LNCS 6080

Friedrich Eisenbrand
F. Bruce Shepherd (Eds.)

Integer Programming and Combinatorial Optimization

14th International Conference, IPCO 2010
Lausanne, Switzerland, June 2010
Proceedings

IPCO 2010
LAUSANNE

Springer

Friedrich Eisenbrand   F. Bruce Shepherd (Eds.)

Integer Programming and Combinatorial Optimization

14th International Conference, IPCO 2010
Lausanne, Switzerland, June 9-11, 2010
Proceedings

Springer

Volume Editors

Friedrich Eisenbrand
École Polytechnique Féderale de Lausanne
Institute of Mathematics
1015 Lausanne, Switzerland
E-mail: friedrich.eisenbrand@epfl.ch

F. Bruce Shepherd
McGill University
Department of Mathematics and Statistics
805 Sherbrooke West, Montreal, Quebec, H3A 2K6, Canada
E-mail: bruce.shepherd@mcgill.ca

Library of Congress Control Number: 2010926408

CR Subject Classification (1998): F.2, E.1, I.3.5, G.2, G.1.6, F.2.2

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

ISSN	0302-9743
ISBN-10	3-642-13035-6 Springer Berlin Heidelberg New York
ISBN-13	978-3-642-13035-9 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

The idea of a refereed conference for the mathematical programming community was proposed by Ravi Kannan and William Pulleyblank to the Mathematical Programming Society (MPS) in the late 1980s. Thus IPCO was born, and MPS has sponsored the conference as one of its main events since IPCO I at the University of Waterloo in 1990. The conference has become the main forum for recent results in Integer Programming and Combinatorial Optimization in the non-Symposium years.

This volume compiles the papers presented at IPCO XIV held June 9-11, 2010, at EPFL in Lausanne. The scope of papers considered for IPCO XIV is likely broader than at IPCO I. This is sometimes due to the wealth of new questions and directions brought from related areas. It can also be due to the successful application of "math programming" techniques to models not traditionally considered. In any case, the interest in IPCO is greater than ever and this is reflected in both the number (135) and quality of the submissions. The Programme Committee with 13 members was also IPCO's largest. We thank the members of the committee, as well as their sub-reviewers, for their exceptional (and time-consuming) work and especially during the online committee meeting held over January. The process resulted in the selection of 34 excellent research papers which were presented in non-parallel sessions over three days in Lausanne. Unavoidably, this has meant that many excellent submissions were not able to be included. As is typical, we would expect to see full versions of many of the IPCO papers in scientific journals in the not too distant future. Finally, a sincere thanks to all authors who submitted their current research to IPCO. It is this support that determines the excellence of the conference.

March 2010

Friedrich Eisenbrand
Bruce Shepherd

Conference Organization

Programme Committee

Alper Atamtürk	UC Berkeley
David Avis	McGill
Friedrich Eisenbrand	EPFL
Marcos Goycoolea	Adolfo Ibañez
Oktay Günlük	IBM
Satoru Iwata	Kyoto
Tamás Király	Eötvös Budapest
François Margot	CMU
Bruce Shepherd (Chair)	McGill
Levent Tunçel	Waterloo
Santosh Vempala	Georgia Tech
Peter Winkler	Dartmouth
Neal E. Young	UC Riverside

Local Organization

Michel Bierlaire
Jocelyne Blanc
Friedrich Eisenbrand (Chair)
Thomas Liebling
Martin Niemeier
Thomas Rothvoß
Laura Sanità

External Reviewers

Tobias Achterberg
Ernst Althaus
Reid Andersen
Matthew Andrews
Elliot Anshelevich
Gary Au
Mourad Baiou
Nina Balcan
Nikhil Bansal
Andre Berger
Attila Bernáth
Dan Bienstock
John Birge
Jaroslaw Byrka
Alberto Caprara
Deeparnab Chakrabarty
Chandra Chekuri
Kevin Cheung
Marek Chrobak
Jose Coelho de Pina
Michelangelo Conforti
Miguel Constantino
Jose Correa
Sanjeeb Dash

Santanu Dey
David Eppstein
Daniel Espinoza
Guy Even
Uriel Feige
Zsolt Fekete
Christina Fernandes
Carlo Filippi
Samuel Fiorini
Nathan Fisher
Lisa Fleischer
Keith Frikken
Tetsuya Fujie
Toshihiro Fujito
Ricardo Fukasawa
Joao Gouveia
Marcos Goycoolea
Fabrizio Grandoni
Betrand Guenin
Dave Hartvigsen
Christoph Helmberg
Hiroshi Hirai
Dorit Hochbaum
Chien-Chung Huang
Cor Hurkens
Sungjin Im
Nicole Immorlica
Toshimasa Ishii
Takehiro Ito
Garud Iyengar
Kamal Jain
Klaus Jansen
David Johnson
Tibor Jordan
Vincent Jost
Alpár Jüttmer
Satyen Kale
George Karakostas
Anna Karlin
Sean Kennedy
Rohit Khandekar
Sanjeev Khanna
Samir Khuller
Shuji Kijima
Zoltan Kiraly
Tamas Kis
Robert Kleinberg
Yusuke Kobayashi
Jochen Könemann
Lingchen Kong
Nitish Korula
Christos Koufogiannakis
Erika Kovacs
Marek Krcal
Sven Krumke
Simge Kucukyavuz
Lap Chi Lau
Monique Laurent
Adam Letchford
Asaf Levin
Sven Leyffer
Christian Liebchen
Jeff Linderoth
Quentin Louveaux
James Luedtke
Avner Magen
Dániel Marx
Monaldo Mastrolilli
Kurt Mehlhorn
Zoltan Miklos
Hiroyoshi Miwa
Atefeh Mohajeri
Eduardo Moreno
Yiannis Mourtos
Kiyohito Nagano
Arkadi Nemirovski
Martin Niemeier
Neil Olver
Gianpaolo Oriolo
Gyula Pap
Julia Pap
Gabor Pataki
Sebastian Pokutta
Imre Polik
David Pritchard
Kirk Pruhs
Linxia Qin
Maurice Queyranne
R Ravi
Gerhard Reinelt

Table of Contents

Solving LP Relaxations of Large-Scale Precedence Constrained Problems

Daniel Bienstock[1] and Mark Zuckerberg[2]

[1] APAM and IEOR Depts., Columbia University
[2] Resource and Business Optimization Group Function, BHP Billiton Ltd.

Abstract. We describe new algorithms for solving linear programming relaxations of very large precedence constrained production scheduling problems. We present theory that motivates a new set of algorithmic ideas that can be employed on a wide range of problems; on data sets arising in the mining industry our algorithms prove effective on problems with many millions of variables and constraints, obtaining provably optimal solutions in a few minutes of computation[1].

1 Introduction

We consider problems involving the scheduling of jobs over several periods subject to precedence constraints among the jobs as well as side-constraints. We must choose the subset of jobs to be performed, and, for each of these jobs, how to perform it, choosing from among a given set of options (representing facilities or modes of operation). Finally, there are side-constraints to be satisfied, including period-wise, per-facility processing capacity constraints, among others. There are standard representations of these problems as (mixed) integer programs.

Our data sets originate in the mining industry, where problems typically have a small number of side constraints - often well under one hundred – but may contain millions of jobs and tens of millions of precedences, as well as spanning multiple planning periods. Appropriate formulations often achieve small integrality gap in practice; unfortunately, the linear programming relaxations are far beyond the practical reach of commercial software.

We present a new iterative algorithm for solving the LP relaxation of this problem. The algorithm incorporates, at a low level, ideas from Lagrangian relaxation and column generation, but is however based on fundamental observations on the underlying combinatorial structure of precedence constrained, capacitated optimization problems. Rather than updating dual information, the algorithm uses primal structure gleaned from the solution of subproblems in order to accelerate convergence. The general version of our ideas should be applicable to a wide class of problems. The algorithm can be proved to converge to optimality; in practice we have found that even for problems with millions of variables

[1] The first author was partially funded by a gift from BHP Billiton Ltd., and ONR Award N000140910327.

and tens of millions of constraints, convergence to *proved* optimality is usually obtained in under twenty iterations, with each iteration requiring only a few seconds on current computer hardware.

2 Definitions and Preliminaries

2.1 The Precedence Constrained Production Scheduling Problem

Definition 1. *We are given a directed graph $G = (\mathcal{N}, \mathcal{A})$, where the elements of $\mathcal{N}$ represent jobs, and the arcs $\mathcal{A}$ represent precedence relationships among the jobs: for each $(i, j) \in \mathcal{A}$, j can be performed no later than job i. Denote by F, the number of facilities, and T, the number of scheduling periods.*

Let $y_{j,t} \in \{0, 1\}$ represent the choice to process job j in period t, and $x_{j,t,f} \in [0, 1]$ represent the proportion of job j performed in period t, and processed according to processing option, or "facility", f.

Let $c^T x$ be an objective function, and let $Dx \leq d$ be a collection of arbitrary "side" constraints.

The linear programming relaxation of the resulting problem, which we will refer to as PCPSP, is as follows:

$$\textbf{(PCPSP):} \qquad \max \quad c^T x \tag{1}$$

$$\textit{Subject to:} \quad \sum_{\tau=1}^{t} y_{i,\tau} \leq \sum_{\tau=1}^{t} y_{j,\tau}, \ \forall (i,j) \in \mathcal{A}, \ 1 \leq t \leq T \tag{2}$$

$$Dx \leq d \tag{3}$$

$$y_{j,t} = \sum_{f=1}^{F} x_{j,t,f}, \ \forall j \in \mathcal{N}, \ 1 \leq t \leq T \tag{4}$$

$$\sum_{t=1}^{T} y_{j,t} \leq 1, \ \forall j \in \mathcal{N} \tag{5}$$

$$x \geq 0. \tag{6}$$

For precedence constrained production scheduling problems that occur in the mining industry some typical numbers are as follows:

- 1 million – 10 million jobs, and 1 million – 100 million precedences,
- 20 – 200 side-constraints, 10 – 20 periods, and 2 – 3 facilities.

These numbers indicate that the number of constraints of the form (2), (4) and (5) can be expected to be very large.

2.2 Background

The Open Pit Mine Scheduling Problem. The practical motivating problem behind our study is the open pit mine scheduling problem. We are given a three-dimensional region representing a mine to be exploited; this region is divided into "blocks" (jobs, from a scheduling perspective) corresponding to units of earth ("cubes") that can be extracted in one step. In order for a block to be extracted, the set of blocks located (broadly speaking) in a cone above it must be extracted first. This gives rise to a set of precedences, i.e. to a directed graph whose vertices are the blocks, and whose arcs represent the precedences. Finally, the extraction of a block entails a certain (net) profit or cost.

The problem of selecting which blocks to extract so as to maximize profit can be stated as follows:

$$\max \left\{ c^T x \, : \, x_i \leq x_j \; \forall \, (i,j) \in \mathcal{A}, \; x_j \in \{0,1\} \; \forall j \right\},$$

where as before $\mathcal{A}$ indicates the set of precedences. This is the so-called *maximum weight closure problem* – in a directed graph, a closure is a set S of vertices such that there exist no arcs (i,j) with $i \in S$ and $j \notin S$. It can be solved as a minimum $s-t$ cut problem in a related graph of roughly the same size. See [P76], and also [J68], [Bal70] and [R70]. Further discussion can be found in [HC00], where the authors note (at the end of Section 3.4) that it can be shown by reduction from max clique that adding a single cardinality constraint to a max closure problem is enough to make it NP-hard. For additional related material see [F06], [LG65], [CH03], and references therein.

The problem we are concerned with here, by contrast, also incorporates production scheduling. When a block is extracted it will be processed at one of several facilities with different operating capabilities. The processing of a given block i at a given facility f consumes a certain amount of processing capacity v_{if} and generates a certain net profit p_{if}. This overall planning problem spans several time periods; in each period we will have one or more knapsack (capacity) constraints for each facility. We usually will also have additional, ad-hoc, non-knapsack constraints. In this version the precedence constraints apply across periods as per (2): if $(i,j) \in \mathcal{A}$ then j can only be extracted in the same or in a later period than i.

Typically, we need to produce schedules spanning 10 to 20 periods. Additionally, we may have tens of thousands (or many more) blocks; this can easily make for an optimization problem with millions of variables and tens of millions of precedence constraints, but with (say) on the order of one hundred or fewer processing capacity constraints (since the total number of processing facilities is typically small).

Previous work. A great deal of research has been directed toward algorithms for the maximum weight closure problems, starting with [LG65] and culminating in the very efficient method described in [H08] (also see [CH09]). A "nested shells" heuristic for the capacitated, multiperiod problem, based on the work in [LG65], is applicable to problems with a single capacity constraint, among other simplifications. As commercial integer programming software has improved, mine scheduling software packages have recently emerged that aggregate blocks in order to yield a mixed integer program of tractable size. The required degree of aggregation can however be enormous; this can can severely compromise the validity and the usefulness of the solution. For an overview of other heuristic approaches that have appeared in the open mine planning literature [HC00] and [F06].

Recently (and independent of our work) there has been some new work relevant to the solution of the LP relaxation of the open pit mine scheduling problem. [BDFG09][2] have suggested a new approach in which blocks are aggregated only with respect to the digging decisions but not with respect to the processing decisions, i.e. all original blocks in an aggregate must be extracted in a common period, but the individual blocks comprising an aggregate can be processed in different ways. This problem is referred to by the authors as the "Optimal Binning Problem". As long as there is more than one processing option this approach still maintains variables for each block and period and is therefore still very large, but the authors propose an algorithm for the LP relaxation of this problem that is only required to solve a sequence of linear programs with a number of variables on the order of the number of aggregates (times the number of periods) in order to come to a solution of the large LP. Thus if the number of aggregates is small the LP can be solved quickly.

Another development that has come to our attention recently is an algorithm by [CEGMR09] which can solve the LP relaxation of even very large instances of the open pit mine scheduling problem very efficiently. This algorithm is only applicable however to problems for which there is a single processing option and for which the only constraints are knapsacks and there is a single such constraint in each scheduling period. The authors note however that more general problems can be relaxed to have this form in order to yield an upper bound on the solution value.

From a broad perspective, the method we give below uses dual information in order to effectively reduce the size of the linear program; in this sense our work is similar to that in [BDFG09]. In the full version of this paper we describe what our algorithm would look like when applied to the aggregated problem treated by [BDFG09], which is a special case of ours. The relationship between the max closure problem and the LP is a theme in common with the work of [CEGMR09].

3 Our Results

Empirically, it can be observed that formulation (1-6) frequently has small integrality gap. We present a new algorithm for solving the continuous relaxation of this formulation and generalizations. Our algorithm is applicable to problems with an arbitrary number of process options and arbitrary side constraints, and it requires no aggregation. On very large, real-world instances our algorithm proves very efficient.

[2] We interacted with [BDFG09] as part of an industrial partnership, but our work was performed independently.

Our algorithmic developments hinge on three ideas. In order to describe these ideas, we will first recast PCPSP as a special case of a more general problem, to which these results (and our solution techniques) apply.

Definition 2. *Given a directed graph $G = (\mathcal{N}, \mathcal{A})$ with n vertices, and a system $Dx \leq d$ of d constraints on n variables, the General Precedence Constrained Problem is the following linear program:*

$$\textbf{(GPCP):} \qquad \max \quad c^T x \tag{7}$$

$$Dx \leq d \tag{8}$$

$$x_i - x_j \leq 0, \quad \forall (i,j) \in \mathcal{A}, \tag{9}$$

$$0 \leq x_j \leq 1, \quad \forall j \in \mathcal{N}. \tag{10}$$

This problem is more general than PCPSP:

Lemma 1. *Any instance of PCPSP can be reduced to an equivalent instance of GPCP with the same number of variables and of constraints.*

Proof. Consider an instance of PCPSP on $G = (\mathcal{N}, \mathcal{A})$, with T time periods, F facilities and side constraints $Dx \leq d$. Note that the y variables can be eliminated. Consider the following system of inequalities on variables $z_{j,t,f}$ ($j \in \mathcal{N}$, $1 \leq t \leq T$, $1 \leq f \leq F$):

$$z_{j,t,f} - z_{j,t,f+1} \leq 0, \quad \forall j \in \mathcal{N},\ 1 \leq t \leq T,\ 1 \leq f < F, \tag{11}$$

$$z_{j,t,F} - z_{j,t+1,1} \leq 0, \quad \forall j \in \mathcal{N},\ 1 \leq t < T, \tag{12}$$

$$z_{j,T,F} \leq 1, \quad j \in \mathcal{N}, \tag{13}$$

$$z_{i,t,F} - z_{j,t,F} \leq 0, \quad \forall (i,j) \in \mathcal{A},\ 1 \leq t \leq T, \tag{14}$$

$$z \geq 0. \tag{15}$$

Given a solution (x, y) to PCPSP, we obtain a solution z to (11)-(15) by setting, for all j, t and f:

$$z_{j,t,f} = \sum_{\tau=1}^{t-1} \sum_{f'=1}^{F} x_{j,\tau,f'} + \sum_{f'=1}^{f} x_{j,t,f'},$$

and conversely. Thus, for an appropriate system $\bar{D}z \leq \bar{d}$ (with the same number of rows as $Dx \leq d$) and objective $\bar{c}^T z$, PCPSP is equivalent to the linear program:

$$\min\{\bar{c}^T z \ : \ \bar{D}z \leq \bar{d}, \text{ and constraints (11)-(15)}\}. \qquad \blacksquare$$

Note: In Lemma 1 the number of precedences in the instance of *GPCP* is larger than in the original instance of PCPSP; nevertheless we stress that the number of constraints (and variables) is indeed the same in both instances.

We will now describe ideas that apply to *GPCP*. First, we have the following remark.

Observation 1. *Consider an instance of problem GPCP, and let $\pi \geq 0$ be a given vector of dual variables for the side-constraints (8). Then the Lagrangian obtained by dualizing (8) using π,*

$$\max \quad c^T x + \pi^T (d - Dx) \tag{16}$$

$$\textit{Subject to: } x_i - x_j \leq 0, \ \forall \ (i,j) \in \mathcal{A} \tag{17}$$

$$0 \leq x_j \leq 1, \quad \forall j \in \mathcal{N}. \tag{18}$$

is a maximum closure problem with $|\mathcal{A}|$ precedences.

Note: There is a stronger version of Observation 1 in the specific case of problem PCPSP; namely, the x variables can be *eliminated* from the Lagrangian (details: full paper, also see [BZ09]).

Observation 1 suggests that a Lagrangian relaxation algorithm for solving problem *GPCP* – that is to say, an algorithm that iterates by solving problems of the form (16-18) for various vectors π – would enjoy fast individual iterations. This is correct, as our experiments confirm that even extremely large max closure instances can be solved quite fast using the appropriate algorithm (details, below). However, in our experiments we also observed that traditional Lagrangian relaxation methods (such as subgradient optimization), applied to *GPCP*, performed quite poorly, requiring vast numbers of iterations and not quite converging to solutions with desirable accuracy.

Our approach, instead, relies on leveraging combinatorial structure that optimal solutions to *GPCP* must satisfy. Lemmas 2 and 3 are critical in suggesting such structure.

Lemma 2. *Let $P = \{x \in R^n \,:\, Ax \leq b, \ Dx \leq d\}$, where A, D, b, d are matrices and vectors of appropriate dimensions. Let $\hat{x}$ be an extreme point of P. Let $\bar{A}x = \bar{b}, \ \bar{D}x = \bar{d}$ be the set of binding constraints at $\hat{x}$. Assume $\bar{D}$ has q linearly independent rows, and let $N^{\hat{x}}$ be the null space of $\bar{A}$. Then $dim(N^{\hat{x}}) \leq q$.*

Proof: $\bar{A}$ must have at least $n - q$ linearly independent rows and thus its null space must have dimension $\leq q$. ■

Lemma 3. *Let P be the feasible space of a GPCP with q side constraints. Denote by $Ax \leq b$ the subset of constraints containing the precedence constraints and the constraints $0 \leq x \leq 1$, and let $Dx \leq d$ denote the side constraints. Let $\hat{x}$ be an extreme point of P, and the entries of $\hat{x}$ attain k distinct fractional values $\{\alpha_1, \ldots, \alpha_k\}$. For $1 \leq r \leq k$, let $\theta^r \in \{0,1\}^n$ be defined by:*

$$\textit{for} \ \ 1 \leq j \leq n, \quad \theta^r_j \ = \begin{cases} 1, \textit{ if } \ \hat{x}_j = \alpha_r, \\ 0, \textit{ otherwise.} \end{cases}$$

Let $\bar{A}$ be the submatrix of A containing the binding constraints at $\hat{x}$. Then the vectors θ^r are linearly independent and belong to the null space of $\bar{A}$. As a consequence, $k \leq q$.

Proof: First we prove that $\bar{A}\theta^r = 0$. Given a precedence constraint $x_i - x_j \leq 0$, if the constraint is binding then $\hat{x}_i = \hat{x}_j$. Thus if $\hat{x}_i = \alpha_r$, so that $\theta^r_i = 1$, then

$\hat{x}_j = \alpha_r$ also, and so $\theta^r_j = 1$ as well, and so $\theta^r_i - \theta^r_j = 0$. By the same token if $\hat{x}_i \neq \alpha_r$ then $\hat{x}_j \neq \alpha_r$ and again $\theta^r_i - \theta^r_j = 0$. If a constraint $x_i \geq 0$ or $x_i \leq 1$ is binding at $\hat{x}$ then naturally $\theta^r_i = 0$ for all r as $\hat{x}_i$ is not fractional. The supports of the θ^r vectors are disjoint, yielding linear independence. Finally, $k \leq q$ follows from Lemma 2. ■

Observation 1 implies that an optimal solution x^* to an instance of *GPCP* can be written as a weighted sum of incidence vectors of closures, i.e.,

$$x^* = \sum_{q=1}^{Q} \mu_q v^q, \tag{19}$$

where $\mu \geq 0$, and, for each q, $v^q \in \{0,1\}^n$ is the incidence vector of a closure $S^q \subset \mathcal{N}$. [In fact, the S^q can be assumed to be nested]. So for any $i, j \in \mathcal{N}$, $x^*_j = x^*_i$ if i and j belong to precisely the same family of sets S^q. Also, Lemma 3 states that the number of distinct values that x^*_j can take is *small*, if the number of side constraints is small. Therefore it can be shown that when the number of side constraints is small the number of closures (terms) in (19) must also be small. In the full paper we show that a rich relationship exists between the max closures produced by Lagrangian problems and the optimal dual and primal solutions to *GPCP*. Next, we will develop an algorithm that solves *GPCP* by attempting to "guess" the correct representation (19).

First, we present a result that partially generalizes Lemma 3.

Theorem 2. *Let P, A, $\bar{A}$, D, q, $\hat{x}$ and $N^{\hat{x}}$ be as in Lemma 2, and assume additionally that A is totally unimodular and that b is integral. Define*

$$I^{\hat{x}} = \{y \in R^n : y_i = 0,\ \forall i \text{ s.t. } \hat{x}_i \text{ is integer}\}. \tag{20}$$

Then there exists an integral vector $x^i \in R^n$, and vectors $\theta^h \in R^n$, $1 \leq h \leq q$, such that:

(a) $Ax^i \leq b$,
(b) $\bar{A}x^i = \bar{b}$,
(c) $x^i_j = \hat{x}_j$, $\forall j$ *s.t.* $\hat{x}_j$ *is integer,*
(d) $\hat{x} = x^i + \sum_{r=1}^{q} \alpha_r \theta^r$, *for some* $\alpha \in R^q$,
(e) The set $\{\theta^1, \ldots, \theta^q\}$ *spans* $N^{\hat{x}} \cap I^{\hat{x}}$,
(f) $|\theta^h_j| \leq \operatorname{rank}(\bar{A})$, *for all* $1 \leq h \leq q$ *and* $1 \leq j \leq n$,

In the special case of the GPCP, we can choose x^i satisfying the additional condition:

(g) $x^i_j = 0$, *for all j such that $\hat{x}_j$ is fractional.*

Proof sketch: Let us refer to the integer coordinates of x as x_I and to the corresponding columns of A as A^I, and to the fractional coordinates of x as x_F, and to the corresponding columns of A as A^F. Let h be the number of columns in A^F. Note that $b - A^I x_I$ is integer, and so by total unimodularity there exists integer $y \in R^h$ satisfying $A^F y \leq b - A^I x_I$, $\bar{A}^F y = \bar{b} - \bar{A}^I x_I$. Defining now $x^i = (x_I, y)$ then x^i is integer; it is equal to x everywhere that x is integer, and it satisfies $Ax^i \leq b$ and $\bar{A}x^i = \bar{b}$. Clearly $x - x^i$ belongs to I^x, and moreover $\bar{A}(x - x^i) = 0$ so that it belongs to N^x as well, and so it can be decomposed as $x - x^i = \sum_{r=1}^{q} \alpha_r \theta^r$. For the special case of *GPCP* we have already described a decomposition for which x^i equals x everywhere that x is integer and is zero elsewhere. See the full paper for other details. ■

Comment: Note that rank($\bar{A}$) can be high and thus condition (d) is not quite as strong as Lemma 3; nevertheless q is small in any case and so we obtain a decomposition of $\hat{x}$ into "few" terms when the number of side-constraints is "small". Theorem 2 can be strengthened for specific families of totally unimodular matrices. For example, when A is the node-arc incidence matrix of a digraph, the θ vectors are incidence vectors of cycles, which yields the following corollary.

Corollary 1. *Let P be the feasible set for a minimum cost network flow problem with integer data and side constraints. Let $\hat{x}$ be an extreme point of P, and let q be the number of linearly independent side constraints that are binding at $\hat{x}$. Let $\zeta = \{j : \hat{x}_j \text{ integral}\}$. Then $\hat{x}$ can be decomposed into the sum of an integer vector v satisfying all network flow (but not necessarily side) constraints, and with $v_j = \hat{x}_j \; \forall j \in \zeta$, and a sum of no more than q fractional cycle flows, over a set of cycles disjoint from ζ.*

4 A General Algorithmic Template

Now we return to the generic algorithm for *GPCP* that attempts to guess the right representation of an optimal solution as a weighted sum of incidence vectors of "few" closures. To motivate our approach, we first consider a more general situation. We are given a linear program:

$$\begin{aligned} (P_1): \quad & \max \; c^T x \\ \text{s.t.} \quad & Ax \leq b \\ & Dx \leq d. \end{aligned} \tag{21}$$

Denote by $L(P_1, \mu)$ the Lagrangian relaxation in which constraints (21) are dualized with penalties μ, i.e. the problem $\max\{c^T x + \mu^T(d - Dx) : Ax \leq b\}$.

