the fractional (resp. integer) MFMC property is closed under taking minors.

1.2 The Integer MFMC Property

The following seminal result of Seymour [19] characterizes the binary clutters with the integer MFMC property.

Theorem 1. *A binary clutter has the integer MFMC property if and only if it does not have a minor that is isomorphic* [3] *to*

$$\mathcal{Q}_6 = \big\{\{1,2,4\},\{1,3,5\},\{2,3,6\},\{4,5,6\}\big\} \ .$$

Consider a binary clutter $\mathcal{F}$ and $w : E \to \mathbb{Z}_+$. In light of the previous result, it is not always possible to find an integer solution to (3) and an integer solution to (4) with the same objective value. It is natural to ask if there exists an efficient algorithm that, given a binary clutter, will either find such a pair of solutions or find a $\mathcal{Q}_6$ minor. Using techniques from structural matroid theory, Truemper [21] proved that such an algorithm exists.

[3] Two clutters are isomorphic to one another if one can be obtained from the other by relabelling elements in the ground set.

Theorem 2. *Let M be a binary matroid represented by a $0,1$ matrix A and let $e \in E(M)$. Let $\mathcal{F}$ be the clutter of e-paths of M and let $w : E(\mathcal{F}) \to \mathbb{Z}_+$. Then in time polynomial in $\langle A \rangle$ and $\langle w \rangle$, one can either find*

1. *$I, J \subseteq E(\mathcal{F})$ such that $\mathcal{F}/I \setminus J$ is isomorphic to $\mathcal{Q}_6$, or*
2. *an integer solution to (3) and an integer solution to (4) with the same value.*

Recall that the encoding length of a rational number $\alpha = \frac{p}{q}$, is the total number of bits needed to represent both p and q in base two. The encoding length $\langle v \rangle$ of a rational vector v is the total encoding length of all entries of v.

1.3 The Fractional MFMC Property

The following conjecture of Seymour [19] would characterize the binary clutters with the fractional MFMC property.

Conjecture 1. A binary clutter has the fractional MFMC property if and only if it does not have any of the following clutters as a minor: $\mathcal{O}_{K_5}$, $b(\mathcal{O}_{K_5})$, or $\mathcal{L}_7$.

The clutter $\mathcal{O}_{K_5}$ has ground set corresponding to the edges of the graph K_5 and sets corresponding to each of the circuits of K_5 with an odd number of edges. The clutter $\mathcal{L}_7$ corresponds to the lines of the Fano matroid; i.e.,

$$\mathcal{L}_7 = \big\{\{1,2,3\},\{1,4,5\},\{1,6,7\},\{3,5,7\},\{2,4,7\},\{2,5,6\},\{3,4,6\}\big\} \ .$$

1.4 Lifts of Graphic and Co-graphic Matroids

Consider a binary matroid M represented by a $0,1$ matrix A; i.e., a set of elements of M is dependent if and only if the corresponding columns of A are dependent over GF(2). Let A' be obtained from A by adding some $0,1$ row outside the row space of A. The binary matroid M' with representation A' is called a *lift* of M [7]. (Lifts of graphic and co-graphic matroids are also known as even-cycle and even-cut matroids [17]; they were introduced in [22].)

Let M be a binary matroid and let $e \in E(M)$. We observed that the pair (M, e) has the integer MFMC property if M is either graphic or co-graphic. The following theorem proves Conjecture 1 for lifts of graphic and co-graphic matroids [12].

Theorem 3. *Let M be a binary matroid, let $e \in E(M)$, and let $\mathcal{F}$ be the clutter of e-paths of M.*

1. *If M is a lift of a graphic matroid, then $\mathcal{F}$ has the fractional MFMC property if and only if it does not have a minor isomorphic to $\mathcal{O}_{K_5}$ or $\mathcal{L}_7$.*
2. *If M is a lift of a co-graphic matroid, then $\mathcal{F}$ has the fractional MFMC property if and only if it does not have a minor isomorphic to $b(\mathcal{O}_{K_5})$ or $\mathcal{L}_7$.*

1.5 The Main Results

The goal of this paper is to prove the fractional analogue of Theorem 2 for lifts of graphic or co-graphic matroids. Before we can formalize our main results, we describe the clutter of e-paths for these classes of matroids.

A signed graph is a pair (G, Σ) where G is a graph and $\Sigma \subseteq E(G)$. We say $B \subseteq E$ is odd (resp. even) if $|B \cap \Sigma|$ is odd (resp. even). In particular, edges in Σ are odd. The set of odd circuits of (G, Σ) is the same as the set of odd circuits of (G, Σ') whenever $\Sigma' = \Sigma \triangle \delta(S)$ for $S \subseteq V$; we call $B \subseteq E$ a *signature* of (G, Σ) if $B = \Sigma \triangle \delta(S)$ for $S \subseteq V$. Given a signed graph (G, Σ), and $s, t \in V(G)$, $L \subseteq E$ is an *odd-st-walk* if it is either an odd st-path or the union of an even st-path P and an odd circuit C where P and C share at most one vertex.

A graft is a pair (G, T) where G is a graph and $T \subseteq V(G)$ such is that $|T|$ even. A T-join is an inclusion-wise minimal set of edges J such that T is the set of vertices of odd degree in $G[J]$. A T-cut is a set of edges $\delta(U)$ where U satisfies $|U \cap T|$ odd. An *st-T-cut* is a T-cut $\delta(U)$ where $s \in U$, $t \in V(G) - U$.

The following result appears in [12].

Proposition 2. *Consider a binary matroid M. Let $e \in E(M)$ and let $\mathcal{F}$ denote the clutter of e-paths of M.*

1. *If M is a lift of a graphic matroid then $\mathcal{F}$ is a clutter of odd st-walks.*
2. *If M is a lift of a co-graphic matroid then $\mathcal{F}$ is a clutter of st-T-cuts.*

We are now ready to state the main results of the paper.

Theorem 4. *Let (G, Σ) be a signed graph and $w : E(G) \to \mathbb{Z}_+$. Let $s, t \in V(G)$ and $\mathcal{F}$ be the clutter of odd-st-walks in (G, Σ). Then in time polynomial in $|V(G)| + \langle w \rangle$, one can either find*

1. *$I, J \subseteq E(\mathcal{F})$ such that $\mathcal{F}/I \setminus J$ is isomorphic to $\mathcal{O}_{K_5}$ or $\mathcal{L}_7$, or*
2. *an integer solution to (3) and a solution to (4) with the same value.*

Theorem 5. *Let (G, T) be a graft and $w : E(G) \to \mathbb{Z}_+$. Let $s, t \in V(G)$ and $\mathcal{F}$ be the clutter of st-T-cuts in (G, T). Then in time polynomial in $|V(G)| + \langle w \rangle$, one can either find*

1. *$I, J \subseteq E(\mathcal{F})$ such that $\mathcal{F}/I \setminus J$ is isomorphic to $b(\mathcal{O}_{K_5})$ or $\mathcal{L}_7$, or*
2. *an integer solution to (3) and a solution to (4) with the same value.*

Thus, Theorem 4 (resp. 5) is the fractional analogue of Theorem 2 for lifts of graphic (resp. co-graphic) matroids. In the previous two theorems, the solution to (4) can, of course, be fractional.

2 Overview of the Proofs

In this section we give an outline of the proofs of Theorems 4 and 5.

2.1 Lehman's Theorem

Given a 0,1 matrix M we define the affiliated set covering polyhedron by

$$Q(M) := \{x \geq \mathbb{0} : Mx \geq \mathbb{1}\} \ .$$

We say that M is *ideal* if $Q(M)$ is integral; i.e., if every extreme point of $Q(M)$ is integral. Given a clutter $\mathcal{F}$, we let $M(\mathcal{F})$ denote the $0,1$ matrix with columns indexed by elements $e \in E(M)$ and rows indexed by sets $S \in \mathcal{F}$, where entry (S, e) is 1 if and only if $e \in S$. Observe that $M(\mathcal{F})$ is only defined up to permutations of the rows. We say clutter $\mathcal{F}$ is ideal if $M(\mathcal{F})$ is ideal, and write $Q(\mathcal{F})$ for $Q\big(M(\mathcal{F})\big)$. Note that $\mathcal{F}$ is ideal if and only if it has the fractional MFMC property. If $\mathcal{F}$ is ideal, then so is any minor of $\mathcal{F}$ [2]. We say that a clutter is *minimally non-ideal* (mni) if it is non-ideal but all of its proper minors are ideal. An example of a mni clutter is

$$\mathcal{J}_s = \big\{\{1 \dots s\}, \{0,1\}, \{0,2\}, \dots, \{0,s\}\big\}$$

for $s \geq 2$. Lehman gave the following characterization of mni clutters [15],

Theorem 6. *Let $\mathcal{F}$ be a minimally non-ideal clutter that is not isomorphic to $\mathcal{J}_s$ for $s \geq 2$. Let $\mathcal{K}$ be the blocker of $\mathcal{F}$. Denote by $\bar{\mathcal{F}}$ (resp. $\bar{\mathcal{K}}$) the clutter formed by the sets of minimum cardinality of $\mathcal{F}$ (resp. $\mathcal{K}$). Then*

1. *$M(\bar{\mathcal{F}})$ and $M(\bar{\mathcal{K}})$ are square matrices, and*
2. *after possibly rearranging rows of $M(\bar{\mathcal{F}})$ we have for some $d \geq 1$*

$$M(\bar{\mathcal{F}})M(\bar{\mathcal{K}})^T = J + dI \ .^4 \tag{5}$$

In the previous theorem (for $\mathcal{F}$ not isomorphic to $\mathcal{J}_s$) if r (resp. ℓ) denotes the cardinality of the sets of $\bar{\mathcal{F}}$ (resp. $\bar{\mathcal{K}}$), then $\frac{1}{r}\mathbb{1}$ (resp. $\frac{1}{\ell}\mathbb{1}$) is a fractional extreme point of $Q(\mathcal{F})$ (resp. $Q(\mathcal{K})$).

2.2 An Algorithmic Version

Let $\mathcal{F}, \mathcal{K}$ be a blocking pair and let $\bar{\mathcal{F}}$ (resp. $\bar{\mathcal{K}}$) denote the clutter formed by the sets of minimum cardinality of $\mathcal{F}$ (resp. $\mathcal{K}$). We say that $\mathcal{F}$ is a *Lehman clutter* when conditions (1) and (2) of Theorem 6 are satisfied. Let $P \subseteq \mathbb{R}^n$ be a polyhedron. The separation problem for P and a point $\bar{x} \in \mathbb{Q}^n$ is to either determine that $\bar{x} \in P$, or to find a separating constraint $a^T x \leq \beta$ (i.e., $a^T \bar{x} > \beta$ and $a^T x \leq \beta$ for all $x \in P$). A separation oracle for P is a function that solves the separation problem for any $\bar{x} \in \mathbb{Q}^n$.

We are now ready to state our algorithmic version of Theorem 6.

Theorem 7. *Let $\mathcal{F}$ be a clutter and suppose that $Q(\mathcal{F}) \subseteq \mathbb{R}^n$ is given by a separation oracle. Let x be a fractional extreme point of $Q(\mathcal{F})$ and suppose that we are given n facets of $Q(\mathcal{F})$ that define x. Then in oracle polynomial time of n we can find disjoint sets $I, J \subseteq E(\mathcal{F})$ such that for $\mathcal{F}' = \mathcal{F}/I \setminus J$ either*

[4] J is a square matrix of all 1's and I is identity matrix.

1. *$\mathcal{F}'$ is isomorphic to $\mathcal{J}_s$, or*
2. *$\mathcal{F}'$ is a Lehman clutter.*

Moreover, in case (2) we also find all the minimum cardinality sets of $\mathcal{F}'$.

Consider the decision problem: is a given clutter $\mathcal{F}$ ideal? It is clearly in Co-NP as it suffices to exhibit a fractional extreme point of $Q(\mathcal{F})$. Outcomes (1) and (2) of the previous theorem are also a Co-NP certificate. In other words, the result allows us to construct a highly regular Co-NP certificate from an arbitrary fractional extreme point (and defining facets).

2.3 Outline of the Proof of Theorems 4 and 5

The first step is to show that the separation problem for the odd-st-walk and st-T-cut set covering polyhedra are polynomially solvable. For the odd-st-walk polyhedron, the proof follows techniques of [10].

Remark 1. Let (G, Σ) be a signed graph and let $s, t \in V(G)$. Let $\mathcal{F}$ be the clutter of odd st-walks of (G, Σ). Given $\bar{x} \in \mathbb{Q}^{E(G)}$, we can solve the separation problem for $\bar{x} \in \mathbb{Q}^n$ and $Q(\mathcal{F})$ in time polynomial in $|V(G)| + \langle \bar{x} \rangle$.

Proof. We may assume that $\bar{x}_e \geq 0$ for every $e \in E(G)$ for otherwise $x_e \geq 0$ is a separating constraint. It suffices to show that is it possible to find, in polynomial time of $|V(G)| + \langle \bar{x} \rangle$, a minimum weight odd-$st$-walk L with edge weights given by $\bar{x}$. If $\bar{x}(L) \geq 1$ then $\bar{x} \in Q(\mathcal{F})$; otherwise $\sum_{e \in L} x_e \geq 1$ is the separating constraint. Find a minimum weight st-path P. If P is odd, let $L = P$. Otherwise, find a minimum weight odd st-path P_{odd} and a minimum weight odd circuit C_{odd}. If the weight of P_{odd} is smaller than the weight of $P \triangle C_{odd}$ then let $L = P_{odd}$ and otherwise let $L = P \triangle C_{odd}$. It can be readily checked that L is a minimum weight odd-st-walk. The result follows since the shortest st-path and minimum weight odd circuit (or path) problems are polynomially solvable [10]. □

For the st-T-cut polyhedron, a proof of the following is given in [8].

Remark 2. Let (G, T) be a graft and let $s, t \in V(G)$. Let $\mathcal{F}$ be the clutter of st-T-cuts of (G, T). Given any $x \in \mathbb{Q}^{E(G)}$, we can solve the separation problem for $\bar{x} \in \mathbb{Q}^n$ and $Q(\mathcal{F})$ in time polynomial in $|V(G)| + \langle \bar{x} \rangle$.

The proofs of the following two lemmas are constructive variants of structures found in [6] and [12].

Lemma 1. *Let (G, Σ) be a signed graph and let $s, t \in V(G)$. Let $\mathcal{F}$ be the clutter of odd st-walks of (G, Σ). Suppose that $\mathcal{F}$ is a Lehman clutter and that we are given the minimum cardinality sets of $\mathcal{F}$. Then in time polynomial in $|V(G)|$ we can find $I, J \subseteq E(\mathcal{F})$ such that $\mathcal{F}/I \setminus J$ is isomorphic to $\mathcal{O}_{K_5}$ or $\mathcal{L}_7$.*

Lemma 2. *Let (G, T) be a graft and let $s, t \in V(G)$. Let $\mathcal{F}$ be the clutter of st-T-cuts of G. Suppose that $\mathcal{F}$ is a Lehman clutter and that we are given the minimum cardinality sets of $\mathcal{F}$. Then in time polynomial in $|V(G)|$ we can find $I, J \subseteq E(\mathcal{F})$ such that $\mathcal{F}/I \setminus J$ is isomorphic to $b(\mathcal{O}_{K_5})$ or $\mathcal{L}_7$.*

The following is obtained by combining (6.5.9) and (6.5.15) in [9].

Proposition 3. *Let $\mathcal{F}$ be a clutter and suppose that $Q(\mathcal{F}) \subseteq \mathbb{R}^n$ is given by a separation oracle. Let $w : E(\mathcal{F}) \to \mathbb{Z}_+$. Then in oracle polynomial time in $n + \langle w \rangle$ we can find an optimal solution $\bar{y}$ to (4) and an extreme point $\bar{x}$ of $Q(\mathcal{F})$ that is optimal for (3) together with a set of n constraints of $Q(\mathcal{F})$ defining $\bar{x}$.*

Let (G, Σ) be a signed graph with $s, t \in V(G)$ and let I, J be disjoint subsets of $E(G)$ where I contains no odd circuit of (G, Σ). We denote by $(G, \Sigma)/I \setminus J$ a signed graph $(G/I \setminus J, \Sigma' - J)$ where $\Sigma' \cap I = \emptyset$ and $\Sigma' = \Sigma \triangle \delta_G(U)$ for some $U \subseteq V(G) - \{s, t\}$. Moreover, the vertex of $G/I \setminus J$ corresponding to the component of G induced by I that contains s (resp. t) is labeled s (resp. t).

Remark 3. If $\mathcal{F}$ is the clutter of odd st-walks of (G, Σ) then $\mathcal{F}/I \setminus J$ is the clutter of odd st-walks of $(G, \Sigma)/I \setminus J$.

Proof (Theorem 4). By Remark 1 and Proposition 3 we can, in time polynomial in $|V(G)| + \langle w \rangle$, find an optimal solution $\bar{y}$ to (4) and an extreme point $\bar{x}$ of $Q(\mathcal{F})$ that is optimal for (3) together with a set of n constraints of $Q(\mathcal{F})$ that define $\bar{x}$. We may assume $\bar{x}$ is fractional for otherwise we are done. Since $\mathcal{F}$ is binary, $\mathcal{F}$ is not isomorphic to $\mathcal{J}_s$. Hence, by Theorem 7 and Proposition 3 we find in time polynomial in $|V(G)| + \langle w \rangle$, sets $I, J \subseteq E(\mathcal{F})$ such that $\mathcal{K} = \mathcal{F}/I \setminus J$ is a Lehman clutter. We can also find the minimum cardinality sets $\bar{\mathcal{K}}$ of $\mathcal{K}$ and by Remark 3, $\mathcal{K}$ is the clutter of odd st-walks of $(G, \Sigma)/I \setminus J$. Thus we can apply Lemma 1 and find $I', J' \subseteq E(\mathcal{K})$ where $\mathcal{K} \setminus I'/J'$ is isomorphic to $\mathcal{O}_{K_5}$ or $\mathcal{L}_7$. □

Let (G, T) be a graft with $s, t \in V(G)$ and let I, J be disjoint subsets of $E(G)$ where J contains no odd bond of (G, T). We denote by $(G, T)/I \setminus J$ the graft $(G/I \setminus J, T')$ where $B - I$ is a T'-join of G for some T-join B of G where $B \cap J = \emptyset$. Moreover, the vertex of $G/I \setminus J$ corresponding to the component of G induced by I that contains s (resp. t) is labeled s (resp. t).

Remark 4. If $\mathcal{F}$ is the clutter of st-T-cuts of (G, T) then $\mathcal{F}/I \setminus J$ is the clutter of st-T-cuts of $(G, T) \setminus I/J$.

Proof (Theorem 5). Similar to the proof of Theorem 4. Replace Remark 1 by Remark 2; Lemma 1 by Lemma 2; Remark 3 by Remark 4; $(G, \Sigma)/I \setminus J$ by $(G, T) \setminus I/J$, and $\mathcal{O}_{K_5}$ by $b(\mathcal{O}_{K_5})$ □

2.4 Remarks

The problem of finding a fixed minor of a binary clutter is equivalent to the problem of finding a rooted minor in a binary matroid. It follows from recent development in the matroid minor project that this problem can be solved in polynomial time [5]. However, these algorithms are very complicated and not practical because they arise from Ramsey-type arguments and thus the constants are astronomical. Using these algorithms, however, it is possible to avoid using Lemmas 1, 2 and Theorem 7 in the proof of Theorems 4 and 5. Similarly, in the

next theorem we could rely on the graph minor testing algorithm with parity condition in [14].

A graph G contains a graph H as an *odd minor* if H can be obtained from G by contracting all edges on a cut, and then deleting a subset of the edges. If G contains H as an odd minor, then it contains H as a minor; however, the converse does not hold. It follows from [9] and [11] that if G does not contain K_5 as an odd minor then we can find a maximum cut in polynomial time. We can generalize the result as follows.

Theorem 8. *Let G be a graph and let $w : E(G) \to \mathbb{Z}_+$. Then in time polynomial in $|V(G)| + \langle w \rangle$ we can either*

1. *find a maximum weight cut of G, or*
2. *find K_5 as an odd minor of G.*

Proof. Let $\Sigma = E(G)$ and pick $s = t$ arbitrarily. If $\mathcal{F}$ is the clutter of odd-st-walks of G then it is, in fact, the clutter of odd circuits of G. By Remark 1 and Proposition 3 we can, in polynomial time, find an extreme point $\bar{x}$ of $Q(\mathcal{F})$ that is optimal for (3) together with a set of n constraints of $Q(\mathcal{F})$ that define $\bar{x}$. If $\bar{x}$ is integer, then we may assume that it is the characteristic vector of a set of edges B that intersects every odd circuit, i.e. $E(G) - B$ is the maximum cut. Otherwise, use Theorem 7 and Lemma 1 to find I, J such that $\mathcal{F}/I \setminus J$ is isomorphic to $\mathcal{O}_{K_5}$. It can be then readily checked that I forms a cut of G, and that K_5 is obtained by contracting all edges of I and deleting parallel edges. □

2.5 Organization of the Remainder of the Paper

In Section 3 we describe the algorithm in Theorem 7; that is, we show how to find a Lehman clutter efficiently. In this extended abstract we shall not give a complete proof of correctness (omitting the proofs of Lemmas 1 and 2).

3 Finding Lehman Clutters

3.1 The Algorithm

We first require a number of preliminaries. Throughout this section, $\mathcal{F}$ will always denote a clutter with $E(\mathcal{F}) = [n]$. [5] Given a clutter $\mathcal{F}$, we denote by $P(\mathcal{F})$ the polytope $Q(\mathcal{F}) \cap [0,1]^n$. It is well known that $P(\mathcal{F})$ is integral if and only if $Q(\mathcal{F})$ is integral. Consider a point $\bar{x} \in P(\mathcal{F})$ and let $j \in [n]$; we define

$$\bar{x}^j = (\bar{x}_1, \ldots, \bar{x}_{j-1}, 1, \bar{x}_{j+1}, \ldots, \bar{x}_n)^T, \quad \text{and}$$
$$F^j = P(\mathcal{F}) \cap \{x : x_j = 1\} \ .$$

We say that $\bar{x} \in P(\mathcal{F})$ is *special* if it is an extreme point and for all $j \in [n]$,

(S1) $0 < \bar{x}_j < 1$, and

[5] $[n] = \{1, \ldots, n\}$

(S2) $\bar{x}^j$ is a convex combination of integer extreme points of F^j.

We are now ready to state the main result upon which our algorithm relies. In this extended abstract we omit the proofs of Lemmas 4, 6 and 7.

Lemma 3. *Suppose $P(\mathcal{F})$ is given by a separation oracle. Given an extreme point $\bar{x}$ of $P(\mathcal{F})$ with n facets that define $\bar{x}$, in oracle polynomial time in n one can either*

1. *deduce that $\bar{x}$ is special, or*
2. *find $j \in [n]$ and a fractional extreme point x' of $P(\mathcal{F}/j)$, or*
3. *find $j \in [n]$ and a fractional extreme point x' of $P(\mathcal{F} \setminus j)$.*

Moreover, for (2) and (3) we also find $n-1$ facets that define x'.

The proof requires the following algorithmic version of Caratheodory's Theorem (see (6.5.11) [9]).

Proposition 4. *Let $P \subseteq \mathbb{R}^n$ be a well-described polytope given by a separation oracle and let $\bar{x} \in P \cap \mathbb{Q}^n$. There exists an oracle-polynomial algorithm that will express $\bar{x}$ as a convex combination of at most $\dim(P)+1$ extreme points of P. Moreover, for each of these points we can find the defining facets of P.*

Proof (Lemma 3). Suppose $\bar{x}_j \in \{0,1\}$ for some $j \in [n]$. Let $x' = (\bar{x}_1, \ldots, \bar{x}_{j-1}, \bar{x}_{j+1}, \ldots, \bar{x}_n)^T$. If $\bar{x}_j = 0$ (resp. $\bar{x}_j = 1$) then x' is an extreme point of $P(\mathcal{F}/j)$ (resp. $P(\mathcal{F}\backslash j)$) and the facets that define $\bar{x}$, omitting $x_j = 0$ (resp. $x_j = 1$) define x' and outcome (2) (resp. (3)) of the lemma occurs. Thus we may assume condition (S1) holds. For every $j \in [n]$, we use Proposition 4 to express $\bar{x}^j$ as a convex combination of extreme points $y^1, \ldots, y^k$ of F^j where $k \leq n$. Note, that as $\bar{x}$ is an extreme point of $P(\mathcal{F})$, it has encoding length $\langle \bar{x} \rangle$ polynomial in n [9](6.2.4). Suppose y^s is fractional for some $s \in [k]$, then $(y^s_1, \ldots, y^s_{j-1}, y^s_{j+1}, \ldots, y^s_n)^T$ is an extreme point $P(\mathcal{F} \setminus j)$ and outcome (3) occurs. If this never occurs, condition (S2) holds and $\bar{x}$ is special. □

Two extreme points in a polytope are *adjacent* if they are contained in a face of dimension 1. The set of all extreme points that are adjacent to extreme point $\bar{x}$ are the *neighbours* of $\bar{x}$. An extreme point $\bar{x}$ of a polytope in $\mathbb{R}^n$ is *non-degenerate* if there are exactly n facets satisfied at equality at $\bar{x}$ or, equivalently, if $\bar{x}$ has exactly n neighbours.

Lemma 4. *Special points of $P(\mathcal{F})$ are non-degenerate.*

Lemma 5. *Suppose we are given: a non-degenerate extreme point $\bar{x}$ of $P(\mathcal{F})$ where $0 < \bar{x}_j < 1$ for all $j \in [n]$, and n facets that define $\bar{x}$. If $P(\mathcal{F})$ is described by a membership oracle, then in oracle polynomial time of n one can find the n neighbours of $\bar{x}$.*

Proof (Sketch). As $\bar{x}$ is non-degenerate, exactly n constraints of $M(\mathcal{F})\bar{x} \geq \mathbb{1}$ are tight for $\bar{x}$. Find $d \neq \mathbb{0}$ satisfying $n-1$ of these constraints. Then a neighbour b of $\bar{x}$ is on the line $L = \{\bar{x} + \lambda d : \lambda \in \mathbb{R}\}$. Use the membership oracle to do a binary search to find $b \in L$. □

Lemma 6. *No two special points of $P(\mathcal{F})$ are adjacent.*

Note that in the next statement we know $\bar{x}$ is non-degenerate from Lemma 4.

Lemma 7. *Suppose that $\bar{x}$ is a special point of $P(\mathcal{F})$ and that all neighbours of $\bar{x}$ are integer. The facets that define $\bar{x}$ are of the form $\sum_{k \in S_i} x_k \geq 1$ where $S_i \in \mathcal{F}$ for $i = 1, \ldots, n$. Let $\bar{\mathcal{F}} = \{S_1, \ldots, S_n\}$; either*

1. *$\mathcal{F}$ is isomorphic to $\mathcal{J}_s$ and $\bar{\mathcal{F}} = \mathcal{F}$, or*
2. *$\mathcal{F}$ is a Lehman clutter and the elements of $\bar{\mathcal{F}}$ are the minimum cardinality sets of $\mathcal{F}$.*

We are now ready to describe the algorithm referenced in Theorem 7.

Let $\bar{x}$ be an extreme point of $P(\mathcal{F})$ and suppose that we are given n facets of $P(\mathcal{F})$ that define $\bar{x}$. We apply the algorithm referenced in Lemma 3. If outcome (2) or (3) occurs, then we apply the main algorithm recursively to $P(\mathcal{F}/j)$ or $P(\mathcal{F} \setminus j)$ respectively, with the new extreme point x' and the $n-1$ facets defining x'. (Note, that the separation oracle for $P(\mathcal{F})$ extends to a separation oracle for $P(\mathcal{F}/j)$ and $P(\mathcal{F} \setminus j)$, as contracting j correspond to setting $\bar{x}_j = 0$ and deleting j to setting $\bar{x}_j = 1$.) Otherwise outcome (1) of Lemma 3 occurs. Because of Lemma 4, $\bar{x}$ is non-degenerate. Using the algorithm referenced in Lemma 5 we can find its neighbours $b^1, \ldots, b^n$. Consider first the case where all of $b^1, \ldots, b^n$ are integer. Since we have the facets that define $\bar{x}$, we can construct $\bar{\mathcal{F}}$ as in Lemma 7. Then outcome (1) and (2) of Lemma 7 correspond to respectively outcomes (1) and (2) of Theorem 7 and we can stop. Thus we may assume that b^i is fractional for some $i \in [n]$. We apply the algorithm referenced in Lemma 3 to b^i. As b^i is not special (see Lemma 6), outcome (2) or (3) occurs, and we can again apply the main algorithm recursively.

Finally let us verify that the algorithm runs in oracle polynomial time. Note, that if $Q(\mathcal{F})$ is described by a separation oracle then so is $P(\mathcal{F})$ as it suffices to check in addition that $x_e \leq 1$ for all $e \in E(\mathcal{F})$. The claim follows from the fact that the algorithms referenced in Lemmas 3 and 5 run in oracle polynomial time of n, and that every time we call the algorithm recursively we decrease the dimension of the polytope considered by one.

References

1. Bridges, W.G., Ryser, H.J.: Combinatorial Designs and Related Systems. Journal of Algebra 13, 432–446 (1969)
2. Cornuéjols, G.: Combinatorial Optimization: Packing and covering. CBMS-NSF Regional Conference Series in Applied Mathematics, vol. 72 (2001)
3. Duffin, R.J.: The extremal length of a network. Journal of Mathematical Analysis and Applications 5(2), 200–215 (1962)
4. Ford, L.R., Fulkerson, D.R.: Maximal flow through a network. Canadian J. of Math. 8, 399–404 (1956)
5. Geelen, J.J.: Personal Communication
6. Geelen, J.F., Guenin, B.: Packing odd-circuits in Eulerian graphs. J. Comb. Theory Ser. B 86(2), 280–295 (2002)

7. Geelen, J.F., Gerards, A.M.H., Whittle, G.: Towards A Matroid-Minor Structure Theory. In: Combinatorics, Complexity, and Chance. A tribute to Dominic Welsh, pp. 72–82. Oxford University Press (2007)
8. Goemans, M.X., Ramakrishnan, V.S.: Minimizing Submodular Functions over Families of Sets. Combinatorica 15, 499–513 (1995)
9. Grötschel, M., Lovász, L., Schrijver, A.: Geometric Algorithms and Combinatorial Optimization. Springer (1988)
10. Grötschel, M., Pulleyblank, W.R.: Weakly bipartite graphs and the max-cut problem. Operations Research Letters 1(1), 23–27 (1981)
11. Guenin, B.: A characterization of weakly bipartite graphs. J. of Comb. Theory, Ser. B 83(1), 112–168 (2001)
12. Guenin, B.: Integral polyhedra related to even-cycle and even-cut matroids. Math. Oper. Res. 29(4), 693–710 (2002)
13. Guenin, B.: A short proof of Semour's max-flow min-cut theorem. J. Comb. Theory Ser. B 86(2), 273–279 (2002)
14. Kawarabayashi, K., Reed, B., Wollan, P.: The Graph Minor Algorithm with Parity Conditions. In: IEEE 52nd Annual Symposium Foundations of Computer Science, pp. 27–36 (2011)
15. Lehman, A.: On the width-length inequality and degenerate projective planes. In: Cook, W., Seymour, P.D. (eds.) Polyhedral Combinatorics. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 1, pp. 101–105. American Mathematical Society, Providence (1990)
16. Oxley, J.: Matroid theory, 2nd edn. Oxford Graduate Texts in Mathematics, vol. 21. Oxford University Press, Oxford (2011)
17. Pivotto, I.: Even-cycle and even-cut matroids. Ph.D thesis, University of Waterloo (2011)
18. Schrijver, A.: A short proof of Guenin's characterization of weakly bipartite graphs. Journal of Combinatorial Theory, Series B 85, 255–260 (2002)
19. Seymour, P.D.: The matroids with the Max-Flow Min-Cut property. J. Combin. Theory Ser. B 23, 189–222 (1977)
20. Seymour, P.D.: Matroids and multicommodity flows. European J. of Comb. 2 28, 257–290 (1981)
21. Truemper, K.: Max-flow min-cut matroids: Polynomial testing and polynomial algorithms for maximum flow and shortest routes. Mathematics of Operations Research 12(1), 72–96 (1987)
22. Zaslavsky, T.: Biased graphs. I. Bias, balance, and gains. J. Comb. Theory Ser. B 47, 32–52 (1989)

A Stochastic Probing Problem with Applications

Anupam Gupta[1] and Viswanath Nagarajan[2]

[1] Computer Science Department, Carnegie Mellon University
[2] IBM T.J. Watson Research Center

Abstract. We study a general *stochastic probing* problem defined on a universe V, where each element $e \in V$ is "active" independently with probability p_e. Elements have weights $\{w_e : e \in V\}$ and the goal is to maximize the weight of a chosen subset S of active elements. However, we are given only the p_e values—to determine whether or not an element e is active, our algorithm must *probe* e. If element e is probed and happens to be active, then e must irrevocably be added to the chosen set S; if e is not active then it is not included in S. Moreover, the following conditions must hold in every random instantiation:

- the set Q of probed elements satisfy an "outer" packing constraint,
- the set S of chosen elements satisfy an "inner" packing constraint.

The kinds of packing constraints we consider are intersections of matroids and knapsacks. Our results provide a simple and unified view of results in stochastic matching [1, 2] and Bayesian mechanism design [3], and can also handle more general constraints. As an application, we obtain the first polynomial-time $\Omega(1/k)$-approximate "Sequential Posted Price Mechanism" under k-matroid intersection feasibility constraints, improving on prior work [3–5].

1 Introduction

We study an adaptive stochastic optimization problem along the lines of [6–9]. The *stochastic probing* problem is defined on a universe V of elements with weights $\{w_e : e \in V\}$. We are also given two downwards-closed set systems $(V, \mathcal{I}_{in})$ and $(V, \mathcal{I}_{out})$, which we call the *inner* and *outer* packing constraints, whose meanings we shall give shortly. For each element $e \in V$, there is a probability p_e, where element e is *active/present* with this probability, independently of all other elements. We want to choose a set $S \subseteq V$ of active elements belonging to $\mathcal{I}_{in}$, i.e., all elements in the chosen set S must be active and also independent according to the inner packing constraint ($S \in \mathcal{I}_{in}$). The goal is to maximize the expected weight of the chosen set.

However, the information about which elements are active and which are inactive is not given up-front. All we know are the probabilities p_e, and that the active set is a draw from the product distribution given by $\{p_e\}_{e \in V}$—to determine if an element e is active or not, we must *probe* e. Moreover, if we probe e, and e happens to be active, then we *must* irrevocably add e to our chosen set S—we do not have a right to discard any probed element that turns out to be active. This "query and commit" model is quite natural in a number

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 205–216, 2013.

of applications such as kidney exchange, online dating and auction design (see below for details).

Finally, there is a constraint on which elements we can probe: the set Q of elements probed in any run of the algorithm must be independent according to the outer packing constraint $\mathcal{I}_{out}$—i.e., $Q \in \mathcal{I}_{out}$. This is the constraint that gives the probing problem its richness. Since every probed element that is active must be included in the solution which needs to maintain independence in $\mathcal{I}_{in}$, at any point t (with current solution S_t and currently probed set Q_t) we can only probe those elements e with $Q_t \cup \{e\} \in \mathcal{I}_{out}$ and $S_t \cup \{e\} \in \mathcal{I}_{in}$.[1]

While the stochastic probing problem seems fairly abstract, it has interesting applications: we give two applications of this problem, to designing posted-price Bayesian auctions, and to modeling problems in online dating/kidney exchange. We first state our results and then describe these applications.

Our Results. For the *unweighted* stochastic probing problem (i.e., $w_e = 1$ for all $e \in V$), if both inner and outer packing constraints are given by k-systems[2], we consider the greedy algorithm which considers elements in decreasing order of their probability p_e, probing them whenever feasible.

Theorem 1 (Unweighted Probing). *The greedy algorithm for unweighted stochastic probing achieves a tight* $\frac{1}{k_{in}+k_{out}}$*-approximation ratio, when* $\mathcal{I}_{in}$ *is a* k_{in}*-system and* $\mathcal{I}_{out}$ *is a* k_{out}*-system.*

This result generalizes the greedy 4-approximation algorithm for unweighted stochastic matching, by Chen et al. [1], where both inner and outer constraints are b-matchings (and hence 2-systems). For the special case of stochastic matching, Adamczyk [10] gave an improved factor-2 bound. However, Theorem 1 is tight in our setting of general k-systems; its proof is LP-based, and we feel it is much simpler than previous proofs for the special cases. The main idea of our proof is a dual-fitting argument that extends the Fisher et al. [11] analysis of the greedy algorithm for k-matroid intersection.

There is no known greedy algorithm for stochastic probing in the *weighted* case (as opposed to the deterministic setting of finding the maximum weight set subject to a k-system, where greedy gives a $1/k$-approximation [12, 11]); indeed, natural greedy approaches can be arbitrarily bad even for weighted stochastic matching [1]. Hence, we use an LP relaxation for the weighted probing problem, where variables correspond to marginal probabilities of probing/choosing elements in the optimal policy. This is similar to previous works on such adaptive stochastic problems [7, 8, 2]. Our rounding algorithm is based on the recently introduced notion of *contention resolution (CR) schemes* for packing constraints due to Chekuri et al. [13]. We show that the existence of suitable CR-schemes for both $\mathcal{I}_{in}$ and $\mathcal{I}_{out}$ imply an approximation algorithm for weighted stochastic

[1] Indeed, if $p_e = 0$ there is no point probing e; and if $p_e > 0$ there is a danger that e is active and we will be forced to add it to S_t, which we cannot if $S_t \cup \{e\} \notin \mathcal{I}_{in}$.

[2] For any integer k, a *k-system* is a downwards-closed collection of sets $\mathcal{I} \subseteq 2^V$ such that for any $S \subseteq V$, the maximal subsets of S that belong to $\mathcal{I}$ can differ in size by at most a factor of k. Examples are intersections of k matroids, and k-set packing.

probing, where the approximation ratio depends on the quality of the two CR-schemes. Our main result for weighted stochastic probing is Theorem 5 (which requires some notation to state precisely), but a representative corollary is an $\Omega\left(\frac{1}{k_{in}+k_{out}}\right)$-approximation algorithm when the inner and outer constraints are intersections of k_{in} and k_{out} matroids, respectively. Some of the other allowed constraints are unsplittable flow on trees (under the "no-bottleneck" assumption) and packing integer programs. Details on the weighted case appear in Section 3.

Applications. We now give two applications: the first shows how our algorithm for the weighted probing problem immediately gives us posted price auctions for single parameter settings where the feasibility set is given by intersections of matroids, the second is an application for dating/kidney exchange. Both of these extend and generalize previous results in these areas.

Bayesian Auction Design. Consider a mechanism design setting for a single seller facing n single-parameter buyers. The seller has a feasibility constraint given by a downward-closed set system $\mathcal{I} \subseteq 2^{[n]}$ and is allowed to serve any set of buyers from $\mathcal{I}$. Buyers are *single-parameter*; i.e., buyer i's private data is a single real number v_i which denotes his valuation of being served (if i is not served then he receives zero value). In the *Bayesian* setting, the valuation v_i is drawn from some set $\{0, 1, \ldots, B\}$ according to probability distribution $\mathcal{D}_i$; here we assume that the valuations of buyers are discrete and independently drawn. The valuation v_i is private to the buyer, but the distribution $\mathcal{D}_i$ is public knowledge. The goal in these problems is a revenue-maximizing truthful mechanism that accepts bids from buyers and outputs a feasible allocation (i.e., a set $S \in \mathcal{I}$ of buyers that receive service), along with a price that each buyer has to pay for service. A very special type of mechanism is a *Sequential Posted Pricing Mechanism* (SPM) that chooses a price for each buyer and makes "take-it-or-leave-it" offers to the buyers in some order [14, 15, 3]. Such mechanisms are simple to run and obviously truthful (see [3] for a discussion of other advantages), hence it is of interest to design SPMs which achieve revenue comparable to the revenue-optimal mechanism.

Designing the best SPM can be cast as a stochastic probing problem on a universe $V = \{1, 2, \ldots, n\} \times \{0, 1, \ldots, B\}$, where element (i, c) corresponds to offering a price c to buyer i. Element (i, c) has weight $w_{ic} = c$, which is the revenue obtained if the offer "price c for buyer i" is accepted, and has probability $p_{ic} = \Pr_{v_i \sim \mathcal{D}_i}[v_i \geq c]$, which is the probability that i will indeed accept service at price c. The inner constraint $\mathcal{I}_{in}$ is now the natural lifting of the actual constraints $\mathcal{I}$ to the universe V, where $\{(i, c)\}_{c \geq 0}$ are copies of i. The outer constraint $\mathcal{I}_{out}$ requires that at most one of the elements $\{(i, c) \mid c \geq 0\}$ can be probed for each i: i.e., each buyer i can be offered at most one price. This serves two purposes: firstly, it gives us a posted-price mechanism. Secondly, we required in our model that each element (i, c) is active with probability p_{ic}, *independently* of the other elements (i, c'); however, the underlying semantics imply that if i accepts price c, then she would also accept any $c' \leq c$, which would give us correlations. Constraining ourselves to probe at most one element corresponding

to each buyer i means we never probe two correlated elements, and hence the issue of correlations never arises.

Our results for stochastic probing give near-optimal SPMs for many feasibility constraints. Moreover, we show that our LP relaxation not only captures the best possible SPMs, but also captures the optimal truthful mechanism *of any form* under the Bayes-Nash equilibrium (and hence Myerson's optimal mechanism [16]). In the case of k matroid intersection feasibility constraints, our results give the first polynomial-time sequential posted price mechanisms whose revenue is $\Omega(1/k)$ times the optimum. Previous papers [3–5] proved the existence of such SPMs, but they were polynomial-time only for $k \leq 2$. For larger k, previous works only showed *existence* of $\Omega(1/k)$-approximate SPMs, and polynomial-time implementations of these SPMs only obtained an $\Omega(1/k^2)$ fraction of the optimal revenue. The previous results also compare the performance of their SPMs directly to the revenue of the optimal mechanism [16], whereas we compare our SPMs to an LP relaxation of this mechanism, which is potentially larger. Moreover, our general framework gives us more power:

- We can handle broader classes of feasibility constraints $\mathcal{I}$, not just matroid intersections: e.g., we can model auctions involving unsplittable flow on trees, which can be used to capture allocations of point-to-point bandwidths in a tree-shaped network. This is because the feasibility constraints $\mathcal{I}$ for the auction directly translate into inner constraints for the probing problem.
- We can also handle additional side-constraints to the auction via a richer class of outer constraints $\mathcal{I}_{out}$. For example, the seller may incur costs in the form of time/money to make offers. Such budget limits can be modeled in the stochastic probing problem as an extra outer knapsack constraint, and our algorithm finds approximately optimal SPMs even in this case. More generally, our algorithm can easily handle a rich class of other resource constraints (matroid intersections, packing IPs etc) on the auction. However, in the presence of these side-constraints, our algorithm's revenue is an approximation only to the best SPM satisfying these constraints, and no longer comparable to the unconstrained optimal mechanism.

Online Dating and Kidney Exchange [1]. Consider a dating agency with several users. Based on the profiles of users, the agency can compute the probability that any pair of users will be compatible. Whether or not a pair is successfully matched is only known after their date; moreover, in the case of a match, both users immediately leave the site (happily). Furthermore, each user has a patience/timeout level, which is the maximum number of failed dates after which he/she drops out of the site (unhappily). The objective of the dating site is to schedule dates so as to maximize the expected number of matched pairs. (Similar constraints arise in kidney exchange systems.) This can be modeled as stochastic probing with the universe V being edges of the complete graph whose nodes correspond to users. The inner constraints specify that the chosen edges be a matching in G. The outer constraints specify that for each node j, at most t_j edges incident to j can be probed, where t_j denotes the patience level of user

j. Both these are b-matching constraints; in fact when the graph is bipartite, they are intersections of two partition matroids.

Our results will give an alternate way to obtain constant factor approximation algorithms for this stochastic matching problem. Such algorithms were previously given by [1, 2], but they relied heavily on the underlying graph structure. Additionally, our techniques allow for more general sets of constraints. E.g., not all potential dates may be equally convenient to a user, and (s)he might prefer dates with other nearby users. This can be modeled as a sequence of patience bounds for the user, specifying the maximum number of dates that the user is willing to go outside her neighborhood/city/state etc. In particular, if $u_1, u_2, \ldots, u_n$ denote the users in decreasing distance from user j then there is a non-decreasing sequence $\langle t_j^1, \ldots, t_j^n \rangle$ of numbers where user j wishes to date at most t_j^r users among the r farthest other users $\{u_1, \ldots, u_r\}$. This corresponds to the stochastic probing problem, where the inner constraint remains matching but the outer constraint becomes a 2-system. Our algorithm achieves a constant approximation even here.

Other Related Work. Dean et al. [7, 8] were the first to consider approximation algorithms for *stochastic packing* problems in the adaptive optimization model. For the stochastic knapsack problem, where items have random sizes (that instantiate immediately after selection), [7] gave a $(3 + \epsilon)$-approximation algorithm; this was improved to $2+\epsilon$ in [17, 18]. [8] considered stochastic packing integer programs (PIPs) and gave approximation guarantees matching the best known deterministic bounds. Our stochastic probing problem can be viewed as a two-level generalization of stochastic packing, with two different packing constraints: one for probed elements, and one for chosen elements. However, all random variables in our setting are $\{0, 1\}$-valued (each element is either active or not), whereas [7, 8] allow arbitrary non-negative random variables.

Chen et al. [1] first studied a *stochastic probing* problem: they introduced the unweighted stochastic matching problem and showed that greedy is a 4-approximation algorithm. Adamczyk [10] improved the analysis to show a bound of 2. Both these proofs involve intricate arguments on the optimal decision tree. In contrast, our analysis of greedy is much simpler and LP-based, and extends to the more general setting of k-systems. (For the stochastic matching, our result implies a 4-approximation.) Bansal et al. [2] gave a different LP proof that greedy is a 5-approximation for stochastic matching, but their proof relied heavily on the graph structure, making the extension to general k-systems unclear. [2] also gave the first $O(1)$-approximation for *weighted* stochastic matching, which was LP-based. ([1] showed that natural greedy approaches for weighted stochastic matching are arbitrarily bad.) Our algorithm for weighted probing is also LP-based, where we make use of the elegant abstraction of "contention resolution schemes" introduced by Chekuri et al. [13] (see Section 3), which provides a clean approach to rounding the LP.

The papers of Chawla et al. [3], Yan [4], and Kleinberg and Weinberg [5] study the performance of Sequential Posted Price Mechanisms (SPMs) for Bayesian single-parameter auctions, and relate the revenue obtained by SPMs to the

optimal (non-posted-price) mechanism given by Myerson [16]. Our algorithm for stochastic probing also yields SPMs for Bayesian auctions where the feasible sets of buyers are specified by, e.g., k-matroid intersection and unsplittable flow on trees. Our proof relates an LP relaxation of the optimal mechanism to the LP used for stochastic probing. Linear programs have been used to model optimal auctions in a number of settings; e.g., see Vohra [19]. Bhattacharya et al. [20] also used LP relaxations to obtain approximately optimal mechanisms in a Bayesian setting with multiple items and budget constrained buyers.

Specifying Probing Algorithms. A solution (policy) to the stochastic probing problem is an *adaptive* strategy of probing elements satisfying the constraints imposed by $\mathcal{I}_{out}$ and $\mathcal{I}_{in}$. At any time step $t \geq 1$, let Q_t denote the set of elements already probed and S_t the current solution (initially $Q_1 = S_1 = \emptyset$); an element $e \in V \setminus Q_t$ can be probed at time t if and only if $Q_t \cup \{e\} \in \mathcal{I}_{out}$ and $S_t \cup \{e\} \in \mathcal{I}_{in}$. If e is probed then exactly one of the following happens:

- e is active (with probability p_e), and $Q_{t+1} \leftarrow Q_t \cup \{e\}$, $S_{t+1} \leftarrow S_t \cup \{e\}$, or
- e is inactive (with probability $1 - p_e$), and $Q_{t+1} \leftarrow Q_t \cup \{e\}$, $S_{t+1} \leftarrow S_t$.

Hence the policy is a decision tree with nodes representing elements that are probed and branches corresponding to their random instantiations. Note that an optimal policy may be exponential sized, and designing a polynomial-time algorithm requires tackling the question of whether there exist poly-sized near-optimal strategies. A *non adaptive* policy is simply given by a permutation on V, where elements are considered in this order and probed whenever feasible in both $\mathcal{I}_{out}$ and $\mathcal{I}_{in}$. The *adaptivity gap* compares the best non-adaptive policy to the best adaptive policy.

Packing Constraints. We model packing constraints as *independence systems*, which are of the form $(V, \mathcal{I} \subseteq 2^V)$ where V is the *universe* and $\mathcal{I}$ is a collection of *independent sets*. We assume $\mathcal{I}$ is *downwards closed*, i.e., $A \in \mathcal{I}$ and $B \subseteq A \implies B \in \mathcal{I}$. Some examples are:

- *Knapsack constraint*: each element $e \in V$ has size $s_e \in [0, 1]$ and $\mathcal{I} = \{A \subseteq V \mid \sum_{e \in A} s_e \leq 1\}$.
- *Matroid constraint*: an independence system $(V, \mathcal{I})$ where for any subset $S \subseteq V$, every maximal independent subset of S has the same size. See [21] for many properties and examples.
- *k-system:* an independence system $(V, \mathcal{I})$ where for any subset $S \subseteq V$, every *maximal* independent subset of S has size at least $\frac{1}{k}$ times the size of the *maximum* independent subset of S. For example: matroids are 1-systems, matchings are 2-systems, and intersections of k matroids form k-systems.
- *Unsplittable Flow Problem (UFP) on trees:* there is an edge-capacitated tree T, and each element $e \in V$ corresponds to a path P_e in T and demand d_e. Subset $S \subseteq V$ is independent (i.e. $S \in \mathcal{I}$) iff $\{\text{path } P_e \text{ with demand } d_e\}_{e \in S}$ is routable in T. We assume the "no-bottleneck" condition, where the maximum demand $\max_{e \in V} d_e$ is at most the minimum capacity in T.

When the universe is clear from context, we refer to an independence system $(V, \mathcal{I})$ just as $\mathcal{I}$. We also make use of linear programming relaxations for independence

systems: the LP relaxation of $\mathcal{I}$ is denoted by $\mathcal{P}(\mathcal{I}) \subseteq [0,1]^V$ and contains the convex hull of all independent sets. (Since $\mathcal{P}(\mathcal{I})$ is a relaxation it need not equal the convex hull). For example: $\mathcal{P}(\mathcal{I}) = \left\{\mathbf{x} \in [0,1]^V : \sum_{e \in V} s_e \cdot x_e \leq 1\right\}$ for knapsacks; $\mathcal{P}(\mathcal{I}) = \left\{\mathbf{x} \in [0,1]^V : \sum_{e \in S} x_e \leq r_{\mathcal{I}}(S), \forall S \subseteq V\right\}$ for matroids, where $r_{\mathcal{I}}(\cdot)$ denotes the rank function.

2 Unweighted Stochastic Probing

In this section, we study the stochastic probing problem with unit weights, i.e., $w_e = 1$ for all $e \in V$. We assume the inner and outer packing constraints are a k_{in}-system and a k_{out}-system, repectively. We show that the greedy algorithm, which considers elements in non-increasing order of their probabilities p_e and probes them when feasible, has performance claimed in Theorem 1. We give an LP-based dual-fitting proof of this result.

For brevity, let us use k to denote k_{in}, and k' to denote k_{out}. Let the *rank function* of $\mathcal{I}_{in}$ be $r : 2^V \to \mathbb{N}$, where for each $S \subseteq V$, $r(S) = \max\{|I| \mid I \in \mathcal{I}, I \subseteq S\}$ be the *maximum* size of an independent subset of S. By definition of k-systems, for any $S \subseteq V$, any maximal independent set of S (according to $\mathcal{I}_{in}$) has size at least $r(S)/k$. Similarly, let $r' : 2^V \to \mathbb{N}$ denote the rank function of $\mathcal{I}_{out}$. We may not be able to evaluate the rank function, since this is NP-complete for $k \geq 3$. For any $T \subseteq V$, let $\text{span}(T) = \{e \in V : r(T \cup \{e\}) = r(T)\}$ be the *span* of T. Likewise, let span' denote the span function for $\mathcal{I}_{out}$.

Claim 2. *For any $T \subseteq V$, the maximum independent subset of T (which has size $r(T)$) is a maximal independent subset of* $\text{span}(T)$. *Hence, for $T \subseteq V$ and $R \subseteq V$, we have $r(\text{span}(T)) \leq k \cdot r(T) \leq k \cdot |T|$ and $r'(\text{span}'(R)) \leq k' \cdot r'(R) \leq k' \cdot |R|$.*

Let us write the natural LP relaxation and dual for the probing problem:

$$\begin{array}{lll} \max & \sum_{e \in V} p_e y_e & \\ \text{s.t.} & \sum_{e \in S} p_e y_e \leq r(S) & \forall S \subseteq V \\ & \sum_{e \in S} y_e \leq r'(S) & \forall S \subseteq V \\ & \mathbf{y} \geq 0. & \end{array} \qquad \begin{array}{lll} \min & \sum_S r(S)\,\alpha(S) + \sum_S r'(S)\,\beta(S) & \\ \text{s.t.} & p_e \sum_{S: e \in S} \alpha(S) + \sum_{S: e \in S} \beta(S) \geq p_e & \forall e \in V \\ & \alpha(S), \beta(S) \geq 0 & \forall S \subseteq V. \end{array}$$

Claim 3 in the next section shows that this LP is a valid relaxation. It is not known if these linear programs can be solved in polynomial time for arbitrary p-systems $\mathcal{I}_{in}$ and $\mathcal{I}_{out}$; we use them only for the analysis. Note that the greedy algorithm defines a non-adaptive strategy. Consider a sample path π down the natural decision tree associated with the above algorithm; it is completely defined by the randomness in which elements are active. Let $\Pr[\pi]$ denote its probability, and Q_π, S_π be the sets probed and picked on taking this path.

Lemma 1. *If* alg *is the random variable denoting the number of elements picked,*

$$\mathbb{E}[\mathsf{alg}] = \sum_\pi \Pr(\pi) \cdot |S_\pi| = \sum_\pi \Pr(\pi) \cdot \sum_{e \in Q_\pi} p_e.$$

Lemma 2. *For each outcome π, there is a feasible dual of value at most $k|S_\pi| + k' \sum_{e \in Q_\pi} p_e$. Moreover, there is a feasible dual of value at least $(k + k')\mathbb{E}[\mathsf{alg}]$.*

The following proof is similar to that of Fisher et al. [11] showing that the greedy algorithm is a k-approximation for the intersection of k matroids.

Proof. Let $A = \text{span}(S_\pi)$ be the span of the set of picked elements S_π; this is well-defined since S_π is independent in $\mathcal{I}_{in}$. We set $\alpha(A) = 1$, and all other α variables to zero.

Let the set of probed elements $Q_\pi = \{a_1, a_2, \ldots, a_\ell\}$ in this order. Define

$$\beta(\text{span}'(\{a_1, a_2, \ldots, a_h\})) := p_{a_h} - p_{a_{h+1}} \geq 0$$

for all $h \in \{1, \ldots, \ell\}$ (where we imagine $p_{a_{\ell+1}} = 0$). This is also well-defined since every subset of Q_π is independent in $\mathcal{I}_{out}$. The non-negativity follows from the greedy algorithm that probes elements in decreasing probabilities. The dual objective value equals:

$$r(A) + \sum_{h=1}^{\ell} r'(\text{span}'(\{a_1, a_2, \ldots, a_h\})) \cdot (p_{a_h} - p_{a_{h+1}}) \;\leq\; k \cdot |S_\pi| + \sum_{h=1}^{\ell} k' \cdot h \cdot (p_{a_h} - p_{a_{h+1}}),$$

which is $k \cdot |S_\pi| + k' \sum_{e \in Q_\pi} p_e$. The inequality is by Claim 2. Next we show that the dual solution is feasible. The non-negativity is clearly satisfied, so it remains to check feasibility of the dual covering constraints. For any $e \in V$,

- Case I: $e \in Q_\pi$. Say $e = a_g$ in the ordering of the set Q_π . Then e lies in $\text{span}'(\{a_1, a_2, \ldots, a_h\})$ for all $h \geq g$. Hence, the left hand side of e's covering constraint contributes at least

$$\sum_{h=g}^{\ell} \beta(\text{span}'(\{a_1, a_2, \ldots, a_h\})) = \sum_{h=g}^{\ell} (p_{a_h} - p_{a_{h+1}}) = p_{a_g} = p_e.$$

- Case II: $e \notin Q_\pi$ because of the outer constraint. Say e was seen when the Q set was $\{a_1, a_2, \ldots, a_g\}$. Then $e \in \text{span}'(\{a_1, a_2, \ldots, a_h\})$ for all $h \geq g$. In this case, the left hand side contributes at least

$$\sum_{h=g}^{\ell} \beta(\text{span}'(\{a_1, a_2, \ldots, a_h\})) = \sum_{h=g}^{\ell} (p_{a_h} - p_{a_{h+1}}) = p_{a_g} \geq p_e.$$

  Here we used the fact that elements are considered in decreasing order of their probabilities.
- Case III: $e \notin Q_\pi$ because of the inner constraint. Then $e \in \text{span}(S_\pi) = A$, and hence the $p_e \sum_{S: e \in S} \alpha(S) = p_e \, \alpha(A) = p_e$.

This proves the first part of the lemma. Taking expectations over π, the resulting convex combination $\sum_\pi \Pr[\pi](\boldsymbol{\alpha}_\pi, \boldsymbol{\beta}_\pi)$ of these feasible duals is another feasible dual of value $k\,\mathbb{E}[|S_\pi|] + k'\,\mathbb{E}[\sum_{e \in Q_\pi} p_e]$, which by Lemma 1 equals $(k + k')\mathbb{E}[\mathsf{alg}]$.

Our analysis for the greedy algorithm is tight. In particular, if all p_e's equal one, and the inner and outer constraints are intersections of (arbitrary) partition matroids, then we obtain the greedy algorithm for $(k_{in} + k_{out})$-dimensional matching. The approximation ratio in this case is known to be exactly $k_{in} + k_{out}$.

3 Weighted Stochastic Probing

We now turn to the general weighted case of stochastic probing. Here the natural combinatorial algorithms perform poorly, so we use linear programming relaxations of the problem, which we round to get non-adaptive policies. Given an instance of the stochastic probing problem with inner constraints $(V, \mathcal{I}_{in})$ and outer constraints $(V, \mathcal{I}_{out})$, we use the following LP relaxation:

$$\begin{array}{lll} \max & \sum_{e \in V} w_e \cdot x_e & \\ s.t. & x_e = p_e \cdot y_e & \forall e \in V \qquad (\mathcal{LP}) \\ & x \in \mathcal{P}(\mathcal{I}_{in}) & \\ & y \in \mathcal{P}(\mathcal{I}_{out}) & \end{array}$$

We assume that the LP relaxations of the inner and outer constraints can be solved efficiently: this is true for matroids, knapsacks, UFP on trees, and their intersections. For general k-systems, it is not known if this LP can be solved exactly. However, using the fact that the greedy algorithm achieves a $\frac{1}{k}$-approximation for maximizing linear objective functions over k-systems (even with respect to the LP relaxation, which follows from [11], or the proof of Lemma 2), and the equivalence of separation and optimization, we can obtain a $\frac{1}{k}$-approximate LP solution when $\mathcal{I}_{in}$ and $\mathcal{I}_{out}$ are k-systems.

Claim 3. *The optimal value of $(\mathcal{LP}) \geq$ optimal value of the probing instance.*

Given a solution (x, y) for the LP relaxation, we need to get a policy from it. Our rounding algorithm is based on the elegant abstraction of *contention resolution schemes (CR schemes)*, as defined in Chekuri et al. [13]. Here is the formal definition, and the main theorem we will use.

Definition 1. *An independence system $(V, \mathcal{J} \subseteq 2^V)$ with LP-relaxation $\mathcal{P}(\mathcal{J})$ admits a monotone (b, c)* **CR-scheme** *if, for any $z \in \mathcal{P}(\mathcal{J})$ there is a (possibly randomized) mapping $\pi : 2^V \to \mathcal{J}$ such that:*

(i) If $I \subseteq V$ is a random subset where each element $e \in V$ is chosen independently with probability $b \cdot x_e$, $\Pr_{I,\pi}[e \in \pi(I) \mid e \in I] \geq c$ for all $e \in V$.
(ii) For any $e \in I_1 \subseteq I_2 \subseteq V$, $\Pr_{\pi}[e \in \pi(I_1)] \geq \Pr_{\pi}[e \in \pi(I_2)]$.
(iii) The map π can be computed in polynomial time.

Moreover, $\pi : 2^V \to \mathcal{J}$ is a (b, c) **ordered CR-scheme** *if there is a (possibly random) permutation σ on V so that for each $I \subseteq V$, $\pi(I)$ is the maximal independent subset of I obtained by considering elements in the order of σ.*

Theorem 4 ([13, 22, 23, 3]). *There are monotone CR-schemes for the following independence systems (below, $0 < b \leq 1$ is any value unless specified otherwise)*

- *$(b, (1 - e^{-b})/b)$ CR-scheme for matroids.*
- *$(b, 1 - k \cdot b)$ ordered CR-scheme for k-systems.*
- *$(b, 1 - 6b)$ ordered CR-scheme for unsplittable flow on trees, with the "no bottleneck" assumption, for any $0 < b \leq 1/60$.*
- *$(b, 1 - 2kb)$ CR-scheme for k-column sparse packing integer programs.*

Given the formalism of CR schemes, we can now state our main result for rounding a solution to the relaxation ($\mathcal{LP}$).

Theorem 5. *Consider any instance of the stochastic probing problem with*

(i) (b, c_{out}) *CR-scheme for* $\mathcal{P}(\mathcal{I}_{out})$.
(ii) Monotone (b, c_{in}) **ordered** *CR-scheme for* $\mathcal{P}(\mathcal{I}_{in})$.

Then there is a $b \cdot (c_{out} + c_{in} - 1)$*-approximation algorithm for the weighted stochastic probing problem.*

Before we prove Theorem 5, we observe that combining Theorems 5 and 4 gives us, for example:

- a $1/(4(k+\ell))$-approximation algorithm when the inner and outer constraints are intersections of k and ℓ matroids respectively.
- an $\Omega(1)$-approximation algorithm when the inner and outer constraints are unsplittable flows on trees/paths satisfying the no-bottleneck assumption.

The Rounding Algorithm. Let π_{out} denote the randomized mapping corresponding to a (b, c_{out}) CR-scheme for $y \in \mathcal{P}(\mathcal{I}_{out})$, and π_{in} be that corresponding to a (b, c_{in}) CR-scheme for $x \in \mathcal{P}(\mathcal{I}_{in})$. The algorithm to round the LP solution (x, y) for weighted stochastic probing appears as Algorithm 3.1.

Algorithm 3.1. Rounding Algorithm for Weighted Probing

1: Pick $I \subseteq 2^V$ by choosing each $e \in V$ independently with probability $b \cdot y_e$.
2: Let $P = \pi_{out}(I)$. (By definition of the CR scheme, $P \in \mathcal{I}_{out}$ with probability one.)
3: Order elements in P according to σ (the inner ordered CR scheme) to get $e_1, e_2, \ldots, e_{|P|}$.
4: Set $S \leftarrow \emptyset$.
5: **for** $i = 1, \ldots, |P|$ **do**
6: **if** $(S \cup \{e_i\} \in \mathcal{I}_{in})$ **then**
7: Probe e_i: set $S \leftarrow S \cup \{e_i\}$ if e_i is active, and $S \leftarrow S$ otherwise.

The Analysis. We now show that $\mathbb{E}[w(S)]$ is large compared to the LP value $\sum_e w_e x_e$. To begin, a few observations about this algorithm. Note that this is a randomized strategy, since there is randomness in the choice of I and maybe in the maps π_{out} and π_{in}. Also, by the CR scheme properties, the probed elements are in $\mathcal{I}_{out}$, and the chosen elements in $\mathcal{I}_{in}$. Finally, having chosen the set P to (potentially) probe, the elements actually probed in step 7 relies on the *ordered* CR scheme for the inner constraints.

Recall that $I \subseteq V$ is the random set where each element e is included independently with probability $b \cdot y_e$; also $P = \pi_{out}(I)$. Let $J \subseteq V$ be the set of active elements; i.e., each $e \in V$ is present in J independently with probability p_e. The set of chosen elements is now $S = \pi_{in}(P \cap J)$. The main lemma is now:

Lemma 3. *For any $e \in V$,*

$$\Pr_{I,\pi_{out},J,\pi_{in}}[e \in \pi_{in}(\pi_{out}(I) \cap J)] \quad \geq \quad b \cdot (c_{out} + c_{in} - 1) \cdot x_e,$$

where b, c_{out}, c_{in} are parameters given by our CR-schemes.

Proof. Recall that $P = \pi_{out}(I)$, so we want to lower bound:

$$\begin{aligned}\Pr[e \in \pi_{in}(P \cap J)] &= \Pr[e \in \pi_{in}(P \cap J) \wedge e \in I \cap J \cap P] \\ &= \Pr[e \in I \cap J \cap P] - \Pr[e \notin \pi_{in}(P \cap J) \wedge e \in I \cap J \cap P] \\ &\geq b x_e \cdot c_{out} - \Pr[e \notin \pi_{in}(P \cap J) \wedge e \in I \cap J \cap P], \qquad (1)\end{aligned}$$

where the inequality uses $\Pr[e \in I \cap J] = b y_e \cdot p_e = b x_e$ and $\Pr[e \in P = \pi_{out}(I) | e \in I \cap J] \geq c_{out}$ by Definition 1(i) applied to the outer CR scheme, since I is a random subset chosen according to $b \cdot y$ where $y \in \mathcal{P}(\mathcal{I}_{out})$.

We now upper bound $\Pr[e \notin \pi_{in}(P \cap J) \wedge e \in I \cap J \cap P]$ by $(1 - c_{in}) \cdot b x_e$ which combined with (1) would prove the lemma. Now, condition on any instantiation $I = I_1$, $P = \pi_{out}(I_1) = P_1 \subseteq I_1$ and $J = J_1$ such that $e \in I_1 \cap J_1 \cap P_1$. Then,

$$\Pr[e \notin \pi_{in}(P_1 \cap J_1)] \quad \leq \quad \Pr[e \notin \pi_{in}(I_1 \cap J_1)], \qquad (2)$$

by Definition 1(ii) applied to the inner CR scheme (since $e \in P_1 \cap J_1 \subseteq I_1 \cap J_1$). Taking a linear combination of the inequalities in (2) with respective multipliers $\Pr[I = I_1, J = J_1, P = P_1]$ (where $e \in I_1 \cap J_1 \cap P_1$), we obtain

$$\begin{aligned}\Pr[e \notin \pi_{in}(P \cap J) \wedge e \in I \cap J \cap P] &\leq \Pr[e \notin \pi_{in}(I \cap J) \wedge e \in I \cap J \cap P] \\ &\leq \Pr[e \notin \pi_{in}(I \cap J) \wedge e \in I \cap J] \\ &= b x_e \cdot \Pr[e \notin \pi_{in}(I \cap J) | e \in I \cap J]\end{aligned}$$

where the equality uses $\Pr[e \in I \cap J] = b y_e \cdot p_e = b x_e$. The last expression above is at most $b x_e (1 - c_{in})$ by Definition 1(i) applied to the inner CR scheme, since $I \cap J$ is a random subset chosen according to $b \cdot x$ where $x \in \mathcal{P}(\mathcal{I}_{in})$. This proves the desired upper bound $\Pr[e \notin \pi_{in}(P \cap J) \wedge e \in I \cap J \cap P] \leq (1 - c_{in}) \cdot b x_e$.

Consequently, the expected weight of the chosen set S is

$$\mathbb{E}\left[\sum_{e \in S} w_e\right] \;=\; \sum_{e \in V} w_e \cdot \Pr\left[e \in \pi_{in}\left(P \cap J\right)\right] \;\geq\; b(c_{in} + c_{out} - 1) \cdot \sum_{e \in V} w_e \cdot x_e.$$

The inequality uses Lemma 3. This completes the proof of Theorem 5.

Acknowledgments. We thank Shuchi Chawla, Bobby Kleinberg, Tim Roughgarden, Rakesh Vohra, and Matt Weinberg for helpful clarifications and discussions. We also thank an anonymous reviewer for pointing out that the LP for weighted stochastic probing can be solved approximately for general k-systems. Part of this work was done when the first-named author was visiting the IEOR Department at Columbia University, and IBM Thomas J. Watson Research Center; he thanks them for their generous hospitality.

References

1. Chen, N., Immorlica, N., Karlin, A.R., Mahdian, M., Rudra, A.: Approximating Matches Made in Heaven. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009, Part I. LNCS, vol. 5555, pp. 266–278. Springer, Heidelberg (2009)
2. Bansal, N., Gupta, A., Li, J., Mestre, J., Nagarajan, V., Rudra, A.: When LP Is the Cure for Your Matching Woes: Improved Bounds for Stochastic Matchings. Algorithmica 63, 733–762 (2012)
3. Chawla, S., Hartline, J.D., Malec, D.L., Sivan, B.: Multi-parameter mechanism design and sequential posted pricing. In: STOC, pp. 311–320 (2010)
4. Yan, Q.: Mechanism Design via Correlation Gap. In: SODA, pp. 710–719 (2011)
5. Kleinberg, R., Weinberg, S.M.: Matroid prophet inequalities. In: STOC, pp. 123–136 (2012)
6. Möhring, R.H., Schulz, A.S., Uetz, M.: Approximation in stochastic scheduling: the power of LP-based priority policies. Journal of the ACM (JACM) 46, 924–942 (1999)
7. Dean, B.C., Goemans, M.X., Vondrák, J.: Approximating the stochastic knapsack problem: the benefit of adaptivity. Math. Oper. Res. 33, 945–964 (2008)
8. Dean, B.C., Goemans, M.X., Vondrák, J.: Adaptivity and approximation for stochastic packing problems. In: SODA, pp. 395–404 (2005)
9. Guha, S., Munagala, K.: Approximation algorithms for budgeted learning problems. In: STOC, pp. 104–113 (2007)
10. Adamczyk, M.: Improved analysis of the greedy algorithm for stochastic matching. Inf. Process. Lett. 111, 731–737 (2011)
11. Fisher, M., Nemhauser, G., Wolsey, L.: An analysis of approximations for maximizing submodular set functions II. Mathematical Programming Study 8, 73–87 (1978)
12. Jenkyns, T.: The efficiency of the "greedy" algorithm. In: 7th South Eastern Conference on Combinatorics, Graph Theory and Computing, pp. 341–350 (1976)
13. Chekuri, C., Vondrák, J., Zenklusen, R.: Submodular function maximization via the multilinear relaxation and contention resolution schemes. In: STOC, pp. 783–792 (2011), Full version `http://arxiv.org/abs/1105.4593`
14. Sandholm, T., Gilpin, A.: Sequences of take-it-or-leave-it offers: near-optimal auctions without full valuation revelation. In: 5th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS, pp. 1127–1134 (2006)
15. Blumrosen, L., Holenstein, T.: Posted prices vs. negotiations: an asymptotic analysis. In: ACM Conference on Electronic Commerce (2008)
16. Myerson, R.: Optimal auction design. Mathematics of Operations Research 6, 58–73 (1981)
17. Bhalgat, A., Goel, A., Khanna, S.: Improved approximation results for stochastic knapsack problems. In: SODA, pp. 1647–1665 (2011)
18. Bhalgat, A.: A $(2+\epsilon)$-approximation algorithm for the stochastic knapsack problem (2011) (manuscript)
19. Vohra, R.: Mechanism Design: A Linear Programming Approach. Cambridge University Press (2011)
20. Bhattacharya, S., Goel, G., Gollapudi, S., Munagala, K.: Budget constrained auctions with heterogeneous items. In: STOC, pp. 379–388 (2010)
21. Schrijver, A.: Combinatorial Optimization. Springer (2003)
22. Chekuri, C., Mydlarz, M., Shepherd, F.B.: Multicommodity demand flow in a tree and packing integer programs. ACM Transactions on Algorithms 3 (2007)
23. Bansal, N., Korula, N., Nagarajan, V., Srinivasan, A.: On k-Column Sparse Packing Programs. In: Eisenbrand, F., Shepherd, F.B. (eds.) IPCO 2010. LNCS, vol. 6080, pp. 369–382. Springer, Heidelberg (2010)

Thrifty Algorithms for Multistage Robust Optimization

Anupam Gupta[1], Viswanath Nagarajan[2], and Vijay V. Vazirani[3]

[1] Computer Science Department, Carnegie Mellon University
[2] IBM T.J. Watson Research Center
[3] College of Computing, Georgia Institute of Technology

Abstract. We consider a class of multi-stage robust covering problems, where additional information is revealed about the problem instance in each stage, but the cost of taking actions increases. The dilemma for the decision-maker is whether to wait for additional information and risk the inflation, or to take early actions to hedge against rising costs. We study the "k-robust" uncertainty model: in each stage $i = 0, 1, \ldots, T$, the algorithm is shown some subset of size k_i that completely contains the eventual demands to be covered; here $k_1 > k_2 > \cdots > k_T$ which ensures increasing information over time. The goal is to minimize the cost incurred in the *worst-case* possible sequence of revelations.

For the *multistage k-robust set cover* problem, we give an $O(\log m + \log n)$-approximation algorithm, nearly matching the $\Omega\left(\log n + \frac{\log m}{\log \log m}\right)$ hardness of approximation [4] even for $T = 2$ stages. Moreover, our algorithm has a useful "thrifty" property: it takes actions on just two stages. We show similar thrifty algorithms for multi-stage k-robust *Steiner tree*, *Steiner forest*, and *minimum-cut*. For these problems our approximation guarantees are $O(\min\{T, \log n, \log \lambda_{\max}\})$, where $\lambda_{\max}$ is the maximum inflation over all the stages. We conjecture that these problems also admit $O(1)$-approximate thrifty algorithms.

1 Introduction

This paper considers approximation algorithms for a set of multi-stage decision problems. Here, additional information is revealed about the problem instance in each stage, but the cost of taking actions increases. The decision-making algorithm has to decide whether to wait for additional information and risk the rising costs, or to take actions early to hedge against inflation. We consider the model of robust optimization, where we are told what the set of possible information revelations are, and want to minimize the cost incurred in the *worst-case* possible sequence of revelations.

For instance, consider the following multi-stage set cover problem: initially we are given a set system $(U, \mathcal{F})$. Our eventual goal is to cover some subset $A \subseteq U$ of this universe, but we don't know this "scenario" A up-front. All we know is that A can be any subset of U of size at most k. Moreover we know that on each day i, we will be shown some set A_i of size k_i, such that A_i contains the scenario

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 217–228, 2013.

A—these numbers k_i decrease over time, so that we have more information as time progresses, until $\cap_{i=0}^{T} A_i = A$. We can pick sets from $\mathcal{F}$ toward covering A whenever we want, but the costs of sets increase over time (in a specified fashion). Eventually, the sets we pick must cover the final subset A. We want to minimize the worst-case cost

$$\max_{\sigma=\langle A_1, A_2, \ldots, A_T\rangle : |A_t| = k_t \ \forall t} \text{total cost of algorithm on sequence } \sigma \tag{1.1}$$

This is a basic model for multistage robust optimization and requires minimal specification of the uncertainty sets (it only needs the cardinality bounds k_is).

Robust versions of Steiner tree/forest, minimum cut, and other covering problems are similarly defined. This tension between waiting for information vs. the temptation to buy early and beat rising costs arises even in 2-stage decision problems—here we have T stages of decision-making, making this more acute.

A comment on the kind of partial information we are modeling: in our setting we are given progressively more information about events that *will not* happen, and are implicitly encouraged (by the rising prices) to plan prudently for the (up to k) events that will indeed happen. For example, consider a farmer who has a collection of n possible bad events ("high seed prices in the growing season", $\{$"no rains by month i"$\}_{i=1}^{5}$, etc.), and who is trying to guard against up to k of these bad events happening at the end of the planning horizon. Think of k capturing how risk-averse he is; the higher the k, the more events he wants to cover. He can take actions to guard against these bad events (store seed for planting, install irrigation systems, take out insurance, etc.). In this case, it is natural that the information he gets is about the bad events that do not happen.

This should be contrasted with online algorithms, where we are only given events that do happen—namely, demands that need to be immediately and irrevocably covered. This difference means that we cannot directly use the ideas from online competitive analysis, and consequently our techniques are fairly different.[1] A second difference from online competitive analysis is, of course, in the objective function: we guarantee that the cost incurred on the worst-case sequence of revelations is approximately minimized, as opposed to being competitive to the best series of actions for every set of revelations—indeed, the rising prices make it impossible to obtain a guarantee of the latter form in our settings.

Our Results. In this paper, we give the first approximation algorithms for standard covering problems (set cover, Steiner tree and forest, and min-cut) in the model of multi-stage robust optimization with recourse. A feature of our algorithms that make them particularly desirable is that they are "thrifty": *they actually take actions in just two stages*, regardless of the number of stages T. Hence, even if T is polynomially large, our algorithms remain efficient and simple (note that the optimal decision tree has potentially exponential size even for

[1] It would be interesting to consider a model where a combination of positive and negative information is given, i.e., a mix of robust and online algorithms. We leave such extensions as future work.

constant T). For example, the set cover algorithm covers some set of "dangerous" elements right off the bat (on day 0), then it waits until a critical day t^* when it covers all the elements that can concievably still like in the final set A. We show that this set-cover algorithm is an $O(\log m + \log n)$-approximation, which almost matches the hardness result of $\Omega(\log n + \frac{\log m}{\log \log m})$ [4] for $T = 2$.

We also give thrifty algorithms for three other covering problems: Steiner tree, Steiner forest, Min-cut—again, these algorithms are easy to describe and to implement, and have the same structure:

> We find a solution in which decisions need to be made *only at two points in time*: we cover a set of dangerous elements in stage 0 (before any additional information is received), and then we cover all surviving elements at stage t^*, where $t^* = \text{argmin}_t \lambda_t k_t$.

For these problems, the approximation guarantee we can currently prove is no longer a constant, but depends on the number of stages: specifically, the dependence is $O(\min\{T, \log n, \log \lambda_{\max}\})$, where $\lambda_{\max}$ is the maximum inflation factor. While we conjecture this can be improved to a constant, we would like to emphasize that even for T being a constant more than two, previous results and techniques do not imply the existence of a constant-factor approximation algorithm, let alone the existence of a thrifty algorithm.

The definition of "dangerous" in the above algorithm is, of course, problem dependent: e.g., for set cover these are elements which cost more than Opt/k_{t^*} to cover. In general, this defintion is such that bounding the cost of the elements we cover on day t^* is immediate. And what about the cost we incur on day 0? This forms the technical heart of the proofs, which proceeds by a careful backwards induction over the stages, bounding the cost incurred in covering the dangerous elements that are still uncovered by Opt after j stages. These proofs exploit some net-type properties of the respective covering problems, and extend the results in Gupta et al. [6]. While our algorithms appear similar to those in [6], the proofs require new technical ideas such as the use of non-uniform thresholds in defining "nets" and proving properties about them.

The fact that these multistage problems have near-optimal strategies with this simple structure is quite surprising. One can show that the optimal solution may require decision-making at all stages (we show an example for set cover in the full version). It would be interesting to understand this phenomenon further. For problems other than set cover (i.e., those with a performance guarantee depending on T), can we improve the guarantees further, and/or show a tradeoff between the approximation guarantee and the number of stages we act in? These remain interesting directions for future research.

We also observe in the full version that thrifty algorithms perform poorly for multistage robust set-cover even on slight generalizations of the above "k-robust uncertainty sets". In this setting it turns out that any reasonable near-optimal solution must act on all stages. This suggests that the k-robust uncertainty sets studied in this paper are crucial to obtaining good thrifty algorithms.

Related Work. Demand-robust optimization has long been studied in the operations research literature, see eg. the survey article by Bertsimas et al. [2]

and references therein. The multistage robust model was studied in Ben-Tal et al. [1]. Most of these works involve only continuous decision variables. On the other hand, the problems considered in this paper involve making discrete decisions.

Approximation algorithms for robust optimization are of more recent vintage: all these algorithms are for two-stage optimization with discrete decision variables. Dhamdhere et al. [3] studied two-stage versions when the scenarios were *explicitly* listed, and gave constant-factor approximations for Steiner tree and facility location, and logarithmic approximations to mincut/multicut problems. Golovin et al. [5] gave $O(1)$-approximations to robust mincut and shortest-paths. Feige et al. [4] considered *implicitly* specified scenarios and introduced the k-robust uncertainty model ("scenarios are all subsets of size k"); they gave an $O(\log m \log n)$-approximation algorithm for 2-stage k-robust set cover using an LP-based approach. Khandekar et al. [8] gave $O(1)$-approximations for 2-stage k-robust Steiner tree, Steiner forest on trees and facility location, using a combinatorial algorithm. Gupta et al. [6] gave a general framework for two-stage k-robust problems, and used it to get better results for set cover, Steiner tree and forest, mincut and multicut. We build substantially on the ideas from [6].

Approximation algorithms for multistage stochastic optimization have been given in [9,7]; in the stochastic world, we are given a probability distribution over sequences, and consider the average cost instead of the worst-case cost in (1.1). However these algorithms currently only work for a constant number of stages, mainly due to the explosion in the number of potential scenarios. The current paper raises the possibility that for "simple" probability distributions, the techniques developed here may extend to stochastic optimization.

Notation. We use $[T]$ to denote $\{0, \cdots, T\}$, and $\binom{X}{k}$ to denote the collection of all k-subsets of the set X.

2 Multistage Robust Set Cover

In this section, we give an algorithm for multistage robust set cover with approximation ratio $O(\log m + \log n)$; this approximation matches the previous best approximation guarantee for two-stage robust set cover [6]. Moreover, our algorithm has the advantage of picking sets only in two stages.

The multistage robust set cover problem is specified by a set-system $(U, \mathcal{F})$ with $|U| = n$, set costs $c : \mathcal{F} \to \mathbb{R}_+$, a time horizon T, integer values $n = k_0 \geq k_1 \geq k_2 \geq \cdots \geq k_T$, and inflation parameters $1 = \lambda_0 \leq \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_T$. Define $A_0 = U$, and $k_0 = |U|$. A *scenario-sequence* $\mathbb{A} = (A_0, A_1, A_2, \ldots, A_T)$ is a sequence of $T + 1$ 'scenarios' such that $|A_i| = k_i$ for each $i \in [T]$. Here A_i is the information revealed to the algorithm on day i. The elements in $\cap_{i \leq j} A_i$ are referred to as being *active* on day j.