One can approach problem P_1 by means of Lagrangian relaxation, i.e. an algorithm that iterates by solving multiple problems $L(P_1, \mu)$ for different choices of μ; the multipliers μ are updated according to some procedure. A starting point for our work concerns the fact that traditional Lagrangian relaxation schemes (such as subgradient optimization) can prove frustratingly slow to achieve convergence, often requiring seemingly instance-dependent choices of algorithmic parameters. They also do not typically yield optimal feasible primal solutions; in fact frequently failing to deliver a sufficiently accurate solutions (primal or dual). However, as observed in [B02] (and also see [BA00]) Lagrangian relaxation schemes *can* discover useful "structure."

For example, Lagrangian relaxation can provide early information on which constraints from among those that were dualized are likely to be tight, and on which variables x are likely to be nonzero, even if the actual numerical values for primal or dual variables computed by the relaxation are inaccurate. The question then is how to use such structure in order to accelerate convergence and to obtain higher accuracy. In [B02] the following approach was used:

– Periodically, interrupt the Lagrangian relaxation scheme to solve a *restricted* linear program consisting of P_1 with some additional constraints used to impose the desired structure. Then use the duals for constraints (21) obtained in the solution to the restricted LP to restart the Lagrangian procedure.

The restricted linear program includes all constraints, and thus could (potentially) still be very hard – the idea is that the structure we have imposed renders the LP much easier. Further, the LP includes all constraints, and thus the solution we obtain is fully feasible for P_1, thus proving a lower bound. Moreover, if our guess as to "structure" is correct, we also obtain a high-quality dual feasible vector, and our use of this vector so as to restart the Lagrangian scheme should result in accelerated convergence (as well as proving an upper bound on P_1). In [B02] these observations were experimentally verified in the context of several problem classes.

```
1. Set $\mu_0 = 0$ and set $k = 1$.

2. Solve $L(P_1, \mu_{k-1})$. Let $w^k$ be an optimal solution.
   If $k > 1$ and $H^{k-1}w^k = h^{k-1}$, STOP.

3. Let $H^k x = h^k$ be a system of equations satisfied by $w^k$.

4. Define the restricted problem:

   $(P_2^k): \quad \max \; c^T x$
   $\text{s.t.} \quad Ax \le b, \quad Dx \le d, \quad H^k x = h^k.$

5. Solve $P_2^k$ to obtain $x^k$, an optimal primal vector (with value $z^k$) and
   $\mu_k$, an optimal dual vector corresponding to constraints $Dx \le d$.
   If $\mu_k = \mu_{k-1}$, STOP.

6. Set $k = k + 1$ and goto Step 2.
```

Fig. 1. Algorithmic template for solving $\boldsymbol{P_1}$

In this work we extend and recast these ideas in a generalized framework as an algorithm to systematically extract information from the Lagrangian and from restricted LP's symbiotically so as to solve the Lagrangian and the primal LP simultaneously.

In the template in Figure 1, at each iteration k we employ a linear system $H^k x = h^k$ that represents a structure satisfied by the current iteration's Lagrangian solution and which can be interpreted as an educated guess for conditions that an optimal solution to P_1 should satisfy. This is problem-specific; we will indicate later how this structure is discovered in the context of $GPCP$.

Notes:
1. Ideally, imposing $H^k x = h^k$ in Step 4 should result in an *easier* linear program.
2. For simplicity, in what follows we will assume that P_2^k is always feasible; though this is a requirement that can be easily circumvented in practice (full paper).

Theorem 3. *(a) If the algorithm stops at iteration k in Step 2, then x^{k-1} is optimal for P_1. (b) If it stops in Step 5 then x^k is optimal for P_1.*

Proof: (a) We have

$$z^{k-1} = \max\{c^T x \,+\, \mu_{k-1}^T(d - Dx) \,:\, Ax \le b,\ H^{k-1}x = h^{k-1}\} \ = \\ c^T w^k \,+\, \mu_{k-1}^T(d - Dw^k),$$

where the first equality follows by duality and the second by definition of w^k in Step 2 since $H^{k-1}w^k = h^{k-1}$. Also, clearly $z^{k-1} \le z^*$, and so in summary

$$z^* \le c^T w^k \,+\, \mu_{k-1}^T(d - Dw^k) = z^{k-1} \le z^*. \tag{22}$$

(b) $\mu_k = \mu_{k-1}$ implies that w^k optimally solves $L(P_1, \mu_k)$, so that we could choose $w^{k+1} = w^k$ and so $H^k w^{k+1} = h^k$, obtaining case (a) again. ■

4.1 Applying the Template

We will now apply the above template to a case where P_1 is an instance of GPCP where, as before, we denote by $\mathcal{N}$ the set of jobs; and we use $Ax \le b$ to describe the precedences and the box constraints $0 \le x_j \le 1$ ($\forall j \in \mathcal{N}$), and $Dx \le d$ denotes the side-constraints.

Thus, in Step 2 of the template, $L(P_1, \mu_{k-1})$ can be solved as a max closure problem (Observation 1); therefore its solution can be described by a 0/1-vector which we will denote by y^k. Recall that Lemma 3 implies that where D has m rows, an optimal extreme point solution to GPCP has $q \le m + 2$ distinct values $0 \le \alpha_1 < \alpha_1 < \cdots < \alpha_q \le 1$ and can therefore be written $x^* = \sum_{r=1}^{q} \alpha_r \theta^r$, where for $1 \le r \le q$, $V_j^r = \{j \in \mathcal{N} : x_j^* = \alpha_r\}$, and θ^r is the incidence vector of V^r.

The structure that we will "guess" has been exposed by the current iterate's Lagrangian solution is that the nodes inside the max closure should be distinguished from those nodes outside, i.e. that the nodes inside should not be required to take the same value in the LP solution as those outside. Given an existing partition of the nodeset $\mathcal{N}$ that represented our previous guess as to the sets $\{V^r\}$, this guess at structure implies a refinement of this partition. We will note later that this partition never needs more than a small number of elements for the algorithm to converge.

At iteration k, we denote by $\mathcal{C}^k = \{C_1^k, \dots, C_{r_k}^k\}$ the constructed partition of $\mathcal{N}$. Our basic application of the template is as follows:

GPCP Algorithm

1. Set $\mu_0 = 0$. Set $r_0 = 1$, $C_1^0 = \mathcal{N}$, $\mathcal{C}^0 = \{C_1^0\}$, $z^0 = -\infty$, and $k = 1$.
2. Let y^k be an optimal solution to $L(P_1, \mu_{k-1})$, and define
$$I^k = \{j \in \mathcal{N} : y_j^k = 1\} \tag{23}$$
and define
$$O^k = \{j \in \mathcal{N} : y_j^k = 0\}. \tag{24}$$
If $k > 1$, and, for $1 \le h \le r_{k-1}$, either $C_h^{k-1} \cap I^k = \emptyset$ or $C_h^{k-1} \cap O^k = \emptyset$, then **STOP**.
3. Let $\mathcal{C}^k = \{C_1^k, \dots, C_{r_k}^k\}$ consist of all *nonempty* sets in the collection
$$\left\{I^k \cap C_h^{k-1} \ : \ 1 \le h \le r_{k-1}\right\} \cup \left\{O^k \cap C_h^{k-1} \ : \ 1 \le h \le r_{k-1}\right\}.$$
Let $H^k x = h^k$ consist of the constraints:
$$x_i = x_j, \ \text{ for } 1 \le h \le r_k, \text{ and each pair } i, j \in C_h^k.$$
4. Let P_2^k consist of P_1, plus the additional constraints $H^k x = h^k$.
5. Solve P_2^k, with optimal solution x^k, and let μ_k denote the optimal duals corresponding to the side-constraints $Dx \le d$. If $\mu_k = \mu_{k-1}$, **STOP**.
6. Set $k = k + 1$ and goto Step **2**.

We have:

Lemma 4. *(a) For each k, problem P_2^k is an instance of GPCP with r_k variables and the same number of side-constraints as in $Dx \le d$. (b) If P_2^1 is feasible, the above algorithm terminates finitely with an optimal solution.*

Proof: full paper. ■

Comments: Since each problem P_2^k is a GPCP, its extreme point solution x^k never attains more than $m+2$ distinct values (where m is the number of linearly independent rows in D), and thus the partition $\mathcal{C}^k$ can be coarsened while maintaining the feasibility of x^k by merging the sets C_j^k with common x^k values. Note also that in choosing $\mathcal{C}^{k+1}$ to be a refinement of $\mathcal{C}^k$ the LP solution x^k remains available to the problem P_2^{k+1}. The above algorithm is a basic application of the template. Finer partitions than $\{I^k, O^k\}$ may also be used. The feasibility assumption in (b) of Lemma 4 can be bypassed. Details will be provided in the full paper.

In the full paper an analysis is presented that explains why the structure exposed by the Lagrangian solutions can be expected to point the algorithm in the right direction. In particular, the solution to the Lagrangian obtained by using optimal duals for the side constraints can be shown to exhibit significant structure.

Table 1. Sample runs, 1

	Marvin	**Mine1B**	**Mine2**	**Mine3,s**	**Mine3,b**
Jobs	9400	29277	96821	2975	177843
Precedences	145640	1271207	1053105	1748	2762864
Periods	14	14	25	8	8
Facilities	2	2	2	8	8
Variables	199626	571144	3782250	18970	3503095
Constraints	2048388	17826203	26424496	9593	19935500
Problem arcs	2229186	18338765	30013104	24789	23152350
Side-constraints	28	28	50	120	132
Binding side-constr. at optimum	14	11	23	33	44
Cplex time (sec)	55544	—	—	5	—
Algorithm Performance					
Iterations to 10^{-5} optimality	8	8	9	13	30
Time to 10^{-5} optimality (sec)	10	60	344	1	1076
Iterations to comb. optimality	11	12	16	15	39
Time to comb. optimality (sec)	15	96	649	1	1583

5 Computational Experiments

In this section we present results from some of our experiments. A more complete set of results will be presented in the full paper. All these tests were conducted using a single core of a dual quad-core 3.2 GHz Xeon machine with 64 GB of memory. The LP solver we used was Cplex, version 12 and the min cut solver we used was our implementation of Hochbaum's pseudoflow algorithm ([H08]).

The tests reported on in Tables 1 and 2 are based on three real-world examples provided by BHP Billiton[3], to which we refer as 'Mine1', 'Mine2' and 'Mine3' and a synthetic but realistic model called 'Marvin' which is included with Gemcom's Whittle [W] mine planning software. 'Mine1B' is a modification of Mine1 with a denser precedence graph. Mine3 comes in two versions to which we refer as 'big' and 'small'. Using Mine1, we also obtained smaller and larger problems by modifying the data in a number of realistic ways. Some of the row entries in these tables are self-explanatory; the others have the following meaning:

[3] Data was masked.

- **Problem arcs**. The number of arcs in the graph that the algorithm creates to represent the scheduling problem (i.e., the size of the min cut problems we solve).
- **Iterations, time to 10^{-5} optimality**. The number of iterations (resp., the CPU time) taken by the algorithm until it obtained a solution it could certify as having $\leq 10^{-5}$ **relative** optimality error.
- **Iterations, time to combinatorial optimality**. The number of iterations (resp., the CPU time) taken by the algorithm to obtain a solution it could certify as *optimal* as per the stopping criteria in Steps 2 or 5. Notice that this implies that the solution is optimal as per the numerical tolerances of Cplex.

Finally, an entry of "—" indicates that Cplex was unable to terminate after 100000 seconds of CPU time. More detailed analyses will appear in the full paper.

Table 2. Sample runs, 2

	Mine1 very small	**Mine1 medium**	**Mine1 large**	**Mine1 full**	**Mine1,3 weekly**
Jobs	755	7636	15003	29277	87831
Precedences	222	22671	113703	985011	2955033
Periods	12	12	12	12	100
Facilities	2	2	2	2	2
Variables	14282	160944	292800	489552	12238800
Constraints	8834	327628	1457684	11849433	295591331
Problem arcs	22232	477632	1727565	12280407	307654269
Side-constraints	24	24	24	24	200
Binding side-constr. at optimum	12	11	11	11	151
Cplex time (sec)	1	12424	—	—	—
Algorithm Performance					
Iterations to 10^{-5} optimality	6	6	8	7	10
Time to 10^{-5} optimality (sec)	0	1	7	45	2875
Iterations to comb. optimality	7	7	11	9	20
Time to comb. optimality (sec)	0	2	10	61	6633

References

[Bal70] Balinsky, M.L.: On a selection problem. Management Science 17, 230–231 (1970)

[BA00] Barahona, F., Anbil, R.: The Volume Algorithm: producing primal solutions with a subgradient method. Math. Programming 87, 385–399 (2000)

[B02] Bienstock, D.: Potential Function Methods for Approximately Solving Linear Programming Problems, Theory and Practice. Kluwer Academic Publishers, Boston (2002), ISBN 1-4020-7173-6

[BZ09] Bienstock, D., Zuckerberg, M.: A new LP algorithm for precedence constrained production scheduling, posted on Optimization Online (August 2009)

[BDFG09] Boland, N., Dumitrescu, I., Froyland, G., Gleixner, A.M.: LP-based disaggregation approaches to solving the open pit mining production scheduling problem with block processing selectivity. Computers and Operations Research 36, 1064–1089 (2009)

[CH03] Caccetta, L., Hill, S.P.: An application of branch and cut to open pit mine scheduling. Journal of Global Optimization 27, 349–365 (2003)

[CH09] Chandran, B., Hochbaum, D.: A Computational Study of the Pseudoflow and Push-Relabel Algorithms for the Maximum Flow Problem. Operations Research 57, 358–376 (2009)

[CEGMR09] Chicoisne, R., Espinoza, D., Goycoolea, M., Morena, E., Rubio, E.: A New Algorithm for the Open-Pit Mine Scheduling Problem (submitted for publication), `http://mgoycool.uai.cl/`

[F06] Fricke, C.: Applications of integer programming in open pit mine planning, PhD thesis, Department of Mathematics and Statistics, The University of Melbourne (2006)

[H08] Hochbaum, D.: The pseudoflow algorithm: a new algorithm for the maximum flow problem. Operations Research 58, 992–1009 (2008)

[HC00] Hochbaum, D., Chen, A.: Improved planning for the open - pit mining problem. Operations Research 48, 894–914 (2000)

[J68] Johnson, T.B.: Optimum open pit mine production scheduling, PhD thesis, Operations Research Department, University of California, Berkeley (1968)

[LG65] Lerchs, H., Grossman, I.F.: Optimum design of open-pit mines. Transactions C.I.M. 68, 17–24 (1965)

[P76] Picard, J.C.: Maximal Closure of a graph and applications to combinatorial problems. Management Science 22, 1268–1272 (1976)

[R70] Rhys, J.M.W.: A selection problem of shared fixed costs and network flows. Management Science 17, 200–207 (1970)

[W] Gemcom Software International, Vancouver, BC, Canada

Computing Minimum Multiway Cuts in Hypergraphs from Hypertree Packings[⋆]

Takuro Fukunaga

Department of Applied Mathematics and Physics,
Graduate School of Informatics, Kyoto University, Japan
takuro@amp.i.kyoto-u.ac.jp

Abstract. Hypergraph k-cut problem is a problem of finding a minimum capacity set of hyperedges whose removal divides a given hypergraph into k connected components. We present an algorithm for this problem which runs in strongly polynomial-time if both k and the rank of the hypergraph are constants. Our algorithm extends the algorithm due to Thorup (2008) for computing minimum k-cuts of graphs from greedy packings of spanning trees.

1 Introduction

Let $\mathbb{Q}_+$ denote the set of non-negative rationals. For a connected hypergraph $H = (V, E)$ with a non-negative hyperedge capacity $c : E \to \mathbb{Q}_+$ and an integer $k \geq 2$, a *k-cut* of H is defined as a subset of E whose removal divides H into k connected components. *Hypergraph k-cut problem* is a problem of finding a minimum capacity k-cut of a hypergraph. If the given hypergraph is a graph, then the problem is called *graph k-cut problem.*

The graph k-cut problem is one of the fundamental problems in combinatorial optimization. It is closely related to the reliability of networks, and has many applications, for example, to the traveling salesperson problem, VLSI design, and evolutionary tree construction [4,14]. By Goldschmidt and Hochbaum [6], it is shown that the problem is NP-hard when k is not fixed, and polynomial-time solvable when k is fixed to a constant. After their work, there are many works on the algorithmic aspect of this problem.

In spite of these active studies on the graph k-cut problem, there are few works on the hypergraph k-cut problem. If k is not fixed, the NP-hardness of the graph k-cut problem implies that of the hypergraph k-cut problem. When $k = 2$, the k-cut problem is usually called *the minimum cut problem.* Klimmek and Wagner [9] and Mak and Wong [13] extended an algorithm proposed by Stoer and Wagner [16] for the minimum cut problem in graphs to hypergraphs. Lawler [10] showed that the (s, t)-cut problem in hypergraphs can be reduced to computing maximum flows in digraphs. For the case of $k = 3$, Xiao [19] gave a polynomial-time algorithm.

⋆ This work was partially supported by Grant-in-Aid for Scientific Research from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

However, it is not known whether the hypergraph k-cut problem is polynomial solvable or NP-hard when k is a constant larger than 3.

In this paper, we partially answer this question. We present an algorithm which runs in strongly polynomial-time if k and the rank γ of hyperedges (i.e., $\gamma = \max_{e \in E} |e|$) are fixed to constants. Since graphs can be regarded as hypergraphs with $\gamma = 2$, this result extends the polynomial-solvability of the graph k-cut problem.

Our algorithm is based on the idea due to Thorup [18], which is successfully applied to the graph k-cut problem. He showed that a maximum spanning tree packing of a graph contains a spanning tree sharing at most a constant number of edges with a minimum k-cut of the graph. Although this fact itself gives a strongly polynomial-time algorithm for computing the minimum k-cuts of graphs, he also showed that a set of spanning trees constructed in a greedy way has the same property. Based on this fact, he gave the fastest algorithm to the graph k-cut problem. In this paper, we show that these facts can be extended to hypergraphs with a hypertree packing theorem due to Frank, Király and Kriesell [2] (see Section 3).

Let us mention the previous works on problems related to the hypergraph k-cut problem. As mentioned above, the first polynomial-time algorithm for the graph k-cut problem with fixed k was presented by Goldschmidt and Hochbaum [6]. Its running time is $O(n^{k^2} T(n,m))$ where $T(n,m)$ is time for computing max-flow in a graph consisting of n vertices and m edges. $T(n,m)$ is known to be $O(mn \log(n^2/m))$ for now [5]. After their work, many polynomial-time algorithms for fixed k are obtained. An algorithm due to Kamidoi, Yoshida and Nagamochi [7] runs in $O(n^{4k/(1-1.71/\sqrt{k})-34} T(n,m))$. An algorithm due to Xiao [20] runs in $O(n^{4k-\log k})$. An algorithm due to Thorup [17] runs in $\tilde{O}(n^{2k})$. In addition, Karger and Stein [8] gave a randomized algorithm running in $O(n^{2(k-1)} \log^3 n)$.

For the hypergraph k-cut problem, Xiao [19] gave a polynomial-time divide-and-conquer algorithm for $k = 3$. Zhao, Nagamochi and Ibaraki [21] gave an approximation algorithm. It achieves the approximation factor $(1 - \frac{2}{k}) \min\{k, \gamma\}$ for $k \geq 4$ by using the Xiao's algorithm due for $k = 3$ as a subroutine. Moreover, it is shown by Okumoto, Fukunaga and Nagamochi [15] that the problem can be reduced to the terminal k-vertex cut problem in bipartite graphs (refer to [15] for the definition of the terminal k-vertex cut problem). Hence the LP-rounding algorithm due to Garg, Vazirani and Yannakakis [3] for the terminal k-vertex cut problem achieves approximation factor $2 - \frac{2}{k}$ also for the hypergraph k-cut problem. Recently Chekuri and Korula [1] claims that the randomized algorithm proposed by Karger and Stein [8] for the graph k-cut problem can be extended to the hypergraph k-cut problem.

Okumoto, Fukunaga and Nagamochi [15] showed that the hypergraph k-cut problem is contained by the the submodular system k-partition problem. Zhao, Nagamochi and Ibaraki [21] presented a $(k-1)$-approximation algorithm to this problem. Okumoto, Fukunaga and Nagamochi [15] presented an approximation algorithm whose approximation factor is 1.5 for $k = 4$ and $k + 1 - 2\sqrt{k-1}$ for

$k \geq 5$. They also showed that, for the hypergraph 4-cut problem, their algorithm achieves approximation factor 4/3.

The rest of this paper is organized as follows. Section 2 introduces basic facts and notations. Section 3 explains outline of our result and presents our algorithm. Sections 4 shows that a maximum hypertree packing contains a hypertree sharing at most a constant number of hyperedges with a minimum k-cut. Section 5 discusses a property of a set of hypertrees constructed greedily. Section 6 concludes this paper and mentions the future works.

2 Preliminaries

Let $H = (V, E)$ be a hypergraph with a capacity $c : E \to \mathbb{Q}_+$. Throughout this paper, we denote $|V|$ by n, $|E|$ by m, and $\max_{e \in E} |e|$ by γ. We sometimes denote the vertex set of H by V_H, and the edge set of H by E_H, respectively. For non-empty $X \subset V$ and $F \subseteq E$, $\delta_F(X)$ denotes the set of hyperedges in F intersecting both X and $V \setminus X$. When $F = E$, we may represent $\delta_F(X)$ by $\delta(X)$. For some function $f : E \to \mathbb{Q}_+$ and $F \subseteq E$, $\sum_{e \in F} f(e)$ is represented by $f(F)$. For non-empty $X \subset V$, $E[X]$ denotes the set of hyperedges in E contained in X, and $H[X]$ denotes the sub-hypergraph $(X, E[X])$ of H.

It is sometimes convenient to associate a hypergraph $H = (V, E)$ with a bipartite graph $B_H = (V, V_E, E')$ as follows. Each vertex in V_E corresponds to an edge in E. $x \in V$ and $y \in V_E$ is joined by an edge in E' if and only if x is contained by the hyperedge corresponding to y in H.

H is called *ℓ-connected* if $c(\delta(X)) \geq \ell$ for all non-empty $X \subset V$. H is called *connected* if $|\delta(X)| \geq 1$ for all non-empty $X \subset V$. Notice that the 1-connectedness is not equivalent to the connectedness.

A partition $\mathcal{V} = \{V_1, V_2, \ldots, V_k\}$ of V into k non-empty subsets is called *k-partition* of V. We let $\delta_F(\mathcal{V}) = \cup_{i=1}^{\ell} \delta_F(V_i)$. H is called *ℓ-partition-connected* if $c(\delta(\mathcal{V})) \geq \ell(|\mathcal{V}| - 1)$ for all partitions $\mathcal{V}$ of V into non-empty subsets. It is easy to see that the ℓ-partition-connectivity is a stronger condition than the ℓ-connectivity. We call a partition $\mathcal{V}$ achieving $\min\{c(\delta(\mathcal{V}))/(|\mathcal{V}| - 1)\}$ *weakest*. $\min\{c(\delta(\mathcal{V}))/(|\mathcal{V}| - 1)\}$ is denoted by ℓ_H. If $|\delta(\mathcal{V})| \geq |\mathcal{V}| - 1$ for all $\mathcal{V}$, H is called *partition-connected*. Notice that the 1-partition-connectedness is not equivalent to the partition-connectedness

A minimal k-cut of H is represented by $\delta(\mathcal{V})$ where $\mathcal{V}$ is the k-partition of V consisting of the connected components after removing the k-cut. Hence the hypergraph k-cut problem is equivalent to the problem of finding a k-partition $\mathcal{V}$ of H minimizing $c(\delta(\mathcal{V}))$. We call such a partition *minimum k-partition*.

A *hyperforest* in $H = (V, E)$ is defined as $F \subseteq E$ such that $|F[X]| \leq |X| - 1$ for every non-empty $X \subseteq V$. A hyperforest F is called *hypertree* if $|F| = |V| - 1$ and $\cup_{e \in F} e = V$. Notice that if $H = (V, E)$ is a graph, $F \subseteq E$ is a hypertree if and only if F is a spanning tree. Actually a hypertree is an extension of a spanning tree which inherits many important properties of spanning trees. However, there is also a difference between them. For example, in contrast to spanning trees, a connected hypergraph may contain no hypertree.

A *hypertree packing* of H is a pair of a set $\mathscr{T}$ of hypertrees in H and a non-negative weight $\alpha : \mathscr{T} \rightarrow \mathbb{Q}_+$ such that $\alpha(\mathscr{T}_e) \leq c(e)$ holds for all $e \in E$ where $\mathscr{T}_e$ denotes the set of hypertrees in $\mathscr{T}$ containing e. A hypertree packing is called *maximum* if $\alpha(\mathscr{T})$ is maximum. We define the *packing value* of a hypergraph H as the maximum of $\alpha(\mathscr{T})$ over all hypertree packings of H. If α is integer, then the hypertree packing $(\mathscr{T}, \alpha)$ is called *integer*.

Frank, Király and Kriesell [2] characterized hypergraphs containing hypertrees as follows.

Theorem 1 (Frank, Király, Kriesell [2]). *Let H be a hypergraph with integer hyperedge capacity. H has an integer hypertree packing $(\mathscr{T}, \alpha)$ such that $\alpha(\mathscr{T}) \geq \ell$ if and only if H is ℓ-partition-connected.*

Let $\mathcal{F}$ be the family of hyperforests. In the proof of Theorem 1, it is mentioned that $(E, \mathcal{F})$ is a matroid, which is originally proven by Lorea [11]. Matroids defined from hypergraphs in such a way are called *hypergraphic matroids*. Hypertrees are bases of the hypergraphic matroid.

Independence testing in hypergraphic matroids needs to judge whether a given $F \subseteq E$ satisfies $F[X] \leq |X| - 1$ for every $\emptyset \neq X \subseteq V$. By Hall's theorem, this condition holds if and only if the bipartite graph $B_H = (V, V_E, E')$ defined from H contains a matching covering V after removing any vertex $v \in V_E$. Thus the independence testing can be done in $O(n\theta(n, m-1, \gamma m))$ time where $\theta(n, m-1, \gamma m)$ denotes the time for computing maximum matchings in bipartite graph $B_H = (V, V_E, E') - v$ with $|V| = n$, $|V_E| = m$ and $|E'| \leq \gamma m$.

3 Outline of Our Result

The first step of our result is to prove the following theorem originally proven for graphs by Thorup [18]. A *recursively maximum hypertree packing* is a maximum hypertree packing that satisfies some condition, which will be defined formally in Section 4.

Theorem 2. *A recursively maximum hypertree packing of H contains a hypertree that shares at most $\gamma k - 3$ hyperedges with a minimum k-cut of H.*

We prove Theorem 2 in Section 4.

Assume that the hypertree and the $h = \gamma k - 3$ hyperedges in Theorem 2 are specified. Since each of the other $n-1-h$ hyperedges in the hypertree intersects only one elements of the minimum k-partition, shrinking them into single vertices preserves the minimum k-partition. If $\gamma = 2$, these $n-1-h$ hyperedges form at most $h+1$ connected components. Hence the hypergraph obtained by shrinking them contains at most $h+1$ vertices, for which the minimum k-partition can be found by enumerating all k-partitions. If $\gamma \geq 3$, the number of the connected components cannot be bounded in general because one large deleted hyperedge may connect many components. However, a characterization of hypertrees due to Lovász [12] tells that such a case does not occur even if $\gamma \geq 3$.

Theorem 3 (Lovász [12]). *Consider an operation that replaces each hyperedge by an edge joining two vertices chosen from the hyperedge. It is possible to construct a spanning tree from a hypergraph by this operation if and only if the hypergraph is a hypertree.*

Corollary 1. *After removing h hyperedges from a hypertree, there exist at most $h+1$ connected components.*

Proof. Consider the spanning tree constructed from a hypertree as shown by Theorem 3. After removing h edges from the spanning tree, the remaining edges forms $h+1$ connected components. The vertices in the same connected component are also connected by the hyperedges corresponding to the remaining edges. Hence removing h hyperedges from a hypertree results in at most $h+1$ connected components. □

Another thing to care is the existence of hypertrees. As mentioned in Section 2, there exist connected hypergraphs which contain no hypertrees. For such hypergraphs, hypertree packings give no information on minimum k-cuts. We avoid this situation by replacing each hyperedge $e \in E$ by its $|e|$ copies with capacity $c(e)/|e|$. Obviously this replacement makes no effect on capacities of k-partitions while the obtained hypergraphs contain hypertrees. Notice that after the replacement, the number of hyperedges are increased to at most γm.

Theorem 4. *Let $H' = (V, E')$ be the hypergraph obtained from a connected hypergraph $H = (V, E)$ by replacing each $e \in E$ by $|e|$ copies of e. Then H' contains a hypertree.*

Proof. Let $\mathcal{V}$ be a partition of V. Since each $e' \in \delta_{E'}(\mathcal{V})$ intersects at most $|e'|$ components of $\mathcal{V}$, $|\delta_{E'}(\mathcal{V})| \geq \sum_{U \in \mathcal{V}} \sum_{e' \in \delta_{E'}(U)} (1/|e'|)$. Since each $e \in E$ has $|e|$ copies in E', $\sum_{e' \in \delta_{E'}(U)} (1/|e'|) = \sum_{e \in \delta_E(U)} 1 = |\delta_E(U)|$. Moreover, $|\delta_E(U)| \geq 1$ because H is connected. Thus $|\delta_{E'}(\mathcal{V})| \geq \sum_{U \in \mathcal{V}} 1 = |\mathcal{V}|$, which implies that H' is partition-connected. Hence by Theorem 1, H' contains a hypertree. □

Now we describe our algorithm for the hypergraph k-cut problem.

```
Algorithm 1: Hypergraph $k$-Cut Algorithm
Input: A connected hypergraph $H = (V, E)$ with capacity $c : E \to \mathbb{Q}_+$ and an
    integer $k \geq 2$
Output: A minimum $k$-cut of $H$
Step 1: For each $e \in E$, prepare $|e|$ copies $e_1, e_2, \ldots, e_{|e|}$ of $e$ with capacity
    $c(e_i) = c(e)/|e|$, $i \in \{1, 2, \ldots, |e|\}$, and replace $e$ by them.
Step 2: Define $F = E$. Compute a recursively maximum hypertree $(\mathscr{T}^*, \alpha^*)$ of
    $H$.
Step 3: For each $T \in \mathscr{T}^*$ and each set $T'$ of $h = \gamma k - 3$ hyperedges in $T$,
    execute the following operations.
    3-1: Compute a hypergraph $H'$ obtained by shrinking all hyperedges in
        $T \setminus T'$.
    3-2: Compute a minimum $k$-cut $F'$ of $H'$.
    3-3: Let $F := F'$ if $c(F') \leq c(F)$.
Step 4: Output $F$.
```

Let us discuss the running time of this algorithm. For each hypertree, there are $O(n^h)$ ways to choose h hyperedges. By Corollary 1, shrinking $n-1-h$ hyperedges in a hypertree results in a hypergraph with at most $h+1$ vertices. Hence Step 3-2 can be done in $O(k^{h+1})$ time. It means that Step 3 of the algorithm runs in $O(k^{h+1}n^h)$ time per one hypertree in $\mathscr{T}^*$.

To bound the running time of all the steps, we must consider how to compute a recursively maximum hypertree packing and how large its size is. A recursively maximum hypertree packing can be computed in polynomial time. However we know no algorithm to compute small recursively maximum hypertree packings. Hence this paper follows the approach taken by Thorup [18] for the graph k-cut problem. We show that a set of hypertrees constructed as below approximates a recursively maximum hypertree packing well. It enables us to avoid computing a recursively maximum hypertree packing.

Algorithm 2: Greedy Algorithm for Computing a Set of Hypertrees

```
Input: A connected hypergraph $H = (V, E)$ with capacity $c : E \to \mathbb{Q}_+$ and an
    integer $t$.
Output: A set of $t$ hypertrees of $H$.
Step 1: Let $\mathscr{T} := \emptyset$.
Step 2: Compute a minimum cost hypertree $T$ of $H$ with respects to the cost
    defined as $|\mathscr{T}_e|/c(e)$ for each $e \in E$, and $\mathscr{T} := \mathscr{T} \cup \{T\}$.
Step 3: If $|\mathscr{T}| = t$, then output $\mathscr{T}$. Otherwise, return to Step 2.
```

As mentioned in Section 2, hypertrees are bases of a hypergraphic matroid. Hence a minimum cost hypertree can be computed by a greedy algorithm. The running time of Algorithm 2 is $O(t\gamma m \log(\gamma m) n\theta(n, \gamma m - 1, \gamma^2 m))$. The set of hypertrees computed by Algorithm 2 approximates the recursively maximum hypertree packing well.

Theorem 5. *Let $H = (V, E)$ be a hypergraph such that each $e \in E$ has at least $|e| - 1$ copies in $E \setminus \{e\}$ of the same capacity. For this H, Algorithm 2 with $t = 24\gamma^4 mk^3 \ln(2\gamma^2 kmn)$ outputs a set of hypertrees which contains a hypertree sharing at most $h = \gamma k - 2$ hyperedges with a minimum k-cut of H.*

Replace the computation of recursively maximum hypertree packings in Step 2 of Algorithm 1 by Algorithm 2 with $t = 24\gamma^4 mk^3 \ln(2\gamma^2 kmn)$. Moreover, change the definition of h in Step 3 as $\gamma k - 2$. Then we obtain another algorithm for the hypertree k-cut problem as summarized in the next corollary.

Corollary 2. *The hypergraph k-cut problem is solvable in time*

$$O(k^{\gamma k+2} n^{\gamma k-1} \gamma^5 m^2 \theta(n, \gamma m - 1, \gamma^2 m) \log(k\gamma^2 mn) \log(\gamma m)).$$

4 Proof of Theorem 2

Let $\mathcal{V}$ be an arbitrary partition of V, and $(\mathscr{T}, \alpha)$ be an arbitrary hypertree packing of H. Since every hypertree $T \in \mathscr{T}$ satisfies $|T| = |V| - 1$ and has at most $|U| - 1$ hyperedges contained by U for each $U \in \mathcal{V}$, we have

$$|\delta_T(\mathcal{V})| = |T| - \sum_{U \in \mathcal{V}} |T[U]| \geq |V| - 1 - \sum_{U \in \mathcal{V}} (|U| - 1) = |\mathcal{V}| - 1. \tag{1}$$

Moreover,

$$c(e) \geq \alpha(\mathscr{T}_e) \quad \text{for each } e \in \delta(\mathcal{V}) \tag{2}$$

by the definition of hypertree packings. Thus it follows that

$$\frac{c(\delta(\mathcal{V}))}{|\mathcal{V}| - 1} \geq \frac{\sum_{e \in \delta(\mathcal{V})} \alpha(\mathscr{T}_e)}{|\mathcal{V}| - 1} = \frac{\sum_{T \in \mathscr{T}} \alpha(T) |\delta_T(\mathcal{V})|}{|\mathcal{V}| - 1} \geq \alpha(\mathscr{T}). \tag{3}$$

Let $\mathcal{V}^*$ be a weakest partition of V (i.e., it attains $\min_{\mathcal{V}} c(\delta(\mathcal{V}))/(|\mathcal{V}|-1)$), and $(\mathscr{T}^*, \alpha^*)$ be a maximum hypertree packing of H (i.e., it attains $\max_{(\mathscr{T},\alpha)} \alpha(\mathscr{T})$). From Theorem 1, we can derive their important properties.

Lemma 1. *$\mathcal{V}^*$ and $(\mathscr{T}^*, \alpha^*)$ satisfy $c(\delta(\mathcal{V}^*))/(|\mathcal{V}^*| - 1) = \alpha^*(\mathscr{T}^*)$. Moreover, $|\delta_T(\mathcal{V}^*)| = |\mathcal{V}^*| - 1$ holds for each $T \in \mathscr{T}^*$, and $\alpha^*(\mathscr{T}_e^*) = c(e)$ holds for each $e \in \delta(\mathcal{V}^*)$. $T[U]$ defined from any $T \in \mathscr{T}^*$ and $U \in \mathcal{V}^*$ is a hypertree on $H[U]$.*

Proof. Let M be a positive integer such that all of $Mc(e)$, $e \in E$ and $M\alpha^*(T)$, $T \in \mathscr{T}^*$ are integers. Notice that $(\mathscr{T}^*, M\alpha^*)$ is a maximum hypertree packing of the hypergraph H associated with hyperedge capacity Mc. Applying Theorem 1 to this hypergraph shows that $Mc(\delta(\mathcal{V}^*))/(|\mathcal{V}^*| - 1) = \sum_{T \in \mathscr{T}^*} M\alpha^*(T)$ holds. That is to say, $\mathcal{V}^*$ and $(\mathscr{T}^*, \alpha^*)$ satisfy $c(\delta(\mathcal{V}^*))/(|\mathcal{V}^*| - 1) = \alpha^*(\mathscr{T}^*)$.

Since $\mathcal{V}^*$ and $(\mathscr{T}^*, \alpha^*)$ satisfy (3) with equality, they also satisfy (1) and (2), used for deriving (3), with equality. This proves the remaining part of the lemma. □

Let $U \in \mathcal{V}^*$, $\mathscr{T}' = \{T[U] \mid T \in \mathscr{T}^*\}$, and α' be the weight defined on the hypertrees in $\mathscr{T}'$ such that $\alpha'(T[U]) = \alpha^*(T)$ for $T \in \mathscr{T}^*$. Since $\mathscr{T}'$ consists of hypertrees of $H[U]$ by Lemma 1, $(\mathscr{T}', \alpha')$ is a hypertree packing of $H[U]$. However, it may not be a maximum hypertree packing of $H[U]$ since the partition-connectivity of $H[U]$ may be larger than $\alpha'(\mathscr{T}')$. Let $(\mathscr{S}, \beta)$ be a maximum hypertree packing of $H[U]$. For each $T \in \mathscr{T}^*$ and $S \in \mathscr{S}$, replacing hyperedges in T contained by U with those in S generates another hypertree of H because hypertrees are bases of hypergraphic matroids. Hence, from $(\mathscr{T}, \alpha)$ and $(\mathscr{S}, \beta)$, we can construct another maximum hypertree packing $(\mathscr{U}, \zeta)$ of H such that $|\mathscr{U}| \leq |\mathscr{T}| + |\mathscr{S}|$ and $(\mathscr{U}', \zeta')$ is a maximum hypertree packing of $H[U]$ where $\mathscr{U}' = \{T[U] \mid T \in \mathscr{U}\}$ and $\zeta'(T[U]) = \zeta(T)\beta(\mathscr{S})/\zeta(\mathscr{U})$ for each $T[U] \in \mathscr{U}'$. A maximum hypertree packing obtained by repeating this operation is called *recursively maximum*. That is to say, a recursively maximum hypertree packing is defined as a hypertree packing computed by the following algorithm.

```
Algorithm 3: Computing a Recursively Maximum Hypertree Packing

Input: A connected hypergraph $H = (V, E)$ with capacity $c : E \to \mathbb{Q}_+$.
Output: A recursively maximum hypertree packing of $H$.
Step 1: Compute a maximum hypertree packing $(\mathscr{T}^*, \alpha^*)$ of $H$, and a weakest
    partition $\mathcal{V}^*$ of $H$.
Step 2: While there exits $U \in \mathcal{V}^*$ such that $|U| > 1$, execute the following
    operations.
    2-1: Compute a maximum hypertree packing $(\mathscr{S}, \beta)$ of $H[U]$ and a weakest
        partition $\mathcal{V}$ of $H[U]$. Define $\mathscr{T} := \emptyset$, and $\beta'(S) := \beta(S)\alpha^*(\mathscr{T}^*)/\beta(\mathscr{S})$
        for each $S \in \mathscr{S}$.
    2-2: Choose $T \in \mathscr{T}^* \setminus \mathscr{T}$ and $S \in \mathscr{S}$. If $\alpha^*(T) < \beta'(S)$, then replace the
        hyperedges in $T[U]$ by those in $S$, $\beta'(S) := \beta'(S) - \alpha^*(T)$, and $\mathscr{T} :=$
        $\mathscr{T} \cup \{T\}$. Otherwise, i.e., $\alpha^*(T) \geq \beta'(S)$, then construct a hypertree $T' =$
        $(T \setminus T[U]) \cup S$ with $\alpha^*(T') := \beta'(S)$, and update $\alpha^*(T) := \alpha^*(T) - \beta(S)$,
        $\mathscr{T} := \mathscr{T} \cup \{T'\}$, and $\mathscr{S} := \mathscr{S} \setminus \{S\}$. If $\alpha^*(T) = \beta'(S)$ in the latter case,
        remove $T$ from $\mathscr{T}^*$ in addition.
    2-3: If $\mathscr{T}^* \setminus \mathscr{T} \neq \emptyset$, then return to Step 2-2.
    2-4: $\mathcal{V}^* := (\mathcal{V}^* \setminus \{U\}) \cup \mathcal{V}$.
Step 3: Output $(\mathscr{T}^*, \alpha^*)$.
```

From now on, we let $(\mathscr{T}^*, \alpha^*)$ stand for a recursively maximum hypertree packing. For $U \in \mathcal{V}^*$, let $\mathscr{T}' = \{T[U] \mid T \in \mathscr{T}^*\}$ and $\alpha'(T[U]) = \alpha^*(T)\ell_{H[U]}/\alpha^*(\mathscr{T}^*)$ where $\ell_{H[U]}$ is the partition-connectivity of $H[U]$. The definition of $(\mathscr{T}^*, \alpha^*)$ implies that $(\mathscr{T}', \alpha')$ is a recursively maximum hypertree packing of $H[U]$ for any $U \in \mathcal{V}^*$.

From $\mathscr{T}^*$ and given k, define $\mathcal{V}_k$ as the k-partition of V constructed by the following algorithm.

```
Algorithm 4: Computing $\mathcal{V}_k$
Input: A connected hypergraph $H = (V, E)$ with capacity $c : E \to \mathbb{Q}_+$, and an
    integer $k \geq 2$.
Output: A $k$-partition of $V$.
Step 1: Define $\mathcal{V}_k := \{V\}$.
Step 2: Let $U \in \mathcal{V}_k$ be a set attaining $\min\{\ell_{H[U]} \mid U \in \mathcal{V}_k, |U| \geq 2\}$. Compute
    a weakest partition $\mathcal{U} = \{U_1, U_2, \ldots, U_{|\mathcal{U}|}\}$ of $H[U]$, where we assume that
    $\sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(U_i) \cap T[U]| \leq \sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(U_j) \cap T[U]|$ for $1 \leq i < j \leq$
    $|\mathcal{U}|$.
Step 3: If $|\mathcal{V}_k| - 1 + |\mathcal{U}| < k$, then $\mathcal{V}_k := (\mathcal{V}_k \setminus \{U\}) \cup \mathcal{U}$ and return to Step 2.
Step 4: If $|\mathcal{V}_k| - 1 + |\mathcal{U}| \geq k$, then $\mathcal{V}_k := (\mathcal{V}_k \setminus \{U\}) \cup \{U_1, U_2, \ldots, U_{k-|\mathcal{V}_k|}, U \setminus$
    $\cup_{i=1}^{k-|\mathcal{V}_k|} U_i\}$, and output $\mathcal{V}_k$.
```

Lemma 2

$$\sum_{e \in \delta(\mathcal{V}_k)} \alpha^*(\mathscr{T}^*_e) < (\gamma k - 2)\alpha^*(\mathscr{T}^*).$$

Proof. In this proof, $\mathcal{V}'_k$ stands for $\mathcal{V}_k$ immediately before executing Step 4 of Algorithm 4, and $\mathcal{V}_k$ stands for it outputted by Algorithm 4.

By the definition of recursively maximum hypertree packings, $T[U]$ is a hypertree of $H[U]$ for every pair of $U \in \mathcal{V}'_k$ and $T \in \mathscr{T}^*$. Thus $|\delta(\mathcal{V}'_k) \cap T| = |T| - \sum_{U \in \mathcal{V}'_k} |T[U]| = |V| - 1 - \sum_{U \in \mathcal{V}'_k}(|U| - 1) = |\mathcal{V}'_k| - 1$ holds for each $T \in \mathscr{T}^*$, and hence $\sum_{e \in \delta(\mathcal{V}'_k)} \alpha^*(\mathscr{T}^*_e) = \sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(\mathcal{V}'_k) \cap T| = (|\mathcal{V}'_k| - 1)\alpha^*(\mathscr{T}^*)$ holds.

Let U be the element of $\mathcal{V}'_k$ and $\mathcal{U} = \{U_1, U_2, \ldots, U_{|\mathcal{U}|}\}$ be the weakest partition of U computed in Step 2 immediately before executing Step 4. Note that $|\mathcal{U}| > k - |\mathcal{V}'_k|$ holds by the condition of Step 4. By the same reason with above, $|\delta(\mathcal{U}) \cap T[U]| = |\mathcal{U}| - 1$ holds for each $T \in \mathscr{T}^*$. Hence $\sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(\mathcal{U}) \cap T[U]| = (|\mathcal{U}| - 1)\alpha^*(\mathscr{T}^*)$.

Let $\mathcal{V}_U = \{U_1, U_2, \ldots, U_{k-|\mathcal{V}'_k|}, U \setminus \cup_{j=1}^{k-|\mathcal{V}'_k|} U_j\}$. Then

$$\sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(\mathcal{V}_U) \cap T[U]| \le \sum_{j=1}^{k-|\mathcal{V}'_k|} \sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(U_j) \cap T[U]|.$$

The elements in $\mathcal{U}$ are ordered so that they satisfy $\sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(U_i) \cap T[U]| \le \sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(U_j) \cap T[U]|$ for $1 \le i < j \le |\mathcal{U}|$. Hence it holds that

$$\sum_{j=1}^{k-|\mathcal{V}'_k|} \sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(U_j) \cap T[U]| \le \frac{k - |\mathcal{V}'_k|}{|\mathcal{U}|} \sum_{j=1}^{|\mathcal{U}|} \sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(U_j) \cap T[U]|.$$

Since each hyperedge intersects at most γ elements in $\delta(\mathcal{U})$, it holds that

$$\begin{aligned}\frac{k - |\mathcal{V}'_k|}{|\mathcal{U}|} \sum_{j=1}^{|\mathcal{U}|} \sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(U_j) \cap T[U]| &\le \frac{k - |\mathcal{V}'_k|}{|\mathcal{U}|} \gamma \sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(\mathcal{U}) \cap T[U]| \\ &= \frac{k - |\mathcal{V}'_k|}{|\mathcal{U}|} \gamma (|\mathcal{U}| - 1)\alpha^*(\mathscr{T}^*) \\ &< (k - |\mathcal{V}'_k|)\gamma\alpha^*(\mathscr{T}^*).\end{aligned}$$

Combining these implies that $\sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(\mathcal{V}_U) \cap T[U]| < (k - |\mathcal{V}'_k|)\gamma\alpha^*(\mathscr{T}^*)$.

Notice that $\delta(\mathcal{V}_k) \cap T = (\delta(\mathcal{V}'_k) \cap T) \cup (\delta(\mathcal{V}_U) \cap T[U])$. Recall that $|\mathcal{V}'_k| \ge 1$ and $\gamma \ge 2$. Therefore it follows that

$$\begin{aligned}\sum_{e \in \delta(\mathcal{V}_k)} \alpha^*(\mathscr{T}^*_e) &= \sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(\mathcal{V}_k) \cap T| \\ &< \{(|\mathcal{V}'_k| - 1) + (k - |\mathcal{V}'_k|)\gamma\}\alpha^*(\mathscr{T}^*) \le (\gamma k - 2)\alpha^*(\mathscr{T}^*).\end{aligned}$$

□

Lemma 3. *For each $e \in \delta(\mathcal{V}_k)$ and $f \in E \setminus \delta(\mathcal{V}_k)$, $\alpha^*(\mathscr{T}^*_e)/c(e) \ge \alpha^*(\mathscr{T}^*_f)/c(f)$ holds.*

Proof. Let $U \in \mathcal{V}_k$ denote the set containing f (i.e., $f \in E[U]$). Let $\mathcal{V}'_k$ denote $\mathcal{V}_k$ immediately before e enters $\delta(\mathcal{V}_k)$ in Algorithm 4. Assume that e is contained by $U' \in \mathcal{V}'_k$ (i.e., $e \in E[U']$). Moreover, let ℓ_U (resp., $\ell_{U'}$) denote the packing value of $H[U]$ (resp., $H[U']$).

From $(\mathscr{T}^*, \alpha^*)$, define $\mathscr{T}' = \{T[U'] \mid T \in \mathscr{T}^*\}$. Moreover, define $\alpha'(T[U']) = \alpha^*(T)\ell_{U'}/\alpha^*(\mathscr{T}^*)$ for $T \in \mathscr{T}^*$. By the definition of recursively maximum hypertree packings, $(\mathscr{T}', \alpha')$ is a maximum hypertree packing of $H[U']$. By Lemma 1, the capacity constraint of edge e is tight for any maximum hypertree packing of $H[U']$, i.e., $c(e) = \alpha'(\mathscr{T}'_e)$. Since $\alpha'(\mathscr{T}'_e) = \alpha^*(\mathscr{T}^*_e)\ell_{U'}/\alpha^*(\mathscr{T}^*)$, it holds that $\ell_{U'} = c(e)\alpha^*(\mathscr{T}^*)/\alpha^*(\mathscr{T}^*_e)$.

On the other hand, a maximum hypertree packing of $H[U]$ satisfies the capacity constraint for edge f. Hence, similarly with above, $\ell_U \leq c(f)\alpha^*(\mathscr{T}^*)/\alpha^*(\mathscr{T}^*_f)$.

$\mathcal{V}'_k$ contains U'' such that $U \subseteq U''$. In other words, $U = U''$, or U is obtained by dividing U'' in Algorithm 4. As explained when recursively maximum hypertree packings are defined, $\ell_{U''} \leq \ell_U$ holds. Since Step 2 chose U' immediately before e enters $\delta(\mathcal{V}_k)$, $\ell_{U'} \leq \ell_{U''}$ holds. These facts show that

$$\frac{c(e)\alpha^*(\mathscr{T}^*)}{\alpha^*(\mathscr{T}^*_e)} = \ell_{U'} \leq \ell_{U''} \leq \ell_U \leq \frac{c(f)\alpha^*(\mathscr{T}^*)}{\alpha^*(\mathscr{T}^*_f)},$$

implying the required inequality. □

Let $\mathcal{V}^{\text{opt}}$ denote a minimum k-partition of H.