- On day 0, all elements are deemed active and any set $S \in \mathcal{F}$ may be chosen at the cost $c(S)$.

- On each day $j \geq 1$, the set A_j with k_j elements is revealed to the algorithm, and the active elements are $\cap_{i \leq j} A_i$. The algorithm can now pick any sets, where the cost of picking set $S \in \mathcal{F}$ is $\lambda_j \cdot c(S)$.

Feasibility requires that all the sets picked over all days $j \in [T]$ cover $\cap_{i \leq T} A_i$, the elements that are still active at the end. The goal is to minimize the worst-case cost incurred by the algorithm, the worst-case taken over all possible scenario sequences. Let Opt be this worst-case cost for the best possible algorithm; we will formalize this soon. The main theorem of this section is the following:

Theorem 1. *There is an $O(\log m + \log n)$-approximation algorithm for the T-stage k-robust set cover problem.*

The algorithm is easy to state: For any element $e \in U$, let $\mathsf{MinSet}(e)$ denote the minimum cost set in $\mathcal{F}$ that contains e. Define $\tau := \beta \cdot \max_{j \in [T]} \frac{\mathsf{Opt}}{\lambda_j \, k_j}$ where $\beta := 36 \ln m$ is some parameter. Let $j^* = \operatorname{argmin}_{j \in [T]}(\lambda_j \, k_j)$. Define the "net" $N := \{e \in U \mid c(\mathsf{MinSet}(e)) \geq \tau\}$. Our algorithm's strategy is the following:

On day zero, choose sets $\phi_0 :=$ Greedy-Set-Cover(N).
On day j^*, for any yet-uncovered elements e in A_{j^*},
    pick a min-cost set in $\mathcal{F}$ covering e.
On all other days, do nothing.

It is clear that this is a feasible strategy; indeed, all elements that are still active on day j^* are covered on that day. (In fact, it would have sufficed to just cover all the elements in $\cap_{i \leq j^*} A_i$.) Note that this strategy pays nothing on days other than 0 and j^*; we now bound the cost incurred on these two days.

Claim 2. *For any scenario-sequence $\mathbb{A}$, the cost on day j^* is at most $\beta \cdot \mathsf{Opt}$.*

Proof. The sets chosen in day j^* on sequence $\mathbb{A}$ are $\{\mathsf{MinSet}(e) \mid e \in A_{j^*} \setminus N\}$, which costs us

$$\lambda_{j^*} \sum_{e \in A_{j^*} \setminus N} c(\mathsf{MinSet}(e)) \leq \lambda_{j^*} \, |A_{j^*}| \cdot \tau = \lambda_{j^*} \, k_{j^*} \tau = \beta \cdot \mathsf{Opt}.$$

The first inequality is by the choice of N, the last equality is by τ's definition. □

Lemma 1. *The cost of covering the net N on day zero is at most $O(\log n) \cdot \mathsf{Opt}$.*

The proof of Lemma 1 will occupy the rest of this section; before we do that, note that Claim 2 and Lemma 1 complete the proof for Theorem 1. Note that while the definition of the set N requires us to know Opt, we can just run over polynomially many guesses for Opt and choose the one that minimizes the cost for day zero plus $\tau \cdot k_{j^*} \lambda_{j^*}$ (see [6] for a rigorous argument).

The proof will show that the fractional cost of covering the elements in the net N is at most Opt, and then invoke the integrality gap for the set covering LP. For the fractional cost, the proof is via a careful backwards induction on the number of stages, showing that if we mimic the optimal strategy for the first $j - 1$ steps, then the fractional cost of covering the remaining active net

elements at stage j is related to a portion of the optimal value as well. This is easy to prove for the stage T, and the claim for stage 0 exactly bounds the cost of fractionally covering the net. To write down the precise induction, we next give some notation and formally define what a strategy is (which will be used in the subsequent sections for the other problems as well), and then proceed with the proof.

Formalizing What a Strategy Means. For any collection $\mathcal{G} \subseteq \mathcal{F}$ of sets, let $\mathsf{Cov}(\mathcal{G}) \subseteq U$ denote the elements covered by the sets in $\mathcal{G}$, and let $c(\mathcal{G})$ denote the sum of costs of sets in $\mathcal{G}$. At any day i, the *state of the system* is given by the subsequence $(A_0, A_1, \ldots, A_i)$ seen thus far. Given any scenario sequence $\mathbb{A}$ and $i \in [T]$, we define $\mathbb{A}_i = (A_0, A_1, \ldots, A_i)$ to be the partial scenario sequence for days 0 through i.

A solution is a *strategy* Φ, given by a sequence of maps $(\phi_0, \phi_1, \ldots, \phi_T)$, where each one of these maps ϕ_i maps the state $\mathbb{A}_i$ on day i to a collection of sets that are picked on that day. For any scenario-sequence $\mathbb{A} = (A_1, A_2, \ldots, A_T)$, the strategy Φ does the following:

- On day 0, when all elements in U are active, the sets in ϕ_0 are chosen, and $\mathcal{G}_1 \leftarrow \phi_0$.
- At the beginning of day $i \in \{1, \cdots, T\}$, sets in $\mathcal{G}_i$ have already been chosen; moreover, the elements in $\cap_{j \le i} A_j$ are the active ones. Now, sets in $\phi_i(\mathbb{A}_i)$ are chosen, and hence we set $\mathcal{G}_{i+1} \leftarrow \mathcal{G}_i \cup \phi_i(\mathbb{A}_i)$.

The solution $\Phi = (\phi_i)_i$ is *feasible* if for every scenario-sequence $\mathbb{A} = (A_1, A_2, \ldots, A_T)$, the collection $\mathcal{G}_{T+1}$ of sets chosen at the end of day T covers $\cap_{i \le T} A_i$, i.e. $\mathsf{Cov}(\mathcal{G}_{T+1}) \supseteq \cap_{i \le T} A_i$. The cost of this strategy Φ on a fixed sequence $\mathbb{A}$ is the total effective cost of sets picked:

$$C(\Phi \mid \mathbb{A}) = c(\phi_0) + \sum_{i=1}^{T} \lambda_i \cdot c\left(\phi_i(\mathbb{A}_i)\right).$$

The objective in the robust multistage problem is to minimize $\mathsf{RobCov}(\Phi)$, the effective cost under the *worst case* scenario-sequence, namely:

$$\mathsf{RobCov}(\Phi) := \max_{\mathbb{A}} C(\Phi \mid \mathbb{A})$$

The goal is to find a strategy with least cost; for the rest of the section, fix $\Phi^* = \{\phi_i^*\}$ to be such a strategy, and let $\mathsf{Opt} = \mathsf{RobCov}(\Phi^*)$ denote the optimal objective value.

Completing Proof of Lemma 1. First, we assume that the inflation factors satisfy $\lambda_{j+1} \ge 12 \cdot \lambda_j$ for all $j \ge 0$. If the instance does not have this property, we can achieve this by merging consecutive days having comparable inflations, and lose a factor of 12 in the approximation ratio. The choice of constant 12 comes from a lemma from [6].

Lemma 2 ([6]). *Consider any instance of set cover; let $B \in \mathbb{R}_+$ and $k \in \mathbb{Z}_+$ be values such that*

- *the set of minimum cost covering any element costs* $\geq 36 \ln m \cdot \frac{B}{k}$*, and*
- *the minimum cost of fractionally covering any* k*-subset of elements* $\leq B$*.*

Then the minimum cost of fractionally covering **all** *elements is at most* $r \cdot B$*, for a value* $r \leq 12$*.*

For a partial scenario sequence $\mathbb{A}_i$ on days upto i, we use $\phi_i^*(\mathbb{A}_i)$ to denote the sets chosen *on day* i by the optimal strategy, and $\phi_{\leq i}^*(\mathbb{A}_i)$ to denote the sets chosen on days $\{0, 1, \ldots, i\}$, again by the optimal strategy.

Definition 1. *For any* $j \in [T]$ *and* $\mathbb{A}_j = (A_1, \ldots, A_j)$*, define*

$$V_j(\mathbb{A}_j) := \max_{\substack{(A_{j+1}, \cdots, A_T) \\ |A_t| = k_t \ \forall t}} \sum_{i=j}^{T} r^{i-j} \cdot c\left(\phi_i^*(\mathbb{A}_i)\right).$$

That is, $V_j(\mathbb{A}_j)$ is the worst-case cost incurred by Φ^* on days $\{j, \ldots, T\}$ conditioned on $\mathbb{A}_j$, under modified inflation factors r^{i-j} for each day $i \in \{j, \ldots, T\}$. We use this definition with r being the constant from Lemma 2. Recall that we assumed that $\lambda_i \geq r^i$.

Fact 1. *The function* $V_0(\cdot)$ *takes the empty sequence as its argument, and returns* $V_0 = \max_{\mathbb{A}} \sum_{i=0}^{T} r^i \cdot c(\phi_i^*(\mathbb{A}_i)) \leq \max_{\mathbb{A}} \sum_{i=0}^{T} \lambda_i \cdot c(\phi_i^*(\mathbb{A}_i)) = \mathsf{Opt}$*,*

For any subset $U' \subseteq U$ and any collection of sets $\mathcal{G} \subseteq \mathcal{F}$, define $\mathsf{LP}(U' \mid \mathcal{G})$ as the minimum cost of *fractionally covering* U', given all the sets in $\mathcal{G}$ at zero cost. Given any sequence $\mathbb{A}$, it will also be useful to define $\widehat{A}_j = \cap_{i \leq j} A_j$ as the active elements on day j. Our main technical lemma is the following:

Lemma 3. *For any* $j \in [T]$ *and partial scenario sequence* $\mathbb{A}_j$*, we have:*

$$\mathsf{LP}\left(N \cap \widehat{A}_j \mid \phi_{\leq j-1}^*\left(\mathbb{A}_{j-1}\right)\right) \leq V_j(\mathbb{A}_j).$$

In other words, the fractional cost of covering $N \cap \widehat{A}_i$ *(the "net" still active in stage* j*) given sets* $\phi_{\leq j-1}^*\left(\mathbb{A}_{j-1}\right)$ *for free is at most* $V_j(\mathbb{A}_j)$*.*

Before we prove this, note that for $j = 0$, the lemma implies that $LP(N) \leq V_0 \leq \mathsf{Opt}$. Since the integrality gap for the set cover LP is at most H_n (as witnessed by the greedy algorithm), this implies that the cost on day 0 is at most $O(\log n)\mathsf{Opt}$, which proves Lemma 1.

Proof. We induct on $j \in \{0, \cdots, T\}$ with $j = T$ as base case. In this case, we have a complete scenario-sequence $\mathbb{A}_T = \mathbb{A}$, and the feasibility of the optimal strategy implies that $\phi_{\leq T}^*(\mathbb{A}_T)$ completely covers $\widehat{A}_T$. So,

$$\mathsf{LP}\left(\widehat{A}_T \mid \phi_{\leq T-1}^*(\mathbb{A}_{T-1})\right) \leq c\left(\phi_T^*(\mathbb{A}_T)\right) = V_T(\mathbb{A}_T).$$

For the induction step, suppose now $j < T$, and assume the lemma for $j + 1$. Here's the roadmap for the proof: we want to bound the fractional cost to cover

elements in $N \cap \widehat{A}_j \setminus \mathsf{Cov}(\phi^*_{\leq j-1}(\mathbb{A}_{j-1}))$ since the sets $\phi^*_{\leq j-1}(\mathbb{A}_{j-1})$ are free. Some of these elements are covered by $\phi^*_j(\mathbb{A}_j)$, and we want to calculate the cost of the others—for these we'll use the inductive hypothesis. So given the scenarios $A_1, \ldots, A_j$ until day j, define

$$W_j(\mathbb{A}_j) := \max_{|B_{j+1}|=k_{j+1}} V_{j+1}(A_1, \ldots, A_j, B_{j+1}) \implies V_j(\mathbb{A}_j) = c(\phi^*_j(\mathbb{A}_j)) + r \cdot W_j(\mathbb{A}_j). \tag{2.2}$$

Let us now prove two simple subclaims.

Claim 3. $W_j(\mathbb{A}_j) \leq \mathsf{Opt}/\lambda_{j+1}$.

Proof: Suppose that $W_j(\mathbb{A}_j)$ is defined by the sequence $(A_{j+1}, \ldots, A_T)$; i.e. $W_j(\mathbb{A}_j) = \sum_{i=j+1}^{T} r^{i-j-1} \cdot c\left(\phi^*_i(\mathbb{A}_i)\right)$. Then, considering the scenario-sequence $\mathbb{A} = (A_1, \ldots, A_j, A_{j+1}, \ldots, A_T)$, we have:

$$\mathsf{Opt} \geq \sum_{i=0}^{T} \lambda_i \cdot c\left(\phi^*_i(\mathbb{A}_i)\right) \geq \sum_{i=j+1}^{T} \lambda_i \cdot c\left(\phi^*_i(\mathbb{A}_i)\right) \geq \sum_{i=j+1}^{T} \lambda_{j+1}\, r^{i-j-1} \cdot c\left(\phi^*_i(\mathbb{A}_i)\right) = \lambda_{j+1} \cdot W_j(\mathbb{A}_j).$$

The third inequality uses the assumption that $\lambda_{\ell+1} \geq r \cdot \lambda_\ell$ for all days ℓ. ◀

Claim 4. *For any A_{j+1} with $|A_{j+1}| = k_{j+1}$, we have*

$$\mathsf{LP}\left(N \cap \widehat{A}_{j+1} \mid \phi^*_{\leq j}(\mathbb{A}_j)\right) \leq W_j(\mathbb{A}_j).$$

Proof: By the induction hypothesis for $j+1$, and $V_{j+1}(\mathbb{A}_{j+1}) \leq W_j(\mathbb{A}_j)$. ◀
Now we are ready to apply Lemma 2 to complete the proof of the inductive step.

Claim 5. *Consider the set-system $\mathcal{G}$ with elements $N' := N \bigcap \left(\widehat{A}_j \setminus \mathsf{Cov}(\phi^*_{\leq j}(\mathbb{A}_j))\right)$ and the sets $\mathcal{F} \setminus \phi^*_{\leq j}(\mathbb{A}_j)$. The fractional cost of covering N' is at most $r \cdot W_j(\mathbb{A}_j)$.*

Proof: In order to use Lemma 2 on this set system, let us verify the two conditions:

1. Since $N' \subseteq N$, the cost of the cheapest set covering any $e \in N'$ is at least $\tau \geq \beta \cdot \frac{\mathsf{Opt}}{\lambda_{j+1}\, k_{j+1}} \geq \beta \cdot \frac{W_j(\mathbb{A}_j)}{k_{j+1}}$ using the definition of the threshold τ, and Claim 3; recall $\beta = 36 \ln m$.
2. For every $X \subseteq N'$ with $|X| \leq k_{j+1}$, the minimum cost to fractionally cover X in $\mathcal{G}$ is at most $W_j(\mathbb{A}_j)$. To see this, augment X arbitrarily to form A_{j+1} of size k_{j+1}; now Claim 4 applied to A_{j+1} implies that the fractional covering cost for $N \cap \widehat{A}_{j+1} = N \cap \left(A_{j+1} \cap \widehat{A}_j\right)$ in $\mathcal{G}$ is at most $W_j(\mathbb{A}_j)$; since $X \subseteq N' \subseteq N \cap \widehat{A}_j$ and $X \subseteq A_{j+1}$ the covering cost for X in $\mathcal{G}$ is also at most $W_j(\mathbb{A}_j)$.

We now apply Lemma 2 on set-system $\mathcal{G}$ with parameters $B := W_j(\mathbb{A}_j)$ and $k = k_{j+1}$ to infer that the minimum cost to fractionally cover N' using sets from $\mathcal{F} \setminus \phi^*_{\leq j}(\mathbb{A}_j)$ is at most $r \cdot W_j(\mathbb{A}_j)$. ◀

To fractionally cover $N \cap \widehat{A}_j$, we can use the fractional solution promised by Claim 5, and integrally add the sets $\phi^*_j(\mathbb{A}_j)$. This implies that

$$\mathsf{LP}\left(N \cap \widehat{A}_j \mid \phi^*_{\le j-1}(\mathbb{A}_{j-1})\right) \le c(\phi^*_j(\mathbb{A}_j)) + r \cdot W_j(\mathbb{A}_j) = V_j(\mathbb{A}_j),$$

where the last equality follows from (2.2). This completes the induction and proves Lemma 3.

3 Multistage Robust Minimum Cut

We now turn to the multistage robust min-cut problem, and show:

Theorem 6. *There is an $O(\min\{T, \log n, \log \lambda_{max}\})$-approximation algorithm for T-stage k-robust minimum cut.*

In this section we prove an $O(T)$-approximation ratio where T is the number of stages; in the full version we show that simple scaling arguments can be used to ensure T is at most $\min\{\log n, \log \lambda_{max}\}$, yielding Theorem 6. Unlike set cover, the guarantee here depends on the number of stages. Here is the high-level reason for this additional loss: in each stage of an optimal strategy for set cover, any element was either completely covered or left completely uncovered—there was no partial coverage. However in min-cut, the optimal strategy could keep whittling away at the cut for a node in each stage. The main idea to deal with this is to use a stage-dependent definition of "net" in the inductive proof (see Lemma 6 for more detail), which in turn results in an $O(T)$ loss.

The input consists of an undirected graph $G = (U, E)$ with edge-costs $c : E \to \mathbb{R}_+$ and root ρ. For any subset $U' \subseteq U$ and subgraph H of G, we denote by $\mathsf{MinCut}_H(U')$ the minimum cost of a cut separating U' from ρ in H. If no graph is specified then it is relative to the original graph G. Recall that a scenario sequence $\mathbb{A} = (A_0, A_1, \ldots, A_T)$ where each $A_i \subseteq U$ and $|A_i| = k_i$, and we denote the partial scenario sequence $(A_0, A_1, \ldots, A_j)$ by $\mathbb{A}_j$.

We will use notation developed in Section 2. Let the optimal strategy be $\Phi^* = \{\phi^*_j\}_{j=0}^T$, where now $\phi^*_j(\mathbb{A}_j)$ maps to a set of edges in G to be cut in stage j. The feasibility constraint is that $\phi^*_{\le T}(\mathbb{A}_T)$ separates the vertices in $\cap_{i \le T} A_i$ from the root ρ. Let the cost of the optimal solution be $\mathsf{Opt} = \mathsf{RobCov}(\Phi^*)$.

Again, the algorithm depends on showing a near-optimal two-stage strategy: define $\tau := \beta \cdot \max_{j \in [T]} \frac{\mathsf{Opt}}{\lambda_j\, k_j}$, where $\beta = 50$. Let $j^* = \operatorname{argmin}_{j \in [T]}(\lambda_j\, k_j)$. Let the "net" $N := \{v \in U \mid \mathsf{MinCut}(v) > 2T \cdot \tau\}$. The algorithm is:

> On day 0, delete $\phi_0 := \mathsf{MinCut}(N)$ to separate the "net" N from ρ.
> On day j^*, for each vertex u in $A_{j^*} \setminus N$, delete a minimum u-ρ cut in G.
> On all other days, do nothing.

Again, it is clear that this strategy is feasible: all vertices in $\cap_{i \le T} A_i$ are either separated from the root on day 0, or on day j^*. Moreover, the effective cost of the cut on day j^* is at most $\lambda_{j^*} \cdot 2T\tau \cdot |A_{j^*}| = 2\beta T\, \mathsf{Opt} = O(T) \cdot \mathsf{Opt}$. Hence it suffices to show the following:

Lemma 4. *The min-cut separating N from the root ρ costs at most $O(T) \cdot \mathsf{Opt}$.*

Again, the proof is via a careful induction on the stages. Loosely speaking, our induction is based on the following: amongst scenario sequences $\mathbb{A}$ containing any fixed "net" vertex $v \in N$ (i.e. $v \in \cap_{i \le T} A_i$) the optimal strategy must reduce the min-cut of v (in an average sense) by a factor $1/T$ in some stage.

The proof of Lemma 4 again depends on a structural lemma proved in [6]:

Lemma 5 ([6]). *Consider any instance of minimum cut in an undirected graph with root ρ and terminals X; let $B \in \mathbb{R}_+$ and $k \in \mathbb{Z}_+$ be values such that*

- *the minimum cost cut separating ρ and x costs $\ge 10 \cdot \frac{B}{k}$, for every $x \in X$.*
- *the minimum cost cut separating ρ and L is $\le B$, for every $L \in \binom{X}{k}$.*

Then the minimum cost cut separating ρ and **all** *terminals X is at most $r \cdot B$, for a value $r \le 10$.*

In this section, we assume $\lambda_{j+1} \ge 10 \cdot \lambda_j$ for all $j \in [T]$. Recall the quantity $V_j(\mathbb{A}_j)$ from Definition 1:

$$V_j(\mathbb{A}_j) := \max_{\substack{(A_{j+1},\cdots,A_T) \\ |A_t|=k_t \ \forall t}} \sum_{i=j}^{T} r^{i-j} \cdot c\left(\phi_i^*(\mathbb{A}_i)\right)$$

where $r := 10$ from Lemma 5. Since $\lambda_i \ge r^i$, it follows that $V_0 \le \mathsf{Opt}$. The next lemma is now the backwards induction proof that relates the cost of cutting subsets of the net N to the V_js. This finally bounds the cost of separating the entire net N from ρ in terms of $V_0 \le \mathsf{Opt}$. Given any $\mathbb{A}_j$, recall that $\widehat{A}_j = \cap_{i \le j} A_i$.

Lemma 6. *For any $j \in [T]$ and partial scenario sequence $\mathbb{A}_j$,*

- *if $H := G \setminus \phi^*_{\le j-1}(\mathbb{A}_{j-1})$ (the residual graph in OPT's run at the beginning of stage j), and*
- *$N_j := \{v \in \widehat{A}_j \mid \mathsf{MinCut}_H(v) > (2T - j) \cdot \tau\}$ (the "net" elements)*

then $\mathsf{MinCut}_H(N_j) \le 5T \cdot V_j(\mathbb{A}_j)$.

Before we prove the lemma, note that when we set $j = 0$ the lemma claims that in G, the min-cut separating $N_0 = N$ from ρ costs at most $5T \cdot V_0 \le O(T)\,\mathsf{Opt}$, which proves Lemma 4. Hence it suffices to prove Lemma 6. Note the difference from the induction used for set-cover: the thresholds used to define nets is non-uniform over the stages.

Proof. We induct on $j \in \{0, \cdots, T\}$. The base case is $j = T$, where we have a complete scenario-sequence $\mathbb{A}_T$: by feasibility of the optimum, $\phi^*_{\le T}$ cuts $\widehat{A}_T \supseteq N_T$ from r in G. Thus the min-cut for N_T in $G \setminus \phi^*_{\le T-1}(\mathbb{A}_{T-1})$ costs at most $c(\phi^*_T(\mathbb{A}_T)) = V_T(\mathbb{A}_T) \le 5T \cdot V_T(\mathbb{A}_T)$.

Now assuming the inductive claim for $j + 1 \le T$, we prove it for j. Let $H = G \setminus \phi^*_{\le j-1}(\mathbb{A}_{j-1})$ be the residual graph after $j - 1$ stages, and let $H' = H \setminus \phi^*_j(\mathbb{A}_j)$ the residual graph after j stages. Let us divide up N_j into two parts, $N_j^1 := \{v \in N_j \mid \mathsf{MinCut}_{H'}(v) > (2T - j - 1) \cdot \tau\}$ and $N_j^2 = N_j \setminus N_j^1$, and bound the mincut of the two parts in H' separately.

Claim 7. $\mathsf{MinCut}_{H'}(N_j^2) \le 4T \cdot c(\phi_j^*(\mathbb{A}_j))$.

Proof: Note that the set N_j^2 consists of the points that have "high" mincut in the graph H after $j-1$ stages, but have "low" mincut in the graph H' after j stages. For these we use a *Gomory-Hu tree*-based argument like that in [5]. Formally, let $t := (2T-j)\cdot\tau \le 2T\tau$. Hence for every $u \in N_j^2$, we have:

$$\mathsf{MinCut}_H(u) > t \qquad \text{and} \qquad \mathsf{MinCut}_{H'}(u) \le \left(1 - \frac{1}{2T}\right) t. \tag{3.3}$$

Consider the Gomory-Hu (cut-equivalent) tree $\mathcal{T}(H')$ on graph H', and root it at ρ. For each vertex $u \in N_j^2$, let $(X_u, \overline{X}_u)$ denote the minimum ρ-u cut in $\mathcal{T}(H')$, where $u \in X_u$ and $\rho \notin X_u$. Pick a subset $N' \subseteq N_j^2$ such that the union of their respective min-cuts in $\mathcal{T}(H')$ separate all of N_j^2 from ρ and their corresponding sets X_u are disjoint—the set of cuts in tree $\mathcal{T}(H')$ closest to the root ρ gives such a collection. Define $F := \cup_{u\in N'} \partial_{H'}(X_u)$; this is a feasible cut in H' separating N_j^2 from ρ.

Note that (3.3) implies that for all $u \in N_j^2$ (and hence for all $u \in N'$), we have

(i) $c(\partial_{H'}(X_u)) \le (1 - \frac{1}{2T}) \cdot t$ since X_u is a minimum ρ-u cut in H', and
(ii) $c(\partial_H(X_u)) \ge t$ since it is a feasible ρ-u cut in H.

Thus $c(\partial_{H\setminus H'}(X_u)) = c(\partial_H(X_u)) - c(\partial_{H'}(X_u)) \ge \frac{1}{2T}\, t \ge \frac{1}{2T} \cdot c(\partial_{H'}(X_u))$. So

$$c(\partial_{H'}(X_u)) \le 2T \cdot c(\partial_{H\setminus H'}(X_u)) \quad \text{for all } u \in N', \tag{3.4}$$

Consequently,

$$c(F) \le \sum_{u\in N'} c(\partial_{H'}(X_u)) \le 2T \cdot \sum_{u\in N'} c(\partial_{H\setminus H'}(X_u)) \le 4T \cdot c(H\setminus H') = 4T \cdot c(\phi_j^*(\mathbb{A}_j)).$$

The first inequality follows from subadditivity, the second from (3.4), the third uses disjointness of $\{X_u\}_{u\in N'}$, and the equality follows from $H \setminus H' = \phi_j^*(\mathbb{A}_j)$. Thus $\mathsf{MinCut}_{H'}(N_j^2) \le 4T \cdot c(\phi_j^*(\mathbb{A}_j))$. ◀

Now to bound the cost of separating N_j^1 from ρ. Recall the quantity $W_j(\mathbb{A}_j)$ from (2.2),

$$W_j(\mathbb{A}_j) := \max_{|B_{j+1}|=k_{j+1}} V_{j+1}(A_1, \ldots, A_j, B_{j+1}).$$

and that $V_j(\mathbb{A}_j) = \phi_j^*(\mathbb{A}_j) + r \cdot W_j(\mathbb{A}_j)$.

Claim 8. $\mathsf{MinCut}_{H'}(N_j^1) \le 5r\,T \cdot W_j(\mathbb{A}_j)$.

Proof: The definition of N_j^1 implies that for each $u \in N_j^1$ we have:

$$\mathsf{MinCut}_{H'}(u) \;>\; (2T-j-1)\tau \;\ge\; T\,\tau \;\ge\; T\cdot\beta\,\frac{\mathsf{Opt}}{\lambda_{j+1}\,k_{j+1}} \;\ge\; \beta T\,\frac{W_j(\mathbb{A}_j)}{k_{j+1}}, \tag{3.5}$$

where the last inequality is by Claim 3. Furthermore, for any k_{j+1}-subset $L \subseteq N_j^1 \subseteq A_j$ we have:

$$\mathsf{MinCut}_{H'}(L) \;\le\; 5T \cdot V_{j+1}(A_1, \cdots, A_j, L) \;\le\; 5T \cdot W_j(\mathbb{A}_j). \tag{3.6}$$

The first inequality is by applying the induction hypothesis to $(A_1, \cdots, A_j, L)$; induction can be applied since L is a "net" for this partial scenario sequence (recall $L \subseteq N_j^1$ and the definition of N_j^1). The second inequality is by definition of $W_j(\mathbb{A}_j)$.

Now we apply Lemma 5 on graph H' with terminals $X = N_j^1$, bound $B = 5T \cdot W_j(\mathbb{A}_j)$, and $k = k_{j+1}$. Since $\beta = 50$, equations (3.5)-(3.6) imply that the conditions in Lemma 5 are satisfied, and we get $\mathsf{MinCut}_{H'}(N_j^1) \leq 5r\,T \cdot W_j(\mathbb{A}_j)$ to prove Claim 8. ◀

Finally,

$$\begin{aligned}\mathsf{MinCut}_H(N_j) \;&\leq\; \mathsf{MinCut}_{H'}(N_j^1) + \mathsf{MinCut}_{H'}(N_j^2) + c(\phi_j^*(\mathbb{A}_j)) \\ &\leq\; 5rT \cdot W_j(\mathbb{A}_j) + 4T\,c(\phi_j^*(\mathbb{A}_j)) + c(\phi_j^*(\mathbb{A}_j)) \;\leq\; 5T \cdot V_j(\mathbb{A}_j).\end{aligned}$$

The first inequality uses subadditivity of the cut function, the second uses Claims 7 and 8, and the third uses $T \geq 1$ and definition of $W_j(\mathbb{A}_j)$. This completes the proof of the inductive step, and hence of Lemma 6.

Acknowledgments. We thank F. B. Shepherd, A. Vetta, and M. Singh for their generous hospitality during the initial stages of this work.

References

1. Ben-Tal, A., Goryashko, A., Guslitzer, E., Nemirovski, A.: Adjustable robust solutions of uncertain linear programs. Mathematical Programming 99(2), 351–376 (2004)
2. Bertsimas, D., Brown, D.B., Caramanis, C.: Theory and applications of robust optimization. SIAM Review 53(3), 464–501 (2011)
3. Dhamdhere, K., Goyal, V., Ravi, R., Singh, M.: How to pay, come what may: Approximation algorithms for demand-robust covering problems. In: FOCS, pp. 367–378 (2005)
4. Feige, U., Jain, K., Mahdian, M., Mirrokni, V.S.: Robust Combinatorial Optimization with Exponential Scenarios. In: Fischetti, M., Williamson, D.P. (eds.) IPCO 2007. LNCS, vol. 4513, pp. 439–453. Springer, Heidelberg (2007)
5. Golovin, D., Goyal, V., Ravi, R.: Pay Today for a Rainy Day: Improved Approximation Algorithms for Demand-Robust Min-Cut and Shortest Path Problems. In: Durand, B., Thomas, W. (eds.) STACS 2006. LNCS, vol. 3884, pp. 206–217. Springer, Heidelberg (2006)
6. Gupta, A., Nagarajan, V., Ravi, R.: Thresholded Covering Algorithms for Robust and Max-min Optimization. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010, Part I. LNCS, vol. 6198, pp. 262–274. Springer, Heidelberg (2010) Full version: CoRR abs/0912.1045
7. Gupta, A., Pál, M., Ravi, R., Sinha, A.: What About Wednesday? Approximation Algorithms for Multistage Stochastic Optimization. In: Chekuri, C., Jansen, K., Rolim, J.D.P., Trevisan, L. (eds.) APPROX and RANDOM 2005. LNCS, vol. 3624, pp. 86–98. Springer, Heidelberg (2005)
8. Khandekar, R., Kortsarz, G., Mirrokni, V.S., Salavatipour, M.R.: Two-Stage Robust Network Design with Exponential Scenarios. In: Halperin, D., Mehlhorn, K. (eds.) ESA 2008. LNCS, vol. 5193, pp. 589–600. Springer, Heidelberg (2008)
9. Swamy, C., Shmoys, D.B.: Sampling-based approximation algorithms for multistage stochastic optimization. SIAM J. Comput. 41(4), 975–1004 (2012)

Shallow-Light Steiner Arborescences with Vertex Delays

Stephan Held and Daniel Rotter

Research Institute for Discrete Mathematics, University of Bonn
{held,rotter}@or.uni-bonn.de

Abstract. We consider the problem of constructing a Steiner arborescence broadcasting a signal from a root r to a set T of sinks in a metric space, with out-degrees of Steiner vertices restricted to 2. The arborescence must obey delay bounds for each r-t-path ($t \in T$), where the path delay is imposed by its total edge length and its inner vertices.

We want to minimize the total length. Computing such arborescences is a central step in timing optimization of VLSI design where the problem is known as the repeater tree problem [1,5]. We prove that there is no constant factor approximation algorithm unless $P = NP$ and develop a bicriteria approximation algorithm trading off signal speed (shallowness) and total length (lightness). The latter generalizes results of [8,3], which do not consider vertex delays. Finally, we demonstrate that the new algorithm improves existing algorithms on real world VLSI instances.

1 Introduction

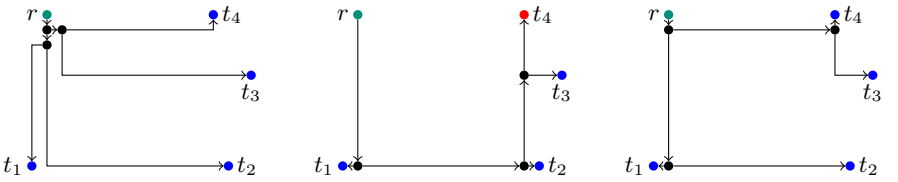
The input to our problem is a set T of sink vertices and a root r, which are embedded into a metric space (M, dist) by a function $p : T \cup \{r\} \to M$. We are mostly interested in the cases where (M, dist) is the metric closure of a weighted graph or $\mathbb{R}^2$ with the l_1-norm, but our main algorithm works in general.

A solution, which we also call *topology for* $T + r$, consists of an arborescence A rooted at r such that the set of leaves of A is exactly the set T, together with an extension of p to the internal vertices (*Steiner vertices*) in $V(A) \setminus (T \cup \{r\})$. We require that the root r has exactly one outgoing edge and each Steiner vertex has exactly two outgoing edges. This structural restriction is tributed to the delay model we use. By splitting and contracting vertices and orienting edges, a Steiner tree for $T \cup \{r\}$ in (M, dist) can be transformed in linear time into a topology for $T + r$ with the same total edge length and vice versa.

To shorten the notation, we use $A = (A, p)$ to denote an arborescence A together with a placement function $p : V(A) \to M$.
Also, we define $\mathit{dist}(v, w) := \mathit{dist}(p(v), p(w))$ for all vertices $v, w \in V(A)$ associated with placements $p(v), p(w) \in M$.

The topology might be considered as a broadcast network that delivers a signal originating in r to each sink $t \in T$. The cost of a topology A is given by its total edge length $\mathit{cost}(A) := \sum_{e=(v,w)\in E(A)} \mathit{dist}(v, w)$. We assume that there is a constant $b \in \mathbb{R}_+$ that specifies a time penalty for traversing a Steiner vertex,

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 229–241, 2013.



(a) A shallow/fast topology with high cost. (b) A light/cheap topology with $delay_A(t_4)$ large. (c) A light and shallow topology.

Fig. 1. Tradeoff between shallowness and lightness of a topology for r and $T = \{t_1, t_2, t_3, t_4\}$ embedded into $(\mathbb{R}^2, l_1)$

the time needed for splitting the signal. Following the model in [1], the *delay* of an r-v-path ($v \in V(A)$) in A is given by its length plus b times the number of bifurcations on the path:

$$delay_A(v) := \left(\sum_{e=(v',w') \in E\left(A_{[r,v]}\right)} dist(v', w') \right) + b \cdot (|E(A_{[r,v]})| - 1). \quad (1)$$

Restricting the out-degrees of Steiner vertices to 2 prevents saving vertex delays by using high-fanout Steiner vertices. Each sink $t \in T$ is associated with a delay bound or *required arrival time* $rat(t) \in \mathbb{R}_+$. A topology A meets the required arrival times if its *worst slack* is non-negative:

$$wsl(A) := \min_{t \in T} \{rat(t) - delay_A(t)\} \geq 0.$$

Figure 1 shows examples of topologies.

In the *shallow light Steiner arborescence problem with vertex delays* (SLAP) we wish to compute a topology A for $T + r$ and positions $p(s) \in M$ for each Steiner point s such that $wsl(A) \geq 0$ and $cost(A)$ is as small as possible or to decide that a topology with $wsl(A) \geq 0$ does not exist.

The existence is easy to check. As shown in [1], we can always place all Steiner points at the root location $p(r)$ as indicated in Figure 1(a). This way all paths are shortest possible w.r.t *dist*. For each sink $t \in T$ an upper bound $bif(t)$ for the number of bifurcations on an r-t-path is imposed by $rat(t)$ and $dist(r, t)$, the minimum possible delay for covering the distance:

$$bif(t) := \left\lfloor \frac{rat(t) - dist(r,t)}{b} \right\rfloor \text{ if } b > 0 \text{ and } bif(t) := |T| - 1 \text{ if } b = 0. \quad (2)$$

For the case $b = 0$, where the delays depend only on the distances, a topology with non-negative worst slack exists if and only if $rat(t) \geq dist(r, t)$ for all $t \in T$. Any topology consisting of shortest paths would be feasible.

Otherwise, $b > 0$. Since the subtree rooted at the unique child of r is a binary tree in which each sink $t \in T$ has depth exactly $|E(A_{[r,t]})| - 1$, by Kraft's inequality [9] a topology with non-negative worst slack exists if and only if

$$\sum_{t \in T} 2^{-bif(t)} \leq 1. \tag{3}$$

Such a topology can be computed in $\mathcal{O}(|T| \log |T|)$ time using Huffman-coding [6], which iteratively takes two vertices with maximum *bif*-value and replaces them by a Steiner vertex with position $p(r)$ and a suitable required arrival time.

SLAP in $(\mathbb{R}^2, l_1)$ is known as the *repeater tree problem* [1,5]. In [1] a greedy algorithm was proposed that starts with an empty topology A and adds sinks in non-increasing order of their *bif*-value, subdividing an edge in A for which the resulting topology maximizes the worst slack minus the cost weighted by some adjustment factor. Although it proved to be effective in practice, theoretical results are only known for the cases $rat \equiv \infty$, where it provides a $\frac{3}{2}$-approximation [7], and $dist \equiv 0$, where the worst slack is maximized.

For the special case $b = 0$ the first significant result was given in [2], providing a bicriteria approximation achieving path lengths within $(1+\epsilon) \cdot \max\{dist(r,t) : t \in T\}$ and cost within $1 + \frac{2}{\epsilon}$ times the cost of a minimum spanning tree. By a modification of this algorithm, [8] achieved a length of at most $(1+\epsilon) \cdot dist(r,t)$ for each r-t-path ($t \in T$).

In [3] the cost bound was improved further to $3 + 2 \cdot \lceil \log(\frac{2}{\epsilon}) \rceil$ using Steiner vertices. However, the Steiner points are not embedded into $(M, dist)$ but into an extended metric space, making this improvement less interesting for practical problems. For the Euclidean space $(\mathbb{R}^2, l_2)$ [3] construct instances for which the cost of each topology A with $delay_A(t) \leq (1+\epsilon) \cdot rat(t)$ ($t \in T$) varies from the cost of a minimum spanning tree by a factor of $\Omega(\frac{1}{\epsilon})$ for each $\epsilon > 0$.

Our problem is loosely related to delay or hop constrained tree problems (see [4,12] for recent references), where edge costs are unrelated to their lengths, which makes it more difficult to trade off delays and costs. [13] have proved *NP*-hardness of computing a rectilinear Steiner tree rooted at a vertex r with minimum cost in which all paths are shortest paths. This problem, also known as the *Rectilinear Steiner Arborescence Problem*, has a 2-factor approximation algorithm (see [11]). [10] have shown that the hop constrained tree problem in graphs cannot be approximated within a constant factor unless $P = NP$. But the proof uses non-metric edge weights violating the triangle inequality and does not bound out-degrees, so that it does not apply to our problem.

In Section 2 we prove that there is no constant factor approximation algorithm for SLAP unless $P = NP$. Then, in Section 3 we develop a new bicriteria algorithm for SLAP, generalizing algorithms from [8,3] for $b = 0$. For $b = 0$ and $(M, dist) = (\mathbb{R}^2, l_1)$ we adapt the algorithm of [3] so that Steiner vertices are embedded into $(\mathbb{R}^2, l_1)$ obtaining an algorithm which guarantees bounds of $delay_A(t) \leq (1+\epsilon) \cdot rat(t)$ for all $t \in T$ and $cost(A) \leq (2 + \lceil \log(\frac{2}{\epsilon}) \rceil) \cdot cost(A_c)$, where A_c is an initial short topology. Finally, we demonstrate in Section 4 that the new algorithm achieves significant improvements over the industrially employed algorithm from [1] on practical instances from VLSI design.

2 Non-approximability

Although the question of existence of a feasible solution is easy to answer, it is very hard to find an optimum solution as the next theorem shows.

Theorem 1. *There is no constant factor approximation algorithm for SLAP unless P=NP.*

Proof. Assume, there is an approximation algorithm with approximation ratio $\alpha > 1$. We use this algorithm to decide an *NP*-complete variant of SATISFIABILITY. Let $\mathcal{C}$ be a set of clauses over variables $X = \{x_1, \ldots, x_n\}$ where $n = 2^k$ for some $k \in \mathbb{N}$ and each literal appears in at most two clauses. It is *NP*-complete to decide if a set of clauses of this special form is satisfiable (the proof immediately follows from [14]). Furthermore, we may assume that $|\mathcal{C}| \leq 2 \cdot n$.

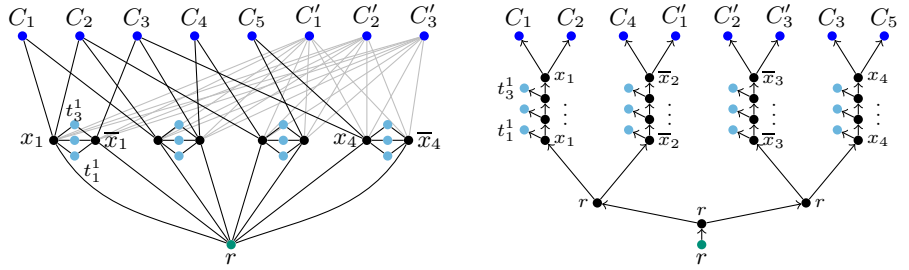
Define $\overline{X} := \{\overline{x}_1, \ldots, \overline{x}_n\}$ and let $\mathcal{C}'$ be a set of $2n - |\mathcal{C}|$ elements. We define $(M, dist)$ as the metric closure of an undirected graph G which is defined as follows (see also Figure 2(a)). Let $m := \lceil 6\alpha n - 4n - 2k \rceil + 1$, $\epsilon := \frac{1}{2(\alpha-1)n}$, and $V(G) = \mathcal{C} \cup \mathcal{C}' \cup \{r\} \cup X \cup \overline{X} \cup \{t^i_j : i \in \{1, \ldots, n\}, j \in \{1, \ldots, m\}\}$. Include edges

- $\{r, \lambda\}$ for all $\lambda \in X \cup \overline{X}$ with length 1,
- $\{\lambda, C\}$ for all $\lambda \in X \cup \overline{X}$, $C \in \mathcal{C}$ such that $\lambda \in C$ with length 1,
- $\{\lambda, C'\}$ for all $\lambda \in X \cup \overline{X}$, $C' \in \mathcal{C}'$ with length 1,
- $\{\lambda, t^i_j\}$ for all $\lambda \in X \cup \overline{X}$ s.t. $\lambda = x_i$ or $\lambda = \overline{x}_i$, $j \in \{1, \ldots, m\}$ with length ϵ.

Let $T := \mathcal{C} \cup \mathcal{C}' \cup \{t^i_j : i \in \{1, \ldots, n\}, j \in \{1, \ldots, m\}\}$, $b = 1$, $p(t) = t$ for all $t \in T \cup \{r\}$, $rat(C) = k + m + 3$ for $C \in \mathcal{C} \cup \mathcal{C}'$, $rat(t^i_j) = 1 + \epsilon + j + k$ for all $i \in \{1, \ldots, n\}, j \in \{1, \ldots, m\}$. Note that $\sum_{t \in T} 2^{-bif(t)} = 1$ and by (3), a topology with non-negative worst slack for the constructed instance exists. Because equality holds, the number of Steiner vertices on an r-t-path is uniquely determined by $bif(t)$ for every $t \in T$ and a feasible solution cannot have a Steiner vertex simultaneously at $p(x_i)$ and $p(\overline{x}_i)$ for $i = 1, \ldots, n$. The following claim proves the theorem.

Claim: If $\mathcal{C}$ is satisfiable, a topology for $T + r$ with non-negative worst slack and cost at most $3n + nm \cdot \epsilon$ exists (see Figure 2(b)). Otherwise, each topology with non-negative worst slack has cost at least $2n + k + (1 + \epsilon n) \cdot m > \alpha \cdot (3n + nm \cdot \epsilon)$.

The idea is to show that in every topology with non-negative worst slack the set $\mathcal{C} \cup \mathcal{C}'$ must be arranged pairwise such that each pair is connected to a common Steiner point s with $|E(A_{[r,s]})| = k + m$. If $\mathcal{C}$ is not satisfiable we have to place one of these Steiner points and all of its predecessors at $p(r)$. The total cost of a topology gets large that way. If $\mathcal{C}$ is satisfiable we find a pairing such that we can place all of these Steiner points and m of its predecessors at a position of a true literal. We obtain a short topology. A detailed proof will be provided in a full version of this paper. □



(a) The graph G which defines $(M, dist)$ in the proof of Theorem 1 ($m = 3$).

(b) Light topology with $wsl = 0$. Labels indicate positions. $x_1 = x_4 =$ true, $x_2 = x_3 =$ false satisfies all clauses.

Fig. 2. Graph G and a light topology with non-negative worst slack in the proof of Theorem 1. The corresponding instance of SATISFIABILITY is $X = \{x_1, x_2, x_3, x_4\}$, $C_1 = \{x_1, x_2\}$, $C_2 = \{x_1, x_2, x_3\}$, $C_3 = \{\overline{x}_1, \overline{x}_2, x_4\}$, $C_4 = \{\overline{x}_2, x_3\}$, $C_5 = \{\overline{x}_3, x_4\}$. For simplicity, we chose $m = 3$.

3 Bicriteria Approximation Algorithms

We have seen there is no algorithm with a constant approximation ratio unless *P=NP*. We now relax the constraint that the computed topology should have a non-negative worst slack. Instead, we wish to obtain a bicriteria approximation algorithm, i.e. an algorithm that computes a topology A such that $cost(A)$ is at most a factor of β away from optimum while $delay_A(t) \leq \alpha \cdot rat(t)$ for each sink t and constants $\alpha, \beta \geq 1$.

3.1 An Algorithm for General Metric Spaces

Let $\epsilon > 0$ and let $T + r, p, rat, b$ be an instance of SLAP for which a topology with non-negative worst slack exists. The following algorithm is inspired by [8], which was developed for $b = 0$.

Algorithm 1. Let A_c be any (light) topology for $T + r$. A_c can be obtained from an approximate minimum Steiner tree by directing all edges away from r and applying local transformations such that all degree constraints are met.

Let r' be the successor of r in A_c and let $\overleftrightarrow{A_c}$ be the directed graph with vertex set $V(A_c)$ and edge set $E(A_c) \cup \overleftarrow{E}(A_c)$ where $\overleftarrow{E}(A_c) := \{(w, v) : (v, w) \in E(A_c)\}$. Note that $\overleftrightarrow{A_c}$ is Eulerian. The idea is to perform an Eulerian walk in $\overleftrightarrow{A_c} - r$ starting at r'. During the walk we keep track of a branching B and an estimate $d(v)$ on the delay of the r-v path in the final topology for each vertex v. Initially, set $B := A_c - r$ (see Fig. 3(a) for an example) and $d(r') := dist(r, r')$. Throughout the whole algorithm, for vertices $v \in V(B)$ that are not roots ($|\delta_B^-(v)| = 1$) we recursively set $d(v) := d(u) + b + dist(u, v)$, where $(u, v) \in E(B)$. By construction, each forward edge $(v, w) \in E(A_c)$ is visited prior

to its backward counterpart $(w,v) \in \overleftarrow{E}(A_c)$ and when (w,v) is visited, the tour finished visiting vertices in the subtree of A_c rooted at w.

When we visit a forward edge $(v,w) \in E(A_c)$, we do nothing if $w \in V(A_c) \setminus T$. Otherwise, $w \in T$ is a leaf and we check whether

$$d(w) > (1+\epsilon) \cdot rat(w). \tag{I}$$

If this is the case, we delete the edge (v,w). The sink w becomes a new root of B and we set $d(w) = dist(r,w) + b \cdot bif(w)$ (see Fig. 3(a) and 3(b)).

When we visit a backward edge $(w,v) \in \overleftarrow{E}(A_c)$, we check whether it is better to merge the (current) subtree of B rooted at v with the connected component of B containing w. More precisely, we check whether

$$d(v) > d(w) + dist(w,v) + b. \tag{II}$$

Note that by the definition of d, this can only be the case if the edge (v,w) is not in B anymore. If condition (II) is true, we

- delete the edge currently entering v (unless v is a root of B),
- subdivide the edge currently entering w by a Steiner vertex placed at $p(w)$ and connect it to w and v if w is not a root of B,
- create a new Steiner point s placed at $p(w)$, connect it to v and w, and set $d(s) = d(w)$ if w is a root (see Fig. 3(c) for an illustration). The vertex s is the new root of the connected component of B containing v and w.

When we have finished our Eulerian walk, we make sure that $|\delta^+(s)| = 2$ for all $s \in V(B) \setminus T$. If $|\delta^+(s)| + |\delta^-(s)| \leq 1$ for a Steiner point s, we delete it. If $s \in V(B) \setminus T$ has both out-degree and in-degree equal to one, delete it and connect its predecessor with its successor.

Let T' be the set of roots of connected components of B (e.g. boxed vertices in Fig. 3(c)). Note that $r' \in T'$ unless there are no sinks left in the connected component of B containing r' after the Eulerian walk. Set $rat'(t') := d(t') + b$ for $t' \in T'$. Let $bif' : T' \to \mathbb{N}$ be defined analogously to bif in (2).

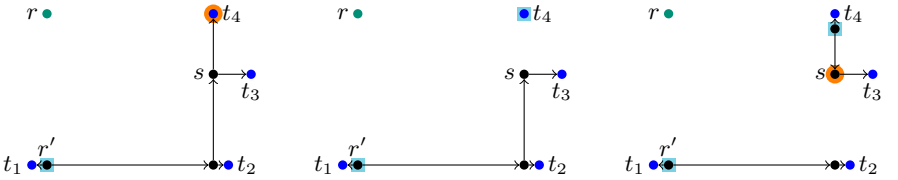
We have $\sum_{t' \in T'} 2^{-bif'(t')} \leq \frac{1}{2} + \frac{1}{2} \cdot \sum_{t \in T} 2^{-bif(t)} \leq 1$ and hence, a topology A' with non-negative worst slack for the instance $T' + r, p, rat', b$ exists as (3) holds. We do not need to be overly careful in bounding the cost of A' and can place all Steiner vertices at $p(r)$. Thus, we can compute A' by Huffman-coding or the greedy-algorithm from [1] as described in the introduction.

Finally, the algorithm returns $A := A' + B$ (Fig. 1(c) in our example).

Remark 1. in the case $(M, dist) = (\mathbb{R}^2, l_1)$, we can use an improved version of Huffman-coding. Instead of placing a Steiner vertex s replacing sinks t, t' in T at position $p(r)$, we may also place them at the median of r, t, and t':

$$p(s) = (\text{median}(p(r)_x, p(t)_x, p(t')_x), \text{median}(p(r)_y, p(t)_y, p(t')_y)).$$

Whenever the cardinality of the set of sinks with maximum bif-value, $T_{\max bif}$, is larger than 2, we can compute a matching in $T_{\max bif}$ with low cost and



(a) Initial branching B, when A_c is given by Fig. 1(b). When visiting $(s, t_4) \in E(A_c)$, we check if $d(t_4) > (1+\epsilon) \cdot rat(t_4)$.

(b) If so, a new connected component of B with root t_4 is created.

(c) When visiting $(t_4, s) \in \overleftarrow{E}(A_c)$, we reconnect s if (II) holds.

Fig. 3. The branching B at different stages of the Eulerian walk. The wide orange circles mark the head vertex of the currently visited edge in $E(\overleftrightarrow{A_c})$. The sky-blue boxes mark roots in B.

select the sink pairs according to the matching instead of arbitrarily. In order to achieve that $|T_{\max bif}|$ is large, we can compute $H \in \mathbb{N}$ minimal such that $\sum_{t \in T} 2^{-\min\{bif(t), H\}} \leq 1$ and decrease all bif-values to $\min\{bif(t), H\}$.

Theorem 2. *Let $T + r, p, rat, b$ be an instance of SLAP satisfying (3), $\epsilon > 0$, and A_c a topology for $T + r$. Algorithm 1 computes in $\mathcal{O}(n \log n + \Psi(A_c))$ time a topology A with*

$$wsl(A) \geq -2 \cdot b - \epsilon \cdot \max\{rat(t) : t \in T\} \text{ and} \tag{4}$$

$$cost(A) < \left(1 + \frac{2}{\epsilon}\right) \cdot cost(A_c) + \frac{4b \cdot n}{\epsilon}, \tag{5}$$

where $n := |T|$ and $\Psi(A_c)$ is the time needed to query $dist(v, w)$ for all $(v, w) \in E(A_c)$ and $dist(r, t)$ for all $t \in T$.

Remark 2.
- In many scenarios (e.g. $(\mathbb{R}^2, l_1)$), $\Psi(A_c) = \mathcal{O}(|E(A_c)|) = \mathcal{O}(n)$.
- With more effort one can prove $wsl(A) \geq -b - \min\{-b, -\epsilon \cdot \max_{t \in T}\{rat(t)\}\}$.

Proof. The algorithm uses only distances of edges in $E(A_c)$ and distances from r to all $t \in T$. Now, the running time follows from the fact that the Eulerian walk takes $\mathcal{O}(n)$ time, including all transformations of B and necessary updates of d. Huffman-coding for constructing A' runs in $\mathcal{O}(n \log n)$ time.

Now we prove inequality (4). Let $t \in T$ be a sink. After the first visit of t, $d(t) \leq (1+\epsilon) \cdot rat(t)$ by (I). Note that $d(t)$ increases only if an edge on the path from the root of its containing connected component and t is subdivided by a Steiner point during its second visit, i.e. after checking (II). Due to the subdivision, this can happen at most once. With $rat'(t') = d(t') + b$ for all $(t' \in T')$, we conclude that $delay_A(t) \leq (1+\epsilon) \cdot rat(t) + 2b$.

For the proof of inequality (5) first note that $cost(B) \leq cost(A_c) - dist(r, r')$ holds at the end of the Eulerian walk. Since $cost(A)$ is equal to the sum of $cost(B)$ and $cost(A')$, it suffices to estimate $cost(A')$. Let $T_1 := \{t_1, \ldots, t_k\}$ be the set of sinks for which condition (I) was true when we traversed the edge entering it ordered by the time they are traversed by the Eulerian walk (i.e. we visited t_i before t_{i+1} for all $1 \leq i \leq k-1$). Let T' be the set of roots of B at the end of the Eulerian walk (as defined in the algorithm). By construction, for each $t' \in T' \backslash \{r'\}$ it holds that $p(t') = p(t_i)$ $(i \in \{1, \ldots, k\})$, where t_i is the unique sink from T_1 in the connected component of B rooted at t'. Hence, $cost(A') = \sum_{t' \in T'} dist(r, t') \leq dist(r, r') + \sum_{i=1}^{k} dist(r, t_i)$.
In the remaining part of the proof we show that

$$\sum_{i=1}^{k} dist(r, t_i) < \frac{2}{\epsilon} \cdot cost(A_c) + \frac{4b \cdot n}{\epsilon}.$$

Define $t_0 := r$ and $d(r) := 0$ at any time of the algorithm. Consider the time when we visit a sink $t_i \in T_1$ $(i \in \{2, \ldots, k\})$ in the Eulerian walk. Let P_i be the $t_{i-1} - t_i$-subtour of the Eulerian walk produced by the algorithm and let $w \in V(P_i)$ be the lowest common ancestor of t_{i-1} and t_i in A_c. Let $V\left(A_{c[w,t_{i-1}]}\right) = \{w_1, w_2, \ldots, w_l\}$, $w_1 = w$, $w_l = t_{i-1}$ (in that order). For $y, z \in V(A_c) \backslash \{r\}$ denote by $d^y(z)$ the value of $d(z)$ at the time of the Eulerian walk just before the edge in $\overleftarrow{E}(A_c)$ leaving y is visited. By convention let $d^r(z)$ denote the value for $d(z)$ at the very beginning of the algorithm.
Due to condition (II), $d^x(x) \leq d^x(y) + dist(x, y) + b \leq d^y(y) + dist(x, y) + b$ for all edges $(x, y) \in E(A_c)$. Consequently, for all $i > 1$ it holds that

$$\begin{aligned} d^w(w) &\leq d^{w_2}(w_2) + dist(w, w_2) + b \\ &\leq d^{w_3}(w_3) + (dist(w, w_2) + dist(w_2, w_3)) + 2b \\ &\leq \ldots \\ &\leq d^{t_{i-1}}(t_{i-1}) + cost(A_{c[w,t_{i-1}]}) + b \cdot |E\left(A_{c[w,t_{i-1}]}\right)|. \end{aligned} \tag{6}$$

Now consider the time when the edge in $E(A_c)$ entering t_i is visited (i.e. the time the algorithm determines that $d(t_i) > (1+\epsilon) \cdot rat(t_i)$). Let $d^*_A(z)$ denote the value of $d(z)$ at that time for all $z \in V(A_c)$. No edge of $E\left(A_{c[w,t_i]}\right)$ has been deleted since the first traversal of the first edge of $A_{c[w,t_i]}$ by the choice of t_{i-1} and t_i. Hence, $d^*(w) = d^w(w)$ and $d^*(t_i) \leq d^*(w) + cost(A_{c[w,t_i]}) + b \cdot |E\left(A_{c[w,t_i]}\right)|$.
Since $t_i \in T_1$, it holds that $d^*(t_i) > (1+\epsilon) \cdot rat(t_i)$ and with (6):

$$\begin{aligned} (1+\epsilon) \cdot rat(t_i) &< d^*(t_i) \\ &\leq d^*(w) + cost(A_{c[w,t_i]}) + b \cdot |E\left(A_{c[w,t_i]}\right)| \\ &\leq d^{t_{i-1}}(t_{i-1}) + cost(P_i) + b \cdot |E(P_i)|. \end{aligned}$$

For each $i > 1$, t_{i-1} has become the root of a new connected component of B after we have visited t_{i-1} which implies $d^{t_{i-1}}(t_{i-1}) = dist(r, t_{i-1}) + b \cdot bif(t_{i-1})$. If $i = 1$, the inequality $(1+\epsilon) \cdot rat(t_i) < d^{t_{i-1}}(t_{i-1}) + cost(P_i) + b \cdot |E(P_i)|$ is trivial. Using $rat(.) \geq dist(r, .) + b \cdot bif(.)$ we obtain

$$(1+\epsilon)(dist(r, t_i) + b \cdot bif(t_i)) < dist(r, t_{i-1}) + b \cdot bif(t_{i-1}) + cost(P_i) + b \cdot |E(P_i)|.$$

Summing up over all $i = 1, \ldots, k$ yields

$$(1+\epsilon)\sum_{i=1}^{k} dist(r,t_i) + (1+\epsilon)\sum_{i=1}^{k} b \cdot bif(t_i) < \sum_{i=0}^{k-1} dist(r,t_i) + \sum_{i=0}^{k-1} b \cdot bif(t_i) + \sum_{i=1}^{k} (cost(P_i) + b \cdot |E(P_i)|).$$

Now note that the P_i are pairwise disjoint parts of the Eulerian walk through A_c. We conclude that $\sum_{i=1}^{k} cost(P_i) \leq 2 \cdot cost(A_c)$ and $\sum_{i=1}^{k} |E(P_i)| \leq 2 \cdot |E(A_c)| = 4n-2$. By combining these inequalities we obtain $\sum_{i=1}^{k} dist(r,t_i) < \frac{2 \cdot cost(A_c)}{\epsilon} + \frac{4b \cdot n}{\epsilon}$ which concludes the proof of Theorem 2. □

Remark 3. If $(M, dist) = (\mathbb{R}^2, l_1)$, we can find in $\mathcal{O}(|T| \cdot \log(|T|))$ time a minimum spanning tree to initialize A_c and by Algorithm 1 a topology A satisfying (4) and

$$cost(A) \leq \frac{3}{2}\left(1 + \frac{2}{\epsilon}\right) cost(SMT) + \frac{4b \cdot |T|}{\epsilon},$$

where SMT is a minimum Steiner tree for $T \cup \{r\}$ and $\frac{3}{2}$ the Steiner ratio [7].

3.2 An Algorithm for $(\mathbb{R}^2, l_1)$ and $b = 0$

If the number of bifurcations of a path does not influence its delay ($b = 0$) and the metric space is $(\mathbb{R}^2, l_1)$, we can prove a much better cost bound.

Theorem 3. *Let $(M, dist) = (\mathbb{R}^2, l_1)$ and let $T + r, p, rat, b$ be an instance of SLAP such that $b = 0$ and $rat(t) \geq ||p(t) - p(r)||_1$ for all $t \in T$. For any topology A_c for $T + r$ and for each $\epsilon > 0$ we can compute a topology A for $T + r$ in $\mathcal{O}(n \cdot \log n)$ time, where $n = |T|$, such that*

$$wsl(A) \geq -\epsilon \cdot \max\{rat(t) : t \in T\} \quad \text{and}$$

$$cost(A) \leq \begin{cases} \left(2 + \left\lceil \log\left(\frac{2}{\epsilon}\right)\right\rceil\right) \cdot cost(A_c) & \text{if } 0 < \epsilon \leq 2 \\ \left(1 + \frac{2}{\epsilon}\right) \cdot cost(A_c) & \text{if } \epsilon > 2. \end{cases}$$

The proofs of Theorem 3 and Lemma 1 are based on Lemma 3.1 of [3]. Since the tree they compute in their Section 2 contains Steiner points not belonging to the metric space they are working in, we use a similar algorithm as [11] to compute a topology with the same properties for the $(\mathbb{R}^2, l_1)$ case:

Algorithm 2. W.l.o.g. we may assume that $n = |T|$ is a power of 2 (if this is not the case, add $2^{\lceil \log(n) \rceil} - n$ new sinks placed at $p(r)$). Let F be any Steiner tree for $T + r$. We use Remark 1 to find a topology for $T + r$. Note that the set $T_{\max bif}$ is equal to the set of remaining sinks in each iteration. The Steiner tree F induces a Hamiltonian cycle through these sinks with cost at most $2 \cdot cost(F)$ which

contains a perfect matching M on $T_{\max bif}$ with cost at most $cost(F)$ which we may choose to reduce the number of current sinks to $\lceil |T_{\max bif}|/2 \rceil$.

After $\log(n) - 1$ reductions, there are exactly two sinks t_1, t_2 left. In each iteration we have included edges of cost at most $cost(F)$. Since, by construction, t_1 and t_2 are placed within the bounding box of $T \cup \{r\}$, a minimum cost topology for $r + \{t_1, t_2\}$ provides a feasible topology with cost bounded by $cost(F)$. All paths contained in the topology A for $T + r$ found that way are shortest paths.

Lemma 1. *Let $(M, dist) = (\mathbb{R}^2, l_1)$ and let $T+r, p, rat, b$ be an instance of SLAP such that $b = 0$. Furthermore, let F be a Steiner tree for $T \cup \{r\}$, $\alpha \geq 1$, and $\eta > 0$ with*

$$\alpha \cdot \eta \geq \sum_{t \in T} ||p(r) - p(t)||_1 = \sum_{t \in T} dist_A(r, t). \tag{7}$$

The topology A computed by Algorithm 2 fulfills $cost(A) \leq \eta + \lceil \log(\alpha) \rceil \cdot cost(F)$ and can be computed in $\mathcal{O}(n)$ time, where $n = |T|$.

Proof. As in Algorithm 2 we assume that n is a power of 2. If $\lceil \log(\alpha) \rceil \geq \log(n) + 1$, the statement is trivial. Assume $\lceil \log(\alpha) \rceil \leq \log(n)$. For $i \in \{0, \ldots, \log(n)\}$ let E_i denote the set of edges $(v, w) \in E(A)$ for which $|E\left(A_{[r,v]}\right)|$ is equal to $\log(n) - i$. Note that the number of sinks reachable from the endpoint of an edge in E_i is 2^i. Consequently,

$$\sum_{t \in T} dist_A(r, t) = \sum_{i=0}^{\log(n)} \sum_{e \in E_i} |\{t \in T : e \in E(A_{[r,t]})\}| \cdot cost(e) = \sum_{i=0}^{\log(n)} 2^i cost(E_i).$$

Thus $\alpha \cdot \eta \geq \sum_{i=\lceil \log(\alpha) \rceil}^{\log(n)} 2^i \cdot cost(E_i) \geq \alpha \cdot \sum_{i=\lceil \log(\alpha) \rceil}^{\log(n)} cost(E_i)$ and hence,

$$cost(A) = \sum_{i=0}^{\lceil \log(\alpha) \rceil - 1} cost(E_i) + \sum_{i=\lceil \log(\alpha) \rceil}^{\log(n)} cost(E_i) \leq \lceil \log(\alpha) \rceil \cdot cost(F) + \eta.$$

The running time is obvious. □

Proof (of Theorem 3). We use Algorithm 1 and use Lemma 1 to compute A' at the very end. Let A be the output. Theorem 2 implies the claimed properties on the worst slack of A. If $\epsilon > 2$, Theorem 2 implies the claimed cost bound. Let $0 < \epsilon \leq 2$. As seen in the proof of Theorem 2, $\sum_{t \in T'} ||p(r) - p(t)||_1 \leq \frac{2}{\epsilon} \cdot cost(A_c)$. By Lemma 1 ($F := A_c, \alpha := \frac{2}{\epsilon}, \eta := cost(A_c)$), $cost(A') \leq \left\lceil \log\left(\frac{2}{\epsilon}\right) \right\rceil \cdot cost(A_c) + cost(A_c)$. We conclude that the returned topology has cost at most $\left(2 + \left\lceil \log\left(\frac{2}{\epsilon}\right) \right\rceil\right) \cdot cost(A_c)$. The claim about the running time follows from Remark 3 and Theorem 3. □

4 Experimental Results

We used Algorithm 1 with improved Huffman-coding as in Remark 1 on instances arising as repeater topology problems in VLSI-design provided by IBM. Here, an electrical signal is distributed from one logic gate to a set of destination gates on a chip (see [1,5] for details).

Table 1. Comparison of the results of Algorithm 1 A with A_c and A_{huf}

		$\epsilon = 0$		$\epsilon = 0.1$		$\epsilon = 0.3$		$\epsilon = 1.0$	
$\lvert T\rvert$		wsl (in ps)	cost ratio	wsl (ps)	cost r.	wsl (ps)	cost r.	wsl (ps)	cost r.
≤ 100	max	0.000	4.385	0.000	3.393	0.000	2.747	0.000	2.224
	min	-9.726	0.995	-82.281	0.995	-284.963	0.998	-497.640	0.998
	av	-0.141	1.030	-0.330	1.022	-0.795	1.012	-1.567	1.002
	total		1.077		1.049		1.019		1.002
≥ 101	max	0.000	9.229	0.000	3.305	0.000	2.673	0.000	1.888
	min	-9.726	1.000	-164.175	1.000	-556.700	1.000	-1497.640	1.000
	av	-0.149	1.456	-6.772	1.316	-19.511	1.198	-61.107	1.083
	total		1.427		1.218		1.099		1.060
all	max	0.000	9.229	0.000	3.393	0.000	2.747	0.000	2.224
	min	-9.726	0.995	-164.175	0.995	-556.700	0.998	-1497.640	0.998
	av	-0.679	1.032	-0.363	1.023	-0.891	1.013	-1.870	1.003
	total		1.093		1.054		1.013		1.004

wsl: $\min\{0, wsl(A)\} - \min\{0, wsl(A_{\text{huf}})\}$ in picoseconds, cost ratio: $cost(A)/cost(A_c)$.

Excluding trivial instances with one or two sinks, there were 718 379 instances with at least 3 and up to 169 150 sinks. 3 656 of them had more than 100 sinks. The values for b varied between 4 and 10 picoseconds, depending on the chip-technology. The initial topologies A_c were computed by heuristics guaranteeing minimum Steiner trees for up to eight sinks and $\frac{3}{2}$-approximations otherwise.

In Table 1 we compare the cost of topologies A computed by Algorithm 1 with the cost of A_c and the best possible worst slack attained by A_{huf}, the topology arising from Huffman-coding. In addition, we ran an optimized variant of the greedy algorithm in [1], as it is used at IBM, to obtain a reference topology A_{ref}. This comparison is shown in Table 2.

Running times are negligible for all compared algorithms. Algorithm 1 needs roughly 2 minutes to compute topologies for all 718 379 instances on a 3GHz Xeon machine. More than 90 % of the time was spent on computing A_c.

In both tables, for a topology A generated by Algorithm 1 and a respective reference topology A', the column "cost ratio" shows the maximum, minimum, and average of the ratios $cost(A)/cost(A')$ as well as the ratio of the total costs added up over all instances. The columns "wsl" show the maximum, minimum and average worst slack difference $\min\{0, wsl(A)\} - \min\{0, wsl(A')\}$ in picoseconds. We ran Algorithm 1 with four different values for ϵ (0, 0.1, 0.3, and 1.0). Table 1 shows that for $\epsilon = 0$ we achieve near-feasible solutions at 10% higher total cost compared to the short topologies A_c. With higher values of ϵ the worst slack decreases moderately, except for a few instances, while the costs approach the costs of A_c. As A_c is not minimum, its length can be underpriced by A.

Table 2 shows that the greedy algorithm [1] is not able to bound the worst slack tightly. It looses almost a nanosecond on some instances. In contrast, Algorithm 1 with $\epsilon = 0$ guarantees near-optimum worst slacks and slight improvements

Table 2. Comparison of the results of Algorithm 1 A with A_{ref}

$\|T\|$		$\epsilon = 0$		$\epsilon = 0.1$		$\epsilon = 0.3$		$\epsilon = 1.0$	
		wsl (in ps)	cost ratio	wsl (ps)	cost r.	wsl (ps)	cost r.	wsl (ps)	cost r.
≤ 100	max	919.533	2.957	885.573	2.936	783.098	2.048	664.193	1.647
	min	-9.726	0.176	-77.096	0.176	-253.053	0.176	-466.440	0.176
	av	0.002	0.969	-0.188	0.962	-0.653	0.954	-1.425	0.946
	total		1.015		0.986		0.960		0.944
≥ 101	max	358.295	6.136	350.783	2.679	328.639	1.980	219.752	1.426
	min	-9.726	0.202	-164.175	0.190	-477.800	0.190	-1447.180	0.190
	av	2.244	1.114	-2.697	1.018	-15.436	0.938	-57.031	0.859
	total		1.166		0.995		0.929		0.866
all	max	919.533	6.136	885.573	2.936	783.098	2.048	664.193	1.647
	min	-9.726	0.176	-164.175	0.176	-477.800	0.176	-1447.180	0.176
	av	0.013	0.970	-0.201	0.962	-0.728	0.954	-1.708	0.947
	total		1.023		0.987		0.959		0.940

wsl: $\min\{0, wsl(A)\} - \min\{0, wsl(A_{\text{ref}})\}$ in picoseconds, cost ratio: $cost(A)/cost(A_{\text{ref}})$.

w.r.t. average cost and worst slack. The total netlength, which is dominated by a few very large instances, is only 2.3% larger. By increasing ϵ we achieve a large improvement in average and total cost while the average worst slack decreases moderately below the reference worst slack.

References

1. Bartoschek, C., Held, S., Maßberg, J., Rautenbach, D., Vygen, J.: The Repeater Tree Construction Problem. Information Processing Letters 110, 1079–1083 (2010)
2. Cong, J., Kahng, A.B., Robins, G., Sarrafzadeh, M., Wong, C.K.: Provably good performance-driven global routing. IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems 11(6), 739–752 (1992)
3. Elkin, M., Solomon, S.: Steiner Shallow-Light Trees are Exponentially Lighter than Spanning Ones. In: Proc. 52nd FOCS, pp. 373–382 (2011)
4. Gouveia, L., Simonetti, L., Uchoa, E.: Modeling hop-constrained and diameter-constrained minimum spanning tree problems as Steiner tree problems over layered graphs. Mathematical Programming A 128(1-2), 123–148 (2011)
5. Hrkic, M., Lillis, J.: Generalized Buffer Insertion. In: Alpert, C., Sapatnekar, S., Mehta, D.D. (eds.) Handbook of Algorithms for VLSI Physical Design Automation, pp. 557–567. CRC Press (2007)
6. Huffman, D.A.: A Method for the Construction of Minimum-Redundancy Codes. In: Proc. of the IRE, vol. 40(9), pp. 1098–1101 (1952)
7. Hwang, F.K.: On steiner minimal trees with rectilinear distance. SIAM Journal of Applied Mathematics 30, 104–114 (1976)
8. Khuller, S., Raghavachari, B., Young, N.: Balancing Minimum Spanning Trees and Shortest-Path Trees. Algorithmica 14, 305–321 (1995)
9. Kraft, S.G.: A device for quantizing grouping and coding amplitude modulated pulses. Master's thesis. MIT, Cambridge (1949)

10. Manyem, P., Stallmann, M.: Some Approximation Results in Multicasting. Working Paper, North Carolina State University (1996)
11. Rao, S., Sadayappan, P., Hwang, F.K., Shor, P.: The Rectilinear Steiner Arborescence Problem. Algorithmica 7, 277–288 (1992)
12. Ruthmair, M., Raidl, G.R.: A Layered Graph Model and an Adaptive Layers Framework to Solve Delay-Constrained Minimum Tree Problems. In: Günlük, O., Woeginger, G.J. (eds.) IPCO 2011. LNCS, vol. 6655, pp. 376–388. Springer, Heidelberg (2011)
13. Shi, W., Su, C.: The Rectilinear Steiner Arborescence Problem is NP-Complete. In: Proc. 11th ACM-SIAM Symp. on Discrete Algorithms, pp. 780–787 (2000)
14. Tovey, C.A.: A Simplified NP-Complete Satisfiability Problem. Discrete Applied Mathematics 8(1), 85–89 (1984)

Two Dimensional Optimal Mechanism Design for a Sequencing Problem

Ruben Hoeksma and Marc Uetz

University of Twente, Dept. Applied Mathematics, P.O. Box 217,
7500AE Enschede, The Netherlands
{r.p.hoeksma,m.uetz}@utwente.nl

Abstract. We propose an optimal mechanism for a sequencing problem where the jobs' processing times and waiting costs are private. Given public priors for jobs' private data, we seek to find a scheduling rule and incentive compatible payments that minimize the total expected payments to the jobs. Here, incentive compatible refers to a Bayes-Nash equilibrium. While the problem can be efficiently solved when jobs have single dimensional private data, we here address the problem with two dimensional private data. We show that the problem can be solved in polynomial time by linear programming techniques, answering an open problem in [13]. Our implementation is randomized and truthful in expectation. The main steps are a compactification of an exponential size linear program, and a combinatorial algorithm to decompose feasible interim schedules. In addition, in computational experiments with random instances, we generate some more insights.

1 Introduction and Contribution

In this paper, we address an optimal mechanism design problem for a sequencing problem introduced by Heydenreich et al. in [13]. While that paper mainly addresses the version with single dimensional private data, we focus on the case with two dimensional private data. Indeed, starting with the seminal paper by Myerson [16], optimal mechanism design with single dimensional private data is pretty well understood, also from an algorithmic point of view, e.g. [12], while algorithmic results for optimal mechanism design with multi dimensional private data have been obtained only recently, e.g. [1,3].

Our starting point is the **open problem** formulated in [13], who 'leave it as an open problem to identify (closed formulae for) optimal mechanisms for the 2-d case.' Here, the '2-d case' refers to the problem of computing a Bayes-Nash optimal mechanism for the following sequencing problem on a single machine: There are n jobs with two dimensional private data, namely a cost per unit time w_j and a processing time p_j. Jobs need to be processed sequentially, and each job requires a compensation for the disutility of waiting. With given priors on the private data of jobs, the optimal mechanism seeks to minimize the total expected payments made to the jobs, while being Bayes-Nash incentive compatible. This problem is an abstraction of economic situations where clients queue

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 242–253, 2013.

for a single scarce resource (e.g., a specialized operation theatre), while the information on the urgency and duration to treat each client is private, yet known probabilistically. A concrete example are waiting lists for medical treatments in the Netherlands, see [14].

The **main contribution of this paper** is to answer the open problem in [13], by giving an optimal mechanism and showing that it can be computed and implemented in polynomial time. Our solution is based on linear programming techniques, and results in an optimal randomized mechanism. In that sense, we do not obtain analytic 'closed formulae' for the solution, and our results can be seen in the tradition of 'automated mechanism design' as proposed e.g. by Conitzer and Sandholm [4,20], in that the design of the mechanism itself is based on (integer) linear programming.

The major **technical contributions** are twofold: The first is the compactification of an exponential size linear programming formulation of the mechanism design problem, which is the crucial ingredient that allows a polynomial time algorithm to compute payments and a so-called interim schedule. The second is an algorithm that allows to compute, in polynomial time, the implementation for the given interim schedule. To that end, we give a combinatorial $O(n^3 \log n)$ algorithm that computes, for any given point s in the single machine scheduling polytope as defined by Queyranne [18], a representation of s as convex combination of $\leq n$ vertices. This result generalizes a similar result for the permutahedron by Yasutake et al. [23], but in contrast to that paper, our algorithm follows the geometric construction as proposed by Grötschel et al. in [11, Thm. 6.5.11].

Finally, again in the flavor of automated mechanism design, we present **computational results** based on the (integer) linear programming formulations. These computations have the primary goal to test and validate hypotheses on the structure of solutions. Our computations, based on randomly generated instances, show that optimal mechanisms in the two dimensional setting do *not* share several of the nice properties of the solutions to the single dimensional problem: The scheduling rules of optimal Bayes-Nash incentive compatible mechanisms are not necessarily *iia* (a desirable property to be defined later), and neither do optimal Bayes-Nash mechanisms allow an implementation in dominant strategies. This in contrast to the single dimensional problem which has these properties [13,5].

We conclude this section with a brief discussion of our result in relation to the recent results of Cai et al. [3]. Apart from some methodological similarities in Section 4, we specifically ask the question if the problem that we consider here fits into the general framework presented there. This is not the case: In order to formulate the problem considered here in that context, we can either represent a schedule as an assignment of n jobs to n slots, in which case the problem has informational externalities because the utility of a job for a given slot then depends on the types (specifically, processing times) of other jobs. Or, we can represent a schedule as a vector of starting times, but then the feasibility of such vector depends on the types (specifically, processing times) of jobs. Either way, we leave the framework of [3], and we do not see a simple way to fix this.

2 Definitions, Preliminary and Related Results

We consider a sequencing (or single machine scheduling) problem with n agents denoted $j \in N$, each owning a job with weight w_j and processing time p_j. We identify jobs with agents. The jobs need to be processed (sequenced) on a single machine, with the interpretation that w_j is job j's individual cost for waiting one unit of time, while p_j is the time it requires to process job j. In a schedule that yields a start time s_j for job j, the cost for waiting is $w_j s_j$. The *type* of a job j is the vector of weight and processing time, denoted $t_j = (w_j, p_j)$. Note that the type is two dimensional. With t_j being public, the total waiting cost is well known to be minimized by sequencing the jobs in order of non increasing ratios w_j/p_j, also known as Smith's rule [21].

In the setting we consider here, weight and processing time are private to the agent that owns the job. There is a public belief about this private information, which is[1]

- the types that job j might have are $T_j = \{t_j^1, \ldots, t_j^{m_j}\}$, and
- the probability of job j having type t_j^i is $\varphi_j(t_j^i)$, $i = 1 \ldots, m_j$.

By $T = T_1 \times \ldots \times T_n$ we denote the type space of all jobs, with $t = (t_1, \ldots, t_n) \in T$. Define $m := \sum_{j \in N} m_j$, and note that $m \geq n$. For a type $t_j^i \in T_j$, we let w_j^i and p_j^i be the corresponding weight and processing time, respectively. We sometimes abuse notation by identifying i with t_j^i, to avoid excessive notation. Moreover, (t_j, t_{-j}) denotes a type vector where t_j is the type of job j and t_{-j} are the types of all jobs except j, with $t_{-j} \in T_{-j} := \prod_{k \neq j} T_k$. For given $t \in T$ and $K \subseteq N$, we also define the shorthand notation $\varphi(t_K) := \prod_{k \in K} \varphi_k(t_k)$ for the product distribution of the types of jobs in K, particularly $\varphi(t_{-j}) := \prod_{k \neq j} \varphi_k(t_k)$.

We assume, just like [13], that the mechanism designer needs to compensate the jobs for waiting by a payment π_j that the job receives. We seek to compute and implement a (direct) mechanism, consisting of a scheduling rule and a payment rule, assigning to any $t \in T$ a permutation $\sigma(t)$ of jobs which yields a schedule $s^\sigma(t)$ of start times, together with compensation payments $\pi(t)$. In the mechanism design and auction literature, for obvious reasons, what is a scheduling rule here is referred to as *allocation rule*. Clearly, jobs may have an incentive to strategically misreport their true types in order to receive higher compensation payments. The optimal mechanism that we seek, however, is one that minimizes the total payments made to the jobs. Since reporting a processing time smaller than the true processing time is verifiable while processing a job, we assume, again like [13], that only larger than the true processing times can be reported by any job.

It is Myerson's revelation principle [16] that makes this problem (and many others [22]) amendable to optimization techniques: it asserts that it is no loss of generality to restrict to *truthful* mechanisms, where each job maximizes utility by reporting the type truthfully. In the considered setting with given priors on

[1] Note that the discrete type space make the problem amendable for (I)LP techniques.

private data, a mechanism is truthful, or more precisely *Bayes-Nash incentive compatible*, if it fulfills the following, linear constraint

$$\pi_j^i - w_j^i Es_j^i \geq \pi_j^{i'} - w_j^i Es_j^{i'} \quad \text{for all jobs } j \text{ and types } t_j^i, t_j^{i'} \in T_j\,.$$

Here, Es_j^i and π_j^i are defined as expected start time and payment for job j when he reports to be of type t_j^i, where the expectation is taken over all (truthful) reports of other jobs $t_{-j} \in T_{-j}$. Then, assuming utilities are quasi-linear, the expected utility for job j with true type t_j^i is $\pi_j^i - w_j^i Es_j^i$ for reporting truthfully, while a false report $t_j^{i'}$ yields expected utility $\pi_j^{i'} - w_j^i Es_j^{i'}$. The scheduling rule corresponding to a Bayes-Nash incentive compatible mechanism is called Bayes-Nash *implementable*.

Moreover, in order to have the problem bounded, we make the standard assumption that the expected utilities of truthful jobs are nonnegative, known as *individual rationality*,

$$\pi_j^i - w_j^i Es_j^i \geq 0\,.$$

It is interesting to ask if a scheduling rule (more generally, allocation rule) can even be implemented in the stronger *dominant strategy* equilibrium; in [15] the equivalence of Bayes-Nash and dominant strategy implementations is shown for the case of standard single unit private value auctions. In a dominant stategy equilibrium, reporting the true type maximizes the utility of a job not only in expectation but for *any* report t_{-j} of the other jobs, that is, $\pi_j(t_j^i, t_{-j}) - w_j^i s_j(t_j^i, t_{-j}) \geq \pi_j(t_j^{i'}, t_{-j}) - w_j^i s_j(t_j^{i'}, t_{-j})$ for all $t_j^i, t_j^{i'} \in T_j$ and all $t_{-j} \in T_{-j}$. The latter obviously implies the former, but generally not vice versa [10].

In the setting considered here, a mechanism is Bayes-Nash implementable if and only if the expected start times Es_j^i are monotonically increasing in the reported weight w_j^i. The same result holds for dominant strategy implementability, but then the start times $s_j(t_j^i, t_{-j})$ need to be monotonically increasing in the reported weight w_j^i, for all $t_{-j} \in T_{-j}$. This is a standard result in single-dimensional mechanism design [17], but it is also true for the 2-dimensional problem considered here [13]. The problem to find an *optimal* mechanism for the 2-dimensional mechanism design problem was left open in [13].

For the single dimensional mechanism design problem, where only weights are private information and processing times are known, the optimal mechanism has a simple structure: It is Smith's rule, but with respect to virtual instead of the original weights w_j; see [13] for details. In particular, in that case the optimal Bayes-Nash incentive compatible mechanism can be computed and implemented in polynomial time, and it can even be implemented (with the same expected cost) in dominant strategies [5].

3 Problem Formulations and Linear Relaxation

Let us start by giving a natural, albeit exponential size ILP formulation for the mechanism design problem at hand. Recall that $s_j^\sigma(t)$ denotes the start time of

job j if the permutation of jobs is σ under type vector t. We use the natural variables

$$x_\sigma(t) = \begin{cases} 1 & \text{if for type vector } t \text{ permutation } \sigma \text{ is used}, \\ 0 & \text{otherwise}. \end{cases}$$

Then the formulation reads as follows.

$$\min \sum_{j\in N}\sum_{i\in T_j} \varphi_j^i \pi_j^i \tag{1}$$

$$\pi_j^i \geq w_j^i Es_j^i \qquad \forall j \in J, i \in T_j \tag{2}$$

$$\pi_j^i \geq \pi_j^{i'} - w_j^i(Es_j^{i'} - Es_j^i) \qquad \forall j \in N, i \in T_j, i' \in T_j, p_j^{i'} \geq p_j^i \tag{3}$$

$$Es_j^i = \sum_{t_{-j}\in T_{-j}} \varphi(t_{-j}) \sum_\sigma x_\sigma(t_j^i, t_{-j}) s_j^\sigma(t_j^i, t_{-j}) \quad \forall j \in N, t_j^i \in T_j \tag{4}$$

$$\sum_\sigma x_\sigma(t) = 1 \qquad \forall t \in T \tag{5}$$

$$x_\sigma(t) \in \{0,1\} \qquad \forall \sigma \in \Sigma, t \in T \tag{6}$$

Here we use the shorthand notation φ_j^i for $\varphi_j(t_j^i)$, and Σ is the set of all permutations of N. The objective (1) is the total expected payment. Constraints (2) and (3) are the individual rationality and incentive compatibility constraints: (2) requires the expected payment to at least match the expected cost of waiting when the type is t_j^i, and (3) makes sure that the expected utility is maximized when reporting truthfully. Values Es_j^i are also referred to as *interim schedule*, and equations (4) are the feasibility constraints for interim schedules, expressing the fact that the expected starting times in the interim schedule need to comply with the scheduling rule encoded by x. While the input size of the mechanism design problem is $\mathrm{O}(m)$, this ILP formulation is colossal as the number of variables $x_\sigma(t)$ is $|T|\, n!$ with $|T| = \prod_j m_j$.

Observe that, for given type vector t, the vectors $s^\sigma(t)$ are the vertices of the well known single machine scheduling polytope $Q(t)$ [6,18], only here we consider start instead of completion times. In other words, $s^\sigma(t)$ are the start times of permutation schedules. Recall from [18] that the polytope $Q(t)$ is defined by

$$\sum_{j\in K} p_j(t)s_j(t) \geq \frac{1}{2}\left(\sum_{j\in K} p_j(t)\right)^2 - \frac{1}{2}\sum_{j\in K} p_j(t)^2 \qquad \forall K \subseteq N \tag{7}$$

$$\sum_{j\in N} p_j(t)s_j(t) = \frac{1}{2}\left(\sum_{j\in N} p_j(t)\right)^2 - \frac{1}{2}\sum_{j\in N} p_j(t)^2\ , \tag{8}$$

where we use $p_j(t)$ to denote the processing time of job j in type profile t. The last equality excludes schedules with idle time. Allowing randomization, any point of $Q(t)$ represents feasible expected start times. Note that the scheduling polytope $Q(t)$ is a polymatroid via variable transform to $p(t)s(t)$. In this particular case, both optimization and separation for $Q(t)$ can be done in time $\mathrm{O}(n^2)$ [7,18].

3.1 Linear Ordering Formulation

It turns out to be convenient for our purpose to consider another formulation, namely using linear ordering variables d_{kj}, with intended meaning

$$d_{kj}(t) = \begin{cases} 1 & \text{if for type vector } t \text{ we use a schedule where job } k \text{ precedes job } j\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

Using linear ordering variables yields the following formulation of the optimal mechanism design problem.

$$\min \sum_{j\in N} \sum_{i\in T_j} \varphi_j^i \pi_j^i \tag{9}$$

$$\pi_j^i \geq w_j^i Es_j^i \qquad \forall j, i \tag{10}$$

$$\pi_j^i \geq \pi_j^{i'} - w_j^i (Es_j^{i'} - Es_j^i) \qquad \forall j, i, i' \tag{11}$$

$$Es_j^i = \sum_{t_{-j}\in T_{-j}} \varphi(t_{-j}) s_j(t_j^i, t_{-j}) \qquad \forall j, i \tag{12}$$

$$s_j(t) = \sum_{k\in N} d_{kj}(t) p_k(t) \qquad \forall j, t \tag{13}$$

$$d_{jj}(t) = 0 \qquad \forall j, t \tag{14}$$

$$d_{kj}(t) + d_{jk}(t) = 1 \qquad \forall j, k, t\ j \neq k \tag{15}$$

$$d_{jk}(t) \geq 0 \qquad \forall j, k, t \tag{16}$$

$$d_{jk}(t) + d_{kl}(t) \leq 1 + d_{jl}(t) \qquad \forall j, k, l, t \tag{17}$$

$$d_{jk}(t) \in \{0, 1\} \qquad \forall j, k, t\ . \tag{18}$$

Observe that, in contrast to the previous x_σ formulation, the number of variables $d_{jk}(t)$ now equals $n^2 \cdot |T|$. However this formulation is in general exponential as well, since the type space T can be exponential in m.

The vertices of $Q(t)$ are the solutions $s(t)$ of (13)-(18), and moreover, a vector of starting times $s(t)$ satisfies (13)-(16) if and only if it satisfies (7) and (8); see for instance [19, Thm. 4.1]. More specifically, via (13), the scheduling polytope $Q(t)$ is an affine image of both the linear ordering polytope (14)-(18) and its relaxation (14)-(16). This important observation is crucial for what follows, as we can continue to work with the relaxation (14)-(16) instead of (14)-(18).

3.2 Relaxation and Compactification

A linear relaxation of the optimal mechanism design problem (9)-(18) is obtained by dropping the last two sets of constraints (17) and (18). By moving from the ILP formulation to its LP relaxation, we in fact move from deterministic scheduling rules to randomized ones, which follows from our previous discussion about the equivalence of (13)-(16) and (7)-(8), as well as the fact that the scheduling polytope $Q(t)$ is an affine image of the relaxation (14)-(16) via (13).

In what follows we also combine (12) and (13) into just one constraint, and (17) and (18) are omitted. This gives us the following formulation.

$$\min \sum_{j\in N}\sum_{i\in T_j} \varphi_j^i \pi_j^i \tag{19}$$

$$\pi_j^i \geq w_j^i Es_j^i \qquad \forall j,i \tag{20}$$

$$\pi_j^i \geq \pi_j^{i'} - w_j^i (Es_j^{i'} - Es_j^i) \qquad \forall j,i,i' \tag{21}$$

$$Es_j^i = \sum_{t_{-j}\in T_{-j}} \sum_{k\in N} \varphi(t_{-j}) d_{kj}(t_j^i, t_{-j}) p_k(t_{-j}) \qquad \forall j,i \tag{22}$$

$$d_{jj}(t) = 0 \qquad \forall j,t \tag{23}$$

$$d_{kj}(t) + d_{jk}(t) = 1 \qquad \forall j,k,t,\ k\neq j \tag{24}$$

$$d_{kj}(t) \geq 0 \qquad \forall j,k,t\ . \tag{25}$$

We now focus on the projection to variables Es_j^i, that is, vectors $Es \in \mathbb{R}^m$ satisfying (22)-(25). These are interim schedules in the linear relaxation. Let us refer to this projection as the *relaxed interim scheduling polytope*. Notice that, even though it is a linear relaxation, (22)-(25) is still an exponential size formulation, as it depends on the size of T. The crucial insight is that, in the linear relaxation, this exponential size formulation is actually not necessary. Instead of using $d_{kj}(t)$ where $t \in T$, we propose an *LP compactification* by restricting to variables

$$d_{kj}(t_k, t_j)\,,$$

where t_k and t_j are the types of jobs k and j, respectively. This reduces the number of d_{kj}-variables to $\mathrm{O}(m^2)$, yielding a polynomial size formulation. Doing so, we obtain

$$Es_j^i = \sum_{k\in N}\sum_{t_k\in T_k} \varphi(t_k) d_{kj}(t_j^i, t_k) p_k(t_k) \qquad \forall j,i \tag{26}$$

$$d_{jj}(t_j, t_j) = 0 \qquad \forall j,t_j \tag{27}$$

$$d_{kj}(t_k, t_j) + d_{jk}(t_j, t_k) = 1 \qquad \forall j,k,t_j,t_k,\ k\neq j \tag{28}$$

$$d_{kj}(t_k, t_j) \geq 0 \qquad \forall j,k,t_j,t_k\ . \tag{29}$$

The following lemma is the core insight of the results in this paper.

Lemma 1. *The relaxed interim scheduling polytope defined by* (22)-(25) *can be equivalently described by* (26)-(29)*.*

Proof. Let P be the projection of (22)-(25) to variables Es_j^i, and P' be the projection of (26)-(29) to variables Es_j^i. It is obvious that if $Es \in P'$, then $Es \in P$, simply by letting $d_{kj}(t) = d_{kj}(t_k, t_j)$, for all $t \ni t_k, t_j$. So all we need to show is that, if $Es \in P$, then $Es \in P'$. So let $Es \in P$ with corresponding $d_{kj}(t)$. Now define

$$d_{kj}(t_k, t_j) = \sum_{t\ni t_k, t_j} \frac{\varphi(t)}{\varphi(t_k)\varphi(t_j)} d_{kj}(t)\ ,$$

then the $d_{kj}(t_k, t_j)$ clearly satisfy (27)-(29). Moreover, we have for all $j \in N$ and $i \in T_j$,

$$\begin{aligned} Es_j^i &= \sum_{t_{-j} \in T_{-j}} \sum_{k \in N} \varphi(t_{-j}) d_{kj}(t_j^i, t_{-j}) p_k(t_{-j}) \\ &= \sum_{k \in N} \sum_{t \ni t_j^i} \frac{\varphi(t)}{\varphi(t_j^i)} d_{kj}(t) p_k(t) \\ &= \sum_{k \in N} \sum_{t_k \in T_k} \varphi(t_k) \sum_{t \ni t_k, t_j^i} \frac{\varphi(t)}{\varphi(t_j^i)\varphi(t_k)} d_{kj}(t) p_k(t_k) \\ &= \sum_{k \in N} \sum_{t_k \in T_k} \varphi(t_k) d_{kj}(t_k, t_j) p_k(t_k) \ , \end{aligned}$$

which is exactly the RHS of (26). □

We conclude with the following theorem.

Theorem 1. *Computing an optimal interim schedule together with optimal payments for the mechanism design problem can be done in time polynomial in the input size of the problem.*

Proof. The input size of the problem is $\Theta(m)$. The formulation (19)-(21) together with (26)-(29) has $O(m^2)$ variables and $O(m^2)$ constraints. Hence, this linear program can be solved in time polynomial in the input size. □

Now that we can compute optimal payments and interim schedule, two issues remain: The first is the interpretation of Theorem 1, because it is based on a relaxation and has a reduced number of variables. The second, which is an issue because we consider a relaxation, is the actual implementation of the optimal mechanism: We have to link the computed solution of the LP relaxation, specifically the computed interim schedule Es, to a (randomized) schedule $s(t)$ for any given type profile $t \in T$. The first issue is discussed next, the second in Section 4.

3.3 Discussion of the Result

We consider a true relaxation of the linear ordering polytope by dropping triangle and integrality constraints, yet the affine image of the variables $d_{kj}(t)$, respectively $d_{kj}(t_k, t_j)$, via (13) still yields a feasible point in the scheduling polytope. This allows us to interpret the solution as a (randomized) schedule; this is discussed in the next section. Also, we have drastically reduced the number of variables. It seems that thereby we are reducing the (number of) feasible mechanisms, because the variables $d_{kj}(t_k, t_j)$ only depend on the types of jobs k and j, while $d_{kj}(t)$ depends on the whole type vector t. For deterministic mechanisms, this is also known as *iia-property* [13].

Definition 1 (iia). *A deterministic scheduling rule is* independent of irrelavent alternatives*, or iia, if the relative order of two jobs does not depend on anything but the types of those two jobs, that is, $d_{kj}(t) = d_{kj}(t_k, t_j)$. We call a mechanism for which the scheduling rule is iia, an iia*-mechanism*.*

Lemma 1 shows that the reduction of variables is in fact no loss of generality for the relaxation. Interestingly, it *is* a loss of generality for the linear ordering polytope itself, respectively for the deterministic optimal mechanism design problem (9)-(18): Theorem 3 in Section 5 shows an optimality gap in general. With this in mind, a possible interpretation of Lemma 1 would be that the restriction to *iia*-mechansims is no loss of generality once randomization is allowed. But this interpretation is problematic, as the variables d_{kj} in the relaxation cannot in general be interpreted as the probability of job k preceding job j: By definition of the relaxation, neither the vector of variables $d_{kj}(t_k, t_j)$ nor $d_{kj}(t)$ do necessarily lie in the linear ordering polytope; see e.g. [8]. A detour via the scheduling polytope, however, fixes this.

4 Implementation

Recall from the previous discussion that the fractional solution in variables d_{kj} as suggested by the LP relaxation cannot in general be decomposed into linear orders, as it may lie outside the linear ordering polytope. Yet by taking the detour via the scheduling polytope we can easily fix this.

First, observe that for given solution Es and $d_{jk}(t_j, t_k)$, and fixed type vector t we can compute a corresponding vector of start times $s(t)$ by

$$s_j(t) = \sum_{k \in N} d_{kj}(t_j, t_k) p_k(t_k) \text{ for all } j\,.$$

Recall that $s(t)$ is simply a point in the scheduling polytope $Q(t)$ defined in (7) and (8), and the dimension of the scheduling polytope is $n-1$. It follows from Caratheodory's Theorem that $s(t)$ can be expressed as the convex combination of at most n vertices of $Q(t)$, that is, permutation schedules. In what follows, we describe a combinatorial algorithm to compute this representation, where for convenience, we drop the dependence on t.

A straightforward adaptation of a recent algorithm by Yasutake et al. [23] for the permutahedron results in an $O(n^2)$ algorithm. However, this outputs a convex combination of $O(n^2)$ vertices, while we know that a convex combination of at most n vertices exists. Therefore, we follow a geometric approach proposed by by Grötschel, Lovász and Schrijver in [11, Thm. 6.5.11]: Given some $s \in Q$, pick a (random) vertex v of Q, and compute the point $s' \in Q$ where the half-line through v and s leaves Q. This point lies on a facet of Q, and we can recurse on that facet[2]. However, we need a way to efficiently compute s' and a facet on

[2] Note that, independent of our work, and apparently also independent of [11], a similar decomposition algorithm is also suggested by Cai et al. [2,3]. References [11,2,3] do not result in combinatorial algorithms. However, in contrast to our work, they do address arbitrary polytopes.

which it lies. This can be done with an algorithm described by Fonlupt and Skoda in $O(n^8)$ time [9]. Here, we improve on this result for the scheduling polytope and give a simple algorithm that runs in $O(n^2 \log n)$ time. The total time for computing the representation of $s(t)$ as convex combination of $\leq n$ permutation schedules will be $O(n^3 \log n)$.

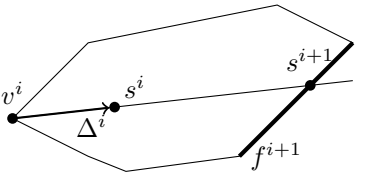


Fig. 1. Illustration of one iteration of Algorithm 1

Algorithm 1 (Decomposition Algorithm). *For a given point $s^i \in Q$ (in iteration i), order the jobs ascending in their start time s^i_j and define vertex v^i corresponding to that permutation schedule. We aim to find a point $s^{i+1} \in Q$ on a facet of Q such that $s^i = \lambda^i v^i + (1 - \lambda^i)s^{i+1}$, for some $\lambda^i \in [0, 1]$. Let $\Delta^i = s^i - v^i$. Then $\delta_{\max} := \max_{\delta \geq 0}\{v^i + \delta\Delta^i \in Q\}$, so that $s^{i+1} = v^i + \delta_{\max}\Delta^i$ and $\lambda^i = (1 - 1/\delta_{\max})$. If we now compute a facet f^{i+1} of Q containing s^{i+1}, we recurse with $s^{i+1} \in f^{i+1}$, and terminate after n iterations.*

The algorithm is illustrated in Figure 1. The following lemma is a consequence of our choice of vertex v^i; it shows that Algorithm 1 is well defined.

Lemma 2. *Both $v^i \in f^i$ and $s^i \in f^i$ (where $f^0 := Q$), hence $s^{i+1} \in f^i$.*

We are left to show that, in any iteration, computing s^{i+1} and f^{i+1} can be done in time $O(n^2 \log n)$. The crucial idea is that the set K^{i+1} that defines facet f^{i+1} can be computed from one of the $O(n^2)$ different orderings of the elements of the vectors on the half-line $L = \{v^i + \delta\Delta^i \mid \delta \geq 0\}$. There are no more than $O(n^2)$ such orderings, because the relative order of any two elements x_j an x_k, with $x \in L$, can change at most once while moving along L, by linearity.

Now imagine that the target point s^{i+1} lies on a facet defined by set $K^{i+1} \subseteq N$. Then, assuming for simplicity of notation that the ordering of elements of s^{i+1} is $s^{i+1}_1 \leq \cdots \leq s^{i+1}_n$, the set K^{i+1} appears as as one of the n nested sets $[k] := \{1, \ldots, k\}$, $k = 1, \ldots, n$. This follows directly from the simple separation algorithm for the scheduling polytope Q [18].

Since we do not know a priori which ordering the elements of s^{i+1} have, the simplest algorithm is to try them all, which works because we know that there are no more than $O(n^2)$ such orders for all points of L. Each of them gives n candidates for K^{i+1}, and computing their intersection with L yields s^{i+1} as the intersection point closest to s^i. This argument directly yields a $O(n^4)$ algorithm. With a more clever bookkeeping of the candidate sets, we end up with the following lemma; for details we refer to the full version of this paper.

Lemma 3. *The computation of vector s^{i+1} with $s^i = \lambda^i v^i + (1-\lambda^i)s^{i+1}$, and facet $f^{i+1} \ni s^{i+1}$ of Q in Algorithm 1 can be done in time $O(n^2 \log n)$.*

We can now conclude.

Theorem 2. *A point $s \in Q$ can be decomposed into the convex combination of at most n vertices (= permutation schedules) of Q in $O(n^3 \log n)$ time.*

5 Computational Results

We have implemented all models discussed in this paper; let us briefly comment on these experiments. As already mentioned, the most straightforward ILP formulation (1)-(6) for the deterministic mechanism design problem is colossal, which is confirmed by large computation times. In comparison, the linear ordering formulation (9)-(18), even though exponential in size as well, yields an average improvement in computation times of a factor 3-40 for small scale instances, depending on the model considered. In particular, the latter allows to drastically reduce the number of variables and constraints for *iia*-mechanisms, while the former formulation doesn't.

We end this short computational section by listing the following insights that we could obtain through generating random instances, and comparing the corresponding optimal solutions for different models. More detailed computational results are deferred to a full version of this paper.

Theorem 3. *Optimal deterministic mechanisms for both Bayes-Nash and dominant strategy implementations, in general do not satisfy the iia condition.*[3]

Theorem 4. *The optimal deterministic Bayes-Nash mechanism is generally not implementable in dominant strategies.*

Theorem 5. *Randomized Bayes-Nash mechanisms perform better than deterministic Bayes-Nash mechanisms in terms of total optimal payment.*

Proof. These theorems follow from instances which exhibit corresponding optimality gaps; they are deferred to a full version of this paper. □

6 Concluding Remarks

Our solution is randomized and truthful in expectation. The complexity to find an optimal deterministic mechanism remains open, and it is not even clear if the decision problem is contained in NP. An interesting future path to follow is to worst-case analyze the gaps between the solutions of different models.

Acknowledgements. Thanks to Maurice Queyranne for pointing us to [23], to Jelle Duives for his contribution in the experiments, and to Walter Kern, Marc Pfetsch, Rudolf Müller, and Gergely Csapó for helpful discussions. Also thanks to the thoughtful comments by an anonymous referee.

[3] Note: The example given in [13] to prove the same theorem is flawed.

References

1. Alaei, S., Fu, H., Haghpanah, N., Hartline, J., Malekian, A.: Bayesian Optimal Auctions via Multi- to Single-agent Reduction. In: Proc. 13th EC, p. 17 (2012)
2. Cai, Y., Daskalakis, C., Weinberg, S.M.: An Algorithmic Characterization of Multi-Dimensional Mechanisms. In: Proc. 44th STOC (2012)
3. Cai, Y., Daskalakis, C., Weinberg, S.M.: Optimal Multi-Dimensional Mechanism Design: Reducing Revenue to Welfare Maximization. In: Proc. 53rd FOCS (2012)
4. Conitzer, V., Sandholm, T.: Complexity of mechanism design. In: Proc. 18th Annual Conference on Uncertainty in Artificial Intelligence, UAI 2002, pp. 103–110 (2002)
5. Duives, J., Heydenreich, B., Mishra, D., Müller, R., Uetz, M.: Optimal Mechanisms for Single Machine Scheduling (2012) (manuscript)
6. Dyer, M.E., Wolsey, L.A.: Formulating the single machine sequencing problem with release dates as a mixed integer program. Disc. Appl. Math. 26, 255–270 (1990)
7. Edmonds, J.: Matroids and the greedy algorithm. Math. Prog. 1, 127–136 (1971)
8. Fishburn, P.C.: Induced binary probabilities and the linear ordering polytope: A status report. Mathematical Social Sciences 23, 67–80 (1992)
9. Fonlupt, J., Skoda, A.: Strongly polynomial algorithm for the intersection of a line with a polymatroid. In: Research Trends in Combinatorial Optimization, pp. 69–85. Springer (2009)
10. Gershkov, A., Goeree, J.K., Kushnir, A., Moldovanu, B., Shi, X.: On the Equivalence of Bayesian and Dominant Strategy Implementation. Econometrica 81, 197–220 (2013)
11. Grötschel, M., Lovász, L., Schrijver, A.: Geometric algorithms and combinatorial optimization. Algorithms and combinatorics. Springer (1988)
12. Hartline, J.D., Karlin, A.: Profit Maximization in Mechanism Design. In: Nisan, N., Roughgarden, T., Tardos, É., Vazirani, V. (eds.) Algorithmic Game Theory, ch. 13. Cambridge University Press (2007)
13. Heydenreich, B., Mishra, D., Müller, R., Uetz, M.: Optimal Mechanisms for Single Machine Scheduling. In: Papadimitriou, C., Zhang, S. (eds.) WINE 2008. LNCS, vol. 5385, pp. 414–425. Springer, Heidelberg (2008)
14. Kenis, P.: Waiting lists in Dutch health care: An analysis from an organization theoretical perspective. J. Health Organization and Mgmt. 20, 294–308 (2006)
15. Manelli, A.M., Vincent, D.R.: Bayesian and Dominant Strategy Implementation in the Independent Private Values Model. Econometrica 78, 1905–1938 (2010)
16. Myerson, R.B.: Optimal Auction Design. Math. OR 6, 58–73 (1981)
17. Nisan, N.: Introduction to Mechanism Design (for Computer Scientists). In: Nisan, N., Roughgarden, T., Tardos, É., Vazirani, V. (eds.) Algorithmic Game Theory, ch. 9. Cambridge University Press (2007)
18. Queyranne, M.: Structure of a simple scheduling polyhedron. Math. Prog. 58, 263–285 (1993)
19. Queyranne, M., Schulz, A.S.: Polyhedral Approaches to Machine Scheduling. TU Berlin Technical Report 408/1994
20. Sandholm, T.W.: Automated Mechanism Design: A New Application Area for Search Algorithms. In: Rossi, F. (ed.) CP 2003. LNCS, vol. 2833, pp. 19–36. Springer, Heidelberg (2003)
21. Smith, W.E.: Various optimizers for single-stage production. Naval Research Logistics Quarterly 3, 59–66 (1956)
22. Vohra, R.: Optimization and mechanism design. Math. Prog. 134, 283–303 (2012)
23. Yasutake, S., Hatano, K., Kijima, S., Takimoto, E., Takeda, M.: Online Linear Optimization over Permutations. In: Asano, T., Nakano, S.-i., Okamoto, Y., Watanabe, O. (eds.) ISAAC 2011. LNCS, vol. 7074, pp. 534–543. Springer, Heidelberg (2011)

Advances on Matroid Secretary Problems: Free Order Model and Laminar Case

Patrick Jaillet[1,⋆], José A. Soto[2,⋆⋆], and Rico Zenklusen[3,⋆⋆⋆]

[1] Dept. of Electrical Engineering and Computer Science, MIT
jaillet@mit.edu
[2] DIM-CMM University of Chile, and Technische Universität Berlin
jsoto@dim.uchile.cl
[3] Dept. of Applied Mathematics and Statistics, Johns Hopkins University
ricoz@jhu.edu

Abstract. The best-known conjecture in the context of matroid secretary problems claims the existence of an $O(1)$-approximation applicable to any matroid. Whereas this conjecture remains open, modified forms of it were shown to be true, when assuming that the assignment of weights to the secretaries is not adversarial but uniformly at random [20,18]. However, so far, no variant of the matroid secretary problem with adversarial weight assignment is known that admits an $O(1)$-approximation. We address this point by presenting a 9-approximation for the *free order model*, a model suggested shortly after the introduction of the matroid secretary problem, and for which no $O(1)$-approximation was known so far. The free order model is a relaxed version of the original matroid secretary problem, with the only difference that one can choose the order in which secretaries are interviewed.

Furthermore, we consider the classical matroid secretary problem for the special case of laminar matroids. Only recently, a $O(1)$-approximation has been found for this case, using a clever but rather involved method and analysis [12] that leads to a 16000/3-approximation. This is arguably the most involved special case of the matroid secretary problem for which an $O(1)$-approximation is known. We present a considerably simpler and stronger $3\sqrt{3}e \approx 14.12$-approximation, based on reducing the problem to a matroid secretary problem on a partition matroid.

1 Introduction

The secretary problem is a classical online selection problem of unclear origin [6,8,9,10,16]. In its original form, the task is to choose the best out of n

⋆ Supported in part by NSF grant 1029603, and by ONR grants N00014-12-1-0033 and N00014-09-1-0326.
⋆⋆ Supported in part by Núcleo Milenio Información y Coordinación en Redes ICM/FIC P10-024F.
⋆⋆⋆ Supported by NSF grants CCF-1115849 and CCF-0829878, and by ONR grants N00014-12-1-0033, N00014-11-1-0053 and N00014-09-1-0326.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 254–265, 2013.

secretaries, also called *elements* or *items*. Secretaries arrive (or are interviewed) one by one in random order. As soon as a secretary arrives, it can be ranked against all previously seen secretaries. Then, before the next one arrives, one has to decide irrevocably whether to choose the current secretary or not. There is a classical algorithm that selects the best secretary with probability $1/e$ [6], and this is known to be asymptotically optimal. In its initial form, the secretary problem was essentially a stopping time problem, and not surprisingly, it mainly attracted the interest of probabilists.

Recently, secretary problems enjoyed a revival, and various generalizations were studied. These developments are strongly motivated by a close connection to online mechanism design, where a good is sold to agents arriving online [13,2]. Here, the agents correspond to the secretaries and they reveal prices that they are willing to pay in exchange for goods. This leads to secretary problems where more than one secretary can be chosen. The most canonical generalization asks to hire k out of n secretaries, each revealing a non-negative weight upon arrival, and the goal is to hire a maximum weight subset of k secretaries. This interesting variant was introduced and studied by Kleinberg [13], who presented a $(1 - O(1/\sqrt{k}))$-approximation for this setting. However, in many applications, additional constraints have to be imposed on the elements that can be chosen. A very general class of constrained secretary problems, where the chosen elements have to form an independent set of a given matroid $M = (N, \mathcal{I})$, was introduced by Babaioff, Immorlica and Kleinberg [2][1]. This setting, now generally termed *matroid secretary problem*, covers at the same time many interesting cases and has a rich structure that can be exploited to design strong approximation algorithms.

To give a concrete example of a matroid secretary problem, and to motivate some of our results, consider the following connection problem. Given is an undirected graph $G = (V, E)$, representing a communication network, with non-negative edge-capacities $c : E \to \mathbb{N}$ and a server $r \in V$. Clients, which are the equivalent of candidates in the secretary problem, reside at vertices of the graph and are interested to connect to the server r via a unit-capacity path. The number of clients and their locations are known. Each client has a price that she is willing to pay to connect to the server. These prices are unknown and no assumptions are made on them except for being non-negative. Clients then reveal themselves one by one in random order, announcing their price. Whenever a client reveals herself, the network operator has to decide irrevocably before the next client appears whether to serve this client and receive the announced price. The goal is to choose a maximum weight subset of clients that can be served simultaneously without exceeding the given capacities c. It is well-known that the constraints imposed by the limited capacity on the clients that can be chosen is a special type of matroid constraint, namely a gammoid constraint [19].

[1] A matroid $M = (N, \mathcal{I})$ consists of a finite set N, called the *ground set*, and a non-empty family $\mathcal{I} \subseteq 2^N$ of subsets of N, called *independent sets*, satisfying: (i) $I \in \mathcal{I}, J \subseteq I \Rightarrow J \in \mathcal{I}$, and (ii) $I, J \in \mathcal{I}, |I| > |J| \Rightarrow \exists f \in I \setminus J$ with $J \cup \{f\} \in \mathcal{I}$. For more information on matroids we refer the reader to [19]

For the classical matroid secretary problem, as discussed above, the currently best approximation algorithm is a $O(\sqrt{\log \rho})$-approximation by Chakraborty and Lachish [4], where ρ is the rank of the matroid. This improved on an earlier $O(\log \rho)$-approximation of Babaioff, Immorlica and Kleinberg [2]. Babaioff et al. conjectured that there is an $O(1)$-approximation for the matroid secretary problem. This conjecture remains open and is arguably the currently most important open question regarding the matroid secretary problem.

Motivated by this conjecture, many interesting advances have been made to obtain $O(1)$-approximations, either for special cases of the matroid secretary problem or variants thereof. In particular, $O(1)$-approximations have been found for graphic matroids [2,15] (currently best approximation factor: $2e$), transversal matroids [2,5,15] (8-approximation), co-graphic matroids [20] ($3e$-approximation), linear matroids with at most k non-zero entries per column [20] (ke-approximation), and most recently laminar matroids [12] (16000/3-approximation). For most of the above special cases, strong approximation algorithms have been found, typically based on very elegant techniques. However for the laminar matroid, only a considerably higher approximation factor is known due to Im and Wang [12], using a very clever but quite involved method and analysis.

Furthermore, variants of the matroid secretary problem have been investigated that assume random instead of adversarial assignment of the weights, and for which $O(1)$-approximations can be obtained without any restriction on the underlying matroid. Recall that the classical matroid secretary problem does not make any assumptions on how weights are assigned to the elements, which means that we have to assume a worst-case, i.e., *adversarial*, weight assignment. However, the order in which the elements reveal themselves is assumed to be random. Soto [20] considered the variant where not only the arrival order of the elements is assumed to be uniformly random but also the assignment of the weights to the elements, and presented a $2e^2/(e-1)$-approximation for this case. More precisely, in this model, the weights can still be chosen by an adversary, but are then assigned uniformly at random to the elements of the matroid. Building on Soto's work, Vondrák and Oveis Gharan [18] showed that a $40e/(e-1)$-approximation can even be obtained when the arrival order of the elements is adversarial and the assignment of weights remains uniformly at random. Hence, this model is somehow the opposite of the classical matroid secretary problem, where assignment is adversarial and arrival order is random.

However, so far, no progress has been made in variants with adversarial assignment. One such variant, suggested shortly after the introduction of the matroid secretary problem [14], assumes that the appearance order of elements can be chosen by the algorithm. More precisely, in this model, which we call the *free order model*, whenever a next element has to reveal itself, the algorithm can choose the element to be revealed. E.g. in the above network connection problem, one could decide at each step which is the next client to reveal its price, by using for this decision the network structure and the elements observed so far. A main further complication when dealing with adversarial assignments—as in the free order model—contrary to random assignment, is that the knowledge of the

initial structure of the matroid seems to be of little help. This is due to the fact that an adversary can assign a weight of zero to most elements of the matroid, and only give a non-negative weight to a selected subset $A \subseteq N$ of elements. Hence, the problem essentially reduces to the restriction $M|_A$ of the matroid M over the elements A. However, the structure of $M|_A$ is essentially impossible to guess from M. This is in stark contrast to models with random assignment, e.g., in the model considered by Soto, the mentioned $2e^2/(e-1)$-approximation right at the start exploits the given structure of the matroid M, by partitioning N and solving a standard single secretary problem on each part of the partition. Different approaches are needed for adversarial weight assignments.

We are interested in the following two questions. First, is there an $O(1)$-approximation for the free order model? Second, we are interested in getting a better understanding of the laminar case of the classical secretary problem, with the goal to find considerably stronger and simpler procedures.

As it is common in this context, when we talk about a c-approximation we always compare against the *offline* optimum solution, i.e., the maximum weight independent set. In this type of analysis, known as *competitive analysis*, a c-approximation is also called a *c-competitive algorithm*.

Our Results and Techniques. We present a 9-approximation for the free order model, thus obtaining the first $O(1)$-approximation for a variant of the matroid secretary problem with adversarial weight assignment, without any restriction on the underlying matroid. This algorithm is in particular applicable to the mentioned network connection problem, when the order, in which the network operator negotiates with the clients, can be chosen. Previously, no matroid secretary model with adversarial weight assignment was known to admit an $O(1)$-approximation for this problem setting.

On a high level our algorithm follows a quite intuitive idea, which, interestingly, does not work in the traditional matroid secretary problem. In a first phase, we draw each element with probability 0.5 to obtain a set $A \subseteq N$, without selecting any element of A. Let OPT_A be the best offline solution in A. We call an element $f \in N \setminus A$ *good*, if it can be used to improve OPT_A, in the sense that either $\mathrm{OPT}_A \cup \{f\}$ is independent or there is an element $g \in \mathrm{OPT}_A$ such that $(\mathrm{OPT}_A \setminus \{g\}) \cup \{f\}$ is independent and has a higher value than OPT_A. In the second phase, we go through the remaining elements $N \setminus A$, drawing element by element in a well-chosen way to be specified soon. We accept an element $f \in N \setminus A$ if it is good and does not destroy independence when added to the elements accepted so far. Our approach fails if elements are drawn randomly in the second phase. The main problem when drawing randomly, is that we may accept good elements of relatively low value that may later *block* some high-valued good elements, in the sense that they cannot be added anymore without destroying independence of the selected elements. To overcome this problem, we determine after the first phase a specific order of how elements will be drawn in the second phase. The idea is to first draw elements of $N \setminus A$ that are in the span of elements of A of high weight. More precisely, let $A = \{a_1, \ldots, a_m\}$ be the numbering of

the elements of A according to decreasing weights. In the second phase we start by drawing elements of $(N \setminus A) \cap \text{span}(\{a_1\})$, then $(N \setminus A) \cap \text{span}(\{a_1, a_2\})$, and so on[2]. Intuitively, if there is a set $S \subseteq N$ with a high density of high-valued elements, then it is likely that many elements of S are part of A. Hence, high-valued elements of A span further high-valued elements in S. Thus, by the above order, we are likely to draw high-valued elements of S early, before they can be blocked by the inclusion of lower-valued elements.

Similar to previous secretary algorithms, we show that our algorithm is a $O(1)$-approximation by proving that each element $f \in \text{OPT}$ of the global offline optimum OPT will be chosen with probability at least $1/9$. However, the way we prove this is based on a novel approach. Broadly speaking, we show that for any element $f \in \text{OPT}$ there is a threshold weight $\overline{w}_f$ such that with constant positive probability we have simultaneously: (i) $f \notin A$, (ii) f is spanned by the elements in A with weight $\geq \overline{w}_f$, and (iii) good elements considered in the second phase with weight at least $\overline{w}_f$ do not block f. From this we can observe that f gets selected with constant probability. Interestingly, several probabilities of interest that appear in our analysis are very hard to compute exactly. E.g., even when all weights are known and a threshold $\overline{w}_f$ is given, it is in general $\#P$-hard to compute the probability that f is in the span of all elements of A of weight at least $\overline{w}_f$ [3]. Still, we can show that a good threshold weight $\overline{w}_f$ exists, which is all we need to guarantee that our algorithm is a $O(1)$-approximation.

Furthermore, we present a new approach to deal with laminar matroids in the classical matroid secretary model. Our technique leads to a $3\sqrt{3}e \approx 14.12$-approximation, thus considerably improving on the $16000/3 \approx 5333$-approximation of Im and Wang [12]. Our main contribution here is to present a simple way to transform the matroid secretary problem on a laminar matroid M to one on a unitary partition matroidM_P by losing only a small constant factor of $3\sqrt{3} \approx 5.2$. The secretary problem on M_P can then simply be solved by applying the classical e-approximation for the standard secretary problem to each partition of M_P. We first observe a constant fraction of all elements, on the basis of which a partition matroid M_P on the remaining elements is then constructed. To assure feasibility, M_P is defined such that each independent set of M_P is as well an independent set of M. To best convey the main ideas of our procedure, we focus on a very simple method to obtain a weaker $27e/2 \approx 36.7$-approximation, which already improves considerably on the 16000/3-approximation of Im and Wang. The $3\sqrt{3}e$-approximation is obtained through a strengthening of this approach by using a stronger partition matroid M_P and a tighter analysis.

We remark that the algorithms we present do not need to observe the exact weights of the items when they reveal themselves, but only need to be able to

[2] We recall that $\text{span}(S)$ for $S \subseteq N$ is the unique maximal set $U \supseteq S$ with the same rank as S.

[3] Consider for example the graphic matroid with underlying graph $G = (V, E)$. Here, the question whether some edge $\{s, t\} \in E$ is in the span of a random set of edge $A \subseteq E$ containing each edge with probability 0.5, reduces to the question of whether A contains an s-t path. This is the well-known $\#P$-hard s-t reliability problem [21].

compare the weights of elements observed so far. This is a common feature of many matroid secretary algorithms and matroid algorithms more generally.

To simplify the exposition, we assume that all weights are distinct, i.e., they induce a linear order on the elements. This implies in particular, that there is a unique maximum weight independent set. The general case with possibly equal weights easily reduces to this case by breaking ties arbitrarily between elements of equal weight to obtain a linear order.

Related Work. We mention briefly that recently, matroid secretary problems with submodular objective functions have been considered. For this setting, $O(1)$-approximations have been found for knapsack constraints, uniform matroids, and more generally for partition matroids if the submodular objective function is furthermore monotone [3,7,11].

Additionally, variations of the matroid secretary problem have been considered with restricted knowledge on the underlying matroid type. This includes the case where no prior knowledge of the underlying matroid is assumed except for the size of the ground set. Or even more extremely, the case without even knowing the size of the ground set. For more information on such variations we refer to the excellent overview in [18].

Subsequent Results. We would like to highlight that very recently, Ma, Tang and Wang [17] further improved the currently best approximation ratio for the secretary problem on laminar matroids by presenting a 9.6-approximation.

Organization of the Paper. Our 9-approximation for the free order model is presented in Section 2. Section 3 discusses our simple $27e/2$-approximation for the classical matroid secretary problem restricted to laminar matroids. Due to space constraints, we defer details of how this algorithm can be strengthened to obtain the claimed $3\sqrt{3}e$-approximation to a long version of this paper.

2 A 9-Approximation for the Free Order Model

To simplify the writing we use "+" and "−" for the addition and subtraction of single elements from a set, i.e., $S+f-g = (S\cup\{f\})\setminus\{g\}$. Algorithm 1 describes our 9-approximation for the free order model. It operates in two phases.

As mentioned previously, a *good* element $f \in N \setminus A$ is an element that allows for improving the maximum weight independent set in A. Using standard results on matroids, an element f is good if either $f \notin \operatorname{span}(A)$, or if there is an index $i \in \{1,\dots m\}$ such that $f \in \operatorname{span}(A_i) \setminus \operatorname{span}(A_{i-1})$ and $\mathrm{w}(f) > \mathrm{w}(a_i)$. Hence, our algorithm indeed only accepts good elements.

To show that Algorithm 1 is a 9-approximation, we show that each element f of the offline optimum OPT will be contained in the set I returned by the algorithm with probability at least $1/9$. We distinguish two cases:

(i) $\Pr[f \in \operatorname{span}(A-f)] \leq 1/3$, and
(ii) $\Pr[f \in \operatorname{span}(A-f)] > 1/3$.

Algorithm 1. A 9-approximation for the free order model

1. **Draw** each element with probability 0.5 to obtain $A \subseteq N$, without selecting any element of A. We number the elements of $A = \{a_1, \ldots, a_m\}$ in decreasing order of weights. Define $A_i = \{a_1, \ldots, a_i\}$, with $A_0 = \emptyset$.
   Initialize: $I \leftarrow \emptyset$.
2. **For** $i = 1$ to m:
   draw one by one (in any order) all elements $f \in (\text{span}(A_i) \setminus \text{span}(A_{i-1})) \setminus A$.
   if $I + f \in \mathcal{I}$ and $\text{w}(f) > \text{w}(a_i)$, **then** $I = I + f$.
   For all remaining elements $f \in N \setminus \text{span}(A)$ (drawn in any order):
   if $I + f \in \mathcal{I}$, **then** $I = I + f$.
   Return I

The following lemma handles the simpler first case, which allows us to highlight some ideas that we will also employ to prove the more interesting second case. Notice that in the following statement we do not even have to assume $f \in \text{OPT}$.

Lemma 1. *Let $f \in N$ with $\Pr[f \in \text{span}(A - f)] \leq 1/3$. Then f is selected by Algorithm 1 with probability at least $1/6$.*

Proof. We start by observing that f is selected by Algorithm 1 if the three events $E_1 : f \notin A$, $E_2 : f \notin \text{span}(A - f)$ and $E_3 : f \notin \text{span}((N \setminus A) - f)$ happen simultaneously. Indeed, if $E_1 \cap E_2$ occurs, then f will be considered during the second for-loop of the second phase of Algorithm 1. Furthermore, adding f at that moment will not violate independence since the elements selected so far are a subset of $N \setminus A$, and if E_3 holds we have $f \notin \text{span}((N \setminus A) - f)$. It therefore suffices to show that the probability of E_1, E_2, E_3 happening simultaneously is at least $1/6$.

Notice that E_1 is independent of E_2, E_3. Hence,

$$\Pr[E_1 \cap E_2 \cap E_3] = \Pr[E_1] \cdot \Pr[E_2 \cap E_3] = \frac{1}{2} \cdot \Pr[E_2 \cap E_3]. \tag{1}$$

Furthermore, we observe that A and $N \setminus A$ have the same distribution since they contain each element of N with probability 0.5. Hence,

$$\Pr[E_3] = \Pr[E_2] = 1 - \Pr[f \in \text{span}(A - f)] \geq \frac{2}{3}.$$

Denoting by $\overline{E_2}$ and $\overline{E_3}$ the complements of E_2 and E_3, respectively, we thus obtain by the union bound:

$$\Pr[E_2 \cap E_3] = 1 - \Pr[\overline{E_2} \cup \overline{E_3}] \geq 1 - \Pr[\overline{E_2}] - \Pr[\overline{E_3}] = \Pr[E_2] + \Pr[E_3] - 1 \geq \frac{1}{3}.$$

Combining the above with (1) we obtain $\Pr[E_1 \cap E_2 \cap E_3] \geq 1/6$. □

We now consider the case $f \in \text{OPT}$ with $\Pr[f \in \text{span}(A - f)] > 1/3$. Let $N = \{f_1, \ldots, f_n\}$ be the numbering of all elements in decreasing order of weights, and

let $N_j = \{f_1, \ldots, f_j\}$ with $N_0 = \emptyset$. This time, we want to show that with constant probability, f is chosen in the first for-loop of the second phase of Algorithm 1. More precisely, we want to find a good threshold weight $\overline{\mathrm{w}}_f$ as discussed in the introduction. For this we determine an index $\bar{j} \in \{1, \ldots, n\}$—and $\overline{\mathrm{w}}_f$ will then correspond to $\mathrm{w}(f_{\bar{j}})$—satisfying two properties. First, we want that with constant positive probability, $f \in \mathrm{span}((A \cap N_{\bar{j}}) - f)$. The benefit of having $f \in \mathrm{span}((A \cap N_{\bar{j}}) - f)$ is that if additionally $f \notin A$, then f will be considered in the first for-loop of phase two at some iteration i with $\mathrm{w}(a_i) \geq \mathrm{w}(f_{\bar{j}})$. Hence, up to that point, only elements with weight $\geq \mathrm{w}(f_{\bar{j}})$ have been selected. Thus, when checking whether f can be added without violating independence, only those elements have to be considered. Second, we want that $\Pr[f \in \mathrm{span}((A \cap N_{\bar{j}}) - f)]$ is also bounded away from 1, because this implies that $\Pr[f \notin \mathrm{span}((N_{\bar{j}} \setminus A) - f)] = \Pr[f \notin \mathrm{span}((A \cap N_{\bar{j}}) - f)]$ is some constant > 0. Whenever $f \notin \mathrm{span}((N_{\bar{j}} \setminus A) - f)$ occurs, then f will not violate independence when added to any set of selected elements with weight at least $\mathrm{w}(f_{\bar{j}})$, since they are a subset of $N_{\bar{j}} \setminus A$. Hence, intuitively, for our analysis we want to find an index $\bar{j}$ such that $\Pr[f \in \mathrm{span}((A \cap N_{\bar{j}}) - f)]$ is bounded away from zero and from one. The following lemma shows that such an index indeed exists. In Lemma 3, we then show how the above sketch of our proof can be formalized, and in particular, how to deal with dependencies of the different events discussed above.

For brevity, let $p_j = \Pr[f \in \mathrm{span}((A \cap N_j) - f)]$.

Lemma 2. *Let $f \in N$ with $\Pr[f \in \mathrm{span}(A - f)] \geq 1/3$. There exists an index $\bar{j} \in \{1, \ldots, n\}$, such that $p_{\bar{j}} \in [1/3, 2/3]$.*

Proof. By assumption we have $p_n = \Pr[f \in \mathrm{span}(A - f)] > 1/3$. Furthermore, $p_0 = 0$. Since p_j is increasing in j, to prove the proposition it suffices to show that $p_{j+1} \leq 2/3$, for all $j \in \{0, \ldots, n-1\}$ such that $p_j < 1/3$. This indeed holds due to the following:

$$p_{j+1} = \underbrace{\Pr[f_{j+1} \notin A]}_{=0.5} \cdot \underbrace{\Pr[f \in \mathrm{span}((A \cap N_{j+1}) - f) \mid f_{j+1} \notin A]}_{=p_j} +$$

$$\underbrace{\Pr[f_{j+1} \in A]}_{=0.5} \cdot \underbrace{\Pr[f \in \mathrm{span}((A \cap N_{j+1}) - f) \mid f_{j+1} \in A]}_{\leq 1} \leq \frac{1}{2}(p_j + 1).$$

□

The following completes the proof of the case $\Pr[f \in \mathrm{span}(A - f)] > 1/3$.

Lemma 3. *Let $f \in \mathrm{OPT}$ with $\Pr[f \in \mathrm{span}(A - f)] > 1/3$. Then f is selected by Algorithm 1 with probability at least $1/9$.*

Proof. Let $\bar{j} \in \{1, \ldots, n\}$ be an index with $p_{\bar{j}} \in [1/3, 2/3]$ as claimed by Lemma 2. We start by reasoning that f will be selected by Algorithm 1 if the three events $E_1 : f \notin A$, $E_2 : f \in \mathrm{span}((A \cap N_{\bar{j}}) - f)$, and $E_3 : f \notin \mathrm{span}((N_{\bar{j}} \setminus A) - f)$ happen simultaneously. Notice that $E_1 \cap E_2$ implies that f will be considered during the first for-loop of the second phase of Algorithm 1, at some iteration i with $\mathrm{w}(a_i) \geq \mathrm{w}(f_{\bar{j}})$. Since the elements selected so far—at the

time f is considered—must all have a weight of at least $\mathrm{w}(a_i) \geq \mathrm{w}(f_{\bar{j}})$, the occurrence of E_3 guarantees that f can be added without violating independence since the selected elements at that point are a subset of $N_{\bar{j}} \setminus A$. Also notice that since $f \in \mathrm{OPT}$ and $f \in \mathrm{span}(A_i)$, we have $\mathrm{w}(f) > \mathrm{w}(a_i)$, i.e., f is good. Therefore, f indeed gets selected if E_1, E_2, E_3 occur simultaneously. Hence it suffices to show that $E_1 \cap E_2 \cap E_3$ occurs with probability $\geq 1/9$. Again, E_1 is independent of E_2, E_3, and hence

$$\Pr[E_1 \cap E_2 \cap E_3] = \Pr[E_1] \cdot \Pr[E_2 \cap E_3] = \frac{1}{2} \cdot \Pr[E_2 \cap E_3]. \tag{2}$$

To deal with the dependence between the events E_2 and E_3 we invoke the FKG inequality (see [1]). Notice that both events E_2 and E_3 are *increasing* in A, i.e., for any two sets $Q, P \subseteq N$ with $Q \subseteq P$, if E_2 (or E_3) occurs for $A = Q$ then it also occurs if $A = P$. The FKG inequality then implies

$$\Pr[E_2 \cap E_3] \geq \Pr[E_2] \cdot \Pr[E_3]. \tag{3}$$

Furthermore, since $A \cap N_{\bar{j}}$ has the same distribution as $N_{\bar{j}} \setminus A$, we have $\Pr[E_3] = 1 - \Pr[E_2]$. Hence, together with (2) and (3) we obtain

$$\Pr[E_1 \cap E_2 \cap E_3] = \frac{1}{2} \cdot \Pr[E_2] \cdot (1 - \Pr[E_2]).$$

Due to our choice of $\overline{j}$, we have $\Pr[E_2] \in [1/3, 2/3]$, and hence $\Pr[E_2] \cdot (1 - \Pr[E_2]) \geq 2/9$, thus leading to $\Pr[E_1 \cap E_2 \cap E_3] \geq 1/9$ as desired. □

Finally, by combining Lemma 1 and Lemma 3 we obtain.

Corollary 1. *Algorithm 1 is a 9-approximation for the free order model.*

3 Classical Secretary Problem for Laminar Matroids

Let $M = (N, \mathcal{I})$ be a laminar matroid whose constraints are defined by the laminar family $\mathcal{L} \subseteq 2^N$ with upper bounds b_L for $L \in \mathcal{L}$ on the number of elements that can be chosen from $\mathcal{L}$, i.e., $\mathcal{I} = \{I \subseteq N \mid |I \cap L| \leq b_L \ \forall L \in \mathcal{L}\}$. Without loss of generality we assume $b_L \geq 1$ for $L \in \mathcal{L}$, since otherwise we can simply remove all elements of L from M. Furthermore, we assume $N \in \mathcal{L}$, since otherwise a redundant constraint $|I \cap N| \leq b_N$ can be added by choosing a sufficiently large right-hand side b_N.

To reduce the matroid secretary problem on M to a problem on a partition matroid, we first number the elements $N = \{f_1, \ldots, f_n\}$ such that for any set $L \in \mathcal{L}$, the elements in L are numbered consecutively, i.e., $L = \{f_p, \ldots, f_q\}$ for some $1 \leq p < q \leq n$. Figure 1 shows an example of such a numbering.

For the sake of exposition, we start by presenting a conceptually simple algorithm and analysis, based on the introduced numbering of the ground set, that leads to a $27e/2$-approximation. The claimed $3\sqrt{3}e$-approximation follows the same ideas, but strengthens both the approach and analysis. Algorithm 2

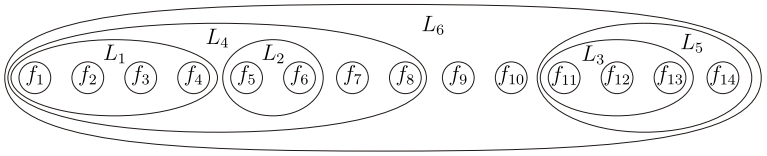


Fig. 1. An example of a numbering of the elements of the ground set such that each set $L \in \mathcal{L} = \{L_1, \ldots, L_6\}$ of the laminar family contains consecutively numbered elements

Algorithm 2. A $27e/2$-approximation for laminar matroids

1. **Observe** $\mathrm{Binom}(n, 2/3)$ elements of N, which we denote by $A \subseteq N$.
   Determine maximum weight independent set $\mathrm{OPT}_A = \{f_{i_1}, \ldots, f_{i_p}\}$ in A where $1 \leq i_1 < \cdots < i_p \leq n$. Define $P_j = \{f_k \mid k \in \{i_{j-1}, \ldots, i_j\}\} \setminus A$ for $j \in \{1, \ldots, p+1\}$, where we set $i_0 = 0, i_{p+1} = n$. Let
   $$\mathcal{P}_{\mathrm{odd}}(A) = \{P_j \mid j \in \{1, \ldots, p+1\}, j \text{ odd}\},$$
   $$\mathcal{P}_{\mathrm{even}}(A) = \{P_j \mid j \in \{1, \ldots, p+1\}, j \text{ even}\}.$$
   If $\mathrm{OPT}_A = \emptyset$ **then** set $\mathcal{P} = \{N \setminus A\}$,
   else set $\mathcal{P} = \mathcal{P}_{\mathrm{odd}}(A)$ with probability 0.5, otherwise set $\mathcal{P} = \mathcal{P}_{\mathrm{even}}(A)$.
2. **Apply** to each set $P \in \mathcal{P}$ an e-approximate classical secretary algorithm to obtain an element $g_P \in P$.
   Return $\{g_P \mid P \in \mathcal{P}\}$.

describes our $27e/2$-approximation. Notice that applying a standard secretary algorithm to the sets of $\mathcal{P}$ in step 2 can easily be performed by running $|\mathcal{P}|$ many e-approximate secretary algorithms in parallel, one for each set $P \in \mathcal{P}$. Elements are drawn one by one in the second phase, and they are forwarded to the secretary algorithm corresponding to the set P that contains the drawn element, and are discarded if no set of $\mathcal{P}$ contains the element. Furthermore, observe that A contains each element of N independently with probability $2/3$.

We start by observing that Algorithm 2 returns an independent set.

Lemma 4. *Let $A \subseteq N$ with $\mathrm{OPT}_A \neq \emptyset$ and let $\mathcal{P} \in \{\mathcal{P}_{\mathrm{even}}(A), \mathcal{P}_{\mathrm{odd}}(A)\}$. For each $P \in \mathcal{P}$, let g_p be any element in P. Then $\{g_P \mid P \in \mathcal{P}\} \in \mathcal{I}$.*

Proof. Let $I = \{g_P \mid P \in \mathcal{P}\}$ be a set as stated in the lemma. Notice that for any two elements $f_k, f_\ell \in I$ with $k < \ell$ we have $|\mathrm{OPT}_A \cap \{f_k, f_{k+1}, \ldots, f_\ell\}| \geq 2$. Now consider a set $L \in \mathcal{L}$ corresponding to one of the constraints of the underlying laminar matroid. By the above observation and since L is consecutively numbered, at least one of the following holds: (i) $|L \cap I| = 1$, or (ii) $|L \cap \mathrm{OPT}_A| \geq |L \cap I|$. If case (i) holds, then the constraint corresponding to L is not violated since we assumed $b_L \geq 1$. If (ii) holds, then L is also not violated since $|L \cap I| \leq |L \cap \mathrm{OPT}_A| \leq b_T$ because $\mathrm{OPT}_A \in \mathcal{I}$. Hence $I \in \mathcal{I}$. □

Theorem 1. *Algorithm 2 is a $27e/2$-approximation for the laminar matroid secretary problem.*

Proof. Let $\mathrm{OPT} \in \mathcal{I}$ be the maximum weight independent set in N, i.e., the offline optimum. Furthermore, let I be the set returned by Algorithm 2, and let $f \in \mathrm{OPT}$. We say that f is *solitary* if $\exists\, P \in \mathcal{P}$ with $P \cap \mathrm{OPT} = \{f\}$. Similarly we call $P \in \mathcal{P}$ *solitary* if $|P \cap \mathrm{OPT}| = 1$. We prove the theorem by showing that each element $f \in \mathrm{OPT}$ is solitary with probability $\geq 2/27$. This indeed implies the theorem since we can do the following type of accounting. Let X_f be the random variable which is zero if f is not solitary, and otherwise if f is solitary, X_f equals the weight of the element $g \in I$ that was chosen by the algorithm out of P that contains f. By only considering the weights of elements chosen in solitary sets $\mathcal{P}$ we obtain

$$\mathbf{E}[w(I)] \geq \sum_{f \in \mathrm{OPT}} \mathbf{E}[X_f]. \tag{4}$$

However, if each element $f \in \mathrm{OPT}$ is solitary with probability $2/27$, we obtain $\mathbf{E}[X_f] \geq \frac{2w(f)}{27e}$, because the classical secretary algorithm will choose with probability $1/e$ the maximum weight element of the set P that contains the solitary element f. Combining this with (4) yields $\mathbf{E}[w(I)] \geq \frac{2}{27e} w(\mathrm{OPT})$ as desired. It remains to show that each $f \in \mathrm{OPT}$ is solitary with probability $\geq 2/27$.

Let $f_i \in \mathrm{OPT}$. We assume that OPT contains an element with a lower index than i and one with a higher index than i. The cases of f_i being the element with highest or lowest index in OPT follow analogously. Let $f_j \in \mathrm{OPT}$ by the element of OPT with the largest index $j < i$. Similarly, let $f_k \in \mathrm{OPT}$ be the element of OPT with the smallest index $k > i$. One well-known matroidal property that we use is $\mathrm{OPT} \cap A \subseteq \mathrm{OPT}_A$. Hence, if $f_j, f_k \in A$ then also $f_j, f_k \in \mathrm{OPT}_A$, and if furthermore $f_i \notin A$, then f_i will be the only element of OPT in the set $P \in \mathcal{P}_{\mathrm{odd}}(A) \cup \mathcal{P}_{\mathrm{even}}(A)$ that contains f_i. Hence, if the coin flip in Algorithm 2 chooses the family $\mathcal{P} \in \{\mathcal{P}_{\mathrm{odd}}(A), \mathcal{P}_{\mathrm{even}}(A)\}$ that contains P, then f_i is solitary. To summarize, f_i is solitary if $f_j, f_k \in A$, $f_i \notin A$ and the coin flip for $\mathcal{P}$ turns out right. This happens with probability $\left(\frac{2}{3}\right)^2 \cdot \left(1 - \frac{2}{3}\right) \cdot \frac{1}{2} = \frac{2}{27}$. □

One conservative aspect of the proof of Lemma 1 is that we only consider the contribution of solitary elements. Additionally, a drawback of Algorithm 2 itself is that about half of the elements of $N \setminus A$ are ignored as we only select from either $\mathcal{P}_{\mathrm{odd}}(A)$ or $\mathcal{P}_{\mathrm{even}}(A)$. Addressing these weaknesses, the claimed $3\sqrt{3}e$-approximation can be obtained. Details are omitted due to space constraints.

References

1. Alon, N., Spencer, J.: The Probabilistic Method, 3rd edn. John Wiley & Sons (2008)
2. Babaioff, M., Immorlica, N., Kleinberg, R.: Matroids, secretary problems, and online mechanisms. In: Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, pp. 434–443 (2007)
3. Bateni, M., Hajiaghayi, M., Zadimoghaddam, M.: Submodular Secretary Problem and Extensions. In: Serna, M., Shaltiel, R., Jansen, K., Rolim, J. (eds.) APPROX and RANDOM 2010. LNCS, vol. 6302, pp. 39–52. Springer, Heidelberg (2010)

4. Chakraborty, S., Lachish, O.: Improved competitive ratio for the matroid secretary problem. In: Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, pp. 1702–1712 (2012)
5. Dimitrov, N.B., Plaxton, C.G.: Competitive Weighted Matching in Transversal Matroids. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part I. LNCS, vol. 5125, pp. 397–408. Springer, Heidelberg (2008)
6. Dynkin, E.B.: The optimum choice of the instant for stopping a markov process. Soviet Mathematics, Doklady 4 (1963)
7. Feldman, M., Naor, J(S.), Schwartz, R.: Improved Competitive Ratios for Submodular Secretary Problems (Extended Abstract). In: Goldberg, L.A., Jansen, K., Ravi, R., Rolim, J.D.P. (eds.) APPROX/RANDOM 2011. LNCS, vol. 6845, pp. 218–229. Springer, Heidelberg (2011)
8. Ferguson, T.S.: Who solved the secretary problem? Statistical Science 4(3), 282–296 (1989)
9. Gardner, M.: Mathematical games column. Scientific American 202(2), 150–154 (1960)
10. Gardner, M.: Mathematical games column. Scientific American 202(3), 172–182 (1960)
11. Gupta, A., Roth, A., Schoenebeck, G., Talwar, K.: Constrained Non-monotone Submodular Maximization: Offline and Secretary Algorithms. In: Saberi, A. (ed.) WINE 2010. LNCS, vol. 6484, pp. 246–257. Springer, Heidelberg (2010)
12. Im, S., Wang, Y.: Secretary problems: Laminar matroid and interval scheduling. In: Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, pp. 1265–1274 (2011)
13. Kleinberg, R.: A multiple-choice secretary algorithm with applications to online auctions. In: Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, pp. 630–631 (2005)
14. Kleinberg, R.: Personal Communication (2012)
15. Korula, N., Pál, M.: Algorithms for Secretary Problems on Graphs and Hypergraphs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009, Part II. LNCS, vol. 5556, pp. 508–520. Springer, Heidelberg (2009)
16. Lindley, D.V.: Dynamic programming and decision theory. Journal of the Royal Statistical Society. Series C (Applied Statistics) 10(1), 39–51 (1961)
17. Ma, T., Tang, B., Wang, Y.: The simulated greedy algorithm for several submodular matroid secretary problems. CoRR, abs/1107.2188v2 (2012)
18. Oveis Gharan, S., Vondrák, J.: On Variants of the Matroid Secretary Problem. In: Demetrescu, C., Halldórsson, M.M. (eds.) ESA 2011. LNCS, vol. 6942, pp. 335–346. Springer, Heidelberg (2011)
19. Schrijver, A.: Combinatorial Optimization, Polyhedra and Efficiency. Springer (2003)
20. Soto, J.A.: Matroid secretary problem in the random assignment model. In: Proceedings of the 22nd Annual ACM -SIAM Symposium on Discrete Algorithms, SODA, pp. 1275–1284 (2011)
21. Valiant, L.G.: The complexity of enumeration and reliability problems. SIAM Journal on Computing 8(3), 410–421 (1979)

A Polynomial-Time Algorithm to Check Closedness of Simple Second Order Mixed-Integer Sets

Diego Alejandro Morán Ramirez and Santanu S. Dey

Industrial and Systems Engineering, Georgia Institute of Technology

Abstract. Let $\mathbf{L}^m$ be the Lorentz cone in $\mathbb{R}^m$. Given $A \in \mathbb{Q}^{m\times n_1}$, $B \in \mathbb{Q}^{m\times n_2}$ and $b \in \mathbb{Q}^m$, a *simple* second order conic mixed-integer set (SOCMIS) is a set of the form $\{(x,y) \in \mathbb{Z}^{n_1} \times \mathbb{R}^{n_2} \mid Ax + By - b \in \mathbf{L}^m\}$. We show that there exists a polynomial-time algorithm to check the closedness of the convex hull of simple SOCMISs. Moreover, in the special case of pure integer problems, we present sufficient conditions, that can be checked in polynomial-time, to verify the closedness of intersection of simple SOCMISs.

Keywords: Closedness, Polynomial-time algorithm, Mixed-integer convex programming.

1 Introduction

Understanding the structure of convex hulls of mixed-integer feasible solutions has proven to be critical in the design of various algorithms for solving mixed-integer programs. In the case of mixed-integer linear programs, a particularly important result in this direction, due to Meyer [1], states that the integer hull of a rational polyhedron is a rational polyhedron.

In this paper, we study properties of convex hulls of a class of simple mixed-integer nonlinear sets of the form:

$$P := \{(x,y) \in \mathbb{Z}^{n_1} \times \mathbb{R}^{n_2} \mid Ax + By - b \in \mathbf{L}^m\},$$

where A and B are rational matrices of suitable dimensions, b is a rational vector and $\mathbf{L}^m$ is the Lorentz cone in $\mathbb{R}^m$. In contrast to the case of mixed-integer linear programs, the convex hull of feasible solutions of P is unlikely a rational polyhedron. We therefore explore a more basic question:

Is the convex hull of P closed?

While the convex hull of P is not always closed, we are able to provide a characterization of when it is closed. This characterization yields a polynomial-time algorithm to verify if the convex hull of P is closed or not. We find it interesting that it is possible to construct an algorithm (let alone one that runs in polynomial-time) to test the closedness of integer hulls of unbounded nonlinear sets.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 266–277, 2013.

2 Main Results

The Lorentz cone $\mathbf{L}^m$ in $\mathbb{R}^m$ is defined as

$$\mathbf{L}^m := \{(w, z) \in \mathbb{R}^{m-1} \times \mathbb{R} \,|\, \|w\| \leq z\},$$

where $\|\cdot\|$ is the usual Euclidean norm. For $a, b \in \mathbb{R}^m$ we say that $a \succeq_{\mathbf{L}^m} b$ if and only if $a - b \in \mathbf{L}^m$. Given a matrix B, we use the notation $\langle B \rangle$ to denote the linear subspace generated by the columns of the matrix B. For a set U we denote its dimension by $\dim(U)$.

Our first result is a characterization of closedness of integer hulls of simple second order mixed-integer sets.

Theorem 1. *Let $\mathbf{L}^m \subseteq \mathbb{R}^m$ be the Lorentz cone. Let $A \in \mathbb{Q}^{m \times n_1}$, $B \in \mathbb{Q}^{m \times n_2}$. Let*

$$\mathcal{P} := \{(x, y) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \,|\, Ax + By \succeq_{\mathbf{L}^m} b\}, \tag{1}$$

$V := \{Ax + By - b \,|\, (x, y) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}\}$ and $\mathcal{L} := \{Ax + By \,|\, (x, y) \in \mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}\}$. Then $\operatorname{conv}(\mathcal{P} \cap (\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}))$ is closed if and only if one of the following holds:

1. *$b \notin \mathcal{L}$.*
2. *$b \in \mathcal{L}$, and $\dim(\mathbf{L}^m \cap V) \leq 1$.*
3. *$b \in \mathcal{L}$, $\dim(\mathbf{L}^m \cap V) = 2$, $\dim(\langle B \rangle) \leq 0$ and the two extreme rays of the cone $\mathbf{L}^m \cap V$ can be scaled by a non-zero scalar so that they belong to the lattice $\{Ax \,|\, x \in \mathbb{Z}^{n_1}\}$.*
4. *$b \in \mathcal{L}$, $\dim(\mathbf{L}^m \cap V) \geq 2$ and $\dim(\langle B \rangle) \geq \dim(V) - 1$.*

The proof of Theorem 1 relies on three sets of results: (1) Understanding when affine rational maps preserve closedness. (2) In a recent paper [2] we presented some properties on the closedness of integer hulls of closed convex sets in the pure integer case, with applications to strictly convex sets and cones. We generalize these results from the pure integer case to the mixed-integer case. (3) Geometric properties of the Lorentz cone. We present a sketch of the proof in Section 3.

Theorem 1 yields a polynomial-time algorithm to check the closedness of simple second order mixed-integer sets. Note that for $\mathcal{P}$ given by (1) we denote by $\operatorname{size}(\mathcal{P})$ the sum of the size of the (usual) binary representation of the matrices A, B and vector b. Formally we have the following result.

Theorem 2. *Let $A \in \mathbb{Q}^{m \times n_1}$, $B \in \mathbb{Q}^{m \times n_2}$, $b \in \mathbb{Q}^m$ and let $\mathcal{P}$ be as defined in (1). There exists an algorithm that runs in polynomial-time with respect to the $\operatorname{size}(\mathcal{P})$ to check whether $\operatorname{conv}(\mathcal{P} \cap (\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}))$ is closed or not.*

The algorithm in Theorem 2 is constructed by showing that each of the cases described in Theorem 1 can be verified in polynomial-time. Among the four cases, the most 'interesting' case is case (3.). To check this case in polynomial-time, the key idea is to reduce this case to checking whether a suitable number is a perfect square, via the use of Hermite normal form algorithm and properties of the Lorentz cone. We present a proof of Theorem 2 in Section 4.

In the case of pure integer programs, we prove the following general result.

Theorem 3. *Let $K_i \subseteq \mathbb{R}^n$, $i = 1, 2$ be closed convex sets. Assume* $\text{conv}(K_i \cap \mathbb{Z}^n)$, $i = 1, 2$ *is closed. If* $L = \text{lin.space}(K_1 \cap K_2)$ *is generated by integer points, then* $\text{conv}\left[(K_1 \cap K_2) \cap \mathbb{Z}^n\right]$ *is closed.*

The proof of Theorem 3 uses as its building block a charaterization of closedness of integer hulls of general closed convex sets from [2]. A proof of this result is presented in Section 5. We obtain the following straightforward corollary of Theorem 3.

Corollary 1. *Consider the sets* $\mathcal{P}_i := \{x \in \mathbb{R}^n \mid \ A_i x - b_i \in \mathbf{L}^{m_i}\}$, *where for all* $i = 1, \ldots, q$, *we have* $A_i \in \mathbb{Q}^{m_i \times n}$, $b_i \in \mathbb{Q}^{m_i}$, *and* $\mathbf{L}^{m_i} \subseteq \mathbb{R}^{m_i}$ *is the Lorentz cone in* $\mathbb{R}^{m_i}$. *If the integer hull of* $\mathcal{P}_i$ *is closed for all* $i = 1, \ldots, q$, *then* $\text{conv}(\bigcap_{i=1}^q \mathcal{P}_i \cap \mathbb{Z}^n)$ *is closed.*

Notice that by the application of Theorem 2 for each $\mathcal{P}_i$, the sufficient condition of Corollary 1 can be verified in polynomial-time in the size of the input data. We finally note that Theorem 3 does not hold for the mixed-integer case as illustrated in the next example.

Example 1. Let $K_1 = \{(x, y) \in \mathbb{R}^2_+ \times \mathbb{R}_+ \,|\, y \geq x_2 - \sqrt{2}x_1\}$ and $K_2 = \{(x, y) \in \mathbb{R}^2_+ \times \mathbb{R}_+ \,|\, y \geq \sqrt{2}x_1 - x_2\}$. It is straightforward to check that $\text{conv}(K_1 \cap (\mathbb{Z}^2 \times \mathbb{R})) = K_1$ and that $\text{conv}(K_2 \cap (\mathbb{Z}^2 \times \mathbb{R})) = K_2$. Thus, the integer hulls of K_1 and K_2 are closed. However, we will verify next that $\text{conv}((K_1 \cap K_2) \cap (\mathbb{Z}^2 \times \mathbb{R}))$ is not closed. Denote $X = \{(x, y) \in \mathbb{R}^2_+ \times \mathbb{R}_+ \,|\, y = 0\}$. Let $r := \{\lambda(1, \sqrt{2}, 0) \,|\, \lambda \geq 0\} = K_1 \cap K_2 \cap X$. Thus, r is a ray with irrational slope contained in X. By the application of Dirichlet Approximation Theorem, we can verify that there are mixed-integer points $(x, y) \in \mathbb{Z}^2_+ \times \mathbb{R}_+$ in $K_1 \cap K_2$ that are arbitrarily close to the ray r. This implies that r belongs to the closure of $\text{conv}((K_1 \cap K_2) \cap (\mathbb{Z}^2 \times \mathbb{R}))$. On the other hand, since r is a face of $K_1 \cap K_2$ and $(0, 0, 0)$ is the only mixed-integer point in this face, we obtain that $r \cap \text{conv}((K_1 \cap K_2) \cap (\mathbb{Z}^2 \times \mathbb{R})) = \{(0, 0, 0)\}$. Therefore, we conclude that $\text{conv}((K_1 \cap K_2) \cap (\mathbb{Z}^2 \times \mathbb{R}))$ is not a closed set. □

We note here that Example 1 does not exclude the possibility of a result of the form of Corollary 1 for the mixed-integer case when each of the simple second order conic sets are defined using rational data. We have not been able to resolve this question.

3 Sketch of Proof of Theorem 1

Definition 1 (Mixed-integer lattice [3]). *Let* $A = [a_1|\cdots|a_{n_1}] \in \mathbb{R}^{m \times n_1}$ *and* $B = [b_1|\cdots|b_{n_2}] \in \mathbb{R}^{m \times n_2}$, *where* $\{a_1, \ldots, a_{n_1}, b_1, \ldots, b_{n_2}\}$ *is a linearly independent set of* $\mathbb{R}^m$. *Then*

$$\mathcal{L} = \{x \in \mathbb{R}^m \,|\, x = Az + By, \ z \in \mathbb{Z}^{n_1}, y \in \mathbb{R}^{n_2}\}$$

is said to be the mixed-integer lattice generated by A and B.

We note here that in the case $A \in \mathbb{Q}^{m\times n_1}$, $B \in \mathbb{Q}^{m\times n_2}$ it can be proved that a set $\mathcal{L}$ defined as above is a mixed-integer lattice, even when the set $\{a_1, \ldots, a_{n_1}, b_1, \ldots, b_{n_2}\}$ is not linearly independent.

Proof Outline:

1. **Simplifying the set $\mathcal{P} := \{(x, y) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \,|\, Ax + By - b \in \mathbf{L}^m\}$.** To simplify the analysis, we apply the affine map $T : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}^m$ defined as $T(x, y) = Ax + By - b$ to the set $(\mathcal{P} \cap (\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}))$. The image of $(\mathcal{P} \cap (\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}))$ under the map T is the set $((\mathbf{L}^m \cap V) \cap (\mathcal{L} - b))$ where $V := \{Ax + By - b \,|\, (x, y) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}\}$ and $\mathcal{L} := \{Ax + By \,|\, (x, y) \in \mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}\}$ is a mixed-integer lattice since A and B are rational matrices. Thus, we obtain the 'simple' set $\mathbf{L}^m \cap V$ in place of $\mathcal{P}$, at the cost of a 'complicated' translated mixed-integer lattice $\mathcal{L} - b$ in place of the mixed-integer lattice $\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}$. The closedness property is usually not invariant under affine transformations. However, we verify the following result:

   Theorem 4. *Let $K \subseteq \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$ be a closed convex set where $n_1, n_2 \in \mathbb{N}$. Let $G : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}^m$ defined as $G(x, y) := Ex + Fy - g$ be an affine map where $E \in \mathbb{R}^{m\times n_1}$, $F \in \mathbb{R}^{m\times n_2}$ and $g \in \mathbb{R}^m$. Assume that E, F satisfy the following:*

   (a) $\text{Kernel}([E\ F]) \subseteq \text{linspace}(K)$,
   (b) $\text{Kernel}([E\ F])$ *is generated by points in the lattice* $\mathbb{Z}^{n_1} \times \mathbb{Z}^{n_2}$.
   Then

   $$\text{conv}(K \cap (\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2})) \text{ is closed} \Leftrightarrow \text{conv}\,[G(K) \cap G(\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2})] \text{ is closed.}$$

   As a consequence of Theorem 4 we obtain that

   $$\text{conv}\,(\mathcal{P} \cap (\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2})) \text{ is closed} \Leftrightarrow \text{conv}\,[(\mathbf{L}^m \cap V) \cap (\mathcal{L} - b)] \text{ is closed.} \quad (2)$$

2. **Case Analysis.** Next we analyze the set $(\mathbf{L}^m \cap V)$. Observe that since $\mathbf{L}^m$ is a cone and V is an affine set, there are two natural cases (see Figure 1):

   (a) **Case 1: $\mathbf{L}^m \cap V$ is strictly convex set.** If $0 \notin V$, then $(\mathbf{L}^m \cap V)$ is a strictly convex set. We verify the following result.

   Theorem 5. *Let $K \subseteq \mathbb{R}^n$ be a closed strictly convex set, $t \in \mathbb{R}^n$ and $\mathcal{L}$ a mixed-integer lattice. Then* $\text{conv}(K \cap [\mathcal{L} + t])$ *is closed.*

   Theorem 5 is a generalization of a result about integer hulls of strictly convex sets from [2]. As a consequence of Theorem 5 and (2) we obtain that $\text{conv}(\mathcal{P} \cap (\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}))$ is always closed in this case. Observe that this is case (1.) in Theorem 1 when $0 \notin V$.

   (b) **Case 2: $\mathbf{L}^m \cap V$ is a cone.** If $0 \in V$, then $(\mathbf{L}^m \cap V)$ is a closed pointed convex cone. We have two subcases.

   Subcase 1: $b \notin \mathcal{L}$. In this case, $\mathcal{L} - b \neq \mathcal{L}$. Moreover, $\mathcal{L} - b$ is not a mixed-integer lattice. We need the following property.

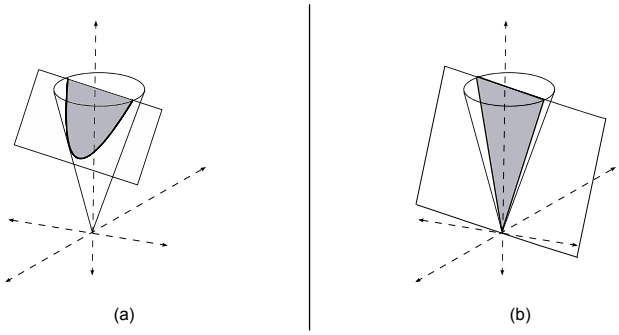


Fig. 1. Different cases for $\mathbf{L}^m \cap V$: (a) Strictly convex set (b) Pointed closed convex cone.

Lemma 1. *Let* $\mathbf{L}^m$ *be the Lorentz cone in* $\mathbb{R}^m$*. Let* $\mathcal{L} = \{x \in \mathbb{R}^m \,|\, x = Az + By,\ z \in \mathbb{Z}^{p_1}, y \in \mathbb{R}^{p_2}\}$ *be a mixed-integer lattice, where* A, B *are rational matrices. Denote* $V = \text{aff}(\mathcal{L})$*. Let* $b \in (V \cap \mathbb{Q}^m) \setminus \mathcal{L}$*. Then* $\text{conv}((\mathbf{L}^m \cap V) \cap (\mathcal{L} - b))$ *is closed.*

This result is a consequence of some properties on the closedness of integer hulls of general closed convex sets from [2]. We can apply Lemma 1 to verify that $\text{conv}\,[(\mathbf{L}^m \cap V) \cap (\mathcal{L} - b)]$ is a closed set in this case. Notice this is case (1.) in Theorem 1 when $0 \in V$. In particular this completes the examination of (1.) in Theorem 1.

Subcase 2: $b \in \mathcal{L}$**.** We begin the analysis of this case by verifying the following result.

Theorem 6. *Let* K *be a closed convex pointed cone in* $\mathbb{R}^n$ *and let* $\mathcal{L}$ *be a mixed-integer lattice. Then* $\overline{\text{conv}}(K \cap \mathcal{L}) = K \cap W$*, where* $W = \text{aff}(K \cap \mathcal{L})$*. In particular,* $\text{conv}(K \cap \mathcal{L})$ *is closed if and only if every extreme ray of* $K \cap W$ *can be scaled by a non-zero scalar to belong to* $\mathcal{L}$*.*

Theorem 6 is a generalization of a result about integer hulls of cones from [2]. As a consequence of Theorem 6, verifying closedness is equivalent to verifying whether the extreme rays of $\mathbf{L}^m \cap V$ can be scaled by a non-zero number to belong to $\mathcal{L}$.

When $\dim(\mathbf{L}^m \cap V) \leq 1$, it is straighforward to verify that this is always the case. This is case (2.) in Theorem 1.

For analyzing the case where $\dim(\mathbf{L}^m \cap V) > 1$ we need the following additional result.

Lemma 2. *Assume that* $0 \in \mathbf{L}^m \cap V$ *and that* $[A\ B] \in \mathbb{Q}^{m \times n}$*. Let* $\mathcal{L} = \{x \in \mathbb{R}^m \,|\, x = Az + By,\ z \in \mathbb{Z}^{n_1}, y \in \mathbb{R}^{n_2}\}$*. Then*

i. Assume $\dim(\mathbf{L}^m \cap V) = 2$*. If* $\dim(\langle B \rangle) \geq \dim(V) - 1$*, then every extreme ray of* $\mathbf{L}^m \cap V$ *can be scaled by a non-zero scalar to belong to* $\mathcal{L}$*.*

ii. Assume $\dim(\mathbf{L}^m \cap V) \geq 3$. *Then* $\dim(\langle B \rangle) \geq \dim(V) - 1$ *if and only if every extreme ray of* $\mathbf{L}^m \cap V$ *can be scaled by a non-zero scalar to belong to* $\mathcal{L}$.