Lemma 4

$$\sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(\mathcal{V}^{\text{opt}}) \cap T| \leq \sum_{e \in \delta(\mathcal{V}_k)} \alpha^*(\mathscr{T}^*_e).$$

Proof. Let $\eta = \min_{e \in \delta(\mathcal{V}_k)} \alpha^*(\mathscr{T}^*_e)/c(e)$. By Lemma 3, each hyperedge $e \in \delta(\mathcal{V}^{\text{opt}}) \setminus \delta(\mathcal{V}_k)$ satisfies $\alpha^*(\mathscr{T}^*_e)/c(e) \leq \eta$. Hence it holds that

$$\sum_{e \in \delta(\mathcal{V}^{\text{opt}}) \setminus \delta(\mathcal{V}_k)} \alpha^*(\mathscr{T}^*_e) = \sum_{e \in \delta(\mathcal{V}^{\text{opt}}) \setminus \delta(\mathcal{V}_k)} c(e)\frac{\alpha^*(\mathscr{T}^*_e)}{c(e)} \leq \eta c(\delta(\mathcal{V}^{\text{opt}}) \setminus \delta(\mathcal{V}_k)).$$

The definition of $\mathcal{V}^{\text{opt}}$ implies that $c(\delta(\mathcal{V}^{\text{opt}})) \leq c(\delta(\mathcal{V}_k))$, and hence $c(\delta(\mathcal{V}^{\text{opt}}) \setminus \delta(\mathcal{V}_k)) \leq c(\delta(\mathcal{V}_k) \setminus \delta(\mathcal{V}^{\text{opt}}))$. Thus

$$\begin{aligned}
\sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(\mathcal{V}^{\text{opt}}) \cap T| &= \sum_{e \in \delta(\mathcal{V}^{\text{opt}}) \cap \delta(\mathcal{V}_k)} \alpha^*(\mathscr{T}^*_e) + \sum_{e \in \delta(\mathcal{V}^{\text{opt}}) \setminus \delta(\mathcal{V}_k)} \alpha^*(\mathscr{T}^*_e) \\
&\leq \sum_{e \in \delta(\mathcal{V}^{\text{opt}}) \cap \delta(\mathcal{V}_k)} \alpha^*(\mathscr{T}^*_e) + \eta c(\delta(\mathcal{V}^{\text{opt}}) \setminus \delta(\mathcal{V}_k)) \\
&\leq \sum_{e \in \delta(\mathcal{V}^{\text{opt}}) \cap \delta(\mathcal{V}_k)} \alpha^*(\mathscr{T}^*_e) + \eta c(\delta(\mathcal{V}_k) \setminus \delta(\mathcal{V}^{\text{opt}})) \\
&\leq \sum_{e \in \delta(\mathcal{V}_k)} \alpha^*(\mathscr{T}^*_e).
\end{aligned}$$

□

From Lemmas 2 and 4, we can observe that

$$\frac{\sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(\mathcal{V}^{\mathrm{opt}}) \cap T|}{\alpha^*(\mathscr{T}^*)} < \gamma k - 2.$$

This means that $|\delta(\mathcal{V}^{\mathrm{opt}}) \cap T| < \gamma k - 2$ holds for some $T \in \mathscr{T}^*$. Therefore Theorem 2 has been proven.

5 Proof of Theorem 5

In this section, we present a proof of Theorem 5. Although it is almost same with that for $\gamma = 2$ presented by Thorup [18], we sketch it for self-containment.

Throughout this section, we let $H = (V_H, E_H)$ be a hypergraph such that each $e \in E_H$ has at least $|e| - 1$ copies in $E_H \setminus \{e\}$ of the same capacity. We denote $|E_H|$ by γm, and the capacity of hyperedges in H by c_H in order to avoid confusion. Moreover, we assume that a recursively maximum hypertree packing $(\mathscr{T}^*, \alpha^*)$ of H satisfies $\alpha^*(\mathscr{T}^*_e) = \alpha^*(\mathscr{T}^*_{e'})$ for $e \in E_H$ and a copy $e' \in E_H$ of e.

For a set $\mathscr{T}$ of hypertrees of H and $e \in E_H$, define $u^{\mathscr{T}}_H(e) = |\mathscr{T}_e|/(c_H(e)|\mathscr{T}|)$. For each $e \in E_H$, we also define $u^*_H(e)$ as $\alpha^*(\mathscr{T}^*_e)/(c_H(e)\alpha^*(\mathscr{T}^*))$ from a recursively maximum hypertree packing $(\mathscr{T}^*, \alpha^*)$ of H. Since $c_H(e) \geq \alpha^*(\mathscr{T}^*_e)$ for all $e \in E_H$, $1/u^*_H(e)$ is at least the packing value of H, i.e., $1/u^*_H(e) \geq \alpha^*(\mathscr{T}^*)$. Moreover, since $c_H(e) = \alpha^*(\mathscr{T}^*_e)$ holds for some $e \in E_H$ by the maximality of $(\mathscr{T}^*, \alpha^*)$, $\min_{e \in E_H} 1/u^*_H(e) = \alpha^*(\mathscr{T}^*)$ holds.

Recall that Algorithm 3 updates $\mathcal{V}^*$ by partitioning non-singleton sets in $\mathcal{V}^*$ repeatedly until no such sets exist. For $e \in E_H$, define U_e as the last set in $\mathcal{V}^*$ such that $e \in E_H[U_e]$ during the execution of the algorithm. Then $\max_{e' \in E_H[U_e]} u^*_{H[U_e]}(e') = u^*_{H[U_e]}(e)$. The definition of recursively maximum hypertree packings implies that $u^*_{H[U_e]}(e') = u^*_H(e')$ for each $e' \in E_H[U_e]$ because $\alpha^*(\mathscr{T}^*_{e'})/\alpha^*(\mathscr{T}^*) = \beta(\mathscr{S}_{e'})/\beta(\mathscr{S})$ holds with a maximum hypertree packing $(\mathscr{S}, \beta)$ of $H[U_e]$. Therefore, the partition-connectivity of $H[U_e]$ is $1/u^*_H(e)$.

Lemma 5. *Let I be a subgraph of H and assume that each hyperedge e in I has capacity $c_I(e)$ such that $c_{\min} \leq c_I(e) \leq c_H(e)$. Let $C = \sum_{e \in E_I} c_I(e)$, and $u_I = \max_{e \in E_I} u^*_I(e)$. Moreover, let ϵ be an arbitrary real such that $0 < \epsilon < 1/2$, and $\mathscr{T}^{\mathrm{g}}$ be a set of hypertrees in H constructed by Algorithm 2 with $t \geq 3\ln(C/c_{\min})/(c_{\min} u_I \epsilon^2)$. Then*

$$u^{\mathscr{T}^{\mathrm{g}}}_H(e) < (1+\epsilon)u_I \tag{4}$$

holds for each $e \in E_I$.

Proof. Scaling hyperedge capacity makes no effect on the claim. Hence we assume without loss of generality that $c_{\min} = 1$.

Let $\mathscr{T}$ denote a set of hypertrees kept by Algorithm 2 at some moment during it is running for computing $\mathscr{T}^{\mathrm{g}}$. The key is the following quantity:

$$\sum_{e \in E_I} c_I(e) \frac{(1+\epsilon)^{|\mathscr{T}_e|/c_H(e)}(1+\epsilon u_I)^{t-|\mathscr{T}|}}{(1+\epsilon)^{(1+\epsilon)u_I t}}. \tag{5}$$

This quantity has the following properties:

(i) When $\mathscr{T} = \emptyset$, (5) is less than 1;
(ii) If (5) is less than 1 when $|\mathscr{T}| = t$, then (4) holds for all $e \in E_I$;
(iii) When a tree is added to $\mathscr{T}$ in Step 2 of Algorithm 2, then (5) is not increased.

Clearly these three facts imply (4) for all $e \in E_I$. We do not prove these properties here due to the space limitation. Refer to Thorup [18] or full version of this paper for their proofs. We would like to note that an important fact for having (iii) is that hypertrees are bases of the hypergraphic matroid. □

By applying Lemma 5 to some subgraph of H, we obtain the next lemma. We skip the proof due to the space limitation.

Lemma 6. *Let $0 < \epsilon \le 1/2$, and $\mathscr{T}^{\mathrm{g}}$ be a set of hypertrees of H constructed by Algorithm 2 with $|\mathscr{T}^{\mathrm{g}}| = t \ge 3\gamma m \ln(\gamma mn/\epsilon)/\epsilon^3$. Then*

$$|\mathscr{T}^{\mathrm{g}}_e| \le \frac{1+\epsilon}{1-\epsilon} \cdot \frac{\alpha^*(\mathscr{T}^*_e)}{\alpha^*(\mathscr{T}^*)} \cdot |\mathscr{T}^{\mathrm{g}}| + 1$$

holds for each $e \in E_H$.

Lemma 6 proves Theorem 5 as follows. Let $\mathcal{V}^{\mathrm{opt}}$ stand for a minimum k-partition of H. Lemma 6 shows that

$$\sum_{T \in \mathscr{T}^{\mathrm{g}}} |\delta(\mathcal{V}^{\mathrm{opt}}) \cap T| = \sum_{e \in \delta(\mathcal{V}^{\mathrm{opt}})} |\mathscr{T}^{\mathrm{g}}_e|$$
$$\le \sum_{e \in \delta(\mathcal{V}^{\mathrm{opt}})} \left(t\frac{1+\epsilon}{1-\epsilon} \cdot \frac{\alpha^*(\mathscr{T}^*_e)}{\alpha^*(\mathscr{T}^*)} + 1\right) \le t\frac{1+\epsilon}{1-\epsilon} \cdot \frac{\sum_{e \in \delta(\mathcal{V}^{\mathrm{opt}})} \alpha^*(\mathscr{T}^*_e)}{\alpha^*(\mathscr{T}^*)} + \gamma m.$$

In the last of Section 4, we have observed that

$$\frac{\sum_{e \in \delta(\mathcal{V}^{\mathrm{opt}})} \alpha^*(\mathscr{T}^*_e)}{\alpha^*(\mathscr{T}^*)} = \frac{\sum_{T \in \mathscr{T}^*} \alpha^*(T)|\delta(\mathcal{V}^{\mathrm{opt}}) \cap T|}{\alpha^*(\mathscr{T}^*)} < \gamma k - 2.$$

These mean that

$$\frac{\sum_{T \in \mathscr{T}^{\mathrm{g}}} |\delta(\mathcal{V}^{\mathrm{opt}}) \cap T|}{t} < \frac{1+\epsilon}{1-\epsilon}(\gamma k - 2) + \frac{\gamma m}{t}.$$

Recall that $t = 3\gamma m \ln(\gamma mn/\epsilon)/\epsilon^3$. Assume that $n, m \ge 2$. Then $t \ge 6\gamma m/\epsilon^3$, and hence the right-hand side of the above inequality is at most

$$\frac{1+\epsilon}{1-\epsilon}(\gamma k - 2) + \epsilon^3/6 = \gamma k - 2 + 2\gamma k\epsilon + \epsilon\left(\frac{2k\gamma\epsilon - 4}{1-\epsilon} + \frac{\epsilon^2}{6}\right).$$

Setting ϵ to $1/(4k)$, the right-hand side is at most $\gamma k - 1$, which means that

$$\frac{\sum_{T \in \mathscr{T}^{\mathrm{g}}} |\delta(\mathcal{V}^{\mathrm{opt}}) \cap T|}{t} < \gamma k - 1.$$

This implies that $\mathscr{T}^{\mathrm{g}}$ contains a hypertree T such that $|\delta(\mathcal{V}^{\mathrm{opt}}) \cap T| < \gamma k - 1$. Moreover, $t = 3\gamma m \ln(\gamma mn/\epsilon)/\epsilon^3 = 24\gamma^4 mk^3 \ln(2\gamma^2 kmn)$. Therefore the proof has been completed.

6 Concluding Remarks

Our algorithm proposed in this paper is not polynomial if γ is not fixed. A reason for this fact is that a bound obtained in Theorem 2 depends on γ. If we can remove γ from the bound, we have a polynomial algorithm even if γ is not fixed. However there exists a hypergraph in which every hypertree in a recursively maximum hypertree packing shares $\gamma + k - 3$ hyperedges with any minimum k-cuts.

Define a set V of vertices as $\{v_1, v_2, \ldots, v_n\}$. We identify i with $i + n$ for each $i \in \{1, 2, \ldots, n\}$ for convenience. We also define a set E of hyperedges as $\{e_1, e_2, \ldots, e_{n-1}\}$ where each hyperedge e_i is defined as $\{v_i, v_{i+1}, \ldots, v_{i+\gamma}\}$. Let $H = (V, E)$ be the hypergraph with uniform hyperedge capacity. Figure 1 illustrates H. The intervals represented by gray lines in the figure denote the hyperedges of H.

Observe that H is a hypertree. Hence a recursively maximum hypertree packing of H consists of a single hypertree H. On the other hand, any minimum k-partition of H is represented by $\{\{v_i\}, \{v_{i+1}\}, \{v_{i+2}\}, \ldots, \{v_{i+k-2}\}, V - \cup_{j=i}^{i+k-2}\{v_j\}\}$ with some $i \in \{1, 2, \ldots, n\}$ because each hyperedge in H contains vertices of consecutive indices. Since less number of hyperedges contain v_j, $j \in \{n, 1, \ldots, \gamma - 1\}$, $i \leq n < \gamma - 1 \leq i + k - 2$ holds. Hence any minimum k-cut of H contains $\gamma + k - 3$ hyperedges (A minimum k-partition is represented by the dotted lines in Figure 1). Therefore, any hypertree in a recursively maximum hypertree packing of H and any minimum k-cut shares $\gamma + k - 3$ hyperedges.

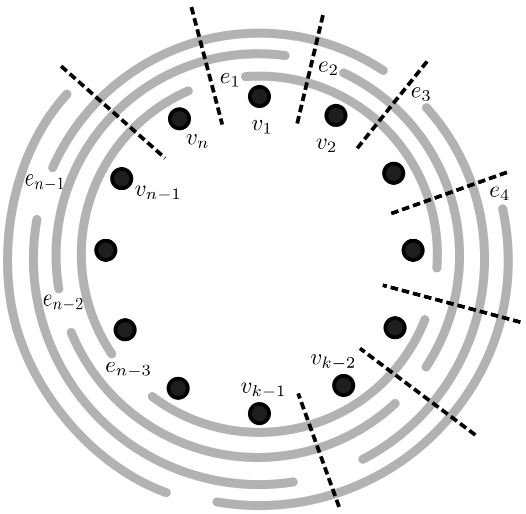


Fig. 1. A hypergraph H in which every hypertree in a recursively maximum hypertree packing shares $\gamma + k - 3$ hyperedges with any minimum k-cuts. Dotted lines represent a minimum k-partition $\{\{v_n\}, \{v_1\}, \ldots, \{v_{k-2}\}, \{v_{k-1}, v_k, \ldots, v_{n-1}\}\}$.

References

1. Chekuri, C., Korula, N.: Personal Communication (2010)
2. Frank, A., Király, T., Kriesell, M.: On decomposing a hypergraph into k connected sub-hypergraphs. Discrete Applied Mathematics 131(2), 373–383 (2003)
3. Garg, N., Vazirani, V.V., Yannakakis, M.: Multiway cuts in node weighted graphs. Journal of Algorithms 50, 49–61 (2004)
4. Gasieniec, L., Jansson, J., Lingas, A., Óstlin, A.: On the complexity of constructing evolutionary trees. Journal of Combinatorial Optimization 3, 183–197 (1999)
5. Goldberg, A.V., Tarjan, R.E.: A new approach to the maximum flow problem. Journal of the ACM 35, 921–940 (1988)
6. Goldschmidt, O., Hochbaum, D.: A polynomial algorithm for the k-cut problem for fixed k. Mathematics of Operations Research 19, 24–37 (1994)
7. Kamidoi, Y., Yoshida, N., Nagamochi, H.: A deterministic algorithm for finding all minimum k-way cuts. SIAM Journal on Computing 36, 1329–1341 (2006)
8. Karger, D.R., Stein, C.: A new approach to the minimum cut problem. Journal of the ACM 43, 601–640 (1996)
9. Klimmek, R., Wagner, F.: A simple hypergraph min cut algorithm. Internal Report B 96-02, Bericht FU Berlin Fachbereich Mathematik und Informatik (1995)
10. Lawler, E.L.: Cutsets and partitions of hypergraphs. Networks 3, 275–285 (1973)
11. Lorea, M.: Hypergraphes et matroides. Cahiers Centre Etudes Rech. Oper. 17, 289–291 (1975)
12. Lovász, L.: A generalization of König's theorem. Acta. Math. Acad. Sci. Hungar. 21, 443–446 (1970)
13. Mak, W.-K., Wong, D.F.: A fast hypergraph min-cut algorithm for circuit partitioning. Integ. VLSI J. 30, 1–11 (2000)
14. Nagamochi, H.: Algorithms for the minimum partitioning problems in graphs. IEICE Transactions on Information and Systems J86-D-1, 53–68 (2003)
15. Okumoto, K., Fukunaga, T., Nagamochi, H.: Divide-and-conquer algorithms for partitioning hypergraphs and submodular systems. In: Dong, Y., Du, D.-Z., Ibarra, O. (eds.) ISAAC 2009. LNCS, vol. 5878, pp. 55–64. Springer, Heidelberg (2009)
16. Stoer, M., Wagner, F.: A simple min-cut algorithm. J. the ACM 44, 585–591 (1997)
17. Thorup, M.: Fully-dynamic min-cut. Combinatorica 27, 91–127 (2007)
18. Thorup, M.: Minimum k-way cuts via deterministic greedy tree packing. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, pp. 159–166 (2008)
19. Xiao, M.: Finding minimum 3-way cuts in hypergraphs. In: Agrawal, M., Du, D.-Z., Duan, Z., Li, A. (eds.) TAMC 2008. LNCS, vol. 4978, pp. 270–281. Springer, Heidelberg (2008)
20. Xiao, M.: An improved divide-and-conquer algorithm for finding all minimum k-way cuts. In: Hong, S.-H., Nagamochi, H., Fukunaga, T. (eds.) ISAAC 2008. LNCS, vol. 5369, pp. 208–219. Springer, Heidelberg (2008)
21. Zhao, L., Nagamochi, H., Ibaraki, T.: A unified framework for approximating multiway partition problems. In: Eades, P., Takaoka, T. (eds.) ISAAC 2001. LNCS, vol. 2223, pp. 682–694. Springer, Heidelberg (2001)

Eigenvalue Techniques for Convex Objective, Nonconvex Optimization Problems

Daniel Bienstock

APAM and IEOR Depts., Columbia University

Abstract. A fundamental difficulty when dealing with a minimization problem given by a nonlinear, convex objective function over a nonconvex feasible region, is that even if we can efficiently optimize over the convex hull of the feasible region, the optimum will likely lie in the interior of a high dimensional face, "far away" from any feasible point, yielding weak bounds. We present theory and implementation for an approach that relies on (a) the S-lemma, a major tool in convex analysis, (b) efficient projection of quadratics to lower dimensional hyperplanes, and (c) efficient computation of combinatorial bounds for the minimum distance from a given point to the feasible set, in the case of several significant optimization problems. On very large examples, we obtain significant lower bound improvements at a small computational cost[1].

1 Introduction

We consider problems with the general form

$$
\begin{aligned}
(\boldsymbol{\mathcal{F}}):\qquad & \bar{F} := \min\ F(x), && (1)\\
s.t.\qquad & x \in \mathcal{P}, && (2)\\
& x \in \mathcal{K}, \quad \text{where} && (3)
\end{aligned}
$$

- F is a convex quadratic, i.e. $F(x) = x^T M x + v^T x$ (with $M \succeq 0$ and $v \in \mathcal{R}^n$). Extensions to the non-quadratic case are possible (see below).
- $\mathcal{P} \subseteq \mathcal{R}^n$ is a convex set over which we can efficiently optimize F,
- $\mathcal{K} \subseteq \mathcal{R}^n$ is a non-convex set with "special structure".

We assume that a given convex relaxation of the set described by (2), (3) is under consideration. A fundamental difficulty is likely to be encountered: because of the convexity of F, the optimum solution to the relaxation will frequently be attained in the interior of a high-dimensional face of the relaxation, and far from the set $\mathcal{K}$. Thus, the lower bound proved by the relaxation will often be weak. What is more, if one were to rely on branch-and-cut the proved lower bound may improve little if at all when n is large, even after massive amounts of branching.

This *stalling* of the lower bounding procedure is commonly encountered in practice and constitutes a significant challenge, the primary subject of our study.

[1] Work partially funded by ONR Award N000140910327 and a gift from BHP Billiton Ltd.

After obtaining the solution x^* to the given relaxation for problem $\mathcal{F}$, our methods will use techniques of convex analysis, of eigenvalue optimization, and combinatorial estimations, in order to quickly obtain a valid lower bound on $\bar{F}$ which is strictly larger than $F(x^*)$. Our methods apply if F is not quadratic but there is a convex quadratic G such that $F(x) - F(x^*) \geq G(x - x^*)$ for all feasible x.

We will describe an important class of problems where our method, applied to a "cheap" but weak formulation, produces bounds comparable to or better than those obtained by much more sophisticated formulations, and at a small fraction of the computational cost.

Cardinality constrained optimization problems. Here, for some integer $0 < K \leq n$, $\mathcal{K} = \{ x \in \mathcal{R}^n \, : \, \|x\|_0 \leq K \}$, where the zero-norm $\|v\|_0$ of a vector v is used to denote the number of nonzeros in v. This constraint arises in portfolio optimization (see e.g. [2]) but modern applications involving this constraint arise in statistics, machine learning [13], and, especially, in engineering and biology [19]. Problems related to *compressive sensing* have an explicit cardinality constraint (see www.dsp.ece.rice.edu/cs for material). Also see [7].

The simplest canonical example of problem $\mathcal{F}$ is as follows:

$$\bar{F} = \min \; F(x), \tag{4}$$

$$s.t. \quad \sum_j x_j = 1, \quad x \geq 0, \tag{5}$$

$$\|x\|_0 \leq K. \tag{6}$$

This problem is strongly NP-hard, and it does arise in practice, exactly as stated.

In spite of its difficulty, this example already incorporates the fundamental difficulty alluded to above: clearly, $conv\left\{ x \in \mathcal{R}^n_+ \, : \, \sum_j x_j = 1, \, \|x\|_0 \leq K \right\} = \left\{ x \in \mathcal{R}^n_+ \, : \, \sum_j x_j = 1 \right\}$. In other words, from a convexity standpoint the cardinality constraint disappears. Moreover, if the quadratic in F is positive definite and dominates the linear term, then the minimizer of F over the unit simplex will be an interior point (all coordinates positive) whereas $K \ll n$ in practice.

A second relevant example is given my a system of multiple (linear) disjunctions, such as split-cuts [6]. Also see [3], [4]. Details in full paper. To the extent that disjunctive sets are a general-purpose technique for formulating combinatorial constraints, the methods in this paper apply to a wide variety of optimization problems.

1.1 Techniques

Our methods embody two primary techniques:

(a) The S-lemma (see [20], also [1], [5], [15]). Let $f, \, g \, : \, \mathcal{R}^n \to \mathcal{R}$ be quadratic functions and suppose there exists $\bar{x} \in \mathcal{R}^N$ such that $g(\bar{x}) > 0$. Then

$$f(x) \geq 0 \quad \text{whenever} \quad g(x) \geq 0$$

if and only if there exists $\mu \geq 0$ such that $(f \, - \, \mu g)(x) \, \geq \, 0$ for all x.

Remark: here, a "quadratic" may contain a linear as well as a constant term. The S-lemma can be used as an algorithmic framework for minimizing a quadratic subject to a quadratic constraint. Let p, q be quadratic functions and let α, β be reals. Then

$$\min\{p(x) \,:\, q(x) \geq \beta\} \,\geq\, \alpha, \text{ iff } \exists\, \mu \geq 0 \;\text{ s.t. } p(x) \,-\, \alpha \,-\, \mu q(x) + \mu\beta \geq 0 \;\; \forall x \tag{7}$$

In other words, the minimization problem in (7) can be approached as a simultaneous search for two reals α and $\mu \geq 0$, with α largest possible such that the last inequality in (7) holds. The S-lemma is significant in that it provides a good characterization (i.e. polynomial-time) for a usually non-convex optimization problem. See [14], [16], [17], [18], [21] and the references therein, in particular regarding the connection to the trust-region subproblem.

(b) Consider a given nonconvex set $\mathcal{K}$. We will assume, as a primitive, that (possibly *after an appropriate change of coordinates*, given a point $\hat{x} \in \mathcal{R}^n$, we can efficiently compute a strong (combinatorial) lower bound for the Euclidean distance between $\hat{x}$ and the nearest point in $\mathcal{P} \cap \mathcal{K}$. We will show that this is indeed the case for the cardinality constrained case (see Section 1.4. Roughly, we exploit the "structure" of a set $\mathcal{K}$ of interest. We will denote by $D(\hat{x})$ our lower bound on the minimum distance from $\hat{x}$ to $\mathcal{P} \cap \mathcal{K}$.

Using (a) and (b), we can compute a lower bound for $\bar{F}$:

Simple Template

S.1 Compute an optimal solution x^* to the given relaxation to problem $\mathcal{F}$.
S.2 Obtain the quantity $D(x^*)$.
S.3 Apply the S-lemma as in (7), using $F(x)$ for $p(x)$, and (the exterior of) the ball centered at x^* with radius $D(x^*)$ for $q(x) - \beta$.

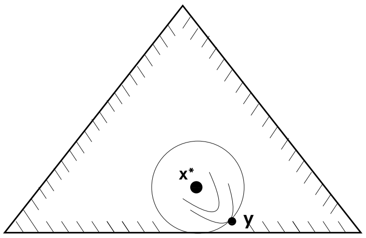


Fig. 1. A simple case

For a simple application of this template, consider Figure 1. This shows an instance of problem (4)-(6), with $n = 3$ and $K = 2$ where all coordinates of x^* are positive. The figure also assumes that $D(x^*)$ is exact – it equals the minimum distance from x^* to the feasible region. If we minimize $F(x)$, subject to being on the *exterior* of this ball the optimum will be attained at y. Thus, $F(y) \leq \bar{F}$; we have $F(y) = F(x^*) + \tilde{\lambda}_1 R^2$, where R is the radius of the ball and $\tilde{\lambda}_1$ is the minimum eigenvalue of the *restriction* of $F(x)$ to the unit simplex.

Now consider the example in Figure 2, corresponding to the case of a single disjunction. Here, x^F is the optimizer of $F(x)$ over the affine hull of the set $\mathcal{P}$. A straightforward application of the S-Lemma will yield as a lower bound (on $\bar{F}$) the value $F(y)$, which is weak – weaker, in fact, than $F(x^*)$. The problem is caused by the fact that x^F is not in the relative interior of the convex hull of the feasible region. In summary, a direct use of our template will not work.

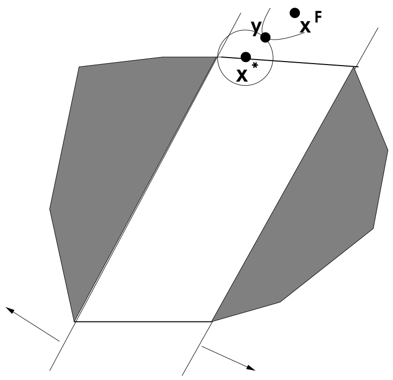


Fig. 2. The simple template fails

1.2 Adapting the Template

In order to correct the general form of the difficulty depicted by Figure 2 we would need to solve a problem of the form:

$$\mathcal{V} \ := \ \min\left\{F(x) \ : \ x - x^* \in \mathcal{C}, \ (x-x^*)^T(x-x^*) \geq \delta^2\right\} \tag{8}$$

where $\delta > 0$, and $\mathcal{C}$ is the cone of feasible directions (for $\mathcal{P}$) at x^*. We can view this as a 'cone constrained' version of the problem addressed by the S-Lemma. Clearly, $F(x^*) \leq \mathcal{V} \leq \bar{F}$ with the first inequality in general strict. If we are dealing with polyhedral sets, (8) becomes (after some renaming):

$$\mathcal{V} = \min\left\{F(\omega) \ : \ C\omega \geq 0, \ \omega^T\omega \geq \delta^2\right\} \tag{9}$$

where C is an appropriate matrix. However, we have (proof in full paper):

Theorem 1. *Problem (9) is strongly NP-hard.* ■

We stress that the NP-hardness result is *not* simply a consequence of the non-convex constraint in (9) – without the *linear* constraints, the problem becomes polynomially solvable (i.e., it is handled by the S-lemma, see the references).

To bypass this negative result, we will adopt a different approach. We assume that there is a positive-definite quadratic function $q(x)$ such that for any $y \in \mathcal{R}^n$, in polynomial time we can produce a (strong, combinatorial) lower bound $D^2_{min}(y,q)$ on the quantity

$$\min\{q(y-x) \ : \ x \in \mathcal{P} \cap \mathcal{K}\}.$$

In Section 1.4 we will address how to produce the quadratic $q(x)$ and the value $D^2(y,q)$ when $\mathcal{K}$ is defined by a cardinality constraint.

Let $c = \nabla F(x^*)$ (other choices for c discussed in full paper). Note that for any $x \in \mathcal{P} \cap \mathcal{K}$, $c^T(x - x^*) \geq 0$. For $\alpha \geq 0$, let $p^\alpha = x^* + \alpha c$, and let H^α be the hyperplane through p^α orthogonal to c. Finally, define

$$V(\alpha) := \min\{F(x) \,:\, q(x - p^\alpha) \geq D^2(p^\alpha, q),\; x \in H^\alpha\}, \tag{10}$$

and let y^α attain the minimum. Note: computing $V(\alpha)$ entails an application of the S-lemma, “restricted” to H^α. See Figure 3. Clearly, $V(\alpha) \leq \bar{F}$. Then

- Suppose $\alpha = 0$, i.e. $p^\alpha = x^*$. Then x^* is a minimizer of $F(x)$ subject to $x \in H^0$. Thus $V(0) > F(x^*)$ when F is positive-definite.
- Suppose $\alpha > 0$. Since $c^T(y^\alpha - x^*) > 0$, by convexity $V(\alpha) = F(y) > F(x^*)$.

Thus, $F(x^*) \leq \inf_{\alpha \geq 0} V(\alpha) \leq \bar{F}$; the first inequality being strict in the positive-definite case. [It can be shown that the “inf” is a “min”]. Each value $V(\alpha)$ incorporates combinatorial information (through the quantity $D^2(p^\alpha, q)$) and thus the computation of $\min_{\alpha \geq 0} V(\alpha)$ cannot be obtained through direct convex optimization techniques. As a counterpoint to Theorem 1 one can prove (using the notation in eq. (8)):

Theorem 2. *In (9), if C has one row and $q(x) = \sum_j x_j^2$, $\mathcal{V} \leq \inf_{\alpha \geq 0} V(\alpha)$.* ■

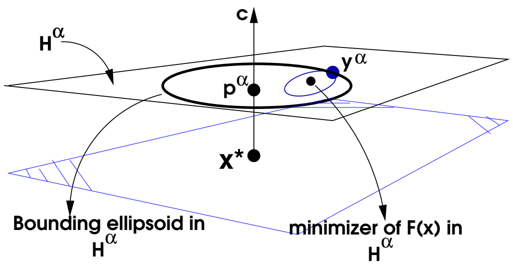

c
H^α
y^α
p^α
x^*
Bounding ellipsoid in H^α
minimizer of F(x) in H^α

Fig. 3. A better paradigm

In order to develop a computationally practicable approach that uses these observations, let $0 = \alpha^{(0)} < \alpha^{(1)} < \ldots < \alpha^{(J)}$, such that for any $x \in P \cap \mathcal{K}$, $c^T(x - x^*) \leq \alpha^{(J)} \|c\|_2^2$. Then:

Updated Template

1. For $0 \leq i < J$, compute a value $\tilde{V}(i) \;\leq\; \min\{\, V(\alpha) \,:\, \alpha^{(i)} \leq \alpha \leq \alpha^{(i+1)} \,\}$.
2. Output $\min_{0 \leq i < J} \tilde{V}(i)$.

The idea here is that if (for all i) $\alpha^{(i+1)} - \alpha^{(i)}$ is small then $V(\alpha^{(i)}) \approx V(\alpha^{(i+1)})$. Thus the quantity output in (2) will closely approximate $\min_{\alpha \geq 0} V(\alpha)$.

In our implementation, we compute $\tilde{V}(i)$ by appropriately interpolating between $V(\alpha^{(i)})$ and $V(\alpha^{(i+1)})$ (details, full paper). Thus our approach reduces to computing quantities of the form $V(\alpha)$. We need a fast procedure for this task (since J may be large). Considering eq. (10) we see that this involves an application of the S-lemma, "restricted" to the hyperplane H^α. An efficient realization of this idea, which allows for additional leveraging of combinatorial information, is obtained by computing the *projection* of the quadratic $F(x)$ to H^α. This is the subject of the next section.

1.3 Projecting a Quadratic

Let $M = Q\Lambda Q^T$ be a $n \times n$ matrix. Here the columns of Q are the eigenvectors of M and $\Lambda = \mathrm{diag}\{\lambda_1, \ldots, \lambda_n\}$ where the λ_i are the eigenvalues of M. We assume $\lambda_1 \leq \ldots \leq \lambda_n$. Let $c \neq 0$ be an arbitrary vector, and denote $H = \left\{x \in \mathcal{R}^n : c^T x = 0\right\}$, and let P be the projection matrix onto H. In this section we describe an efficient algorithm for computing an eigenvalue-eigenvector decomposition of the "projected quadratic" PMP. Note that if $x \in H$, $x^T PMPx = x^T Mx$. The vector c could be dense (is dense in important cases) and Q could also be dense.

In [8] (also see Section 12.6 of [9] and references therein) the following "inverse" problem is considered. Suppose $\lambda_i < \lambda_{i+1}$ $(1 \leq i < n)$ and that for $1 \leq i \leq n-1$ we are given a number $\tilde{\lambda}_i$ with $\lambda_i < \tilde{\lambda}_i < \lambda_{i+1}$. Then we want to find c (and hence, P) such that the $\tilde{\lambda}_i$ are the nonzero eigenvalues of PMP. Our approach reverse engineers that of [8], and extends it so as to handle the case where the λ_i are not distinct.

Returning to our problem, clearly c is an eigenvector of PMP (corresponding to eigenvalue 0). The remaining eigenvalues $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{n-1}$ are known to satisfy $\lambda_1 \leq \tilde{\lambda}_1 \leq \lambda_2 \leq \tilde{\lambda}_2 \leq \ldots \leq \lambda_{n-1} \leq \tilde{\lambda}_{n-1} \leq \lambda_n$.

Definition 1. *An eigenvector q of M is called* acute *if $q^T c \neq 0$. An eigenvalue λ of M is called acute if at least one eigenvector corresponding to λ is acute.*

In (e.2) below we will use the convention $0/0 = 0$.

Lemma 1. *Let $\alpha_1 < \alpha_2 < \ldots < \alpha_q$ be the acute eigenvalues of M. Write $d = Q^T c$. Then, for $1 \leq i \leq q-1$,*

(e.1) The equation $\sum_{j=1}^n \frac{d_j^2}{\lambda_j - \lambda} = 0$ has a unique solution $\hat{\lambda}_i$ in (α_i, α_{i+1}).
(e.2) Let $w^i = Q(\lambda - \hat{\lambda}_i I)^{-1} d$. Then $c^T w^i = 0$ and $PMPw^i = \hat{\lambda}_i w^i$.

Proof (e.2) Note that the expression in (e.1), evaluated at $\hat{\lambda}_i$, can be written as

$$0 = d^T(\Lambda - \hat{\lambda}_i I)^{-1} d = c^T Q(\Lambda - \hat{\lambda}_i I)^{-1} Q^T c = c, \tag{11}$$

Thus, we have that w^i is a linear combination of acute eigenvectors of M and that $w^i \in H$, and therefore $Pw^i = w^i$. So

$$(M - \hat{\lambda}_i I)w^i = Q(\Lambda - \hat{\lambda}_i I)Q^T w^i = QQ^T c,$$

and therefore

$$PMPw^i = PMw^i = \hat{\lambda}_i Pw^i = \hat{\lambda}_i w^i,$$

as desired. ■

Altogether, Lemma 1 produces $q-1$ eigenvalue/eigenvector pairs of PMP. The vector in (e.2) should not be explicitly computed; rather the factorized form in (e.2) will suffice. The root to the equation in (e.1) can be quickly obtained using numerical methods (such as golden section search) since the expression in (e.1) is monotonely increasing in (α_i, α_{i+1}) (it may also be possible to adapt the basic trust-region algorithm [14], which addresses a similar but not identical problem).

Lemma 2. *Let α be an eigenvalue of M, V^α the set of columns of Q with eigenvalue α, and $\mathcal{A} = \mathcal{A}(\alpha)$ denote the acute members of V^α. If $|\mathcal{A}| > 0$, then we can construct $|\mathcal{A}| - 1$ eigenvectors of PMP corresponding to eigenvalue α, each of which is a linear combination of elements of $\mathcal{A}$ and is orthogonal to c.*

Proof: Write $m = |\mathcal{A}|$, and let H be the $m \times m$ Householder matrix [9] corresponding to $d_{\mathcal{A}}$, i.e. H is a symmetric matrix with $H^2 = I_m$ such that

$$Hd_{\mathcal{A}} = (\|d_{\mathcal{A}}\|_2, 0, ..., 0)^T \in \mathcal{R}^m.$$

Let $Q_{\mathcal{A}}$ be the $n \times m$ submatrix of Q consisting of the columns corresponding to $\mathcal{A}$, and define

$$W = Q_{\mathcal{A}} H. \tag{12}$$

Then $c^T W = d_{\mathcal{A}}^T H = (\|d_{\mathcal{A}}\|_2, 0, ..., 0)$. In other words, the columns of the submatrix $\hat{W}$ consisting of the last $m-1$ columns of W are orthogonal to c. Denoting by $\hat{H}$ the submatrix of H consisting of the last $m-1$ columns of H, we therefore have

$$\hat{W} = Q_{\mathcal{A}} \hat{H}, \text{ and}$$

$$PMP\hat{W} = PQ\Lambda Q^T \hat{W} = PQ\Lambda Q^T Q_{\mathcal{A}} \hat{H} = \alpha PQ_{\mathcal{A}} \hat{H} = \alpha \hat{W}.$$

Finally, $\hat{W}^T \hat{W} = \hat{H}^T \hat{H} = I_m$, as desired. ■

Now suppose that

$$\alpha_1 < \alpha_2 < \ldots < \alpha_q$$

denote the distinct acute eigenvalues of M (possibly $q = 0$). Let p denote the number of columns of Q which are perpendicular eigenvectors. Writing $m_i = |\mathcal{A}(\alpha_i)| > 0$ for $1 \le i \le q$, we have that

$$n = \sum_{i=1}^{q} m_i \quad + \quad p.$$

(p.1) Using Lemma 1 we obtain $q-1$ eigenvectors of PMP, each of which is a linear combination of acute eigenvectors among Q. Any eigenvalue of PMP constructed in this manner is different from all acute eigenvalues of M.

(p.2) Using Lemma 2 we obtain, for each i, a set of $m_i - 1$ eigenvectors of PMP, orthogonal to c and with eigenvalue α_i, each of which is a linear combination of elements of $\mathcal{A}(\alpha_i)$. In total, we obtain $n-q-p$ eigenvectors of PMP.

(p.3) Let p denote the number of perpendicular vectors among Q. Any such vector v (with eigenvalue λ, say) by definition satisfies $PMPv = PMv = \lambda Pv = \lambda v$.

By construction, all eigenvectors of PMP constructed as per (p.1) and (p.2) are distinct. Those arising in (p.3) are different from those in (p.1) and (p.2) since no column of Q is a linear combination of other columns of Q. Thus, altogether, (p.1)-(p.3) account for $n-1$ distinct eigenvectors of PMP, all of them orthogonal to c, by construction. Finally, the vector c itself is an eigenvector of PMP, corresponding to eigenvalue 0.

To conclude this section, we note that it is straightforward to iterate the procedure in this section, so as to project a quadratic to hyperplanes of dimension less than $n-1$. More details will be provided in the full paper.

1.4 Combinatorial Bounds on Distance Functions

Here we take up the problem of computing strong lower bounds on the Euclidean distance from a point to the set $\mathcal{P} \cap \mathcal{K}$. In this abstract we will focus on the cardinality constrained problem, but results of a similar flavor hold for the case of disjunctive sets.

Let $a \in \mathcal{R}^n$, $b \in \mathcal{R}$, $K < n$ be a positive integer, and $\omega \in \mathcal{R}^n$. Consider the problem

$$D^2_{min}(\omega, a) := \min \left\{ \sum_{j=1}^{n} (x_j - \omega_j)^2, \quad : \quad a^T x = b \quad \text{and} \quad \|x\|_0 \le K \right\}. \tag{13}$$

Clearly, the sum of smallest $n-K$ values ω_j^2 constitutes a ("naive") lower bound for problem (13). But it is straightforward to show that an exact solution to (13) is obtained by choosing $S \subseteq \{1, \ldots, n\}$ with $|S| \le K$, so as to minimize

$$\frac{(b - \sum_{j \in S} a_j \omega_j)^2}{\sum_{j \in S} a_j^2} \quad + \quad \sum_{j \notin S} \omega_j^2. \tag{14}$$

[We use the convention that $0/0 = 0$.] Empirically, the naive bound mentioned above is very weak since the first term in (14) is typically at least an order of magnitude larger than the second; and it is the bound, rather than the set S itself, that matters.

Suppose $a_j = 1$ for all j. It can be shown, using (14), that the optimal set S has the following structure: $S = P \cup N$, where $|P| + |N| \le K$, and P consists of the indices of the $|P|$ smallest nonnegative ω_j (resp., N consists of the indices of the $|N|$ smallest $|\omega_j|$ with $\omega_j < 0$). The optimal S can be computed in $O(K)$

time, after sorting the ω_j. When $\omega \geq 0$ or $\omega \leq 0$ we recover the naive procedure mentioned above (though again we stress that the first term in (14) dominates). In general, however, we have:

Theorem 3. *(a) It is NP-hard to compute $D^2_{\min}(\omega, a)$. (b) Let $0 < \epsilon < 1$. We can compute a vector $\hat{x}$ with $\sum_j a_j \hat{x}_j = b$ and $\|\hat{x}\|_0 \leq K$, and such that*

$$\sum_{j=1}^{n} (\hat{x}_j - \omega_j)^2 \ \leq\ (1+\epsilon) D^2_{min}(\omega, a),$$

in time polynomial in n, ϵ^{-1}, and the number of bits needed to represent ω and a. ■

In our current implementation we have not used the algorithm in part (b) of the Lemma, though we certainly plan to evaluate this option. Instead, we proceed as follows. Assume $a_j \neq 0$ for all j. Rather than solving problem (13), instead we consider

$$\min \left\{ \sum_{j=1}^{n} a_j^2 (x_j - \omega_j)^2 \quad : \quad a^T x = b \ \text{ and } \ \|x\|_0 \leq K \right\}.$$

Writing $\bar{\omega}_j = a_j \omega_j$ (for all j), this becomes $\min \left\{ \sum_{j=1}^{n} (x_j - \bar{\omega}_j)^2 \ : \ \sum_j x_j = b \ \textit{ and } \ \|x\|_0 \leq K \right\}$, which as noted above can be efficiently solved.

1.5 Application of the S-Lemma

Let $M = Q \Lambda Q^T \succeq 0$ be a matrix given by its eigenvector factorization. Let H be a hyperplane through the origin, $\hat{x} \in H$, $v \in \mathcal{R}^n$, $\delta_j > 0$ for $1 \leq j \leq n$, $\beta > 0$, and $v \in \mathcal{R}^n$. Here we solve the problem

$$\min \quad x^T M x + v^T x, \quad \text{subject to} \quad \sum_{i=1}^{n} \delta_i (x_i - \hat{x}_i)^2 \ \geq\ \beta, \quad \text{and} \quad x \ \in\ H. \tag{15}$$

By rescaling, translating, and appropriately changing notation, the problem becomes:

$$\min \quad x^T M x + v^T x, \quad \text{subject to} \quad \sum_{i=1}^{n} x_i^2 \ \geq\ \beta, \quad \text{and} \quad x \in H. \tag{16}$$

Let P be the $n \times n$ matrix corresponding to projection onto H. Using Section 1.3 we can produce a representation of PMP as $\tilde{Q} \tilde{\Lambda} \tilde{Q}^T$, where the the n^{th} eigenvector $\tilde{q}_n$ is orthogonal to H, and $\tilde{\lambda}_1 = \min_{i<n}\{\tilde{\lambda}_i\}$. Thus, problem (16) becomes, for appropriately defined $\tilde{v}$,

$$\Gamma := \min \sum_{j=1}^{n-1} \tilde{\lambda}_j y_j^2 + 2\tilde{v}^T y, \quad \text{subject to} \quad \sum_{j=1}^{n-1} y_j^2 \ \geq\ \beta. \tag{17}$$

Using the S-lemma, we have that $\Gamma \geq \gamma$, iff there exists $\mu \geq 0$ s.t.

$$\sum_{j=1}^{n-1} \tilde{\lambda}_j y_j^2 + 2\tilde{v}^T y - \gamma \quad - \quad \mu \left(\sum_{j=1}^{n-1} y_j^2 \ - \ \beta \right) \ \geq \ 0 \quad \forall\, y \in \mathcal{R}^{n-1}. \tag{18}$$

Using some linear algebra, this is equivalent to

$$\Gamma = \max \left\{ \mu\beta - \sum_{i=1}^{n-1} \frac{\tilde{v}_i^2}{\tilde{\lambda}_i - \mu} \ : \ 0 \leq \mu < \tilde{\lambda}_1 \right\}. \tag{19}$$

This is a simple task, since in $[0, \tilde{\lambda}_1)$ the objective in (19) is concave in μ.

Remarks:
(1) Our updated template in Section 1.2 requires the solution of multiple problems of the form (19) but just *one* computation of $\tilde{Q}$ and $\tilde{\Lambda}$.
(2) Consider any integer $1 \leq p < n-1$. When $\mu < \tilde{\lambda}_1$, the expression maximized in (19) is lower bounded by $\mu\beta - \sum_{i=1}^{p} \frac{\tilde{v}_i^2}{\tilde{\lambda}_i - \mu} - \frac{\sum_{i=p+1}^{n-1} \tilde{v}_i^2}{\lambda_{p+1} - \mu}$. This, and related facts, yield an approximate version of our approach which only asks for the first p elements of the eigenspace of PMP (and M).

Capturing the second eigenvalue. We see that $\Gamma < \tilde{\lambda}_1 \beta$ (and frequently this bound is close). In experiments, the solution y^* to (16) often "cheats" in that y_1^* is close to zero. We can then improve on our procedure if the second projected eigenvalue, $\tilde{\lambda}_2$, is significantly larger than $\tilde{\lambda}_1$. Assuming that is the case, pick a value θ with $y_1^{*2}/\beta < \theta < 1$.

(a) If we assert that $y_1^2 \geq \theta\beta$ then we may be able to strengthen the constraint in (15) to $\sum_{i=1}^{n} \delta_i (x_i - \hat{x}_i)^2 \geq \gamma$, where $\gamma = \gamma(\theta) > \beta$. See Lemma 3 below. So the assertion amounts to applying the S-lemma, but using γ in place of β.
(b) Otherwise, we have that $\sum_{i=2}^{n-1} y_i^2 \geq (1-\theta)\beta$. In this case, instead of the right-hand side of (19), we will have

$$\max \left\{ \mu(1-\theta)\beta - \sum_{i=2}^{n-1} \frac{\tilde{v}_i^2}{\tilde{\lambda}_i - \mu} \ : \ 0 \leq \mu \leq \tilde{\lambda}_2 \right\}. \tag{20}$$

The minimum of the quantities obtained in **(a)** and **(b)** yields a valid lower bound on Γ; we can evaluate several candidates for θ and choose the strongest bound. When $\tilde{\lambda}_2$ is significantly larger than $\tilde{\lambda}_1$ we often obtain an improvement over the basic approach as in Section 1.5.

Note: the approach in this section constitutes a form of *branching* and in our testing has proved very useful when $\lambda_2 > \lambda_1$. It is, intrinsically, a combinatorial approach, and thus not easily reproducible using convexity arguments alone.

To complete this section, we point out that the construction of the quantities $\gamma(\beta)$ above is based on the following observation:

Lemma 3. *Let $v \in \mathcal{R}^n$, let $H \subset \mathcal{R}^n$ be a $(n-1)$-dimensional hyperplane with $v \notin H$, and w be the projection of v onto H. Let $G \subset \mathcal{R}^n$ be a $\leq (n-1)$-dimensional hyperplane, K the intersection of G with the closed half-space of $\mathcal{R}^n$ separated from v by H, $D_{w,G}$ the distance from w to G and $\bar{D}_{v,K}$ the distance from v to K ($\bar{D}_{v,K} = +\infty$ if $K = \emptyset$). Then $\bar{D}^2_{v,K} \geq \|v - w\|^2 + D^2_{w,G}$.* ■

2 Computational Experiments

We consider problems $\min\{\, x^T M x + v^T x \,:\, \sum_j x_j = 1,\, x \geq 0,\, \|x\|_0 \leq K \,\}$. The matrix $M \succeq 0$ is given in its eigenvector/eigenvalue factorization $Q \Lambda Q^T$. To stress-test our linear algebra routines, we construct Q as the product of random rotations: as the number of rotations increases, so does the number of nonzeroes in Q, and the overall "complexity" of M. We ran our procedure after computing the solution to the (diagonalized) "weak" formulation

$$\min\{\, y^T \Lambda y + v^T x \,:\, Q^T x = y,\, \sum_j x_j = 1,\, x \geq 0\}.$$

We also ran the (again, diagonalized) *perspective formulation* [10], [12], a strong conic formulation (here, λ_{min} is the minimum λ_i):

$$\begin{aligned} \min\quad & \lambda_{min} \sum_j w_j \;+\; \sum_j (\lambda_j - \lambda_{min}) y_j^2 \\ s.t.\quad & Q^T x \;=\; y,\;\; \sum_j x_j = 1 \\ & x_j^2 - w_j z_j \;\leq\; 0, \quad 0 \;\leq\; z_j \;\leq\; 1 \quad \forall\, j, \\ & \sum_j z_j \;\leq\; K, \quad x_j \;\leq\; z_j \;\; \forall\, j, \;\; x, w \;\in\; \mathcal{R}^n_+. \end{aligned} \tag{21}$$

We used the Updated Template given above, with $c = \nabla(x^*)$ and with the $\alpha^{(i)}$ quantities set according to the following method: (a) $J = 100$, and (b) $\alpha^{(J)} = \text{argmax}\{\alpha \geq 0 \,:\, H^\alpha \cap S^{n-1} \neq \emptyset\}$ (S^{n-1} is the unit simplex). The improvement technique involving the second eigenvalue was applied in all cases.

For the experiments in Tables 1 and 2, we used Cplex 12.1 on a single core of a 2.66 GHz quad-core Xeon machine with 16 GB of RAM, which was never exceeded. In the tests in Table 1, $n = 2443$ and the eigenvalues are from a finance application. Q is the product of 5000 random rotations, resulting in 142712 nonzeros in Q.

Here, **rQMIP** refers to the weak formulation, **PRSP** to the perspective formulation, and **SLE** to the approach in this paper. "LB" is the **lower bound** produced by a given approach, and "sec" is the CPU time in seconds. The second eigenvalue technique proved quite effective in all these tests.

In Table 2 we consider examples with $n = 10000$ and random Λ. In the table, **Nonz** indicates the number of nonzeroes in Q; as this number increases the quadratic becomes less diagonal dominant.

Table 1. Examples with few nonzeroes

K	**rQMIP** LB	**PRSP** LB	**SLE** LB	**rQMIP** sec	**PRSP** sec	**SLE** sec
200	0.031	0.0379	0.0382	14.02	59.30	5.3
100	0.031	0.0466	0.0482	13.98	114.86	5.8
90	0.031	0.0485	0.0507	14.08	103.38	5.9
80	0.031	0.0509	0.0537	14.02	105.02	6.2
70	0.031	0.0540	0.0574	13.95	100.06	6.2
60	0.031	0.0581	0.0624	15.64	111.63	6.4
50	0.031	0.0638	0.0696	13.98	110.78	6.4
40	0.031	0.0725	0.0801	14.03	104.48	6.5
30	0.031	0.0869	0.0958	14.17	104.48	6.8
20	0.031	0.1157	0.1299	15.69	38.13	6.9
10	0.031	0.2020	0.2380	14.05	43.77	7.2

Table 2. Larger examples

Nonz in Q	**rQMIP** LB	**PRSP** LB	**SLE** LB	**rQMIP** sec	**PRSP** sec	**SLE** sec
5.3e+05	2.483e-03	1.209e-02	1.060e-02	332	961.95	57.69
3.7e+06	2.588e-03	1.235e-02	1.113e-02	705	2299.75	57.55
1.8e+07	2.671e-03	1.248e-02	1.117e-02	2.4e+03	1.3e+04	57.69
5.3e+07	2.781e-03	1.263e-02	1.120e-02	1.1e+04	8.5e+04	58.44
8.3e+07	2.758e-03	1.262e-02	1.211e-02	2.3e+04	1.4e+05	57.38

As in Table 1, **SLE** and **PRSP** provide similar improvements over **rQMIP** (which is clearly extremely weak). **SLE** proves uniformly fast. In the examples in Table 2, the smallest ten (or so) eigenvalues are approximately equal, with larger values after that. As a result, on these examples our second eigenvalue technique proved ineffective.