The proof of Lemma 2 is based on the cardinality of the set of extreme rays in different dimensions (countable or not) and other geometric properties of the cone $\mathbf{L}^m \cap V$.

Lemma 2 is essentially stating that when $\dim(\mathbf{L}^m \cap V) \geq 3$ in order for every extreme ray to be scalable to belong to the mixed-integer lattice $\mathcal{L}$, there should be "sufficient" number of continuous components in the mixed-integer lattice $\mathcal{L}$. See Figure 2 for an illustration. Therefore we obtain that if $\dim(\mathbf{L}^m \cap V) \geq 3$, then $\text{conv}\,(\mathcal{P} \cap (\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}))$ is closed if and only if $\dim(\langle B \rangle) \geq \dim(V) - 1$. Moreover if $\dim(\mathbf{L}^m \cap V) = 2$ and $\dim(\langle B \rangle) \geq 1$, then $\text{conv}\,(\mathcal{P} \cap (\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}))$ is also closed. Together, this constitutes case (4.) in Theorem 1.

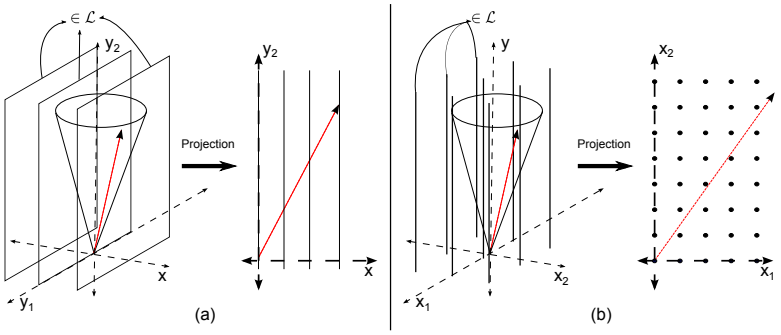


Fig. 2. $V = \mathbb{R}^3$. Extreme rays of $\mathbf{L}^m \cap V$: (a) All scalable to belong to $\mathcal{L} = \mathbb{Z} \times \mathbb{R}^2$ (b) Not all scalable to belong to $\mathcal{L} = \mathbb{Z}^2 \times \mathbb{R}$.

The only case that remains is where $\dim(\mathbf{L}^m \cap V) = 2$ and $\dim(\langle B \rangle) \leq 0$. In this case, we need to explicitly check whether the two extreme rays of $\mathbf{L}^m \cap V$ can be scaled by a non-zero scalar to belong to the lattice $\mathcal{L}$. This is case (3.) in Theorem 1.

4 Algorithm for Checking Closedness of Simple Second Order Conic Mixed-Integer Sets

In this section we prove Theorem 2, that is, we show that the closedness of $\text{conv}(\mathcal{P} \cap (\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}))$ can be checked in polynomial-time. We prove this result by showing that the conditions of Theorem 1 can be checked in polynomial-time with respect to the size of the data. Throughtout this section $V = \{Ax + By - b \,|\, (x, y) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}\}$. For a set $X \subseteq \mathbb{R}^n$ we denote $\text{cone}(X) = \{\sum_{i=1}^{n} \lambda_i x_i \,|\, \lambda_i \geq 0,\ x_i \in X, \forall\, i\}$. In Section 4.1 we present all the required results for testing cases (1.), (2.), and (4.) in polynomial-time. Checking case (3.) in polynomial-time is more involved, and is presented in Section 4.2.

4.1 Preliminary Results for Cases (1.), (2.) and (4.)

Let Proj_V denote the orthogonal projection onto the linear subspace V.

Lemma 3. *Let $a \in \mathbb{Q}^m$ be a vector of polynomial size with respect to the size of A, B, b. Then $\text{Proj}_V(a)$ can be computed in polynomial-time with respect to the size of A, B, b.*

Lemma 4. *Let $b \in \mathbb{Q}^m$ and $\mathcal{L} = \{Ax + By \,|\, (x, y) \in \mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}\}$ be a mixed-integer lattice, where $A \in \mathbb{Q}^{m \times n_1}$ and $B \in \mathbb{Q}^{m \times n_2}$. Then the condition $b \in \mathcal{L}$ can be checked in polynomial-time with respect to the size of A, B, b.*

The following lemma shows how to check if $\text{int}(\mathbf{L}^m) \cap V \neq \emptyset$ or not[1].

Lemma 5. *Let V be a linear subspace. Then*

1. $\dim(\mathbf{L}^m \cap V) \leq 1$ *if and only if* $(\text{int}(\mathbf{L}^m) \cap V = \emptyset$ *or* $\dim(V) \leq 1)$.
2. *Lets denote $a := (0, 1) \in \mathbb{R}^{m-1} \times \mathbb{R}$. Then*

$$\text{int}(\mathbf{L}^m) \cap V \neq \emptyset \text{ if and only if } \text{Proj}_V(a) \in \text{int}(\mathbf{L}^m).$$

Denote $\mathbf{S}^m := \{(w, z) \in \mathbb{R}^{m-1} \times \mathbb{R} \,|\, \|w\| = 1, z = 1\}$. We also need some additional properties of the Lorentz cone.

Lemma 6. *Let $\mathbf{L}^m \subseteq \mathbb{R}^m$ be the Lorentz cone and $W \subseteq \mathbb{R}^m$ an affine subspace such that $\dim(\mathbf{L}^m \cap W) \geq 2$, then*

1. $\text{int}(\mathbf{L}^m) \cap W \neq \emptyset$. *Consequently,* $\text{rel.int}(\mathbf{L}^m \cap W) = \text{int}(\mathbf{L}^m) \cap W$ *and* $\dim(\mathbf{L}^m \cap W) = \dim(W)$.
2. *If $0 \in W$, then we have $\mathbf{L}^m \cap W = \text{cone}(\mathbf{S}^m \cap W)$. And r is an extreme ray of $\mathbf{L}^m \cap W$ if and only if r can be scaled to belong to $\mathbf{S}^m \cap W$.*

Lemma 7

1. *The condition $\dim(\mathbf{L}^m \cap V) \leq 1$ can be checked in polynomial-time with respect to the size of A, B, b.*
2. *If $\dim(\mathbf{L}^m \cap V) \geq 2$, then $\dim(\mathbf{L}^m \cap V)$ can be computed in polynomial-time with respect to the size of A, B, b.*

Proof. Observe that $\dim(V) = \dim(\langle [A\ B] \rangle)$ and thus, since A, B are rational matrices, it can be computed in polynomial-time with respect to the size of A, B, b by the Gaussian algorithm [4].

1. By (1.) of Lemma 5 it suffices to check $\dim(V) \leq 1$ or $\text{int}(\mathbf{L}^m) \cap V = \emptyset$. Since $\dim(V)$ can be computed in polynomial-time, we can check whether $\dim(V) \leq 1$ or not in polynomial-time with respect to the size of A, B, b. Now, to verify $\text{int}(\mathbf{L}^m) \cap V = \emptyset$, we use (2.) of Lemma 5. By (2.) of Lemma 5, to verify whether $\text{int}(\mathbf{L}^m) \cap V = \emptyset$ or not, we need to check if $\text{Proj}_V(a) = (u, u_m) \notin \text{int}(\mathbf{L}^m)$ or not. Thus, we only need to compute $\|u\|^2$, and compare it with u_m^2. By Lemma 3 the size of (u, u_m) is polynomial in the size$(\mathcal{P})$, therefore we obtain that this comparison also can be done in polynomial-time with respect to the size$(\mathcal{P})$.

[1] We are grateful to Arkadi Nemirovski for a preliminary version of this idea.

2. Since $\dim(\mathbf{L}^m \cap V) \geq 2$, then by (1.) of Lemma 6 we obtain $\dim(\mathbf{L}^m \cap V) = \dim(V)$. By previous claim, this can be done in polynomial-time. □

4.2 Preliminary Results for Case (3.)

In this section we assume $V := \{Ax \,|\, x \in \mathbb{R}^{n_1}\}$, $\mathcal{L} := \{Ax \,|\, x \in \mathbb{Z}^{n_1}\}$, $b \in \mathcal{L}$ and $\dim(\mathbf{L}^m \cap V) = 2$. Since A is a rational matrix, a basis for $\mathcal{L}$ can be found in polynomial-time with respect to size(A) by computing the Hermite normal form of A. Let the vectors $(A_1, a_1), (A_2, a_2) \in \mathbb{Q}^{m-1} \times \mathbb{Q}$ form a basis of $\mathcal{L}$. We denote $S_V := \mathbf{S}^m \cap V$.

The following lemma characterizes the extreme rays of $\mathbf{L}^m \cap V$ in terms of the basis of the lattice $\mathcal{L}$.

Lemma 8. *Let $V := \{Ax \,|\, x \in \mathbb{R}^n\}$. Assume that $\dim(\mathbf{L}^m \cap V) = 2$. Then*

1. *a_1 and a_2 cannot be both zero.*
2. *S_V is given by the solutions of the following system of two equations:*

$$\begin{aligned} \|\alpha_1 A_1 + \alpha_2 A_2\|^2 &= 1 \\ \alpha_1 a_1 + \alpha_2 a_2 &= 1. \end{aligned} \tag{3}$$

3. *Let (α_1, α_2) and (α'_1, α'_2) be the solutions of the system of equations (3). Then the two extreme rays of $\mathbf{L}^m \cap V$ can be written as*

$$\alpha_1 \begin{pmatrix} A_1 \\ a_1 \end{pmatrix} + \alpha_2 \begin{pmatrix} A_2 \\ a_2 \end{pmatrix} \quad \text{and} \quad \alpha'_1 \begin{pmatrix} A_1 \\ a_1 \end{pmatrix} + \alpha'_2 \begin{pmatrix} A_2 \\ a_2 \end{pmatrix}.$$

Proof. Observe that

$$\{(A_1, a_1), (A_2, a_2)\} \text{ is a basis of } V. \tag{4}$$

1. Since $\dim(\mathbf{L}^m \cap V) = 2$, by (1.) of Lemma 6 we obtain that $\text{int}(\mathbf{L}^m) \cap V \neq \emptyset$. Thus, since $\text{int}(\mathbf{L}^m) \cap \mathbb{R}^{m-1} \times \{0\} = \emptyset$, we have that $\mathbf{L}^m \cap V \subsetneq \mathbb{R}^{m-1} \times \{0\}$. In particular, $V \subsetneq \mathbb{R}^{m-1} \times \{0\}$. Therefore, by (4), we conclude that a_1 and a_2 cannot be both zero.
2. Since $S_V = \mathbf{S}^m \cap V \subseteq V$ and by (4), we obtain that

$$S_V = \left\{ \alpha_1 \begin{pmatrix} A_1 \\ a_1 \end{pmatrix} + \alpha_2 \begin{pmatrix} A_2 \\ a_2 \end{pmatrix} \mid \|\alpha_1 A_1 + \alpha_2 A_2\|^2 = 1;\ \alpha_1 a_1 + \alpha_2 a_2 = 1 \right\}.$$

3. By (2.) of Lemma 6 we have $\mathbf{L}^m \cap V = \text{cone}(S_V)$ and that r is an extreme ray of $\mathbf{L}^m \cap V$ if and only if r can be scaled to belong to S_V. Therefore, the extreme rays of $\mathbf{L}^m \cap V$ can be found using equation (3). □

Notice that by (1.) of Lemma 8 we have that either $a_1 \neq 0$ or $a_2 \neq 0$. Thus, we may assume without loss of generality throughout the rest of this section that $a_2 \neq 0$.

Lemma 9. *Let (α_1, α_2) and (α'_1, α'_2) be the solutions of the system of equations (3). Then the extreme rays of $\mathbf{L}^m \cap V$ can be scaled to belong to $\mathcal{L}$ if and only if $\alpha_1, \alpha'_1 \in \mathbb{Q}$.*

Proof

($\Rightarrow$) We use (3.) of Lemma 8 to characterize the extreme rays of $\mathbf{L}^m \cap V$ in terms of (α_1, α_2) and (α'_1, α'_2). First we consider the extreme ray associated to (α_1, α_2). There exists $\lambda > 0$ and $\gamma \in \mathbb{Q}^m$ such that

$$\lambda \left[\alpha_1 \begin{pmatrix} A_1 \\ a_1 \end{pmatrix} + \alpha_2 \begin{pmatrix} A_2 \\ a_2 \end{pmatrix} \right] = \gamma. \tag{5}$$

Since $\alpha_1 a_1 + \alpha_2 a_2 = 1$, by considering the last constraint in (5) we obtain that $\lambda \in \mathbb{Q} \setminus \{0\}$. Thus, we obtain that (α_1, α_2) is the unique solution to a system of linear equations with rational data and thus α_1, α_2 are rational. Similarly α'_1, α'_2 are also rational.

($\Leftarrow$) Observe first that since (α_1, α_2) and (α'_1, α'_2) are the solutions to (3), we obtain that

$$\alpha_1 a_1 + \alpha_2 a_2 = 1 \quad \text{and} \quad \alpha'_1 a_1 + \alpha'_2 a_2 = 1.$$

If $\alpha_1 = 0$, then α_2 is rational. If $\alpha_1 \neq 0$, then α_2 is rational if and only if α_2 is rational, since a_1 and a_2 are rational. Therefore in general α_1 is rational if and only if α_2 is rational. Thus by hypothesis we obtain that $(\alpha_1, \alpha_2), (\alpha'_1, \alpha'_2) \in \mathbb{Q}^2$. Hence, there exists $\lambda, \lambda' > 0$ such that $\lambda(\alpha_1, \alpha_2), \lambda'(\alpha'_1, \alpha'_2) \in \mathbb{Z}^2$. Therefore, by (3.) of Lemma 8 we obtain that the extreme rays of $\mathbf{L}^m \cap V$ can be scaled to belong to $\mathcal{L}$. □

Lemma 10. *If* $\dim(\mathbf{L}^m \cap V) = 2$*, then whether the two extreme rays of the cone* $\mathbf{L}^m \cap V$ *can be scaled to belong to* $\mathcal{L}$ *can be checked in polynomial-time.*

Proof. Let (α_1, α_2) and (α'_1, α'_2) be the solutions of the system of equations (3). Since $a_2 \neq 0$, we can write $\alpha_2 = \frac{1-\alpha_1 a_1}{a_2}$ and $\alpha'_2 = \frac{1-\alpha'_1 a_1}{a_2}$. Therefore, by Lemma 9, in order to check whether the extreme rays of the cone $\mathbf{L}^m \cap V$ can be scaled to belong to $\mathcal{L}$, we only need to verify if the solutions α_1, α'_1 to the quadratic equation

$$\left\| \alpha A_1 + \frac{1 - \alpha a_1}{a_2} A_2 \right\|^2 = 1 \tag{6}$$

belong to $\mathbb{Q}$. We will show that this can be done in polynomial-time with respect to the data $A_1, A_2 \in \mathbb{Q}^{m-1}$, $a_1, a_2 \in \mathbb{Q}$. Since the size of the product of all the denominators of the components of the vectors and the scalars appearing in (6) is polynomial with respect to the size of the original data, without loss of generality we obtain the following equivalent equation

$$\|\alpha p + q\|^2 = r, \tag{7}$$

where $p, q \in \mathbb{Z}^{m-1}$, $r \in \mathbb{Z}$ and size(p), size(q) and size(r) are polynomial with respect to the size of the original data. Notice that equation (7) can be written as

$$\left(\sum_{i=1}^{m-1} p_i^2 \right) \alpha^2 + \left(\sum_{i=1}^{m-1} 2 p_i q_i \right) \alpha + \sum_{i=1}^{m-1} q_i - r = 0. \tag{8}$$

Let $c_1 = \sum_{i=1}^{m-1} p_i^2$, $c_2 = \sum_{i=1}^{m-1} 2p_iq_i$ and $c_3 = \sum_{i=1}^{m-1} q_i - r$. Observe that size($c_1$), size($c_2$) and size($c_3$) are polynomial with respect to the size of the original data. Using this notation, and by solving the quadratic equation (8) we obtain

$$\alpha_1 = \frac{-c_2 + \sqrt{c_2^2 - 4c_1c_3}}{2c_1} \quad \text{and} \quad \alpha_1' = \frac{-c_2 - \sqrt{c_2^2 - 4c_1c_3}}{2c_1}.$$

Therefore, $\alpha, \alpha' \in \mathbb{Q}$ if and only if $c_2^2 - 4c_1c_3$ is a perfect square. Since the latter can be checked in polynomial-time with respect to the size of c_1, c_2, c_3 (see, for example, Section 1.7 of [5]), we conclude that we can determine if $\alpha, \alpha' \in \mathbb{Q}$ in polynomial-time with respect to size of the original data. □

4.3 Proof of Theorem 2

Proof (of Theorem 2). The following 'algorithm' checks all the conditions of Theorem 1 in polynomial-time: First, by Lemma 4 we can verify whether condition (1.) of Theorem 1 is satisfied. If not, we can verify in polynomial-time whether $\dim(\mathbf{L}^m \cap V) \leq 1$ (by (1.) of Lemma 7). If condition (2.) of Theorem 1 is satisfied, stop. Otherwise, by (2.) of Lemma 7, we can compute $\dim(\mathbf{L}^m \cap V)$ in polynomial-time. If $\dim(\mathbf{L}^m \cap V) = 2$ and $\dim(\langle B \rangle) \leq 0$, then, by Lemma 10 we can verify in polynomial-time whether the two extreme rays of the cone $\mathbf{L}^m \cap V$ can be scaled to belong to $\mathcal{L}$. If condition (3.) of Theorem 1 is satisfied, stop. If not, then we have $\dim(\mathbf{L}^m \cap V) = \dim(V)$ (Lemma 7). Since B is a rational matrix, we obtain that $\dim(\langle B \rangle)$ can be computed in polynomial-time by the Gaussian algorithm [4]. Therefore, we conclude that we can check condition (4.) of Theorem 1 in polynomial-time. □

5 Invariance of Closedness of Integer Hulls under Finite Intersection in the Pure Integer Case

The proof of Theorem 3 relies on a characterization of closedness of integer hulls that we proved in a recent paper [2]. In order to present this characterization we begin with a definition.

Definition 2 ($u(K)$). *Given a convex set $K \subseteq \mathbb{R}^n$ and $u \in K \cap \mathbb{Z}^n$, we define* $u(K) = \{d \in \mathbb{R}^n \mid u + \lambda d \in \text{conv}(K \cap \mathbb{Z}^n)\ \forall \lambda \geq 0\}$.

The following result is modified from [2].

Theorem 7. *If* $\text{conv}(K \cap \mathbb{Z}^n)$ *is closed, then* $u(K)$ *is identical for all* $u \in K \cap \mathbb{Z}^n$. *Conversely, if* $u(K)$ *is identical for all* $u \in K \cap \mathbb{Z}^n$ *and* K *contains no lines, then* $\text{conv}(K \cap \mathbb{Z}^n)$ *is closed.*

Next we present two results regarding operations that preserve closedness of integer hulls that are used to proof Theorem 3. The following lemma is a straightforward result that we present without proof.

Lemma 11. *Let U be a $n \times n$ unimodular matrix and let $K \subseteq \mathbb{R}^n$ be a closed convex set. Then* $\text{conv}(K \cap \mathbb{Z}^n)$ *is closed iff* $\text{conv}((UK) \cap \mathbb{Z}^n)$ *is closed.*

Denote by $\text{Proj}_{L^\perp}(\cdot)$ the orthogonal projection onto $L^\perp$.

Proposition 1. *Let $\mathcal{L}' = \{x \in \mathbb{R}^n \,|\, x = Az + By,\ z \in \mathbb{Z}^{p_1}, y \in \mathbb{R}^{p_2}\}$ be a mixed-integer lattice. Let $K \subseteq \mathbb{R}^n$ be a closed convex set. Denote $L = \text{linspace}(K \cap \text{aff}(K \cap \mathcal{L}'))$. If L is generated by points in the lattice $\{x \in \mathbb{R}^m \,|\, x = Az + By,\ z \in \mathbb{Z}^{p_1}, y \in \mathbb{Z}^{p_2}\}$, then*

$$\text{conv}(K \cap \mathcal{L}') = \text{conv}((K \cap L^\perp) \cap \text{Proj}_{L^\perp}(\mathcal{L}')) + L.$$

In particular, $\text{conv}(K \cap \mathcal{L}')$ *is closed* $\Leftrightarrow$ $\text{conv}((K \cap L^\perp) \cap \text{Proj}_{L^\perp}(\mathcal{L}'))$ *is closed.*

The next result is from [2].

Lemma 12. *Let $K \subseteq \mathbb{R}^n$ be a closed convex set, let $u \in K \cap (\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2})$ and let $d = \{u + \lambda r \,|\, \lambda > 0\} \subseteq \text{int}(K)$. Then $\{u\} \cup d \subseteq \text{conv}(K \cap (\mathbb{Z}^{n_1} \times \mathbb{R}^{n_2}))$.*

We obtain the next result as a consequence of Lemma 11 and Lemma 12.

Corollary 2. *Let $K \subseteq \mathbb{R}^n$ be a closed convex set such that* $\text{aff}(K)$ *is a rational affine set. Let $u \in K \cap \mathbb{Z}^n$. If $\{u + \lambda d | \lambda > 0\} \subseteq \text{rel.int}(K)$, then $\{u + d\lambda | \lambda \geq 0\} \subseteq \text{conv}(K \cap \mathbb{Z}^n)$.*

Theorem 3. *Let $K_i \subseteq \mathbb{R}^n$, $i = 1, 2$. Assume* $\text{conv}(K_i \cap \mathbb{Z}^n)$, $i = 1, 2$ *is closed. If $L = \text{lin.space}(K_1 \cap K_2)$ is generated by integer points, then* $\text{conv}\,[(K_1 \cap K_2) \cap \mathbb{Z}^n]$ *is closed.*

Proof. If $(K_1 \cap K_2) \cap \mathbb{Z}^n = \emptyset$, then we are done. Assume $(K_1 \cap K_2) \cap \mathbb{Z}^n \neq \emptyset$.

We may assume that $K_1 = \text{conv}(K_1 \cap \mathbb{Z}^n)$ and $K_2 = \text{conv}(K_2 \cap \mathbb{Z}^n)$. By Theorem 7 we know that $u(K_i) = U_i$ for all $u \in K_i \cap \mathbb{Z}^n$, $i = 1, 2$.

We have two cases:

Case 1: $L = \{0\}$, that is, $(K_1 \cap K_2)$ does not contain lines
By Theorem 7, to prove that $\text{conv}((K_1 \cap K_2) \cap \mathbb{Z}^n)$ is closed it is sufficient to show that for all $u \in (K_1 \cap K_2) \cap \mathbb{Z}^n$ we have $u(K_1 \cap K_2) = U_1 \cap U_2$.

We first verify $u(K_1 \cap K_2) \subseteq U_1 \cap U_2$. Since $\text{conv}\,[(K_1 \cap K_2) \cap \mathbb{Z}^n] \subseteq \text{conv}(K_1 \cap \mathbb{Z}^n) \cap \text{conv}(K_2 \cap \mathbb{Z}^n)$, we have $u(K_1 \cap K_2) \subseteq u(K_1) \cap u(K_2) = U_1 \cap U_2$.

Now we verify that $u(K_1 \cap K_2) \supseteq U_1 \cap U_2$. Let $u \in (K_1 \cap K_2) \cap \mathbb{Z}^n$ and let $d \in U_1 \cap U_2$. Since K_1 is a closed convex set, there exists a face F_1 of K_1 such that $u \in F_1$ and $\{u + \lambda d | \lambda > 0\} \subseteq \text{rel.int}(F_1)$. Similarly, let F_2 be the face of K_2 such that $u \in F_2$ and $\{u + \lambda d | \lambda > 0\} \subseteq \text{rel.int}(F_2)$. Let $Q = F_1 \cap F_2$. Observe that $\{u + \lambda d | \lambda > 0\} \subseteq \text{rel.int}(F_1) \cap \text{rel.int}(F_2)$, thus we have $\text{rel.int}(Q) = \text{rel.int}(F_1) \cap \text{rel.int}(F_2)$. Hence, by a standard result in convex analysis, we obtain that $\text{aff}(Q) = \text{aff}(F_1) \cap \text{aff}(F_2)$. Thus, since $\text{aff}(F_1)$ and $\text{aff}(F_2)$ are rational affine subspaces, we obtain that $\text{aff}(Q)$ is a rational affine subspace. Therefore, by Corollary 2, $\{u + \lambda d | \lambda \geq 0\} \subseteq \text{conv}(Q \cap \mathbb{Z}^n) \subseteq \text{conv}\,[(K_1 \cap K_2) \cap \mathbb{Z}^n]$ and so, $d \in u(K_1 \cap K_2)$.

Therefore, for all $u \in (K_1 \cap K_2) \cap \mathbb{Z}^n$, $u(K_1 \cap K_2) = u(K_1) \cap u(K_2) = U_1 \cap U_2$.

Case 2: $L \neq \{0\}$, that is, $(K_1 \cap K_2)$ contains lines
Since L is generated by integer points, by Hermite normal form algorithm, there exists an unimodular matrix U such that $UL = \mathbb{R}^p \times \{0\}^{n-p}$. Thus, by Lemma 11, we may assume that $L = \mathbb{R}^p \times \{0\}^{n-p}$. For $i = 1, 2$ let $K_i' \subseteq \mathbb{R}^{n-p}$ be the convex set such that $K_i \cap L^\perp = \{0\}^p \times K_i'$. Notice that by Proposition 1 we only need to show that $\text{conv}((K_1 \cap K_2 \cap L^\perp) \cap \text{Proj}_{L^\perp}(\mathbb{Z}^n)) = \text{conv}(\{0\}^p \times (K_1' \cap K_2' \cap \mathbb{Z}^{n-p}))$ is closed. This is equivalent to show that $\text{conv}(K_1' \cap K_2' \cap \mathbb{Z}^{n-p})$ is closed. Observe that for $i = 1, 2$ we have that $\text{conv}(K_i \cap \mathbb{Z}^n)$ is closed. Hence, by Proposition 1 we obtain that $\text{conv}((K_i \cap L^\perp) \cap \text{Proj}_{L^\perp}(\mathbb{Z}^n)) = \text{conv}(\{0\}^p \times (K_i' \cap \mathbb{Z}^{n-p}))$ is closed, $i = 1, 2$. Equivalently, $\text{conv}(K_i' \cap \mathbb{Z}^{n-p})$ is closed, $i = 1, 2$. Now, notice that the set $(K_1' \cap K_2')$ does not contain lines. Thus, by Case 1 applied to the sets K_1' and K_2', we obtain that $\text{conv}(K_1' \cap K_2' \cap \mathbb{Z}^{n-p})$ is closed, as desired. □

References

1. Meyer, R.R.: On the existence of optimal solutions of integer and mixed-integer programming problems. Mathematical Programming 7, 223–225 (1974)
2. Dey, S., Morán, R.D.: Some properties of convex hulls of integer points contained in general convex sets. Mathematical Programming, 1–20, doi:10.1007/s10107-012-0538-7
3. Bertsimas, D., Weismantel, R.: Optimization over integers, vol. 13. Dynamic Ideas (2005)
4. Edmonds, J.: Systems of distinct representatives and linear algebra. Journal of Research of the National Bureau of Standards (B) 71, 241–245 (1967)
5. Cohen, H.: A Course in Computational Algebraic Number Theory. Graduate Texts in Mathematics. Springer (1993)

The Complexity of Scheduling for p-Norms of Flow and Stretch

(Extended Abstract)

Benjamin Moseley[1], Kirk Pruhs[2,⋆], and Cliff Stein[3,⋆⋆]

[1] Toyota Technological Institute, Chicago IL, 60637, USA
moseley@ttic.edu
[2] Computer Science Department, University of Pittsburgh, Pittsburgh, PA 15260, USA
kirk@cs.pitt.edu
[3] Department of Industrial Engineering & Operations Research, Columbia University, Mudd 326, 500W 120th Street, New York, NY 10027, USA
cliff@ieor.columbia.edu

Abstract. We consider computing optimal k-norm preemptive schedules of jobs that arrive over time. In particular, we show that computing the optimal k-norm of flow schedule, $1 \mid r_j, pmtn \mid \sum_j (C_j - r_j)^k$ in standard 3-field scheduling notation, is strongly NP-hard for $k \in (0,1)$ and integers $k \in (1,\infty)$. Further we show that computing the optimal k-norm of stretch schedule, $1 \mid r_j, pmtn \mid \sum_j ((C_j - r_j)/p_j)^k$ in standard 3-field scheduling notation, is strongly NP-hard for $k \in (0,1)$ and integers $k \in \cup(1,\infty)$.

1 Introduction

In the ubiquitous client-server computing model, multiple clients issue requests over time, and a request specifies a job for the server to perform. When the requested jobs have widely varying processing times — as is the case for compute servers, database servers, web servers, etc. — the server system generally must allow (presumably long) jobs to be preempted for waiting (presumably smaller) jobs in order to provide a reasonable quality of service to the clients. The most commonly considered and most natural quality of service measure for a job j is the flow/waiting/response time, which is $C_j - r_j$ the duration of time between time r_j when the request is issued, and time C_j when the job is completed. Another commonly considered and natural quality of service measure for a job j is the stretch/slowdown, $(C_j - r_j)/p_j$, the flow time divided by the processing time requirement p_j of the job. The stretch of a job measures how much time the job took relative to how long the job would have taken on a dedicated server. Flow time is probably more appropriate when the client has little idea of the time required for this requested job, as might be the case for a non-expert database client. Stretch is probably more appropriate when the client has at least an approximate idea of the time required for the job, as would be the case when clients are requesting static content from

⋆ Supported in part by NSF grants CCF-0830558, CCF-1115575, CNS-1253218, and an IBM Faculty Award.
⋆⋆ Research partially supported by NSF grant CCF-0915681.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 278–289, 2013.

a web server (e.g. when requesting large video files clients will expect/tolerate a longer response than requesting small text files).

The server must have some scheduling policy to determine which requests to prioritize in the case that there are multiple outstanding requests. To measure the quality of service of the schedule produced by the server's scheduling policy, one needs to combine the quality of service measures of the individual requests. In the computer systems literature, the most commonly considered quality of service measure for a schedule is the 1-norm, or equivalently average or total, of the quality of service provided to the individual jobs. Despite the widespread use, one often sees the concern expressed that average flow is not the ideal quality of service measure in that an optimal average flow schedule may "unfairly starve" some longer jobs. Commonly what is desired is a quality of service measure that "balances" the competing priorities of optimizing average quality of service and maintaining fairness among jobs. The mathematically most natural way to achieve this balance would be to use the 2-norm (or more generally the k-norm for some small integer k).

The k-norms of flow time and stretch have been studied in the scheduling theory literature in a variety of settings: on a single machine [BP10b, BP10a], multiple machines [CGKK04, BT06, FM11, IM11, AGK12], in broadcast scheduling [EIM11, CIM09, GIK$^+$10], for parallel processors [EIM11, GIK$^+$10] and on speed scalable processors [GKP12]. The choice of k depends on the desired balance of average performance with fairness. For example, the 2-norm is used in the standard least-squares approach to linear regression, but the 3-norm is used within LaTeX to determine the best line breaks. Conceivably there are also situations in which one may want to choose $k < 1$, say when a client wants a job to be completed quickly, but if the job is not completed quickly, the client does not care so much about how long the job is delayed.

Directly Related Previous Results: In what is essentially folklore, the following is known for optimizing a norm of flow time offline with release dates and preemption:

- the optimal 1-norm schedule can be computed in polynomial time by the greedy algorithm Shortest Remaining Processing Time (SRPT), and
- the optimal schedule for the ∞-norm, of either flow and stretch, can be computed in polynomial time by combining a binary search over the maximum flow or stretch and the Earliest Deadline First (EDF) scheduling algorithm, which produces a deadline feasible schedule if one exists.

Surprisingly, despite the interest the in k-norms of flow time and stretch, the complexity of computing an optimal k-norm of flow schedule, for $k \neq 1$ or ∞, and the complexity of computing an optimal k-norm of stretch schedule, for any k, were all open.

1.1 Our Results

We show that for all integers $k \geq 2$, and for all $k \in (0, 1)$, the problem of finding a schedule that minimizes the k-norm of flow is strongly NP-hard. Similarly, we show that for all integer $k \geq 2$, and for all $k \in (0, 1)$, the problem of finding a schedule that minimizes the k-norm of stretch is strongly NP-hard. This rules out the existence

Table 1. A summary of results

Folklore Results		
	Flow	Stretch
$k = 1$	SRPT is optimal	Open
$k = \infty$	EDF and Binary Search	EDF and Binary Search
Our Results		
	Flow	Stretch
$k \in (0, 1)$ and integers $k \in (1, \infty)$	NP-hard	NP-hard

of a fully polynomial time approximation scheme (FPTAS) for these problems unless $P = NP$. See Table 1 for a summary.

The starting point for our NP-hardness proofs is the NP-hardness proof in [LLLRK82] for the problem of finding optimal weighted flow schedules, $1 \mid r_j, pmtn \mid \sum_j w_j(C_j - r_j)$ in the standard 3-field scheduling notation. In this problem, each job has a positive weight, and the quality of service measure is a weighted average of the flow of the individual jobs. The NP-hardness proof of weighted flow in [LLLRK82] is a reduction from 3-partition. For each element of size x in the 3-partition instance, there is a job of weight x and processing time x released at time 0 in the weighted flow instance. Further, in the weighted flow instance, there are intermittent streams of small jobs that partition the remaining time into open time intervals of length equal to the partition size in the 3-partition instance. [LLLRK82] shows that in this case, the best possible schedule 3-partitions the large jobs among the open time intervals. In some sense, the reduction in [LLLRK82] is fragile in that it critically relies on the equality of weights and execution times. [1]

In order to prove our new results, several additional ideas are needed. We define the age of a job to be the difference between the current time and the job's release date. For k-norms of flow, the age of a job to the $(k-1)$st power can be thought of as the job's weight at time t, in that the integral over time of this quantity is the quality of service measure for the job. Thus the "weight" of a job varies over time. Our main insight is that the reduction in [LLLRK82] can be extended for k-norms if it is modified so that the amount of time that a job is released before the first open time interval is proportional to the size of the corresponding 3-partition element. We then need to add a third class of jobs to make sure that the partition jobs do not run during the early part of the schedule. The time-varying nature of the "weight" of the jobs requires a more involved analysis, as we need to be able to bound powers of flow.

We note that it is easy to see that these NP-hardness easily generalize to more complicated settings, e.g. broadcast scheduling, speed scaling and parallel processors.

The rest of the paper is structured as follows. Some moderately related results are summarized in the next subsection. Section 2 gives some preliminary definitions. Section 3 gives the NP-hardness proof for 2-norm of flow. Section 4, briefly summarizes the remaining NP-hardness proofs, as there is insufficient space to give fuller proofs.

[1] We do not know for example if weighted flow is NP-hard or in P for instances where shorter jobs have larger weights. If this was in P, this would imply that average stretch is in P.

1.2 Other Related Results

Despite the lack of NP-completeness results, approximation algorithms and on-line algorithms have been developed for k-norms of flow and stretch. Off-line, polynomial-time approximation schemes are known for computing optimal 1-norm of stretch schedules [BMR04, CK02]. For k-norms of flow and stretch, and for weighted flow, polynomial-time $O(\log \log Pn)$-approximation is achievable [BP10a]. For on-line algorithms, in [BP10b] it is shown that several standard online scheduling algorithms, such as SRPT, are scalable ($(1+\epsilon)$-speed $O(1)$-competitive for any fixed constant $\epsilon > 0$) for k-norms of flow and stretch.

2 Preliminaries

In our scheduling instances, each job $i \in [n] = \{1, 2, \ldots, n\}$ has a positive rational processing time p_i and a rational arrival time r_i. (In the typical definition, arrival times are non-negative, but since our objective is flow time or stretch, allowing arrival times to be negative does not change the complexity of the problem.) A (preemptive) schedule is a function that maps some times t to a job i, released by t time, that is run at time t. A job i is completed at the first time C_i when it has been scheduled for p_i units of time. In the k-norm problem, the objective is to minimize $\sqrt[k]{\sum_{i\in[n]}(C_i - r_i)^k}$. It will be convenient to abuse terminology and use the term k-norm to refer to the objective $\sum_{i\in[m]}(C_i - r_i)^k$, which gives rise to the same optimal schedules as the other objective. For the rest of this paper, we will consider this objective. We define the increase of the k-norm for a job i during a time period $[b, e]$ with $r_i \leq e$ by $(e-r_i)^k - (b-\min(b, r_i))^k$. Similarly, the increase in the k-norm for a collection of jobs is the aggregate increase of the individual jobs.

An instance of the 3-Partition problem consists of a set $S = \{b_1, b_2, \ldots, b_{3m}\}$ of $3m$ positive integers, and a positive integer B, where B is polynomially bounded in m. In the classical definition of 3-Partition, b_i are restricted to be between $B/4$ and $B/2$. By adding some large number to each element of S, one can assume without loss of generality the tighter bound that $\frac{m}{3m+1/2}B \leq b_i \leq B/2$ for all i. The problem is to determine a partition of S into m subsets $P_1, P_2, \ldots, P_m$ such that for any i it is the case that $|P_i| = 3$ and $\sum_{b_j \in P_i} b_j = B$. The 3-Partition problem is strongly NP-complete [GJ79].

We will use the term *volume* of work to refer to an amount of work.

3 NP-Hardness for 2-Norm of Flow

In this section, we show that the problem of determining if there exists a schedule with the 2-norm of flow less than some specified value f is NP-hard by a reduction from the 3-partition problem. We start by describing the reduction, which uses parameters α, β, and ρ, and is illustrated in Figure 1:

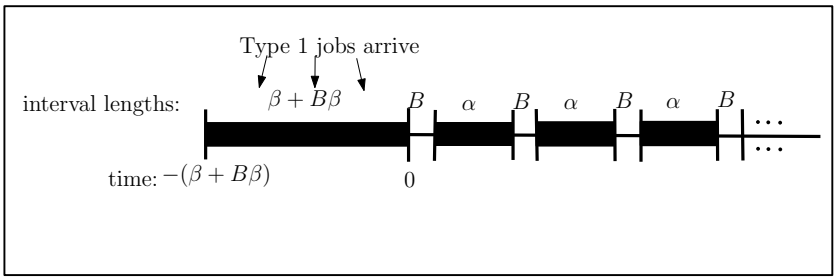


Fig. 1. The scheduling instance

The Reduction:

- **Type 1 jobs:** For each integer $b_i \in S$ from the 3-partition instance, we create a job of processing time b_i and arrival time $-(\beta + \beta b_i)$. Let T_1 denote the set of Type 1 jobs.
- **Type 2 jobs:** During the interval I_i, between time $s_i = iB + (i-1)\alpha$ and time $s_i + \alpha$, for $i \in [1, m-1]$, there is a job, with processing time ρ, released every ρ time steps.
- **Type 3 jobs:** During the interval $I_0 = [-(\beta + \beta B), 0]$ a job of processing time ρ is released every ρ time steps.

Intuitively, the type 2 and type 3 jobs are so short that they must be processed essentially as they are released. Thus we say that the times in I_i are *closed*, while other times are *open*. One can map a 3-partition to a *partition schedule* by scheduling the type 1 jobs corresponding the ith partition in the ith open time interval, and scheduling type 2 and type 3 jobs as they arrive. To complete the reduction, we set f to be an upper bound on the 2-norm of flow for a partition schedule (this is proved in Lemma 1):

$$f := f_{2,3} + f_o + \sum_{i=0}^{n} f_1(i)$$

where $f_{2,3} := \rho(\beta B + \beta + (m-1)\alpha)$, $f_o := 6m^2B(\beta B + \beta + (m-1)B + (m-1)\alpha) + B^2$, $f_1(i) := (3m - 3i)((s_i + \beta)\alpha + \alpha^2) + 2\beta\alpha(mB - iB)$, and $f_1(0) := \sum_{i \in T_1}(\beta + \beta b_i)^2$. Eventually we will need that $\max(m, B) \ll \alpha \ll \beta \ll \frac{1}{\rho} \ll poly(m, B)$. Foreshadowing slightly, more specifically we will need in the proof of Lemma 2 that $\frac{1}{4m\rho} > f$, and we will need in Lemma 5 and Lemma 6 that $\alpha\beta > f_o$. We need that the parameters are bounded by a polynomial in m and B so that the scheduling instance is of polynomial size. We shall see that it is sufficient to set $\alpha = m^2B^3$, $\beta = m^5B^4$, and $\rho = 1/(\beta m)^3$.

Lemma 1. *Let A be an arbitrary partition schedule. The contribution of the type 2 and type 3 jobs towards the 2-norm of flow for A is, at most $f_{2,3}$. In A the increase in the 2-norm of flow of the type 1 jobs during the I_i, $i \in [0, m-1]$, is at most $f_1(i)$. In A the increase in the 2-norm of the type 1 jobs during the open time intervals is at most f_o. Thus, the 2-norm of flow for A is at most f.*

Proof. We address these claims in order. The length of I_0 is $(\beta B + \beta)$, and the length of I_i, $i \in [1, m-1]$ is α. Thus there are $(\beta B + \beta)/\rho + (m-1)\alpha/\rho$ type 2 and type 3 jobs in the instance. Each of these jobs contributes ρ^2 towards the 2-norm of flow. Thus the 2-norm of flow for the type 2 and type 3 jobs in A is $f_{2,3} = \rho(\beta B + \beta + (m-1)\alpha)$.

The increase in the 2-norm of flow in A of the type 1 jobs during the I_i, $i \geq 1$, is at most:

$$\begin{aligned}
&\sum_{l \in U_i} \left((s_i + \beta + \alpha + \beta b_l)^2 - (s_i + \beta + \beta b_l)^2\right) \\
&= \sum_{l \in U_i} \left(2(s_i + \beta)\alpha + \alpha^2 + 2\beta b_l \alpha\right) \\
&= |U_i|((s_i + \beta)\alpha + \alpha^2) + 2\beta\alpha \sum_{l \in U_i} b_l \\
&\geq (3m - 3i)((s_i + \beta)\alpha + \alpha^2) + 2\beta\alpha \sum_{l \in U_i} b_l \quad \text{[Since } |U_i| \geq 3m - 3i\text{]} \\
&\geq (3m - 3i)((s_i + \beta)\alpha + \alpha^2) + 2\beta\alpha(mB - iB) \quad \text{[Since } \textstyle\sum_{l \in U_i} b_l \geq mB - iB\text{]} \\
&= f_1(i)
\end{aligned}$$

The increase in the 2-norm of flow of the type 1 jobs during I_0 is $f_1(0) = \sum_{i \in T_1} (\beta + \beta b_i)^2$ since each type 1 job waits $\beta + \beta b_i$ time steps until the end of I_0 by construction.

The maximum increase in the 2-norm of flow for a type 1 job during an open interval is $(\beta B + \beta + mB + (m-1)\alpha)^2 - (\beta B + \beta + (m-1)B + (m-1)\alpha)^2 = 2B(\beta B + \beta + (m-1)B + (m-1)\alpha) + B^2$; this would be the increase in the last open interval if the job was released at time $-(\beta + B\beta)$. There are m open time intervals at most $3m$ jobs, so the total increase in the 2-norm for type 1 jobs during open time intervals is upper bounded by $f_o = 6m^2B(\beta B + (m-1)B + (m-1)\alpha) + 3mB^2$.

The last statement follows by the definition of f. □

For the remainder of this section, let A be a schedule, with 2-norm of flow of at most f. To complete the proof we need to show that a 3-partition can be obtained by making the ith partition equal to the elements of S corresponding to the type 1 jobs in A finished between end of I_{i-1} and the end of I_i. This argument is structured as follows. Note that these lemmas are sufficient to find a valid solution to the 3-partition instance. This is because these lemmas show that between the end of I_{i-1} and the end of I_i exactly three jobs are completed and their total size is B.

- Lemma 2 states that A can only process a negligible amount of type 1 jobs during the closed time intervals I_i.
- At the end of each closed time interval I_i, for $i \in [0, m-1]$:
  - Lemma 3 states that A can have at most $3i$ type 1 jobs completed,
  - Lemma 4 states that in A the aggregate processing times of the unfinished type 1 jobs must be at least $B(m-i)$,
  - Lemma 5 states that A must have at least $3i$ type 1 jobs completed, and
  - Lemma 6 states that in A the aggregate processing times of the unfinished type 1 jobs can be at most $B(m-i)$.

Let U_i, $i \in [0, m-1]$ be the collection of type 1 jobs unfinished in A by the end of I_i.

Lemma 2. *For $i \in [0, m-1]$, the amount of time that A is not processing type 2 and type 3 jobs during I_i is at most $\frac{1}{2m}$.*

Proof. Let I_i be an interval where a $\frac{1}{2m}$ volume of work of Type 2 or type 3 jobs that arrived during I_i are not completed during I_i in A's schedule. Then at least $\frac{1}{4m\rho}$ jobs wait at least $\frac{1}{4m}$ time steps to be completed. Thus the cost of the schedule is at least $\frac{1}{16m^2\rho}$. This is strictly more than f. Informally this holds because $\max(m, B) \ll \alpha \ll \beta \ll \frac{1}{\rho}$. Formally this holds by our choice of parameters and algebraic calculations. □

Lemma 3. *For $i \in [0, m-1]$, $|U_i| \geq 3(m-i)$.*

Proof. By the end of an interval I_i there are have been at exactly iB time steps in the prior open time intervals. From Lemma 2 at most a $1/2$ volume of work can be processed by A on Type 1 jobs during closed time intervals. Knowing that the smallest Type 1 job has size $\frac{m}{3m+1/2}B$ and $B \geq 3$, the total number of jobs that can be completed before the end of I_i is $\left\lfloor (iB + 1/2) \Big/ (\frac{m}{3m+1/2}B) \right\rfloor \leq 3i$. □

Lemma 4. *For $i \in [0, m-1]$, $\sum_{j \in U_i} b_j \geq B(m-i)$.*

Proof. By Lemma 2 at most a $1/2$ volume of work on Type 1 jobs can be processed by A during closed time steps. The claim then follows by the integrality of B and the elements of S. □

Lemma 5. *For $i \in [0, m-1]$, $|U_i| \leq 3(m-i)$.*

Proof. Let I_j be an interval such that $3j-1$ or less Type 1 jobs are completed by the end of I_j in A. The increase the 2-norm of flow for type 1 jobs for A during I_j is then:

$$\begin{aligned}
&\sum_{l \in U_j} \left((s_j + \beta + \alpha + \beta b_l)^2 - (s_j + \beta + \beta b_l)^2 \right) \\
&= \sum_{l \in U_j} \left(2(s_j + \beta)\alpha + \alpha^2 + 2\beta b_l \alpha \right) \\
&= |U_j|((s_j + \beta)\alpha + \alpha^2) + 2\beta\alpha \sum_{l \in U_j} b_l \\
&\geq (3m - 3j + 1)((s_j + \beta)\alpha + \alpha^2) + 2\beta\alpha \sum_{l \in U_j} b_l \quad \text{[By definition of } I_j\text{]} \\
&\geq (3m - 3j + 1)((s_j + \beta)\alpha + \alpha^2) + 2\beta\alpha(mB - Bj) \quad \text{[By Lemma 4]} \\
&\geq f_1(j) + \beta\alpha
\end{aligned}$$

By Lemma 3, and the calculations in Lemma 1, the increase in the the 2-norm of flow for type 1 jobs for A during any I_i is at least $f_1(i)$. And the 2-norm of flow for type 2 and type 3 jobs in A is at least $f_{2,3}$. Thus to reach a contradiction, it is sufficient to show that $\beta\alpha > f_o = 6m^2B(\beta B + \beta + (m-1)B + (m-1)\alpha) + B^2$. Informally this holds because $\max(m, B) \ll \alpha \ll \beta$. Formally this holds by our choice of parameters and algebraic calculations. □

Lemma 6. *For* $i \in [0, m-1]$, $\sum_{j \in U_i} b_j \geq B(m-i)$.

Proof. The claim clearly holds $i = 0$. Assume to reach a contradiction that there is an interval I_j, $j \in [1, m-1]$ such that $\sum_{l \in U_j} b_l > B(m-j)$. Since the type 1 jobs have integral sizes, it must be the case that $\sum_{l \in U_j} b_l \geq B(m-j)+1$. Thus in increase in the 2-norm of flow for type 1 jobs in A during I_j must be at least:

$$\begin{aligned}
&\sum_{l \in U_j} \left((s_j + \beta + \alpha + \beta b_l)^2 - (s_j + \beta + \beta b_l)^2\right) \\
&= \sum_{l \in U_j} \left(2(s_j + \beta)\alpha + \alpha^2 + 2\beta b_l \alpha\right) \\
&= |U_j|((s_j + \beta)\alpha + \alpha^2) + 2\beta\alpha \sum_{l \in U_j} b_l \\
&\geq (3m - 3j)((s_j + \beta)\alpha + \alpha^2) + 2\beta\alpha \sum_{l \in U_j} b_l \quad \text{[By Lemma 3]} \\
&\geq (3m - 3j)((s_j + \beta)\alpha + \alpha^2) + 2\beta\alpha(mB - Bj + 1) \quad \text{[By assumption]} \\
&\geq f_1(j) + 2\beta\alpha
\end{aligned}$$

By Lemma 3, and the calculations in Lemma 1, the increase in the the 2-norm of flow for type 1 jobs for A during any I_i is at least $f_1(i)$. And the 2-norm of flow for type 2 and type 3 jobs in A is at least $f_{2,3}$. Thus to reach a contradiction, it is sufficient to show that $2\beta\alpha > f_o = 6m^2B(\beta B + \beta + (m-1)B + (m-1)\alpha) + B^2$. Informally this holds because $\max(m, B) \ll \alpha \ll \beta$. Formally this holds by our choice of parameters and algebraic calculations. □

4 Summary of the Other NP-Hardness Proofs

In the appendices, we give the complete proofs for k-norm, with $k > 2$ and $p \in (0, 1)$, and for stretch when $k \neq 1$. In this section, we give high level sketches of the proofs.

The proofs for all cases follow the same high level structure. They reduce from 3-partition, and the reduction introduces 3 different types of jobs. These classes will always play the same roles. Class 1 will contain the jobs corresponding to the 3-partition instance, released at some large negative time. Class 2 will contain jobs released during the positive time period, and will serve to partition time into intervals, each of which holds 3 type 1 jobs. Class 3 jobs will be released during the negative time period and there will be enough of them, released frequently enough, so that the type 1 jobs cannot run during this negative time.

The proofs will differ in the particular values used and on the methods of analysis needed to obtain the proofs. We summarize these differences for each problem below.

4.1 Hardness Proof for the k-Norm of Flow When $k \geq 3$

To emphasize the differences, we give the parameters of the reduction here.

- **Type 1 jobs:** For each integer $b_i \in S$ from the 3-partition instance, we create a job of processing time b_i and arrival time $-(\lambda\beta + \beta b_i)$. The value of β is set to be $2^{10k}m^7B^7$. Let T_1 denote the set of Type 1 jobs. These jobs will be indexed for $i \in [3m]$.
- **Type 2 jobs:** There is a job of size ρ is released every ρ time steps during the intervals $[iB + (i-1)\alpha, i(B+\alpha))$ for i from 1 to $m-1$. Here ρ and α are set such that $\alpha = 2^{6k}m^6B^6$ and $\rho = 1/(2m\beta\alpha)^{2k}$.
- **Type 3 jobs:** During the interval $[-(\lambda\beta + \beta B), 0]$ a job of size ρ is released every ρ time steps. Here $\lambda = B\sqrt{\alpha} = 2^{3k}B^4m^3$.

Note the differences from the case when $k = 2$. Most notably, we now have that β, α and ρ have an exponential dependency in k. We also have a new parameter λ, which is part of the definition of the negative time period. This period is much longer and the release date of the type 1 jobs is significantly smaller. We maintain the same relative (but not absolute) values of the other parameters, adding λ, we now have $\max(m, B) \ll \lambda \ll \alpha \ll \beta$. The proof uses a value of f that is

$$\begin{aligned} f := {} & \rho^{k-1}(\beta B + \lambda\beta + (m-1)\alpha) + \sum_{l \in T_1} (b_l\beta + \beta^2)^k \\ & + \sum_{i=1}^{m-1} \left((3m-3i)\left((s_i + \lambda\beta + \alpha)^k - (s_i + \lambda\beta)^k\right)\right. \\ & + \beta k \left((s_i + \lambda\beta + \alpha)^{k-1} - (s_i + \lambda\beta)^{k-1}\right)(mB - iB) + m2^{2k}(s_i + \lambda\beta)^{k-1}\Big) \\ & + 3m^2 2^k B(\beta B + \lambda\beta + (m-1)(\alpha + B))^{k-1}\,. \end{aligned}$$

The main technical challenge comes from the fact that the age of a job is no longer linear, but is now itself a polynomial function of degree $k-1$. In the proofs, we then evaluate the higher degree polynomial using the binomial theorem. By the choice of parameters, the terms form a series whose values are decreasing rapidly as the exponent decreases and we are therefore able to bound the polynomial by the first two terms of the expansion. We see the reason for the exponential dependence here, as we need the contributions to the objective function from moving the class 2 or 3 jobs by even a little bit to dominate the cost of packing the class 1 jobs. Without the exponential dependence on k, we would have too large a contribution from the aging of the class 1 jobs.

4.2 Hardness Proof for the k-Norm When $0 < k < 1$

We again give the parameters of the reduction.

- **Type 1 jobs:** For each integer $b_i \in S$ from the 3-partition instance, we create a job of processing time b_i and arrival time $-\beta + \lambda b_i$. The value of β is set to be $(30mkB)^{5/k^2+2}$ and λ is set to $\beta^{1/4}$. Let T_1 denote the set of Type 1 jobs. These jobs will be indexed for $i \in [3m]$. Note that because $\lambda B < \beta$ and $b_i \leq B$ for all i, it is the case that all jobs arrive before time 0.
- **Type 2 jobs:** There is a job of size ρ is released every ρ time steps during the intervals $[iB + (i-1)\alpha, i(B+\alpha))$ for i from 1 to $m-1$. Here ρ and α are set such that $\rho = 1/(100m^4\beta)$ and $\alpha = 10\beta^{3/4}m^2B^2$. We will assume without loss of generality that α/ρ and α/B are integral.

– **Type 3 jobs:** During the interval $[-\beta, 0]$ a job of size ρ is released every ρ time steps. We will assume that without loss of generally that β/ρ is integral.

Again, we have a dependence on k, but k appears in the exponent both as k and as $1/k$ in the numerator and as $1-k$ in the denominator. The proof uses a value of f that is

$$f := (\beta + (m-1)\alpha)/\rho^{1-k} + \frac{3m^2kB}{(\beta - \lambda B)^{1-k}} + \sum_{l \in T_1} (\beta - \lambda b_l)^k + \sum_{i=1}^{m-1} (3m - 3i) \cdot \left(\frac{k\alpha}{(s_i + \beta - \beta^{k/2})^{1-k}} - \frac{k(1-k)(\beta^{k/2})^2}{2(s_i + \beta - \beta^{k/2})^{2-k}} \right) \cdot \left(\frac{k(1-k)(\beta^{k/2})\lambda(mB - iB)}{(s_i + \beta - \beta^{k/2})^{2-k}} \right)$$

In this case, the main technical difference from the $k \geq 3$ case is that we use a Taylor series expansion to bound polynomials that have exponents that depend on k. Again, we have chosen the parameters so that the Taylor series used are rapidly decreasing and we are able to bound the series by just the first two or three terms. This allows us to make the cost of delaying the class 2 or class 3 jobs to be prohibitively large.

4.3 Hardness Proof for the 2-Norm of Stretch

We now outline the approach for stretch. Recall that stretch is flow time over processing time. Thus, if we think about the age of a job as a weight, we now have an age that depends not only on the release date and current time, but also on the processing time. Our first modification is to further restrict the range of values that the processing times can take on. By adding an appropriate constant to each item in the 3-partition instance, we can assume without loss of generality that in the 3-partition instance that $B/3 - \epsilon \leq b_i \leq B/3 + \epsilon$ for all $i \in [m]$ and $\epsilon \leq 1/(mB)^9$.

We then construct the identical instance as that for the 2-norm of flow time.

Let $\Delta_s = B/3 - \epsilon$ and $\Delta_b = B/3 + \epsilon$. Although the instance is the same as for flow, the value of f will be different, with, not surprisingly terms depending on the processing time in the denominator. More precisely, we will have terms with Δ_s^2 in the denominator, since, by our restriction on the b_i values, this approximates processing times well. The value of f is

$$f := \sum_{i \in T_1} \frac{1}{\Delta_s^2}(\beta + \beta b_i)^2 + \sum_{i=1}^{m-1} \left(\frac{1}{\Delta_s^2}(3m - 3i)((s_i + \beta)\alpha + \alpha^2) + \frac{1}{\Delta_s^2} 2\beta\alpha(mB - iB) \right) + \frac{1}{\Delta_s^2}(6m^2B(\beta B + (m-1)B + (m-1)\alpha) + 3mB^2) + (\beta B + \beta + (m-1)\alpha)/\rho \,.$$

We also have terms that use the ratio of Δ_s and Δ_b which, by the choice of these parameters is close to 1. The proof then follows along lines similar to that for flow squared.

4.4 Hardness Proofs for the k-Norm of Stretch, $k \geq 3$ and $k \in (0,1)$

Given the previous proofs, the hardness proofs for other norms of stretch combine the ideas from the corresponding proofs for that norm of flow, with the modifications made for the 2-norm of stretch. In particular, we restrict the range of b_i and then use the exact parameters from the corresponding reduction for flow. The analysis uses the same techniques as for flow, with the changed values to take into account the difference in objective. Again, because of our restriction on the range of b_i, the objective is actually close to the corresponding flow objective, which motivates the proofs.

5 Conclusion

We have shown the NP-completeness the k-norm of flow and stretch for integers $k \geq 2$ and $k \in (0,1)$. We believe that the techniques extend to non-integral $k > 1$, but we have not verified the details.

Our results leaves, as the most natural open problem, the complexity of computing the optimal 1-norm of stretch schedule. We believe that this problem is of fundamental importance, and is mathematically interesting. We advocate for this problem as the scheduling representative for a rumored second-generation list of NP-hardness open problems [GJ79]. The results of this paper can be taken as evidence that there is something uniquely interesting about the complexity of 1-norm of stretch schedules, and gives some explanation of the difficulties of finding an NP-hardness proof, if in case the problem is NP-hard.

References

[AGK12] Anand, S., Garg, N., Kumar, A.: Resource augmentation for weighted flow-time explained by dual fitting. In: ACM-SIAM Symposium on Discrete Algorithms, pp. 1228–1241 (2012)

[BMR04] Bender, M., Muthukrishnan, S., Rajaraman, R.: Approximation algorithms for average stretch scheduling. Journal of Scheduling 7, 195–222 (2004)

[BP10a] Bansal, N., Pruhs, K.: The geometry of scheduling. In: IEEE Symposium on Foundations of Computer Science, pp. 407–414 (2010)

[BP10b] Bansal, N., Pruhs, K.: Server scheduling to balance priorities, fairness, and average quality of service. SIAM J. Comput. 39(7), 3311–3335 (2010)

[BT06] Bussema, C., Torng, E.: Greedy multiprocessor server scheduling. Oper. Res. Lett. 34(4), 451–458 (2006)

[CGKK04] Chekuri, C., Goel, A., Khanna, S., Kumar, A.: Multi-processor scheduling to minimize flow time with epsilon resource augmentation. In: ACM Symposium on Theory of Computing, pp. 363–372 (2004)

[CIM09] Chekuri, C., Im, S., Moseley, B.: Longest Wait First for Broadcast Scheduling [Extended Abstract]. In: Bampis, E., Jansen, K. (eds.) WAOA 2009. LNCS, vol. 5893, pp. 62–74. Springer, Heidelberg (2010)

[CK02] Chekuri, C., Khanna, S.: Approximation schemes for preemptive weighted flow time. In: Symposium on Theory of Computing, pp. 297–305 (2002)

[EIM11] Edmonds, J., Im, S., Moseley, B.: Online scalable scheduling for the ℓk-norms of flow time without conservation of work. In: ACM-SIAM Symposium on Discrete Algorithms, pp. 109–119 (2011)

[FM11] Fox, K., Moseley, B.: Online scheduling on identical machines using srpt. In: ACM-SIAM Symposium on Discrete Algorithms, pp. 120–128 (2011)

[GIK+10] Gupta, A., Im, S., Krishnaswamy, R., Moseley, B., Pruhs, K.: Scheduling jobs with varying parallelizability to reduce variance. In: Symposium on Parallelism in Algorithms and Architectures, pp. 11–20 (2010)

[GJ79] Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman (1979)

[GKP12] Gupta, A., Krishnaswamy, R., Pruhs, K.: Online primal-dual for non-linear optimization with applications to speed scaling. In: Workshop on Approximation and Online Algorithms (2012)

[IM11] Im, S., Moseley, B.: Online scalable algorithm for minimizing;k-norms of weighted flow time on unrelated machines. In: ACM-SIAM Symposium on Discrete Algorithms, pp. 95–108 (2011)

[LLLRK82] Labetoulle, J., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G.: Preemptive scheduling of uniform machines subject to release dates. In: Progress in Combinatorial Optimization (January 1982)

The Euclidean k-Supplier Problem

Viswanath Nagarajan[1], Baruch Schieber[1], and Hadas Shachnai[2,⋆]

[1] IBM T.J. Watson Research Center, Yorktown Heights, NY 10598
[2] Computer Science Department, Technion, Haifa 32000, Israel
{viswanath,sbar}@us.ibm.com, hadas@cs.technion.ac.il

Abstract. In the *k-supplier* problem, we are given a set of clients C and set of facilities F located in a metric $(C \cup F, d)$, along with a bound k. The goal is to open a subset of k facilities so as to minimize the maximum distance of a client to an open facility, i.e., $\min_{S \subseteq F:|S|=k} \max_{v \in C} d(v, S)$, where $d(v, S) = \min_{u \in S} d(v, u)$ is the minimum distance of client v to any facility in S. We present a $1 + \sqrt{3} < 2.74$ approximation algorithm for the k-supplier problem in Euclidean metrics. This improves the previously known 3-approximation algorithm [9] which also holds for general metrics (where it is known to be tight). It is NP-hard to approximate Euclidean k-supplier to better than a factor of $\sqrt{7} \approx 2.65$, even in dimension two [5]. Our algorithm is based on a relation to the *edge cover* problem. We also present a nearly linear $O(n \cdot \log^2 n)$ time algorithm for Euclidean k-supplier in constant dimensions that achieves an approximation ratio of 2.965, where $n = |C \cup F|$.

1 Introduction

Location problems are an important class of combinatorial optimization problems that arise in a number of applications, e.g., choosing sites for opening plants, placing servers in a network, and clustering data. Moreover, the underlying distance function in many cases is Euclidean (ℓ_2 distance). In this paper, we study a basic location problem on Euclidean metrics.

The *Euclidean k-supplier* problem consists of n points in p-dimensional space, that are partitioned into a client set C and a set of facilities F. Additionally, we are given a bound $k \leq |F|$. The objective is to open a set $S \subseteq F$ of k facilities that minimizes the maximum distance of a client to its closest open facility. The k-supplier problem is a generalization of the *k-center* problem, where the client and facility sets are identical.

On general metrics, the k-supplier problem admits a 3-approximation algorithm [9]. There is a better 2-approximation algorithm for k-center, due to Hochbaum and Shmoys [8] and Gonzalez [6]. Moreover, these bounds are best possible assuming $P \neq NP$. On Euclidean metrics, Feder and Greene [5] showed that it is NP-hard to approximate k-supplier better than $\sqrt{7}$ and k-center better than $\sqrt{3}$. Still, even on 2-dimensional Euclidean metrics, the best known

⋆ Work partially supported by the Israel Science Foundation (grant number 1574/10), and by funding for DIMACS visitors.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 290–301, 2013.

approximation ratios remain 3 for k-supplier and 2 for k-center. In this paper, we derive the following improvement for Euclidean k-supplier:

Theorem 1. *There is a $(1+\sqrt{3})$-approximation algorithm for the Euclidean k-supplier problem in any dimension.*

It is worth noting that it remains NP-hard to approximate the k-supplier problem better than 3 if we use ℓ_1 or ℓ_∞ distances, even in 2-dimensional space [5]. Thus, our algorithms make heavy use of the Euclidean metric properties.

In many applications, such as clustering, the size of the input data may be very large. In such settings, it is particularly useful to have fast (possibly linear time) algorithms. Geometry plays a crucial role here, and many optimization problems have been shown to admit much faster approximation algorithms in geometric settings than in general metrics, for example TSP [1], k-median [7,10], or matching [15,14,1]. These papers consider the setting of low constant dimension, which is also relevant in practice; the running time is typically exponential in the dimension. For the Euclidean k-supplier problem in constant dimension, [5] gave a nearly linear $O(n \log k)$ time 3-approximation algorithm; whereas the best running time in general metrics is quadratic $O(nk)$ [9,6]. Extending some ideas from Theorem 1, we obtain a nearly linear time algorithm for Euclidean k-supplier having an approximation ratio better than 3.

Theorem 2. *There is an $O(n \cdot \log^2 n)$ time algorithm for Euclidean k-supplier in constant dimensions that achieves an approximation ratio ≈ 2.965.*

It is unclear if our algorithm from Theorem 1 admits a near-linear time implementation: the best running time that we obtain for constant dimensions is $O(n^{1.5} \log n)$. Both of our algorithms extend easily to the *weighted* k-supplier problem, where facilities have weights $\{w_f : f \in F\}$, and the goal is to open a set of facilities having total weight at most k.

Our Techniques and Outline. The $(1+\sqrt{3})$-approximation algorithm (Theorem 1) is based on a relation to the *minimum edge cover* problem, and is very simple. Recall, the edge cover problem [13] involves computing a subset of edges in an undirected graph so that every vertex is incident to some chosen edge; this problem is equivalent to maximum matching.[1] The entire algorithm is:

"Guess" the value of opt. *P is a maximal subset of clients C whose pairwise distance is more than $\sqrt{3} \cdot \mathsf{opt}$. Construct a graph G on vertices P that contains an edge for each pair $u, v \in P$ of clients that are both within distance* opt *from the same facility. Compute the minimum edge cover in G and output the corresponding facilities.*

The key property (which relies on the Euclidean metric) is that any facility can "cover" (within distance opt) at most two clients of P, which leads to a correspondence between k-supplier solutions and edge-covers in G. The main difference from [9,8,6] is that our algorithm uses information on *pairs* of clients that can be covered by a single facility.

[1] In any n-vertex graph without isolated vertices, it is easy to see that the minimum edge cover has size n minus the cardinality of maximum matching.

To implement the algorithm, we apply the fastest known algorithm for edge-cover, due to Micali and Vazirani [11], that runs in time $O(E_G\sqrt{V_G})$. In our setting this is $O(n^{1.5})$. However, in p-dimensional space, the algorithm to construct graph G takes $O(p\,n^2)$ time in general,[2] which results in an overall runtime of $O(p\,n^2)$. These results appear in Section 2.

When the *dimension is constant*, which is often the most interesting setting for optimization problems in Euclidean space, we show that a much better runtime can be achieved. Here, we can make use of good *approximate nearest neighbor* (ANN) data structures and algorithms [2,4,3]. These results state that with $O(n \log n)$ pre-processing time, one can answer $(1+\epsilon)$-approximate nearest-neighbor queries in $O(\log n)$ time each; where $\epsilon > 0$ is any constant. This immediately gives us an $O(n \log n)$ time algorithm to construct G, and hence an $\tilde{O}(n^{1.5})$ time implementation of Theorem 1 at the loss of a $1+\epsilon$ factor. Also, in the special case of dimension two, we can show that G is *planar* (see Section 2), so we can use the faster $O(n^{1.17})$ time planar matching algorithm due to Mucha and Sankowski [12] and obtain an $\tilde{O}(n^{1.17})$ time implementation of Theorem 1. However, it seems that there are no additional properties of G that we are able to use due to the following.

- Any degree 3 planar graph can be obtained as graph G for some instance of 2-D Euclidean k-supplier, and the fastest known matching algorithm even on this family of graphs still runs in $O(n^{1.17})$ time [12]. Indeed, there is a linear time reduction [12] from matching on general planar graphs to degree 3 planar graphs.
- Even in 3-D, the graph G does not necessarily exclude any fixed minor.[3] So, for higher constant dimensions, we need a general edge cover algorithm [11].

In Theorem 2 we provide a different algorithm (building on ideas from Theorem 1) that achieves near-linear running time, but a somewhat worse approximation ratio of 2.965: which is still better than the previous best bound of 3. The main idea here is to reduce to an edge cover problem on (a special class of) *cactus graphs*. Since (weighted) edge cover (and matching) on cactus graphs can be solved in linear time, the overall running time is dominated by the procedure to construct this edge-cover instance. Although the graph construction procedure here is more complicated, we show that it can also be implemented in $\tilde{O}(n)$ time using ANN algorithms [2]. The details appear in Section 3.

Remark: Our ideas do not extend directly to give a better than 2-approximation for the Euclidean k-center problem, which remains an interesting open question.

2 The $(1+\sqrt{3})$-Approximation Algorithm

For any instance of the k-supplier problem, it is clear that the optimal value is one of the $|F| \cdot |C|$ distances between clients and facilities. As with most

[2] The factor of p appears because, given a pair of points in p-dimension, it takes $O(p)$ time to even compute the Euclidean distance between them.

[3] Recall that a graph is planar *iff* it does not contain K_5 nor $K_{3,3}$ as a minor.

bottleneck optimization problems [9], our Algorithm 2.1 uses a procedure that takes as input an additional parameter L (which is one of the possible objective values) and outputs one of the following:

1. a certificate showing that the optimal k-supplier value is more than L, or
2. a k-supplier solution of value at most $\alpha \cdot L$.

Above, $\alpha = 1 + \sqrt{3}$ is the approximation ratio. The final algorithm uses binary search to find the best value of L.

Algorithm 2.1. Algorithm for Euclidean k-supplier

1: pick a maximal subset $P \subseteq C$ of clients such that each pairwise distance in P is more than $\sqrt{3} \cdot L$.
2: construct graph $G = (P, E)$ with vertex set P and edge set $E = E_1 \cup E_2$

$$E_1 = \{(u, v) \ : \ u, v \in P, \exists f \in F \text{ with } d(u, f) \leq L \text{ and } d(v, f) \leq L\}. \quad (1)$$

$$E_2 = \{(u, u) \ : \ u \in P, \exists f \in F \text{ with } d(u, f) \leq L \text{ and } \forall v \in P, (u, v) \notin E_1\}. \quad (2)$$

3: compute the *minimum edge cover* $\Gamma \subseteq E$ in G.
4: **if** $|\Gamma| > k$ **then**
5: the optimal value is larger than L.
6: **else**
7: output the facilities corresponding to Γ as solution.

We now prove the correctness of this algorithm. We start with a key property that makes use of Euclidean distances.

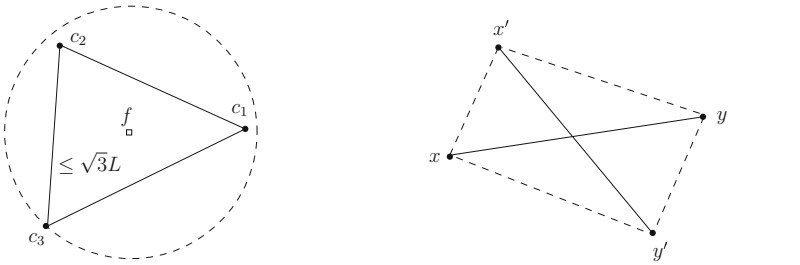
Lemma 1. *For any facility $f \in F$, the number of clients in P that are within distance L from f is at most two.*

Proof. To obtain a contradiction suppose that there is a facility f with three clients $c_1, c_2, c_3 \in P$ having $d(f, c_i) \leq L$ for $i \in \{1, 2, 3\}$. Consider now the plane containing c_1, c_2 and c_3 (which need not contain f). By taking the projection f' of f onto this plane, we obtain a circle centered at f' of radius at most L that has $\{c_i\}_{i=1}^3$ in its interior. See Figure 1 (A). Hence, the minimum pairwise distance in $\{c_i\}_{i=1}^3$ is at most $\sqrt{3} \cdot L$. This contradicts the fact every pairwise distance between vertices in P is greater than $\sqrt{3} \cdot L$. The lemma now follows. ■

This lemma provides a one-to-one correspondence between the edges E defined in (1)-(2) and facilities $H = \{f \in F : \exists u \in P \text{ with } d(u, f) \leq L\}$. Note that facilities in H that are within distance L of exactly one client in P give rise to self loops in E. Clearly, if the optimal k-supplier value is at most L then there is a set H' of at most k facilities in H so that each client in P lies within distance L of some facility of H'. In other words,

Claim 3. *If the optimal k-supplier value is at most L then graph G contains an edge cover of size at most k.*

Recall that an *edge cover* in an undirected graph is a subset of edges where each vertex of the graph is incident to at least one edge in this subset. The minimum size edge cover of a graph can be computed in polynomial time using algorithms



(A) Clients c_1, c_2, c_3 within distance L from facility f

(B) Possible edge crossing between (x, y) and (x', y')

Fig. 1. Examples for (A) Lemma 1 and (B) Lemma 3

for *maximum matching*, see, e.g., [13]. By Claim 3, if the minimum edge cover Γ is larger than k then we have a certificate for the optimal k-supplier value being more than L. This justifies Step 5. On the other hand, if the minimum edge cover Γ has size at most k then (in Step 7) we output the corresponding facilities (from H) as the solution.

Lemma 2. *If the algorithm reaches Step 7 then Γ corresponds to a k-supplier solution of value at most $(1 + \sqrt{3})L$.*

Proof. To reduce notation, we use Γ to denote both the edge cover in G and its corresponding facilities from H. Since Γ is an edge cover in G, each client $u \in P$ is within distance L of some facility in Γ.

$$\max_{u \in P} d(u, \Gamma) \quad \leq \quad L. \tag{3}$$

Now, since $P \subseteq C$ is a maximal subset satisfying the condition in Step 1, for each client $v \in C \setminus P$ there is some $u \in P$ with $d(u, v) \leq \sqrt{3}L$. Using (3) and triangle inequality, it follows that $\max_{v \in C} d(v, \Gamma) \leq (\sqrt{3} + 1)L$. ∎

Finally, we perform a binary search over the parameter L to determine the smallest value for which there is a solution. This proves Theorem 1.

Weighted Supplier Problem. Our algorithm extends easily to the weighted supplier problem, where facilities have weights $\{w_f : f \in F\}$, and the objective is to open facilities of total weight at most k that minimizes the maximum distance to any client. In defining edges in the graph G (Equation (1)-(2)) we also include weights of the respective facilities. Then we find a *minimum weight* edge cover Γ, which can also be done in polynomial time [13].

2.1 Running Time

We use $n = |F| + |C|$ to denote the total number of vertices. For arbitrary dimension $p \geq 2$, the running time can be naïvely bounded by $O(p\, n^2)$. This running

time is dominated by the time it takes to construct the graph G. The edge cover problem can be solved via a matching algorithm [11,13] that runs in time $O(E(G)\sqrt{V(G)}) = O(n^{3/2})$ since here $V(G) \leq |C|$ and $E(G) \leq |F|$.

When dimension p is constant, we provide a better running time implementation. There are two main parts in our algorithm: constructing the graph G and solving the edge cover problem on G. A naïve implementation of the first step results in an $O(n^2)$ running time. We show below that the runtime can be significantly improved, while incurring a small loss in the approximation ratio.

Constructing Graph G. The main component here is a fast data structure for *approximate nearest neighbor search* from Arya et al. [2].

Theorem 4 ([2]). *Consider a set of n points in $\mathbb{R}^p$. Given any $\epsilon > 0$, there is a constant $c \leq p\lceil 1 + 6p/\epsilon \rceil^p$ such that it is possible to construct a data structure in $O(pn \log n)$ time and $O(pn)$ space, with the following properties:*

- *For any "query point" $q \in \mathbb{R}^p$ and integer $\ell \geq 1$, a sequence of ℓ $(1+\epsilon)$-approximate nearest neighbors of q can be computed in $O((c+\ell p)\log n)$ time.*
- *Point insertion and deletion can be supported in $O(\log n)$ time per update.*

We will maintain such a data structures $\mathcal{P}$ for clients. First, we implement the step of finding a maximal "net" $P \subseteq C$ in Algorithm 2.2.

Algorithm 2.2. Algorithm for computing vertices P of G

1: initialize $P \leftarrow \emptyset$ and $\mathcal{P} \leftarrow \emptyset$.
2: **for** $v \in C$ **do**
3: $\quad v' \leftarrow$ approximate nearest neighbor of v in $\mathcal{P}$ (or NIL if $\mathcal{P} = \emptyset$).
4: $\quad$ **if** $d(v, v') > \sqrt{3}(1+\epsilon)L$ or v'=NIL **then**
5: $\quad\quad P \leftarrow P \cup \{v\}$ and insert v into $\mathcal{P}$.
6: output P.

Since we use $(1+\epsilon)$-approximate distances, the condition in Step 4 ensures that every pairwise distance in the final set P is at least $\sqrt{3}L$. Moreover, for each $u \in C \setminus P$, there is some $v \in P$ satisfying $d(u, v) \leq \sqrt{3}(1+\epsilon)L$. By Theorem 4, the time taken for each insertion and nearest-neighbor query in $\mathcal{P}$ is $O(\log n)$; so the total running time of this Algorithm 2.2 is $O(n \log n)$.

Next, Algorithm 2.3 shows how to compute the edge set E in (1)-(2).

Since all pairwise distances in P are larger than $\sqrt{3}L$, Lemma 1 implies that each facility $f \in F$ is within distance L of at most two clients in P. This is the reason we only look at the *two* approximate nearest neighbors (u and v) of f. Again, the condition for adding edges ensures that there is an edge in E for every facility in the set $H = \{f \in F : \exists u \in P \text{ with } d(u, f) \leq L\}$; since we use approximate distances, there might be more edges in E. By Theorem 4, the time for each 2-nearest neighbors query is $O(c \log n)$. Thus, the total time is $O(cn \log n)$, which is $O(n \log n)$ for any constant dimension p.

Computing Edge-Cover on G. Finding a minimum size edge cover is equivalent to finding a maximum cardinality matching on G. The fastest algorithm

Algorithm 2.3. Algorithm for computing edges E of G

```
construct data structure P containing points P, and initialize E ← ∅.
for f ∈ F do
    u ← approximate nearest neighbor of f in P.
    v ← approximate second nearest neighbor of f in P.
    if d(u, f) ≤ (1 + ε)L and d(v, f) > (1 + ε)L then
        set E ← E ∪ {(u, u)}.
    if d(u, f) ≤ (1 + ε)L and d(v, f) ≤ (1 + ε)L then
        set E ← E ∪ {(u, v)}.
output E.
```

for matching on general graphs takes $O(E(G)\sqrt{V(G)})$ time [11]. This results in an $O(n^{3/2})$ running time in our setting, since we only deal with sparse graphs.

We can obtain a better running time in $p = 2$ dimensions, by using the following additional property of the graph G.

Lemma 3. *If dimension is $p = 2$ and $\epsilon \leq 0.2$, the graph G is planar.*

Proof. Consider the natural drawing of graph G in the plane: each vertex in P is a point, and each edge $(u, v) \in E$ is represented by the line segment connecting u and v. To obtain a contradiction suppose that there is some crossing, say between edges (x, y) and (x', y'), see also Figure 1 (B). Notice that the distance between the end-points of any edge in E is at most $2(1 + \epsilon)L$, and the distance between any pair of points in P is at least $\sqrt{3}L$. Hence (setting $\epsilon \leq 0.2$), for any edge (u, v) and vertex $w \in P$, the angle uwv is strictly less than 90^o. Using this observation on edge (x, y) and points x' and y', we obtain that the angles $xx'y$ and $xy'y$ are both strictly smaller than 90^o. Similarly, for edge (x', y') and points x and y, angles $x'xy'$ and $x'yy'$ are also strictly smaller than 90^o. This contradicts with the fact that the sum of interior angles of quadrilateral $xx'yy'$ must equal 360^o. ∎

Based on this lemma, we can use the faster $O(n^{\omega/2})$ time randomized algorithm for matching on planar graphs, due to Mucha and Sankowski [12]. Here, $\omega < 2.38$ is the matrix multiplication exponent. Thus, we have shown:

Theorem 5. *For any constant $0 < \epsilon < 0.2$, there is a $(1 + \epsilon)(1 + \sqrt{3})$ factor approximation algorithm for Euclidean k-supplier that runs in time: $O(n^{1.5} \log n)$ for any constant dimension p, and $O(n^{1.17} \log n)$ for $p = 2$ dimensions.*

The additional $\log n$ factor comes from the binary search that we perform over the parameter L. We note that for larger dimension $p \geq 3$, the graph G does not necessarily have such nice properties. In particular, even in 3-dimensions G does not exclude any fixed minor.

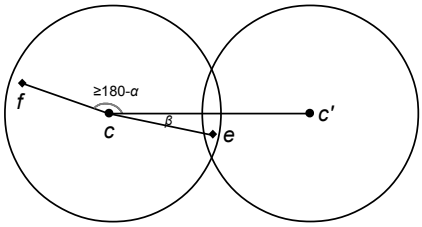
3 Nearly Linear Time 2.965-Approximation Algorithm

In this section, we give an $O(n \log^2 n)$ time approximation algorithm for Euclidean k-supplier in fixed dimensions. The approximation ratio we obtain is

2.965 which is worse than the $1 + \sqrt{3}$ bound from the previous section. We do not know a near-linear time implementation achieving that stronger bound. The algorithm here uses some ideas from the previous reduction to an edge-cover problem. But to achieve near-linear running time, we do not want to solve a general matching problem (even on planar graphs). Instead, we show that using additional Euclidean properties one can reduce to an edge-cover problem on (a special case of) *cactus* graphs. This approach gives a nearly linear time algorithm, since matching on cactus graphs can be solved in linear time.

Let $0 < \rho < 1$ be some constant and $0^o < \alpha, \beta < 30^o$ be angles, the values of which will be set later. To reduce notation, throughout this section, we normalize distances so that the parameter $L = 1$ (guess of the optimal value). For any point v, we denote the *ball* of radius one centered at v by $B(v)$. Two clients c and c' are said to *intersect* if **(i)** $d(c, c') \leq 2$ and **(ii)** there is some facility $f \in B(c) \cap B(c')$; if in addition, $d(c, c') > 2 \cos \beta$ then we call it a *fringe intersection*. Note that when c and c' have a fringe intersection, for any point $v \in B(c) \cap B(c')$ the angles $\angle vcc'$ and $\angle vc'c$ are at most β.

Given a client c and facility $f \in B(c)$, we say that another client c' is in *antipodal position* with respect to (abbreviated w.r.t.) $\langle f, c \rangle$ if the angle fcc' is more than $180^o - \alpha$; if in addition, c and c' have a fringe intersection, we say that c' has a fringe antipode intersection with $\langle f, c \rangle$. See figure to the right.



As in the previous section, the algorithm here builds a graph G on clients as vertices and facilities as edges. However, this procedure is more complex, since we want the resulting graph to have simpler structure: so that the edge cover problem on G can be solved in linear time.

The graph G is constructed iteratively, where each iteration adds a new component H as follows. We initialize H with an arbitrary pair c_1, c_2 of clients that have a fringe intersection, say with facility $f_0 \in B(c_1) \cap B(c_2)$; so H has vertices $V(H) = \{c_1, c_2\}$ and an edge (c_1, c_2) that is labeled f_0. (If there is no such pair, we pick an arbitrary client c_0 and set $H = \{c_0\}$ to be a singleton component.) Throughout the iteration, we maintain (at most) two *endpoint clients* x and y along with facilities $f \in B(x)$ and $g \in B(y)$; the role of these will become clear shortly. We will also refer to the tuples $\langle x, f \rangle$ and $\langle y, g \rangle$ as endpoints. Initially, set $x \leftarrow c_1$, $f \leftarrow f_0$, $y \leftarrow c_2$ and $g \leftarrow f_0$.

We repeatedly add to component H a new client c satisfying the following:

- c has a fringe antipode intersection with either $\langle x, f \rangle$ or $\langle y, g \rangle$.
- If $x \neq y$ and c intersects both, then c must be fringe antipode w.r.t. both $\langle x, f \rangle$ and $\langle y, g \rangle$.
- c does not intersect any client in $V(H) \setminus \{x, y\}$.

For a client c that satisfies these three conditions and is added to H, we distinguish the following two cases:

Case 1: Client c intersects exactly one of $\{x, y\}$, say x (the other case is identical). Let $f' \in B(x) \cap B(c)$ denote the facility in the (fringe) intersection of x and c. Add vertex c to $V(H)$ and an edge (x, c) labeled f'. Also set $x \leftarrow c$ and $f \leftarrow f'$.

Case 2: Client c intersects both x and y. Let $f_1 \in B(x) \cap B(c)$ and $f_2 \in B(y) \cap B(c)$ denote the facilities in the (fringe) intersections of x and c and of y and c, respectively. In this case, add vertex c to $V(H)$, and edges (c, x) labeled f_1 and (c, y) labeled f_2. Set $x \leftarrow c$, $f \leftarrow f_1$, $y \leftarrow c$ and $g \leftarrow f_2$.

The construction of component H ends when there are no new clients that can be added. At this point, we remove all clients that intersect with any client in $V(H)$ (these will be covered by a subset of facilities in $E(H)$), and iterate building the next component of G. Finally, we output an *edge cover* of G as the solution to the k-supplier problem.

Next, we prove some useful properties of the graph G.

Lemma 4. *Each component H is a cactus, where its simple cycles are linearly ordered. Hence, the edge-cover problem on G is solvable in linear time.*

Proof. It is easy to show by induction that H is a cactus with linearly ordered simple cycles. In each step, H grows by a new vertex c and (i) one edge from c to x (after which $x \leftarrow c$), or (ii) two edges, from c to x and y (after which $x, y \leftarrow c$).

A linear time algorithm for (weighted) edge-cover (and weighted matching) on cactus graphs can be obtained via a dynamic program. Here we just state an algorithm for the unweighted case of linearly ordered cycles. Such a graph G is given by a sequence $\langle v_1, v_2, \ldots, v_r \rangle$ of vertices, disjoint cycles $C_1, \ldots, C_{r-1}$ and a path C_r containing v_r. The cycles $C_1, \ldots, C_{r-1}$ and path C_r are vertex-disjoint except at the v_is: for each $j \in [r-1]$, $C_j \cap \{v_i\}_{i=1}^r = \{v_j, v_{j+1}\}$, and $C_r \cap \{v_i\}_{i=1}^r = \{v_r\}$.

For any $i \in [r]$, let $T[i, 0]$ denote the minimum edge cover in the graph $G_i := C_i \cup C_{i+1} \cdots \cup C_r$; and $T[i, 1]$ the minimum edge cover for graph G_i when vertex v_i is not required to be covered. The base cases $T[r, 0]$ and $T[r, 1]$ can be easily computed by considering all minimal edge covers of path C_r. We can write a recurrence for $T[i, *]$ as follows. Let e_{i+1} and f_{i+1} denote the two edges incident to v_{i+1} in the cycle C_i. Define the following minimal edge covers in C_i (each is unique subject to its condition).

- Γ_i^1 (resp. Γ_i^2) contains neither e_{i+1} nor f_{i+1}, and covers vertices $C_i \setminus v_{i+1}$ (resp. $C_i \setminus \{v_i, v_{i+1}\}$).
- Γ_i^3 (resp. Γ_i^4) contains e_{i+1} but not f_{i+1}, and covers vertices C_i (resp. $C_i \setminus \{v_i\}$).
- Γ_i^5 (resp. Γ_i^6) contains f_{i+1} but not e_{i+1}, and covers vertices C_i (resp. $C_i \setminus \{v_i\}$).
- Γ_i^7 (resp. Γ_i^8) contains both f_{i+1} and e_{i+1} and thus not the other edge incident to e_{i+1}, and covers vertices C_i (resp. $C_i \setminus \{v_i\}$).

Then we have for all $r \in [r-1]$,

$$T[i,0] := \min\left\{T[i+1,0] + \Gamma_i^1,\ T[i+1,1] + \min\left\{\Gamma_i^3, \Gamma_i^5, \Gamma_i^7\right\}\right\}$$

$$T[i,1] := \min\left\{T[i+1,0] + \Gamma_i^2,\ T[i+1,1] + \min\left\{\Gamma_i^4, \Gamma_i^6, \Gamma_i^8\right\}\right\}$$

Clearly this dynamic program can be solved in linear time. ■

Claim 6. *If the optimal k-supplier value is at most 1, then graph G has an edge cover of size k.*

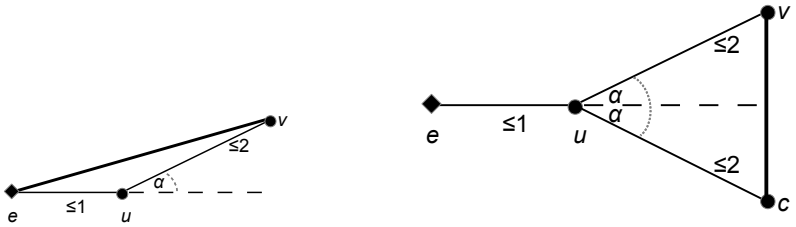
Proof. We show that each facility can cover at most two clients in $V(G)$, and that there is an edge in G between *every* pair of clients in $V(G)$ that can be covered by a single facility. This would imply the claim.

Note that if a pair of vertices in $V(G)$ intersect then this intersection is fringe intersection, i.e., any such pairwise distance is at least $2\cos\beta \geq \sqrt{3}$. Hence, by Lemma 1, each facility can cover (within distance one) at most two clients of $V(G)$. Moreover, by the construction of each component H, the edge set $E(H)$ contains *all* intersections between pairs of vertices in $V(H)$. Also, clients in different components of G do not intersect. This is because we remove all clients intersecting with $V(H)$ after constructing component H. ■

We now prove that this algorithm achieves an approximation ratio $3-\rho$. Below we consider a particular component H. The variables x, f, y, g will denote their values at the end of H's construction (unless specified otherwise).

Claim 7. *For any client $u \in V(H)$ and edge (facility) $e = (u, u') \in E(H)$ such that $\langle e, u\rangle \notin \{\langle f, x\rangle, \langle g, y\rangle\}$, and client $v \in C$ that intersects u, either $d(e,v) \leq 3-\rho$ or $d(v, V(H)) \leq 2-\rho$.*

Proof. The Claim holds trivially for $v \in V(H)$. Consider clients $u \in V(H)$ and $v \in C \setminus V(H)$ as stated. If $d(u,v) \leq 2\cos\beta$ then, clearly, $d(v, V(H)) \leq d(v,u) \leq 2\cos\beta$. Else, if v is *not* in antipodal position w.r.t. $\langle e, u\rangle$, then by the cosine rule, $d(e,v) \leq \sqrt{1^2 + 2^2 + 2 \cdot 2\cos\alpha}$ (see Figure 2a)



(a) v is not antipodal w.r.t. $\langle e, u\rangle$. (b) $\langle e, u\rangle$ was an endpoint and c was added.

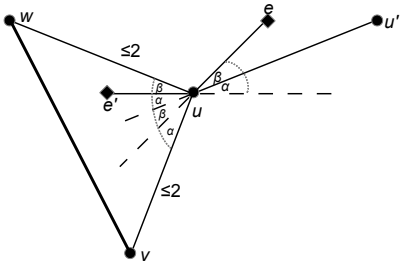
Fig. 2. Cases from Claim 7

Below, we assume that $2\cos\beta \leq d(u,v) \leq 2$ and u is in antipode position w.r.t. $\langle e, u\rangle$. Since $\langle e, u\rangle \notin \{\langle f, x\rangle, \langle g, y\rangle\}$, one of the following two cases must be true.

Case 1: At some earlier iteration, $\langle e, u\rangle$ was an endpoint and some client c was added due to a fringe antipode intersection w.r.t. $\langle e, u\rangle$. In this case we will bound $d(v, V(H)) \leq d(v, c)$. Since both v and c are in antipode position w.r.t. $\langle e, u\rangle$, the angle $\angle vuc$ is at most 2α. Again by the cosine rule, $d(v, c) \leq \sqrt{2^2 + 2^2 - 2 \cdot 2 \cdot 2\cos 2\alpha}$ (see Figure 2b).

Case 2: At some earlier time, $\langle e', u\rangle$ was an endpoint where $e' = (w, u) \neq e$, and $e = (u, u')$ was added due to u' having a fringe antipode intersection w.r.t. $\langle e', u\rangle$. In this case, we will bound $d(v, V(H)) \leq d(v, w)$; recall $w \in V(H)$ is the earlier occurring vertex of e'. See also the figure on the right.

Since u' is antipodal w.r.t. $\langle e', u\rangle$, the angle $\angle e'uu'$ is between $180^o - \alpha$ and 180^o. Moreover, u' has a fringe intersection with u and $e \in B(u) \cap B(u')$, so the angle $\angle euu'$ is at most β. Hence $\angle eue'$ is between $180^o - (\alpha + \beta)$ and 180^o. Again, $\angle euv$ is between $180^o - \alpha$ and 180^o, since v is in antipodal position with $\langle e, u\rangle$. So $\angle vue'$ is at most $2\alpha + \beta$. Finally, $\angle e'uw$ is at most β since u and w have a fringe intersection with $e' \in B(w) \cap B(u)$.



Thus we have $\angle vuw \leq 2\alpha + 2\beta$. By the cosine rule,

$$d(v, w) \leq \sqrt{2^2 + 2^2 - 2 \cdot 2 \cdot 2\cos(2\alpha + 2\beta)}.$$

We need to choose ρ, α, and β so that the following three constraints hold:

1. $2\cos\beta \leq 2 - \rho$
2. $\sqrt{5 + 4\cos\alpha} \leq 3 - \rho$
3. $\sqrt{8 - 8\cos(2\alpha + 2\beta)} \leq 2 - \rho$

Setting $\rho < 0.035$, $\alpha \approx 18.59^o$, and $\beta \approx 10.73^o$ satisfies all three constraints. Thus, the claim holds in all the above cases. ∎

Lemma 5. *Any edge cover Γ of G corresponds to a k-supplier solution of value at most $3 - \rho$.*

Proof. Since Γ is an edge cover of G, it covers clients $V(G)$ within distance one. It is clear that each client in C intersects with some client in $V(G)$. It suffices to show that for each component H and client $v \in C \setminus V(H)$ that intersects some $u \in V(H)$, the distance $d(v, \Gamma) \leq 3 - \rho$.

Let $e \in \Gamma \cap B(u)$ be the edge (facility) in the edge cover Γ that is incident to vertex $u \in V(H)$. If $\langle e, u\rangle \notin \{\langle f, x\rangle, \langle g, y\rangle\}$ (at the end of constructing component H) then by Claim 7, either $d(e, v) \leq 3 - \rho$ or $d(v, V(H)) \leq 2 - \rho$. So, $d(v, \Gamma) \leq \min\{d(e, v), 1 + d(v, V(H))\} \leq 3 - \rho$.

Now, suppose $\langle e, u\rangle = \langle f, x\rangle$ when the construction of H is complete (the other case of $\langle g, y\rangle$ is identical). We consider the following cases:

Case 1: Client v does not have a fringe antipode intersection w.r.t. $\langle f, x\rangle$. Then, as in the initial cases of Claim 7, $d(v, \Gamma) \leq d(v, f) \leq 3 - \rho$.

Case 2: Client v intersects with some client $u' \in V(H) \setminus \{x, y\}$. Then applying Claim 7 to u' and edge $e' \in \Gamma \cap B(u')$ yields $d(v, \Gamma) \leq 3 - \rho$.

Case 3: If none of the above two cases hold, then we must have $y \neq x$ and v has a non antipode intersection w.r.t. $\langle g, y\rangle$: otherwise, v would have been added to H as a new client. Let $e' \in \Gamma \cap B(y)$, and consider two sub-cases:

- If $e' = g$, then since v has a *non antipodal* intersection w.r.t. $\langle g, y\rangle$, $d(v, \Gamma) \leq d(v, e') = d(v, g) \leq 3 - \rho$.
- If $e' \neq g$, then Claim 7 applies since $e' \in E(H) \setminus \{f, g\}$ and yields $d(v, \Gamma) \leq 3 - \rho$.

In all the cases, we have shown $d(v, \Gamma) \leq 3 - \rho$, which proves the lemma. ∎

In the full version we give the details of implementing this algorithm in near-linear time, which completes the proof of Theorem 2.

References

1. Arora, S.: Nearly linear time approximation schemes for Euclidean TSP and other geometric problems. In: FOCS, pp. 554–563 (1997)
2. Arya, S., Mount, D.M., Netanyahu, N.S., Silverman, R., Wu, A.Y.: An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. J. ACM 45(6), 891–923 (1998)
3. Chan, T.M.: Approximate nearest neighbor queries revisited. Discrete & Computational Geometry 20(3), 359–373 (1998)
4. Clarkson, K.L.: An algorithm for approximate closest-point queries. In: Symposium on Computational Geometry, SoCG, pp. 160–164 (1994)
5. Feder, T., Greene, D.H.: Optimal algorithms for approximate clustering. In: STOC, pp. 434–444 (1988)
6. Gonzalez, T.F.: Clustering to minimize the maximum intercluster distance. Theor. Comput. Sci. 38, 293–306 (1985)
7. Har-Peled, S., Mazumdar, S.: On coresets for k-means and k-median clustering. In: STOC, pp. 291–300 (2004)
8. Hochbaum, D.S., Shmoys, D.B.: A best possible heuristic for the k-center problem. Mathematics of Operations Research 10(2), 180–184 (1985)
9. Hochbaum, D.S., Shmoys, D.B.: A unified approach to approximation algorithms for bottleneck problems. J. ACM 33(3), 533–550 (1986)
10. Kolliopoulos, S.G., Rao, S.: A nearly linear-time approximation scheme for the Euclidean k-median problem. SIAM J. Comput. 37(3), 757–782 (2007)
11. Micali, S., Vazirani, V.V.: An $O(\sqrt{V}E)$ Algorithm for Finding Maximum Matching in General Graphs. In: FOCS, pp. 17–27 (1980)
12. Mucha, M., Sankowski, P.: Maximum matchings in planar graphs via gaussian elimination. Algorithmica 45(1), 3–20 (2006)
13. Schrijver, A.: Combinatorial optimization. Springer, New York (2003)
14. Vaidya, P.M.: Approximate minimum weight matching on points in k-dimensional space. Algorithmica 4(4), 569–583 (1989)
15. Vaidya, P.M.: Geometry helps in matching. SIAM J. Comput. 18(6), 1201–1225 (1989)

Facial Structure and Representation of Integer Hulls of Convex Sets

Vishnu Narayanan

Industrial Engineering and Operations Research, Indian Institute of Technology Bombay, Powai, Mumbai 400076, India
vishnu@iitb.ac.in

Abstract. For a convex set S, we study the facial structure of its integer hull, $S_{\mathbb{Z}}$. Crucial to our study is the decomposition of the integer hull into the convex hull of its extreme points, $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$, and its recession cone. Although $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ might not be a polyhedron, or might not even be closed, we show that it shares several interesting properties with polyhedra: all faces are exposed, perfect, and isolated, and maximal faces are facets. We show that $S_{\mathbb{Z}}$ has an infinite number of extreme points if and only if $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ has an infinite number of facets. Using these results, we provide a necessary and sufficient condition for semidefinite representability of $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$.

Keywords: Integer hull, Facial structure, Extreme points.

1 Introduction

Convex integer optimization problems (i.e., nonlinear integer programs whose continuous relaxations have convex feasible regions) have attracted a lot of attention recently. Several computational schemes (e.g. [1–7, 11, 19, 21, 22, 34]) have been successfully developed for large classes of structured NLMIPs. However, fundamental results on the structure of feasible solutions to NLMIPs have been very few. There has been some work on structure of elementary (Chvátal-Gomory and Split) cut closures of NLMIPs and irrational polyhedra with compact feasible regions [12–14, 16, 17]. Braun and Pokutta [8] give a short proof that the Chvátal-Gomory closure of a compact convex body is a polytope. Moussafir [28] shows that the integer hull of a polyhedron (not necessarily rational) is locally polyhedral under some conditions. Dey and Morán R. [15] examine the closedness of convex hulls of integer points, and give necessary and sufficient conditions for the integer hull being a polyhedron. Morán R. et al. [27] develop a strong dual for conic mixed-integer programming. Burer and Letchford [9] study the extreme points and facet-defining inequalities of a class of unbounded integer hulls arising in mixed-integer quadratic programming. For classical results on lattice points in convex bodies, we direct the reader to Cassels [10].

It is well-known that for rational polyhedra, the convex hull of mixed-integer feasible points is a rational polyhedron [26]. However, such a result is not true in the nonlinear case, as the following example shows.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 302–313, 2013.

Example 1. Consider the set $Y = \left\{x \in \mathbb{R}^2_+ \ : \ x_2 \geq x_1^2\right\}$, the convex set bounded by a parabola and the vertical axis in $\mathbb{R}^2$. It is easy to prove ([see 9]) that

$$\operatorname{conv}(Y \cap \mathbb{Z}^2) = \left\{x \in \mathbb{R}^2_+ \ : \ x_2 \geq (2t+1)x_1 - t(t+1), \ t \in \mathbb{Z}_+\right\},$$

which is not polyhedral. However, $Y_{\mathbb{Z}} := \operatorname{conv}(Y \cap \mathbb{Z}^2)$ is locally polyhedral, i.e., $Y_{\mathbb{Z}} \cap P$ is a polytope for every polytope P. It is also easy to see that the maximal proper faces of $Y_{\mathbb{Z}}$ are facets.

Notation. Let $S \subseteq \mathbb{R}^n$ be a closed, convex set. We assume that S is line free. We define the *integer hull of* S as

$$S_{\mathbb{Z}} := \operatorname{cl}\operatorname{conv}(S \cap \mathbb{Z}^n) \ .$$

Here, $\operatorname{conv}(X)$ denotes the convex hull of a set X, and $\operatorname{cl}(X) = X \cup \operatorname{rbd}(X)$ denotes its closure, where $\operatorname{rbd} X$ denotes the relative boundary of X. We let $\operatorname{ri}(X)$, $\operatorname{aff}(X)$, and $\operatorname{cone}(X)$, denote the relative interior, affine hull, and the conical hull, respectively, of X. If X is convex, we use $\dim(X)$, $\operatorname{ext}(X)$ and $\operatorname{rec}(X)$ to denote the (affine) dimension, set of extreme points, and the recession cone of X, respectively. Also, let let $\mathbf{0}$ denote the vector of all zeros, and let $\mathbf{e}_j$ denote the j-th standard basis vector in $\mathbb{R}^n$. For any set $X \subseteq \mathbb{R}^n$, let

$$\sigma_X(u) := \sup\left\{\langle u, x\rangle \ : \ x \in X\right\}$$

be the support function of X.

Contributions and Outline. First, we show that if $\operatorname{rec}(S)$ has a tractable representation, then so does $\operatorname{rec}(S_{\mathbb{Z}})$ (Proposition 1). Then, $\operatorname{rec}(S_{\mathbb{Z}})$ does not create any problems as far as representation is concerned, and we can study separately the extreme points and recession cone of $S_{\mathbb{Z}}$. Our main object of study in this paper is $\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))$ as opposed to $S_{\mathbb{Z}}$ as its faces have better structural properties (see Examples 3, 4). Other reasons for studying $\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))$ are

- any face of $S_{\mathbb{Z}}$ is the Minkowski sum of faces of $\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))$ and $\operatorname{rec}(S_{\mathbb{Z}})$, and
- $\sigma_{S_{\mathbb{Z}}}(u) = \sigma_{\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))}(u)$, i.e., if a linear optimization problem over $S_{\mathbb{Z}}$ has a solution, then it has a solution in $\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))$.

We study separation properties of faces of $\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))$ in Section 2. Using these, we show that all faces of $\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))$ are exposed (Proposition 2), and that any bounded subset of a face F of $\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))$ can be strongly separated from the extreme points that do not contain F. Using these separation properties, we study the properties of tangent and normal cones of faces of $\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))$ and show in Section 3 that all faces of $\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))$ are perfect (Theorem 1). The results in this Section are sufficient to discuss the semidefinite representability of $\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))$. Note that $\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))$ need not be a closed set in general.

However, our results are not affected because of this. For closedness properties of $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$, see Dey and Vielma [16].

However, we go a step further (Section 4) and show that the faces of $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ are isolated (i.e., points from relative interiors of faces converge to only faces of lower dimension, see Definition 4). Isolation, along with properties of normal vectors of $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$, allows us to establish that all maximal faces of $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ are facets (Theorem 2), i.e., they have dimension $n-1$ (assuming that $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ is full-dimensional). Thus, maximal faces of $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ are facets, despite the fact that $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ need not be locally polyhedral (Example 4). We also establish that if $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ has infinitely many extreme points, then $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ also has infinitely many facets (Theorem 4).

In Section 5, we show that if $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ has infinitely many facets, then $\text{cl}(\text{conv}(\text{ext}(S_{\mathbb{Z}})))$ is not semidefinite representable. Specifically, the set $Y_{\mathbb{Z}}$ of Example 1 is not semidefinite representable.

Before we start studying the faces of $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$, we first show that $\text{rec}(S_{\mathbb{Z}})$ can be tractably represented if $\text{rec}(S)$ can be.

Proposition 1. *Let $W \subseteq \mathbb{R}^n$ be the subspace parallel to $\text{aff}(S_{\mathbb{Z}})$. Then, $\text{rec}(S_{\mathbb{Z}}) = W \cap \text{rec}(S)$.*

Proof. We may assume that $\mathbf{0} \in S$, and thus, $W = \text{aff}(S)$. It is clear that $\text{rec}(S_{\mathbb{Z}}) \subseteq \text{aff}(S_{\mathbb{Z}}) \cap \text{rec}(S)$.

For the reverse inclusion, let $d \in \text{ri}\,\text{rec}(S) \cap \text{aff}(S_{\mathbb{Z}})$. Using an argument similar to that of Lemma 2 of Braun and Pokutta [8], for any $k_0 \in \mathbb{N}$ and $\varepsilon > 0$, there exists integer $k \geq k_0$ and $a \in \mathbb{Z}^n$ with $a - kd \in S$ and $\|a - kd\| < \varepsilon$. Hence, the integer point $a = (a - kd) + kd \in S + \text{rec}(S) = S$; i.e., there exist integer points of S arbitrarily close to $\mathbf{0} + kd$, which implies that d is a recession direction for $S_{\mathbb{Z}}$. □

2 Separation Properties of Faces

By restriction to the affine hull of $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$, which is generated by integral vectors, we may assume throughout this paper without loss of generality that the dimension of $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ is n.

We first show here that for any face F of $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$, F can be separated from the extreme points of $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ not lying in F. Note that this property is not true for convex sets in general, e.g., the unit ball.

Definition 1. *Let $S, T \subseteq \mathbb{R}^n$ be disjoint convex sets, and let H be an affine hyperplane in $\mathbb{R}^n$. We say that S and T are properly separated by H if S and T belong to opposing closed halfspaces defined by H, and $(S \cup T) \nsubseteq H$.*

We say that S and T are strongly separated by H if there exists an $\varepsilon > 0$ such that $S + \varepsilon\mathbb{B}^n$ and $T + \varepsilon\mathbb{B}^n$ belong to opposing open halfspaces defined by H.

For any face F of $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$, Let $E_F := \text{ext}(F)$, and let $\widehat{E}_F := (S \cap \mathbb{Z}^n) \setminus F$ for the remainder of this section.

Lemma 1. *The sets* $\mathrm{cl}(F)$ *and* $\mathrm{cl}(\mathrm{conv}(\widehat{E}_F))$ *can be properly separated by a hyperplane.*

Proof. As $\widehat{E}_F$ is closed, all extreme points of $\mathrm{cl}(\mathrm{conv}(\widehat{E}_F))$ belong to $\widehat{E}_F$ [25]. As a result, $\mathrm{cl}(F) \cap \mathrm{cl}(\mathrm{conv}(\widehat{E}_F)) = \varnothing$, and $\mathrm{ri}(F) \cap \mathrm{ri}(\mathrm{conv}(\widehat{E}_F) = \varnothing$. By Theorem 11.3 of Rockafellar [30], there exists an affine hyperplane H properly separating $\mathrm{cl}(F)$ and $\mathrm{cl}\,\mathrm{conv}(\widehat{E}_F)$, i.e., $\widehat{E}_F \nsubseteq H$. □

Proposition 2. *All faces of* $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ *are exposed.*

Proof. Follows as a corollary to Lemma 1.

Lemma 2. *Let* $W \subseteq \mathrm{cl}(F)$ *be compact. Then,* W *and* $\mathrm{cl}(\mathrm{conv}(\widehat{E}_F))$ *can be separated strongly by a hyperplane.*

Proof. We know that $\mathrm{cl}(F) \cap \mathrm{cl}(\mathrm{conv}(\widehat{E}_F)) = \varnothing$. Therefore, $W \cap \mathrm{cl}(\mathrm{conv}(\widehat{E}_F)) = \varnothing$. The result then follows from Corollary 11.4.2 of Rockafellar [30]. □

Although we do not need the stronger result, we conjecture that $\mathrm{cl}(F)$ and $\mathrm{cl}(\mathrm{conv}(\widehat{E}_F))$ can be separated strongly.

3 Perfection of Faces

If F is a face of a polyhedron P, it is true that there are $\dim(P) - \dim(F)$ linearly independent normal vectors for F. Here, we generalize this property to $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$.

Definition 2. *Let* $C \subseteq \mathbb{R}^n$ *be a convex set and let* $x \in C$*. Then, the tangent cone to* C *at* x *is* $\mathbf{T}_C(x) := \mathrm{cl}\,\mathrm{cone}(C - \{x\})$*. The normal cone to* C *at* x *is the polar of the tangent cone:* $\mathbf{N}_C(x) := \mathbf{T}_C(x)^\circ = \{\xi \ : \ \langle \xi, \zeta \rangle \leq 0 \text{ for all } \zeta \in \mathbf{T}_C(x)\}$*. If* F *is a face of* C*, then* $\mathbf{T}_C(F) := \mathbf{T}_C(x)$*, and* $\mathbf{N}_C(F) := \mathbf{N}_C(x)$ *for any* $x \in \mathrm{ri}(F)$*.*

Remark 1. $\mathbf{T}_C(x)$ is the closure of the cone of feasible directions at $x \in C$. Also, $\mathrm{ri}(\mathbf{N}_C(F))$ is the set of "objective functions" that have their maximum on a face F of C. We will omit the subscript C when the context is clear.

Definition 3. *Let* $C \subseteq \mathbb{R}^n$ *be a convex set, and let* F *be a face of* C*. Then,* F *is said to be perfect* [33, Section 2.2] *if* $\dim(F) + \dim \mathbf{N}_C(F) = n$*.*

Lemma 3. *Let* F *be a face of* $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}})) \subseteq \mathbb{R}^n$*, and let* $x \in \mathrm{ri}(F)$*. If* $d, -d \in \mathbf{T}_{\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))}(x)$*, then* $x + \gamma d \in F$ *for some* $\gamma > 0$*. In other words, the lineality space of* $\mathbf{T}_{\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))}(x)$ *has dimension* $\dim(F)$*.*

Proof. For the purpose of computing the Tangent cone, we translate $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ by $-x$ and assume that $x = \mathbf{0}$ throughout this proof.

We know that $\mathbf{T}(\mathbf{0}) := \mathbf{T}_{\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))}(\mathbf{0}) = \mathrm{cl}\,\mathrm{cone}(\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}})) - \{\mathbf{0}\}) = \mathrm{cl}\,\mathrm{cone}(\mathrm{ext}(S_{\mathbb{Z}}))$. Splitting $\mathrm{ext}(S_{\mathbb{Z}})$ into those in F and otherwise, we get $\mathbf{T}(\mathbf{0}) = \mathrm{cl}\left(\mathrm{cone}(E_F) + \mathrm{cone}\,\widehat{E}_F\right)$.

Claim. $\mathbf{T}(\mathbf{0}) = \operatorname{cl}\operatorname{cone} E_F + C$ for some closed cone C.

Proof of Claim. Since $\mathbf{0} \in \operatorname{ri}(F)$, we see that $V := \operatorname{cone}(\widehat{E}_F) = \operatorname{cl}\operatorname{cone}(\widehat{E}_F)$ is a vector space. Let $C' := \operatorname{cone}(E_F)$, and let $\widetilde{C} := \operatorname{proj}_{V^\perp} C'$ denote the projection of C' onto the orthogonal complement $V^\perp$ of V. Then, $C' + V = \widetilde{C} + V$, and we have $\operatorname{cl}(\widetilde{C}) \cap V = \{\mathbf{0}\}$, as $\widetilde{C} \subseteq V^\perp$. Therefore, we may assume without loss of generality that $\operatorname{cl}(C') \cap V = \{\mathbf{0}\}$. Let $C := \operatorname{cl}(C')$.

We then have $C = (C^*)^*$, and $V = -V = (-V^\perp)^*$ (here, L^* is used to denote the dual cone of L). Also, $(C^*)^\perp \cap V \subseteq (C^*)^* \cap V$, and $\mathbf{0} \in (C^*)^\perp \cap V$ as it is an intersection of vector spaces. Therefore, we have

$$(C^*)^* \cap (-V)^\perp)^* = C \cap V = \{\mathbf{0}\} = (C^*)^\perp \cap (V^\perp)^\perp \ ,$$

and $C + V$ is closed [see 29, p. 408]. It is well known that $\operatorname{cl}(C') + V \subseteq \operatorname{cl}(C' + V)$ [see 30, Theorem 6.6]. Since $\operatorname{cl}(C') + V$ is a closed set containing $C' + V$, we have $C + V = \operatorname{cl}(C') + V = \operatorname{cl}(C' + V)$, and the claim is true. □

Returning to the proof of Lemma 3, we first show that $\operatorname{cl}\operatorname{cone}(\widehat{E}_F)$ is pointed. Suppose that $d, -d \in \operatorname{cl}\operatorname{cone}(\widehat{E}_F) \setminus \{\mathbf{0}\}$. Then, there exist sequences $\{\xi_k\}_{k\in\mathbb{N}}$, $\{\zeta_k\}_{k\in\mathbb{N}} \subseteq \operatorname{cone}(\widehat{E}_F)$ with $\xi_k \to d$, and $\zeta_k \to -d$. Rewriting

$$\xi_k = \sum_{i=1}^{n} \alpha_{ki} z_{ki}, \text{ and } \zeta_k = \sum_{i=1}^{n} \beta_{ki} w_{ki}, \ k \in \mathbb{N} \ ,$$

where $\alpha_{ki}, \beta_{ki} \geq 0$; $z_{ki}, w_{ki} \in \widehat{E}_F$, for all $k \in \mathbb{N}$, $i = 1, \ldots, n$ (the upper limit of n in the summation comes from Carathéodory's Theorem). Letting $\alpha_k := \sum_{i=1}^{n} \alpha_{ki} > 0$ and $\beta_k := \sum_{i=1}^{n} \beta_{ki} > 0$, we get for any $\varepsilon > 0$, there exists a $k_0 \in \mathbb{N}$ such that for all $k \geq k_0$, we have

$$\left\| \sum_{i=1}^{n} \left(\frac{\alpha_{ki}}{\alpha_k + \beta_k} z_{ki} + \frac{\beta_{ki}}{\alpha_k + \beta_k} w_{ki} \right) - \mathbf{0} \right\| < \varepsilon \ .$$

In other words, $\operatorname{dist}(\mathbf{0}, \operatorname{conv}\{z_{ki}, w_{ki}, i = 1, \ldots, n\}) \to 0$ as $k \to \infty$. However, $\mathbf{0} \in W \subseteq \operatorname{ri}(F)$, where W is compact, and $\{z_{ki}, w_{ki}, i = 1, \ldots, n\} \subseteq \operatorname{conv} \widehat{E}_F$, and by Lemma 2, W and $\operatorname{cl}\operatorname{conv}(\widehat{E}_F)$ can be strongly separated, which is a contradiction, and hence, $\operatorname{cl}\operatorname{cone}(\widehat{E}_F)$ is pointed.

The proof is now completed using the fact that $\operatorname{ri}(\operatorname{cone}(E_F)) \cap \operatorname{ri}(C) = \varnothing$. □

We now present the main result of this section.

Theorem 1. *Let F be a face of* $\operatorname{conv}(\operatorname{ext}(S_{\mathbb{Z}}))$. *Then, F is a perfect face.*

Proof. From Lemma 3, the lineality space V of the tangent cone $\mathbf{T}(F)$ has dimension $\dim(F)$. Writing $\mathbf{T}(F) = C + V$ as in Lemma 3, where $C = \mathbf{T}(F) \cap V^\perp$ is a pointed cone, the normal cone

$$\mathbf{N}(F) = (C + V)^\circ = C^\circ \cap V^\perp \ .$$

As C is pointed, C° is full dimensional, and $\dim(\mathbf{N}(F)) = \dim(V^\perp) = n - \dim(F)$. □

Remark 2. Theorem 1 is sufficient to show that if $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ is noncompact, then $\mathrm{cl}(\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}})))$ has infinitely many facets, using polarity theory for noncompact sets [24]. Without loss of generality, $\mathrm{cl}(\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}})))$ is full dimensional, and therefore its polar $\mathrm{cl}(\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}})))^\circ$ is compact. The conjugate faces of exposed points of $\mathrm{cl}(\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}})))^\circ$ are closures of maximal faces of $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$, and therefore, perfect. Combining this with the fact that $\mathrm{cl}(\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}})))^\circ$ is the closed convex hull of its exposed points, we see that $\mathrm{cl}(\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}})))$ has infinitely many facets.

4 Isolation of Faces

Although perfect faces have several useful properties, there can be inclusionwise maximal perfect faces of low dimension.

Example 2. Let $C' \subseteq \mathbb{R}^2$ be the intersection of two discs with unit radius, and centres at $\mathbf{0}$ and $\mathbf{e}_2$. Then, the two points of intersection of the boundaries, $(\pm\sqrt{3}/2, 1/2)$, are maximal faces with two dimensional normal cones (i.e., they are maximal perfect faces), whose dimension is $0 < \dim(C') - 1$.

However, we will now show that faces of $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ are isolated:

Definition 4. *A face F of a convex set $C \subseteq \mathbb{R}^n$ is said to be isolated* [see 18] *if for every $x \in \mathrm{ri}(F)$, there exists a neighbourhood U_x of x such that if $y \in (U_x \cap C) \setminus F$, then $y \in \mathrm{ri}(G)$, where G is a face of C with $\dim(G) > \dim(F)$.*

Fedotov [18] shows that if for any one $x \in \mathrm{ri}(F)$ there exists a neighbourhood U_x satisfying the conditions of Definition 4, then F is an isolated face.

Theorem 2. *Let F be a face of $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$. Then, F is isolated.*

Proof. First, it is clear that if $\dim(F) = 0$, then F is isolated. Hence, we assume that F is not an extreme point. Let $z \in E_F$. Since $\{z\}$ can be strongly separated (Lemma 2) from $\widehat{E}_z$, there exists a convex set $Q \subsetneqq \mathrm{ri}(F)$ such that any $x \in Q$ written as a convex combination of points from E_F, must give a nonzero weight to z, i.e.,

$$x = \sum_{i=0}^{\dim(F)} \gamma_i w_i,\ (\gamma_0, \ldots, \gamma_{\dim(F)}) \in \Delta^{\dim F}, w_i \in E_F \Rightarrow w_i = z \text{ for some } i \in \{0, \ldots, \dim(F)\} \text{ with } \gamma_i > 0\ ,$$

for all $x \in Q$.

Let $\{x_k\}_{k\in\mathbb{N}}$ be a sequence with $x_k \in \mathrm{ri}(F_k)$, where F_k is a face of $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ for all $k \in \mathbb{N}$, with $x_k \to x \in Q$. Passing on to a subsequence if necessary, we assume that $\dim(F_k) = r$ for all $k \in \mathbb{N}$. For sufficiently large k, we may assume that $x_k = \alpha_k z + (1 - \alpha_k) y_k$, where $y_k \in \mathrm{ri}(G_k)$ for some face G_k of F_k. It is sufficient to show that $F_k \supseteq F$ for sufficiently large k. It is also clear that the sequence of line segments $[z, x_k]$ converge to $[z, x]$ in the Hausdorff metric.

Our proof is based on induction on r, the common dimension of the F_k's. The base case $r = 0$ is trivially ruled out as $\text{ext}(S_{\mathbb{Z}})$ is a closed set, and let $r \geq 1$. We consider the following two cases.

Case 1. There exists $\rho > 0$ with $\|z - y_k\| \leq \rho$ for all $k \in \mathbb{N}$. In this case, passing on to a subsequence, we see that $[z, y_k] \to [z, w]$ for some $w \in F$ (Blaschke selection theorem [33, Theorem 1.8.6]), or $y_k \to w \in F$. As $x \in \text{ri}(F)$, we can assume without loss of generality that $w \in \text{ri}(F)$ (using the "extension principle"[30, Theorem 6.4]). As y_k lies in $\text{ri}(G_k)$, a face of dimension less than r, we see by the induction hypothesis that $F_k \supseteq F$ for sufficiently large k.

Case 2. $\lim_{k\to+\infty} \|z - y_k\| = +\infty$. Now consider

$$x_k - z = \frac{y_k - z}{\|y_k - z\|} \left((1 - \alpha_k) \|y_k - z\| \right), k \in \mathbb{N} \ .$$

We know that $(x_k - z) \to (x - z) \neq \mathbf{0}$, and passing on to a subsequence, we may assume that $\lim_{k\to+\infty} \frac{y_k - z}{\|y_k - z\|} = u \in \text{rec}(\text{conv}(\text{ext}(S_{\mathbb{Z}})))$, with $\|u\| = 1$ [28, Lemma 1]. Hence, $\lim_{k\to+\infty} (1 - \alpha_k) \|y_k - z\| = \eta > 0$. Therefore, for any $\theta \geq 0$, we have

$$S \ni z + \theta u = z + \frac{\theta}{\eta}(x - z) \ ,$$

and $u \in \text{rec}(F)$.

Since F is not isolated at $x \in Q$, it is isolated at no point in Q. From the argument in the previous paragraph, $x - z \in \text{rec}(F)$ for all $x \in Q$. However, we know that $z + \text{cl cone}(Q - \{z\}) \supseteq F$, and therefore, $E_F = \{z\}$. However, we assumed that $\dim(F) > 0$, which gives the required contradiction. □

We now present a couple of examples that demonstrate the importance of studying $\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ as opposed to $S_{\mathbb{Z}}$ or $\text{cl}(\text{conv}(\text{ext}(S_{\mathbb{Z}})))$. The key difference turns out to be the isolation of faces.

Example 3 ($\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ *vs.* $\text{conv}(S \cap \mathbb{Z}^n)$*).* Let S be the Lorentz cone in $\mathbb{R}^n$. Then, $\text{conv}(S \cap \mathbb{Z}^n)$ is the cone generated by all rational rays of S, and $\text{conv}(\text{ext}(S_{\mathbb{Z}})) = \{\mathbf{0}\}$. As the rationals are dense on the unit sphere [32], we see that the faces of $\text{conv}(S \cap \mathbb{Z}^n)$ are not isolated.

Example 4 ($\text{conv}(\text{ext}(S_{\mathbb{Z}}))$ *vs* $\text{cl}(\text{conv}(\text{ext}(S_{\mathbb{Z}})))$*).* Let $S = \text{conv}(\{\mathbf{0}\} \cup T)$, where $T := \left\{(x_1, x_2, x_3) \in \mathbb{R}^3 \ : \ x_1 = 1, x_3 \geq x_2^2\right\}$. Then,

$$\text{conv}(\text{ext}(S_{\mathbb{Z}})) = \text{conv}\left(\{\mathbf{0}\} \cup \left\{(1, k, k^2) \ : \ k \in \mathbb{Z}\right\}\right) \subseteq \mathbb{R}^3 \ ,$$

and $F := \mathbb{R}_+ \mathbf{e}_3$ is a one dimensional face of $\text{cl}(\text{conv}(\text{ext}(S_{\mathbb{Z}})))$. However, $F \notin \text{conv}(\text{ext}(S_{\mathbb{Z}}))$, and it can be verified that the sequence of points $\left\{(1/k^2, 1/k, 1)\right\}_{k\in\mathbb{N}}$ converges to $\mathbf{e}_3 \in \text{ri}(F)$, and for each k,

$$\left(\frac{1}{k^2}, \frac{1}{k}, 1\right) = \left(1 - \frac{1}{k^2}\right) \mathbf{0} + \frac{1}{k^2}(1, k, k^2) \in \text{ri}[\mathbf{0}, (1, k, k^2)] \ ,$$

which are one dimensional faces of $\text{cl}(\text{conv}(\text{ext}(S_{\mathbb{Z}})))$. Hence, $\text{cl}(\text{conv}(\text{ext}(S_{\mathbb{Z}})))$ can have faces that are not isolated.

We now show that the situation of Example 2 does not arise in the case of our set $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$. Our proof uses Lemmas 4 and 5, which are presented after the proof of Theorem 3.

Theorem 3. *Any inclusionwise maximal isolated face of* $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ *is a facet (i.e., has dimension* $n-1$*).*

Proof. Let F be a face of $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$. Then, there exists an extreme ray u of $\mathbf{N}(F)$ such that

$$F_u := \left\{x \in \mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}})) \ : \ \langle u, x\rangle = \sigma_{\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))}(u)\right\}$$

is a nonempty exposed (and isolated, by Theorem 2) face of $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$. As u is an extreme ray of $\mathbf{N}(F)$, the face F_u is also inclusionwise maximal.

We first show that $u \in \mathrm{ri}\,\mathbf{N}(F_u)$. Otherwise by Lemma 5, there exists a sequence of normal vectors $\{v_i\}_{i\in\mathbb{N}} \subseteq K^\circ \setminus \mathrm{cl}\left(\cup_{z\in\mathrm{ext}(F_u)}\mathbf{N}(z)\right)$ with $v_i \in \mathbf{N}(x_i)$ for some $x_i \in \mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$, and $v_i \to v$. From the properties of v_i, we see that $x_i \notin F_u$. We may assume w.l.o.g. that $x_i \to \bar{x} \in F'$, where F' is a face of $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ with $\mathrm{ext}(F') \cap \mathrm{ext}(F_u) = \varnothing$, because $v_i \in \mathbf{N}(x_i)$, and if x_i lies on a face that shares an extreme point z with F_u, we have $\mathbf{N}(x_i) \subseteq \mathbf{N}(z)$.

Let $w \in \mathrm{ext}(F') \setminus \mathrm{ext}(F_u)$. As $\bar{x} \in F'$ and $v \in \mathbf{N}(x)$ [see 31, Proposition 6.6], we have $v \in \mathbf{N}(w)$, i.e., $\langle v, w\rangle = \sigma_{\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))}(v)$, or $w \in F_u$, which is a contradiction.

As $u \in \mathrm{ri}\,\mathbf{N}(F_u)$, and $\mathbb{R}_+u$ is a one dimensional face of $\mathbf{N}(F)$, we infer that $\mathbb{R}_+u = \mathbf{N}(F_u)$ from the facts for a convex set C: (a) if G, G' are distinct faces of C, then $\mathrm{ri}(G) \cap \mathrm{ri}(G') = \varnothing$, and (b) $\mathrm{ri}(\mathbf{N}_C(G))$ and $\mathrm{ri}(\mathbf{N}_C(G))$ are disjoint. □

Lemma 4. *Let* $C \subseteq \mathbb{R}^n$ *be a line free convex set with* $\mathrm{ext}(C)$ *closed, and* $C = \mathrm{conv}(\mathrm{ext}(C))$*. Then,*

$$\mathrm{cl}\left\{\bigcup_{F\lhd C} \mathbf{N}_C(F)\right\} = \mathrm{rec}(\mathrm{cl}\, C)^\circ \ .$$

Proof. Since C is line free, so is $\mathrm{cl}(C)$, and $K := \mathrm{rec}(\mathrm{cl}\, C)$ is pointed (and closed), and K° is full-dimensional. If K is trivial, then C is compact ($\mathrm{ext}(C)$ is closed), and the result is known [see 33, Section 2.2]. Otherwise, let $u \in \mathrm{int}(K^\circ)$, i.e., $\langle u, w\rangle < 0$ for all $w \in \mathrm{rec}(C)$. Then, we claim that u is a normal vector of C.

To show the claim, note first that $\sigma_C(u) < +\infty$ [23, Proposition C.2.2.4]. Let $\{x_k\}_{k\in\mathbb{N}}$ be a sequence in C with $\langle u, x_k\rangle \nearrow \sigma_C(u) = \sigma_{\mathrm{cl}(C)}(u)$. If a subsequence is contained in a compact set in $\mathrm{cl}(C)$, there exists $x_u \in \mathrm{cl}(C)$ with $\langle u, x_u\rangle = \sigma_C(u)$. Else, let $\lim_{k\to+\infty} \|x_k\| = +\infty$. Then, we may assume without loss of generality that $x_k/\|x_k\| \to w \in K$ [28, Lemma 1]. For any $\varepsilon > 0$, we have $\langle u, x_k\rangle \geq \sigma_C(u) - \varepsilon$ for sufficiently large k. Dividing by $\|x_k\|$ and letting $k \to +\infty$, we see that $\langle u, w\rangle \geq 0$, which contradicts $u \in \mathrm{int}(K^\circ)$. Therefore, there exists an extreme point z of $\mathrm{cl}(C)$ such that $u \in \mathbf{N}_C(z)$, or, $\langle u, x - z\rangle \leq 0$ for all $x \in C$. As $z \in \mathrm{ext}(C)$ [25], u is a normal vector for C.

If $u \notin K^\circ$, then there exists some $w \in \mathrm{rec}(C)$ with $\langle u, w\rangle > 0$, and hence, u cannot be a normal vector to C. □

Lemma 5. *Let F be face of a line free convex set $C \subseteq \mathbb{R}^n$, all whose faces are isolated, and let $\{v_i\}_{i\in\mathbb{N}}$ be a sequence of normal vectors of C with $v_i \to v \in \mathrm{ri}(\mathbf{N}_C(F)) \setminus \{\mathbf{0}\}$. Then, there exists some $i_0 \in \mathbb{N}$ such that $v_i \in \cup_{x\in\mathrm{ext}(F)}\mathbf{N}_C(x)$ for all $i \geq i_0$.*

Proof. Applying Lemma 4 to F, we see that $v \in \mathrm{ri}(\mathbf{N}_C(F))$ if and only if $v \in \mathrm{ri}\left(\cup_{x\in\mathrm{ext}(F)}\mathbf{N}_C(x)\right) =: N$. The latter set is full dimensional as extreme points have full dimensional normal cones (as they are perfect). Hence, any sequence of normal vectors $v_i \to v$ have to pass through N. □

5 Semidefinite Representations

First, we present a result on the number of facets of $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$, which is crucial for semidefinite representation of $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$.

Theorem 4. *If* $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$ *is unbounded, then* $\mathrm{cl}(\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z})))$ *has infinitely many facets.*

Proof. It is sufficient to show that $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$ has infinitely many facets. We use induction on $\dim(\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z})))$. The result is trivial if $\dim(\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))) < 2$. If $\dim(\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))) = 2$, all maximal facets of $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$ have dimension one. Note that any one dimensional face of $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$ is compact as it is the convex hull of two extreme points. Therefore, if $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$ is unbounded, it has infinitely many facets when $\dim(\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))) = 2$.

Now assume that the result is true when the dimension of $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$ is at most k. If $\dim(\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))) = k+1$, and each facet of $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$ contains only a finite number of extreme points, then we are done. Else, there exists a facet F of $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$ containing infinitely many extreme points. Therefore, applying the induction hypothesis to F, there exist infinitely many faces G_i, $i \in \mathbb{N}$ of $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$ of dimension $k-1$. Let $v \in \mathbf{N}(F)$ be the normal of F (relative to the affine hull of $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$). Let $\mathbf{N}(G_i) = \mathrm{cone}\{v, v_i\}$ for some distinct normal vectors v_i (note that by Theorem 1, $\dim(\mathbf{N}(G_i)) = 2$). Then, for every $i \in \mathbb{N}$, there exists a sequence $\{x_{ij}\}_{j\in\mathbb{N}} \subseteq \mathrm{conv}(\mathrm{ext}(S_\mathbb{Z})) \setminus F$ with $v_{ij} \in \mathbf{N}(x_{ij})$ such that $\lim_{j\to+\infty} x_{ij} = x_i \in \mathrm{ri}(G_i)$, and $\lim_{j\to+\infty} v_{ij} = v_i$ (e.g., apply [31, Ex. 6.18]). Since $x_{ij} \notin F \supsetneqq G_i$, x_{ij} must lie in some face F_i of $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$ strictly containing G_i (as G_i is isolated), but different from F. Clearly, F_i is a facet of $\mathrm{conv}(\mathrm{ext}(S_\mathbb{Z}))$.

Finally, we show that for $i, k \in \mathbb{N}$, $i \neq k$, the only facet containing both G_i and G_k is F. Let Z_i be an affinely independent set of k points from G_i, and let $w \in G_k \setminus G_i$. Then, $\mathrm{ri}(\mathrm{conv}(Z_i \cup \{w\})) \cap \mathrm{ri}(F) \neq \varnothing$, which implies that $\mathrm{conv}(G_i \cup G_k) \subseteq F$, thus completing the proof. □

Note that the converse to Theorem 4 is true as well. We now get to the main result of this section.

Theorem 5. *Let $S \subseteq \mathbb{R}^n$ be a closed, convex semidefinite representable set. Then,* $\mathrm{cl}(\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}})))$ *is semidefinite representable if and only if* $\mathrm{ext}(S_{\mathbb{Z}})$ *is compact.*

For a proof, we require a (modified form of a) result of Grötschel and Henk [20]:

Proposition 3 ([20, Proposition 2.1]). *Let $C \subseteq \mathbb{R}^n$ be a convex semialgebraic set defined by a finite number of polynomials $f_1, \ldots, f_\ell$, $\ell \in \mathbb{N}$. If $\langle a, x\rangle \leq \alpha$ is a facet-defining inequality for C, then the linear polynomial $\alpha - \langle a, x\rangle$ is a factor of one of the f_i's.*

The proof is nearly the same as that of Grötschel and Henk [20], and is omitted.

Proof (Theorem 5). We know that $\mathrm{ext}(S_{\mathbb{Z}})$ is closed, and hence, it is compact if and only if it is bounded, or $|\mathrm{ext}(S_{\mathbb{Z}})| < +\infty$. In this case, $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ is a polytope, and is semidefinite representable.

On the other hand, if $\mathrm{ext}(S_{\mathbb{Z}})$ is unbounded, $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ has infinitely many facets $F_i = \left\{x \in \mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}})) \ : \ \langle u_i, x\rangle = \sigma_{\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))}(u_i)\right\}$, $i \in \mathbb{N}$, by Theorem 4. If $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ is semialgebraic, it is defined by a finite number of polynomials $f_1, \ldots, f_\ell$, and each linear polynomial $\sigma_{\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))}(u_i) - \langle u_i, x\rangle$ is a factor of one of the f_j's. The facet-defining inequalities are distinct, and there are only a finite number of polynomials, each having a finite degree. Since there are infinitely many facet-defining linear polynomials, this is impossible, and thus, $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ cannot be semialgebraic, and thus, is not semidefinite representable. □

6 Closing Remarks

In this paper, we studied the facial structure of integer hulls of convex sets by means of the set $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$. We showed that although $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ is not a polyhedron, it shares several properties with polyhedra: the extreme points are closed, all faces are perfect and isolated, and maximal faces are facets. We end with a couple of open questions:

1. What are the facial properties of $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ in the presence of continuous variables? It is possible to give some sufficiency conditions as corollaries of the results in this paper, but they turn out to be weak.
2. Can one give sufficient conditions on S such that $\mathrm{conv}(\mathrm{ext}(S_{\mathbb{Z}}))$ only has a finite number of extreme points (facets)? If so, questions of representation etc., can be addressed more easily.

Acknowledgements. The author wishes to thank Santanu S. Dey and K. S. Mallikarjuna Rao for several discussions that led to the results presented in this paper. The author also thanks an anonymous referee for pointing out reference [17].

References

1. Abhishek, K., Leyffer, S., Linderoth, J.: FilMINT: An outer-approximation-based solver for nonlinear mixed-integer programs. Preprint, Argonne National Laboratory, Mathematics and Computer Science Division, Argonne, IL (2006)
2. Atamtürk, A., Narayanan, V.: The submodular knapsack polytope. Discrete Optimization 6, 333–344 (2009)
3. Atamtürk, A., Narayanan, V.: Conic mixed-integer rounding cuts. Mathematical Programming 122, 1–20 (2010)
4. Atamtürk, A., Narayanan, V.: Lifting for conic mixed-integer programming. Mathematical Programming 126, 351–363 (2011)
5. Belotti, P., Lee, J., Libreti, L., Margot, F., Waechter, A.: Branching and bound tightening techniques for non-convex MINLP. Optimization Methods and Software 24, 597–634 (2009)
6. Bonami, P., Biegler, L.T., Conn, A.R., Cornuéjols, G., Grossmann, I.E., Laird, C.D., Lee, J., Lodi, A., Margot, F., Sawaya, N., Waechter, A.: An algorithmic framework for convex mixed-integer nonlinear programs. Discrete Optimization 5, 186–204 (2008)
7. Bonami, P., Kilinç, M., Linderoth, J.: Algorithms and software for convex mixed integer nonlinear programs. In: Lee, J., Leyffer, S. (eds.) Mixed Integer Nonlinear Programming. IMA Volumes in Mathematics and its Applications, vol. 154, pp. 1–39. Springer, New York (2012)
8. Braun, G., Pokutta, S.: A short proof of the polyhedrality of the chvátal-gomory closure of a compact set (2012)
9. Burer, S., Letchford, A.: Unbounded convex sets for non-convex mixed-integer quadratic programming. Mathematical Programming (2012) (to appear)
10. Cassels, J.W.: An Introduction to the Geometry of Numbers. Springer (1997)
11. Çezik, M., Iyengar, G.: Cuts for mixed 0–1 conic programming. Mathematical Programming 104, 179–202 (2005)
12. Dadush, D., Dey, S., Vielma, J.: The Chvátal-Gomory closure of a strictly convex body. Mathematics of Operations Research 36, 227–239 (2011a)
13. Dadush, D., Dey, S., Vielma, J.: The split closure of a strictly convex body. Operations Research Letters 39, 121–126 (2011b)
14. Dadush, D., Dey, S., Vielma, J.: On the Chvátal-Gomory closure of a compact convex set (2012), arXiv:1011.1710v1 (math.OC)
15. Dey, S., Morán R., D.A.: Some properties of convex hulls of integer points contained in general convex sets. Mathematical Programming Online First, doi: 10.1007/s10107-012-0538-7
16. Dey, S.S., Vielma, J.P.: The Chvátal-Gomory Closure of an Ellipsoid Is a Polyhedron. In: Eisenbrand, F., Shepherd, F.B. (eds.) IPCO 2010. LNCS, vol. 6080, pp. 327–340. Springer, Heidelberg (2010)
17. Dunkel, J., Schulz, A.: The Gomory-Chvátal closure of a non-rational polytope is a rational polytope. Mathematics of Operations Research (to appear), doi:10.1287/moor.1120.0565
18. Fedotov, V.P.: Isolated faces of a convex compactum. Mathematicheskie Zametki 25(1), 139–147 (1979)
19. Frangioni, A., Gentile, C.: Perspective cuts for a class of 0–1 mixed integer programs. Mathematical Programming 106, 225–236 (2006)
20. Grötschel, M., Henk, M.: The representation of polyhedra by polynomial inequalities. Discrete and Comuputational Geometry 29, 485–504 (2003)

21. Günlük, O., Lee, J., Weismantel, R.: MINLP strengthening for separable convex quadratic transportation-cost UFL. IBM Research Report RC24213, IBM, Yorktown Heights, NY (March 2007)
22. Günlük, O., Linderoth, J.: Perspective relaxation of mixed integer nonlinear programs with indicator variables. Mathematical Programming, Series B 104, 183–206 (2010)
23. Hiriart-Urruty, J., Lemaréchal, C.: Fundamentals of Convex Analysis. Grundlehren Text edn. Springer Heidelberg (2001)
24. Jaume, D.A., Puente, R.: Conjugacy for closed convex sets. Contributions to Algebra and Geometry 46(1), 131–149 (2005)
25. Klee, V.: Extremal structure of convex sets. Archiv der Mathematik 8, 234–240 (1957)
26. Meyer, R.: On the existence of optimal solutions to integer and mixed-integer programming problems. Mathematical Programming 7, 223–235 (1974)
27. Morán R., D.A., Dey, S.S., Vielma, J.P.: A strong dual for conic mixed-integer programs. SIAM Journal on Optimization 22, 1136–1150 (2012)
28. Moussafir, J.O.: Convex hulls of integral points. Journal of Mathematical Sciences 115(5), 647–665 (2003)
29. Pataki, G.: On the closedness of the linear image of a closed convex cone. Mathematics of Operations Research 32(2), 395–412 (2007)
30. Rockafellar, R.T.: Convex Analysis. Princeton Landmarks in Mathematics. Princeton University Press, Princeton (1970)
31. Rockafellar, R.T., Wets, R.J.B.: Variational Analysis. Springer, Berlin (2004)
32. Schmutz, E.: Rational points on the unit sphere. Central European Journal of Mathematics 6(3), 482–487 (2008)
33. Schneider, R.: Convex Bodies: The Brunn-Minkowski Theory. Cambridge University Press (1993)
34. Stubbs, R., Mehrotra, S.: A branch-and-cut method for 0-1 mixed convex programming. Mathematical Programming 86, 515–532 (1999)

An Efficient Polynomial-Time Approximation Scheme for the Joint Replenishment Problem

Tim Nonner[1] and Maxim Sviridenko[2,⋆]

[1] IBM Research - Zurich
tno@zurich.ibm.com
[2] University of Warwick
M.I.Sviridenko@warwick.ac.uk

Abstract. We give an efficient polynomial-time approximation scheme (EPTAS) for the Joint Replenishment Problem (JRP) with stationary demand. Moreover, using a similar technique, we give a PTAS for the capacitated JRP with non-stationary demand but constant size capacities.

1 Introduction

The joint replenishment problem (JRP) with stationary demand is one of the fundamental problem in inventory management, dating back to a paper of Naddor and Saltzman [10] but probably even further. It it arguably the simplest extension of the even more prominent problems of finding the economic order quantity (EOQ) for a single item [5], that is, given stationary demand, a fixed holding cost per unit, and a fixed cost for a single order, the order quantity that optimally balances holding and ordering cost over time. JRP also aims to optimally balance holding and ordering cost, but for the case of multiple items with a common ordering cost, modeling the fact that production or transportation processes often require some setup cost which is independent of how much and what is produced or ordered, respectively. However, in contrast to the single item case where a simple EOQ-policy is optimal, an optimal replenishment policy for JRP might have a complicated non-stationary structure with changing inter-replenishment times [1,9], motivating the use of restricted policies. Roundy [12] showed in a seminal paper that policies with power-of-two ratios between inter-replenishment times yield an 1.02-approximation, see also [9] for an overview and [6,16] for some improvements and extensions. Power-of-two policies can be relaxed by allowing arbitrary periodic inter-replenishment times. In this case, finding an optimal policy is at least as hard as factoring, as recently pointed out by Schulz and Telha [13], and hence it is unlikely to find a polynomial-time algorithm.

All these policies are quite restrictive since they require periodicity of all replenishment cycles, leading to sub-optimal solutions. A reasonable trade-off

⋆ Work supported by EPSRC grant EP/J021814/1, FP7 Marie Curie Career Integration Grant and Royal Society Wolfson Merit Award.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 314–323, 2013.

is to periodically repeat a non-periodic policy for a finite time horizon. Indeed, Adelman and Klabjan [1] showed that such policies offer an $1+\epsilon$-approximation for an arbitrary small $\epsilon > 0$. Once restricted to a finite time horizon, it is natural to partition this horizon into equal-length periods, moving into the realm of combinatorial optimization. Quite recently, Segev made the first major progress since years for this case by presenting a QPTAS [14], which shows that it is probably not APX-hard, and thus motivating the search for other approximation schemes.

In contrast to the infinite-horizon case where relatively little is known about the complexity [13], Arkin, Joneja, and Roundy [3] showed that the finite-horizon case is strongly NP-hard for non-stationary demand. On the other hand, there has been a line of approximation results for this case during the last decade [7,8,11,15], even in the more general setting of the one-warehouse multiple-retailer problem with non-linear holding cost. Nonner and Souza [11] showed that this setting is APX-hard and therefore does not admit a polynomial-time approximation scheme (PTAS), that is, an algorithm that has performance guarantee $1+\varepsilon$ and polynomial running time for any $\varepsilon > 0$. On the other side, the basic question whether JRP with non-stationary demand and linear holding cost is APX-hard (and hence does not admit any approximation scheme, unless P=NP) remains open.

Constributions. Our main contribution is an efficient polynomial-time approximation scheme (EPTAS) for JRP with finite time horizon and stationary demand. Specifically, for any $\epsilon > 0$, we design an $1+\varepsilon$-approximation algorithm with running time $\mathcal{O}(\lambda 2^{\lambda} 2^{2^{\lambda}}(NT^2+T^5))$ for $\lambda = \frac{4\sqrt{2}}{\epsilon}$, where T is the number of periods of the finite time horizon, and N is the number of items. Note here our assumption that the input T is polynomial, which is reasonable once we move to the finite time horizon case. Using a similar technique, we also give a PTAS for the non-stationary demand case with soft capacitated single item orders, that is, for each item i, there is a constant C_i such that ordering y units results in item ordering cost $\left\lceil \frac{y}{C_i} \right\rceil K_i$, where K_i is the cost of a single order. This makes especially sense from a practical point of view, since item orders might be delivered in small batches of size C_i, each implying cost K_i, for example with trucks having limited capacity, but common orders are delivered in huge quantities with comparably small capacity constraints, for example with container ships. It is worth mentioning here that this case is still strongly NP-hard, which is a simple consequence of the fact that JRP with non-stationary demand is NP-hard even if each item faces only three times exactly demand 1 [11]. Consequently, to the best of our knowledge, this is the first approximation scheme for a natural NP-hard variant of JRP with non-stationary demand.

Preliminaries. Throughout this paper, we consider the case of a finite time horizon partitioned into periods $1, 2, \ldots, T$ of equal length, w.l.o.g. say 1. We label the items $1, 2, \ldots, N$, and let integers $d_{it} \geq 0$ denote the demand for item i in time period t. Hence, in case of stationary demand, the d_{it} are equal for each item i, denoted d_i. We assume that demand arrives at the end of the

corresponding time period, but we are allowed to order at the beginning of each time period. Therefore, stock is held in inventory for at least one period. Each order implies a common ordering cost K_0, independent of the number of involved items and the size of the order. However, for each involved item i, we obtain an additional item ordering cost K_i, which is also independent of the size of the order. We also consider the case that item ordering cost are size-dependent such that ordering y units results in cost $\left\lceil \frac{y}{C_i} \right\rceil K_i$ for some constant C_i, called *soft capacitated case*. Holding one unit of item i for one period results in holding cost h_i. Demand needs to be satisfied with items on stock, and hence no backlogging is allowed, called *make-to-stock scenario*. Consequently, since we may w.l.o.g. assume that there is at least one item with demand in the first period, there needs to be a common order in the first period. The objective is to find an optimal ordering schedule σ, that is, a schedule that minimizes $\text{cost}(\sigma)$, the sum of ordering and holding costs of all items. Let σ^* be an optimal schedule with $\text{cost}(\sigma^*) = \text{OPT}$. Observe that for each item i , we may assume that the time horizon $1, 2, \ldots, T$ is partitioned into *order intervals*. Specifically, each order interval has the form $\{a, \ldots, b\}$ with starting and ending periods a and b, respectively, and $\sum_{t=a}^{b} d_{it}$ units of item i are ordered in period a in order to satisfy the demand *between* periods a and b, that is $d_{i,a}, d_{i,a+1}, \ldots, d_{i,b}$. Since we assume that all periods have length 1, we obtain that this order interval has length $x = b - a + 1$, that is, the maximal number of periods stock is held in inventory in this order interval. Consequently, observe that if we are in the stationary demand case with demand d_i per period, then the holding cost associated with the order in period a is exactly $\frac{x(x+1)}{2} d_i h_i$. Because this definition of order intervals ensures that the stock of item i will be empty at the start of each order interval, this is called zero-inventory ordering policy (ZIO). It is clear that in our setting there is an optimal ZIO policy. Finally, for an interval of periods $I = \{a, \ldots, b\}$ and some period t, we write $t \geq I$ if and only of $t \geq a$, and $t > I$ if and only if $t > b$. For simplicity, we sometimes do not distinguish between an order and the period it is executed.

Basic Techniques. One way to solve JRP is to enumerate all possible sets of common orders $W \subseteq \{1, \ldots, T\}$. For each such set W and item i, the optimal orders for item i can then be found by solving a single-item lot-sizing problem [4,2] with the constraint that only periods from the set W may be used as orders. Therefore, the running time of this algorithm is $2^T N$ times the running time to solve the single-item lot-sizing problem, and thus exponential in the input T. An alternative way to solve JPR is to build a dynamic programming table, where each entry considers a subproblem defined by an interval of periods $\{a, \ldots, b\}$, i.e., this subproblem contains all items i and their demands between periods a and b. To fill such a table, we could decompose this interval into two subintervals $\{a, \ldots, t\}$ and $\{t + 1, \ldots, b\}$ at some period $a \leq t < b$ to define a recursion. However, this is not a valid decomposition, since we need to know for each item i the stock level at the end of period t, we also say that this stock is *held* to period $t+1$. Since this stock level could be as large as $D := \max_i \sum_t d_{it}$,

trying all possible stock level patterns results in a factor D^N, making such an approach infeasible. For the case of stationary demand, Segev [14] used such an approach to obtain a QPTAS. However, it is worth mentioning here that the way he trades the problem size for accuracy is completely different from ours. Specifically, he trades the number of items for accuracy, whereas we basically round the positions of item orders. Therefore, both techniques could also be applied in sequence.

Outline. We combine both approaches explained above. Specifically, we do several recursions as in the DP, and then we switch to an enumeration approach. The point when we switch needs to be individually triggered for each item. Hence, the problem is to incorporate all this into a single DP. To this end, we need the shifting procedure explained in Section 2, which builds a hierarchical random tree decomposition of the time horizon. Next, we utilize this decomposition in the main DP described in Section 3. The major bottleneck is its recurrence relation, which would only yield a PTAS if implemented straightforward, as explained in Section 4. To avoid this, we present an approximate recurrence relation in Section 5 that is based on a different helper DP. Then, in Section 6, we show that combining this approximate recurrence relation with the main DP from Section 3 gives an EPTAS for JRP with stationary demand. Finally, in Section 7, we conclude with an adaption of the PTAS to the case of non-stationary demand but soft capacitated item orders. Note that we could simplify the DP in both cases, but for the sake of exposition, we think that it is more convenient to present a general DP that covers everything.

2 Tree Decomposition of Time Horizon

We are given a constant integer λ, which we will define later on, but the goal is that the approximation ratio goes to 1 as $\lambda \to \infty$. Let then κ be a random integer drawn uniformly at random from $\{1, \dots, \lambda\}$. Moreover, assume that the number of periods T is a power of 2. This assumption simplifies the description of the DP. In general, to avoid this assumption, we could also partition the time horizon into intervals of roughly the same length instead of exactly the same length, as explained in the following paragraph.

We now inductively construct a random tree G whose nodes are intervals of periods with the property that an interval $I' \in G$ is a successor (or child) of another interval $I \in G$ if and only if $I' \subseteq I$. To start this inductive construction, let $I = \{1, \dots, T\}$ be the root of G. We then obtain the children of I by partitioning I into 2^κ many subintervals of equal length $T/2^\kappa$. Next, we partition these children into 2^λ many subintervals of equal length $T/2^{\kappa+\lambda}$, and so on. This clearly defines the tree of intervals G. It might happen that some intervals I are not large enough to be partitioned into 2^λ subintervals. In this case, we partition I into intervals each containing a single period, which will be the leaves of G. Let l_I denote the level of an interval $I \in G$, where the root $\{1, \dots, T\}$ has level 0. Note that if $l_I \geq 1$ and I is not a leaf, then the length $|I|$ of I is exactly

$T/2^{\lambda(l_I-1)+\kappa}$. For each intervals I, let c_I denote the number of children of I in G if there are any. We have that $c_I \leq 2^\lambda = \mathcal{O}(1)$. Finally, observe that $|G| = \mathcal{O}(T)$, independent of λ.

In the following sections, we use different *base values* χ_i for each item i which characterize the properties of item i in a single number with respect to K_i, h_i, and its demand. In addition, we define a *level* l_i for each item i. Specifically, if $\chi_i > T/2$, then we define $l_i := 0$. Otherwise, let l_i be the maximal level such that the lengths of all intervals I with $l_I = l_i$ are at least as large as χ_i, i.e., $\chi_i \leq T/2^{\lambda(l_i-1)+\kappa}$ if $l_i > 0$. Note that the length of the intervals I with $l_I = l_i+1$ are then strictly smaller than χ_i. However, since κ is random, we do not know how much larger and smaller the intervals at levels l_i and l_i+1 are, respectively. We need this to deal with arbitrary base values χ_i. We obtain the following simple lemma (proof in full version).

Lemma 1. *It holds that*

(1) for any item i, the expected length of each interval I with $l_I = l_i + 1$ is at most $\frac{2\chi_i}{\lambda}$,
(2) for any item i, the expected number of intervals I with $l_I = l_i$ is at most $1 + \frac{2T}{\chi_i \lambda}$.

3 Dynamic Program

In this section, we describe a DP which can be adapted to the case of stationary demand as well as the case of non-stationary demand with soft capacitated item orders. To this end, we restrict the search space to canonical schedules, where the properties of a *canonical* schedule σ are that

(1) for any item i and interval $I \in G$ with $l_I = l_i$, no stock of item i is held to the period r of the first common order in interval I, that is, the stock of item i is empty after period $r-1$,
(2) for any item i and interval $I \in G$ with $l_I = l_i + 1$, if item i orders in interval I, then the period of this item order is the period of the first common order in this interval.

To compute an optimal canonical schedule, we have a DP array Π with entries $\Pi(I, r, s)$, where I is a non-leaf interval in the tree G defined in Section 2, and r and s are periods with $r \leq s$, $r \geq I$, and $s > I$. Recall that we write $r \geq I = \{a, \ldots, b\}$ if and only of $r \geq a$, and $r > I$ if and only if $r > b$. For technical reasons, we moreover allow that $r = s = T + 1$. Note that $T + 1$ is not officially a period, since the last period is T. The size of Π is hence $\mathcal{O}(T^3)$ since $|G| = \mathcal{O}(T)$, and therefore polynomial. We will first explain how to fill Π without further explanations, and then show some properties of the entries of Π. Array Π is initialized as follows:

(1) We set $\Pi(I, r, s) = 0$ for $r > I$.

(2) We set $\Pi(I,r,s) = K_0 + \sum_i \Phi_i(r,s)$ for each leaf interval $I = \{a\} \in G$ and $r = a$, where $\Phi_i(r,s)$ is the cost of an optimal schedule for the subproblem consisting of item i and its demand between periods r and $s-1$, that is, $d_{i,r}, d_{i,r+1}, \ldots, d_{i,s-1}$, subject to the constraint that there can only be an item order in period r. Hence, if there is demand between these periods, then there needs to be such an order, and otherwise not.

To define the recurrence relation, consider some fixed entry $\Pi(I,r,s)$. We may assume that $r \leq b$, because the initialization makes the case $r > b$ trivial. Moreover, let $I_1, I_2, \ldots, I_{c_I}$ be the natural ordering of the children of I in G. Let then $E(I,r,s)$ denote the set of all period sequences $e_1 = r \leq e_1 \leq \ldots \leq e_{c_I+1} = s$ such that for each $1 \leq z \leq c_I$, $e_z \geq I_z$, and if even $e_z > I_z$, then $e_z = e_{z+1}$. For each such sequence $e \in E(I,r,s)$ and item i, let $\Phi_i(I,e)$ be the cost of an optimal schedule for the subproblem which consists only of item i and its demands between periods r and $s-1$ with the constraint that all item orders need to be selected from the set of periods $\{e_1, e_2, \ldots, e_{c_I}\}$. Finding this schedule is basically a single-item lot-sizing problem which can be solved in polynomial time. For instance, with the classical algorithm of Federgruen and Tzur [4] in $\mathcal{O}(c_I \log c_I)$ time, or even in $\mathcal{O}(c_I)$ time with an algorithm from Aggarwal, Alok, and Park [2]. Using these definition, we are ready to state the recurrence relation:

$$\Pi(I,r,s) = \min_{e \in E(I,r,s)} \left\{ \sum_{i:l_i=l_I} \Phi_i(I,e) + \sum_{z=1}^{c_I} \Pi(I_z, e_z, e_{z+1}) \right\} \tag{1}$$

Since $c_I = \mathcal{O}(1)$, we have the simple polynomial upper bound T^{c_I-1} on the number of sequences in $E(I,r,s)$. This shows that this recurrence relation can be implemented in polynomial time. Specifically, because c_I might be as large as 2^λ and we need to solve at most N single-item lot-sizing problems with c_I periods, we obtain the following lemma.

Lemma 2. *The recurrence relation can be solved in $\mathcal{O}(T^{2^\lambda-1}N2^\lambda)$ time, and hence the complete array Π can be filled in polynomial time.*

However, the exponent of the running time given in Lemma 2 contains λ, a parameter which we will need to set with respect to the required precision ϵ later on. Therefore, using this recurrence relation can only yield a PTAS. To avoid this drawback for the case of stationary demand, we will present an approximate recurrence relation which can be implemented more efficiently in the next Section 5. The following lemma is a direct consequence of the definition of the recurrence relation and states the required properties.

Lemma 3. *Array Π gets filled such that each entry $\Pi(I,r,s)$ with $r \in I$ satisfies $\Pi(I,r,s) = \text{cost}(\sigma)$, where σ is an optimal canonical schedule for the subproblem consisting of all items i with $l_i \geq l_I$ and their demands between periods r and $s-1$ subject to the constraint that only orders in I are allowed.*

Consider now the realization σ of the entry $\Pi(I,r,s)$ for $I = \{1,\ldots,T\}$, $r = 1$, and $s = T+1$. By Lemma 3 we derive that σ is an optimal canonical schedule

for the complete instance. Recall here the initial assumption that there needs to be a common order in the first period r. We conclude that we need to argue in the following Sections 4 and 7 that there is a feasible canonical schedule which is ϵ-close to an optimal one.

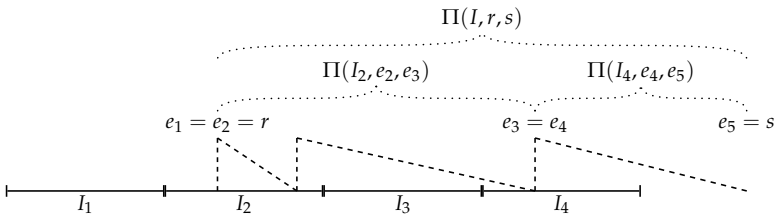


Fig. 1. Example recurrence relation

Example. To illustrate the recurrence relation, let us consider a simple example for an interval I with four children I_1, I_2, I_3, I_4. Moreover, assume that the optimal sequence $e_1 \leq e_1 \leq \ldots \leq e_5$ satisfies the properties $e_3 = e_4 \in I_4$ and $e_1 = e_2 \in I_2$, as depicted in Figure 3. Therefore, since consequently $\Pi(I_1, e_1, e_2) = \Pi(I_3, e_3, e_4) = 0$, the right part of the recurrence relation is simply $\Pi(I_2, e_2, e_3) + \Pi(I_4, e_4, e_5)$, which translates to the cost added by the common orders and the items i with $l_i > l_I$. Since then $l_i \geq l_{I_2} = l_{I_4}$, Property (1) in the definition of a canonical schedule says that no demand is held to periods e_2 and e_4 at any such item i. Therefore, periods e_2 and e_4 decompose the instance for such items as schematically depicted in Figure 3. The dashed lines schematically depict one possible final realization. Observe that we are not allowed to do any ordering in interval I_3, but there might be common orders in I_2 and I_4 except e_2 and e_4, respectively. On the other hand, the left part of the recurrence relation translates to the cost added by the items i with $l_i = l_I$. To satisfy Property (2) in the definition of a canonical schedule, we are only allowed to pick item orders with periods from the set or periods $\{e_2, e_4\}$, which is exactly what we do in an optimal way by the definition of $\Phi_i(I, e)$.

4 PTAS for Stationary Demand

In this section, we show that the DP from Section 3 yields a PTAS for the stationary demand case when using base values $\chi_i := \sqrt{\frac{K_i}{d_i h_i}}$. We need the following lemma (proof in full version).

Lemma 4. *There is a canonical schedule σ with $\mathbb{E}[\text{cost}(\sigma)] \leq (1 + \frac{2\sqrt{2}}{\lambda})\text{OPT}$.*

Theorem 1. *There exists a polynomial-time approximation scheme for the Joint Replenishment Problem with stationary demand.*

Proof. We know from Lemma 4 that there is a canonical schedule σ with $\mathbb{E}[\text{cost}(\sigma)] \leq (1+\epsilon)\text{OPT}$ for $\lambda = 2\sqrt{2}/\epsilon$. On the other hand, Lemma 3 implies that our dynamic programming algorithm yields an optimal canonical schedule, which can be done in polynomial time because of Lemma 2 for any constant λ. The claim follows by combining both facts. Note that this construction can be derandomized by enumerating all possible values κ in the construction of tree G. □

5 Approximate Recurrence Relation

We can think of a schedule as a set of common orders $W \subseteq \{1,\dots,T\}$, and for each item i, a set of item orders $R_i \subseteq W$. The critical property is that the item orders are subsets of the common orders. We slightly modify this to generate canonical pseudo-schedules. Specifically, a *canonical pseudo-schedule* is defined by a set of common orders $W \subseteq \{1,\dots,T\}$ and for each item i, a set of item orders R_i, such that Property (1) from the definition of a canonical schedule is satisfied, and additionally the property that (2) for any item i and interval $I \in G$ with $l_I = l_i + 1$, if $W \cap I \neq \emptyset$ (i.e., there is a common order in interval I) then R_i contains exactly the first period in I. Hence, we do not have a strict subset relation, and therefore, in contrast to a canonical schedule, a canonical pseudo-schedule is not feasible. However, we consider canonical pseudo-schedules as a helper construction.

Our goal is to modify the DP from Section 3 such that it computes an optimal canonical pseudo-schedule using an approximate recurrence relation. To this end, consider a fixed entry $\Pi(I,r,s)$ which we want to fill, and let $I_1, I_2, \dots, I_{c_I}$ be the children of I in G. Then, for some sequence $e \in E(I,r,s)$, let $\overline{e} \in E(I,r,s)$ be the sequence with the property that for each $1 \leq z \leq c_I$, $\overline{e}_z$ is the first period in interval $I_{z'}$ such that $e_z \in I_{z'}$. Intuitively, we can think of $\overline{e}$ as a *rounding* of sequence e to first periods in intervals $I_1, I_2, \dots, I_{c_I}$. Let $\overline{E}(I,r,s) \subseteq E(I,r,s)$ denote the subset of all such rounded sequences. Replacing $\Phi_i(I,e)$ by $\Phi_i(I,\overline{e})$ in recurrence relation (1) gives us a new recurrence relation, which we call *approximate recurrence relation*. The following adaption of Lemma 3 is an immediate consequence of the rounding of sequences.

Lemma 5. *Using the approximate recurrence relation, array Π gets filled such that each entry $\Pi(I,r,s)$ with $r \in I$ satisfies $\Pi(I,r,s) = \text{cost}(\overline{\sigma})$, where $\overline{\sigma}$ is an optimal canonical pseudo-schedule for the subproblem consisting of all items i with $l_i \geq l_I$ and their demands between periods r and $s-1$ subject to the constraint that only orders in I are allowed.*

We will argue that the approximate recurrence relation can be implemented much more efficiently because there are far less sequences in $\overline{E}(I,r,s)$ than in $E(I,r,s)$. Specifically, the number drops from at most T^{c_I-1} to 2^{c_I-1}, which implies the following lemma (proof in full version).

Lemma 6. *The approximate recurrence relation can be solved in $\mathcal{O}(2^\lambda 2^{2^\lambda}(N+T^2))$ time.*

Lemma 7. *Array Π in combination with the approximate recurrence relation can be filled in $\mathcal{O}(2^\lambda 2^{2^\lambda}(NT^2+T^5))$ time.*

Proof. The claim follows by multiplying the time needed to solve the approximate recurrence relation given in Lemma 6 with the size of Π. Note that this would give running time $\mathcal{O}(2^\lambda 2^{2^\lambda}(N+T^2)T^3)$. However, since for each item i, the value $\Phi_i(I,\overline{e})$ needs to be computed only in recurrence relations which fill entries of the form $\Pi(I,r,s)$ with $l_I = l_i$, we obtain running time $\mathcal{O}(2^\lambda 2^{2^\lambda}(NT^2+T^5))$. □

6 EPTAS for Stationary Demand

Note that there is a straightforward way to turn a canonical schedule σ into a canonical pseudo-schedule $\overline{\sigma}$ and vice versa. Specifically, for any item i and interval I with $l_I = l_i + 1$, if σ has an order in interval I for item i, which needs to be in the same period as the first common order in this interval by the definition of a canonical schedule, replace this order by an order in the first period of interval I. This transformation defines a canonical pseudo-schedule $\overline{\sigma}$.