Also note that the perspective formulation quickly proves impractical. A cutting-plane procedure that replaces the conic constraints in (21) with (outer approximating) linear inequalities is outlined in [10], [12] and tested on random problems with $n \leq 400$. The procedure begins by solving **rQMIP** and then iteratively adds the inequalities; or it could simply solve a formulation consisting of **rQMIP**, augmented with a set of pre-computed inequalities. In our experiments with this linearized approximation, we found that (a) it can provide a very good lower bound to the conic perspective formulation, (b) it can run significantly faster than the full conic formulation, but, (c) it proves significantly *slower* than **rQMIP**, and, in particular, still significantly slower than the combination

Table 3. Detailed analysis of $K = 70$ case of Table 1

algorithm	**threads**	**nodes**	**wall-clock time (sec)**	**LB**	**UB**
QPMIP					
mip emph 3	4	10000	41685 (16.67 sec/node)	0.0314	0.241
PRSP-MIP					
mip emph 2	16	14000	39550 (90.4 sec/node)	0	0.8265
mip emph 3	16	7000	19817 (45.30 sec/node)	0	0.8099
LPRSP-MIP					
mip emph 0	4	39000	109333 (11.21 sec/node)	0.0554 root 0.0540	0.305
mip emph 1	16	7000	36751 (84.04 sec/node)	0.0542 root 0.0540	0.412
mip emph 2	16	16000	35222 (35.22 sec/node)	0.0543 root 0.0540	0.309
mip emph 3	16	6000	57469 (153 sec/node)	0.0564 root 0.0540	0.702

of **rQMIP** and **SLE**. A strengthened version of the perspective formulation, which requires the solution of a semidefinite program, is given in [11].

Note that the perspective formulation itself is an example of the paradigm that we consider in this paper: a convex formulation for a nonconvex problem with a convex objective; thus we expect it to exhibit stalling. Table 3 concerns the $K = 70$ case of Table 1, using Cplex 12.1 on a dual 2.93 GHz quad-core "Nehalem" machine with 48GB of physical memory. [This CPU uses "hyperthreading" and Cplex 12.1, as a default, will use 16 threads]. On this machine, rQMIP requires 4.35 seconds (using Cplex) and our method, 3.54 seconds (to prove a lower bound of 0.0574).

In this table, **QPMIP** is the weak formulation, **PRSP-MIP** is the perspective formulation, and **LPRSP-MIP** is the linearized perspective version (constraint (21) is linearized at $x_j = 1/K$ which proved better than other choices). [Comment: Cplex 12.1 states a lower bound of 0 for **PRSP-MIP**]. "wall-clock time" indicates the observed running time. The estimates of CPU time per node were computed using the formula (wall-clock time)*threads/nodes.

References

1. Ben-Tal, A., Nemirovsky, A.: Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications. MPS-SIAM Series on Optimization. SIAM, Philadelphia (2001)
2. Bienstock, D.: Computational study of a family of mixed-integer quadratic programming problems. Math. Programming 74, 121–140 (1996)
3. Bienstock, D., Zuckerberg, M.: Subset algebra lift algorithms for 0-1 integer programming. SIAM J. Optimization 105, 9–27 (2006)
4. Bienstock, D., McClosky, B.:Tightening simple mixed-integer sets with guaranteed bounds (submitted 2008)
5. Boyd, S., El Ghaoui, L., Feron, E., Balakrishnan, V.: Linear matrix inequalities in system and control theory. SIAM, Philadelphia (1994)
6. Cook, W., Kannan, R., Schrijver, A.: Chv'atal closures for mixed integer programs. Math. Programming 47, 155–174 (1990)
7. De Farias, I., Johnson, E., Nemhauser, G.: A polyhedral study of the cardinality constrained knapsack problem. Math. Programming 95, 71–90 (2003)
8. Golub, G.H.: Some modified matrix eigenvalue problems. SIAM Review 15, 318–334 (1973)
9. Golub, G.H., van Loan, C.: Matrix Computations. Johns Hopkins University Press, Baltimore (1996)
10. Frangioni, A., Gentile, C.: Perspective cuts for a class of convex 0-1 mixed integer programs. Mathematical Programming 106, 225–236 (2006)
11. Frangioni, A., Gentile, C.: SDP Diagonalizations and Perspective Cuts for a Class of Nonseparable MIQP. Oper. Research Letters 35, 181–185 (2007)
12. Günlük, O., Linderoth, J.: Perspective Relaxation of Mixed Integer Nonlinear Programs with Indicator Variables. In: Lodi, A., Panconesi, A., Rinaldi, G. (eds.) IPCO 2008. LNCS, vol. 5035, pp. 1–16. Springer, Heidelberg (2008)
13. Moghaddam, B., Weiss, Y., Avidan, S.: Generalized spectral bounds for sparse LDA. In: Proc. 23rd Int. Conf. on Machine Learning, pp. 641–648 (2006)
14. Moré, J.J., Sorensen, D.C.: Computing a trust region step. SIAM J. Sci. Stat. Comput. 4, 553–572 (1983)
15. Pólik, I., Terlaky, T.: A survey of the S-lemma. SIAM Review 49, 371–418 (2007)
16. Rendl, F., Wolkowicz, H.: A semidefinite framework for trust region subproblems with applications to large scale minimization. Math. Program 77, 273–299 (1997)
17. Stern, R.J., Wolkowicz, H.: Indefinite trust region subproblems and nonsymmetric eigenvalue perturbations. SIAM J. Optim. 5, 286–313 (1995)
18. Sturm, J., Zhang, S.: On cones of nonnegative quadratic functions. Mathematics of Operations Research 28, 246–267 (2003)
19. Miller, W., Wright, S., Zhang, Y., Schuster, S., Hayes, V.: Optimization methods for selecting founder individuals for captive breeding or reintroduction of endangered species (2009) (manuscript)
20. Yakubovich, V.A.: S-procedure in nonlinear control theory, vol. 1, pp. 62–77. Vestnik Leningrad University (1971)
21. Ye, Y., Zhang, S.: New results on quadratic minimization. SIAM J. Optim. 14, 245–267 (2003)

Restricted b-Matchings in Degree-Bounded Graphs

Kristóf Bérczi and László A. Végh[⋆]

MTA-ELTE Egerváry Research Group (EGRES),
Department of Operations Research, Eötvös Loránd University,
Pázmány Péter sétány 1/C, Budapest, Hungary, H-1117
{berkri,veghal}@cs.elte.hu

Abstract. We present a min-max formula and a polynomial time algorithm for a slight generalization of the following problem: in a simple undirected graph in which the degree of each node is at most $t+1$, find a maximum t-matching containing no member of a list $\mathcal{K}$ of forbidden $K_{t,t}$ and K_{t+1} subgraphs. An analogous problem for bipartite graphs without degree bounds was solved by Makai [15], while the special case of finding a maximum square-free 2-matching in a subcubic graph was solved in [1].

Keywords: square-free, $K_{t,t}$-free, K_{t+1}-free, b-matching, subcubic graph.

1 Introduction

Let $G = (V, E)$ be an undirected graph and let $b : V \to \mathbb{Z}_+$ be an upper bound on the nodes. An edge set $F \subseteq E$ is called a *b-matching* if $d_F(v)$, the number of edges in F incident to v, is at most $b(v)$ for each node v. (This is often called *simple* b-matching in the literature.) For some integer $t \geq 2$, by a *t-matching* we mean a b-matching with $b(v) = t$ for every $v \in V$. Let $\mathcal{K}$ be a set consisting of $K_{t,t}$'s, complete bipartite subgraphs of G on two colour classes of size t, and K_{t+1}'s, complete subgraphs of G on $t+1$ nodes. The node-set and the edge-set of a subgraph $K \in \mathcal{K}$ are denoted by V_K and E_K, respectively. By a *$\mathcal{K}$-free b-matching* we mean a b-matching not containing any member of $\mathcal{K}$. In this paper, we give a min-max formula on the size of $\mathcal{K}$-free b-matchings and a polynomial time algorithm for finding one with maximum size (that is, a $\mathcal{K}$-free b-matching $F \subseteq E$ with maximum cardinality) under the assumptions that for any $K \in \mathcal{K}$ and any node v of K,

$$V_K \text{ spans no parallel edges} \tag{1}$$
$$b(v) = t \tag{2}$$
$$d_G(v) \leq t + 1. \tag{3}$$

Note that this is a generalization of the problem mentioned in the abstract. The most important special case of $\mathcal{K}$-free b-matching is to find a maximum C_3-free

[⋆] Supported by the Hungarian National Foundation for Scientific Research (OTKA) grant K60802.

or C_4-free 2-matching in a graph where C_k stands for a cycle of length k. The motivation for these problems is twofold. On the one hand, a natural relaxation of the Hamiltonian cycle problem is to find a $C_{\leq k}$-free 2-factor, that is, a 2-factor containing no cycle of length at most k. Cornuéjols and Pulleyblank [2] showed this problem to be NP-complete for $k \geq 5$. In his Ph.D. thesis [6], Hartvigsen proposed a solution for the case $k = 3$. Hence the remaining question is to find a maximum $C_{\leq 4}$-free 2-matching, and another natural question is to find a maximum C_4-free 2-matching (possibly containing triangles).

The other motivation comes from connectivity-augmentation, that is, when one would like to make a graph $G = (V, E)$ k-node-connected by the addition of a minimum number of new edges. It is easy to see that for $k = n - 2$ ($n = |V|$) this problem is equivalent to finding a maximum matching in the complement graph of G. For $k = n - 3$ the problem is equivalent to finding a maximum C_4-free 2-matching.

The C_4-free 2-matching problem admits two natural generalizations. The first one is $K_{t,t}$-free t-matchings considered in this paper, while the second is t-matchings containing no complete bipartite graph $K_{a,b}$ with $a + b = t + 2$. This latter problem is equivalent to connectivity augmentation for $k = n - t - 1$. The complexity of connectivity augmentation for general k is yet open, while connectivity augmentation by one, that is, when the input graph is already $(k - 1)$-connected was recently solved in [20] (this corresponds to the case when the graph contains no $K_{a,b}$ with $a + b = t + 3$, in particular, $d(v) \leq t + 1$).

The weighted versions of these problems are also of interest. The weighted $C_{\leq k}$-free 2-matching problem asks for a $C_{\leq k}$-free 2-matching with maximum weight for a weight function defined on the edge set. For $k = 2$ the problem is just to find a 2-matching with maximum weight, while Király showed [11] that the problem is NP-complete for $k = 4$ even in bipartite graphs with $0 - 1$ weights on the edges. The case of $k = 3$ in general graphs is still open. Hartvigsen and Li [9], and recently Kobayashi [12] gave polynomial-time algorithms for the weighted C_3-free 2-matching problem in subcubic graphs with an arbitrary weight function.

Let us now consider the special case of C_4-free 2-matchings in bipartite graphs. This problem was solved by Hartvigsen [7,8] and Király [10]. A generalization of the problem to maximum $K_{t,t}$-free t-matchings in bipartite graphs was given by Frank [3] who observed that this is a special case of covering positively crossing supermodular functions on set pairs, solved by Frank and Jordán in [4]. Makai [15] generalized Frank's theorem for the case when a list $\mathcal{K}$ of forbidden $K_{t,t}$'s is given (that is, a t-matching may contain $K_{t,t}$'s not in $\mathcal{K}$.) He gave a min-max formula based on a polyhedral description for the minimum cost version for node-induced cost functions. Pap [16] gave a further generalization of the maximum cardinality version for excluded complete bipartite subgraphs and developed a simple, purely combinatorial algorithm. For node induced cost functions, such an algorithm was given by Takazawa [19] for $K_{t,t}$-free t-matching.

Much less is known when the underlying graph is not assumed to be bipartite and finding a maximum C_4-free 2-matching is still open. The special case when the graph is subcubic was solved by the first author and Kobayashi [1]. In terms of connectivity augmentation, the equivalent problem is augmenting an $(n-4)$-connected graph to $(n-3)$ connected. Our theorem is a generalization of this result.

It is worth mentioning that the polynomial solvability of the above problems seems to show a strong connection with jump systems. In [18], Szabó proved that for a list $\mathcal{K}$ of forbidden $K_{t,t}$ and K_{t+1} subgraphs the degree sequences of $\mathcal{K}$-free t-matchings form a jump system in any graph. Concerning bipartite graphs, Kobayashi and Takazawa showed [14] that the degree sequences of $C_{\leq k}$-free 2-matchings do not always form a jump system for $k \geq 6$. These results are consistent with the polynomial solvability of the $C_{\leq k}$-free 2-matching problem, even when restricting it to bipartite graphs. Similar results are known about even factors due to [13]. Although Szabó's result suggests that finding a maximum $\mathcal{K}$-free t-matching should be solvable in polynomial time, the problem is still open.

Among our assumptions, (1) and (2) may be considered as natural ones as they hold for the maximum $K_{t,t}$-free t-matching problem in a simple graph. We exclude parallel edges on the node sets of members of $\mathcal{K}$ in order to avoid having two different $K_{t,t}$'s on the same two colour classes or two K_{t+1}'s on the same ground set. However, the degree bound (3) is a restrictive assumption and dissipates essential difficulties. Our proof strongly relies on this and the theorem cannot be straightforwardly generalized, as it can be shown by using the example in Chapter 6 of [20].

The proof and algorithm use the contraction technique of [11], [16] and [1]. Our contribution on the one hand is the extension of this technique for $t \geq 2$ and forbidding K_{t+1}'s as well, while on the other hand the argument is significantly simpler than the argument in [1].

Throughout the paper we use the following notation. For an undirected graph $G = (V, E)$, the set of edges induced by $X \subseteq V$ is denoted by $E[X]$. For disjoint subsets X, Y of V, $E[X, Y]$ denotes the set of edges between X and Y. The set of nodes in $V - X$ adjacent to X by some edge from $F \subseteq E$ is denoted by $\Gamma_F(X)$. We let $d_F(v)$ denote the number of edges in $F \subseteq E$ incident to v, where loops in G are counted twice, while $d_F(X, Y)$ stands for the number of edges going between disjoint subsets X and Y. For a node $v \in V$, we sometimes abbreviate the set $\{v\}$ by v, e.g. $d_F(v, X)$ is the number of edges between v and X. For a set $X \subseteq V$, let $h_F(X) = \sum_{v \in X} d_F(v)$, the sum of the number of edges incident to X and twice the number of edges spanned by X. We use $b(U) = \sum_{v \in U} b(v)$ for a function $b : V \to \mathbb{Z}_+$ and a set $U \subseteq V$.

Let $\mathcal{K}$ be the list of forbidden $K_{t,t}$ and K_{t+1} subgraphs. For disjoint subsets X, Y of V we denote by $\mathcal{K}[X]$ and $\mathcal{K}[X, Y]$ the members of $\mathcal{K}$ contained in X and having edges only between X and Y, respectively. That is, $\mathcal{K}[X, Y]$ stands for forbidden $K_{t,t}$'s whose colour classes are subsets of X and Y. Recall that V_K and E_K denote the node-set and edge-set of the forbidden graph $K \in \mathcal{K}$, respectively.

The rest of the paper is organized as follows. In Section 2 we formalize the theorem and prove the trivial max $\leq$ min direction. Two shrinking operations are introduced in Section 3, and Section 4 contains the proof of the max $\geq$ min direction. Finally, the algorithm is presented in Section 5.

2 Main Theorem

Before stating our theorem, let us recall the well-known min-max formula on the maximum size of a b-matching (see e.g. [17, Vol A, p. 562.]).

Theorem 1 (Maximum size of a b-matching). *Let $G = (V, E)$ be a graph with an upper bound $b : V \to \mathbb{Z}_+$. The maximum size of a b-matching is equal to the minimum value of*

$$b(U) + |E[W]| + \sum_{T} \left\lfloor \tfrac{1}{2}(b(T) + |E[T, W]|) \right\rfloor \tag{4}$$

where U and W are disjoint subsets of V, and T ranges over the connected components of $G - U - W$.

Let us now formulate our theorem. There are minor technical difficulties when $t = 2$ that do not occur for larger t. In order to make both the formulation and the proof simpler it is worth introducing the following definitions. We refer to forbidden $K_{2,2}$ and K_3 subgraphs as squares and triangles, respectively.

Definition 2. *For $t = 2$, we call a complete subgraph on four nodes* square-full *if it contains three forbidden squares.*

Note that, by assumption (3), every square-full subgraph is a connected component of G. We denote the number of square-full components of G by $S(G)$ for $t = 2$, and define $S(G) = 0$ for $t > 2$. It is easy to see that a $\mathcal{K}$-free b-matching contains at most three edges from each square-full component of G. The following definition will be used in the proof of the theorem.

Definition 3. *For $t = 2$, a forbidden triangle is called* square-covered *if its node set is contained in the node set of a forbidden square, otherwise* uncovered*.*

The theorem is as follows.

Theorem 4. *Let $G = (V, E)$ be a graph with an upper bound $b : V \to \mathbb{Z}_+$ and $\mathcal{K}$ be a list of forbidden $K_{t,t}$ and K_{t+1} subgraphs of G so that (1), (2) and (3) hold. Then the maximum size of a $\mathcal{K}$-free b-matching is equal to the minimum value of*

$$b(U) + |E[W]| - |\dot{\mathcal{K}}[W]| + \sum_{T \in \mathcal{P}} \left\lfloor \tfrac{1}{2}(b(T) + |E[T, W]| - |\dot{\mathcal{K}}[T, W]|) \right\rfloor - S(G) \tag{5}$$

where U and W are disjoint subsets of V, $\mathcal{P}$ is a partition of the connected components of $G - U - W$ and $\dot{\mathcal{K}} \subseteq \mathcal{K}$ is a collection of node-disjoint forbidden subgraphs.

For fixed $U, W, \mathcal{P}$ and $\dot{\mathcal{K}}$ the value of (5) is denoted by $\tau(U, W, \mathcal{P}, \dot{\mathcal{K}})$. It is easy to see that the contribution of a square-full component to (5) is always 3 and a maximum $\mathcal{K}$-free b-matching contains exactly 3 of its edges. Hence we may count these components of G separately, so the following theorem immediately implies the general one.

Theorem 5. *Let $G = (V, E)$ be a graph with an upper bound $b : V \to \mathbb{Z}_+$ and $\mathcal{K}$ be a list of forbidden $K_{t,t}$ and K_{t+1} subgraphs of G so that (1), (2) and (3) hold. Furthermore, if $t = 2$, assume that G has no square-full component. Then the maximum size of a $\mathcal{K}$-free b-matching is equal to the minimum value of*

$$b(U) + |E[W]| - |\dot{\mathcal{K}}[W]| + \sum_{T \in \mathcal{P}} \left\lfloor \tfrac{1}{2}(b(T) + |E[T, W]| - |\dot{\mathcal{K}}[T, W]|) \right\rfloor \tag{6}$$

where U and W are disjoint subsets of V, $\mathcal{P}$ is a partition of the connected components of $G - U - W$ and $\dot{\mathcal{K}} \subseteq \mathcal{K}$ is a collection of node-disjoint forbidden subgraphs.

Proof (of max $\leq$ min in Theorem 5). Let M be a $\mathcal{K}$-free b-matching. Then clearly $|M \cap (E[U] \cup E[U, V - U])| \leq b(U)$ and $|M \cap E[W]| \leq |E[W]| - |\dot{\mathcal{K}}[W]|$. Moreover, for each $T \in \mathcal{P}$ we have

$$\begin{aligned}
2 \cdot |M \cap (E[T] \cup E[T, W])| &= 2 \cdot |M \cap E[T]| + 2 \cdot |M \cap E[T, W]| \\
&\leq 2 \cdot |M \cap E[T]| + |M \cap E[T, W]| \\
&+ |E[T, W]| - |\dot{\mathcal{K}}[T, W]| \\
&\leq b(T) + |E[T, W]| - |\dot{\mathcal{K}}[T, W]|.
\end{aligned}$$

These together prove the inequality. □

3 Shrinking

In the proof of max $\geq$ min we use two shrinking operations to get rid of the $K_{t,t}$ and K_{t+1} subgraphs in $\mathcal{K}$.

Definition 6 (Shrinking a $K_{t,t}$ subgraph). *Let K be a $K_{t,t}$ subgraph of $G = (V, E)$ with colour classes K_A and K_B.* Shrinking *K in G consists of the following operations:*

- *identify the nodes in K_A, and denote the corresponding node by k_a,*
- *identify the nodes in K_B, and denote the corresponding node by k_b, and*
- *replace the edges between K_A and K_B with $t - 1$ parallel edges between k_a and k_b (we call the set of these edges a* shrunk bundle *between k_a and k_b).*

When identifying the nodes in K_A and K_B, the edges (and also loops) spanned by K_A and K_B are replaced by loops on k_a and k_b, respectively. Each edge

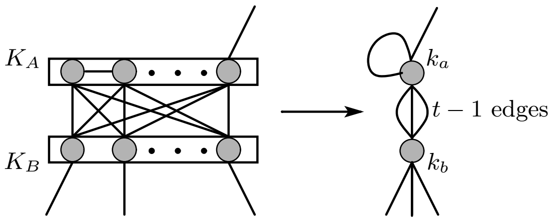


Fig. 1. Shrinking a $K_{t,t}$ subgraph

$e \in E - E_K$ is denoted by e again after shrinking a $K_{t,t}$ subgraph and is called the *image* of the original edge. By abuse of notation, for an edge set $F \subseteq E - E_K$, the corresponding subset of edges in the contracted graph is also denoted by F. Hence for an edge set $F \subseteq E - E_K$ we have $h_F(K_A) = d_F(k_a)$, $h_F(K_B) = d_F(k_b)$.

Definition 7 (Shrinking a K_{t+1} subgraph). *Let K be a K_{t+1} subgraph of $G = (V, E)$.* Shrinking *K in G consists of the following operations:*

- *identify the nodes in V_K, and denote the corresponding node by k,*
- *replace the edges in E_K by $\left\lfloor \frac{t+1}{2} \right\rfloor - 1$ loops on the new node k.*

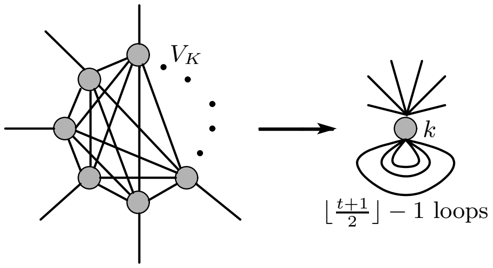


Fig. 2. Shrinking a K_{t+1} subgraph

Again, for an edge set $F \subseteq E - E_K$, the corresponding subset of edges in the contracted graph is also denoted by F.

We usually denote the graph obtained by applying one of the shrinking operations by $G^\circ = (V^\circ, E^\circ)$. Throughout the section, the graph G, the function b and the list $\mathcal{K}$ of forbidden subgraphs are supposed to satisfy the conditions of Theorem 5. It is easy to see, by using (3), that two members of $\mathcal{K}$ are edge-disjoint if and only if they are also node-disjoint, hence we simply call such pairs *disjoint*.

The following two lemmas give the connection between the maximum size of a $\mathcal{K}$-free b-matching in G and a $\mathcal{K}^\circ$-free b°-matching in G° where b° is a properly defined upper bound on V° and $\mathcal{K}^\circ$ is a list of forbidden sugraphs in the contracted graph.

Lemma 8. *Let $G^\circ = (V^\circ, E^\circ)$ be the graph obtained by shrinking a $K_{t,t}$ subgraph K. Let $\mathcal{K}^\circ$ be the set of forbidden subgraphs disjoint from K and define b° as $b^\circ(v) = b(v)$ for $v \in V - V_K$ and $b^\circ(k_a) = b^\circ(k_b) = t$. Then the difference between the maximum size of a $\mathcal{K}$-free b-matching in G and the maximum size of a $\mathcal{K}^\circ$-free b°-matching in G° is exactly $t^2 - t$.*

Lemma 9. *Let $G^\circ = (V^\circ, E^\circ)$ be the graph obtained by shrinking a K_{t+1} subgraph $K \in \mathcal{K}$ where K is uncovered if $t = 2$. Let $\mathcal{K}^\circ$ be the set of forbidden subgraphs disjoint from K and define b° as $b^\circ(v) = b(v)$ for $v \in V - V_K$, $b^\circ(k) = t$ if t is even and $b^\circ(k) = t+1$ if t is odd. Then the difference between the maximum size of a $\mathcal{K}$-free b-matching in G and the maximum size of a $\mathcal{K}^\circ$-free b°-matching in G° is exactly $\left\lfloor \frac{t^2}{2} \right\rfloor$.*

The proof of Lemma 8 is based on the following claim.

Claim 10. *Assume that $K \in \mathcal{K}$ is a $K_{t,t}$ subgraph with colour classes K_A and K_B and M' is a $\mathcal{K}$-free b-matching of $G - E_K$. Then M' can be extended to a $\mathcal{K}$-free b-matching M of G with $|M| = |M'| + t^2 - \max\{1, h_{M'}(K_A), h_{M'}(K_B)\}$.*

Proof. First we consider the case $t \geq 3$. Let P be a minimum size matching of K covering each node $v \in V_K$ with $d_{M'}(v) = 1$ (note that $d_{M'}(v) \leq 1$ for $v \in V_K$ as $d(v) \leq t+1$). If there is no such node, then let P consist of an arbitrary edge in E_K. We claim that $M = M' \cup (E_K - P)$ satisfies the above conditions. Indeed, M is a b-matching and $|M \cap E_K| = t^2 - \max\{1, h_{M'}(K_A), h_{M'}(K_B)\}$ clearly holds, so we only have to verify that it is also $\mathcal{K}$-free.