Analogously, given a canonical pseudo-schedule $\overline{\sigma}$, for each item i and interval I with $l_I = l_i+1$, if $\overline{\sigma}$ has an order in the first period of interval I for item i, then we just move it forward until we find the first common order in that interval. Such an order must exist by the definition of a canonical pseudo-schedule σ. Also this transformation defines the corresponding canonical schedule. We obtain the following lemma (proof in full version).

Lemma 8. *It holds that*

(1) let $\overline{\sigma}$ be the canonical pseudo-schedule generated from a canonical schedule σ as explained above, then $\mathbb{E}[\mathrm{cost}(\overline{\sigma})] \leq \mathrm{cost}(\sigma) + \frac{\sqrt{2}}{\lambda}\mathrm{OPT}$,

(2) let σ be the canonical schedule generated from a canonical pseudo-schedule $\overline{\sigma}$ as explained above, then $\mathbb{E}[\mathrm{cost}(\sigma)] \leq \mathrm{cost}(\overline{\sigma}) + \frac{\sqrt{2}}{\lambda}\mathrm{OPT}$.

Now we are ready to prove the main theorem.

Theorem 2. *There exists an efficient polynomial-time approximation scheme for the Joint Replenishment Problem with stationary demand. For any precision $\epsilon > 0$, the running time is $\mathcal{O}(\lambda 2^\lambda 2^{2^\lambda}(NT^2+T^5))$ for $\lambda = \frac{4\sqrt{2}}{\epsilon}$.*

Proof. We know from Lemma 4 that there is a canonical schedule σ with $\mathbb{E}[\mathrm{cost}(\sigma)] \leq (1+\frac{2\sqrt{2}}{\lambda})\mathrm{OPT}$. Therefore, by the first part of Lemma 8 and linearity of expectation, we obtain that there is a canonical pseudo-schedule $\overline{\sigma}$ with $\mathbb{E}[\mathrm{cost}(\overline{\sigma})] \leq (1+\frac{3\sqrt{2}}{\lambda})\mathrm{OPT}$. On the other hand, Lemma 5 shows that using the DP in combination with the approximate recurrence relation yields an optimal canonical pseudo-schedule $\overline{\sigma}^*$ with $\mathrm{cost}(\overline{\sigma}^*) \leq \mathrm{cost}(\overline{\sigma})$, and the second part of Lemma 8 gives that we can turn $\overline{\sigma}^*$ into a canonical schedule σ' with $\mathbb{E}[\mathrm{cost}(\sigma')] \leq \mathrm{cost}(\overline{\sigma}^*) + \frac{\sqrt{2}}{\lambda}\mathrm{OPT} \leq (1+\frac{4\sqrt{2}}{\lambda})\mathrm{OPT}$. This proves the claim in combination with Lemma 7. Specifically, we obtain approximation ratio $1+\epsilon$ for $\lambda = \frac{4\sqrt{2}}{\epsilon}$. Finally, note that this construction can be derandomized by enumerating all possible values κ in the construction of tree G. This adds another factor λ to the running time. □

7 Soft Capacitated Item Orders

The goal of this section is to prove the following theorem for base values $\chi_i = \frac{K_i}{h_i}$ (proof in full version).

Theorem 3. *The DP yields a polynomial-time approximation scheme for the Joint Replenishment Problem with non-stationary demand but soft capacitated item orders.*

References

1. Adelman, D., Klabjan, D.: Duality and existence of optimal policies in generalized joint replenishment. Mathematics of Operations Research 30(1), 28–50 (2005)
2. Aggarwal, A., Park, J.K.: Improved algorithms for economic lot size problems. Oper. Res. 41(3), 549–571 (1993)
3. Arkin, E., Joneja, D., Roundy, R.: Computational complexity of uncapacitated multi-echelon production planning problems. Operations Research Letters 8(2), 61–66 (1989)
4. Federgruen, A., Tzur, M.: A simple forward algorithm to solve general dynamic lot sizing models with n periods in $O(n \log n)$ or $O(n)$ time. Management Science 37(8), 909–925 (1991)
5. Harris, F.W.: Operations cost. Factory Management Series. A. W. Shaw Co., Chicago (1915)
6. Jackson, P., Maxwell, W., Muckstadt, J.: The joint replenishment problem with power-of-two restriction. AIIE Trans. 17, 25–32 (1985)
7. Levi, R., Roundy, R., Shmoys, D.B.: Primal-dual algorithms for deterministic inventory problems. Mathematics of Operations Research 31, 267–284 (2006)
8. Levi, R., Roundy, R., Shmoys, D.B., Sviridenko, M.: A constant approximation algorithm for the one-warehouse multi-retailer problem. Management Science 54, 763–776 (2008)
9. Roundy, R., Muckstadt, J.: Handbooks in Operations Research and Management Science: Analysis in Multi-Stage Production Systems, pp. 59–131. Elsevier (1993)
10. Naddor, E., Saltzman, S.: Optimal reorder periods for an inventory system with variable costs of ordering. Operations Research 6, 676–685
11. Nonner, T., Souza, A.: Approximating the joint replenishment problem with deadlines. Discrete Mathematics, Algorithms and Applications 1(2), 153–173 (2009)
12. Roundy, R.: 98%-effective integer-ratio lot-sizing for one-warehouse multi-retailer systems. Management Science 31(11), 1416–1430 (1985)
13. Schulz, A.S., Telha, C.: Approximation Algorithms and Hardness Results for the Joint Replenishment Problem with Constant Demands. In: Demetrescu, C., Halldórsson, M.M. (eds.) ESA 2011. LNCS, vol. 6942, pp. 628–639. Springer, Heidelberg (2011)
14. Segev, D.: An approximate dynamic-programming approach to the joint replenishment problem (2012) (manuscript)
15. Stauffer, G., Massonnet, G., Rapine, C., Gayon, J.-P.: A simple and fast 2-approximation algorithm for the one-warehouse multi-retailers problem. In: Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, pp. 67–79 (2011)
16. Teo, C.-P., Bertsimas, D.: Multistage lot sizing problems via randomized rounding. Operations Research 49(4), 599–608 (2001)

Chain-Constrained Spanning Trees*

Neil Olver[1] and Rico Zenklusen[2]

[1] MIT, Cambridge, USA
olver@math.mit.edu
[2] Johns Hopkins University, Baltimore, USA
ricoz@jhu.edu

Abstract. We consider the problem of finding a spanning tree satisfying a family of additional constraints. Several settings have been considered previously, the most famous being the problem of finding a spanning tree with degree constraints. Since the problem is hard, the goal is typically to find a spanning tree that violates the constraints as little as possible.

Iterative rounding became the tool of choice for constrained spanning tree problems. However, iterative rounding approaches are very hard to adapt to settings where an edge can be part of a super-constant number of constraints. We consider a natural constrained spanning tree problem of this type, namely where upper bounds are imposed on a family of cuts forming a chain. Our approach reduces the problem to a family of independent matroid intersection problems, leading to a spanning tree that violates each constraint by a factor of at most 9.

We also present strong hardness results: among other implications, these are the first to show, in the setting of a basic constrained spanning tree problem, a qualitative difference between what can be achieved when allowing multiplicative as opposed to additive constraint violations.

1 Introduction

Spanning tree problems with additional $\{0,1\}$-packing constraints have spawned considerable interest recently. This development was motivated by a variety of applications, including VLSI design, vehicle routing, and applications in communication networks [8,4,14]. Since even finding a feasible solution of a constrained spanning tree problem is typically NP-hard, the focus is on efficient procedures that either certify that the problem has no feasible solution, or find a spanning tree that violates the additional constraints by as little as possible. Often, an objective function to be minimized is also provided; here, however, we focus just on minimizing the constraint violations.

A wide variety of constrained spanning tree problems have been studied. Unfortunately, for most settings, little is known about what violation of the constraints must be accepted in order that a solution can be efficiently attained. An exception is the most classical problem in this context, the degree-bounded spanning tree problem. Here the goal is to find a spanning tree $T \subseteq E$ in a

* This project was supported by NSF grant CCF-1115849.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 324–335, 2013.

graph $G = (V, E)$ such that T satisfies a degree bound for each vertex $v \in V$, i.e., $|\delta(v) \cap T| \leq b_v$. For this problem, Fürer and Raghavachari [8] presented an essentially best possible algorithm that either shows that no spanning tree satisfying the degree constraints exists, or returns a spanning tree violating each degree constraint by at most 1. We call this an *additive* 1-*approximation*, by contrast to an α-approximation, where each constraint can be violated by a *factor* $\alpha > 1$.

Recently, iterative rounding/relaxation algorithms became the tool of choice for dealing with constrained spanning tree problems. A cornerstone for this development was the work of Singh and Lau [16], which extended the iterative rounding framework of Jain [10] with a relaxation step. They obtained an additive 1-approximation even for the *minimum* degree-bounded spanning tree problem, i.e., the cost of the tree they return is upper bounded by the cost of an optimal solution not violating any constraints. This result was the culmination of a long sequence of papers presenting methods with various trade-offs between constraint violation and cost (see [11,12,5,6,9] and references therein).

Singh and Lau's iterative relaxation technique was later generalized by Bansal, Khandekar and Nagarajan [3], to show that even when upper bounds are given on an arbitrary family of edge-sets $E_1, \ldots, E_k$, one can still find a (min cost) spanning tree violating each constraint by at most $\max_{e \in E} |\{i \in [k] \mid e \in E_i\}| - 1$. If each edge is only in a constant number of constraints, this leads to an additive $O(1)$-approximation. But extending the iterative rounding technique beyond such settings seems to typically be very difficult. Some progress was achieved by Bansal, Khandekar, Könemann, Nagarajan and Peis [2], who used an iterative approach that iteratively replaces constraints by weaker ones, leading to an additive $O(\log n)$-approximation if the constraints are upper bounds on a laminar family of cuts. They left open whether an additive or multiplicative $O(1)$-approximation is possible in this setting, even when the cuts form a chain. Recently, Zenklusen [17] presented an additive $O(1)$-approximation for generalized degree bounds, where for each vertex an arbitrary matroid constraint on its adjacent edges has to be satisfied. This algorithms differs from previous iterative rounding approaches in that it successively simplifies the problem to reach a matroid intersection problem, rather than attempting to eliminate constraints until only spanning tree constraints remain.

To the best of our knowledge, with the exception of the setting of Zenklusen [17], no $O(1)$-approximations are known for constrained spanning tree problems where an edge can lie in a super-constant number of (linear) constraints. This seems to be an important difficulty that current techniques have trouble overcoming. Furthermore, in many settings, it is not well understood if finding an additive approximation is any harder than a multiplicative one. In particular, no constrained spanning tree problem was previously known where an $O(1)$-approximation is possible, but an additive $O(1)$-approximation is not. The goal of this paper is to address these points by studying chain-constrained spanning trees—a natural constrained spanning tree problem that evades current techniques.

1.1 Our Results

In this paper, we consider what is arguably one of the most natural constraint families for which finding $O(1)$-approximations seems beyond current techniques—chain constraints. Here we are given an undirected graph $G = (V, E)$ together with a family of cuts $\emptyset \subsetneq S_1 \subsetneq S_2, \ldots, \subsetneq S_\ell \subsetneq V$ forming a chain, and bounds $b_1, \ldots, b_\ell \in \mathbb{Z}_{>0}$. In summary, our reference problem is the following.

$$\begin{gathered}\text{Find a spanning tree } T \in \mathcal{T} \text{ satisfying:} \\ |T \cap \delta(S_i)| \leq b_i \quad \forall i \in [\ell],\end{gathered} \tag{1}$$

where $\mathcal{T} \subseteq 2^E$ is the family of all spanning trees of G, and $\delta(S_i) \subseteq E$ denotes all edges with precisely one endpoint in S_i.

Notice that chain constraints allow edges to be in a super-constant number of constraints. They are also a natural candidate problem that captures many of the difficulties faced when trying to construct $O(1)$-approximations for the laminar case. Our main algorithmic result is the following.

Theorem 1. *There is an efficient* 9*-approximation for the chain-constrained spanning tree problem.*

Contrary to previous procedures, our method is not based on iterative rounding. Instead, we reduce the problem to a family of independent matroid intersection problems. More precisely, we rely on a subprocedure that works in graphs without *rainbows*, by which we mean a pair of edges e, f such that e is in a proper superset of the chain constraints in which f is contained. To reach a rainbow-free setting, we solve a natural LP relaxation with an appropriately chosen objective function. We then show that one can consider a maximal family of linearly independent and tight spanning tree constraints to decompose the problem into rainbow-free subproblems, one for each chosen spanning tree constraint. Even though the high-level approach is quite clean, there are several difficulties we have to address. In particular, to do the accounting of how much a constraint is violated across all subproblems, we define a well-chosen requirement that the solutions of the subproblems must fulfill, and which allows us to bound the total violation of a constraint. Due to space constraints, details will appear in a long version of this paper.

Our main result on the hardness side is the following.

Theorem 2. *For the chain-constrained spanning tree problem it is NP-hard to distinguish between the cases where a spanning tree satisfying the chain constraints exists, and the case that every spanning tree violates some degree bound by* $\Omega(\log n / \log \log n)$ *units.*

This result has several interesting implications. First, it shows that even for chain constraints there is a clear qualitative difference between what can be achieved when considering additive versus multiplicative violation. Hence, Theorem 2 together with Theorem 1 show for the first time that there are constrained spanning tree problems where an additive $O(1)$-approximation would imply P = NP

whereas an $O(1)$-approximation exists. Previously, the only hardness result of a similar nature to Theorem 2 was presented by Bansal et al. [2], for a very general constrained spanning tree problem, where constraints $|T \cap E_i| \leq b_i \; \forall i \in [k]$ are given for an arbitrary family of edge sets $E_1, \ldots, E_k \subseteq E$. They showed that unless NP has quasi-polynomial time algorithms, there is no additive $(\log^c n)$-approximation for this case, for some small constant $c \in (0, 1)$. Notice that our hardness result is stronger in terms of the approximation ratio, the underlying constrained spanning tree model, and the complexity assumption. Furthermore, Theorem 2 shows that the additive $O(\log n / \log \log n)$-approximation of Bansal et al. [2] for the laminar-constrained spanning tree problem is close to optimal.

Due to space constraints, the proof of Theorem 2 and further results on hardness and integrality gaps will appear in a long version of this paper.

1.2 Related Work

The problem of finding a *thin* tree, which recently came to fame, can be interpreted as a constrained spanning tree problem. Here, upper bounds are imposed on the number of edges to be chosen in any cut of the graph. More precisely, given a point x in the spanning tree polytope of G, a spanning tree T is α-thin with respect to x if $|T \cap \delta(S)| \leq \alpha \cdot x(\delta(S)) \; \forall S \subseteq V$. For the thin spanning tree problem the currently best known procedures only lead to a thinness of $\alpha = \Theta(\log n / \log \log n)$ [1,7]. The concept of thin spanning trees gained considerably in relevance when Asadpour et al. [1] showed that an efficient algorithm for finding an α-thin spanning tree leads to an $O(\alpha)$-approximation for the Asymmetric Traveling Salesman Problem (ATSP)[1]. Using this connection they obtained the currently best approximation algorithm for ATSP with an approximation factor of $O(\log n / \log \log n)$. It is open whether $O(1)$-thin spanning trees exist, which would immediately imply an $O(1)$-factor approximation for ATSP.

2 The Algorithm

To simplify the exposition, we assume that we are dealing with a maximal chain of constraints imposed on the spanning tree. Hence, we can choose a numbering of the vertices $V = \{v_1, \ldots, v_n\}$ of the graph $G = (V, E)$ such that we have a constraint $|T \cap \delta(S_i)| \leq b_i$ for $S_i = \{v_1, \cdots, v_i\} \; \forall \; i \in [n-1]$. This is clearly not restrictive since by choosing a large right-hand side, any constraint can be made redundant.

Our algorithms starts by computing an optimal solution to the natural LP relaxation of Problem (1), that asks to find a point in the spanning tree polytope P_{ST} of G satisfying the chain constraints. More precisely, we do not want to start with an arbitrary feasible solution to this LP, but with one that minimizes the

[1] Strictly speaking, Asadpour et al.'s approach required the spanning tree not only to be thin, but also to be of low cost. However this second requirement is not necessary for the mentioned statement to be true (see [13]).

total length of the edges, where the length of an edge $\{v_i, v_j\} \in E$ is $|i-j|$, i.e., the number of chain constraints to which the edge contributes. This leads to the LP (2) shown below. Let x^* be an optimal solution to (2), which can be computed by standard techniques (see [15]). Notice that the objective function of (2) is the same as the total load on all cuts: $\sum_{i=1}^{n-1} x(\delta(S_i))$.

$$\begin{aligned} \min \sum_{\{v_i,v_j\}\in E} |i-j| \cdot x(\{v_i, v_j\}) \\ x \in P_{ST} \\ x(\delta(S_i)) \leq b_i \qquad \forall\, i \in [n-1] \end{aligned} \tag{2}$$

The above objective function is motivated by a subprocedure we use to find a spanning tree in an instance that does not contain what we call a *rainbow*. A rainbow consists of two edges $\{v_i, v_j\}, \{v_p, v_q\}$ with $i \leq p < q \leq j$ and either $i < p$ or $q < j$, i.e., the first edge is in a proper superset of the chain constraints in which the second edge is in. Even though the above objective function does not necessarily lead to an LP solution x^* whose support $\text{supp}(x^*) = \{e \in E \mid x^*(e) > 0\}$ does not contain rainbows—a feasible rainbow-free solution may not even exist—it eliminates rainbows in subproblems we are interested in, as we will see later. Clearly, if LP (2) is not feasible, we know that the reference problem has no feasible solution.

In all what follows, we only work on edges in $\text{supp}(x^*)$. Therefore, to simplify the exposition, we assume from now on that $E = \text{supp}(x^*)$. This can easily be achieved by deleting all edges $e \in E$ with $x^*(e) = 0$ from G.

Our algorithm decomposes the problem of finding an $O(1)$-approximate spanning tree $T \subseteq E$ into an independent family of a special type of spanning tree problem on rainbow-free graphs. To decompose the problem, we consider tight spanning tree constraints. More precisely, let $\mathcal{L} \subseteq 2^V$ be any maximal laminar family of vertex-sets corresponding to spanning tree constraints that are tight with respect to x^*. In other words, $\mathcal{L}$ is maximal laminar family chosen from the sets $L \subseteq V$ satisfying $x^*(E[L]) = |L| - 1$, where, $E[L] \subseteq E$ denotes the set of edges with both endpoints in L. In particular, $\mathcal{L}$ contains all singletons. We say that $L_2 \in \mathcal{L}$ is a child of $L_1 \in \mathcal{L}$ if $L_2 \subsetneq L_1$ and there is no set $L_3 \in \mathcal{L}$ with $L_2 \subsetneq L_3 \subsetneq L_1$. For $L \in \mathcal{L}$, we denote by $\mathcal{C}(L) \subset \mathcal{L}$ the set of all children of L. Notice that $\mathcal{C}(L)$ forms a partition of L.

To construct a spanning tree T in G we will determine for each $L \in \mathcal{L}$ a set of edges T_L in

$$E_L := E[L] \setminus (\cup_{C \in \mathcal{C}(L)} E[C]),$$

that form a spanning tree in the graph G_L obtained from the graph (L, E_L) by contracting all children of L. Hence, the vertex set of G_L is $\mathcal{C}(L)$, and an original edge $\{u, v\} \in E_L$ is simply interpreted as an edge between the two children $C_u, C_v \in \mathcal{C}(L)$ that contain u and v, respectively. For singletons $L \in \mathcal{L}$, we set $T_L = \emptyset$. One can easily observe that a family $\{T_L\}_{L \in \mathcal{L}}$ of spanning trees in $\{G_L\}_{L \in \mathcal{L}}$ leads to a spanning tree $T = \cup_{L \in \mathcal{L}} T_L$ in G. Constructing "good" spanning trees T_L in G_L, for each $L \in \mathcal{L}$, will be our independent subproblems.

As we will argue more formally later, the main benefit of this division is that the edge set E_L used in the subproblem to find T_L does not contain any rainbows. Our goal is to define constraints that the spanning trees T_L have to satisfy, that allow us to conclude that the resulting spanning tree $T = \cup_{L\in\mathcal{L}} T_L$ does not violate the chain constraints by more than a constant factor.

One of the arguably most natural constraint families to impose would be to require that the contribution of T_L to any cut S_i is within a constant factor of the contribution of x^* on S_i when only considering edges in E_L, i.e.,

$$|T_L \cap \delta(S_i)| \leq O(x^*(\delta(S_i) \cap E_L)). \tag{3}$$

If the above inequality holds for each $L \in \mathcal{L}$ and $i \in [n-1]$, then the final spanning tree T will indeed not violate any chain constraint by more than a constant factor: it suffices to sum up the inequalities for a fixed i over all sets L and observe that $\{T_L\}_{L\in\mathcal{L}}$ partitions T, and $\{E_L\}_{L\in\mathcal{L}}$ is a partition of E_L:

$$\begin{aligned} |T \cap \delta(S_i)| &= \sum_{L\in\mathcal{L}} |T_L \cap \delta(S_i)| \leq O\left(\sum_{L\in\mathcal{L}} x^*(\delta(S_i) \cap E_L)\right) \\ &= O(x^*(\delta(S_i))) = O(1)b_i. \end{aligned} \tag{4}$$

Unfortunately, it turns out that it is in general impossible to find spanning trees T_L that satisfy (3). This is because there can be many constraints S_i for which $x^*(\delta(S_i \cap E[L])) = o(1)$, in a setting where one has to include at least one edge in T_L that crosses one of these constraints[2].

We therefore introduce a weaker condition on T_L. For $L \in \mathcal{L}$ and $i \in [n-1]$, let $\mathcal{C}_i(L) \subseteq \mathcal{C}(L)$ be the family of all children $C \in \mathcal{C}(L)$ of L that cross the cut S_i, i.e., $S_i \cap L \neq \emptyset$ and $L \setminus S_i \neq \emptyset$. We want the sets T_L to satisfy the following:

$$|T_L \cap \delta(S_i)| \leq 7 \cdot x^*(\delta(S_i) \cap E_L) + 2 \cdot \mathbf{1}_{\{|\mathcal{C}_i(L)|\geq 2\}} \quad \forall i \in [n-1]. \tag{5}$$

Here, $\mathbf{1}_{\{|\mathcal{C}_i(L)|\geq 2\}}$ is the indicator that is equal to 1 if $|\mathcal{C}_i(L)| \geq 2$ and 0 otherwise.

We first show in Section 2.1 that satisfying the above condition indeed leads to a good spanning tree T.

Theorem 3. *For $L \in \mathcal{L}$, let T_L be a spanning tree in G_L that satisfies* (5)*. Then $T = \cup_{L\in\mathcal{L}} T_L$ is a spanning tree in G satisfying*

$$|T \cap \delta(S_i)| \leq 9x^*(\delta(S_i)) \leq 9b_i \quad i \in [n-1].$$

We then show in Section 2.2 that such spanning trees can indeed be found efficiently.

Theorem 4. *For each $L \in \mathcal{L}$, we can efficiently find a spanning tree T_L in G_L satisfying* (5)*.*

[2] Details will be provided in the full version of this paper.

Combining the above two theorems immediately leads to an efficient algorithm to find a spanning tree in G that violates each chain constraint by at most a factor of 9 whenever LP (2) is feasible, and thus proves Theorem 1. For convenience, a summary of our algorithm is provided below.

Algorithm to find $T \in \mathcal{T}$ that violates chain constraints by a factor of at most 9.

1. Compute optimal solution x^* to the linear program (2).
2. Independently for each $L \in \mathcal{L}$, invoke Theorem 4 to obtain a spanning tree T_L in G_L satisfying (5).
3. Return $T = \cup_{L \in \mathcal{L}} T_L$.

2.1 Analysis of Algorithm (Proof of Theorem 3)

For each $L \in \mathcal{L}$, let T_L be a spanning tree in G_L that satisfies (5), let $T = \cup_{L \in \mathcal{L}} T_L$, and let $i \in [n-1]$. Using the same reasoning as in (4) we can bound the load on chain constraint i as follows:

$$\begin{aligned} |T \cap \delta(S_i)| &= \sum_{L \in \mathcal{L}} |T_L \cap \delta(S_i)| \overset{(5)}{\leq} 7 \sum_{L \in \mathcal{L}} x^*(\delta(S_i) \cap E_L) + 2 \sum_{L \in \mathcal{L}} \mathbf{1}_{\{|\mathcal{C}_i(L)| \geq 2\}} \\ &= 7x^*(\delta(S_i)) + 2 \sum_{L \in \mathcal{L}} \mathbf{1}_{\{|\mathcal{C}_i(L)| \geq 2\}}, \end{aligned}$$

using the fact that $\{E_L\}_{L \in \mathcal{L}}$ partitions E. To prove Theorem 3, it thus suffices to show

$$\sum_{L \in \mathcal{L}} \mathbf{1}_{\{|\mathcal{C}_i(L)| \geq 2\}} \leq x^*(\delta(S_i)), \tag{6}$$

which then implies

$$|T \cap \delta(S_i)| \leq 9x^*(\delta(S_i)) \leq 9b_i,$$

where the last inequality follows from x^* being feasible for (2). We complete the analysis by showing the following result, which is a stronger version of (6).

Lemma 1.

$$\sum_{L \in \mathcal{L}} (|\mathcal{C}_i(L)| - 1)^+ \leq x^*(\delta(S_i)),$$

where $(\cdot)^+ = \max(0, \cdot)$.

Proof. Let $\mathcal{L}_i \subseteq \mathcal{L}$ be the family of all sets in $\mathcal{L}$ that cross S_i, and let $\mathcal{L}_i^{\min} \subseteq \mathcal{L}_i$ be all minimal sets of $\mathcal{L}_i$. We will show the following:

$$\sum_{L \in \mathcal{L}} (|\mathcal{C}_i(L)| - 1)^+ = |\mathcal{L}_i^{\min}| - 1. \tag{7}$$

We start by observing how the statement of the lemma follows from (7). Since all sets $W \in \mathcal{L}_i$ correspond to tight spanning tree constraints with respect to

x^*, we have that the restriction $x^*|_{E[W]}$ of x^* to the edges in the graph $G[W]$ is a point in the spanning tree polytope of $G[W]$. In particular, at least one unit of $x^*|_{E[W]}$ crosses any cut in $G[W]$. Since $W \in \mathcal{L}_i$, the set S_i induces a cut $(S_i \cap W, W \setminus S_i)$ in $G[W]$. Hence

$$x^*(\delta(S_i) \cap E[W]) \geq 1 \quad \forall\, W \in \mathcal{L}_i.$$

Now observe that due to minimality of the sets in $\mathcal{L}_i^{\min}$, all sets in $\mathcal{L}_i^{\min}$ are disjoint. Thus

$$x^*(\delta(S_i)) \geq \sum_{W \in \mathcal{L}_i^{\min}} x^*(\delta(S_i) \cap E[W]) \geq |\mathcal{L}_i^{\min}|,$$

which, together with (7), implies Lemma 1. Hence, it remains to show (7).

Let $\mathcal{L}_i^{\mathrm{nm}} = \mathcal{L}_i \setminus \mathcal{L}_i^{\min}$ be all sets in $\mathcal{L}_i$ that are not minimal. Notice that only sets $L \in \mathcal{L}^{\mathrm{nm}}$ can have a strictly positive contribution to the left-hand side of (7) since these are precisely the sets $L \in \mathcal{L}$ with $|\mathcal{C}_i(L)| \geq 1$: for any other set $L \in \mathcal{L}$, either (i) $L \notin \mathcal{L}_i$, in which case non of its children can cross S_i since not even L crosses S_i, or (ii) $L \in \mathcal{L}_i^{\min}$, in which case we again get $|\mathcal{C}_i(L)| = 0$ since L has no children in $\mathcal{L}_i$ due to minimality. We thus obtain

$$\sum_{L \in \mathcal{L}} (|\mathcal{C}_i(L)| - 1)^+ = \sum_{L \in \mathcal{L}_i^{\mathrm{nm}}} (|\mathcal{C}_i(L)| - 1). \tag{8}$$

Observe that $\sum_{L \in \mathcal{L}_i^{\mathrm{nm}}} |\mathcal{C}_i(L)|$ counts each set in $\mathcal{L}_i$ precisely once, except for the set $V \in \mathcal{L}_i$ which is the only set in $\mathcal{L}_i$ that is not a child of some other set in $\mathcal{L}_i$. Hence

$$\sum_{L \in \mathcal{L}_i^{\mathrm{nm}}} |\mathcal{C}_i(L)| = |\mathcal{L}_i| - 1. \tag{9}$$

Finally, combining (8) with (9) we obtain

$$\sum_{L \in \mathcal{L}} (|\mathcal{C}_i(L)| - 1)^+ = \sum_{L \in \mathcal{L}_i^{\mathrm{nm}}} (|\mathcal{C}_i(L)| - 1) = |\mathcal{L}_i| - 1 - |\mathcal{L}_i^{\mathrm{nm}}| = |\mathcal{L}_i^{\min}| - 1,$$

thus proving (7). □

2.2 Main Step of Algorithm (Proof of Theorem 4)

Let $L \in \mathcal{L}$. We now consider the problem of finding a spanning tree T_L in G_L that satisfies (5). Recall that G_L is obtained from the graph (L, E_L) by contracting all children of L. For simplicity, we again interpret an edge $\{v_i, v_j\} \in E_L$ as an edge in G_L between the two vertices corresponding to the sets $C_i, C_j \in \mathcal{L}$ that contain v_i and v_j, respectively.

We start by showing that there are no rainbows in E_L, which is a crucial assumption in the algorithm to be presented in the following.

Lemma 2. *For $L \in \mathcal{L}$, E_L does not contain any rainbows.*

Due to space constraints, the formal proof of Lemma 2 is deferred to the long version of this paper. The intuition behind the result is that when restricting x^* to G_L, a point z in the *interior* of the spanning tree polytope of G_L is obtained with components in $(0, 1)$. Two edges $e, f \in E_L$ forming a rainbow would contradict the optimality of x^* for (2), since one could move some mass from the edge that is in more constraints to the one in fewer. This decreases the objective function, does not violate any chain constraints, and is still in the spanning tree polytope since changes are only done to components represented in z, and z is in the interior of the spanning tree polytope of G_L.

We classify chain constraints S_i into two types, depending on the right-hand side of (5). More precisely, we call a cut S_i *bad* if one can include at most one edge that crosses S_i in T_L without violating (5), i.e.,

$$7x^*(\delta(S_i) \cap E_L) + 2 \cdot \mathbf{1}_{\{|\mathcal{C}_i(L)|\geq 2\}} < 2.$$

Otherwise, a cut S_i is called *good*. Notice that for a cut S_i to be bad, we need to have $|\mathcal{C}_i(L)| = 1$ because of the following. Clearly, if $|\mathcal{C}_i(L)| \geq 2$, then S_i cannot be bad due to the term $2 \cdot \mathbf{1}_{\{|\mathcal{C}_i(L)|\geq 2\}}$. If $|\mathcal{C}_i(L)| = 0$, then we use the fact that all edges in $E[L]$ that cross S_i are part of E_L, hence

$$x^*(\delta(S_i) \cap E_L) = x^*(\delta(S_i) \cap E[L]) \geq 1,$$

where the last inequality follows from the fact that $x^*|_{E[L]}$ is in the spanning tree polytope of the graph $(L, E[L])$. Hence a cut S_i is bad if and only if the following two conditions hold simultaneously:

1. $|\mathcal{C}_i(L)| = 1$,
2. $x^*(\delta(S_i) \cap E_L) < \frac{2}{7}$.

An edge $e \in E_L$ is called *bad* if e crosses at least one bad cut S_i, otherwise it is called *good*. We denote by $A_L \subseteq E_L$ the sets of all good edges.

The procedure we use to find a tree T_L satisfying (5) constructs a tree T_L that consists of only good edges, i.e., $T_L \subseteq A_L$. We determine T_L using a matroid intersection problem that asks to find a spanning tree in G_L satisfying an additional partition matroid constraint.

To define the partition matroid we first number the edges $A_L = \{e_1, \ldots, e_k\}$ as follows. For $e \in A_L$, let $\alpha(e) < \beta(e)$ be the lower and higher index of the two endpoints of e, hence, $e = \{v_{\alpha(e)}, v_{\beta(e)}\}$. (Notice that $\alpha(e) = \beta(e)$ is not possible since $x^*(e) > 0 \ \forall e \in E$ and $x^* \in P_{ST}$.) The edges $e \in A_L$ are numbered lexicographically, first according to lower value of $\alpha(e)$ and then according to lower value of $\beta(e)$, i.e., for any $p \in [k-1]$ either $\alpha(e_p) < \alpha(e_{p+1})$, or $\alpha(e_p) = \alpha(e_{p+1})$ and $\beta(e_p) \leq \beta(e_{p+1})$. Ideally, we would like to group the edges in A_L into consecutive blocks $\{e_p, e_{p+1}, \ldots, e_q\}$ each having a total weight of exactly $x^*(\{e_p, \ldots, e_q\}) = 3/7$. Since this is in general not possible, we will split some of the edges by creating two parallel copies. More precisely, to define the first set P_1 of our partition, let $p \in [k]$ the lagest index for which $x^*(\{e_1, \ldots, e_p\}) \leq 3/7$.

If $x^*(\{e_1,\ldots,e_p\}) = 3/7$ then $P_1 = \{e_1,\ldots,e_p\}$. Otherwise, we replace the edge e_{p+1} by two parallel copies e'_{p+1}, e''_{p+1} of e_{p+1}, and we distribute the weight of $x^*(e_{p+1})$ on e'_{p+1}, e''_{p+1} as follows:

$$x^*(e'_{p+1}) = \frac{3}{7} - x^*(\{e_1,\ldots,e_p\}),$$
$$x^*(e''_{p+1}) = x^*(e_{p+1}) - x^*(e'_{p+1}).$$

This splitting operation does not violate any previous assumptions: the weight x^* on the new edge set $\{e_1,\ldots,e_p,e'_{p+1},e''_{p+1},e_{p+2},\ldots,e_k\}$ is still a point in the spanning tree polytope of the graph over the vertices $\mathcal{C}(L)$ with the new edge set. By applying this splitting operation whenever necessary, we can assume that $A_L = \{e_1,\ldots,e_k\}$ can be partitioned into sets $P_1 = \{e_1,\ldots,e_{p_1}\}$, $P_2 = \{e_{p_1+1},\ldots,e_{p_2}\}$, $\ldots$, $P_s = \{e_{p_{s-1}+1},\ldots,e_k\}$ satisfying:

(i) $x^*(P_h) = 3/7 \quad \forall h \in [s-1]$,
(ii) $x^*(P_s) \leq 3/7$.

Using this partition we define a unitary partition matroid $M = (A_L, \mathcal{I})$ on the good edges A_L, with independent sets

$$\mathcal{I} = \{U \subseteq A_L \mid |U \cap P_h| \leq 1 \ \forall h \in [s]\}.$$

The tree spanning T_L in G_L that our algorithm selects is any spanning tree $T_L \subseteq A_L$ in G_L that is independent in the partition matroid M. Notice that if there exists a spanning tree in G_L that is independent in M, then such a spanning tree can be found in polyonmial time by standard matroid intersection techniques (see [15] for more details about matroids in general and techniques to find common independent sets in the intersection of two matroids). Hence to complete the description and analysis of our algorithm, all that remains is to show the existence of a spanning tree in G_L that is independent in M, and that it satisfies (5). We address these two statements in the following.

The theorem below shows the feasibility of the matroid intersection problem.

Theorem 5. *There exists a spanning tree $T_L \subseteq A_L$ in G_L that is independent in M, i.e., $T_L \in \mathcal{I}$.*

We give a sketch of the proof plan; the full proof is omitted from this extended abstract. We prove that the intersection of the dominant of the spanning tree polytope and the matroid polytope corresponding to M is nonempty. The result then follows by the fact that the intersection of these two polyhedra leads to an integral polytope, a classical result on matroid intersection [15]. More precisely, we show that the point obtained from $\frac{7}{3}x^*$ by setting the values of all bad edges to zero is in both of the above-mentioned polyhedra.

The following theorem finishes the analysis of our algorithm.

Theorem 6. *Let $T_L \subseteq A_L$ be a spanning tree in G_L that is independent in M, then T_L satisfies* (5).

Proof. Consider a cut $S_i \in \mathcal{S}$ for some fixed $i \in [n-1]$. We consider the partition $P_1, \ldots, P_s$ of A_L used to define the partition matroid M. We are interested in all sets in this partition that contain edges crossing S_i. The definition of the partition $P_1, \ldots, P_s$, together with the fact that A_L has no rainbows, implies that the sets of the partition containing edges crossing S_i are consecutively numbered $P_a, P_{a+1}, \ldots, P_b$, for some $1 \leq a \leq b \leq s$. Since T_L contains at most one edge in each partition, we have

$$|T_L \cap \delta(S_i)| \leq b - a + 1. \tag{10}$$

We first consider the case $b - a \geq 2$. Notice that all edges in any set P_h for $a < h < b$ cross S_i. Hence,

$$x^*(\delta(S_i) \cap E_L) \geq \sum_{h=a+1}^{b-1} x^*(P_h) = (b - a - 1) \cdot \frac{3}{7},$$

where we used $x^*(P_h) = \frac{3}{7}$ for $1 \leq h \leq s - 1$. Combining the above inequality with (10), and using that $b - a \geq 2$ in the second inequality, we obtain that

$$|T_L \cap \delta(S_i)| \leq b - a + 1 \leq 3(b - a - 1) \leq 7x^*(\delta(S_i) \cap E_L).$$

Thus T_L satisfies (5).

Assume now $b - a \leq 1$. If S_i is bad, then $|T_L \cap \delta(S_i)| = 0$ since T_L only contains good edges and no good edge crosses any bad cut. Hence, T_L trivially satisfies (5). So assume that S_i is good, i.e., either $|\mathcal{C}(L)| \geq 2$ or $x^*(\delta(S_i) \cap E_L) \geq \frac{2}{7}$. If $|\mathcal{C}(L)| \geq 2$, then beginning again from (10) we have

$$|T_L \cap \delta(S_i)| \leq b - a + 1 \leq 2 = 2 \cdot \mathbf{1}_{|\mathcal{C}_i(L)| \geq 2}.$$

Otherwise, if $x^*(\delta(S_i) \cap E_L) \geq \frac{2}{7}$, then

$$|T_L \cap \delta(S_i)| \leq 2 \leq 7x^*(\delta(S_i) \cap E_L).$$

Either way, T_L satisfies (5). □

3 Conclusions

We would like to close with several interesting directions for future research. One very natural question is whether there is an $O(1)$-approximation for laminar cut constraint; we believe this to be true. Although it seems non-trivial to directly generalize our procedure for the chain-constrained case to the laminar case, we hope that they can be useful in combination with insights from $O(1)$-approximations for the degree-bounded case.

Another natural extension would be to find a weighted $O(1)$-approximation for the chain-constrained spanning tree problem, where the cost of the returned spanning tree should be no larger than the optimal cost of a spanning tree that does not violate the constraints. The main reason our approach does not generalize easily to this setting is that we use a particular objective function to eliminate rainbows in the subproblems.

References

1. Asadpour, A., Goemans, M.X., Madry, A., Oveis Gharan, S., Saberi, A.: An $O(\log n/\log\log n)$-approximation algrithm for the asymmetric traveling salesman problem. In: Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA (2010)
2. Bansal, N., Khandekar, R., Könemann, J., Nagarajan, V., Peis, B.: On generalizations of network design problems with degree bounds. Mathematical Programming, 1–28 (April 2012)
3. Bansal, N., Khandekar, R., Nagarajan, V.: Additive guarantees for degree-bounded directed network design. SIAM Journal on Computing 39(4), 1413–1431 (2009)
4. Bauer, F., Varma, A.: Degree-constrained multicasting in point-to-point networks. In: Proceedings of the Fourteenth Annual Joint Conference of the IEEE Computer and Communication Societies, INFOCOM, pp. 369–376 (1995)
5. Chaudhuri, K., Rao, S., Riesenfeld, S., Talwar, K.: A push-relabel approximation algorithm for approximating the minimum-degree MST problem and its generalization to matroids. Theoretical Computer Science 410, 4489–4503 (2009)
6. Chaudhuri, K., Rao, S., Riesenfeld, S., Talwar, K.: What would Edmonds do? Augmenting paths and witnesses for degree-bounded MSTs. Algorithmica 55, 157–189 (2009)
7. Chekuri, C., Vondrák, J., Zenklusen, R.: Dependent randomized rounding via exchange properties of combinatorial structures. In: Proceedings of the 51st IEEE Symposium on Foundations of Computer Science, FOCS, pp. 575–584 (2010)
8. Fürer, M., Raghavachari, B.: Approximating the minimum-degree Steiner Tree to within one of optimal. Journal of Algorithms 17(3), 409–423 (1994)
9. Goemans, M.X.: Minimum bounded degree spanning trees. In: Proceedings of the 47th IEEE Symposium on Foundations of Computer Science, FOCS, pp. 273–282 (2006)
10. Jain, K.: A factor 2 approximation algorithm for the generalized Steiner Network Problem. Combinatorica 21, 39–60 (2001)
11. Könemann, J., Ravi, R.: A matter of degree: Improved approximation algorithms for degree-bounded minimum spanning trees. SIAM Journal on Computing 31, 1783–1793 (2002)
12. Könemann, J., Ravi, R.: Primal-dual meets local search: approximating MST's with nonuniform degree bounds. In: Proceedings of the 35th Annual ACM Symposium on Theory of Computing, STOC, pp. 389–395 (2003)
13. Oveis Gharan, S., Saberi, A.: The asymmetric traveling salesman problem on graphs with bounded genus. ArXiv (January 2011), `http://arxiv.org/abs/0909.2849`
14. Ravi, R., Marathe, M.V., Ravi, S.S., Rosenkrantz, D.J., Hunt III, H.B.: Approximation algorithms for degree-constrained minimum-cost network-design problems. Algorithmica 31(1), 58–78 (2001)
15. Schrijver, A.: Combinatorial Optimization, Polyhedra and Efficiency. Springer (2003)
16. Singh, M., Lau, L.C.: Approximating minimum bounded degree spanning trees to within one of optimal. In: Proceedings of the 39th Annual ACM Symposium on Theory of Computing, STOC, pp. 661–670 (2007)
17. Zenklusen, R.: Matroidal degree-bounded minimum spanning trees. In: Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, pp. 1512–1521 (2012)

A Simpler Proof for O(Congestion + Dilation) Packet Routing

Thomas Rothvoß*

MIT, Cambridge, USA
rothvoss@math.mit.edu

Abstract. In the *store-and-forward routing* problem, packets have to be routed along given paths such that the arrival time of the latest packet is minimized. A groundbreaking result of Leighton, Maggs and Rao says that this can always be done in time O(congestion + dilation), where the *congestion* is the maximum number of paths using an edge and the *dilation* is the maximum length of a path. However, the analysis is quite arcane and complicated and works by iteratively improving an infeasible schedule. Here, we provide a more accessible analysis which is based on conditional expectations. Like [LMR94], our easier analysis also guarantees that constant size edge buffers suffice.

Moreover, it was an open problem stated e.g. by Wiese [Wie11], whether there is any instance where all schedules need at least $(1 + \varepsilon) \cdot$ (congestion + dilation) steps, for a constant $\varepsilon > 0$. We answer this question affirmatively by making use of a probabilistic construction.

1 Introduction

One of the fundamental problems in parallel and distributed systems is to transport packets within a communication network in a timely manner. Any routing protocol has to make two kinds of decisions: (1) on which paths shall the packets be sent and (2) according to which priority rule should packets be routed along those paths, considering that communication links have usually a limited bandwidth. In this paper, we focus on the second part of the decision process. More concretely, we assume that a network in form of a directed graph $G = (V, E)$ is given, together with source sink pairs $s_i, t_i \in V$ for $i = 1, \ldots, k$ and s_i-t_i paths $P_i \subseteq E$. So the goal is to route the packets from their source along the given path to their sink in such a way that the *makespan* is minimized. Here, the makespan denotes the time when the last packet arrives at its destination. Moreover, we assume *unit bandwidth* and *unit transit time*, i.e. in each time unit only one packet can traverse an edge and the traversal takes exactly one time unit. Since the only freedom for the scheduler lies in the decision when packets move and when they wait, this setting is usually called *store and forward routing*. Note that we make no assumption about the structure of the graph or the paths.

* Supported by the Alexander von Humboldt Foundation within the Feodor Lynen program, by ONR grant N00014-11-1-0053 and by NSF contract CCF-0829878.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 336–348, 2013.

In fact, we can allow that the graph has multi-edges and loops; a path may even revisit the same node several times. We only forbid that a path uses the same edge more than once.

Two natural parameters of the instance are the *congestion* $C := \max_{e \in E} |\{i \mid e \in P_i\}|$, i.e. the maximum number of paths that share a common edge and the *dilation* $D := \max_{i=1,\ldots,k} |P_i|$, i.e. the length of the longest path.

Obviously, for any instance, both parameters C and D are lower bounds on the makespan for any possible routing policy. Surprisingly, Leighton, Maggs and Rao [LMR94] could prove that the optimum achievable makespan is always within a constant factor of $C + D$. Since then, their approach has been revisited several times. First, [LMR99] provided a polynomial time algorithm that makes the approach constructive (which nowadays would be easy using the Moser Tardos algorithm [MT10]). Scheideler [Sch98, Chapter 6] provides a more careful (and more accessible) analysis which reduces the hidden constants to $39(C + D)$. More recently Peis and Wiese [PW11] reduced the constant to 24 (and beyond, for larger minimum bandwidth or transit time).

Already the original paper of [LMR94] also showed that (huge) constant size *edge buffers* are sufficient. Scheideler [Sch98] proved that even a buffer size of 2 is enough. However, all proofs [LMR94, LMR99, Sch98, PW11] use the original idea of Leighton, Maggs and Rao to start with an infeasible schedule and insert iteratively random delays to reduce the infeasibility until no more than $O(1)$ packets use an edge per time step (in each iteration, applying the Lovász Local Lemma).

In this paper, we suggest a somewhat dual approach in which we start with a probabilistic schedule which is feasible in expectation and then reduce step by step the randomness (still making use of the Local Lemma). Our construction here is not fundamentally different from the original work of [LMR94], but the emerging proof is "less iterative" and, in the opinion of the author, also more clear and explicit in demonstrating to the reader *why* a constant factor suffices. Especially obtaining the additional property of constant size edge buffers is fairly simple in our construction.

If it comes to lower bounds for general routing strategies, the following instance is essentially the worst known one: C many packets share the same path of length D. Then it takes C time units until the last packet crosses the first edge; that packet needs $D - 1$ more time units to reach its destination, leading to a makespan of $C + D - 1$. Wiese [Wie11]) states that no example is known where the optimum makespan needs to be even a small constant factor larger. We answer the open question in [Wie11] and show that for a universal constant $\varepsilon > 0$, there is a family of instances in which every routing policy needs at least $(1 + \varepsilon) \cdot (C + D)$ time units (and $C, D \to \infty$)[1]. In our chosen instance, we generate paths from random permutations and use probabilistic arguments for the analysis.

[1] The constant can be chosen e.g. as $\varepsilon := 0.00001$, though we do not make any attempt to optimize the constant, but focus on a simple exposition.

1.1 Related Work

The result of [LMR94, LMR99] could be interpreted as a constant factor approximation algorithm for the problem of finding the minimum makespan. In contrast, finding the optimum schedule is **NP**-hard [CI96]. In fact, even on trees, the problem remains **APX**-hard [PSW09]. If we generalize the problem to finding paths plus schedules, then constant factor approximation algorithms are still possible due to Srinivasan and Teo [ST00] (using the fact that it suffices to find paths that minimize the sum of congestion and dilation). Koch et al. [KPSW09] extend this to a more general setting, where messages consisting of several packets have to be sent.

The Leighton-Maggs-Rao result, apart from being quite involved, has the disadvantage of being a non-local offline algorithm. In contrast, there is a distributed algorithm with makespan $O(C) + (\log^* n)^{O(\log^* n)} D + \log^{O(1)} n$ by Rabani and Tardos [RT96] which was later improved to $O(C + D + \log^{1+\varepsilon} n)$ by Ostrovsky and Rabani [OR97]. If the paths are indeed shortest paths, then there is a randomized online routing policy which finishes in $O(C + D + \log k)$ steps [MV99]. To the best of our knowledge, the question concerning the existence of an $O(C + D)$ online algorithm is still open. We refer to the book of Scheideler [Sch98] for a more detailed overview about routing policies.

One can also reinterpret the packet routing problem as *(acyclic) job shop scheduling* $J \mid p_{ij} = 1, \text{acyclic} \mid C_{\max}$, where jobs J and machines M are given. Each job has a sequence of machines that it needs to be processed on in a given order (each machine appears at most once in this sequence), while all processing times have unit length. For the natural generalization $(J \mid p_{ij}, \text{acyclic} \mid C_{\max})$ with arbitrary processing times p_{ij}, Feige & Scheideler [FS02] showed that schedules of length $O(L \cdot \log L \cdot \log\log L)$ are always possible and for some instances, every schedule needs at least $\Omega(L \cdot \frac{\log L}{\log\log L})$ time units, where we abbreviate $L := \max\{C, D\}$.[2] Svensson and Mastrolilli [MS11] showed that this lower bound even holds in the special case of *flow shop scheduling*, where all jobs need to be processed on all machines in the same order (in packet routing, this corresponds to the case that all paths P_i are identical). In fact, for *flow shop scheduling with jumps* (i.e. each job needs to be processed on a given subset of machines) it is even **NP**-hard to approximate the optimum makespan within any constant factor [MS11].

In contrast, if we allow preemption, then even for acyclic job shop scheduling, the makespan can be reduced to $O(C + D \log\log \max_{ij} p_{ij})$ [FS02] and it is conceivable that even $O(C + D)$ might suffice.

1.2 Organisation

In Section 2, we recall some probabilistic tools. Then in Section 3 we show the existence of an $O(C + D)$ routing policy, which is modified in Section 4

[2] In this setting, one extends $C = \max_{i \in M} \sum_{j \in J : j \text{ uses } i} p_{ij}$ and $D = \max_{j \in J} \sum_{i \in M : j \text{ uses } i} p_{ij}$.

to guarantee that constant size edge buffers suffice. Finally, we show the lower bound in Section 5.

2 Preliminaries

Later, we will need the following concentration result, which is a version of the *Chernov-Hoeffding bound*:

Lemma 1 ([DP09, Theorem 1.1]). *Let $Z_1, \dots, Z_k \in [0, \delta]$ be independently distributed random variables with sum $Z := \sum_{i=1}^{k} Z_i$ and let $\mu \geq \mathbb{E}[Z]$. Then for any $\varepsilon > 0$,*

$$\Pr[Z > (1+\varepsilon)\mu] \leq \exp\Big(-\frac{\varepsilon^2}{3} \cdot \frac{\mu}{\delta}\Big).$$

Moreover, we need the *Lovász Local Lemma* (see also the books [AS08] and [MU05] and for the constructive version, see [MT10]).

Lemma 2 (Lovász Local Lemma [EL75]). *Let $A_1, \dots, A_m$ be arbitrary events such that (1) $\Pr[A_i] \leq p$; (2) each A_i depends on at most d many other events; and (3) $4 \cdot p \cdot d \leq 1$. Then $\Pr\left[\bigcap_{i=1}^{m} \bar{A}_i\right] > 0$.*

3 O(Congestion + Dilation) Routing

After adding dummy paths and edges, we may assume that $C = D$ and every path has length exactly D. In the following we show how to route the packets within $O(D)$ time units such that in each time step, each edge is traversed by at most $O(1)$ many packets (by stretching the time by another $O(1)$ factor, one can obtain a schedule with makespan $O(D)$ in which each edge is indeed only traversed by a single packet). In the following, we call the largest number of packets that traverse the same edge in one time unit the *load* of the schedule.

Let $\Delta > 0$ be a constant that we leave undetermined for now – at several places we will simply assume Δ to be large enough for our purpose. Consider a packet i and partition its path P_i into a laminar family of *blocks* such that the blocks on *level* ℓ contain $D_\ell = D^{(1/2)^\ell}$ many consecutive edges.[3] We stop this dissection, when the last block (whose index we denote by L) has length between Δ and Δ^2.

In other words, the root block (i.e. the path P_i itself) is on level 0 and the depth of that laminar family is $L = \Theta(\log \log D)$ (though this quantity will be irrelevant for the analysis). Each block has 2 *boundary nodes*, a *start node* and an *end node*. Observe that a level ℓ block of length D_ℓ has children of length $D_{\ell+1} = \sqrt{D_\ell}$. Moreover, we define

$$W_\ell := \begin{cases} D_\ell & \ell = 0 \\ D_\ell^{1/4} & \ell \geq 1 \end{cases}$$

[3] Depending on D, the quantity D_ℓ may not be integral. But all our calculations have enough slack so that one could replace D_ℓ with the nearest power of 2. Then we may also assume that for each ℓ, D_ℓ divides $D_{\ell-1}$.

The routing policy for packet i is now as follows: For each level ℓ block, the packet waits a uniformly and independently chosen random time $x \in [1, W_\ell]$ at the start node[4]; furthermore the packet waits $W_\ell - x$ time units at the end node. This policy has two crucial properties:

(A) The total waiting time of each packet is $O(D)$.
(B) The time t at which packet i crosses an edge $e \in P_i$ is a random variable that depends only on the random waiting times of the blocks that contain e — in fact, i.e. only one block from each level. More precisely, let $\alpha(i, e, \ell)$ be the random waiting time for the unique block of P_i that is on level ℓ and contains $e \in P_i$; then the time in which packet i crosses e is a sum of the form $C(i, e) + \sum_{\ell=0}^{L} \alpha(i, e, \ell)$ for some constant $C(i, e)$.

Let us argue, why (A) is true. The waiting time on level $\ell = 0$ will be precisely D, while for each $\ell \geq 1$ the total level-ℓ waiting time for each packet will be $\frac{D}{D_\ell} \cdot W_\ell = \frac{D}{D_\ell^{3/4}}$. Using the crude bound $D_\ell \geq 4 \cdot D_{\ell+1}$ we have $D_{L-j} \geq 4^j$, hence on level $L - j > 0$, the total waiting time will be at most $\frac{D}{D_{L-j}^{3/4}} \leq \frac{D}{2^j}$. Thus the total waiting time for a packet, summed over all levels is at most $D + D\sum_{j=0}^{L-1}(\frac{1}{2})^j = O(D)$. In other words: each packet is guaranteed to arrive after at most $T := O(D)$ time units. Note that there are instances where the vast majority of random outcomes would yield a superconstant load on some edge. However, one can prove that there *exists* a choice of the waiting times such that the load does not exceed $O(1)$.

Let $X(e, t, i) \in \{0, 1\}$ be the random variable that tells us whether packet i is crossing edge e at time t. Moreover, let $X(e, t) = \sum_{i=1}^{k} X(e, t, i)$ be the number of packets crossing e at time t. Since packet i waits a random time from $[1, D]$ in s_i, we have $\Pr[X(e, t, i)] \leq \frac{1}{D}$ for each e, i, t (more formally: no matter how the waiting times on level ≥ 1 are chosen, there is always at most one out of D outcomes for the level 0 waiting time that cause packet i to cross e precisely at time t). Since no edge is contained in more than D paths, we have $\mathbb{E}[X(e, t)] \leq 1$.

In the following, if $\boldsymbol{\alpha} \in [1, W_\ell]^{D/D_\ell}$ is a vector of level ℓ-waiting times, then $\mathbb{E}[X(e, t) \mid \boldsymbol{\alpha}]$ denotes the corresponding conditional expectation, depending on $\boldsymbol{\alpha}$. The idea for the analysis is to fix the waiting times on one level at a time (starting with level 0) such that the conditional expectation $\mathbb{E}[X(e, t)]$ never increases to a value larger than, say 2. Before we continue, we want to be clear about the behaviour of such conditional random variables.

Lemma 3. *Let $\ell \in \{0, \ldots, L-1\}$ and condition on arbitrary waiting times for level $0, \ldots, \ell$. Then for any packet i, edge $e \in E$ and any time $t \in [T]$ one has*

a) $\Pr[X(e, t, i)] \leq \frac{1}{W_{\ell+1}}$.
b) If the event $X(e, t, i)$ has non-zero probability, then $\Pr[X(e, t, i)] \geq \frac{1}{W_{\ell+1}^2}$.

Proof. For (a), suppose also all waiting times except of the level $\ell + 1$ block in which i crosses e are fixed adversarially. Still, there is at most one out of $W_{\ell+1}$ outcomes that cause packet i to cross e at time t.

[4] We define $[a, b] := \{a, a + 1, a + 2, \ldots, b\}$ as the set of integers between a and b.

For (b), observe that the time at which packet i crosses e depends only on the waiting time of the blocks that contain e (i.e. one block per level). The number of possible outcomes of those waiting times is bounded by $\prod_{j=0}^{L-\ell-1} W_{\ell+1+j} \leq (W_{\ell+1})^{\sum_{j\geq 0}(1/2)^j} = W_{\ell+1}^2$. □

The whole analysis boils down to the following lemma, in which we prove that we can always fix the waiting times on level ℓ without increasing the expected load on any edge by more than $D_\ell^{-1/32}$. What happens formally is that we show the existence of a sequence $\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_{L-1}$ such that $\boldsymbol{\alpha}_\ell$ denotes a vector of level ℓ-waiting times and

$$\mathbb{E}[X(e,t) \mid \boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_{\ell-1}, \boldsymbol{\alpha}_\ell] \leq \mathbb{E}[X(e,t) \mid \boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_{\ell-1}] + \frac{1}{D_\ell^{1/32}} \quad \forall e \in E\ \forall t \in [T] \quad (1)$$

(given that the right hand side is at least 1). To do this, suppose we already found and fixed proper waiting times $\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_{\ell-1}$. Then one can interpret the left hand side of (1) as a random variable depending on $\boldsymbol{\alpha}_\ell$, which is the sum of independently distributed values — and hence well concentrated. Moreover the dependence degree of this random variable is bounded by a polynomial in D_ℓ. Thus the Lovász Local Lemma provides the existence of suitable waiting times $\boldsymbol{\alpha}_\ell$.

Lemma 4. *Let $\ell \in \{0, \ldots, L-1\}$ and suppose that we already fixed all waiting times on level $0, \ldots, \ell-1$. Let $X(e,t)$ be the corresponding conditional random variable and assume $\gamma \geq \max_{e\in E, t\in[T]}\{\mathbb{E}[X(e,t)]\}$ and $1 \leq \gamma \leq 2$. Then there are level ℓ waiting times $\boldsymbol{\alpha}$ such that*

$$\mathbb{E}[X(e,t) \mid \boldsymbol{\alpha}] \leq \gamma + \frac{1}{D_\ell^{1/32}} \quad \forall e \in E\ \forall t \in [T]$$

Proof. We abbreviate $m := D_\ell$. First recall that on level ℓ, (1) blocks have length m; (2) the child blocks have length $\sqrt{m}$ and (3) the waiting time on the next level $\ell+1$ is from $[1, m^{1/8}]$.

We define $Y(e,t) := \mathbb{E}[X(e,t) \mid \boldsymbol{\alpha}]$ and consider $Y(e,t)$ as a random variable only depending on $\boldsymbol{\alpha}$. Since the waiting times on levels $0, \ldots, \ell-1$ are already fixed, we know exactly the level ℓ-block in which packet i will cross edge e — let $\alpha_{i,e}$ be the random waiting time for that block. Then we can write

$$Y(e,t) = \sum_{i=1}^{k} \Pr[X(e,t,i) \mid \alpha_{i,e}] \quad (2)$$

By Lemma 3.(b), we know that $\Pr[X(e,t,i) \mid \alpha_{i,e}] \leq \frac{1}{m^{1/8}}$ for every choice of $\alpha_{i,e}$. Thus $Y(e,t)$ is the sum of independent random variables in the interval $[0, m^{-1/8}]$ and the Chernov bound (Lemma 1) provides

$$\Pr\left[Y(e,t) > \gamma + \frac{1}{m^{1/32}}\right] \leq \exp\left(-\frac{1}{3} \cdot \frac{1}{(2m^{1/32})^2} \cdot m^{1/8}\right) \leq e^{-m^{1/16}/12}$$

Now we want to apply the Lovász Local Lemma for the events "$Y(e,t) > \gamma + m^{-1/32}$" to argue that it is possible that none of the events happens. So it suffices

to bound the dependence degree by a polynomial in m. Lemma 3.(b) guarantees that if the event $X(e,t,i)$ is possible at all, then $\Pr[X(e,t,i)] \geq \frac{1}{W_\ell^2} \geq \frac{1}{m}$. Now, reconsider Equation (2) and let $Q(e,t) := \{i \in [k] \mid \Pr[X(e,t,i) > 0\}$ be the set of packets that still have a non-zero chance to cross edge e at time t. Taking expectations of Equation (2), we see that

$$2 \geq \gamma \geq \mathbb{E}[Y(e,t)] = \sum_{i \in Q(e,t)} \Pr[X(e,t,i)] \geq |Q(e,t)| \cdot \frac{1}{m}$$

and hence $|Q(e,t)| \leq 2m$. This means that each random variable $Y(e,t)$ depends on at most $2m$ entries of $\boldsymbol{\alpha}$. Moreover, consider an entry in $\boldsymbol{\alpha}$, say it belongs to packet i and block B. This random variable appears in the definition of $Y(e,t)$ if $e \in B$ and t belongs to B's time frame – these are just $m \cdot O(m)$ many combinations. Here we use that the time difference between entering a level ℓ block and leaving it, is bounded by $O(D_\ell)$. Overall, the dependence degree is $O(m^3)$. Since the probability of each bad event "$Y(e,t) > \gamma + m^{-1/32}$" is superpolynomially small, the claim follows by the Lovász Local Lemma and the assumption that $m \geq \Delta$ is large enough. □

We apply this lemma for $\ell = 0, \ldots, L-1$ and the maximum load after any iteration will be bounded by $1 + \sum_{\ell=0}^{L-1} (D_\ell)^{-1/32} \leq 2$ for Δ large enough. The finally obtained random variables $X(e,t,i)$ are *almost* deterministic — just the waiting times on level L are still probabilistic. But again by Lemma 3, all non-zero probabilities $\Pr[X(e,t,i)]$ are at least $\frac{1}{(\Delta^{1/4})^2} = \Omega(1)$, thus making an arbitrary choice for them cannot increase the load by more than a constant factor. Finally, we end up with a schedule with load $O(1)$.

4 Providing Constant Size Edge Buffers

Now let us imagine that each directed edge $(u,v) \in E$ has an *edge buffer* at the beginning of the edge. Whenever a packet arrives at node u and has e as next edge on its path, the packet waits in e's edge buffer. But a packet i is still allowed to wait an arbitrary amount of time in s_i or t_i.

In the construction that we saw above, it may happen that many packets wait for a long time in one node, i.e. a large edge buffer might be needed. However, as was shown by Leighton, Maggs and Rao [LMR94], one can find a schedule such that edge buffers of size $O(1)$ suffice. More precisely, [LMR94] found a schedule with load $O(1)$ in which each packet waits at most one time unit in every node — after stretching, this results in a schedule with load 1 and $O(1)$ buffer size.

In fact, we can modify the construction from Section 3 in such a way that we *spread* the waiting time over several edges and obtain the same property. Consider the dissection from the last section. Iteratively, for $\ell = 1, \ldots, L$, shift the level ℓ-blocks such that every level $\ell - 1$ boundary node lies in the middle of some level ℓ-block (note that we assume that $D_{\ell-1}$ is an integral multiple of D_ℓ). Fix a packet i and denote the edges of its path by $P_i = (e_1, \ldots, e_D)$, then

we assign all edges e_j whose index j is of the form $(1 + 2\mathbb{Z}) \cdot 2^q$ to level $L - q$ (for $q \in \{0, \ldots, L-1\}$). For example, this means that all odd edges are assigned to the last level; the top level does not get assigned any edges.

Now we again define random waiting times for packet i and a block B: on level $\ell \geq 1$, each block picks a uniform random number $x \in [1, W_\ell]$. The packet waits on each of the first x edges that are assigned to the block. Moreover, it waits on each of the last $W_\ell - x$ edges that are assigned to the block. Observe that regardless of the random outcome, the packet will wait at most once per edge since edges are assigned to at most one level. Using the convenient bound $2^{L-\ell} \leq D_\ell^{1/8}$ for Δ large enough, we see that all level-ℓ randomization takes place within the first and last $D_\ell^{3/8}$ edges of each block.

The top block does not get assigned any edge, so instead for each packet i, we pick a value $x \in [1, D]$ at random and wait x time units in s_i.[5]

Reinspecting Lemma 3, we observe that Lemma 3.*b*) holds without any alterations and Lemma 3.*a*) holds as long as the considered edge e has a minimum distance of $D_{\ell+1}^{3/8}$ from the nearest level $\ell + 1$ boundary node. Surprisingly, also Lemma 4 still holds with a minor modification in the claimed bound.

Lemma 5. *Let $\ell \in \{0, \ldots, L-2\}$ and suppose that we already fixed all waiting times on level $0, \ldots, \ell - 1$. Let $X(e,t)$ be the corresponding conditional random variables and assume $\gamma \geq \max_{e \in E, t \in [T]} \{\mathbb{E}[X(e,t)]\}$ and $1 \leq \gamma \leq 2$. Then there are level ℓ waiting times $\boldsymbol{\alpha}$ such that*

$$\mathbb{E}[X(e,t) \mid \boldsymbol{\alpha}] \leq \gamma + \frac{1}{D_\ell^{1/64}} \quad \forall e \in E \; \forall t \in [T]$$

Proof. Again abbreviate $m := D_\ell$ and consider

$$Y(e,t) := \mathbb{E}[X(e,t) \mid \boldsymbol{\alpha}] = \sum_{i=1}^{k} \Pr[X(e,t,i) \mid \alpha_{i,e}]$$

as a random variable only depending on $\boldsymbol{\alpha}$ (recall that $\alpha_{i,e}$ is the random waiting time for that level ℓ-block in which packet i crosses edge e).

For a fixed edge e, for one of those levels $\ell' \in \{\ell + 1, \ell + 2\}$, the edge e is at least $\frac{1}{4}m^{1/4}$ edges away from the next level ℓ' boundary node. Consider the level ℓ'-block B that contains e. As already argued, all randomization takes place on the first and last $D_{\ell'}^{3/8} \leq D_{\ell+1}^{3/8} = m^{3/16} \ll \frac{1}{4}m^{1/4}$ edges (for $m \geq \Delta$ large enough). So we can still apply Lemma 3.*a*) for level ℓ' to obtain $\Pr[X(e,t,i) \mid \alpha_{i,e}] \leq \frac{1}{W_{\ell'}} \leq \frac{1}{m^{1/16}}$. Again by the Chernov bound (i.e. Lemma 1 with $\delta := \frac{1}{m^{1/16}}$, $\varepsilon := \frac{1}{2m^{1/64}}$, $\mu := \gamma \geq 1$) we have

$$\Pr\left[Y(e,t) > \gamma + \frac{1}{m^{1/64}}\right] \leq \exp\Big(-\frac{1}{3} \cdot \frac{1}{(2m^{1/64})^2} \cdot m^{1/16}\Big) = e^{-m^{1/32}/12}$$

[5] If for a block, due to the shifting, some or all waiting edges are shifted "before" the source s_i, then the packet just waits the missing time in s_i.

Next, note that still $\Pr[X(e,t,i) \mid \alpha_{i,e}] \geq \frac{1}{m}$, given that this probability is positive. Thus from now on we can follow the arguments in the proof of Lemma 4. The dependence degree is still bounded by $O(m^3)$, thus the claim follows by the Lovász Local Lemma since $4 \cdot O(m^3) \cdot e^{-m^{1/32}/12} \leq 1$ for $m \geq \Delta$ large enough. □

Again, we have initially $\mathbb{E}[X(e,t)] \leq 1$ for all e and t, then we fix the waiting times iteratively on level $0, \ldots, L-2$ using Lemma 5 and make an arbitrary choice for the waiting times of level $L-1$ and level L. This results in a schedule of length $O(D)$ and load $O(1)$, in which packets wait at most one time unit before entering an edge.

5 A $(1+\varepsilon) \cdot (C+D)$ Lower Bound

In this section, we prove that there is an instance in which the optimum makespan must be at least $(1+\varepsilon) \cdot (C+D)$, where $\varepsilon > 0$ is a small constant. The definition of the graph $G = (V, E)$ will follow from the choice of the paths that we will make in a second. Edges $e_i = (u_i, v_i)$ are called *critical* edges, while we term (v_i, u_j) *back edges*. We want to choose paths $P_1, \ldots, P_n$ as random paths though the network, all starting at $s_i := s$ and ending at $t_i := t$. More concretely, each packet i picks a uniform random permutation $\pi_i : [n] \to [n]$ which gives the order in which it moves through the critical edges $e_1, \ldots, e_n$. In other words,

$$P_i = (s, s', u_{\pi_i(1)}, v_{\pi_i(1)}, u_{\pi_i(2)}, v_{\pi_i(2)}, \ldots, , u_{\pi_i(n)}, v_{\pi_i(n)}, u_{n+1}, t).$$

Then the congestion is n and the dilation is $2n+3$. We consider the time frame $[1, T]$ with $T = (3+\varepsilon)n$ and claim that for $\varepsilon > 0$ small enough, there will be no valid routing that is finished by time T.

Theorem 1. *Pick paths $P_1, \ldots, P_n$ at random. Then with probability $1 - e^{-\Omega(n^2)}$, there is no packet routing policy with makespan at most $3.000032n$ (even if buffers of unlimited size are used).*

First of all, clearly the makespan must be at least $C + D - 1 \approx 3n$ since all paths have the same length D and all packets must first cross edge (s, s'). So if we allow only time $(3+\varepsilon)n$, then there is only a small slack of εn time units. One can show that the number of different possible routing strategies is bounded by $2^{o(n^2)}$ (for $\varepsilon \to 0$). In contrast, we can argue that a *fixed* routing will fail against random paths with probability $2^{-\Omega(n^2)}$. Then choosing ε small enough, the theorem follows using the union bound over all routing strategies.

We call a packet i *active* at time τ if it is traversing an edge. We say a packet is *parking* at time τ if it is either in the end node t_i nor in the start node s_i. We say a packet is *waiting* if it is neither active nor parking.

5.1 The Number of Potential Routing Strategies

Consider a fixed packet i and let us discuss, how a routing strategy is defined. The only decision that is made, is of the form: "*How many time units shall the*

packet wait in the k-th node on its path (for $k = 0, \ldots, D$)". It is not necessary to wait in s' since a packet could instead move to $u_{\pi_i(1)}$ and wait there. Moreover, it is not needed to wait in one of the nodes v_j, since instead it could also wait in the next $u_{j'}$ node on its way (the reason is that if there would be a collision on a back edge $(v_{\pi_i(j)}, u_{\pi_i(j+1)})$ with packet $i' \neq i$, then this packet i' has crossed the critical edge $(u_{\pi_i(j)}, v_{\pi_i(j)})$ together with i in the previous time step, so there was already a collision). In other words, the complete routing strategy for packet i can be described as a $(n+2)$-dimensional vector $W_i \in \mathbb{Z}_{\geq 0}^{n+2}$, where W_{ij} is the time that packet i stays in node u_j (for convenience, we denote s also as u_0). Then $\sum_{j=1}^{n+1} W_{ij}$ is the total waiting time and for $i \in [n]$ and W_{i0} is the time that i parks in the start node.

Independently from the outcome of the random experiment, we know the time when each packet crosses the edges incident to s and to t. We call W a *candidate routing strategy*, if there is no collision on (s, s') and (u_{n+1}, t) and the makespan of each packet is bounded by $(3+\varepsilon)n$.

Recall that $H(\delta) = \delta \log \frac{1}{\delta} + (1-\delta) \log \frac{1}{1-\delta}$ is the *binary entropy function*[6]. Then we have:

Lemma 6. *The total number of candidate routing matrices W is at most $2^{(\Phi(\varepsilon)+o(1))\cdot n^2}$, where $\Phi(\varepsilon) := H(\frac{\varepsilon}{1+\varepsilon}) \cdot (1+\varepsilon)$.*

Proof. First of all, the parking times in s and the total waiting time $\sum_{j=1}^{n+1} W_{ij}$ for a packet i are between 0 and $(1+\varepsilon)n \leq 2n$; thus there are at most $(2n)^{2n} = 2^{o(n^2)}$ many possibilities to choose them.

Thus assume that the total waiting time $\varepsilon_i n = \sum_{j=1}^{n+1} W_{ij}$ for packet i is fixed. Then the number of possibilities how this waiting time can be distributed among nodes $u_1, \ldots, u_{n+1}$ is bounded by

$$\binom{(n+1) + (\varepsilon_i n) - 1}{\varepsilon_i n} \leq 2^{H(\frac{\varepsilon_i}{1+\varepsilon_i}) \cdot (1+\varepsilon_i) \cdot n} = 2^{\Phi(\varepsilon_i) \cdot n}$$

where we use the bound $\binom{m}{\delta m} \leq 2^{H(\delta)m}$ with $m = (1+\varepsilon_i)n$ and $\delta = \frac{\varepsilon_i}{1+\varepsilon_i}$.

Next, let us upperbound the total waiting time $n \sum_{i=1}^{n} \varepsilon_i$. Of course, the waiting time must fit into the time frame of length $T = (3+\varepsilon)n$. Since edge (s, s') can only be crossed by one packet at a time, the cumulated time that the packets spend in the start node is at least $\sum_{\tau=0}^{n-1} \tau \approx \frac{n^2(1-o(1))}{2}$. The same amount of time is spent by all packets in the end node. Moreover, the packets spend at least $2n^2$ time units traversing edges. We conclude that

$$n \sum_{i=1}^{n} \varepsilon_i \leq nT - \frac{n^2(1-o(1))}{2} - \frac{n^2(1-o(1))}{2} - 2n^2 = (\varepsilon + o(1))n^2,$$

thus $\sum_{i=1}^{n} \varepsilon_i \leq (\varepsilon + o(1))n$. Once the values $\varepsilon_1, \ldots, \varepsilon_n$ are fixed, the total number of routing policies for the n packets is hence upperbounded by

$$\prod_{i=1}^{n} 2^{\Phi(\varepsilon_i)n} = 2^{n \sum_{i=1}^{n} \Phi(\varepsilon_i)} \leq 2^{n^2 \Phi(\frac{1}{n} \sum_{i=1}^{n} \varepsilon_i)} \leq 2^{n^2(\Phi(\varepsilon)+o(1))}$$

[6] Here log is the binary logarithm.

Here we use Jensen's inequality together with the fact that Φ is concave. The claim follows. □

The important property of function Φ apart from concavity is that $\lim_{\varepsilon \to 0} \Phi(\varepsilon) = 0$. Note that for $0 \leq \varepsilon \leq \frac{1}{10}$, one can conveniently upperbound $\Phi(\varepsilon) \leq 2^{1.5\varepsilon \log(\frac{1}{\varepsilon})n}$.

5.2 A Fixed Strategy vs. Random Paths

Now consider a *fixed* candidate routing matrix W and imagine that the paths are taken at random. We will show that this particular routing matrix W is not legal with probability $1 - e^{\Omega(n^2)}$. For this sake, we observe that there must be $\Omega(n)$ time units in which at least a constant fraction of packets cross critical edges. For each such time unit the probability of having no collision is at most $(\frac{1}{2})^{\Omega(n)}$ and the claim follows. The only technical difficulty lies in the fact that the outcomes of values $\pi_i(j)$ and $\pi_i(j')$ for the random permutations are (mildly) dependent.

Lemma 7. *Suppose $\varepsilon \leq \frac{1}{20}$. Let W be a candidate routing matrix. Then take paths $P_1, \ldots, P_n$ at random. The probability that the routing scheme defined by W is collision-free is at most $(\frac{15}{16})^{n^2/128}$.*

Proof. For time τ, let $\beta_\tau n$ be the number of packets that cross one of the critical edges at time τ, thus $\sum_{\tau=1}^{T} \beta_\tau = n$ (note that the β_τ's do not depend on the random experiment). Let $p := \Pr_{\tau \in [T]}[\beta_\tau \geq \frac{1}{4}]$ be the fraction of time units in which at least $\frac{n}{4}$ packets are crossing a critical edge. Then

$$\frac{1}{3+\varepsilon} = \frac{\sum_{\tau=1}^{T} \beta_\tau}{T} = \mathop{\mathbb{E}}_{\tau \in [T]}[\beta_\tau] \leq 1 \cdot p + (1-p) \cdot \frac{1}{4},$$

which can be rearranged to $p \geq \frac{1}{10}$ for $\varepsilon \leq \frac{1}{20}$. In other words, we have $\frac{T}{10} \geq \frac{1}{16}n =: k$ many time units $\boldsymbol{\tau} = \{\tau_1, \ldots, \tau_k\}$ in which at least $\frac{n}{4}$ many packets are crossing an edge in $e_1, \ldots, e_n$. Let $A(\tau)$ be the event that there is no collision at time τ. Then we can bound the probability of having no collision at all, by just considering the time units in $\boldsymbol{\tau}$:

$$\Pr\Big[\bigwedge_{\tau=1}^{T} A(\tau)\Big] \leq \prod_{j=1}^{k} \Pr[A(\tau_j) \mid A(\tau_1), \ldots, A(\tau_{j-1})] \overset{(*)}{\leq} \left(\frac{15}{16}\right)^{\frac{n}{8} \cdot k} = \left(\frac{15}{16}\right)^{n^2/128}$$

It remains to justify the inequality $(*)$.

Claim. For all $j = 1, \ldots, k$ one has $\Pr[A(\tau_j) \mid A(\tau_1), \ldots, A(\tau_{j-1})] \leq (\frac{15}{16})^{n/8}$.

By $P_i(\tau)$ we denote the random variable that gives the edge that i traverses at time τ (in case that i is waiting in a node v, let's say that $P_i(\tau) = (v, v)$). Let $E_i := \{P_i(\tau_1), \ldots, P_i(\tau_{j-1})\} \cap \{e_1, \ldots, e_n\}$ be the critical edges that packet i has visited at $\tau_1, \ldots, \tau_{j-1}$. It suffices to show that $\Pr[A(\tau_j) \mid E_1, \ldots, E_n] \leq (\frac{15}{16})^{n/16}$, i.e. we condition on those edges E_i. Let $I \subseteq [n]$ with $|I| = \frac{n}{4}$ be the indices of

packets that cross a critical edge at time τ_j. We split I into equally sized parts $I = I_1 \dot\cup I_2$, i.e. $|I_1| = |I_2| = \frac{n}{8}$. Consider the critical edges $E^* := \{P_i(\tau_j) \mid i \in I_1\}$ which are chosen by packets in I_1. If $|E^*| < \frac{n}{8}$ then we have a collision, so condition on the event that $|E^*| = \frac{n}{8}$. Now for all other packets $i \in I_2$, the edge $P_i(\tau_j)$ is a uniform random choice from $\{e_1, \ldots, e_n\} \backslash E_i$. Thus we have independently for all $i \in I_2$,

$$\Pr[P_i(\tau_j) \in E^*] = \frac{|E^* \backslash E_i|}{|\{e_1, \ldots, e_n\} \backslash E_i|} \geq \frac{n/8 - n/16}{n} = \frac{1}{16},$$

since $|E_i| \leq k = \frac{n}{16}$. Thus

$$\Pr\left[A(\tau_j) \mid |E^*| = \frac{n}{8};\ E_1, \ldots, E_n\right] \leq \Pr\Big[\bigwedge_{i \in I_2} P_i(\tau_j) \notin E^* \mid |E^*| = \frac{n}{8};\ E_1, \ldots, E_n\Big] \leq \left(\frac{15}{16}\right)^{n/8}$$

and the claim follows. □

Finally one can check that for $\varepsilon := 0.000032$ and n large enough one has $\left(\frac{15}{16}\right)^{n^2/128} \cdot 2^{(\Phi(\varepsilon)+o(1))n^2} < 1$ and Theorem 1 follows.

Observe that in our instance, C and D are within a factor of 2 or each other. In contrast, if $C \gg D$, then there is a schedule of length $(1 + o(1)) \cdot C$ and buffer size $O(\frac{C}{D})$, see [Sch98, Chapter 6].

Acknowledgements. The author is very grateful to Rico Zenklusen for carefully reading a preliminary draft.

References

[AS08] Alon, N., Spencer, J.H.: The probabilistic method, 3rd edn. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons Inc., Hoboken (2008); With an appendix on the life and work of Paul Erdős

[CI96] Clementi, A.E.F., Di Ianni, M.: On the hardness of approximating optimum schedule problems in store and forward networks. IEEE/ACM Trans. Netw. 4(2), 272–280 (1996)

[DP09] Dubhashi, D.P., Panconesi, A.: Concentration of measure for the analysis of randomized algorithms. Cambridge University Press, Cambridge (2009)

[EL75] Erdős, P., Lovász, L.: Problems and results on 3-chromatic hypergraphs and some related questions. In: Infinite and Finite Sets (Colloq., Keszthely, 1973; Dedicated to P. Erdős on his 60th Birthday). Colloq. Math. Soc. János Bolyai, vol. II, 10, pp. 609–627. North-Holland, Amsterdam (1975)

[FS02] Feige, U., Scheideler, C.: Improved bounds for acyclic job shop scheduling. Combinatorica 22(3), 361–399 (2002)

[KPSW09] Koch, R., Peis, B., Skutella, M., Wiese, A.: Real-Time Message Routing and Scheduling. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX and RANDOM 2009. LNCS, vol. 5687, pp. 217–230. Springer, Heidelberg (2009)

[LMR94] Leighton, F.T., Maggs, B.M., Rao, S.B.: Packet routing and job-shop scheduling in O(congestion + dilation) steps. Combinatorica 14(2), 167–186 (1994)

[LMR99] Leighton, T., Maggs, B., Richa, A.W.: Fast algorithms for finding O (congestion + dilation) packet routing schedules. Combinatorica 19(3), 375–401 (1999)

[MS11] Mastrolilli, M., Svensson, O.: Hardness of approximating flow and job shop scheduling problems. J. ACM 58(5), 20 (2011)

[MT10] Moser, R.A., Tardos, G.: A constructive proof of the general lovász local lemma. J. ACM 57(2) (2010)

[MU05] Mitzenmacher, M., Upfal, E.: Probability and computing. Randomized algorithms and probabilistic analysis. Cambridge University Press, Cambridge (2005)

[MV99] Meyer auf der Heide, F., Vöcking, B.: Shortest-path routing in arbitrary networks. J. Algorithms 31(1), 105–131 (1999)

[OR97] Ostrovsky, R., Rabani, Y.: Universal o(congestion + dilation + $\log^{1+\text{epsilon}} n$) local control packet switching algorithms. In: STOC, pp. 644–653 (1997)

[PSW09] Peis, B., Skutella, M., Wiese, A.: Packet Routing: Complexity and Algorithms. In: Bampis, E., Jansen, K. (eds.) WAOA 2009. LNCS, vol. 5893, pp. 217–228. Springer, Heidelberg (2010)

[PW11] Peis, B., Wiese, A.: Universal Packet Routing with Arbitrary Bandwidths and Transit Times. In: Günlük, O., Woeginger, G.J. (eds.) IPCO 2011. LNCS, vol. 6655, pp. 362–375. Springer, Heidelberg (2011)

[RT96] Rabani, Y., Tardos, É.: Distributed packet switching in arbitrary networks. In: STOC, pp. 366–375 (1996)

[Sch98] Scheideler, C.: Universal Routing Strategies for Interconnection Networks. LNCS, vol. 1390. Springer, Heidelberg (1998)

[ST00] Srinivasan, A., Teo, C.: A constant-factor approximation algorithm for packet routing and balancing local vs. global criteria. SIAM J. Comput. 30(6), 2051–2068 (2000)

[Wie11] Wiese, A.: Packet Routing and Scheduling. Dissertation, TU Berlin (2011)

0/1 Polytopes with Quadratic Chvátal Rank

Thomas Rothvoß[1,⋆] and Laura Sanità[2]

[1] MIT, Boston, USA
rothvoss@math.mit.edu
[2] University of Waterloo, Canada
lsanita@uwaterloo.ca

Abstract. For a polytope P, the *Chvátal closure* $P' \subseteq P$ is obtained by simultaneously strengthening all feasible inequalities $cx \leq \beta$ (with integral c) to $cx \leq \lfloor \beta \rfloor$. The number of iterations of this procedure that are needed until the integral hull of P is reached is called the *Chvátal rank*. If $P \subseteq [0,1]^n$, then it is known that $O(n^2 \log n)$ iterations always suffice (Eisenbrand and Schulz (1999)) and at least $(1 + \frac{1}{e} - o(1))n$ iterations are sometimes needed (Pokutta and Stauffer (2011)), leaving a huge gap between lower and upper bounds.

We prove that there is a polytope contained in the 0/1 cube that has Chvátal rank $\Omega(n^2)$, closing the gap up to a logarithmic factor. In fact, even a superlinear lower bound was mentioned as an open problem by several authors. Our choice of P is the convex hull of a semi-random Knapsack polytope and a single fractional vertex. The main technical ingredient is linking the Chvátal rank to *simultaneous Diophantine approximations* w.r.t. the $\|\cdot\|_1$-norm of the normal vector defining P.

1 Introduction

Gomory-Chvátal cuts are among the most important classes of cutting planes used to derive the integral hull of polyhedra. The fundamental idea to derive such cuts is that if an inequality $cx \leq \beta$ is *valid* for a polytope P (that is, $cx \leq \beta$ holds for every $x \in P$) and $c \in \mathbb{Z}^n$, then $cx \leq \lfloor \beta \rfloor$ is valid for the integral hull $P_I := \text{conv}(P \cap \mathbb{Z}^n)$. Formally, for a polytope $P \subseteq \mathbb{R}^n$ and a vector $c \in \mathbb{Z}^n$,

$$GC_P(c) := \left\{ x \in \mathbb{R}^n \mid cx \leq \lfloor \max\{cy \mid y \in P\} \rfloor \right\}$$

is the *Gomory-Chvátal Cut* that is induced by vector c (for polytope P). Furthermore,

$$P' := \bigcap_{c \in \mathbb{Z}^n} GC_P(c)$$

is the *Gomory-Chvátal closure* of P. Let $P^{(i)} := (P^{(i-1)})'$ (and $P^{(0)} = P$) be the *ith Gomory-Chvátal closure* of P. The *Chvátal rank* $\text{rk}(P)$ is the smallest number such that $P^{(\text{rk}(P))} = P_I$.

⋆ Supported by the Alexander von Humboldt Foundation within the Feodor Lynen program, by ONR grant N00014-11-1-0053 and by NSF contract CCF-0829878.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 349–361, 2013.

It is well-known that the Chvátal rank is always finite, but can be arbitrarily large already for 2 dimensional polytopes. However, if we restrict our attention to polytopes P contained in the 0/1 cube the situation becomes much different, and the Chvátal rank can be bounded by a function in n. In particular, Bockmayr, Eisenbrand, Hartmann and Schulz [BEHS99] provided the first polynomial upper bound of $\mathrm{rk}(P) \leq O(n^3 \log n)$. Later, Eisenbrand and Schulz [ES99, ES03] proved that $\mathrm{rk}(P) \leq O(n^2 \log n)$, which is still the best known upper bound. Note that if $P \subseteq [0,1]^n$ and $P \cap \{0,1\}^n = \emptyset$, then even $\mathrm{rk}(P) \leq n$ (and this is tight if and only if P intersects all the edges of the cube [PS11a]). Already [CCH89] provided lower bounds on the rank for the polytopes corresponding to natural problems like stable-set, set-covering, set-partitioning, knapsack, maxcut and ATSP (however, none of the bounds exceeded n). The paper of Eisenbrand and Schulz [ES99, ES03] also provides a lower bound $\mathrm{rk}(P) > (1+\varepsilon)n$ for a tiny constant $\varepsilon > 0$, which has been quite recently improved by Pokutta and Stauffer [PS11b] to $(1 + \frac{1}{e} - o(1))n$. However, as the authors of [PS11a] state, there is still a very large gap between the best known upper and lower bound. In particular, the question whether there is any *superlinear* lower bound on the rank of a polytope in the 0/1 cube is open since many years (see e.g. Ziegler [Zie00]).

In this paper, we prove that there is a polytope contained in the 0/1 cube that has Chvátal rank $\Omega(n^2)$, closing the gap up to a logarithmic factor. Specifically, our main result is:

Theorem 1. *For every n, there exists a vector $c \in \{0, \ldots, 2^{n/16}\}^n$ such that the polytope*

$$P = conv\Big\{\Big\{x \in \{0,1\}^n : \sum_{i=1}^n c_i x_i \leq \frac{\|c\|_1}{2}\Big\} \cup \Big\{\Big(\frac{3}{4}, \ldots, \frac{3}{4}\Big)\Big\}\Big\} \subseteq [0,1]^n$$

has Chvátal rank $\Omega(n^2)$.

Here $\|c\|_1 := \sum_{i=1}^n |c_i|$ and $\|c\|_\infty := \max_{i=1,\ldots,n} |c_i|$.

1.1 Related Work

There is a large amount of results on structural properties of the CG closure. Already Schrijver [Sch80] could prove that the closure of a rational polyhedron is again described by finitely many inequalities. Dadush, Dey and Vielma [DDV11a] showed that K' is a polytope for all compact and strictly convex sets $K \subseteq \mathbb{R}^n$. Later, Dunkel and Schulz [DS10] could prove the same if K is an irrational polytope, while in parallel again Dadush, Dey and Vielma [DDV11b] showed that this holds in fact for *any* compact convex set.

In the last years, automatic procedures that strengthen existing relaxations became more and more popular in theoretical computer science. Singh and Talwar [ST10] showed that few CG rounds reduce the integrality gap for k-uniform hypergraph matchings. However, to obtain approximation algorithms researchers rely more on *Lift-and-Project Methods* such as the hierarchies of *Balas, Ceria, Cornuéjols* [BCC93]; *Lovász, Schrijver* [LS91]; *Sherali, Adams* [SA90] or *Lasserre* [Las01a, Las01b]. One can optimize over the tth level in time $n^{O(t)}$.

Moreover, all those hierarchies converge to the integral hull already after n iterations. In contrast, the membership problem for P' is **coNP**-hard [Eis99]. We refer to the surveys of Laurent [Lau03] and Chlamtáč and Tulsiani [CT11] for a detailed comparison.

2 Outline

In the following, we provide an informal outline of our approach.

(1) *The polytope.* Our main result is to show that the polytope

$$P(c,\varepsilon) := \operatorname{conv}\Big\{\Big\{x \in \{0,1\}^n : cx \le \frac{\|c\|_1}{2}\Big\} \cup \{x^*(\varepsilon)\}\Big\}$$

has a Chvátal rank of $\Omega(n^2)$, where $x^* := x^*(\varepsilon) := (\frac{1}{2}+\varepsilon, \ldots, \frac{1}{2}+\varepsilon)$ (see Figure 1.(a)). We can choose $\varepsilon := \frac{1}{4}$ and each c_i will be an integral coefficient of order $2^{\Theta(n)}$ — however, we postpone the precise choice of c for now. Intuitively spoken, P is a Knapsack polytope defined by inequality $cx \le \frac{\|c\|_1}{2}$ plus an extra fractional vertex x^*. Observe that the vector $x^*(0) = (\frac{1}{2}, \ldots, \frac{1}{2})$ satisfies the Knapsack constraint with equality.

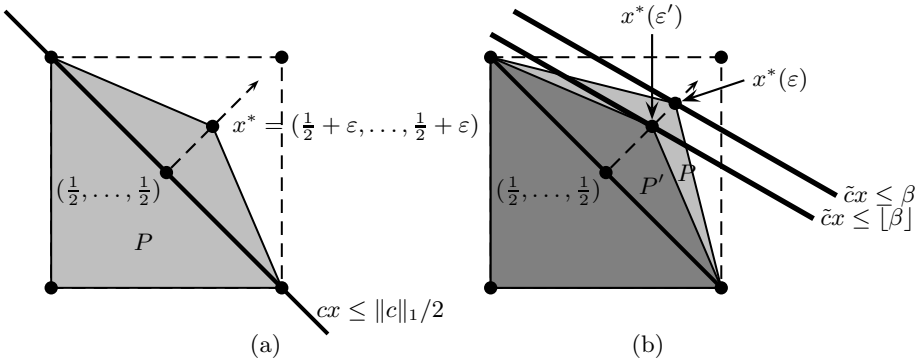


Fig. 1. (a) Polytope $P = P(c,\varepsilon)$ in $n = 2$ dimensions and with $c = (1,1)$. (b) Visualization of the Gomory Chvátal cut $\tilde{c}x \le \beta$ for a critical vector $\tilde{c}$. Note that $\max\{\tilde{c}x \mid x \in P\} = \tilde{c}x^*(\varepsilon)$.

(2) *The progress of the GC operator.* We will measure the progress of the Gomory Chvátal operator by observing how much of the line segment between $\frac{1}{2}\mathbf{1}$ and $\frac{3}{4}\mathbf{1}$ has been cut off. Consider a single Gomory Chvátal round and that Chvátal cut $\tilde{c}x \le \lfloor\beta\rfloor$ that cuts off the longest piece from the line segment. In other words, $\tilde{c}x \le \beta$ is valid for P, but $\tilde{c}x^* > \lfloor\beta\rfloor$. Of course, a necessary condition on such a vector $\tilde{c}$ is that the objective function $\tilde{c}$ is maximized at x^*, and therefore

$$\max\{\tilde{c}x \mid x \in P_I\} \le \tilde{c}x^*.$$

Let us call *critical* any integer vector $\tilde{c}$ satisfying the latter inequality (see Figure 1.(b)). Observe that the point $x^*(\varepsilon') \in P'$ with maximum ε' must have $\tilde{c}x^*(\varepsilon') = \lfloor \beta \rfloor$. But that means

$$1 \geq \tilde{c}x^*(\varepsilon) - \tilde{c}x^*(\varepsilon') = \tilde{c}\mathbf{1} \cdot (\varepsilon - \varepsilon') = \|\tilde{c}\|_1 \cdot (\varepsilon - \varepsilon')$$

and we can bound the progress of the Gomory Chvátal operator by $\varepsilon - \varepsilon' \leq \frac{1}{\|\tilde{c}\|_1}$. In other words, in order to show a high rank, we need to prove that all critical vectors must be long.

We will later propose a choice of c such that any critical vector $\tilde{c}$ has $\|\tilde{c}\|_1 \geq \Omega(\frac{n}{\varepsilon})$ (as long as $\varepsilon \geq (\frac{1}{2})^{O(n)}$). This means that the number of GC iterations until the current value of ε reduces to $\varepsilon/2$ will be $\Omega(n)$; thus it will take $\Omega(n^2)$ iterations until $\varepsilon = (1/2)^{\Theta(n)}$ is reached.

(3) *Critical vectors must be long.* Why should we expect that critical vectors must be long? Intuitively, if ε is getting smaller, then x^* is moving closer to the hyperplane defined by c and the cone of objective functions that are optimal at x^* becomes very narrow. As a consequence, the length of critical vectors should increase as ε decreases.

Recall that we termed $\tilde{c} \in \mathbb{Z}^n$ critical if and only if $\max\{\tilde{c}x \mid x \in P_I\} \leq \tilde{c}x^*$. One of our key lemmas is to show that under some mild conditions, the left hand side of the above inequality can be lowerbounded by $\frac{1}{2}\|\tilde{c}\|_1 + \Theta(\|\tilde{c} - \frac{c}{\lambda}\|_1)$, where $\lambda > 0$ is some scalar. As we will see, an immediate consequence is that for a critical vector $\tilde{c}$ it is a necessary condition that there is a $\lambda > 0$ with

$$\|\lambda\tilde{c} - c\|_1 \leq O(\varepsilon\|c\|_1).$$

In other words, it is necessary that $\tilde{c}$, if suitably scaled, well approximates the vector c. In fact, this problem is well studied under the name *simultaneous Diophantine approximation*. Thus, if we want to show that critical vectors must be long, it suffices to find a vector c that does not admit good approximations using short vectors $\tilde{c}$. The simple solution is to pick c *at random* from a suitable range; then $\|\lambda\tilde{c} - c\|_1$ will be large with high probability for all λ and all short $\tilde{c}$.

3 A General Strategy to Lower Bounding the Chvátal Rank

We focus now on the polytope $P := P(c, \varepsilon)$ defined above and properties of critical vectors. We want to define $L_c(\varepsilon)$ as the $\|\cdot\|_1$-length of the shortest vector, that is $x^*(\varepsilon)$-critical. Formally, let

$$L_c(\varepsilon) := \min_{\tilde{c} \in \mathbb{Z}^n_{\geq 0}} \left\{ \|\tilde{c}\|_1 \mid \tilde{c}x^*(\varepsilon) \geq \max_{x \in P_I} \tilde{c}x \right\} = \min_{\tilde{c} \in \mathbb{Z}^n_{\geq 0}} \left\{ \|\tilde{c}\|_1 \mid \|\tilde{c}\|_1 \cdot \left(\frac{1}{2} + \varepsilon\right) \geq \max_{x \in P_I} \tilde{c}x \right\}$$

By definition, the function L is monotonically non-increasing in ε and $L_c(0) \leq \|c\|_1$.

For example, if $c = (1, \ldots, 1)$, it is not difficult to show that $L_c(\varepsilon) \geq \frac{n}{2}$ for all $0 < \varepsilon < \frac{1}{2}$. In fact, for all c and ε, one can show a general *upper* bound of $L_c(\varepsilon) \leq \frac{n}{\varepsilon}$ (we omit the proofs of these statements from this extended abstract). Later we will see that for some choice of c this bound is essentially tight — for a long range of ε, and this will be crucial to prove our result.

Observe that, in the definition of $L_c(\varepsilon)$, we only admit non-negative entries for $\tilde{c}$. But it is not difficult to prove that the shortest critical vectors will be non-negative.

Lemma 1. *Let $\tilde{c} \in \mathbb{Z}^n$ be $x^*(\varepsilon)$-critical. Then there is a vector $\tilde{c}^+ \in \mathbb{Z}^n_{\geq 0}$ that is also $x^*(\varepsilon)$-critical and has $\sum_{i=1}^n \tilde{c}_i^+ = \sum_{i=1}^n \tilde{c}_i$ and $\|\tilde{c}^+\|_1 \leq \|\tilde{c}\|_1$.*

Proof. First note that any x^*-critical vector must have $\mathbf{1}^T\tilde{c} \geq 0$ since $\mathbf{0} \in P_I$. Now suppose that $\tilde{c}$ has a negative entry, say w.l.o.g. $\tilde{c}_1 < 0$. Then there must also be a positive entry, say $\tilde{c}_2 > 0$. We define a vector $\tilde{c}^+ \in \mathbb{Z}^n$ that has $\sum_{i=1}^n \tilde{c}_i^+ = \sum_{i=1}^n \tilde{c}_i$ and $\|\tilde{c}^+\|_1 \leq \|\tilde{c}\|_1 - 1$. Then iterating the procedure results in a vector satisfying the claim. We define

$$\tilde{c}_1^+ := \tilde{c}_1 + 1, \qquad \tilde{c}_2^+ := \tilde{c}_2 - 1, \qquad \tilde{c}_i^+ := \tilde{c}_i \; \forall i \geq 3.$$

Note that indeed $\sum_{i=1}^n \tilde{c}_i^+ = \sum_{i=1}^n \tilde{c}_i$ and $\tilde{c}x^*(\varepsilon) = \tilde{c}^+x^*(\varepsilon) =: \beta$. Suppose for the sake of contradiction that $\tilde{c}^+$ is not x^*-critical anymore. In other words, there must be a $y \in P_I \cap \{0,1\}^n$ such that $\tilde{c}^+y > \beta \geq \tilde{c}y$ and consequently $1 \leq \tilde{c}^+y - \tilde{c}y = y_1 - y_2$. It follows that $y_1 = 1$ and $y_2 = 0$. In fact, we also know that $\tilde{c}y > \beta - 1$. Moreover P_I is monotone, thus $y - e_1 \in P_I$ and the objective function is $\tilde{c}(y - e_1) > (\beta - 1) - \tilde{c}_1 \geq \beta = \tilde{c}x^*(\varepsilon)$ as $\tilde{c}_1 < 0$. In other words, already $\tilde{c}$ is not x^*-critical, which contradicts the assumption. □

How does the length of critical vectors relate to the Chvátal rank? The next lemma answers this question. In fact, one iteration of the Gomory Chvátal closure, reduces ε by essentially $\frac{1}{L_c(\varepsilon)}$.

Lemma 2. *Suppose $L_c(\varepsilon) \geq \frac{\gamma}{\varepsilon}$ for all $\delta_1 \leq \varepsilon \leq \delta_0$ (with $\gamma \geq 2$). Then $rk(P(c, \delta_0)) \geq \frac{\gamma}{2} \cdot \ln(\frac{\delta_0}{\delta_1})$.*

Proof. Abbreviate $P := P(\delta_0, c)$. To measure the progress of the Chvátal operator, consider $\varepsilon_i := \max\{\varepsilon : x^*(\varepsilon) \in P^{(i)}\}$. Let k be the index such that $\delta_0 = \varepsilon_0 \geq \varepsilon_1 \geq \ldots \geq \varepsilon_{k-1} \geq \delta_1 > \varepsilon_k$. Clearly $\mathrm{rk}(P) \geq k$.

Consider a fixed $i \in \{0, \ldots, k-1\}$. We want to argue that the difference between consecutive ε_i's is very small, i.e. $\frac{\varepsilon_{i+1}}{\varepsilon_i} \geq 1 - \frac{1}{\gamma}$. So assume that $\varepsilon_i > \varepsilon_{i+1}$, otherwise there is nothing to show. Let $\tilde{c}_i x \leq \lfloor \beta_i \rfloor$ be the Gomory Chvátal cutting plane that cuts furthest w.r.t. the line segment defined by $x^*(\varepsilon)$. In other words $\tilde{c}_i x \leq \beta_i$ is feasible for $P^{(i)}$ with $\tilde{c}_i \in \mathbb{Z}^n$ and $\tilde{c}_i x^*(\varepsilon_{i+1}) = \lfloor \beta_i \rfloor$ (similar to Figure 1.(b)). Since $\varepsilon_i > \varepsilon_{i+1}$, we have $\tilde{c}_i x^*(\varepsilon_i) > \lfloor \beta_i \rfloor$. Combining this with the fact that $P(c, \varepsilon_i) \subseteq P^{(i)}$, we know that $\tilde{c}_i$ is critical w.r.t. $x^*(\varepsilon_i)$. It is not necessarily true that $\tilde{c}_i$ has non-negative entries, but by Lemma 1 we know that there is a non-negative vector $\tilde{c}_i^+$ that is also $x^*(\varepsilon_i)$-critical and

satisfies $\mathbf{1}^T\tilde{c}_i^+ = \mathbf{1}^T\tilde{c}_i$ and $\|\tilde{c}_i^+\|_1 \le \|\tilde{c}_i\|_1$. By assumption $\|\tilde{c}_i^+\|_1 \ge L_c(\varepsilon_i) \ge \frac{\gamma}{\varepsilon_i}$. We obtain

$$1 \ge \underbrace{\tilde{c}_i x^*(\varepsilon_i)}_{\le \beta_i} - \underbrace{\tilde{c}_i x^*(\varepsilon_{i+1})}_{=\lfloor \beta_i \rfloor} = \tilde{c}_i \cdot \mathbf{1} \cdot (\varepsilon_i - \varepsilon_{i+1}) = \tilde{c}_i^+ \cdot \mathbf{1} \cdot (\varepsilon_i - \varepsilon_{i+1})$$
$$= \|\tilde{c}_i^+\|_1 \cdot (\varepsilon_i - \varepsilon_{i+1}) \ge \frac{\gamma}{\varepsilon_i} \cdot (\varepsilon_i - \varepsilon_{i+1})$$

which can be rearranged to $\frac{\varepsilon_{i+1}}{\varepsilon_i} \ge 1 - \frac{1}{\gamma}$ as claimed. Finally,

$$\delta_1 > \varepsilon_k = \delta_0 \cdot \prod_{i=0}^{k-1} \frac{\varepsilon_{i+1}}{\varepsilon_i} \ge \delta_0 \cdot \left(1 - \frac{1}{\gamma}\right)^k \ge \delta_0 \cdot e^{-2k/\gamma}$$

using that $1 - x \ge e^{-2x}$ for $0 \le x \le \frac{1}{2}$. Rearranging yields $k \ge \frac{\gamma}{2} \ln(\frac{\delta_0}{\delta_1})$. □

4 Constructing a Good Knapsack Solution

In order to provide a lower bound on $L_c(\varepsilon)$, we inspect the knapsack problem $\max\{\tilde{c}x \mid x \in P_I\}$ for a critical vector $\tilde{c}$. The crucial ingredient for our proof is to find a fairly tight lower bound on this quantity.

In the following key lemma (Lemma 3), we are going to show that (under some conditions on c) we can derive the lower bound: $\max\{\tilde{c}x \mid x \in P_I\} \ge \frac{1}{2}\|\tilde{c}\|_1 + \Omega(\left\|\tilde{c} - \frac{c}{\lambda}\right\|_1)$ for some $\lambda > 0$. Intuitively the vector $x = (\frac{1}{2}, \ldots, \frac{1}{2})$ is already a (fractional) solution to the above knapsack problem of value $\|\tilde{c}\|_1/2$, but if c and $\tilde{c}$ have a large angle, then one actually improve over that solution; in fact one can improve by the "difference" $\|\tilde{c} - \frac{c}{\lambda}\|_1$. Before the formal proof, let us describe, how to derive this lower bound in an ideal world that is free of technicalities.

Sort the items by their profit over cost ratio so that $\frac{\tilde{c}_1}{c_1} \ge \ldots \ge \frac{\tilde{c}_n}{c_n}$. Since we are dealing with a knapsack problem, we start taking the items with the best ratio into our solution. Suppose for the sake of simplicity that we are lucky and the k items with largest ratio fit perfectly into the knapsack, i.e. $\sum_{i=1}^k c_i = \|c\|_1/2$. Then $J := [k]$ must actually be an *optimum* knapsack solution. Next, choose $\lambda > 0$ such that $\frac{1}{\lambda}$ is the profit threshold, i.e. $\frac{\tilde{c}_1}{c_1} \ge \ldots \ge \frac{\tilde{c}_k}{c_k} \ge \frac{1}{\lambda} \ge \frac{\tilde{c}_{k+1}}{c_{k+1}} \ge \ldots \ge \frac{\tilde{c}_n}{c_n}$. Using that $\sum_{i \in J} c_i = \sum_{i \notin J} c_i$, we can express the profit of our solution as

$$\sum_{i \in J} \tilde{c}_i = \frac{1}{2}\|\tilde{c}\|_1 + \frac{1}{2}\sum_{i \in J} \underbrace{\left(\tilde{c}_i - \frac{c_i}{\lambda}\right)}_{\ge 0} - \frac{1}{2}\sum_{i \notin J} \underbrace{\left(\tilde{c}_i - \frac{c_i}{\lambda}\right)}_{\le 0} = \frac{1}{2}\|\tilde{c}\|_1 + \frac{1}{2}\left\|\tilde{c} - \frac{c}{\lambda}\right\|_1$$

proving the claimed lower bound on $\max\{\tilde{c}x \mid x \in P_I\}$. In a non-ideal world, the greedily obtained solution would not perfectly fill the knapsack, i.e. $\sum_{i=1}^k c_i < \|c\|_1/2$. To fill this gap, we rely on the concept of *additive basis*.

Definition 1. *Let $I = [a, b] \cap \mathbb{Z}_{\ge 0}$ be an interval of integers. We call a subset $B \subseteq \mathbb{Z}_{\ge 0}$ an additive basis for I if for every $k \in I$, there are numbers $S \subseteq B$ such that $\sum_{s \in S} s = k$.*

In other words, we can express every number in I as a sum of numbers in B. For example $\{2^0, 2^1, 2^2, \ldots, 2^k\}$ is an additive basis for $\{0, \ldots, 2^0+2^1+\ldots+2^k\}$. The geometric consequence for a knapsack polytope $Q = \{x \in \mathbb{R}^n_{\geq 0} \mid cx \leq \|c\|_1/2\}$ is the following: if $c_1, \ldots, c_n$ are integral numbers that contain an additive basis (with at most $n/2$ elements) for $\{0, \ldots, \|c\|_\infty\}$ and, let's say $\|c\|_\infty \leq O(\frac{\|c\|_1}{n})$, then the face $cx = \|c\|_1/2$ contains $2^{\Omega(n)}$ many 0/1 points. The reason for this fact is that we can extend any subset of items $I \subseteq [n]$ that does not exceed the capacity and that does not contain any basis element, to a solution that fully fills the knapsack (by adding a couple of basis elements). In the following, we abbreviate as usual $c(J) := \sum_{i \in J} c_i$.

Lemma 3. *Let $c \in \mathbb{Z}^n_{>0}$, $\tilde{c} \in \mathbb{R}^n_{>0}$ and 3 disjoint index sets $B_1, B_2, B_3 \subseteq [n]$ such that each set $\{c_i \mid i \in B_\ell\}$ is an additive basis for the interval $I = \{0, \ldots, \|c\|_\infty\}$ with $\|c\|_\infty \leq \delta \|c\|_1$ and $c(B_\ell) \leq \delta \cdot \|c\|_1$ for all $\ell = 1, 2, 3$ with $\delta := \frac{1}{100}$. Then there is a scalar $\lambda := \lambda(c, \tilde{c}) > 0$ such that*

$$\max\left\{\tilde{c}x \mid x \in \{0,1\}^n;\ cx \leq \frac{\|c\|_1}{2}\right\} \geq \frac{1}{2}\|\tilde{c}\|_1 + \frac{1}{16} \cdot \left\|\tilde{c} - \frac{c}{\lambda}\right\|_1$$

Proof. Since we allow $\tilde{c}_i \in \mathbb{R}$, there lies no harm in perturbing the coefficients slightly such that the profit/cost ratios $\frac{\tilde{c}_i}{c_i}$ are pairwise distinct. We sort the indices such that $\frac{\tilde{c}_1}{c_1} > \ldots > \frac{\tilde{c}_n}{c_n}$. Choose $\lambda > 0$ such that

$$\sum_{i : \tilde{c}_i/c_i > 1/\lambda} c_i \in \left[\frac{\|c\|_1}{2} - \|c\|_\infty, \frac{\|c\|_1}{2}\right]$$

and there is no i with $\frac{\tilde{c}_i}{c_i} = \frac{1}{\lambda}$ (recall that $\frac{\tilde{c}_i}{c_i} > \frac{1}{\lambda} \Leftrightarrow \lambda\tilde{c}_i - c_i > 0$). In other words, $\frac{1}{\lambda}$ is a *profit threshold* and ideally we would like to construct a solution for our knapsack problem by selecting the items above the threshold. Let $q \in \{1, \ldots, n\}$ be the number such that $\frac{\tilde{c}_i}{c_i} > \frac{1}{\lambda} \Leftrightarrow i \leq q$, i.e. $\frac{\tilde{c}_1}{c_1} > \ldots > \frac{\tilde{c}_q}{c_q} > \frac{1}{\lambda} > \frac{\tilde{c}_{q+1}}{c_{q+1}} > \ldots > \frac{\tilde{c}_n}{c_n}$. For every item i we define the *relative profit* $w_i := \tilde{c}_i - \frac{c_i}{\lambda}$. Note that $\frac{w_i}{c_i} = \frac{\tilde{c}_i}{c_i} - \frac{1}{\lambda}$ and $w_i > 0 \Leftrightarrow i \leq q$, but the values w_i are not necessarily monotonically decreasing. The way how we defined w yields $\|w\|_1 = \|\tilde{c} - \frac{c}{\lambda}\|_1$. Since the B_ℓ's are disjoint, one has $\sum_{\ell=1}^3 \sum_{i \in B_\ell} |w_i| \leq \|w\|_1$. Thus we can pick one of the sets $B := B_\ell$ such that $\sum_{i \in B} |w_i| \leq \frac{1}{3}\|w\|_1$.

We are now going to construct a knapsack solution that fully fills the knapsack. Let $k \in [n]$ maximal be such that

$$\sum_{i \in \{1,\ldots,k\} \setminus B} c_i \leq \frac{\|c\|_1}{2}$$

In other words, if we take items $\{1, \ldots, k\} \setminus B$ into our knapsack, we have capacity at most $\|c\|_\infty$ left. Next, construct an arbitrary solution $J' \subseteq B$ that perfectly fills the remaining capacity, i.e. for $J := (\{1, \ldots, k\} \setminus B) \cup J'$ we have $c(J) = \frac{\|c\|_1}{2}$.

Observe that

$$c([k]) \leq \underbrace{c([k]\backslash B)}_{\leq \|c\|_1/2} + \underbrace{c(B)}_{\leq \delta\|c\|_1} \leq \left(\frac{1}{2} + \delta\right) \cdot \|c\|_1. \tag{1}$$

Moreover,

$$c([k]) \geq c([k]\backslash B) \geq \frac{\|c\|_1}{2} - \|c\|_\infty \geq \left(\frac{1}{2} - \delta\right) \cdot \|c\|_1 \tag{2}$$

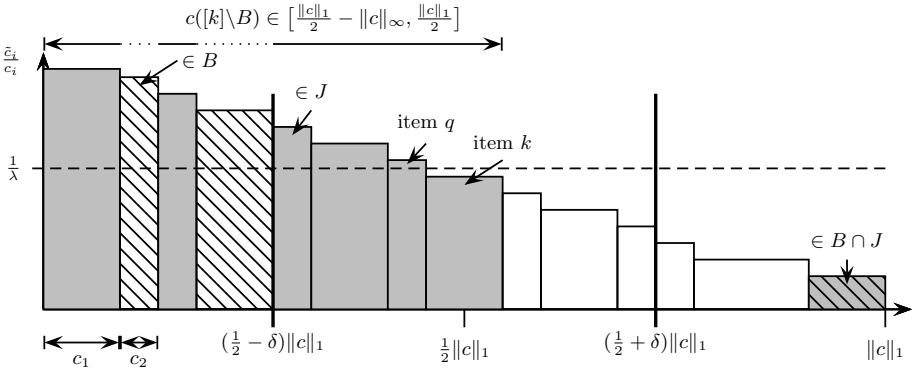


Fig. 2. Visualization of the construction of J: Take items with best profit/cost ratio (skipping items in the basis B) as long as possible. Then fill the remaining gap with arbitrary items from B.

We call an item i *central* if $(\frac{1}{2} - \delta)\|c\|_1 \leq c([i]) \leq (\frac{1}{2} + \delta)$. We cannot be sure appriori whether central items are selected into J or not. However, we can prove that due to the sorting they have a small $|w_i|$-value anyway. Let us abbreviate $W^+ := \sum_{i \leq q} w_i$ and $W^- := \sum_{i > q} |w_i|$ (so that $\|w\|_1 = W^+ + W^-$).

Claim. $\sum_{i:\ (\frac{1}{2}-\delta)\|c\|_1 \leq c([i]) \leq (\frac{1}{2}+\delta)\|c\|_1} |w_i| \leq 9\delta\|w\|_1$.

Proof of claim. We abbreviate $I_+ := \{i \mid w_i > 0\}$ and $I_- := \{i \mid w_i < 0\}$. Furthermore $I_+^\delta := \{i \in I_+ \mid c([i]) \geq (\frac{1}{2} - \delta)\|c\|_1\}$ and $I_-^\delta := \{i \in I_- \mid c([i]) \leq (\frac{1}{2} + \delta)\|c\|_1\}$. Note that $c(I_+), c(I_-) \geq \frac{1}{2}\|c\|_1 - 2\|c\|_\infty \geq \frac{1}{3}\|c\|_1$ (since $\|c\|_1 \geq 12\|c\|_\infty$) and $c(I_+^\delta), c(I_-^\delta) \leq \delta\|c\|_1 + 2\|c\|_\infty \leq 3\delta\|c\|_1$ (since $\|c\|_\infty \leq \delta\|c\|_1$).

Recall that the items are sorted such that the values $\frac{w_i}{c_i} = \frac{\tilde{c}_i}{c_i} - \frac{1}{\lambda}$ decrease and I_+^δ is a set of maximal indices within I_+, thus the average of $\frac{w_i}{c_i}$ over items in I_+^δ cannot be higher than the average over I_+. Formally $\frac{w(I_+^\delta)}{c(I_+^\delta)} \leq \frac{W^+}{c(I_+)}$, thus

$$w(I_+^\delta) \leq \frac{c(I_+^\delta)}{c(I_+)} \cdot W^+ \leq \frac{3\delta\|c\|_1}{\|c\|_1/3} W^+ = 9\delta \cdot W^+. \tag{3}$$

Analogously $\frac{\sum_{i\in I_-^\delta}|w_i|}{c(I_-^\delta)} \le \frac{W^-}{c(I_-)}$, and hence

$$\sum_{i\in I_-^\delta}|w_i| \le \frac{c(I_-^\delta)}{c(I_-)}\cdot W^- \le \frac{3\delta\|c\|_1}{\|c\|_1/3}\cdot W^- \le 9\delta\cdot W^-. \tag{4}$$

Adding up (3) and (4) yields the claim

$$\sum_{i\in I_+^\delta\cup I_-^\delta}|w_i| \le 9\delta\cdot(W^+ + W^-) = 9\delta\|w\|_1.$$

◊

Claim. $w(J) - w([n]\backslash J) \ge \frac{1}{8}\cdot\|w\|_1$.

Proof of claim. We call an index $i\in[n]$ *correct*, if $i\in J \Leftrightarrow w_i > 0$. In other words, indices with $w_i > 0$ that are in J are correct and indices with $w_i < 0$ and $i\notin J$ are correct – all other indices are *incorrect*. A correct index i contributes $+|w_i|$ to the sum $w(J)-w([n]\backslash J)$ and an incorrect index contributes $-|w_i|$. Thus if all indices would be correct, we would have $w(J) - w([n]\backslash J) = \|w\|_1$. From this amount, we want to deduct contributions for incorrect indices. An index can either be incorrect if it is in B (for those we have $\sum_{i\in B}|w_i| \le \frac{1}{3}\|w\|_1$) or if it lies in the central window, i.e. $c([i])\in(\frac{1}{2}\pm\delta)\|c\|_1$ (for those items we have $\sum_{i:\,(\frac{1}{2}-\delta)\|c\|_1\le c([i])\le(\frac{1}{2}+\delta)\|c\|_1}|w_i| \le 9\delta\|w\|_1$ according to Claim 4). Subtracting these quantities, for $\delta = \frac{1}{100}$ we obtain

$$\sum_{i\in J}w_i - \sum_{i\notin J}w_i \ge \left(1 - 2\cdot 9\delta - 2\cdot\frac{1}{3}\right)\cdot\|w\|_1 \ge \frac{1}{8}\|w\|_1.$$

◊

Finally, we note that the vector $\tilde{x}\in\{0,1\}^n$ with $\tilde{x}_i := 1$ if $i\in J$ and 0 otherwise, satisfies the claim.

$$\begin{aligned}
\tilde{c}\tilde{x} &= \frac{1}{2}\|\tilde{c}\|_1 + \frac{1}{2}\sum_{i\in J}\tilde{c}_i - \frac{1}{2}\sum_{i\notin J}\tilde{c}_i \\
&\overset{\sum_{i\in J}c_i=\sum_{i\notin J}c_i}{=} \frac{1}{2}\|\tilde{c}\|_1 + \frac{1}{2}\sum_{i\in J}\left(\tilde{c}_i - \frac{c_i}{\lambda}\right) - \frac{1}{2}\sum_{i\notin J}\left(\tilde{c}_i - \frac{c_i}{\lambda}\right) \\
&= \frac{1}{2}\|\tilde{c}\|_1 + \frac{1}{2}\sum_{i\in J}w_i - \frac{1}{2}\sum_{i\notin J}w_i \\
&\overset{\text{Claim (4)}}{\ge} \frac{1}{2}\|\tilde{c}\|_1 + \frac{1}{16}\|w\|_1 = \frac{1}{2}\|\tilde{c}\|_1 + \frac{1}{16}\left\|\tilde{c} - \frac{c}{\lambda}\right\|_1
\end{aligned}$$

Here we use that $\sum_{i\in J}c_i = \sum_{i\notin J}c_i$. □

Now, we can get a very handy necessary condition on critical vectors. Namely, if the conditions on c (see Lemma 3) are satisfied, then any critical vector must have $\|\lambda\tilde{c} - c\|_1 \le O(\varepsilon)\cdot\|c\|_1$. To prove that critical vectors must be long, it remains to find a vector c such that $\|\lambda\tilde{c} - c\|_1$ is large for all short vectors $\tilde{c}$.

5 Random Normal Vectors

In this section, we will see now, that a random vector a cannot be well approximated by short vectors; later this vector a will be essentially the first half of the normal vector c. In the following, for any vector $a \in \mathbb{R}^m$, and any index subset $I \subseteq [m]$, we let $(a)_I \in \mathbb{R}^{|I|}$ be the vector $(a_i, i \in I)$. For $D := 2^{m/8}$, pick $a_1, \dots, a_m \in \{D, \dots, 2D\}$ uniformly and independently at random.

We first informally describe, why this random vector a is hard to approximate with high probability. Let us fix values of λ and ε and call an index i *good*, if there is an integer $\tilde{a}_i \in \{0, \dots, o(\frac{1}{\varepsilon})\}$ such that $|\lambda \tilde{a}_i - a_i| \leq O(\varepsilon D)$. Since we choose a_i from D many possible choices, we have $\Pr[i \text{ good}] \leq o(\frac{1}{\varepsilon}) \cdot O(\varepsilon D) \cdot \frac{1}{D} = o(1)$. For the event "$\exists \tilde{a} : \|\tilde{a}\|_1 \leq o(\frac{m}{\varepsilon})$ and $\|\lambda \tilde{a} - a\|_1 \leq O(m\varepsilon D)$" one needs $\Omega(m)$ many good indices and by standard arguments the probability for this to happen is $o(1)^m$. Finally we can argue that the number of distinct values of ε and λ that need to be considered is $2^{O(m)}$. Thus by the union bound, the probability that there are *any* ε, λ and $\tilde{a} \in \{0, \dots, o(\frac{m}{\varepsilon})\}^m$ with $\|\lambda \tilde{a}_i - a\|_1 \leq O(\varepsilon m D) = O(\varepsilon \|a\|_1)$ is still upper bounded by $o(1)^m$. We will now give a formal argument.

Lemma 4. *There is a (large enough) constant $\alpha > 0$ such that for m large enough,*

$$\Pr\left[\exists (\varepsilon, \lambda, \tilde{a}) \in [\tfrac{1}{D}, \tfrac{1}{\alpha}] \times \mathbb{R}_{>0} \times \mathbb{Z}^m : \|\tilde{a}\|_1 \leq \frac{m}{\alpha\varepsilon} \text{ and } \|\lambda\tilde{a} - a\|_1 \leq 100\varepsilon m \cdot D\right] \leq \left(\frac{1}{2}\right)^m \tag{5}$$

Proof. We want to bound the above probability in (5) by using the union bound over all $\lambda > 0$ and all $\varepsilon > 0$. First of all, $\|\lambda\tilde{a} - a\|_1$ is a piecewise linear function in λ. Therefore, we can restrict our attention to the values $\lambda = \frac{a_i}{\tilde{a}_i}$ for some $a_i \in \{D, \dots, 2D\}$ and $\tilde{a}_i \in \{0, \dots, \frac{m}{\alpha\varepsilon}\}$. That is, assuming $\varepsilon \geq \frac{1}{D}$, the number of different λ values that really matter are bounded by $(D+1) \cdot (\frac{m}{\alpha\varepsilon} + 1) \leq (2D)^3$. Moreover, we only need to consider those ε, where at least one of the bounds $\|\tilde{a}\|_1 \leq \frac{m}{\alpha\varepsilon}$ or $\|\lambda\tilde{a} - a\|_1 \leq 100\varepsilon m \cdot D$ is tight[1]. But $\|\tilde{a}\|_1$ attains at most $2mD \leq (2D)^2$ many values and $\|\lambda\tilde{a} - a\|_1$ attains at most $(2D)^4$ many values. In total the number of relevant values of pairs (λ, ε) is bounded by $(2D)^9 \leq 2^{2m}$. Thus, by the union bound it suffices to prove that for every *fixed* pair $\lambda > 0$ and ε, one has

$$\Pr\left[\exists \tilde{a} \in \mathbb{Z}^m_{\geq 0} : \|\tilde{a}\|_1 \leq \frac{m}{\alpha\varepsilon} \text{ and } \|\lambda\tilde{a} - a\|_1 \leq 100\varepsilon m \cdot D\right] \leq 2^{-3m} \tag{6}$$

for $\alpha > 0$ large enough. Note that, for any vector $\tilde{a} \in \mathbb{Z}^m_{\geq 0} : \|\tilde{a}\|_1 \leq \frac{m}{\alpha\varepsilon}$ there exists a subset of indices $I \subset [m]$ with $|I| \geq \frac{m}{2}$ such that $\|(\tilde{a})_I\|_\infty \leq \frac{2}{\alpha\varepsilon}$. If not, then $\|\tilde{a}\|_1 > \frac{m}{2} \cdot \frac{2}{\alpha\varepsilon}$ leading to a contradiction. Similarly, we can say that there exists a subset of indices $J \subseteq I$, with $|J| \geq |I|/2$ such that $\|(\lambda\tilde{a} - a)_J\|_\infty \leq 400\varepsilon \cdot D$. If not, then $\|(\lambda\tilde{a} - a)_I\|_1 > \frac{m}{4} \cdot 400\varepsilon \cdot D$ again leading to a contradiction. It follows that the left hand side of (6) is bounded by

[1] The reason is that ε and ε' with $\lfloor m/(\alpha\varepsilon) \rfloor = \lfloor m/(\alpha\varepsilon') \rfloor$ and $\lfloor 100\varepsilon m D \rfloor = \lfloor 100\varepsilon' m D \rfloor$ belong to identical events.

$$\Pr\left[\exists \tilde{a} \in \mathbb{Z}^m_{\geq 0} \text{ and } J \subseteq [m], |J| = \frac{m}{4} : \|(\tilde{a})_J\|_\infty \leq \frac{2}{\alpha\varepsilon} \text{ and } \|(\lambda\tilde{a} - a)_J\|_\infty \leq 400\varepsilon \cdot D\right] \tag{7}$$

For a fixed index $i \in [m]$, we have

$$\begin{aligned}
&\Pr\left[\exists \tilde{a}_i \in \mathbb{Z}_{\geq 0} : \tilde{a}_i \leq \frac{2}{\alpha\varepsilon} \text{ and } |\lambda\tilde{a}_i - a_i| \leq 400\varepsilon \cdot D\right] \\
&\leq \frac{1}{D} \sum_{\tilde{a}_i = 0}^{2/(\alpha\varepsilon)} |\mathbb{Z} \cap [\lambda\tilde{a}_i - 400\varepsilon D, \lambda\tilde{a}_i + 400\varepsilon D]| \\
&\leq \left(\frac{2}{\alpha\varepsilon} + 1\right) \cdot \frac{800\varepsilon D + 1}{D} \leq \frac{3200}{\alpha}.
\end{aligned}$$

Here, we use that $\frac{1}{\alpha} \geq \varepsilon \geq \frac{1}{D}$, and every number $\lambda\tilde{a}_i$ is at distance $400\varepsilon D$ to at most $800\varepsilon D + 1$ many integers. Moreover, we upperbound the number of all different index subsets of cardinality $m/4$ by 2^m. It follows that (7) can be bounded by $2^m \cdot (\frac{3200}{\alpha})^{m/4} \leq (\frac{1}{2})^{3m}$ for $\alpha > 0$ large enough. □

6 A $\Omega(n^2)$ Bound on the Chvátal Rank

Now we have all tools together, to obtain a quadratic lower bound on the Chvátal rank of a 0/1 polytope.

Theorem 2. *Let n be any multiple of 16 and abbreviate $m := \frac{n}{2}$. Choose $c := (a, b, b, b, \mathbf{0}) \in \mathbb{Z}^n_{\geq 0}$, where a satisfies the event in (5) and $b = (2^0, 2^1, 2^2, \ldots, 2^{m/8+1})$. Then the Chvátal rank of $P := P(c, \frac{1}{4})$ is $\Omega(n^2)$.*

Proof. First, note that $m + 3 \cdot (\frac{m}{8} + 2) \leq n$, so that we can indeed fill the vector c with zero's to obtain n many entries. Moreover, observe that $b = (2^0, 2^1, 2^2, \ldots, 2^{m/8+1})$ is a basis for $\{0, \ldots, 2D\}$.

By Lemma 2, the statement follows if we show that for all $\frac{1}{D} \leq \varepsilon \leq \frac{1}{\alpha}$ one has $L_c(\varepsilon) \geq \Omega(\frac{n}{\varepsilon})$ (where $\alpha \geq 32$ is the constant from Lemma 4).

Hence, fix an ε and let $\tilde{c}$ be the $x^*(\varepsilon)$-critical vector with minimal $\|\tilde{c}\|_1$. Obviously, c contains 3 disjoint bases for the interval $\{0, \ldots, \|c\|_\infty\}$. Moreover:

$$\|c\|_\infty \leq \|b\|_1 \leq 4D \overset{n \text{ large enough}}{\leq} \frac{1}{100}\|c\|_1.$$

Therefore, we can apply Lemma 3 to obtain

$$\begin{aligned}
\left(\frac{1}{2} + \varepsilon\right)\|\tilde{c}\|_1 &= \tilde{c}x^*(\varepsilon) \overset{\tilde{c} \text{ critical}}{\geq} \max\left\{\tilde{c}x \mid x \in \{0,1\}^n;\ cx \leq \frac{\|c\|_1}{2}\right\} \\
&\overset{\text{Lem. } 3}{\geq} \frac{1}{2}\|\tilde{c}\|_1 + \frac{1}{16} \cdot \left\|\tilde{c} - \frac{c}{\lambda}\right\|_1
\end{aligned}$$

Subtracting $\frac{1}{2}\|\tilde{c}\|_1$ from both sides and multiplying with $\lambda > 0$ yields $\frac{1}{16}\|\lambda\tilde{c} - c\|_1 \leq \varepsilon\|\lambda\tilde{c}\|_1$. We claim that $\|\lambda\tilde{c}\|_1 \leq 2\|c\|_1$, since otherwise by the reverse

triangle inequality $\|\lambda\tilde{c} - c\|_1 \geq \|\lambda\tilde{c}\|_1 - \|c\|_1 > \frac{1}{2}\|\lambda\tilde{c}\|_1 \geq 16\varepsilon\|\lambda\tilde{c}\|_1$, which is a contradiction. Thus we have $\|\lambda\tilde{c} - c\|_1 \leq 32\varepsilon\|c\|_1$. Now, let $\tilde{a}$ be the first m entries of $\tilde{c}$, then $\|\lambda\tilde{a} - a\|_1 \leq \|\lambda\tilde{c} - c\|_1 \leq 32\varepsilon\|c\|_1 \leq 64\varepsilon nD$.

But inspecting again the properties of vector a (see Lemma 4), any such vector $\tilde{a}$ must have length $\|\tilde{a}\|_1 \geq \Omega(\frac{m}{\varepsilon})$. Since $m = n/2$, this implies $\|\tilde{c}\|_1 \geq \|\tilde{a}\|_1 \geq \Omega(\frac{n}{\varepsilon})$. Eventually, we apply Lemma 2 and obtain that $\mathrm{rk}(P) \geq \Omega(n \cdot \log(\frac{1/\alpha}{1/D})) = \Omega(n^2)$. □

We close the paper with a couple of remarks. A vector d is called *saturated* w.r.t. P if it has an integrality gap of 1, i.e. $\max\{dx \mid x \in P\} = \max\{dx \mid x \in P_I\}$. Of course, if $d \in \mathbb{Z}^n$ is saturated, then the GC cut induced by d does not cut off any point, i.e. $GC_P(d) \cap P = P$. With this definition, one could rephrase the statement of Theorem 2 as: The vector c needs $\Omega(n^2)$ many iterations to be saturated. Note that [ES03] prove that any vector $c \in \mathbb{Z}^n$ is saturated after $O(n^2 + n \log \|c\|_\infty)$ many iterations, which gives the tight bound of $O(n^2)$ for our choice of c.

Finally, we observe that our results imply that also the polytope

$$\tilde{P}(c, 1/4) := \mathrm{conv}\Big\{\Big\{x \in [0,1]^n : cx \leq \frac{\|c\|_1}{2}\Big\} \cup \{x^*(1/4)\}\Big\}$$

has a Chvátal rank of $\Omega(n^2)$. Note that $\tilde{P}$ is now a *fractional* Knapsack polytope plus an extra fractional vertex x^*. Interestingly, $\tilde{P}$ can be described using only a linear number of inequalities (we omit the details from this extended abstract).

References

[Bal85] Balas, E.: Disjunctive programming and a hierarchy of relaxations for discrete optimization problems 6, 466–486 (1985)

[BCC93] Balas, E., Ceria, S., Cornuéjols, G.: A lift-and-project cutting plane algorithm for mixed 0-1 programs. Math. Program. 58, 295–324 (1993)

[BEHS99] Bockmayr, A., Eisenbrand, F., Hartmann, M.E., Schulz, A.S.: On the Chvátal rank of polytopes in the 0/1 cube. Dis. App. Math. 98(1-2), 21–27 (1999)

[CCH89] Chvátal, V., Cook, W., Hartmann, M.: On cutting-plane proofs in combinatorial optimization. Linear Algebra and its Applications 114-115, 455–499 (1989); Special Issue Dedicated to A.J. Hoffman

[CT11] Chlamtáč, E., Tulsiani, M.: Convex relaxations and integrality gaps. In: Handbook on Semidefinite, Cone and Polynomial Optimization (2011)

[DDV11a] Dadush, D., Dey, S.S., Vielma, J.P.: The Chvátal-Gomory closure of a strictly convex body. Math. Oper. Res. 36(2), 227–239 (2011)

[DDV11b] Dadush, D., Dey, S.S., Vielma, J.P.: On the Chvátal-Gomory Closure of a Compact Convex Set. In: Günlük, O., Woeginger, G.J. (eds.) IPCO 2011. LNCS, vol. 6655, pp. 130–142. Springer, Heidelberg (2011)

[DS10] Dunkel, J., Schulz, A.S.: The Gomory-Chvátal closure of a non-rational polytope is a rational polytope (2010), `http://www.optimization-online.org/DB_HTML/2010/11/2803.html`

[Eis99] Eisenbrand, F.: On the membership problem for the elementary closure of a polyhedron. Combinatorica 19(2), 297–300 (1999)

[ES99] Eisenbrand, F., Schulz, A.S.: Bounds on the Chvátal Rank of Polytopes in the 0/1-Cube. In: Cornuéjols, G., Burkard, R.E., Woeginger, G.J. (eds.) IPCO 1999. LNCS, vol. 1610, pp. 137–150. Springer, Heidelberg (1999)

[ES03] Eisenbrand, F., Schulz, A.S.: Bounds on the Chvátal rank of polytopes in the 0/1-cube. Combinatorica 23(2), 245–261 (2003)

[Las01a] Lasserre, J.B.: An Explicit Exact SDP Relaxation for Nonlinear 0-1 Programs. In: Aardal, K., Gerards, B. (eds.) IPCO 2001. LNCS, vol. 2081, pp. 293–303. Springer, Heidelberg (2001)

[Las01b] Lasserre, J.: Global optimization with polynomials and the problem of moments. SIAM Journal on Optimization 11(3), 796–817 (2001)

[Lau03] Laurent, M.: A comparison of the Sherali-Adams, Lovász-Schrijver, and Lasserre relaxations for 0-1 programming. Math. Oper. Res. 28(3), 470–496 (2003)

[LS91] Lovász, L., Schrijver, A.: Cones of matrices and set-functions and 0-1 optimization. SIAM Journal on Optimization 1, 166–190 (1991)

[PS11a] Pokutta, S., Schulz, A.S.: Integer-empty polytopes in the 0/1-cube with maximal Gomory-Chvátal rank. Oper. Res. Lett. 39(6), 457–460 (2011)

[PS11b] Pokutta, S., Stauffer, G.: Lower bounds for the Chvátal-Gomory rank in the 0/1 cube. Oper. Res. Lett. 39(3), 200–203 (2011)

[SA90] Sherali, H., Adams, W.: A hierarchy of relaxation between the continuous and convex hull representations. SIAM J. Dis. Math. 3, 411–430 (1990)

[Sch80] Schrijver, A.: On cutting planes. Annals of Discrete Mathematics 9, Combinatorics 79, Part II, 291–296 (1980)

[ST10] Singh, M., Talwar, K.: Improving Integrality Gaps via Chvátal-Gomory Rounding. In: Serna, M., Shaltiel, R., Jansen, K., Rolim, J. (eds.) APPROX and RANDOM 2010. LNCS, vol. 6302, pp. 366–379. Springer, Heidelberg (2010)

[Zie00] Ziegler, G.M.: Lectures on 0/1-polytopes. In: Polytopes—combinatorics and computation (Oberwolfach, 1997). DMV Sem., vol. 29, pp. 1–41. Birkhäuser, Basel (2000)

Eight-Fifth Approximation for the Path TSP

András Sebő*

CNRS, UJF, Grenoble-INP, Laboratoire G-SCOP, France
Andras.Sebo@g-scop.inpg.fr
http://www.g-scop.grenoble-inp.fr/recherche/

Abstract. We prove the approximation ratio 8/5 for the metric $\{s,t\}$-path-TSP, and more generally for shortest connected T-joins.

The algorithm that achieves this ratio is the simple "Best of Many" version of Christofides' algorithm (1976), suggested by An, Kleinberg and Shmoys (2012), which consists in determining the best Christofides $\{s,t\}$-tour out of those constructed from a family $\mathcal{F}_+$ of trees having a convex combination dominated by an optimal solution x^* of the Held-Karp relaxation. They give the approximation guarantee $\frac{\sqrt{5}+1}{2}$ for such an $\{s,t\}$-tour, which is the first improvement after the 5/3 guarantee of Hoogeveen's Christofides type algorithm (1991). Cheriyan, Friggstad and Gao (2012) extended this result to a 13/8-approximation of shortest connected T-joins, for $|T| \geq 4$.

The ratio 8/5 is proved by simplifying and improving the approach of An, Kleinberg and Shmoys that consists in completing $x^*/2$ in order to dominate the cost of "parity correction" for spanning trees. We partition the edge-set of each spanning tree in $\mathcal{F}_+$ into an $\{s,t\}$-path (or more generally, into a T-join) and its complement, which induces a decomposition of x^*. This decomposition can be refined and then efficiently used to complete $x^*/2$ without using linear programming or particular properties of T, but by adding to each cut deficient for $x^*/2$ an individually tailored explicitly given vector, inherent in x^*.

A simple example shows that the Best of Many Christofides algorithm may not find a shorter $\{s,t\}$-tour than 3/2 times the incidentally common optima of the problem and of its fractional relaxation.

Keywords: traveling salesman problem, path TSP, approximation algorithm, matching, T-join, polyhedron, tree (basis) polytope.

1 Introduction

A Traveling Salesman wants to visit all vertices of a graph $G = (V, E)$, starting from his home $s \in V$, and – since it is Friday – ending his tour at his week-end residence, $t \in V$. Given the nonnegative valued length function $c : E \longrightarrow \mathbb{Q}_+$, he is looking for a shortest $\{s,t\}$-tour, that is, one of smallest possible (total) length.

* Supported by the TEOMATRO grant ANR-10-BLAN 0207 "New Trends in Matroids: Base Polytopes, Structure, Algorithms and Interactions".

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 362–374, 2013.

The Traveling Salesman Problem (TSP) is usually understood as the $s = t$ particular case of the defined problem, where in addition every vertex is visited exactly once. This "minimum length Hamiltonian cycle" problem is one of the main exhibited problems of combinatorial optimization. Besides being *NP*-hard even for very special graphs or lengths [11], even the best up to date methods of operations research, the most powerful computers coded by the brightest programmers fail solving reasonable size problems exactly.

On the other hand, some implementations provide solutions only a few percent away from the optimum on some large "real-life" instances. A condition on the length function that certainly helps both in theory and practice is the triangle inequality. A nonnegative function on the edges that satisfies this inequality is called a *metric* function. The special case of the TSP where G is a complete graph and c is a metric is called the *metric TSP*. For a thoughtful and distracting account of the difficulties and successes of the TSP, see Bill Cook's book [5].

If c is not necessarily a metric function, the TSP is hopeless in general: it is not only *NP*-hard to solve but also to approximate, and even for quite particular lengths, since the Hamiltonian cycle problem in 3-regular graphs is *NP*-hard [11]. The practical context makes it also natural to suppose that c is a metric.

A *ρ-approximation algorithm* for a minimization problem, where $\rho \in \mathbb{R}_+$, $\rho \geq 1$, is a polynomial-time algorithm that computes a solution of value at most ρ times the optimum. The *guarantee* or *ratio* of the approximation is ρ.

The first trace of allowing s and t be different is Hoogeveen's article [16], providing a Christofides type 5/3-approximation algorithm, again in the metric case. There had been no improvement until An, Kleinberg and Shmoys [1] improved this ratio to $\frac{1+\sqrt{5}}{2} < 1.618034$ with a simple algorithm, an ingenious new framework for the analysis, but a technically involved realization.

The algorithm first determines an optimum x^* of the Held-Karp relaxation; writing x^* as a convex combination of spanning trees and applying Christofides' heuristic for each, it outputs the best of the arising tours. For the TSP problem $x^*/2$ dominates *any* possible parity correction, as Wolsey [23] observed, but this is not true if $s \neq t$. However, [1] manages to perturb $x^*/2$, differently for each spanning tree of the constructed convex combination, with small average increase of the length.

We adopt this algorithm and this global framework for the analysis, and develop new tools that essentially change its realization and shortcut the most involved parts. This results in a simpler analysis guaranteeing a solution within 8/5 times the optimum and within less than three pages, at the same time improving Cheriyan, Friggstad and Gao's $13/8 = 1.625$, valid for arbitrary T [4].[1]

We did not fix that the Traveling Salesman visits each vertex exactly once, our problem statement requires only that *every vertex is visited at least once.* This version has been introduced by Cornuéjols, Fonlupt and Naddef [6] and was called the "graphical" TSP. In other words, this version asks for the "shortest spanning

[1] This was the first ratio better than 5/3 for arbitrary T; the proof extended the proof of [1].

Eulerian subgraph" ("tour"), and puts forward an associated polyhedron and its integrality properties, characterized in terms of excluded minors.

This version has many advantages: while the metric TSP is defined on the complete graph, the graphical problem can be sparse, since an edge which is not a shortest path between its endpoints can be deleted; however, it is equivalent to the metric TSP (see Subsection "Tours" below); the length function c does not have to satisfy the triangle inequality; this version has an unweighted special case (all 1 weights), asking for the minimum size (cardinality) of a spanning Eulerian subgraph.

The term "graphic" or "graph-TSP" has eventually been taken by this all 1 special case. We avoid these three terms too close (in Hamming distance) but used in a too diversified way in the literature, different from habits for other problems which also have weighted and unweighted variants called differently.[2]

2 Notation, Terminology and Preliminaries

The set of real numbers is denoted by $\mathbb{R}$; $\mathbb{R}_+$, $\mathbb{Q}_+$ denote the set of non-negative real or rational numbers respectively, and $\mathbb{1}$ denotes the all 1 vector of appropriate dimension. We fix the notation $G = (V, E)$ for the input graph. For $X \subseteq V$ we write $\delta(X)$ for the set of edges with exactly one endpoint in X. If $w : E \longrightarrow \mathbb{R}$ and $A \subseteq E$, then we use the standard notation $w(A) := \sum_{e \in A} w(e)$.

T-joins: For a graph $G = (V, E)$ and $T \subseteq V$ a *T-join* in G is a set $F \subseteq E$ such that $T = \{v \in V : |\delta(v) \cap F| \text{ is odd}\}$. For (G, T), where G is connected, it is well-known and easy to see that a T-join exists if and only if $|T|$ is even [18], [17]. When (G, T) or (G, T, c) are given, we assume that G is a connected graph, $|T|$ is even, and $c : E \longrightarrow \mathbb{Q}_+$, where c is called the *length* function, $c(A)$ $(A \subseteq E)$ is the length of A.

Given (G, T, c), the minimum length of a T-join in G is denoted by $\tau(G, T, c)$. A *T-cut* is a cut $\delta(X)$ such that $|X \cap T|$ is odd. It is easy to see that a T-join and a T-cut meet in an odd number of edges. If in addition c is integer, the maximum number of T-cuts so that every edge e is contained in at most $c(e)$ of them is denoted by $\nu(G, T, c)$. By a theorem of Edmonds and Johnson [8], [18] $\tau(G, T, c) = \nu(G, T, 2c)/2$, and a minimum length T-join can be determined in polynomial time. These are useful for an intuition, even if we only use the weaker Theorem 2 below. For an introduction and more about different aspects of T-joins, see [18], [21], [9], [17].

T-Tours: A *T-tour* $(T \subseteq V)$ of $G = (V, E)$ is a set $F \subseteq 2E$ such that

(i) F is a T-join of $2G$,
(ii) (V, F) is a connected multigraph,

[2] We do not investigate here these unweighted problems. For comparison, however, let us note the guaranteed ratios for the cardinality versions of the problems: the ratio 3/2 has been reached for the minimum size of a T-tour (see the definition a few lines below), and 7/5 for $T = \emptyset$ [22].

where $2E$ is the multiset consisting of the edge-set E, and the multiplicity of each edge is 2; we then denote $2G := (V, 2E)$. It is not false to think about $2G$ as G with a parallel copy added to each edge, but we find the multiset terminology better, since it allows for instance to keep the length function and its notation $c : E \longrightarrow \mathbb{Q}_+$, or in the polyhedral descriptions to allow variables to take the value 2 without increasing the number of variables; the length of a multi-subset will be the sum of the lengths of the edges multiplied by their multiplicities, with obvious, unchanged terms or notations: for instance the size of a multiset is the sum of its multiplicities; χ_A is the multiplicity vector of A; $x(A)$ is the scalar product of x with the multiplicity vector of A; a *subset of a multiset* A is a multiset with multiplicities smaller than or equal to the corresponding multiplicities of A, etc.

A *tour* is a T-tour with $T = \emptyset$.

The *T-tour problem (TTP)* is to minimize the length of a T-tour for (G, T, c) as input. Denote $\text{OPT}(G, T, c)$ this minimum. The subject of this work is the TTP in general.

If $F \subseteq E$, we denote by T_F the set of vertices incident to an odd number of edges in F; if F is a spanning tree, $F(T)$ denotes the *unique T-join of F;* accordingly, $F(s, t) := F(\{s, t\})$ is the (s, t)-path of F.

The *sum* of two (or more) multisets is a multiset whose multiplicities are the sums of the two corresponding multiplicities. If $X, Y \subseteq E$, $X + Y \subseteq 2E$ and $(V, X + Y)$ is a multigraph. Given (G, T), $F \subseteq E$ such that (V, F) is connected,

and a $T_F \triangle T$-join J_F, the multiset $F + J_F$ is a T-tour.

the notation "$\triangle$" stays for the *symmetric difference* (mod 2 sum of sets). This simple operation is the tool for "parity correction".

In [22] T-tours were introduced under the term *connected T-joins*. (This was a confusing term, since T-joins have only 0 or 1 multiplicities.) Even if the main target remains $|T| \leq 2$, the arguments concerning this case often lead out to problems with larger T.

By "Euler's theorem" a subgraph of $2G$ is a tour or $\{s, t\}$-tour if and only if its edges can be ordered to form a closed "walk" or a walk from s to t, that visits every vertex of G at least once, and uses every edge as many times as its multiplicity.

Linear Relaxation: We adopt the polyhedral background and notations of [22], which itself is the adaptation of the so-called "Held-Karp" [15] relaxation to our slightly different context, for slightly improved comfort.

Let $G = (V, E)$ be a graph. For a partition $\mathcal{W}$ of V we introduce the notation $\delta(\mathcal{W}) := \bigcup_{W \in \mathcal{W}} \delta(W)$, that is, $\delta(\mathcal{W})$ is the set of edges that have their two endpoints in different classes of $\mathcal{W}$.

Let G be a connected graph, $T \subseteq V$ with $|T|$ even. Denote

$$P(G,T) := \Big\{ x \in \mathbb{R}^E : \; x(\delta(W)) \geq 2 \text{ for all } \emptyset \neq W \subset V \text{ with } |W \cap T| \text{ even},$$
$$x(\delta(\mathcal{W})) \geq |\mathcal{W}| - 1 \text{ for all partitions } \mathcal{W} \text{ of } V,$$
$$0 \leq x(e) \leq 2 \text{ for all } e \in E \Big\}.$$

Let $x^* \in P(G,T)$ minimize $c^\top x$ on $P(G,T)$.

Fact 1: Given (G,T,c), $\text{OPT}(G,T,c) \geq \min_{x \in P(G,T)} c^\top x = c^\top x^*$.

Indeed, if F is a T-tour, χ_F satisfies the defining inequalities of $P(G,T)$.

The following theorem is essentially the same as Schrijver [21, page 863, Corollary 50.8].

Theorem 1. *Let $x \in \mathbb{R}^E$ satisfy the inequalities*

$$x(\delta(\mathcal{W})) \geq |\mathcal{W}| - 1 \text{ for all partitions } \mathcal{W} \text{ of } V,$$
$$0 \leq x(e) \leq 2 \text{ for all } e \in E.$$

Then there exists a set $\mathcal{F}_+$, $|\mathcal{F}_+| \leq |E|$ of spanning trees and coefficients $\lambda_F \in \mathbb{R}, \lambda_F > 0$, $(F \in \mathcal{F}_+)$ so that

$$\sum_{F \in \mathcal{F}_+} \lambda_F = 1, \qquad x \geq \sum_{F \in \mathcal{F}_+} \lambda_F \chi_F,$$

and for given x as input, $\mathcal{F}_+$, λ_F $(F \in \mathcal{F}_+)$ can be computed in polynomial time.

Proof. Let x satisfy the given inequalities. If $(2 \geq) x(e) > 1$ $(e \in E)$, introduce an edge e' parallel to e, and define $x'(e') := x(e) - 1$, $x'(e) := 1$, and $x'(e) := x(e)$ if $x(e) \leq 1$. Note that the constraints are satisfied for x', and $x' \leq \mathbb{1}$. Apply Fulkerson's theorem [10] (see [21, page 863, Corollary 50.8]) on the blocking polyhedron of spanning trees: x' is then a convex combination of spanning trees, and by replacing e' by e in each spanning tree containing e'; applying then Carathéodory's theorem, we get the assertion. The statement on polynomial solvability follows from Edmonds' matroid partition theorem [7], or the ellipsoid method [13]. □

Note that the inequalities in Theorem 1 form a subset of those that define $P(G,T)$. In particular, any optimal solution $x^* \in P(G,T)$ for input (G,T,c) satisfies the conditions of the theorem. Fix $\mathcal{F}_+$, λ_F provided by the theorem for x^*, that is,

$$\sum_{F \in \mathcal{F}_+} \lambda_F \chi_F \leq x^*.$$

We fix the input (G,T,c) and keep the definitions x^*, $\mathcal{F}_+$, λ_F until the end of the paper.

It would be possible to keep the Held-Karp context of [1] for $s \neq t$ where metrics in complete graphs are kept and only Hamiltonian paths are considered (so the condition $x(\delta(v)) = 2$ if $v \neq s$, $v \neq t$ is added), or the corresponding generalization in [4] for $T \neq \emptyset$. However, we find it more comfortable to have in mind only (G,T,c), where c is the given function which is not necessarily a metric, and G is the original (connected) graph that is not necessarily the complete graph, and T is only required to have even size, with $T = \emptyset$ allowed. The only price to pay for this is to have $\sum_{F \in \mathcal{F}_+} \lambda_F \chi_F \leq x^*$ without the irrelevant

"=". The paper can be also read with the classical Held-Karp definition in mind at the price of minor technical adjustments.

Last, we state a well-known theorem of Edmonds and Johnson for the blocking polyhedron of T'-joins in the form we will use it. (The notation T is now fixed for our input (G, T, c), and the theorem will be applied for several different T' in the same graph.)

Theorem 2. [8], (cf. [18], [21]) *Given* (G, T', c), *let*

$$Q_+(G, T') := \{x \in \mathbb{R}^E : x(C) \geq 1 \text{ for each } T'\text{-cut } C,\ x(e) \geq 0 \text{ for all } e \in E\}.$$

A shortest T'*-join can be found in polynomial time, and if* $x \in Q_+(G, T')$,

$$\tau(G, T', c) \leq c^\top x.$$

Christofides for T-tours: A 2-approximation algorithm for the TTP is trivial by taking a minimum length spanning tree F and doubling the edges of a $T_F \triangle T$-join of F, that is, of $F(T_F \triangle T)$. It is possible to do better by adapting Christofides' algorithm [3], which is usually stated in terms of matchings and in the context of the metric TSP. It can quite easily be generalized to T-tours once the relation of the latter to the metric TSP is clear:

Minimizing the length of a tour or $\{s, t\}$*-tour is equivalent to the metric TSP problem or its path version* (with all degrees 2 except s and t of degree 1, that is, a shortest Hamiltonian cycle or path). Indeed, any length function of a connected graph can be replaced by a function on the complete graph with lengths equal to the lengths of shortest paths (metric completion): then a tour or an $\{s, t\}$-tour can be "shortcut" to a sequence of edges with all inner degrees equal to 2. Conversely, if in the metric completion we have a shortest Hamiltonian cycle or path we can replace the edges by paths and get a tour or $\{s, t\}$-tour.

For $T = \emptyset$, Christofides [3] is equivalent to first determining a minimum length spanning tree F to assure connectivity, and then adding to it a shortest T_F-join. The straightforwardf approximation guarantee 3/2 of this algorithm has not been improved ever since. A *Christofides type algorithm* for general T *adds a shortest* $T_F \triangle T$*-join instead.*

We finish the discussion of TTP with a proof of the 5/3-approximation ratio for Christofides's algorithm. Watch the partition of the edges of a spanning tree into a T-join – if $T = \{s, t\}$, an $\{s, t\}$ path – and the rest of the tree in this proof! For $\{s, t\}$-paths this ratio was first proved by Hoogeveen [16] slightly differently (see for T-tours in the Introduction of [22]), and in [14] in a similar way, as pointed out to me by David Shmoys.

Proposition: *Let* (G, T, c) *be given, and let* F *be an arbitrary shortest spanning tree. Then* $\tau(G, T_F \triangle T, c) \leq \frac{2}{3}\text{OPT}(G, T, c)$.

Proof. $\{F(T), F \setminus F(T)\}$ is a partition of F into a T-join and a $T \triangle T_F$-join (see Fig. 1). The shortest T-tour K has a T_F-join F' by connectivity, so $\{F', K \setminus F'\}$ is a partition of K to a T_F-join and a $T_F \triangle T$-join.

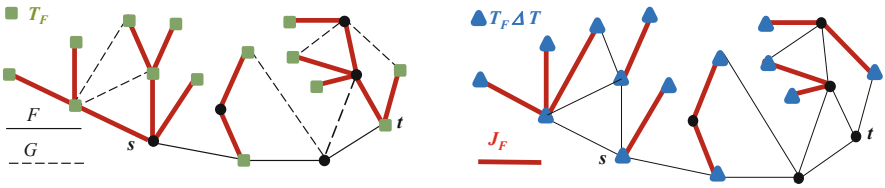


Fig. 1. One of many: $T_F \triangle T$**-joins**, in F (left), minimum in G (right), J_F; $T := \{s, t\}$

If either $c(F \setminus F(T)) \leq \frac{2}{3}c(F)$ or $c(K \setminus F') \leq \frac{2}{3}c(K)$, then we are done, since both are $T \triangle T_F$-joins. If neither hold, then we use the $T \triangle T_F$-join $F(T) \triangle F'$. Since $c(F(T)) \leq \frac{1}{3}c(F) \leq \frac{1}{3}\text{OPT}(G, T, c)$ and $c(F') \leq \frac{1}{3}c(K) = \frac{1}{3}\text{OPT}(G, T, c)$, we have $c(F(T) \triangle F') \leq c(F(T)) + c(F') \leq \frac{2}{3}\text{OPT}(G, T, c)$. □

When $T = \emptyset$ $(s = t)$ Wolsey [23] observed that $x^*/2 \in Q_+(G, T)$ and then by the last inequality of Theorem 2 parity correction costs at most $c^\top x^*/2$, so Christofides's tour is at most 3/2 times $c^\top x^*$; in [1], [4] $\text{OPT}(G, T, c)$ is replaced by $c^\top x^*$ in the Proposition see also the remark after Fact 2 below.

Best of Many Christofides Algorithm (BOM) [1]: Input (G, T, c).

Determine x^* [13] using [2], see [22].
(Recall: x^* is an optimal solution of $\min_{x \in P(G,T)} c^\top x$.)

Determine $\mathcal{F}_+$. (see Theorem 1 and its proof.)

Determine the best *parity correction* for each $F \in \mathcal{F}_+$,
that is, a shortest $T_F \triangle T$-join J_F [8], [17].

Output that $F + J_F$ $(F \in \mathcal{F}_+)$ for which $c(F + J_F)$ is minimum.

The objective value of the T-tour that the BOM algorithm outputs will be upper bounded by the average of the spanning trees in $\mathcal{F}_+$ weighted by the coefficients λ_F $(F \in \mathcal{F})$. The following is a usual, but elegant and useful way of thinking about and working with this average. Our use of it merely notational:

Random Sampling: The coefficient λ_F of each spanning tree $F \in \mathcal{F}_+$ in the convex combination dominated by x^* (see Theorem 1) will be *interpreted as a probability distribution of a random variable* $\mathcal{F}$*,*

$$\Pr(\mathcal{F} = F) := \lambda_F$$

whose values are spanning trees of G*, and*

$$\mathcal{F}_+ = \{F \subseteq E : F \text{ spanning tree of } G,\ \Pr(\mathcal{F} = F) > 0\}.$$

The notations for spanning trees will also be used for random variables whose values are spanning trees. For instance $\mathcal{F}(s, t)$ denotes the random variable whose value is $F(s, t)$ precisely when $\mathcal{F} = F$. Another example is $\chi_{\mathcal{F}}$, a random variable whose value is χ_F when $\mathcal{F} = F$. Similarly, $T_{\mathcal{F}} = T_F$ when $\mathcal{F} = F$.

$$\text{Define } R := \min_{F \in \mathcal{F}_+} \frac{c(F) + \tau(G, T_F \triangle T, c)}{c^\top x^*} \leq \frac{E[c(\mathcal{F}) + \tau(G, T_{\mathcal{F}} \triangle T, c)]}{c^\top x^*} \leq$$

$$\leq 1 + \frac{E[\tau(G, T_{\mathcal{F}} \triangle T, c)]}{c^\top x^*}.$$

Clearly, R is an upper bound for the guarantee of BOM. The main result of the paper is $R \leq 8/5$, and we have just observed that this is implied by $E[\tau(G, T_{\mathcal{F}} \triangle T, c)] \leq 3/5 c^\top x^*$ (Theorem 3).

3 Proving the New Ratio

In this section we prove the result of the paper, the approximation ratio 8/5 for path TSP, achieved by the BOM algorithm. For the simplicity of reading we substitute $\{s,t\}$ for T, and $F(s,t)$ for $F(T)$ without any other change in the proof. The experienced reader can simply change back each occurrence of $\{s,t\}$ to T, and $F(s,t)$ to $F(T)$.

We use now the probability notation for defining two vectors that will be extensively used:

$p^*(e) := \Pr(e \in \mathcal{F}(s,t)); \quad q^*(e) := \Pr(e \in \mathcal{F} \setminus \mathcal{F}(s,t))$ $(e \in E)$.

Fact 2: $E[\chi_{\mathcal{F}(s,t)}] = p^*$, $E[\chi_{\mathcal{F} \setminus \mathcal{F}(s,t)}] = q^*$, $x^* \geq E[\chi_{\mathcal{F}}] = p^* + q^*$. □

Introducing p^* and q^* and observing this fact lead us to a version of the Proposition (end of the previous section) about the expectation of parity correction versus the linear optimum: $E[\tau(G, T_{\mathcal{F}} \triangle \{s,t\}, c)] \leq \frac{2}{3} c^\top q^*$, implying that BOM *outputs a tour of length at most* $\frac{5}{3} c^T x^*$. (Let us sketch the proof (even though we prove our sharper bound in full details below): this inequality follows from $E[\tau(G, T_{\mathcal{F}} \triangle T, c)] \leq \min\{c^\top q^*, c^\top x^* - \frac{c^\top q^*}{2}\}$, which in turn holds because q^* is the mean value of the parity correcting $\mathcal{F} \setminus \mathcal{F}(s,t)$, whereas $c^\top x^* - \frac{c^\top q^*}{2} \geq \frac{c^\top x^*}{2} + \frac{c^\top p^*}{2}$ sums to at least 1 on *every* cut, so the last inequality of Theorem 2 can be applied to both.)

Key definitions, key lemma, key theorem: Define

$$\mathcal{Q} := \{Q \text{ is a cut: } x^*(Q) < 2\}.$$

Every $Q \in \mathcal{Q}$ is an $\{s,t\}$-cut, since non-$\{s,t\}$-cuts C are required to have $x(C) \geq 2$ in the definition of $P(G, \{s,t\})$.[3] Define $x^Q \in \mathbb{Q}_+^E$ with

$$x^Q(e) := \Pr(\{e\} = Q \cap \mathcal{F}).$$

We have from this definition directly that *the support (set of nonzero edges) of x^Q is Q, and* $x^Q(Q) = \sum_{e \in Q} x^Q(e) = \sum_{e \in Q} \Pr(\{e\} = Q \cap \mathcal{F}) = \Pr(|Q \cap \mathcal{F}| = 1)$.

[3] $\mathcal{Q}$ is defined in [1], where its defining vertex-sets are proved to form a chain if $T = \{s,t\}$; in [4] $\mathcal{Q}$ is proved to form a laminar family for general T. These properties are not needed any more.

Lemma: *Let $Q \in \mathcal{Q}$. Then*

(lower bound) $$\mathbb{1}^\top x^Q = x^Q(Q) \geq 2 - x^*(Q),$$

(upper bound) $$\sum_{Q \in \mathcal{Q}} x^Q \leq p^*.$$

Proof. If Q is an arbitrary cut of G (not necessarily in $\mathcal{Q}$), $x^*(Q) = E[|\mathcal{F} \cap Q|] \geq \Pr(|Q \cap \mathcal{F}| = 1) + 2\Pr(|Q \cap \mathcal{F}| \geq 2) = 2 - \Pr(|Q \cap \mathcal{F}| = 1)$, and by the preliminary identity this is equal to $2 - x^Q(Q)$ proving the lower bound for $x^Q(Q)$.

To see the upper bound let us check

$$\sum_{Q \in \mathcal{Q}} \Pr(Q \cap \mathcal{F} = \{e\}) \leq \Pr(e \in \mathcal{F}(s,t)).$$

Indeed, since Q is an $\{s,t\}$-cut, it has a common edge with every $\{s,t\}$-path, so the event $Q \cap \mathcal{F} = \{e\}$ implies $e \in \mathcal{F}(s,t)$; moreover, if $Q_1, Q_2 \in \mathcal{Q}$ are distinct, then the events $Q_1 \cap \mathcal{F} = \{e\}$ and $Q_2 \cap \mathcal{F} = \{e\}$ mutually exclude one another, since for $\mathcal{F} = F$ the set of edges joining the two components of $F \setminus \{e\}$ cannot be equal both to Q_1 and to Q_2. So the left hand side is the probability of the union of disjoint events all of which imply the event on the right hand side, proving the inequality.

Finally, recall $\Pr(e \in \mathcal{F}(s,t)) = p^*(e)$, finishing the proof. □

Theorem 3. $E[\tau(G, T_\mathcal{F} \triangle \{s,t\}, c)] \leq \frac{3}{5} c^\top x^*$.

Proof. We have two upper bounds for $\tau(G, T_F \triangle \{s,t\}, c)$ $(F \in \mathcal{F})$ (Fig. 1). The first is $c(F \setminus F(s,t))$, which is an upper bound because $F \setminus F(s,t)$ is a $T_F \triangle \{s,t\}$-join. The second upper bound will follow as an application of the last inequality of Theorem 2 to a vector z_F, whose feasibility for $P(G, \{s,t\})$ follows from the lower bound of the Lemma, while the length expectation $E[z_\mathcal{F}]$ can be bounded from above by the upper bound of the Lemma. In Case 1 the first bound is small in average, and when it is too large, the second bound is turning out to be small in average (Case 2).

Case 1: $c^\top q^* \leq 3/5\, c^\top x^*$. Then we are done, since

$$E[\tau(G, T_\mathcal{F} \triangle \{s,t\}, c)] \leq E[c(\mathcal{F} \setminus \mathcal{F}(s,t))] = c^\top q^* \leq 3/5\, c^\top x^*.$$

Case 2: $c^\top q^* \geq 3/5\, c^\top x^*$. Then by Fact 2, $c^\top p^* \leq 2/5\, c^\top x^*$.

In this case we construct for every $F \in \mathcal{F}$ a vector $z_F \in Q_+(G, T_F \triangle \{s,t\})$ (Claim). Then the last inequality of Theorem 2 establishes $\tau(G, T_\mathcal{F} \triangle \{s,t\}, c)] \leq c^\top z_F$, and therefore $E[\tau(G, T_\mathcal{F} \triangle \{s,t\}, c)] \leq E[c^\top z_\mathcal{F}]$.