Assume that there is a forbidden $K_{t,t}$ subgraph K' in M with colour classes K'_A, K'_B. $E_{K'}$ must contain an edge $uv \in E_K \cap M$ with $u \in K'_A$ and $v \in K'_B$. By symmetry, we may assume that $u \in K_A$. As $b(u) = t$, $\Gamma_M(u) = K'_B$ and also $|\Gamma_M(u) \cap K_B| \geq t-1$. Hence $|K'_B \cap K_B| \geq t-1$. Consider a node $z \in K_A$. Since $d_M(z, K_B) \geq t-1$ and $t \geq 3$, we get $d_M(z, K'_B) > 0$, thus $K_A \subseteq \Gamma_M(K'_B)$. Because of $\Gamma_M(K'_B) = K'_A$, this gives $K_A = K'_A$. $K_B = K'_B$ follows similarly, giving a contradiction.

If there is a forbidden K_{t+1} subgraph K' in M, then $E_{K'}$ must contain an edge $uv \in E_K \cap M$, $u \in K_A$. As above, $|V_{K'} \cap K_B| \geq t-1$. Using $t \geq 3$ again, $K_A \subseteq \Gamma_M(V_{K'} \cap K_B) \subseteq V_{K'}$. But $K_A \subseteq V_{K'}$ is a contradiction since $t+1 = |V_{K'}| \geq |V_{K'} \cap K_A| + |V_{K'} \cap K_B| \geq t + t - 1 = 2t - 1 > t+1$.

Now let $t = 2$ and $K_A = \{v_1, v_3\}$, $K_B = \{v_2, v_4\}$. If $\max\{h_{M'}(K_A), h_{M'}(K_B)\} \leq 1$, then we may assume by symmetry that $d_{M'}(v_1) = d_{M'}(v_2) = 0$. Clearly, $M = M' \cup \{v_1v_2, v_1v_4, v_2v_3\}$ is a $\mathcal{K}$-free 2-matching. If $\max\{h_{M'}(K_A), h_{M'}(K_B)\} = 2$, we claim that at least one of $M_1 = M' \cup \{v_1v_2, v_3v_4\}$ and $M_2 = M' \cup \{v_1v_4, v_2v_3\}$ is $\mathcal{K}$-free. Assume M_1 contains a forbidden square or triangle K'; by symmetry assume it contains the edge v_1v_2. If K' contains v_3v_4 as well, then K' is the square $v_1v_3v_4v_2$. Otherwise, it consists of v_1v_2 and a path L of length 2 or 3 between v_1 and v_2, not containing v_3 and v_4. In the first case, the only forbidden subgraph possibly contained in M_2 is the square $v_1v_3v_2v_4$, implying that $\{v_1, v_2, v_3, v_4\}$ is a square-full component, a contradiction. In the latter case, it is easy to see that M_2 cannot contain a forbidden subgraph. □

Proof (of Lemma 8). First we show that if M is a $\mathcal{K}$-free b-matching in G then there is a $\mathcal{K}^\circ$-free b°-matching M° in G° with $|M^\circ| \geq |M| - (t^2 - t)$. Let $M' = M - E_K$. Clearly, $|M \cap E_K| \leq t^2 - \max\{1, h_{M'}(K_A), h_{M'}(K_B)\}$. In G°, let M° be the union of M' and $t - \max\{1, d_{M'}(k_a), d_{M'}(k_b)\}$ parallel edges from the shrunk bundle between k_a and k_b. Is is easy to see that M° is a $\mathcal{K}^\circ$-free b°-matching in G° with $|M^\circ| \geq |M| - (t^2 - t)$.

The proof is completed by showing that for an arbitrary $\mathcal{K}^\circ$-free b°-matching M° in G° there exists a $\mathcal{K}$-free b-matching M in G with $|M| \geq |M^\circ| + (t^2 - t)$. Let H denote the set of parallel edges in the shrunk bundle between k_a and k_b, and let $M' = M^\circ - H$. Now $|M^\circ \cap H| \leq t - \max\{1, d_{M'}(k_a), d_{M'}(k_b)\}$ and, by Claim 10, M' may be extended to a $\mathcal{K}$-free b-matching in G with $|M \cap E_K| = t^2 - \max\{1, h_{M'}(K_A), h_{M'}(K_B)\}$, that is

$$\begin{aligned}|M| = |M^\circ| - |M^\circ \cap H| + |M \cap E_K| &\geq |M^\circ| - (t - \max\{1, d_{M'}(k_a), d_{M'}(k_b)\}) \\ &+ (t^2 - \max\{1, h_{M'}(K_A), h_{M'}(K_B)\}) \geq |M^\circ| + (t^2 - t). \qquad \square\end{aligned}$$

Lemma 9 can be proved in a similar way by using the following claim.

Claim 11. *Assume that $K \in \mathcal{K}$ is a K_{t+1} subgraph and M' is a $\mathcal{K}$-free b-matching of $G - E_K$. If $t = 2$, then assume that K is uncovered. Then M' can be extended to obtain a $\mathcal{K}$-free b-matching M of G with $|M| = |M'| + \binom{t+1}{2} - \left\lceil \frac{\max\{1, h_{M'}(V_K)\}}{2} \right\rceil$.*

Proof. Let P be a minimum size subgraph of K covering each node $v \in V_K$ with $d_{M'}(v) = 1$ (so P is a matching or a matching and one more edge in E_K). If there is no such node, then let P consist of an arbitrary edge in E_K. For $t = 2$ and 3, we will choose P in a specific way, as given later in the proof. We show that $M = M' \cup (E_K - P)$ satisfies the above conditions. Indeed, M is a b-matching and $|M \cap E_K| = \binom{t+1}{2} - \left\lceil \frac{\max\{1, h_{M'}(K)\}}{2} \right\rceil$ clearly holds, so we only have to show that it is also $\mathcal{K}$-free.

Assume that there is a forbidden K_{t+1} subgraph K' in M. $E_{K'}$ must contain an edge $uv \in E_K \cap M$. By the minimal choice of P at least one of $|\Gamma_M(u) \cap V_K| \geq$

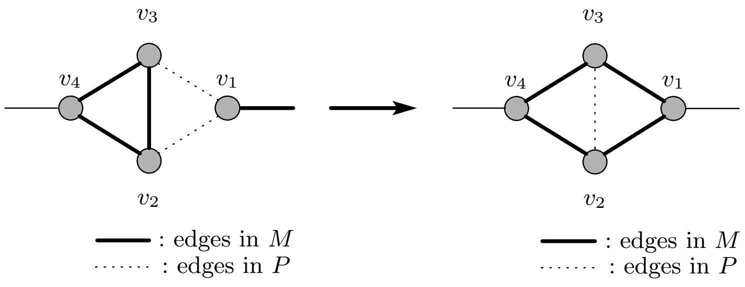


Fig. 3. Choice of P for $t = 2$ in the proof of Claim 11

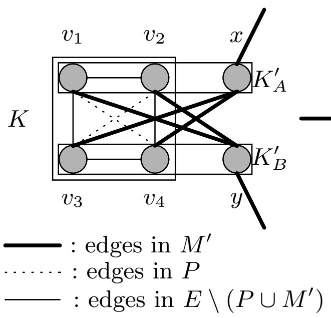


Fig. 4. Choice of P for $t = 3$ in the proof of Claim 11

$t-1$ and $|\Gamma_M(v) \cap V_K| \geq t-1$ is satisfied which implies $|V_{K'} \cap V_K| \geq t-1$. For $t \geq 3$ this immediately implies $V_K \subseteq \Gamma_M(V_{K'} \cap V_K) \subseteq V_{K'}$, a contradiction.

If $t = 2$, then $|V_{K'} \cap V_K| \geq 1$ does not imply $V_K \subseteq V_{K'}$ and an improper choice of P may enable M to contain a forbidden K_3. The only such case is when $h_{M'}(V_K) = 3$, $V_K = \{v_1, v_2, v_3\}$, $V_{K'} = \{v_2, v_3, v_4\}$, $v_2v_4, v_3v_4 \in M'$ and $P = \{v_1v_2, v_1v_3\}$ (Figure 3). In this case, we may leave the edge incident to v_1 from M' and then $P = \{v_2v_3\}$ is a good choice. Indeed, the only problem could be that $v_1v_2v_3v_4$ is a forbidden square, contradicting K being uncovered.

Otherwise $h_{M'}(V_K) \leq 2$ implies $|P| \leq 1$. Hence at least one of $|\Gamma_M(u) \cap V_K| = 2$ and $|\Gamma_M(v) \cap V_K| = 2$ is satisfied meaning $K' = K$, a contradiction again.

Now assume that there is a forbidden $K_{t,t}$ subgraph K' in M with colour classes K'_A, K'_B. The same argument gives a contradiction for $t \geq 4$. However, in case of $t = 3$, choosing P arbitrarily may enable M to contain a forbidden $K_{3,3}$ in the following single configuration: $V_K = \{v_1, v_2, v_3, v_4\}$, $K'_A = \{v_1, v_2, x\}$, $K'_B = \{v_3, v_4, y\}$, $xv_3, xv_4, yv_1, yv_2, xy \in M'$ and $P = \{v_1v_2, v_3v_4\}$ (Figure 4). In this case, $P = \{v_1v_4, v_2v_3\}$ is a good choice.

Finally, for $t = 2$ no forbidden square appears if $h_{M'}(K) \leq 2$ as otherwise K would be a square-covered triangle. If $h_{M'}(K) = 3$, then such a square K' may appear only if $V_K = \{v_1, v_2, v_3\}$, $V_{K'} = \{v_2, v_3, v_4, v_5\}$, $v_3v_4, v_4v_5, v_5v_2 \in M'$, $P = \{v_1v_2, v_1v_3\}$ ($v_1 \neq v_4, v_5$ as K is uncovered). In this case both $P = \{v_1v_2, v_2v_3\}$ and $P = \{v_1v_3, v_2v_3\}$ give a proper M (Figure 5). $\square$

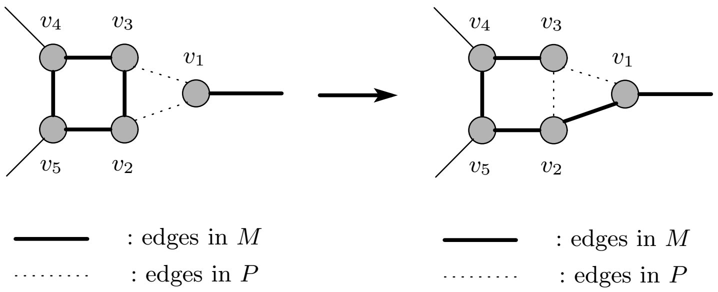


Fig. 5. Choice of P for $t = 2$ in the proof of Claim 11

Proof (of Lemma 9). First we show that if M is a $\mathcal{K}$-free b-matching in G then there is a $\mathcal{K}^\circ$-free b°-matching M° in G° with $|M^\circ| \geq |M| - \left\lfloor \frac{t^2}{2} \right\rfloor$. Let $M' = M - E_K$. Clearly, $|M \cap E_K| \leq \binom{t+1}{2} - \left\lceil \frac{\max\{1, h_{M'}(V_K)\}}{2} \right\rceil$. In G°, let M° be the union of M' and $\left\lfloor \frac{t - \max\{1, d_{M'}(k)\}}{2} \right\rfloor$ or $\left\lfloor \frac{t+1-\max\{1, d_{M'}(k)\}}{2} \right\rfloor$ loops on k depending on whether t is even or not, respectively. Is is easy to see that M° is a $\mathcal{K}^\circ$-free b°-matching in G° with $|M^\circ| \geq |M| - \left\lfloor \frac{t^2}{2} \right\rfloor$.

The proof is completed by showing that for an arbitrary $\mathcal{K}^\circ$-free b°-matching M° in G° there exists a $\mathcal{K}$-free b-matching M in G with $|M| \geq |M^\circ| + \left\lfloor \frac{t^2}{2} \right\rfloor$. Let H denote the set of loops on k obtained when shrinking K, and let $M' = M^\circ - H$. Now $|M^\circ \cap H| \leq \left\lfloor \frac{t - \max\{1, d_{M'}(k)\}}{2} \right\rfloor$ if t is even and $|M^\circ \cap H| \leq \left\lfloor \frac{t+1-\max\{1, d_{M'}(k)\}}{2} \right\rfloor$ if t is odd. By Claim 10, M' can be extended modified as to get a $\mathcal{K}$-free b-matching in G with $|M \cap E_K| = \binom{t+1}{2} - \left\lceil \frac{\max\{1, h_{M'}(V_K)\}}{2} \right\rceil$, that is

$$\begin{aligned}|M| = |M^\circ| - |M^\circ \cap H| + |M \cap E_K| &\geq |M^\circ| - \left\lfloor \frac{t - \max\{1, d_{M'}(k)\}}{2} \right\rfloor \\ + \binom{t+1}{2} - \left\lceil \frac{\max\{1, h_{M'}(V_K)\}}{2} \right\rceil &\geq |M^\circ| + \left\lfloor \frac{t^2}{2} \right\rfloor\end{aligned}$$

if t is even and

$$\begin{aligned}|M| = |M^\circ| - |M^\circ \cap H| + |M \cap E_K| &\geq |M^\circ| - \left\lfloor \frac{t+1-\max\{1, d_{M'}(k)\}}{2} \right\rfloor \\ + \binom{t+1}{2} - \left\lceil \frac{\max\{1, h_{M'}(V_K)\}}{2} \right\rceil &\geq |M^\circ| + \left\lfloor \frac{t^2}{2} \right\rfloor\end{aligned}$$

if t is odd. □

4 Proof of Theorem 5

We prove max $\geq$ min by induction on $|\mathcal{K}|$. For $\mathcal{K} = \emptyset$, this is simply a consequence of Theorem 1.

Assume now that $\mathcal{K} \neq \emptyset$ and let K be a forbidden subgraph such that K is uncovered if $t = 2$. Let $G^\circ = (V^\circ, E^\circ)$ denote the graph obtained by shrinking K, let b° be defined as in Lemma 8 or 9 depending on whether K is a $K_{t,t}$ or a K_{t+1}. We denote by $\mathcal{K}^\circ$ the list of forbidden subgraphs disjoint from K.

By induction, the maximum size of a $\mathcal{K}^\circ$-free b°-matching in G° is equal to the minimum value of $\tau(U^\circ, W^\circ, \mathcal{P}^\circ, \dot{\mathcal{K}}^\circ)$. Let us choose an optimal $U^\circ, W^\circ, \mathcal{P}^\circ, \dot{\mathcal{K}}^\circ$ so that $|U^\circ|$ is minimal. The following claim gives a useful property of U°.

Claim 12. *Assume that $v \in U$ is such that $d(v, W) + |\Gamma(v) \cap (V - W)| \leq b(v) + 1$. Then $\tau(U - v, W, \mathcal{P}', \dot{\mathcal{K}}) \leq \tau(U, W, \mathcal{P}, \dot{\mathcal{K}})$ where $\mathcal{P}'$ is obtained from $\mathcal{P}$ by replacing its members incident to v by their union plus v.*

Proof. By removing v from U, $b(U)$ decreases by $b(v)$. $|E[W]| - |\dot{\mathcal{K}}[W]|$ remains unchanged, while the bound on $d(v, W) + |\Gamma(v) \cap (V - W)|$ implies that the increment in the sum over the components of $G - U - W$ is at most $b(v)$. □

Case 1: K is a $K_{t,t}$ with colour classes K_A and K_B.
By Lemma 8, the difference between the maximum size of a $\mathcal{K}$-free b-matching in G and the maximum size of a $\mathcal{K}^\circ$-free b°-matching in G° is exactly $t^2 - t$. We will define $U, W, \mathcal{P}$ and $\dot{\mathcal{K}}$ such that

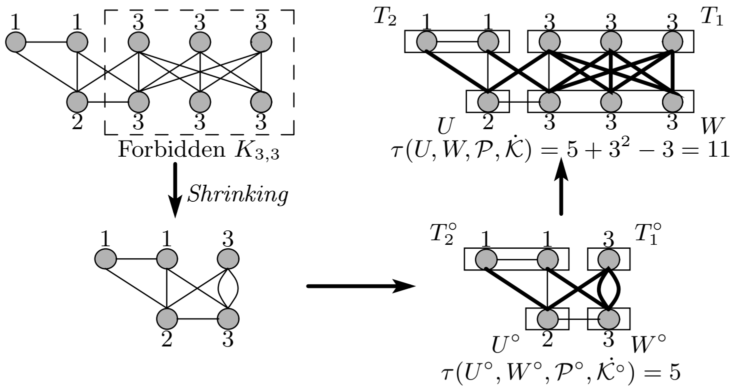
$$\tau(U, W, \mathcal{P}, \dot{\mathcal{K}}) = \tau(U^\circ, W^\circ, \mathcal{P}^\circ, \dot{\mathcal{K}}^\circ) + t^2 - t. \tag{7}$$

The shrinking replaces K_A and K_B by two nodes k_a and k_b with $t-1$ parallel edges between them. Let U, W and $\mathcal{P}$ denote the pre-images of $U^\circ, W^\circ, \mathcal{P}^\circ$ in G, respectively and let $\dot{\mathcal{K}} = \dot{\mathcal{K}}^\circ \cup \{K\}$. By (3), $d_{G^\circ - k_b}(k_a), d_{G^\circ - k_a}(k_b) \leq t$. Since $b^\circ(k_a) = b^\circ(k_b) = t$, Claim 12 and the minimal choice of $|U^\circ|$ implies that if $k_a \in U^\circ$, then $k_b \in W^\circ$.

Hence we have the following cases (T° denotes a member of $\mathcal{P}^\circ$). In each case we are only considering those terms in $\tau(U^\circ, W^\circ, \mathcal{P}^\circ, \dot{\mathcal{K}}^\circ)$ that change when taking $\tau(U, W, \mathcal{P}, \dot{\mathcal{K}})$ instead.

- $k_a \in U^\circ,\ k_b \in W^\circ$: $b(U) = b^\circ(U^\circ) + t^2 - t$.
- $k_a, k_b \in W^\circ$: $|E[W]| = |E^\circ[W^\circ]| + t^2 - t + 1$ and $|\dot{\mathcal{K}}[W]| = |\dot{\mathcal{K}}^\circ[W^\circ]| + 1$.
- $k_a \in W^\circ,\ k_b \in T^\circ$: $|E[T, W]| = |E^\circ[T^\circ, W^\circ]| + t^2 - t + 1$, $b(T) = b^\circ(T^\circ) + t^2 - t$ and $|\dot{\mathcal{K}}[T, W]| = |\dot{\mathcal{K}}^\circ[T^\circ, W^\circ]| + 1$ (see Figure 6 for an example).
- $k_a \in T^\circ,\ k_b \in W^\circ$: similar to the previous case.
- $k_a, k_b \in T^\circ$: $b(T) = b^\circ(T^\circ) + 2t^2 - 2t$.

(7) is satisfied in each of the above cases, hence we are done. Note that in the first and the last case we may leave out K from $\dot{\mathcal{K}}$ as it is not counted in any term.



1
1
3
3
3
2
3
3
3
Forbidden $K_{3,3}$
Shrinking
1
1
3
2
3
T_2
1
1
3
3
3
T_1
U
2
3
3
3
W
$\tau(U, W, \mathcal{P}, \dot{\mathcal{K}}) = 5 + 3^2 - 3 = 11$
T_2°
1
1
3
T_1°
U°
2
3
W°
$\tau(U^\circ, W^\circ, \mathcal{P}^\circ, \dot{\mathcal{K}}^\circ) = 5$

Fig. 6. Extending M°

Case 2: K is a K_{t+1}.
By Lemma 9, the difference between the maximum size of a $\mathcal{K}$-free b-matching in G and the maximum size of a $\mathcal{K}^\circ$-free b°-matching in G° is $\left\lfloor \frac{t^2}{2} \right\rfloor$. We show that for the pre-images U, W and $\mathcal{P}$ of U°, W° and $\mathcal{P}^\circ$ with $\dot{\mathcal{K}} = \dot{\mathcal{K}}^\circ \cup \{K\}$ satisfy

$$\tau(U, W, \mathcal{P}, \dot{\mathcal{K}}) = \tau(U^\circ, W^\circ, \mathcal{P}^\circ, \dot{\mathcal{K}}^\circ) + \left\lfloor \frac{t^2}{2} \right\rfloor . \tag{8}$$

After shrinking $K = (V_K, E_K)$ we get a new node k with $\lfloor \frac{t+1}{2} \rfloor - 1$ loops on it. (3) implies that there are at most $t + 1$ non-loop edges incident to k. Since $b^\circ(k) \geq t$, Claim 12 implies $k \notin U$. Hence we have the following two cases (T° denotes a member of $\mathcal{P}^\circ$).

- $k \in W^\circ$: $|E[W]| = |E^\circ[W^\circ]| + \binom{t+1}{2} - \lfloor \frac{t+1}{2} \rfloor + 1$ and $|\dot{\mathcal{K}}[W]| = |\dot{\mathcal{K}}^\circ[W^\circ]| + 1$.
- $k \in T^\circ$: $b(T) = b^\circ(T^\circ) + t^2$ if t is even and $b(T) = b^\circ(T^\circ) + t^2 - 1$ for an odd t.

(8) is satisfied in both cases, hence we are done. We may also leave out K from $\dot{\mathcal{K}}$ in the second case as it is not counted in any term. □

5 Algorithm

In this section we show how the proof of Theorem 5 immediately yields an algorithm for finding a maximum $\mathcal{K}$-free b-matching in strongly polynomial time. In such problems, an important question from an algorithmic point of view is how $\mathcal{K}$ is represented. For example, in the $\mathcal{K}$-free b-matching problem for bipartite graphs solved by Pap in [16], the set of excluded subgraphs may be exponentially large. Therefore Pap assumes that $\mathcal{K}$ is given by a membership oracle, that is, a subroutine is given for determining whether a given subgraph is a member of $\mathcal{K}$. However, with such an oracle there is no general method for determining whether $\mathcal{K} = \emptyset$. Fortunately, we do not have to tackle such problems: by the next claim, we may assume that $\mathcal{K}$ is given explicitly, as its size is linear in n. We use $n = |V|$, $m = |E|$ for the number of nodes and edges of the graph, respectively.

Claim 13. *If the graph $G = (V, E)$ satisfies (1) and (3), then the total number of $K_{t,t}$ and K_{t+1} subgraphs is bounded by $\frac{(t+3)n}{2}$.*

Proof. Assume that $v \in V$ is contained in a forbidden subgraph and so $d_G(v) = t + 1$. If we select an edge incident to v, the remaining t edges may be contained in at most one K_{t+1} subgraph hence the number of K_{t+1}'s containing v is at most $t + 1$. However, these t edges also determine one of the colour classes of those $K_{t,t}$'s containing them. If we pick a node v' from this colour class (implying $d_G(v') = t + 1$), pick an edge incident to v' (but not to v), then the remaining t edges, if they do so, exactly determine the other colour class of a $K_{t,t}$ subgraph. Therefore the number of $K_{t,t}$ subgraphs containing v is bounded by $(t + 1)t = t^2 + t$. Hence the total number of forbidden $K_{t,t}$ and K_{t+1} subgraphs is at most $\frac{(t^2+t)n}{2t} + \frac{(t+1)n}{t+1} = \frac{(t+3)n}{2}$. □

Now we turn to the algorithm. First we choose an inclusionwise maximal subset $\mathcal{H} = \{H_1, \dots, H_k\}$ of disjoint forbidden subgraphs greedily. For $t = 2$, let us always choose squares as long as possible and then go on with triangles. This can be done in $O(t^3 n)$ time as follows. Maintain an array of size m that encodes for each edge whether it is used in one of the selected forbidden subgraphs or not. When increasing $\mathcal{H}$, one only has to check whether any of the edges of the examined forbidden subgraph is already used, which takes $O(t^2)$ time. This and Claim 13 together give an $O(t^3 n)$ bound.

Let us shrink the members of $\mathcal{H}$ simultaneously (this can be easily done since they are disjoint), resulting in a graph $G' = (V', E')$ with a bound $b' : V' \to \mathbb{Z}_+$ and no forbidden subgraphs since $\mathcal{H}$ was maximal. One can find a maximal b'-matching M' in G' in $O(|V'||E'| \log |V'|) = O(nm \log m)$ time as in [5]. Using the constructions described in Lemmas 8 and 9 for $H_k, ..., H_1$, this can be modified into a maximal $\mathcal{K}$-free b-matching M. Note that, for $t = 2$, H_i is always uncovered in the actual graph by the selection rule. A dual optimal solution $U, W, \mathcal{P}, \dot{\mathcal{K}}$ can be constructed simultaneously as in the proof of Theorem 5. The best time bound of the shrinking and extension steps may depend on the data structure used and the representation of the graph. In any case, one such step may be performed in $O(m)$ time and $|\mathcal{H}| = O(n)$, hence the total running time is $O(t^3 n + nm \log m)$.