Since $c^T z_F$ will be bounded in terms of $c^\top p^*$, itself bounded from above by $\frac{2}{5} c^\top x^*$ in our Case 2, $E[c^\top z_\mathcal{F}] \leq \frac{3}{5} c^\top x^*$ will follow, establishing the theorem.

Claim: $z_F := \frac{4}{9}x^* + \frac{1}{9}\left(\chi_F + \sum_{Q\in\mathcal{Q}, |Q\cap F|\geq 2} \frac{x^Q}{\Pr(|Q\cap\mathcal{F}|\geq 2)}\right) \in Q_+(G, T_F\triangle\{s,t\})$.

Indeed, we check that the inequalities defining $Q_+(G, T_F\triangle\{s,t\})$ (see Theorem 2) are all satisfied by z_F. Let C be a $T_F\triangle\{s,t\}$-cut.

First, if $C \notin \mathcal{Q}$, then regardless of whether it is a $T_F\triangle\{s,t\}$-cut or not,

$$z_F(C) \geq 2\,\frac{4}{9} + \frac{1}{9} = 1,$$

because then $x^*(C) \geq 2$, and by the connectivity of F, $\chi_F(C) \geq 1$.

Second, if $C \in \mathcal{Q}$ and it is a $T_F\triangle\{s,t\}$-cut, denoting $z := x^C(C)$:

$$z_F(C) \geq \frac{4}{9}(2-z) + \frac{1}{9}(2 + \frac{z}{1-z}) \geq 1,$$

because after evaluating z_F at C, the first term is $\frac{4}{9}x^*(C)$, and then we apply the lower bound of the Lemma. For the second term (first term of the second parenthesis), by the connectivity of F, this time $\chi_F(C) \geq 2$, since $|F\cap C| = 1$ would imply that C is a T_F-cut, which is impossible, since it is an $\{s,t\}$-cut. (A T_F-cut which is a $\{s,t\}$-cut is not a $T_F\triangle\{s,t\}$-cut.) Last (for the first inequality), since $C \in \mathcal{Q}$, the expression

$$\frac{x^C}{\Pr(|C\cap\mathcal{F}|\geq 2)}$$

is among the terms of the definition of z_F, and *leaving only this term of the last* $\sum$ of this definition and recalling $z := x^C(C) = \Pr(|C\cap\mathcal{F}| = 1)$, the checking of the first inequality is finished.

The second inequality ≥ 1 follows now from $0 < z < 1$ and that in this interval the unique minimum of the function in variable z to bound is at $z = 1/2$, when its value is 1, finishing the proof of the claim.

Now by the Claim the last inequality of Theorem 2 can be applied to z_F:

$$E[\tau(G, T_{\mathcal{F}}\triangle\{s,t\}, c)] \leq E[c^\top z_{\mathcal{F}}] \leq \frac{4}{9}c^\top x^* + \frac{1}{9}\,c^\top x^* +$$

$$+\frac{1}{9}\sum_{F\in\mathcal{F}_+}\lambda_F \sum_{Q\in\mathcal{Q}, |Q\cap F|\geq 2} \frac{c^\top x^Q}{\Pr(|Q\cap\mathcal{F}|\geq 2)}.$$

Exchanging the summation signs in this double-sum:

$$\sum_{F\in\mathcal{F}_+}\lambda_F \sum_{Q\in\mathcal{Q}, |Q\cap F|\geq 2} \frac{c^\top x^Q}{\Pr(|Q\cap\mathcal{F}|\geq 2)} = \sum_{Q\in\mathcal{Q}} \Pr(|Q\cap\mathcal{F}|\geq 2)\frac{c^\top x^Q}{\Pr(|Q\cap\mathcal{F}|\geq 2)},$$

and now applying the upper bound of the Lemma and then the bound of our Case 2, we get that this expression is equal to:

$$c^\top \sum_{Q\in\mathcal{Q}} x^Q \leq c^\top p^* \leq \frac{2}{5}c^\top x^*.$$

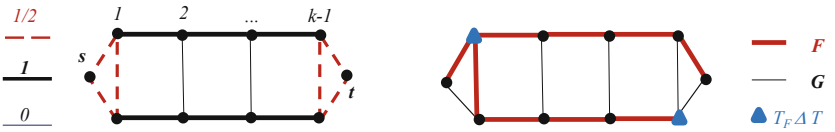


Fig. 2. The approximation guarantee cannot be improved below 3/2 with BOM. This example is essentially the same as the more complicated one in [22, Fig. 3] providing the same lower bound for a more powerful algorithm in the cardinality case. $|V| = 2k$, $\text{OPT}(G, T, \not\vDash) = c^\top x^* = 2k - 1$ (left). BOM output (right): $3k - 2$ if $\mathcal{F}_+$ consists of the thick (red) tree and its central symmetric image. There are more potential spanning trees for $\mathcal{F}_+$, but $\tau(G, T_F \triangle T, \mathbb{1}) \geq k - 2$ for each, so $c(F + J_F) \geq 3k - 3$ for each, and with any $T_F \triangle T$-join J_F.

So we finally got that

$$E[\tau(G, T_{\mathcal{F}} \triangle \{s, t\}, c)] \leq \frac{4}{9} c^\top x^* + \frac{1}{9} c^\top x^* + \frac{1}{9} \frac{2}{5} c^\top x^* = \frac{3}{5} c^\top x^*. \qquad \square$$

In Fig. 2 the optimum and the LP optimum are the same, but the BOM algorithm cannot decrease the approximation guarantee below 3/2.

Some of the questions that arise may be more hopefully tractable than the famous questions of the field:
Can the guarantee of BOM be improved for this problem or for other variants of the TSP ?

Namely are the results of [22] 3/2-*approximating minimum size* T-*tours or* 7/5-*approximating tours* be obtained *by BOM ?*

Could the new methods of analysis that have appeared in the last two years make the so far rigid bound of 3/2 move down at least for *shortest* 2-*edge-connected multigraphs* ?

Acknowledgment. Many thanks to the three IPCO reviewers' criticism concerning the presentation, and to IPCO's page limitation. The resulting reorganization process has eventually led to essential simplification, promptly and efficiently proof-read by Guyslain Naves and Zoli Szigeti, in the last hours. I also used many suggestions of the participants of the Cargèse meeting of Combinatorial Optimization (September 2012) and in particular of Attila Bernáth, R. Ravi, David Shmoys, as well as of Joseph Cheriyan and Zoli Király, concerning a preliminary version[4].

[4] The full manuscript with more motivations and more on the history and the reasons of the particular choices, as well as an analysis of the (im)possibility of mixing several methods, is planned to be submitted to a journal, and to be first presented in "Cahiers Leibniz":

`https://cahiersleibniz.g-scop.grenoble-inp.fr/apps/WebObjects/CahiersLeibnizApplication.woa/`

The previous version of the proof and some of the information that had to be deleted because of space limitation of IPCO can be accessed in the draft at `http://arxiv.org/abs/1209.3523v3`. Nevertheless, the present version is self-contained.

References

1. An, H.-C., Kleinberg, R., Shmoys, D.B.: Improving Christofides' algorithm for the *s-t* path TSP. In: Proceedings of the 44th Annual ACM Symposium on Theory of Computing (2012) (to appear)
2. Barahona, F., Conforti, M.: A construction for binary matroids. Discrete Mathematics 66, 213–218 (1987)
3. Christofides, N.: Worst-case analysis of a new heuristic for the traveling salesman problem. Technical Report 388, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh (1976)
4. Cheriyan, J., Friggstad, Z., Gao, Z.: Approximating Minimum-Cost Connected *T*-Joins, arXiv:1207.5722v1 (cs.DS) (2012)
5. Cook, W.J.: In Pursuit of the Traveling Salesman: Mathematics at the Limits of Computation. Princeton University Press (2012)
6. Cornuéjols, G., Fonlupt, J., Naddef, D.: The traveling salesman problem on a graph and some related integer polyhedra. Mathematical Programming 33, 1–27 (1985)
7. Edmonds, J.: Submodular functions, matroids and certain polyhedra. In: Guy, R., Hanani, H., Sauer, N., Schönheim, J. (eds.) Proceedings of the Calgary International Conference on Combinatorial Structures and Their Applications 1969, pp. 69–87. Gordon and Breach, New York (1970)
8. Edmonds, J., Johnson, E.L.: Matching, Euler tours and the Chinese postman. Mathematical Programming 5, 88–124 (1973)
9. Frank, A.: Connections in Combinatorial Optimization. Oxford University Press (2011)
10. Fulkerson, D.R.: Blocking Polyhedra. In: Graph Theory and Its Applications (Proceedings Advanced Seminar Madison, Wisconsin, 1969; Harris, B. (ed.)), pp. 93–112. Academic Press, New York (1970)
11. Garey, M.R., Johnson, D.S., Tarjan, R.E.: The planar Hamiltonian circuit problem is NP-complete. SIAM Journal on Computing 5, 704–714 (1976)
12. Gharan, S.O., Saberi, A., Singh, M.: A randomized rounding approach to the traveling salesman problem. In: Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science, pp. 550–559 (2011)
13. Grötschel, M., Lovász, L., Schrijver, A.: The ellipsoid method and its consequences in combinatorial optimization. Combinatorica 1(2), 169–197 (1981)
14. Guttmann-Beck, N., Hassin, R., Khuller, S., Raghavachari, B.: Approximation Algorithms with Bounded Performance Guarantees for the Clustered Traveling Salesman Problem. Algorithmica 28, 422–437 (2000)
15. Held, M., Karp, R.M.: The traveling-salesman problem and minimum spanning trees. Operations Research 18, 1138–1162 (1970)
16. Hoogeveen, J.A.: Analysis of Christofides' heuristic, some paths are more difficult than cycles. Operations Research Letters 10(5), 291–295 (1991)
17. Korte, B., Vygen, J.: Combinatorial Optimization, 5th edn. Springer (2012)
18. Lovász, L., Plummer, M.D.: Matching Theory. Akadémiai Kiadó, North-Holland, Budapest, Amsterdam (1986)
19. Mömke, T., Svensson, O.: Approximating graphic TSP by matchings. In: Proceedings of the 52nd Annual Symposium on Foundations of Computer Science, pp. 560–569 (2011)

20. Mucha, M.: $\frac{13}{9}$-approximation for graphic TSP. In: Proceedings of the 29th International Symposium on Theoretical Aspects of Computer Science, pp. 30–41 (2012)
21. Schrijver, A.: Combinatorial Optimization. Springer (2003)
22. Sebő, A., Vygen, J.: Shorter Tours by Nicer Ears: 7/5-approximation for graphic TSP, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs, arXiv:1201.1870v3 (cs.DM) (2012)
23. Wolsey, L.A.: Heuristic analysis, linear programming and branch and bound. Mathematical Programming Study 13, 121–134 (1980)

Fast Deterministic Algorithms for Matrix Completion Problems

Tasuku Soma

Research Institute of Mathematical Sciences,
Kyoto university
tasuku@kurims.kyoto-u.ac.jp

Abstract. Ivanoys, Karpinski and Saxena (2010) have developed a deterministic polynomial time algorithm for finding scalars $x_1, \ldots, x_n$ that maximize the rank of the matrix $B_0 + x_1B_1 + \cdots + x_nB_n$ for given matrices $B_0, B_1, \ldots, B_n$, where $B_1, \ldots, B_n$ are of rank one. Their algorithm runs in $O(m^{4.37}n)$ time, where m is the larger of the row size and the column size of the input matrices.

In this paper, we present a new deterministic algorithm that runs in $O((m+n)^{2.77})$ time, which is faster than the previous one unless n is much larger than m. Our algorithm makes use of an efficient completion method for mixed matrices by Harvey, Karger and Murota (2005). As an application of our completion algorithm, we devise a deterministic algorithm for the multicast problem with linearly correlated sources.

We also consider a skew-symmetric version: maximize the rank of the matrix $B_0 + x_1B_1 + \cdots + x_nB_n$ for given skew-symmetric matrices $B_0, B_1, \ldots, B_n$, where $B_1, \ldots, B_n$ are of rank two. We design the first deterministic polynomial time algorithm for this problem based on the concept of mixed skew-symmetric matrices and the linear delta-covering algorithm of Geelen, Iwata and Murota (2003).

1 Introduction

In the *max-rank matrix completion problems*, or *matrix completion problems* for short, we are given a matrix whose entries may contain indeterminates, and we are to substitute appropriate values to the indeterminates so that the rank of the resulting matrix be maximized. A solution for the matrix completion problem is called a *max-rank completion,* or just a *completion.* Matrices with indeterminates and its completion demonstrate rich properties and applications in various areas: computing the size of maximum matching [14], construction of network codes for multicast problems [9], computing all pairs edge connectivity of a directed graph [2], system analysis for electrical networks [17] and structural rigidity [16].

In this paper, we consider the following three kinds of matrix completion problems.

> **Matrix completion by rank-one matrices:** Max-rank completion for a matrix in the form of $B_0+x_1B_1+\cdots+x_nB_n$, where B_0 is a matrix of arbitrary rank, $B_1, \ldots, B_n$ are matrices of rank one and $x_1, \ldots, x_n$ are indeterminates.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 375–386, 2013.

Mixed skew-symmetric matrix completion: Max-rank completion for a skew-symmetric matrix in which each indeterminate appears once (twice, if we count the symmetric counterpart).

Skew-symmetric matrix completion by rank-two skew-symmetric matrices: Max-rank completion for a matrix in the form of $B_0 + x_1B_1 + \cdots + x_nB_n$, where B_0 is a skew-symmetric matrix of arbitrary rank, $B_1, \ldots, B_n$ are skew-symmetric matrices of rank two and $x_1, \ldots, x_n$ are indeterminates.

For the matrix completion by rank-one matrices, Lovász [15] has solved the special case of $B_0 = 0$ with matroid intersection. For the general case, Ivanoys, Karpinski and Saxena [12] have provided an algebraic approach that yields the first deterministic polynomial time algorithm. The running time is $O(m^{4.37}n)$, where m is the larger of the row size and the column size of given matrices and n is the number of indeterminates.

Mixed skew-symmetric matrices are studied in the works of Geelen, Iwata and Murota [8] and Geelen and Iwata [6]. The former paper provides a deterministic algorithm to compute the rank of mixed skew-symmetric matrices based on the linear delta-matroid parity problem. However, a matrix completion algorithm for mixed skew-symmetric matrices has been unknown.

For the skew-symmetric matrix completion by rank-two skew-symmetric matrices, Lovász [15] has shown that a completion can be found by solving the linear matroid matching problem if $B_0 = 0$. The general case with B_0 being an arbitrary skew-symmetric matrix has been unsolved.

A general matrix completion problem has been shown to be NP-hard by Harvey, Karger and Yekhanin [10], if we allow each indeterminates appears more than once.

1.1 Our Contribution

In this paper, we present new deterministic algorithms for the three matrix completion problems described above. Our approach builds on mixed matrices and mixed skew-symmetric matrices.

First, we prove that the matrix completion by rank-one matrices can be done in $O((m+n)^{2.77})$ time. It is faster than the previous algorithm of Ivanoys, Karpinski and Saxena [12] when $n = O(m^{2.46})$. Our method is based on a reduction to the mixed matrix completion. Furthermore, we provide a min-max theorem for the matrix completion by rank-one matrices. This theorem is a generalization of the result of Lovász [15] for the case $B_0 = 0$. As an application of the matrix completion by rank-one matrices, we devise a deterministic algorithm for the *multicast problem with linearly correlated sources* in *network coding*.

Second, we provide an algorithm for the mixed skew-symmetric matrix completion problem which runs in $O(m^4)$ time. This is the first deterministic polynomial time algorithm for the problem. Our method employs an algorithm for the delta-covering problem, and it can be regarded as a skew-symmetric version

of mixed matrix completion algorithms of Geelen [7] and Harvey, Karger and Murota [9].

Finally, we show that the skew-symmetric matrix completion by rank-two skew-symmetric matrices can be reduced to the mixed skew-symmetric matrix completion. Using this reduction, we design a deterministic polynomial time algorithm, which runs in $O((m+n)^4)$ time.

1.2 Related Works

The beginning of studies for the matrix completion problem was dating back to the works of Edmonds [4] and Lovász [14]. Lovász [14] has showed that random assignment to each indeterminate from a sufficiently large field achieves a max-rank completion with high probability. This randomized completion approach is useful both theoretically and practically. Cheung, Lau and Leung [2] have devised a randomized algorithm to compute edge connectivities for all pairs in a directed graph by the random completion for some matrix constructed from the graph.

While a randomized completion algorithm emerged in the early period of the studies, a satisfactory deterministic algorithm of matrix completion had been open until the end of the twentieth century. Lovász [15] has demonstrated that various completion problems of matrices without constant part admit essential relation between combinatorial optimization problems, along with polynomial time algorithms. Geelen [7] has described the first deterministic polynomial time algorithm for matrices with constant part such that each indeterminate appears only once. Geelen's algorithm takes $O(m^9)$ time and it works only over a field of size at least m. Later, Harvey, Karger and Murota [9] have devised an efficient algorithm for the same setting based on the independent matching problem. Their algorithm runs in $O(m^{2.77})$ time and works over an arbitrary field.

A matrix with indeterminates is called a *mixed matrix* if each indeterminate appears at most once. Mixed matrices are well-studied objects and have a deep connection to linear matroids and bipartite matchings. Murota [17] has presented efficient algorithms to compute the rank of a given mixed matrix and the combinatorial canonical form based on combinatorial properties of mixed matrices. A skew-symmetric version of mixed matrix is called a *mixed skew-symmetric matrix*. Geelen, Iwata and Murota [8] have designed an efficient deterministic algorithm to compute the rank of mixed skew-symmetric matrices with the linear delta-matroid parity problem.

One of the most fruitful application areas of matrix completion is *network coding*, which is a new network communication framework proposed by Ahlswede et al. [1]. They have shown that network coding can achieve the best possible efficiency for the *multicast problem*, in which we have one information source and all sink nodes demand all information of the source. The *multicast problem with linearly correlated sources* is a generalization of the multicast problems. This problem is first considered by Ho et al. [11] and they have shown that the *random network coding* finds a solution with high probability if the field size is sufficiently large. Harvey, Karger and Murota [9] have proposed another

variation of multicast problems called the *anysource multicast problem*. They have designed a deterministic polynomial time algorithm for this problem with the matrix completion technique called the *simultaneous matrix completion*, under some condition for the field size. The linearly correlated multicast can be regarded as a natural generalization of the anysource multicast problem. For further information of network coding, the reader is referred to Yeung [19].

2 Preliminaries

In this section, we introduce the concept and basic facts of mixed matrices and mixed skew-symmetric matrices, along with the corresponding combinatorial optimization problems. For further details, the reader is referred to Murota [17].

2.1 Mixed Matrix

Let $\mathbf{K}$ be a subfield of a field $\mathbf{F}$. A matrix A over $\mathbf{F}$ is called a *mixed matrix* if $A = Q + T$, where Q is a matrix over $\mathbf{K}$ and T is a matrix over $\mathbf{F}$ such that the set of its nonzero entries is algebraically independent over $\mathbf{K}$.

A *layered mixed matrix* (*LM-matrix*) is a mixed matrix whose nonzero rows of Q are disjoint from its nonzero rows of T, i.e., a mixed matrix of the form of $\left[\begin{smallmatrix} Q \\ T \end{smallmatrix}\right]$. Given an $m \times m$ mixed matrix $A = Q + T$, we associate an LM-matrix

$$\tilde{A} := \begin{bmatrix} I_m & Q \\ -Z & T' \end{bmatrix}, \tag{1}$$

where I_m is the identity matrix of size m, $Z := \operatorname{diag}[z_1, \ldots, z_m]$ and $T' := ZT$. We can easily verify that $\operatorname{rank} \tilde{A} = m + \operatorname{rank} A$.

2.2 Independent Matching

We now introduce the *independent matching problem*, which is an equivalent variation of the matroid intersection problem. Let $G = (V^+, V^-; E)$ be a bipartite graph with vertex set $V^+ \dot{\cup} V^-$ and edge set E. Let $\mathbf{M}^+$ and $\mathbf{M}^-$ be matroids on V^+ and V^-, respectively. A matching M in G is said to be *independent* if the sets of vertices in V^+ and V^- incident to M are independent in $\mathbf{M}^+$ and $\mathbf{M}^-$, respectively. The independent matching problem is to find an independent matching of maximum size. The independent matching problem admits a min-max theorem, which is a generalization of the classical König-Egerváry theorem.

Theorem 1 (Welsh [18]). *Let $G = (V^+, V^-; E)$ be a bipartite graph, $\mathbf{M}^+$ and $\mathbf{M}^-$ be matroids on V^+ and V^-, respectively. Then, we have*

$$\begin{aligned} &\max\{|M| : M \text{ is an independent matching}\} \\ &\quad = \min\{r^+(X^+) + r^-(X^-) : (X^+, X^-) \text{ is a cover of } G\}, \end{aligned} \tag{2}$$

where r^+ and r^- are the rank functions of $\mathbf{M}^+$ and $\mathbf{M}^-$, respectively.

2.3 Computing the Rank of Mixed Matrix

The rank of an LM-matrix, and therefore the rank of a general mixed matrix, can be computed by finding an independent matching of maximum size.

For an LM-matrix $A = \left[\begin{smallmatrix} Q \\ T \end{smallmatrix}\right]$, define a bipartite graph $G = (V^+, V^-; E)$ as follows. Put $V^+ := C_Q \cup R_T$ and $V^- := C$, where C and C_Q are the set of column indices and its copy, respectively, and R_T is the set of row indices of T. Let E be the set $\{i_Q i : i_Q \in C_Q \text{ and } i_Q \text{ is the copy of } i\} \cup \{ij : T_{ij} \neq 0\}$. Then, define a matroid $\mathbf{M}^+$ on V^+ as the direct sum of the linear matroid $\mathbf{M}[Q]$ and the free matroid on R_T. Finally, let $\mathbf{M}^-$ be the free matroid on V^-.

We are now ready to state a theorem that reveals a relationship between LM-matrices and independent matchings.

Theorem 2 (Murota [17]). *For an LM matrix* $A = \left[\begin{smallmatrix} Q \\ T \end{smallmatrix}\right]$, *we have*

$$\operatorname{rank} A = \max\{|M| : M \text{ is an independent matching in } G\}.$$

2.4 Mixed Matrix Completion

In the *mixed matrix completion problem*, we are given a mixed matrix A and we are to maximize the rank of the matrix obtained by substituting values to the indeterminates of A.

Harvey, Karger and Murota [9] have developed an elegant algorithm for this problem by constructing an instance of the independent matching problem. Let $\tilde{A} = \left[\begin{smallmatrix} I & Q \\ -Z & T' \end{smallmatrix}\right]$ be the corresponding LM-matrix (1) of A. Construct G, $\mathbf{M}^+$ and $\mathbf{M}^-$ from $\tilde{A}$ as in the previous section. For an independent matching M of G, put $\mathcal{X}_M := \{T_{ij} : \text{the edge corresponding to } T'_{ij} \text{ is contained in } M\}$.

Theorem 3 (Harvey, Karger and Murota [9]). *Let M be a maximum independent matching of the independent matching problem that minimizes $|\mathcal{X}_M|$. Then, substituting 1 to indeterminates in $\mathcal{X}_M$ and substituting 0 to the others yields a max-rank completion.*

Theorem 3 offers a simple algorithm that requires a subroutine to solve the weighted independent matching problem. Using a standard algorithm for the weighted matroid intersection [3], we can find a max-rank completion in $O(m^3 \log m)$ time, where m is the larger of the column size and the row size of the input mixed matrix. With the aid of fast matrix multiplication, the algorithm can be implemented to run in $O(m^{2.77})$ time [5].

2.5 Simultaneous Mixed Matrix Completion

The *simultaneous mixed matrix completion problem* is a more general completion problem: given a collection of mixed matrices that may share indeterminates, we are to maximize the rank of every matrix in the collection by substitutions. Harvey, Karger and Murota [9] showed that a simultaneous mixed matrix completion can be found in polynomial time under some condition of the field size.

Theorem 4 (Harvey, Karger and Murota [9]). *Let $\mathcal{A}$ be a collection of mixed matrices and $\mathcal{X}$ be the set of indeterminates appearing in $\mathcal{A}$. A simultaneous completion for the collection $\mathcal{A}$ can be found deterministically in $O(|\mathcal{A}|(m^3 \log m + |\mathcal{X}|m^2))$ time if the field size is larger than $|\mathcal{A}|$, where m is the maximum of the row size and the column size of matrices in $\mathcal{A}$.*

2.6 Support Graph and Pffafian of Skew-Symmetric Matrix

A matrix A is said to be *skew-symmetric* if A is a square matrix such that $A_{ii} = 0$ for each i and $A_{ij} = -A_{ji}$ for each pair of distinct i and j. The *support graph* of an $m \times m$ skew-symmetric matrix A is an undirected graph $G = (V, E)$ with vertex set $V := \{1, \ldots, m\}$ and edge set $E := \{ij : A_{ij} \neq 0 \text{ and } i < j\}$. The *Pfaffian* of a skew-symmetric matrix A, denoted by $\mathrm{pf}\, A$, is a similar concept of determinants of matrices defined as follows:

$$\mathrm{pf}\, A := \sum_{M} \sigma_M \prod_{ij \in M} A_{ij}, \tag{3}$$

where the sum is taken over all perfect matchings M in the support graph of A and σ_M takes ± 1 in an appropriate manner (see Murota [17]). For each subset I of V, let $A[I]$ denote the submatrix of A whose row and column indexed by I. Such submatrices are called *principal submatrices* of A. The following is basic for skew-symmetric matrices.

Lemma 1. *For a skew-symmetric matrix A, the rank of A equals the maximum size of I such that $A[I]$ is nonsingular, and $\det A = (\mathrm{pf}\, A)^2$ holds.*

2.7 Delta-Matroid

A *delta-matroid* is a generalization of matroids. Let V be a finite set and $\mathcal{F}$ be a non-empty family of subsets of V. The pair $(V, \mathcal{F})$ is called a delta-matroid if it satisfies the following condition:

> For $F, F' \in \mathcal{F}$ and $i \in F \triangle F'$, there exists $j \in F \triangle F'$ such that $F \triangle \{i, j\} \in \mathcal{F}$,

where $F \triangle F' := (F \setminus F') \cup (F' \setminus F)$ denotes the symmetric difference of F and F'. Each member of $\mathcal{F}$ is called a *feasible set* of the delta-matroid $(V, \mathcal{F})$.

We can construct a delta-matroid from an $m \times m$ skew-symmetric matrix A. Let $V := \{1, \ldots, m\}$ and $\mathcal{F}_A := \{I : A[I] \text{ is nonsingular}\}$. Then it is known that $(V, \mathcal{F}_A)$ is a delta-matroid, and we denote this delta-matroid by $\mathbf{M}(A)$.

2.8 Mixed Skew-Symmetric Matrix

Let $\mathbf{K}, \mathbf{F}$ be fields such that $\mathbf{K}$ is a subfield of $\mathbf{F}$. A matrix $A = Q + T \in \mathbf{F}^{m \times m}$ is called a *mixed skew-symmetric matrix* if $Q \in \mathbf{K}^{m \times m}$ and $T \in \mathbf{F}^{m \times m}$ are skew-symmetric and the set $\{T_{ij} : T_{ij} \neq 0 \text{ and } i < j\}$ is algebraically independent over $\mathbf{K}$.

The rank of a mixed skew-symmetric matrix $A = Q + T$ can be characterized in terms of the corresponding delta-matroids $\mathbf{M}(Q)$ and $\mathbf{M}(T)$.

Theorem 5 (Murota [17]). *For a mixed skew-symmetric matrix $A = Q + T$, we have*

$$\operatorname{rank} A = \max\{|F_Q \triangle F_T| : F_Q \in \mathcal{F}_Q \text{ and } F_T \in \mathcal{F}_T\}, \tag{4}$$

where $\mathcal{F}_Q$ and $\mathcal{F}_T$ are the families of feasible sets of $\mathbf{M}(Q)$ and $\mathbf{M}(T)$, respectively.

The maximization that appears in the right-hand side of (4) is the so called *(linear) delta covering problem*. Geelen, Iwata and Murota [8] have devised an algorithm for finding an optimal solution in $O(m^4)$ time, where m is the size of A.

3 Matrix Completion by Rank-One Matrices

In this section, we show a reduction of the matrix completion by rank-one matrices to the mixed matrix completion and devise a faster deterministic polynomial time algorithm. A min-max theorem for the problem is also established. Finally, we consider the multicast problem with linearly correlated sources as an application of the matrix completion by rank-one matrices.

3.1 Reduction to the Mixed Matrix Completion

Let B_0 be an a matrix of arbitrary rank and $B_i = u_i v_i^\top$ be a rank-one matrix for $i = 1, \ldots, n$. We associate the matrix $A = B_0 + x_1 B_1 + \cdots + x_n B_n$ with the mixed matrix $\tilde{A}$ defined as follows:

$$\tilde{A} := \left[\begin{array}{ccc|ccc|c} 1 & & & & & & v_1^\top \\ & \ddots & & & 0 & & \vdots \\ & & 1 & & & & v_n^\top \\ \hline x_1 & & & 1 & & & \\ & \ddots & & & \ddots & & 0 \\ & & x_n & & & 1 & \\ \hline & 0 & & u_1 & \cdots & u_n & B_0 \end{array}\right]. \tag{5}$$

By simple linear algebraic consideration, we obtain the following lemma.

Lemma 2. *The rank of $\tilde{A}$ is equal to $2n + \operatorname{rank} A$.*

Therefore, a max-rank completion of mixed matrix (5) yields an optimal solution of the original completion problem. Obviously, the running time of this algorithm is dominated by that of finding a max-rank completion of $\tilde{A}$. By Theorem 3, this can be done in $O((m+n)^{2.77})$ time with fast matrix multiplication, where m is the larger of the row size and the column size of A. Therefore we obtain the following theorem.

Theorem 6. *An optimal solution of the matrix completion by rank-one matrices can be found in* $O((m+n)^{2.77})$ *time.*

Our approach can be generalized to a collection of matrices, which we call a *simultaneous matrix completion by rank-one matrices*. By the above reduction and Theorem 4, we have the following theorem.

Theorem 7. *Let* $\mathcal{A}$ *be a collection of matrices in the form of* $B_0 + x_1B_1 + \cdots + x_nB_n$*, where* $B_1, \ldots, B_n$ *are rank-one. Let* $\mathcal{X}$ *be the set of indeterminates appearing in* $\mathcal{A}$*. Then, a simultaneous matrix completion by rank-one matrices can be found in* $O\left(|\mathcal{A}|\left((|\mathcal{X}|+m)^3\log(|\mathcal{X}|+m)+|\mathcal{X}|(|\mathcal{X}|+m)^2\right)\right)$ *time if the field size is larger than* $|\mathcal{A}|$*, where* m *is the maximum of the row size and the column size of matrices in* $\mathcal{A}$*.*

3.2 A Min-Max Theorem

In this section, we establish a min-max theorem for the matrix completion by rank-one matrices. A proof is omitted due to the limitation of space.

Theorem 8. *Let* B_0 *be a matrix of arbitrary rank and* $B_i = u_i v_i^\top$ *be a rank-one matrix for* $i = 1, \ldots, n$*. For any subset* $J = \{j_1, \ldots, j_k\} \subseteq \{1, \ldots, n\}$*, let us denote the matrix* $[u_{j_1}, \ldots, u_{j_k}]$ *by* $[u_j : j \in J]$ *and the matrix* $[v_{j_{k+1}}, \ldots, v_{j_n}]$ *by* $[v_j : j \notin J]$*, where* $\{1, \ldots n\} \backslash J = \{j_{k+1}, \ldots, j_n\}$*. For* $A = B_0 + x_1B_1 + \cdots + x_nB_n$*, we have*

$$\begin{aligned}&\max\{\operatorname{rank} A : x_1, \ldots, x_n\}\\ &\quad = \min\left\{\operatorname{rank}\begin{bmatrix} 0 & [v_j : j \notin J]^\top \\ [u_j : j \in J] & B_0 \end{bmatrix} : J \subseteq \{1, \ldots, n\}\right\}. \end{aligned} \tag{6}$$

The following min-max relation due to Lovász [15] is now immediate from Theorem 8.

Theorem 9 (Lovász [15]). *Let* $B_i = u_i v_i^\top$ *be a rank-one matrix for* $i = 1, \ldots, n$*. Then,* $A = x_1B_1 + \cdots + x_nB_n$ *satisfies*

$$\begin{aligned}&\max\{\operatorname{rank} A : x_1, \ldots, x_n\}\\ &\quad = \min\{\dim\langle u_j : j \in J\rangle + \dim\langle v_j : j \in \{1, \ldots, n\} \setminus J\rangle : J \subseteq \{1, \ldots, n\}\},\end{aligned}$$

where $\langle \cdots \rangle$ *denotes the linear span.*

3.3 An Application to Network Coding

In this section we provide an application of the matrix completion by rank-one matrices: a deterministic algorithm for the multicast problem with linearly correlated sources. As is often the case with studies of network coding, we concentrate on finding a *linear* solution, i.e., we assume that messages transmitted in the

network are elements of a finite field and coding operations are restricted to be linear. The following algebraic framework is based on [11,13].

Let $\mathbf{F}$ be a finite field. A row vector $x = [x_1 \cdots x_d] \in \mathbf{F}^d$ is called an *original message*. A *network* is a directed acyclic graph $G = (V, E)$ with node set V and edge set E. Let $S := \{s_1, \ldots, s_r\} \subseteq V$ and $T \subseteq V$ be the sets of *source nodes* and *sink nodes*, respectively. Each source node s_i has correlated messages xC_i, where C_i is a given matrix. Each edge e transmits a scalar message $y_e \in \mathbf{F}$ that is uniquely determined by the messages at its tail node. More precisely, for each edge e, y_e satisfies the following condition:

$$y_e = \begin{cases} \sum_{e':e' \in \mathrm{In}(e)} k_{e',e} y_{e'} + xC_i A_{i,e} & \text{if the tail of } e \text{ is } s_i \in S, \\ \sum_{e':e' \in \mathrm{In}(e)} k_{e',e} y_{e'} & \text{otherwise,} \end{cases} \tag{7}$$

where $\mathrm{In}(e) := \{e' \in E : e' = uv \text{ and } e = vw \text{ for some } u, v \text{ and } w\}$.

Each sink node t has to decode the original message x from the messages $\{y_e : e \in \mathrm{In}(t)\}$. This condition can be represented as follows:

$$x_j = \sum_{e:e \in \mathrm{In}(t)} p_{t,e,j} y_e \quad \text{for } j = 1, \ldots, d, \tag{8}$$

where $\mathrm{In}(t) := \{e \in E : \text{the head of } e \text{ is } t\}$.

Conditions (7) and (8) can be represented with a row vector y and matrices A, C, K and P_t for each $t \in T$, as the following linear equations:

$$y = yK + xCA, \tag{9}$$

$$x = yP_t \quad (t \in T). \tag{10}$$

Our goal is to find matrices A, K and P_t ($t \in T$) satisfying conditions (9) and (10) for an *arbitrary* x. The next lemma is the key observation of our approach. A proof is omitted due to the limitation of space.

Lemma 3. *Matrices A, K and P_t ($t \in T$) satisfy conditions* (9) *and* (10) *for an arbitrary x if and only if $N_t := \left[\begin{smallmatrix} CA & 0 \\ I-K & P_t \end{smallmatrix}\right]$ is nonsingular for each $t \in T$.*

By Lemma 3, finding a simultaneous completion for the collection $\mathcal{N} := \{N_t : t \in T\}$ is equivalent to finding a linear network code for the multicast problem with each nonzero entry of matrices A, K and P_t ($t \in T$) being regarded as indeterminates. Note that N_t is not a mixed matrix since a nonzero entry of A could appear in multiple entries. However, each nonzero entry of A must appear in the same column of N_t. Therefore, N_t can be written as $B_0 + \sum_x xB_x$, where the sum is taken over indeterminates appearing in N_t and B_x is a rank-one matrix for each variable x. Applying Theorem 7 to the collection $\mathcal{N}$, we have the following theorem.

Theorem 10. *A linear code for the multicast problem with linearly correlated sources can be found in polynomial time.*

4 Mixed Skew-Symmetric Matrix Completion

In this section, we give a deterministic algorithm for the mixed skew-symmetric matrix completion. Our method makes use of the linear delta-matroid covering problem.

Let $A = Q+T$ be a mixed skew-symmetric matrix of rank r. An outline of our algorithm is as follows. We set a 0-1 value to carefully chosen indeterminate of T. This is equivalent to updating the constant part Q to a new constant matrix Q'. We argue that Q' has the larger rank than that of Q if we set an appropriate value. By repeating of this process, the rank of the constant part Q' increases gradually and finally reaches r. Then Q' is a completed matrix of maximum rank and the algorithm returns Q'.

The rest of this section describes the details of our algorithm. Let $\mathcal{F}_Q$ and $\mathcal{F}_T$ denote the families of feasible sets of the delta-matroids $\mathbf{M}(Q)$ and $\mathbf{M}(T)$, respectively. Let F_Q and F_T be members of $\mathcal{F}_Q$ and $\mathcal{F}_T$, respectively, with $|F_Q \triangle F_T|$ maximum. Note that $|F_Q \triangle F_T| = r$ from (4). Consider the support graph G of $T[F_T]$. Since $T[F_T]$ is nonsingular, G has a perfect matching M. We show that we can shrink F_T so that F_Q and F_T are disjoint without decreasing the value of $|F_T \triangle F_Q|$.

Lemma 4. *Let j be the vertex of G matched to a vertex $i \in F_Q \cap F_T$ by M. Then, $F_T \setminus \{i,j\}$ is a feasible set of $\mathbf{M}(T)$ and $|F_Q \triangle (F_T \setminus \{i,j\})| = |F_Q \triangle F_T|$.*

Proof. Since $G \setminus \{i,j\}$ has a perfect matching, namely $M \setminus \{ij\}$, $T[F_T \setminus \{i,j\}]$ is nonsingular. Thus $F_T \setminus \{i,j\}$ is feasible in $\mathbf{M}(T)$. Suppose that $j \in F_Q \cap F_T$. Then $|F_Q \triangle (F_T \setminus \{i,j\})| > |F_Q \triangle F_T|$, this contradicts the maximality of $|F_Q \triangle F_T|$. Therefore $j \in F_T \setminus F_Q$ and $|F_Q \triangle (F_T \setminus \{i,j\})| = |F_Q \triangle F_T|$. □

Thus we can assume that F_Q and F_T are disjoint without loss of generality. If F_T is empty, then $r = |F_Q| = \operatorname{rank} Q[F_Q]$ and therefore setting $T := 0$ achieves a max-rank completion. So we consider the case that F_T is nonempty. Let ij be an edge of M. Substituting a value α to T_{ij} (and $-\alpha$ to T_{ji}) is equivalent to adding α to Q_{ij} (and $-\alpha$ to Q_{ji}). Let Q' be the matrix obtained from Q by replacing Q_{ij} and Q_{ji} with $Q_{ij} + \alpha$ and $Q_{ji} - \alpha$, respectively. The following lemma offers the core of our completion algorithm.

Lemma 5. *For any edge ij of M, there exists a value $\alpha \in \{0,1\}$ such that $Q'[F_Q \cup \{i,j\}]$ is nonsingular.*

Proof. From the definition of Pfaffian (3), we have the following identity:

$$\operatorname{pf} Q'[F_Q \cup \{i,j\}] = \operatorname{pf} Q[F_Q \cup \{i,j\}] + (-1)^k \alpha \cdot \operatorname{pf} Q[F_Q], \tag{11}$$

where k is some integer uniquely determined by i and j. Since F_Q is a feasible set of $\mathbf{M}(A)$, $Q[F_Q]$ is nonsingular. Thus $\operatorname{pf} Q[F_Q]$ is nonzero. Set $\alpha := 0$ if $\operatorname{pf} Q[F_Q \cup \{i,j\}] \neq 0$, and otherwise set $\alpha := 1$. This ensures that $\operatorname{pf} Q'[F_Q \cup \{i,j\}] \neq 0$. □

Note that $\operatorname{rank} Q'[F_Q \cup \{i,j\}] = \operatorname{rank} Q[F_Q] + 2$. Applying Lemma 5 to every edge of M, we obtain a skew-symmetric matrix Q' such that $\operatorname{rank} Q'[F_Q \cup F_T] = \operatorname{rank} Q[F_Q] + 2|M| = |F_Q| + |F_T| = r$. Therefore, Q' is a max-rank completion of A.

Now we analyze the running time for an $m \times m$ mixed skew-symmetric matrix. An optimal pair of F_Q and F_T can be found in $O(m^4)$ time by the algorithm of Geelen, Iwata and Murota [8]. A perfect matching M can be found in $O(m^3)$ time. Since the Pfaffian of a principal submatrix of Q' can be computed in $O(m^3)$ time and $|M| = m/2$, the iteration of setting values to indeterminates of T takes $O(m^4)$ time. Thus we obtain the following theorem.

Theorem 11. *A max-rank completion for an $m \times m$ mixed skew-symmetric matrix can be found in $O(m^4)$ time.*

5 Skew-Symmetric Matrix Completion by Rank-Two Skew-Symmetric Matrices

In this section, we consider the skew-symmetric matrix completion by rank-two skew-symmetric matrices. We show that this problem can be reduced to the mixed skew-symmetric matrix completion as in the case of the matrix completion by rank-one matrices. Let $A := B_0 + x_1 B_1 + \cdots + x_n B_n$, where B_0 is an $m \times m$ skew-symmetric matrix and $B_1, \ldots, B_n$ are skew-symmetric matrices of rank two. Fitst, note that for $i = 1, \ldots, n$, there exists some vectors u_i and v_i such that $B_i = u_i v_i^\top - v_i u_i^\top$.

Let $\tilde{A}$ be the following mixed skew-symmetric matrix:

$$\tilde{A} := \begin{bmatrix}
\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} & & & \begin{bmatrix} -v_1^\top \\ \mathbf{0}^\top \end{bmatrix} & \begin{bmatrix} 0 & 0 \\ x_1 & 0 \end{bmatrix} & & \\
& \ddots & & \vdots & & \ddots & \\
& & \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} & \begin{bmatrix} -v_n^\top \\ \mathbf{0}^\top \end{bmatrix} & & \begin{bmatrix} 0 & 0 \\ x_n & 0 \end{bmatrix} & \\
\begin{bmatrix} v_1 & \mathbf{0} \\ 0 & -x_1 \\ 0 & 0 \end{bmatrix} & \cdots & \begin{bmatrix} v_n & \mathbf{0} \end{bmatrix} & B_0 & \begin{bmatrix} \mathbf{0} & u_1 \end{bmatrix} & \cdots & \begin{bmatrix} \mathbf{0} & u_n \end{bmatrix} \\
& & & \begin{bmatrix} \mathbf{0}^\top \\ -u_1^\top \end{bmatrix} & \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} & & \\
& \ddots & & \vdots & & \ddots & \\
& & \begin{bmatrix} 0 & -x_n \\ 0 & 0 \end{bmatrix} & \begin{bmatrix} \mathbf{0}^\top \\ -u_n^\top \end{bmatrix} & & \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} &
\end{bmatrix}. \tag{12}$$

By a sequence of basic operations for $\tilde{A}$, one can easily obtain the following lemma.

Lemma 6. *The skew-symmetric matrices A and $\tilde{A}$ satisfy* $\operatorname{rank} \tilde{A} = 4n + \operatorname{rank} A$.

Thus a max-rank completion for $\tilde{A}$ yields a max-rank completion for A. Using the completion algorithm of Section 4, we can obtain a completion for A.

Theorem 12. *A solution for the skew-symmetric matrix completion by rank-two skew-symmetric matrices can be found in* $O((m+n)^4)$ *time.*

References

1. Ahlswede, R., Cai, N., Li, S.Y.R., Yeung, R.W.: Network information flow. IEEE Transactions on Information Theory 46, 1204–1216 (2000)
2. Cheung, H.Y., Lau, L.C., Leung, K.M.: Graph connectivities, network coding, and expander graphs. In: Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science, pp. 190–199 (2011)
3. Cunningham, W.H.: Improved bounds for matroid partition and intersection algorithms. SIAM Journal on Computing 15, 948–957 (1986)
4. Edmonds, J.: Systems of distinct representatives and linear algebra. Journal of Research of the National Bureau of Standards B71, 241–245 (1967)
5. Gabow, H.N., Xu, Y.: Efficient theoretic and practical algorithms for linear matroid intersection problems. Journal of Computer and System Sciences 53, 129–147 (1996)
6. Geelen, J., Iwata, S.: Matroid matching via mixed skew-symmetric matrices. Combinatorica 25, 187–215 (2005)
7. Geelen, J.F.: Maximum rank matrix completion. Linear Algebra and Its Applications 288, 211–217 (1999)
8. Geelen, J.F., Iwata, S., Murota, K.: The linear delta-matroid parity problem. Journal of Combinatorial Theory B88, 377–398 (2003)
9. Harvey, N.J.A., Karger, D.R., Murota, K.: Deterministic network coding by matrix completion. In: Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 489–498 (2005)
10. Harvey, N.J.A., Karger, D.R., Yekhanin, S.: The complexity of matrix completion. In: Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm, pp. 1103–1111 (2006)
11. Ho, T., Médard, M., Koetter, R., Karger, D.R., Effros, M., Shi, J., Leong, B.: A random linear network coding approach to multicast. IEEE Transactions on Information Theory 52, 4413–4430 (2006)
12. Ivanyos, G., Karpinski, M., Saxena, N.: Deterministic polynomial time algorithms for matrix completion problems. SIAM Journal on Computing 39, 3736–3751 (2010)
13. Koetter, R., Médard, M.: An algebraic approach to network coding. IEEE/ACM Transactions on Networking 11, 782–795 (2003)
14. Lovász, L.: On determinants, matchings and random algorithms. In: Fundamentals of Computation Theory, FCT, pp. 565–574 (1979)
15. Lovász, L.: Singular spaces of matrices and their application in combinatorics. Bulletin of the Brazilian Mathematical Society 20, 87–99 (1989)
16. Lovász, L., Yemini, Y.: On generic rigidity in the plane. SIAM Journal on Algebraic and Discrete Methods 1, 91–98 (1982)
17. Murota, K.: Matrices and Matroids for System Analysis, 2nd edn. Springer, Berlin (2009)
18. Welsh, D.: On matroid theorems of Edmonds and Rado. Journal of the London Mathematical Society S2, 251–256 (1970)
19. Yeung, R.W.: Information Theory and Network Coding. Springer, Berlin (2008)

Approximating the Configuration-LP for Minimizing Weighted Sum of Completion Times on Unrelated Machines

Maxim Sviridenko[1,⋆] and Andreas Wiese[2]

[1] University of Warwick
M.I.Sviridenko@warwick.ac.uk
[2] MPI für Informatik, Saarbrücken
awiese@mpi-inf.mpg.de

Abstract. Configuration-LPs have proved to be successful in the design and analysis of approximation algorithms for a variety of discrete optimization problems. In addition, lower bounds based on configuration-LPs are a tool of choice for many practitioners especially those solving transportation and bin packing problems. In this work we initiate a study of linear programming relaxations with exponential number of variables for unrelated parallel machine scheduling problems with total weighted sum of completion times objective. We design a polynomial time approximation scheme to solve such a relaxation for $R|r_{ij}|\sum w_jC_j$ and a fully polynomial time approximation scheme to solve a relaxation of $R||\sum w_jC_j$. As a byproduct of our techniques we derive a polynomial time approximation scheme for the one machine scheduling problem with rejection penalties, release dates and the total weighted sum of completion times objective.

1 Introduction

In unrelated parallel machine scheduling we are given a set of m machines and a set of n jobs. Each job j is characterized by a processing time $p_{i,j} \in \mathbb{N}$ for each machine i, i.e. it takes $p_{i,j}$ time units to process job j on machine i, by a weight $w_j \in \mathbb{N}$, and by a release time $r_{i,j} \in \mathbb{N}$ for each machine i. The goal is to assign the jobs to the machines and to define a non-preemptive schedule for each machine, such that on every machine i each job j starts at time $r_{i,j}$ or later. Each machine can process at most one job at a time. Given a schedule S, we denote by $C_j(S)$ the completion time of each job j. We write C_j for short if the schedule S is clear from the context. The objective is to compute a schedule S^* which minimizes the total weighted sum of completion times $\sum_j w_j \cdot C_j(S^*)$. Using the standard scheduling notations [6] this problem is commonly denoted as $R|r_{ij}|\sum w_jC_j$.

⋆ Research supported by EPSRC grant EP/J021814/1, FP7 Marie Curie Career Integration Grant and Royal Society Wolfson Research Merit Award.

M. Goemans and J. Correa (Eds.): IPCO 2013, LNCS 7801, pp. 387–398, 2013.

Unrelated parallel machines scheduling is one of the basic scheduling models that is extensively studied by the researchers both from experimental and theoretical viewpoints. Various applications dictate different objectives such as makespan, total throughput etc. For the problem with the total weighted sum of completion times objective Schulz and Skutella [15] and Skutella [17] design 2-approximation algorithms for the general problem and 3/2-approximation algorithms for the problem where all $r_{ij} = 0$, i.e. when all jobs are released at time zero (the standard notation for this scheduling model is $R||\sum w_j C_j$). Their algorithms are based on time indexed linear programming [15] and convex programming [17] relaxations.

On the other side, recent improvements of the performance ratios for unrelated parallel machine scheduling problems with other objectives using linear programming relaxations with exponential number of variables (so-called configuration-LPs) motivated us to consider and study such configuration linear programming relaxations for $R|r_{ij}|\sum w_j C_j$.

In particular, for the restricted assignment special case of the unrelated parallel machine scheduling problem with makespan objective Svensson [19] showed that the configuration-LP has an integrality gap strictly better than 2, improving upon a long standing result by Lenstra, Shmoys and Tardos [12] that was based on a generalized assignment linear programming relaxation. Also many recent results for the Santa Claus problem, i.e. unrelated parallel machine scheduling problem with maxmin objective, are based on configuration-LPs [2,3,10].

In addition to that, many sophisticated transportation problems are solved in practice by using configuration linear programming relaxations (see e.g. [9]). The following meta-algorithm is the algorithm of choice used by many practitioners:

1. formulate your problem using an exponential number of variables where each variable encodes a non-trivial piece of the solution space (truck route, schedule on one machine etc.);
2. generate a set of variables (or "columns") to consider by running a set of heuristics and solve the linear program corresponding to this subset of variables;
3. fix some of the variables to zero or one and repeat the process.

This heuristic algorithm performs amazingly well in practice for a wide variety of problems which indicates the high quality of configuration-LPs as relaxations of the original problem at hand.

In this paper, motivated by the above considerations, we define configuration linear programming relaxations for $R|r_{ij}|\sum w_j C_j$. The first question is how to solve such relaxations. We use the well-known connection between separation and optimization [7,8,13]. We define a dual linear program and notice that the separation problem for the dual corresponds to an interesting NP-hard one machine scheduling problem with rejections. Similar scheduling problems were considered before in the literature [5,11,16]. Unfortunately, known techniques cannot be applied for our scheduling problem with rejection penalties since to design an approximate separation oracle we are allowed to relax only the piece of objective function corresponding to the weighted sum of completion times (but not the rejection penalties).

We explain the connection between approximate separation and solving our relaxation in the next section. The main result of this paper is a polynomial time approximation scheme for solving configuration-LP relaxation of $R|r_{ij}|\sum w_jC_j$. Recall, that a PTAS is a collection of algorithms such that for any $\varepsilon > 0$ there exists a polynomial time $(1+\varepsilon)$-approximation algorithm in the collection. Such a scheme is called fully polynomial time approximation scheme (FPTAS) if the dependence on $1/\varepsilon$ is polynomial. In addition to our main result we design an FPTAS for solving configuration-LP relaxation of $R||\sum w_jC_j$.

We conjecture that the worst case integrality gaps of our linear programming relaxations are strictly better than the worst case integrality gaps of the linear programming relaxations for $R|r_{ij}|\sum w_jC_j$ and $R||\sum w_jC_j$, previously considered in the literature [15,17]. We also believe that such new relaxations will be instrumental in building new practical algorithms for a wide variety of scheduling problems the same way as they were instrumental in solving many practical transportation problems.

1.1 Our Contribution

For any $\epsilon > 0$ we give a polynomial time algorithm which computes a $(1+\epsilon)$-approximation of the configuration-LP relaxation of $R|r_{ij}|\sum w_jC_j$. Key to this is a polynomial time approximation scheme for the separation problem of the dual which is the *scheduling problem with rejection penalties* and release dates, denoted by $1|r_j|\sum_S w_jC_j + \sum_{\bar{S}} e_j$ in the three-field notation. In that problem, one is given a machine and a set of jobs J as above, where additionally each job j has a *rejection penalty* e_j. The goal is to select a subset $J' \subseteq J$ for which we construct a schedule S. The objective is to minimize $\sum_{j\in J'} w_j \cdot C_j(S) + \sum_{j\in J\setminus J'} e_j$. In other words, we can decide to reject, i.e., not to schedule some job j but then we have to pay e_j as a penalty. Of course, this problem is related to the same problem without rejection, i.e., $1|r_j|\sum_j w_jC_j$, for which a PTAS is known [1]. The latter algorithm is crucially based on the fact that we can assume that not too many jobs are released at the same time. Roughly speaking, if in an instance of $1|r_j|\sum_j w_jC_j$ many jobs have the same release date then we can postpone the release of some jobs with e.g., jobs with small weight, since in an optimal solution they are not scheduled straight away. However, if rejection is allowed this argument breaks down completely since in an optimal schedule jobs with high weight might be rejected and ones with smaller weight might be scheduled immediately. Note that scheduling problems with rejection penalties were considered before but all known techniques are not applicable for our purposes (except a pseudo-polynomial algorithm [5] for $1||\sum_S w_jC_j + \sum_{\bar{S}} e_j$).

Hence, new methods are needed to solve the separation problem. First, like in [1] we split the time horizon into intervals of subsequent powers of $1+\epsilon$. Then we show that at the loss of a factor of $1+O(\epsilon)$ in the objective we can split the whole problem into disjoint subproblems which span $O(\log n)$ intervals each. In such a subproblem, we enumerate over the patterns given by the big jobs and the space for small jobs in the optimal solution (big and small refers to the size

of a job with respect to the interval in which it is scheduled). The number of possible such patterns is bounded by a polynomial. Note that enumerating the assignment of the big jobs directly would yield a quasi-polynomially number of options which is too much. Then we use a linear program to assign the jobs into the slots. In particular, we use an LP to decide which jobs are big and which jobs are small in the optimal solution (that is a very important piece of information).

We believe that our new techniques extend the understanding of scheduling problems with a sum-of-completion-time objective. Since we do not use any properties that rely on the rejection cost term in our objective function, our methods might be useful for other settings as well.

2 The Configuration-LP

Our goal is to construct a $(1+\epsilon)$-approximation algorithm for solving the configuration-LP for minimizing the weighted sum of completion times on unrelated machines. Given an instance of the problem, we denote by J and M a set of given jobs and machines, respectively. For each machine i we denote by $\mathcal{S}(i)$ the set of all feasible schedules for machine i (for any subset of the given jobs). For each schedule S for some set of jobs J' on some machine i we define $W_{i,S} := \sum_{j \in J'} w_j \cdot C_j(S)$.

The configuration-LP, or C-LP for short, is then defined by

$$\begin{aligned}
\min \sum_{i \in M} \sum_{S \in \mathcal{S}(i)} & y_{i,S} \cdot W_{i,S} \\
\sum_{S \in \mathcal{S}(i)} & y_{i,S} \leq 1 && \forall i \in M \\
\sum_{i \in M} \sum_{S \in \mathcal{S}(i): j \in S} & y_{i,S} \geq 1 && \forall j \in J \\
& y_{i,S} \geq 0 && \forall i \in M, S \in \mathcal{S}(i)
\end{aligned}$$

where we write $j \in S$ if job j arises in S.

The configuration-LP has only a linear number of constraints but an exponential number of variables. Hence, we cannot solve it directly. Therefore, instead we solve the dual via the ellipsoid method and a polynomial time separation routine. The dual of the configuration-LP is given by

$$\begin{aligned}
\max \sum_{j \in J} \beta_j - \sum_{i \in M} & \alpha_i \\
-\alpha_i + \sum_{j \in S} & \beta_j \leq W_{i,S} && \forall i \in M, \forall S \in \mathcal{S}(i) && (1) \\
& \alpha_i \geq 0 && \forall i \in M \\
& \beta_j \geq 0 && \forall j \in J.
\end{aligned}$$

In the separation problem for the dual, for each machine i and given values for variables α_i and β_j, we either want to find a schedule $S \in \mathcal{S}(i)$ such that $-\alpha_i + \sum_{j \in S} \beta_j > W_{i,S}$ or assert that for each schedule $S \in \mathcal{S}(i)$ it holds that $-\alpha_i + \sum_{j \in S} \beta_j \leq W_{i,S}$. This problem is equivalent to the problem of scheduling with rejection on one machine to minimize the weighted sum of completion time plus the sum of the rejection penalties, where for each job j the rejection penalty e_j equals β_j. It is NP-hard (as already $1|r_j|\sum w_j C_j$ is NP-hard). Similar scheduling problems were studied before under the name of scheduling with rejection [5]. However, for the purpose of approximating the configuration-LP up to an error of $1 + O(\epsilon)$ we use the following strategy: first, we introduce some modifications of the instances and the schedules under consideration which simplify the structure and cost at most a factor of $1+O(\epsilon)$ in the objective. Then we formulate a relaxed linear program C'-LP with the properties that

- the optimal solution of C'-LP is by at most a factor of $1 + O(\epsilon)$ larger than the optimal solution of C-LP ,
- C'-LP can be solved optimally in polynomial time using a separation oracle, and
- any feasible solution of C'-LP can be transformed into a feasible solution of C-LP at the cost of at most a factor of $1 + \epsilon$ in the objective.

We start with simplifications of the input and the considered schedules.

3 Restrictions for Input and Considered Configurations

Let $\epsilon > 0$ and assume for simplicity that $1/\epsilon \in \mathbb{N}$. We prove that we can assume some properties for the input and the schedules under consideration while losing only a factor of $1 + O(\epsilon)$ in the objective. We extend the big-O notation by writing $O_\epsilon(f(n))$ for functions which are in $O(f(n))$ if ϵ is a constant. E.g., $O_\epsilon(1)$ denotes constants which might depend on ϵ.

We define $R_x := (1 + \epsilon)^x$ and an interval $I_x = [R_x, R_{x+1})$ for each integer x. Observe that $|I_x| = \epsilon \cdot R_x$. In the sequel, several times we will stretch time by some factor $1 + \epsilon$. This means that we take a given (e.g., optimal) schedule and shift all work done in every interval $[a, b)$ to the interval $[(1 + \epsilon)a, (1 + \epsilon)b)$. Since then every job j is processed for $(1 + \epsilon)p_j$ time units, we gain slack in the schedule which we can use in order to obtain certain properties. We will write "at $1+\epsilon$ loss" or "at $1+O(\epsilon)$ loss" if we can assume a certain property for the input or the considered schedules by stretching time by a factor of $1 + \epsilon$ or $1 + O(\epsilon)$, respectively.

Proposition 1. *At $1+\epsilon$ loss we can work with the alternative objective function $\sum_{j \in S} w_j \cdot \min\{R_x : C_j(S) \leq R_x\}$ instead of $\sum_{j \in J} w_j C_j(S)$.*

In the next lemma we round the input data and establish that–intuitively speaking–large jobs are released late (since they have a relatively large completion time anyway).

Lemma 1 ([1]). *At $1+O(\epsilon)$ loss we can assume that all processing times and release dates are powers of $1+\epsilon$ and $r_j \geq \epsilon \cdot p_j$ for each job j.*

Let S be a schedule. We define a job j to be *large in S* if j starts in an interval I_x such that $p_j > \epsilon \cdot I_x$ and *small in S* otherwise. In the following lemmas we will stretch time several times in order to gain free space that we will use in order to enforce certain properties of the schedules under consideration. A technical problem is that when we stretch time by a factor $1+\epsilon$ then a large job can become small. To avoid this, we stretch time *once* by the total factor $(1+\epsilon)^{O(1)}$ that is needed by *all* subsequent lemmas. In the resulting schedule we classify jobs to be small or large as defined above. We will write in the statements of the subsequent lemmas "at $1+\epsilon$ loss" when we mean that we use an extra space of $\epsilon \cdot I_x$ in each interval I_x in order to ensure some property. In fact, when stretching time by a factor $1+O(\epsilon)$ we gain an idle period of total length $\epsilon \cdot I_x$ only in intervals I_x where a job finishes. However, it will turn out that only in those intervals we need this extra space.

Lemma 2. *At $1+\epsilon$ loss we can restrict to schedules where for each interval I_x*

- *each large job j, starting during I_x, is started at a time $t = R_x(1+k \cdot \epsilon^3)$ for some $k \in \{0, ..., \frac{1}{\epsilon^2}\}$, and*
- *there is a time interval $[a,b) \subseteq I_x$ in which no large jobs are scheduled and no small jobs are scheduled in $I_x \setminus [a,b)$ (note that the interval [a,b) could be empty).*

Proof. We first ensure the second property by moving the large and small jobs of the schedule for I_x such that all small jobs are scheduled in a (consecutive) interval $[a,b) \subseteq I_x$. Note that since we changed the objective function (Proposition 1) this does not increase the objective value. In each interval I_x there can be at most $\frac{I_x}{\epsilon \cdot I_x} = 1/\epsilon$ jobs that start in I_x as large jobs. Then, using a free space of at most $\frac{1}{\epsilon} R_x \epsilon^3 = \epsilon \cdot I_x$, we move the start time of each large job to the next value t of the form of the lemma statement. □

In the next lemma we establish that each job is pending for at most $O_\epsilon(\log n)$ intervals.

Lemma 3. *At $1+\epsilon$ loss we can assume that there is an integer integer $K \in O_\epsilon(\log n)$ such that each job released at some time R_x finishes during the interval I_{x+K} the latest.*

Proof. Recall that due to Lemma 1 we have $p_j \leq \frac{1}{\epsilon} \cdot r_j$ for each job j. Hence, $p_j \leq \frac{\epsilon}{n} \cdot r_j \cdot (1+\epsilon)^K$ for some integer $K \in O_\epsilon(\log n)$. Since at most n jobs are released at each time R_x, all these jobs fit into an empty space of $\epsilon \cdot I_{x+K}$ in the interval I_{x+K}. □

The crucial part is now to decouple the instance into blocks of $O(\frac{1}{\epsilon} \log n)$ consecutive intervals each such that each block represents a subinstance which is independent of all the other blocks.

We will use the next lemma to identify groups of C consecutive intervals each, such that each two groups are separated by at least c intervals and all intervals *not* in some group contribute only negligibly towards the objective. For any desired separation c and any bound δ on the contribution we can find a suitable value C. While the next lemma works for any values c and δ, for later the reader may think of $c = O(\log n)$ and $C = O(\frac{1}{\epsilon} \log n)$.

Lemma 4. *Consider any (fractional) solution to C-LP. For every $\delta > 0$ and every integer c there exist a value $C \in O(\frac{1}{\delta}c)$ and an offset $a \in \{0, ..., c + C\}$ such that all jobs released or scheduled during an interval I_x with $a + k \cdot C \leq x \leq a + k \cdot C + c$ for some integer k contribute only a δ-fraction to the overall objective.*

Proof. Can be shown using the pigeon hole principle. □

We define $c := K + L$ where L is the smallest integer such that $\frac{1}{\epsilon^2} \leq (1 + \epsilon)^L$. For a value $\delta \in O_\epsilon(1)$ to be defined later let C and a denote the values given by Lemma 4 for c and δ. Note that since $c + C \in O(\frac{1}{\delta}c)$ we can try all possible values for a in polynomial time. For the remainder of our reasoning we assume that we guessed a correctly. We call an interval I_x a *gap-interval* if $a + k \cdot C \leq x \leq a + k \cdot C + c$ for some integer k.

Lemma 5. *For any $\epsilon > 0$ there is a value $\delta_0 > 0$ such that if $\delta \leq \delta_0$ then at $1 + O(\epsilon)$ loss we can assume that in each gap-interval only small jobs are executed.*

Proof. We shift the large jobs scheduled in each gap-interval I_x by L intervals to the future. Observe that due to the choice of L we have that $I_x = \epsilon R_x \leq \epsilon^2 \cdot I_{x+L}$. By stretching time once by a factor of $1+2\epsilon$ we gain enough space in the interval I_{x+L} to fit all large jobs finished in I_x (we need at most $R_x + I_x \leq 2\epsilon I_{x+L}$ time). Since we move only large jobs, it still holds that each job released at a time R_x finishes during the interval I_{x+K} the latest. By choosing δ to be at most $\delta_0 := \epsilon/(1+\epsilon)^L$ then $\delta \cdot (1+\epsilon)^L \leq \epsilon$ and the increase of the total cost is bounded by $\epsilon \cdot OPT$. □

For any integer k we say that all intervals I_x with $a + k \cdot C \leq x \leq a + k \cdot C + c$ lie in the same *gap-block*.

Lemma 6. *For any $\epsilon > 0$ there is a value $\delta_1 > 0$ such that if $\delta \leq \delta_1$ then at $1+\epsilon$ loss we can enforce that at the end of each gap-interval I_x there is an auxiliary interval of length $\epsilon \cdot I_x$. All jobs released during the gap-block and processed within the same gap-block are only allowed to be processed in the auxiliary intervals. These auxiliary intervals are not allowed to process any other job. Also, each job finishes at most $K + L$ intervals after its release.*

Proof. Consider a gap-block $B = \{I_{a+k \cdot C}, ..., I_{a+k \cdot C+c}\}$. Denote by J_B all (small) jobs which are released and scheduled during B. Similarly as in Lemma 5 we shift them by $L = O_\epsilon(1)$ intervals to the future such that all jobs from J_B, scheduled during an interval $I_x \in B$, have a total volume of $\epsilon \cdot I_{x+L}$, assuming an appropriate upper bound for δ. □

We choose $\delta := \min\{\delta_0, \delta_1\}$. The above lemmas split the overall problem into *blocks*, one for all jobs j with $R_{a+k\cdot C} \leq r_j < R_{a+(k+1)\cdot C}$ for some integer k. In the next definition we summarize the problem we are facing in each block.

Definition 1 (Block-Problem). *We are given m unrelated machines, a set of jobs J and an integer k such that $R_{a+k\cdot C} \leq r_j < R_{a+(k+1)\cdot C}$ for all $j \in J$. We want to find a feasible schedule which on each machine*

- *during each interval $I_x \in \{I_{a+k\cdot C}, ..., I_{a+k\cdot C+c} - 1\}$ may use only an interval of length $\epsilon \cdot I_x$ at the end of I_x (and may schedule only small jobs there),*
- *during each interval $I_x \in \{I_{a+k\cdot C+c}, ..., I_{a+(k+1)\cdot C} - 1\}$ may use the entire interval I_x, and*
- *during each interval $I_x \in \{I_{a+(k+1)\cdot C}, ..., I_{a+(k+1)\cdot C+c-1}\}$ may use the entire interval I_x apart from an interval of length $\epsilon \cdot I_x$ at the end of I_x and may schedule only small jobs during I_x.*

The objective is to minimize the weighted sum of completion times.

Now each integer k induces a block of the above form.

Lemma 7. *If there is a polynomial time $(1+\epsilon)$-approximation algorithm for solving the configuration-LP for the block-problem then there is a polynomial time $(1+\epsilon)$-approximation algorithm for solving the overall configuration-LP.*

4 A Relaxation of the Configuration-LP

For each machine i denote by $\mathcal{S}'(i) \subseteq \mathcal{S}(i)$ the set of schedules obeying the restrictions defined in Section 3. Recall that this restriction costs us only a factor of $1+O(\epsilon)$ in the objective. Since we split the overall problem into disjoint blocks, it suffices to be able to solve the configuration-LP for one single block (see Lemma 7).

For solving the separation problem we relax the notion of a configuration and in particular enlarge the set of allowed configurations to a set $\mathcal{S}''(i) \supseteq \mathcal{S}'(i)$. Then, we will show that we can solve the resulting separation problem exactly. We will denote by C'-LP the configuration-LP using configurations in $\mathcal{S}''(i)$. Finally, we show that when given a solution of C'-LP, while losing only a factor of $1+\epsilon$ we can compute a solution to C-LP, i.e., which uses only configurations in $\mathcal{S}(i)$.

The first important observation is that for each interval I_x there are only constantly many possible *patterns* for the big jobs. A pattern P for big jobs is a set of $O(\epsilon)$ integers which defines the start and end times of the big jobs which are executed during I_x. Note that such a job might start before I_x and/or end after I_x.

Proposition 2. *For each interval I_x there are only $N \in O_\epsilon(1)$ many possible patterns. There are only $N^{O_\epsilon(\log n)} \in O(poly(n))$ possible combinations for all patterns in one block together.*

Note that a pattern for an interval alone does not define what exact job is executed during I_x, it describes only the start and end times of the big jobs.

Now we define the relaxed set of configurations $\mathcal{S}''(i)$. It contains each fractional job assignment which can be obtained with the following procedure. Fix a pattern P_x for each interval I_x in the block and denote by $\mathcal{P}$ the overall (global) pattern. Denote by $Q(\mathcal{P})$ the set of slots for big jobs which are given by $\mathcal{P}$. For each interval I_t denote by $rem(t)$ the remaining idle time for small jobs in each interval I_t. We allow any fractional assignment of jobs to slots and the idle time in the intervals which

- assigns at most one fractional unit of each job,
- assigns at most one fractional unit of big jobs to each slot,
- assigns small jobs fractionally to the idle time of each interval I_t, while not exceeding $rem(t)$.

Formally, we allow any feasible solution to the following linear program (the term $size(s)$ denotes the length of the slot s and $begin(s)$ denotes its start time).

$$\sum_t x_{t,j} + \sum_{s \in Q(\mathcal{P})} x_{s,j} \leq 1 \qquad \forall j \in J \tag{2}$$

$$\sum_{j \in J} x_{s,j} \leq 1 \qquad \forall s \in Q(\mathcal{P}) \tag{3}$$

$$\sum_{j \in J} p_j \cdot x_{t,j} \leq rem(t) \qquad \forall t \tag{4}$$

$$x_{t,j} \geq 0 \qquad \forall t \, \forall j \in J : r_j \leq R_t \wedge p_j \leq \epsilon \cdot I_t$$

$$x_{s,j} \geq 0 \qquad \forall s \in Q(\mathcal{P}), \forall j \in J : p_j \leq size(s) \wedge r_j \leq begin(s). \tag{5}$$

Note that we introduce a variable $x_{t,j}$ only if j is available at time R_t and j is small during I_t. Similarly, we introduce a variable $x_{s,j}$ only if j fits into s and is available in the interval where s starts.

So for each machine i, each global pattern $\mathcal{P}$ and each fractional solution to the above LP we introduce a configuration in $\mathcal{S}''(i)$ (formally, there is an infinite number of feasible solutions to the above LP but we care only about basic solutions or vertices it the corresponding polyhedron). We denote by C'-LP the configuration-LP we obtain by taking C-LP as defined in Section 2 but allowing the configurations in $\mathcal{S}''(i)$, rather than the configurations in $\mathcal{S}(i)$. For a configuration in $S \in \mathcal{S}''(i)$ we define its weight $W_{i,S} := \sum_{j \in J} x_{t,j} \cdot R_{t+1} + \sum_{j \in J} \sum_{s \in Q(\mathcal{P})} x_{s,j} \cdot end(s)$ where for each slot s we denote by $end(s)$ the finishing time of the interval in which slot s ends.

Lemma 8. *The optimal solution of C'-LP is by at most a factor of* $1 + O(\epsilon)$ *larger than the optimal solution of C-LP.*

Proof. Restricting to configurations in $\mathcal{S}'(i)$ (rather than allowing all configurations in $\mathcal{S}(i)$) loses at most a factor of $1 + O(\epsilon)$ in the objective. As $\mathcal{S}'(i) \subseteq \mathcal{S}''(i)$

allowing all configuations in $\mathcal{S}''(i)$ rather than only the ones in $\mathcal{S}'(i)$ does not lose anything in the objective. □

The benefit of allowing all configurations in $\mathcal{S}''(i)$ (rather than only configurations in $\mathcal{S}'(i)$) is that we can solve the separation problem of the dual exactly. When separating the dual, we need to either find a schedule $S \in \mathcal{S}''(i)$ such that $-\alpha_i + \sum_{j \in S} \beta_j (\sum_t x_{t,j} + \sum_{s \in Q(\mathcal{P})} x_{s,j}) > W_{i,S}$ or asserts that no such schedule exists. For each machine i we do the following: we enumerate all patterns for the big jobs. For each pattern $\mathcal{P}$, we find the configuration $S \in \mathcal{S}''(i)$ which follows $\mathcal{P}$ and which optimizes $\sum_{j \in S} \sum \beta_j (\sum_t x_{t,j} + \sum_{s \in Q(\mathcal{P})} x_{s,j}) - W_{i,S}$. Formally, for each machine i and each pattern $\mathcal{P}$ we solve the above LP with the linear objective function

$$\max \sum_{j \in J} \beta_j (\sum_t x_{t,j} + \sum_{s \in Q(\mathcal{P})} x_{s,j}) - \sum_{j \in J} x_{t,j} \cdot R_{t+1} - \sum_{j \in J} \sum_{s \in Q(\mathcal{P})} x_{s,j} \cdot end(s).$$

We call the overall linear program the *Slot-LP*. We remark that a similar LP has recently been used in [4].

Lemma 9. *For each $\epsilon > 0$ there is a polynomial time algorithm which solves the separation problem of C'-LP exactly.*

5 Feasible Solution to the Original Configuration-LP

Since we can solve the dual of C'-LP exactly in polynomial time (using the ellipsoid method together with our separation oracle), we can also compute in polynomial time an optimal solution of C'-LP itself. It remains to show that any solution to C'-LP can be transformed to a feasible solution to C-LP (i.e., using only configurations in $S(i)$ for each machine i) while losing at most a factor of $1+\epsilon$. To achieve this we show that by taking each configuration $S \in \mathcal{S}''(i)$ arising in the computed solution for C'-LP and replacing it by a set of configurations in $\mathcal{S}(i)$, each with a suitable coefficient $y_{i,S}$. We choose these configurations and coefficients such that each job is still assigned to the same extent as in S and the total cost increases at most by a factor of $1 + \epsilon$.

The main step is to prove the following lemma.

Lemma 10. *Let $S \in \mathcal{S}''(i)$ be a configuration, defined by a solution x to the Slot-LP. In polynomial time we can compute a set of configurations $S_1, ..., S_B$ and coefficients $\lambda_1, ..., \lambda_B$ such that for each job j we have $\sum_{\ell : j \in \mathcal{S}_\ell} \lambda_\ell = \sum_t x_{t,j} + \sum_s x_{s,j}$ and $\sum_\ell \lambda_\ell \cdot W_{i,S_\ell} \leq (1+\epsilon) \cdot W_{i,S}$ and $\sum_\ell \lambda_\ell = 1$.*

Consider a machine i and a configuration $S \in \mathcal{S}''(i)$, described by a global pattern $\mathcal{P}$ and a vector x for the Slot-LP. We interpret S as a fractional matching in a bipartite graph. Here we borrow some ideas from [12]. For each job j arising in S we introduce a vertex v_j. For each slot s for a big job we introduce a vertex w_s. If in S some job j is (fractionally) assigned to s then we add the edge (v_j, w_s) with weight $w_j \cdot end(s)$ and assign $x_{s,j}$ units of j to w_s. For each interval I_t with

idle time for small jobs, we introduce $k_t := \left\lceil \sum_j x_{t,j} \right\rceil$ vertices $w_{t,1}, ..., w_{t,k_t}$, representing this idle time. For defining the edges of the vertices $w_{t,\ell}$ we do the following procedure: assume that the jobs (fractionally) assigned to I_t as small jobs are labeled $\{1, ..., n_t\}$ and they are ordered by non-increasingly by processing times. We iterate over the jobs in this order. While doing this we maintain the invariant that there is some value ℓ such that all vertices $w_{t,1}, ..., w_{t,\ell-1}$ have one (fractional) unit of jobs assigned to it, vertex $w_{t,\ell}$ has α units of jobs assigned to it for some value $\alpha \in [0, 1)$, and the vertices $w_{t,\ell+1}, ..., w_{t,k_t}$ have no job assigned to them. Consider a job j. If $x_{t,j} \leq 1 - \alpha$ then we assign j completely to $w_{t,\ell}$ and introduce an edge $(v_j, w_{t,\ell})$ with weight $w_j \cdot R_{t+1}$. If $x_{t,j} > 1 - \alpha$ then we assign α units of j to $w_{t,\ell}$ and the remaining $x_{t,j} - \alpha$ units to the next vertex $w_{t,\ell+1}$. In that case we introduce edges $(v_j, w_{t,\ell})$ and $(v_j, w_{t,\ell+1})$ with weights $w_j \cdot R_{t+1}$ and $w_j \cdot R_{t+2}$, respectively. Denote by G the resulting graph.

Lemma 11. *For any integral matching M in G there is a schedule $S' \in \mathcal{S}(i)$ whose cost is at most by a factor $1 + \epsilon$ larger than the weight of M. Given M, the corresponding schedule S' can be computed in polynomial time.*

Using the above lemma, we can compute a convex combination of schedules in $\mathcal{S}(i)$ whose total cost is not much bigger than the cost of S and which assigns (fractionally) the same jobs to the same extent.

Proof (of Lemma 10). From matching theory we know that the bipartite matching polytope is integral. This implies that the fractional assignment induced by S can be written as a convex combination of integral matchings $M_1, ..., M_B$, each having a suitable coefficient λ_ℓ, see e.g., [14]. This representation can be computed in polynomial time. For each matching M_ℓ we define $S_\ell \in \mathcal{S}(i)$ to be the resulting schedule as given by Lemma 11. Then the schedules S_ℓ and the coefficients λ_ℓ have the properties claimed in the lemma. □

Using Lemma 10 we compute a solution $\bar{y}$ for C-LP, given we have a solution y to C'-LP. Initially, we set $\bar{y} := 0$. Consider a machine i. For each $S \in \mathcal{S}''(i)$ with $y_{i,S} > 0$, we compute the convex combination $S_1, ..., S_B$ with coefficients $\lambda_1, ..., \lambda_B$ according to Lemma 10. Then for each $\ell \in \{1, ..., B\}$ we increase the variable $\bar{y}_{i,S_\ell}$ by $\lambda_\ell \cdot y_{i,S}$. Hence, we obtain our main theorem.

Theorem 1. *For any $\epsilon > 0$ there is a polynomial time algorithm which computes a $(1+\epsilon)$-approximation of the configuration-LP for scheduling jobs on unrelated machines to minimize the weighed sum of completion time.*

Using these techniques, we obtain a PTAS for the scheduling problem with rejection on one machine. The details are in the full version of this paper.

Theorem 2. *There is a polynomial time approximation scheme for the problem $1|r_j| \sum_S w_j C_j + \sum_{\bar{S}} e_j$.*

For the setting where all jobs have equal release dates there is an optimal pseudopolynomial time algorithm and an FPTAS for the corresponding problem of scheduling with rejection [5]. This can be turned into an FPTAS for solving the configuration-LP in that setting.

Theorem 3. *For any $\epsilon > 0$ there is an algorithm with running time $O(poly(n, \frac{1}{\epsilon}))$ which computes a $(1+\epsilon)$-approximative solution to C-LP if all jobs are released at time $t = 0$.*

References

1. Afrati, F., Bampis, E., Chekuri, C., Karger, D., Kenyon, C., Khanna, S., Milis, I., Queyranne, M., Skutella, M., Stein, C., Sviridenko, M.: Approximation Schemes for Minimizing Average Weighted Completion Time with Release Dates. In: Proceedings of Symposium on Foundations of Computer Science, FOCS 1999, pp. 32–43 (1999)
2. Asadpour, A., Feige, U., Saberi, A.: Santa claus meets hypergraph matchings. ACM Transactions on Algorithms 8(3), 1–24 (2012)
3. Bansal, N., Sviridenko, M.: The Santa Claus problem. In: STOC 2006, pp. 31–40 (2006)
4. Bonifaci, V., Wiese, A.: Scheduling unrelated machines of few diferent types. CoRR abs/1205.0974 (2012)
5. Engels, D., Karger, D., Kolliopoulos, S., Sengupta, S., Uma, R., Wein, J.: Techniques for scheduling with rejection. J. of Algorithms 49(1), 175–191 (2003)
6. Graham, R., Lawler, E., Lenstra, J., Rinnooy Kan, A.H.G.: Optimization and approximation in deterministic sequencing and scheduling: a survey. Annals of Discrete Mathematics 5, 287–326 (1979)
7. Grigoriadis, M.D., Khachiyan, L.G., Porkolab, L., Villavicencio, J.: Approximate max-min resource sharing for structured concave optimization. SIAM Journal on Optimization 11, 1081–1091 (2001)
8. Grötschel, M., Lovász, L., Schrijver, A.: Geometric algorithms and combinatorial optimization. Springer, Berlin (1988)
9. Gunluk, O., Kimbrel, T., Ladanyi, L., Schieber, B., Sorkin, G.: Vehicle Routing and Staffing for Sedan Service. Transportation Science 40, 313–326 (2006)
10. Haeupler, B., Saha, B., Srinivasan, A.: New Constructive Aspects of the Lovász Local Lemma. J. ACM 58(6), 1–28 (2011)
11. Hoogeveen, H., Skutella, M., Woeginger, G.: Preemptive scheduling with rejection. Mathematical Programming 94(2-3), 361–374 (2003)
12. Lenstra, J., Shmoys, D., Tardos, E.: Approximation Algorithms for Scheduling Unrelated Parallel Machines. Mathematical Programming 46, 259–271 (1990)
13. Plotkin, S.A., Shmoys, D., Tardos, E.: Fast approximation algorithms for fractional packing and covering problems. Math. of Oper. Res. 20, 257–301 (1995)
14. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency. Springer (2003)
15. Schulz, A., Skutella, M.: Scheduling Unrelated Machines by Randomized Rounding. SIAM J. Discrete Math. 15(4), 450–469 (2002)
16. Seiden, S.: Preemptive multiprocessor scheduling with rejection. Theoretical Computer Science 262(1), 437–458 (2001)
17. Skutella, M.: Convex quadratic and semidefinite programming relaxations in scheduling. J. ACM 48(2), 206–242 (2001)
18. Smith, W.E.: Various optimizers for single-stage production. Naval Research and Logistics Quarterly 3, 59–66 (1956)
19. Svensson, O.: Santa Claus schedules jobs on unrelated machines. In: Proceedings of STOC 2011, pp. 617–626 (2011)

Author Index