References

1. Bérczi, K., Kobayashi, Y.: An Algorithm for $(n-3)$–Connectivity Augmentation Problem: Jump System Approach. Technical report, Department of Mathematical Engineering, University of Tokyo, METR 2009-12
2. Cornuéjols, G., Pulleyblank, W.: A Matching Problem With Side Conditions. Discrete Math. 29, 135–139 (1980)
3. Frank, A.: Restricted t-matchings in Bipartite Graphs. Discrete Appl. Math. 131, 337–346 (2003)
4. Frank, A., Jordán, T.: Minimal Edge-Coverings of Pairs of Sets. J. Comb. Theory Ser. B 65, 73–110 (1995)
5. Gabow, H.N.: An Efficient Reduction Technique for Degree-Constrained Subgraph and Bidirected Network Flow Problems. In: STOC '83: Proceedings of the fifteenth annual ACM symposium on Theory of computing, pp. 448–456. ACM, New York (1983)
6. Hartvigsen, D.: Extensions of Matching Theory. PhD Thesis, Carnegie-Mellon University (1984)
7. Hartvigsen, D.: The Square-Free 2-factor Problem in Bipartite Graphs. In: Cornuéjols, G., Burkard, R.E., Woeginger, G.J. (eds.) IPCO 1999. LNCS, vol. 1610, pp. 234–241. Springer, Heidelberg (1999)
8. Hartvigsen, D.: Finding maximum square-free 2-matchings in bipartite graphs. J. Comb. Theory Ser. B 96, 693–705 (2006)
9. Hartvigsen, D., Li, Y.: Triangle-Free Simple 2-matchings in Subcubic Graphs (Extended Abstract). In: Fischetti, M., Williamson, D.P. (eds.) IPCO 2007. LNCS, vol. 4513, pp. 43–52. Springer, Heidelberg (2007)
10. Király, Z.: C_4-free 2-factors in Bipartite Graphs. Technical report, Egerváry Research Group, Department of Operations Research, Eötvös Loránd University, Budapest, TR-2001-13 (2001)
11. Király, Z.: Restricted t-matchings in Bipartite Graphs. Technical report, Egerváry Research Group, Department of Operations Research, Eötvös Loránd University, Budapest, TR-2009-04 (2009)

12. Kobayashi, Y.: A Simple Algorithm for Finding a Maximum Triangle-Free 2-matching in Subcubic Graphs. Technical report, Department of Mathematical Engineering, University of Tokyo, METR 2009-26 (2009)
13. Kobayashi, Y., Takazawa, K.: Even Factors, Jump Systems, and Discrete Convexity. Technical report, Department of Mathematical Engineering, University of Tokyo, METR 2007-36 (2007)
14. Kobayashi, Y., Takazawa, K.: Square-Free 2-matchings in Bipartite Graphs and Jump Systems. Technical report, Department of Mathematical Engineering, University of Tokyo, METR 2008-40 (2008)
15. Makai, M.: On Maximum Cost $K_{t,t}$-free t-matchings of Bipartite Graphs. SIAM J. Discret. Math. 21, 349–360 (2007)
16. Pap, G.: Alternating Paths Revisited II: Restricted b-matchings in Bipartite Graphs. Technical report, Egerváry Research Group, Department of Operations Research, Eötvös Loránd University, Budapest, TR-2005-13 (2005)
17. Schrijver, A.: Combinatorial Optimization - Polyhedra and Efficiency. Springer, Heidelberg (2003)
18. Szabó, J.: Jump systems and matroid parity (in hungarian). Master's Thesis, Eötvös Loránd University, Budapest (2002)
19. Takazawa, K.: A Weighted $K_{t,t}$-free t-factor Algorithm for Bipartite Graphs. Math. Oper. Res. 34, 351–362 (2009) (INFORMS)
20. Végh, L. A.: Augmenting Undirected Node-Connectivity by One. Technical report, Egerváry Research Group, Department of Operations Research, Eötvös Loránd University, Budapest, TR-2009-10 (2009)

Zero-Coefficient Cuts

Kent Andersen and Robert Weismantel

Otto-von-Guericke-University of Magdeburg,
Department of Mathematics/IMO, Universitätsplatz 2,
39106 Magdeburg, Germany
{andersen,weismant}@mail.math.uni-magdeburg.de

Abstract. Many cuts used in practice to solve mixed integer programs are derived from a basis of the linear relaxation. Every such cut is of the form $\alpha^T x \geq 1$, where $x \geq 0$ is the vector of non-basic variables and $\alpha \geq 0$. For a point $\bar{x}$ of the linear relaxation, we call $\alpha^T x \geq 1$ a *zero-coefficient cut* wrt. $\bar{x}$ if $\alpha^T \bar{x} = 0$, since this implies $\alpha_j = 0$ when $\bar{x}_j > 0$. We consider the following problem: Given a point $\bar{x}$ of the linear relaxation, find a basis, and a zero-coefficient cut wrt. $\bar{x}$ derived from this basis, or provide a certificate that shows no such cut exists. We show that this problem can be solved in polynomial time. We also test the performance of zero-coefficient cuts on a number of test problems. For several instances zero-coefficient cuts provide a substantial strengthening of the linear relaxation.

Keywords: Mixed integer program; Lattice basis; Cutting plane; Split cut.

1 Introduction

This paper concerns mixed integer linear sets of the form:

$$P_I := \{x \in \mathbb{R}^n : Ax = b, x \geq 0 \text{ and } x_j \in \mathbb{Z} \text{ for } j \in N_I\}, \tag{1}$$

where $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $N := \{1, 2, \ldots, n\}$ is an index set for the variables, and $N_I \subseteq N$ is an index set for the integer constrained variables. The linear relaxation of P_I is denoted P. For simplicity we assume A has full row rank. A basis of A is a subset $B \subseteq N$ of m variables such that the columns $\{a_{.j}\}_{j \in B}$ of A are linearly independent. From a basis B one obtains the *basic polyhedron* associated with B:

$$P(B) := \{x \in \mathbb{R}^n : Ax = b \text{ and } x_j \geq 0 \text{ for all } j \in N \setminus B\}, \tag{2}$$

and the corresponding *corner polyhedron*:

$$P_I(B) := \{x \in P(B) : x_j \in \mathbb{Z} \text{ for all } j \in N_I\}. \tag{3}$$

Observe that $P(B)$ can be obtained from P by deleting the non-negativity constraints on the basic variables x_i for $i \in B$. Also observe that $P(B)$ can be written in the form:

$$P(B) = \{x \in \mathbb{R}^n : x = \bar{x}^B + \sum_{j \in N \setminus B} x_j r^{j,B} \text{ and } x_j \geq 0 \text{ for } j \in N \setminus B\}, \tag{4}$$

where $\bar{x}^B \in \mathbb{R}^n$ is the basic solution associated with B, and the vectors $r^{j,B} \in \mathbb{R}^n$ for $j \in N \setminus B$ are the extreme rays of $P(B)$. Finally observe that every non-trivial valid inequality for $P_I(B)$ is of the form $\sum_{j \in N \setminus B} \alpha_j x_j \geq 1$ with $\alpha_j \geq 0$ for all $j \in N \setminus B$. We say that a valid inequality $\sum_{j \in N \setminus B} \alpha_j x_j \geq 1$ for $P_I(B)$ is a valid cut for P_I that can be derived from the basis B.

Several classes of cuts can be derived from a basis. Some of these cuts are derived from a single equation. This equation may be one of the equalities $x = \bar{x}^B + \sum_{j \in N \setminus B} x_j r^{j,B}$, or an integer combination of these equalities. The integrality constraints on the variables are then used to obtain a valid cut. Single-row cuts are named either Mixed Integer Gomory (MIG) cuts [10], Mixed Integer Rounding (MIR) cuts [11] or Split Cuts [8].

Recent research has attempted to use several equations simultaneously to generate valid cuts from a basis. In [3,9], two equations were considered, and cuts named disection cuts and lifted two-variable cuts were derived. All these cuts are intersection cuts [4], and their validity is based on lattice-free polyhedra.

This paper is motivated by the following question: Which properties should cuts derived from bases of the linear relaxation have in order to be effective cutting planes for mixed integer programs ? Such properties could be useful for identifying classes of cuts that are effective in practice.

A first realization is that such cuts must be sparse, *i.e.*, the cuts must have many zero coefficients. Dense cuts are hard for linear programming solvers to handle, and they have not been shown to be effective in closing the integrality gap of mixed integer programs.

Secondly, deciding exactly which variables should have a zero coefficient in a cut seems hard. It therefore seems natural to consider a specific point $\bar{x} \in P$ and aim at cuts that form solutions to the following variant of a *separation problem*:

$$\min\{\sum_{j \in N \setminus B} \alpha_j \bar{x}_j : \sum_{j \in N \setminus B} \alpha_j x_j \geq 1 \text{ is valid for } P_I(B) \text{ for some basis } B\} \tag{5}$$

with many zero coefficients on variables $j \in N \setminus B$ for which $\bar{x}_j > 0$. Ideally a cut $\sum_{j \in N \setminus B} \alpha_j x_j \geq 1$ should be *maximally violated*, *i.e.*, the cut should satisfy $\alpha_j = 0$ for all $j \in N \setminus B$ with $\bar{x}_j > 0$. We call a maximally violated cut obtained from a basis of the linear relaxation for a *zero-coefficient cut* wrt. $\bar{x}$. Zero-coefficient cuts are optimal solutions to the above separation problem when they exist, and they necessarily have coordinates with zero coefficients. Zero-coefficient cuts therefore seem to be a class of cuts of high quality for solving mixed integer programs in practice.

The main result in this paper is to show that a zero-coefficient cut wrt. a point $\bar{x} \in P$ can be identified in polynomial time if such a cut exists. In other words, given a point $\bar{x} \in P$, it is possible in polynomial time to find a basis B, and a valid inequality $\sum_{j\in N\setminus B}\alpha_j x_j \geq 1$ for $P_I(B)$ which satisfies $\sum_{j\in N\setminus B}\alpha_j\bar{x}_j = 0$ if such an inequality exists. The cuts we identify are split cuts, and we show that, if there exists a zero-coefficient cut wrt. $\bar{x}$, then there also exists a zero-coefficient cut wrt. $\bar{x}$ which is a split cut. The cut is computed by first pivoting to an appropriate basis, and then computing a lattice basis of a well chosen lattice.

It has been shown that, in general, the separation problem for split cuts is NP-hard [7]. Our result demonstrates that, if one insists on a maximally violated split cut, then the separation problem can be solved in polynomial time. Zero-coefficient cuts therefore seem to provide a reasonably efficient alternative to optimizing over the split closure of a mixed integer set. The quality of the split closure as an approximation of a mixed integer set was demonstrated in [5].

The performance of zero-coefficient cuts is tested computationally on instances from miplib 3.0 [6] and miplib 2003 [1]. We restrict our experiments to the corner polyhedron $P_I(B^*)$ obtained from an optimal basis B^* of the LP relaxation. In other words we do not examine the effect of pivoting in our experiments. On several test problems, zero-coefficient close substantially more integrality gap than the MIG cuts obtained from the equations defining the optimal simplex tableau.

The remainder of the paper is organized as follows. In Sect. 2 we derive an infeasibility certificate for the set $P_I(B)$ for a given basis B. This certificate is key for deriving zero-coefficient cuts. Zero-coefficient cuts are motivated and presented in Sect. 3. Our main theorem is proved in Sect. 4. Finally our computational results are presented in Sect. 5.

2 Infeasibility Certificates for Corner Polyhedra

We now consider a fixed basis B, and we address the question of when $P_I(B)$ is empty. We will present a certificate that proves $P_I(B) = \emptyset$ whenever this is the case. We first derive the representation (4) of $P(B)$. Since we consider a fixed basis B throughout this section, we let $\bar{x} := \bar{x}^B$ and $r^j := r^{j,B}$ for $j \in N\setminus B$. Let A_B denote the induced sub-matrix of A composed of the columns in B, and define $\bar{a}_{.j} := (A_B)^{-1}a_{.j}$ for $j \in N\setminus B$. We may write $P(B)$ in the form:

$$\begin{aligned} P(B) = \{x \in \mathbb{R}^n :\quad & x_i = \bar{x}_i - \sum_{j\in N\setminus B}\bar{a}_{i,j}x_j, && \text{for } i \in B,\\ & x_j = x_j, && \text{for } j \in N\setminus B,\\ & x_j \geq 0 \text{ for } j \in N\setminus B \quad \}. && \end{aligned} \tag{6}$$

Defining the following vectors $r^j \in \mathbb{R}^n$ for $j \in N\setminus B$:

$$r^j_k := \begin{cases} -\bar{a}_{k,j} & \text{if } k \in B,\\ 1 & \text{if } k = j,\\ 0 & \text{otherwise,}\end{cases} \tag{7}$$

the representation (6) of $P(B)$ can be re-written in the form

$$P(B) = \{x \in \mathbb{R}^n : x = \bar{x} + \sum_{j\in N\setminus B} s_j r^j \text{ and } s_j \geq 0 \text{ for } j \in N\setminus B\}. \tag{8}$$

Hence $P_I(B)$ is empty if and only if the translated cone $\bar{x} + \text{cone}(\{r^j\}_{j \in N \setminus B})$ does not contain mixed integer points. Our certificate for proving $P_I(B)$ is empty is a *split disjunction*. A split disjunction is of the form $\pi^T x \leq \pi_0 \vee \pi^T x \geq \pi_0 + 1$, where $(\pi, \pi_0) \in \mathbb{Z}^{n+1}$ and $\pi_j = 0$ for all $j \in N \setminus N_I$. All mixed integer points $x \in P_I(B)$ satisfy all split disjunctions.

Our point of departure is a result characterizing when an affine set contains integer points (see [2]). Specifically, consider the affine set:

$$T^a := f + \text{span}(\{q^j\}_{j \in J}), \tag{9}$$

where $f \in \mathbb{Q}^n$, J is a finite index set and $\{q^j\}_{j \in J} \subset \mathbb{Q}^n$. A result in [2] shows that T^a does not contain integer points if and only if there exists $\pi \in \mathbb{Z}^n$ such that $\pi^T f \notin \mathbb{Z}$ and $\pi^T q^j = 0$ for all $j \in J$. Observe that such a vector $\pi \in \mathbb{Z}^n$ gives a split disjunction $\pi^T x \leq \lfloor \pi^T f \rfloor \vee \pi^T x \geq \lceil \pi^T f \rceil$ which proves $T^a \cap \mathbb{Z}^n = \emptyset$, *i.e.*, we have $T^a \subseteq \{x \in \mathbb{R}^n : \lfloor \pi^T f \rfloor < \pi^T x < \lceil \pi^T f \rceil\}$.

We first generalize this result from integer points in affine sets to *mixed* integer points in affine sets.

Lemma 1. *The set T^a does not contain mixed integer points if and only if there exists $\pi \in \mathbb{Z}^n$ such that $\pi^T f \notin \mathbb{Z}$, $\pi^T q^j = 0$ for all $j \in J$ and $\pi_j = 0$ for all $j \in N \setminus N_I$.*

Proof. We have that $\{x \in T^a : x_j \in \mathbb{Z} \text{ for all } j \in N_I\}$ is empty if and only if $\{x \in \mathbb{R}^n : x_i = f_i + \sum_{j \in J} s_j q^j_i \text{ and } x_i \in \mathbb{Z} \text{ for all } i \in N_I\}$ is empty if and only if there exists a vector $\pi \in \mathbb{Z}^n$ such that $\pi^T f \notin \mathbb{Z}$, $\pi^T q^j = 0$ for all $j \in J$ and $\pi_j = 0$ for all $j \in N \setminus N_I$. ■

Lemma 1 shows that, if T^a does not contain mixed integer points, then there exists split disjunction $\pi^T x \leq \lfloor \pi^T f \rfloor \vee \pi^T x \geq \lceil \pi^T f \rceil$, where $\pi \in \mathbb{Z}^n$ satisfies $\pi_j = 0$ for all $j \in N \setminus N_I$, such that $T^a \subset \{x \in \mathbb{R}^n : \lfloor \pi^T f \rfloor < \pi^T x < \lceil \pi^T f \rceil\}$, *i.e.*, this split disjunction provides a certificate that shows that T^a does not contain any mixed integer points.

We next generalize Lemma 1 from mixed integer points in affine sets to mixed integer points in translates of polyhedral cones.

Lemma 2. *The set $f + \text{cone}(\{q^j\}_{j \in J})$ contains no mixed integer points if and only if the set $f + \text{span}(\{q^j\}_{j \in J})$ contains no mixed integer points.*

Proof. Let $T^c := f + \text{cone}(\{q^j\}_{j \in J})$. We have to show that T^c does not contain mixed integer points if and only if T^a does not contain mixed integer points. Clearly, if T^c contains mixed integer points, then T^a also contains mixed integer points since $T^c \subseteq T^a$. Hence we only need to show the other direction.

Therefore suppose T^c does not contain mixed integer points, and assume, for a contradiction, that T^a contains mixed integer points. Let $x' \in T^a$ satisfy $x'_j \in \mathbb{Z}$ for all $j \in N_I$, and let $s' \in \mathbb{R}^{|J|}$ be such that $x' = f + \sum_{j \in J} s'_j q^j$. Choose an integer $d > 0$ such that $dq^j \in \mathbb{Z}^n$ for all $j \in J$ and define $x'' := x' - \sum_{j \in J} \lfloor \frac{s'_j}{d} \rfloor dq^j$. We have $x'' \in \{x \in \mathbb{R}^n : x_j \in \mathbb{Z} \text{ for } j \in N_I\}$ and $x'' = f + \sum_{j \in J} (\frac{s'_j}{d} - \lfloor \frac{s'_j}{d} \rfloor) dq^j$. Hence $x'' \in T^c$ which is a contradiction. ■

Since $P_I(B)$ is the set of mixed integer points in a translate of a polyhedral cone, we now have the following certificate for when $P_I(B)$ is empty.

Corollary 1. *We have $P_I(B) = \emptyset$ if and only there exists $\pi \in \mathbb{Z}^n$ such that $\pi^T \bar{x} \notin \mathbb{Z}$, $\pi^T r^j = 0$ for all $j \in N \setminus B$ and $\pi_j = 0$ for all $j \in N \setminus N_I$.*

3 Zero-Coefficient Cuts from Corner Polyhedra

We now use the certificate obtained in Sect. 2 to derive zero-coefficient cuts for a corner polyhedron $P_I(B)$ for a fixed basis B. As in Sect. 2, we let $\bar{x} := \bar{x}^B$ and $r^j := r^{j,B}$ for $j \in N \setminus B$. We consider an optimization problem (MIP):

$$\min\{\sum_{j \in N \setminus B} c_j x_j : x \in P_I(B)\},$$

where $c \in \mathbb{R}^{|N \setminus B|}$ denotes the objective coefficients. The linear programming relaxation of (MIP) is denoted (LP). The set of feasible solutions to (LP) is the set $P(B)$. We assume $c_j \geq 0$ for all $j \in N \setminus B$ since otherwise (LP) is unbounded.

To motivate zero-coefficient cuts, we first consider a generic cutting plane algorithm for strengthening the LP relaxation (LP) of (MIP). Let $V \subset \mathbb{Q}_+^{|N \setminus B|}$ be a family of valid inequalities for $P_I(B)$, *i.e.*, we have that $\sum_{j \in N \setminus B} \alpha_j x_j \geq 1$ is valid for $P_I(B)$ for all $\alpha \in V$. Let $x' \in P(B)$ be arbitrary. A *separation problem* (SEP) wrt. x' can be formulated as:

$$\min\{\sum_{j \in N \setminus B} \alpha_j x'_j : \alpha \in V\}.$$

A cutting plane algorithm (CPalg) for using V to strengthen the LP relaxation (LP) of (MIP) can now be designed by iteratively solving (SEP) wrt. various points $x' \in P(B)$:

```
Cutting plane algorithm (CPalg):
  (1) Set $k := 0$. Let $x^k := \bar{x}$ be an optimal solution to (LP).
  (2) Solve (SEP) wrt. $x^k$. Let $\alpha^k \in V$ be an optimal solution.
  (3) While $\sum_{j \in N \setminus B} \alpha_j^k x_j^k < 1$:
      (a) Add $\sum_{j \in N \setminus B} \alpha_j^k x_j \geq 1$ to (LP) and re-optimize.
          Let $x^{k+1}$ be an optimal solution.
      (b) Solve (SEP) wrt. $x^{k+1}$.
          Let $\alpha^{k+1} \in V$ be an optimal solution.
      (c) Set $k := k + 1$.
End.
```

In (CPalg) above, only one cut is added in every iteration. It is also possible to add several optimal and sub-optimal solutions to (SEP). Furthermore, for many classes V of valid cutting planes for $P_I(B)$, (SEP) can not necessarily be solved in polynomial time, and final convergence of (CPalg) is not guaranteed. For instance, if V is the class of split cuts, (SEP) is NP-hard [7].

For $\alpha \in V$ and $x' \in P(B)$, the inequality $\sum_{j\in N\setminus B} \alpha_j x_j \geq 1$ is maximally violated by x' when $\sum_{j\in N\setminus B} \alpha_j x'_j = 0$. We call $\sum_{j\in N\setminus B} \alpha_j x_j \geq 1$ a *zero-coefficient cut* wrt. x' when $\sum_{j\in N\setminus B} \alpha_j x'_j = 0$. Observe that if a zero-coefficient wrt. x^k exists in the family V of valid inequalities for $P_I(B)$ in the k^{th} iteration of (CPalg), then this cut is an optimal solution to (SEP).

Since (SEP) always returns a zero-coefficient cut wrt. the point that is being separated whenever such a cut exists, the structure of (CPalg) is such that zero-coefficient cuts are separated first, *i.e.*, the first iterations of (CPalg) consist of the following cutting plane algorithm (InitCPalg):

```
Cutting plane algorithm (InitCPalg):
  (1) Set $k := 0$. Let $x^k := \bar{x}$ be an optimal solution to (LP).
  (2) While there exists $\alpha^k \in V$ such that $\sum_{j\in N\setminus B} \alpha^k_j x_j \geq 1$
      is a zero-coefficient cut wrt. $x^k$:
      (a) Add $\sum_{j\in N\setminus B} \alpha^k_j x_j \geq 1$ to (LP) and re-optimize.
          Let $x^{k+1}$ be an optimal solution.
      (b) Set $k := k + 1$.
End.
```

When (InitCPalg) terminates, a point $x^* \in P(B)$ is obtained that satisfies $\sum_{j\in N\setminus B} \alpha_j x^*_j > 0$ for all $\alpha \in V$, *i.e.*, there does not exist any zero-coefficient cut wrt. x^* in the family V. Following (InitCPalg), if possible and desirable, (CPalg) can be continued in order to strengthen the LP relaxation of (MIP) further with valid cuts that are not maximally violated.

In the following we show that (InitCPalg) can be implemented to run in polynomial time by using only split cuts. Since the initial phase (InitCPalg) of (CPalg) can be implemented with split cuts only, this could suggest an explanation of why split cuts have been observed to close a large amount of the integrality on many instances [5].

We first review how split cuts are derived for $P_I(B)$. Every split cut is derived from a vector $\pi \in \mathbb{Z}^n$ satisfying $\pi^T\bar{x} \notin \mathbb{Z}$ and $\pi_j = 0$ for all $j \in N \setminus N_I$. Define $f_0(\pi) := \pi^T\bar{x} - \lfloor \pi^T\bar{x} \rfloor$ and $f_j(\pi) := \pi^T r^j - \lfloor \pi^T r^j \rfloor$ for $j \in N_I \setminus B$. The inequality:

$$\sum_{j\in N\setminus B} \frac{x_j}{\alpha_j(\pi)} \geq 1 \tag{10}$$

is the (strengthened) split cut defined by π, where $\alpha_j(\pi)$ for $j \in N \setminus B$ is:

$$\alpha_j(\pi) := \begin{cases} \frac{f_0(\pi)}{1-f_j(\pi)} & \text{if } j \in N_I \text{ and } 0 < f_j(\pi) < 1 - f_0(\pi), \\ \frac{1-f_0(\pi)}{f_j(\pi)} & \text{if } j \in N_I \text{ and } 1 - f_0(\pi) \leq f_j(\pi) < 1, \\ \frac{1-f_0(\pi)}{\pi^T r^j} & \text{if } j \notin N_I \text{ and } \pi^T r^j > 0, \\ -\frac{f_0(\pi)}{\pi^T r^j} & \text{if } j \notin N_I \text{ and } \pi^T r^j < 0, \\ +\infty & \text{otherwise.} \end{cases} \tag{11}$$

We next prove that, for a given point $x' \in P(B)$, if there exists *any* valid inequality for $P_I(B)$ which is maximally violated wrt. x', then there also exists a split cut which is maximally violated wrt. x'.

Theorem 1. *Let $x' \in P(B)$ be arbitrary. If there exists a valid inequality for $P_I(B)$ which is a zero-coefficient cut wrt. x', then there also exists a split cut for $P_I(B)$ which is a zero-coefficient cut wrt. x'.*

Proof. Let $x' \in P(B)$, and let $\sum_{j \in N \setminus B} \alpha'_j x_j \geq 1$ be a valid inequality for $P_I(B)$ which is a zero-coefficient cut wrt. x'. Since $\sum_{j \in N \setminus B} \alpha'_j x'_j = 0$ and $\alpha'_j, s'_j \geq 0$ for all $j \in N \setminus B$, we must have $\alpha'_j = 0$ for all $j \in N \setminus B$ satisfying $x'_j > 0$. Let $X' := \{j \in N \setminus B : x'_j > 0\}$. It follows that $0 \geq 1$ is valid for:

$$\begin{aligned} Q_I &:= \{x \in P_I(B) : x_j = 0 \text{ for all } j \in (N \setminus B) \setminus X'\} \\ &= \{x \in \bar{x} + \text{cone}(\{r^j\}_{j \in X'}) : x_j \in \mathbb{Z} \text{ for all } j \in N_I\}. \end{aligned}$$

Since $Q_I = \emptyset$, Lemma 2 shows there exists $\bar{\pi} \in \mathbb{Z}^n$ such that $\bar{\pi}^T r^j = 0$ for $j \in X'$, $\bar{\pi}_j = 0$ for $j \in N \setminus N_I$ and $\bar{\pi}^T \bar{x} \notin \mathbb{Z}$. From (10) and (11) it now follows that the split cut derived from $\bar{\pi}$ is a zero-coefficient cut wrt. x'. ■

In general it is NP-hard to separate a split cut for $P_I(B)$ [7]. However, as we will show next, it is possible to separate a zero-coefficient split cut wrt. a given point in polynomial time whenever such a split cut exists.

Let $x' \in P(B)$. Define $X' := \{j \in N \setminus B : x'_j > 0\}$. From (11) we have that $\pi \in \mathbb{Z}^n$ defines a maximally violated split cut wrt. x' if and only if

$$\pi^T r^j = 0 \quad \text{for all } j \in X', \tag{12}$$

$$\pi_j = 0 \quad \text{for all } j \in N \setminus N_I, \tag{13}$$

$$\pi^T \bar{x} \notin \mathbb{Z}. \tag{14}$$

If $\pi \in \mathbb{Z}^n$ satisfies (12)-(14), then a split cut can be derived from π since $\pi^T \bar{x} \notin \mathbb{Z}$, and we have $\alpha_j(\pi) = +\infty$ for all $j \in X'$, which implies that the coefficients on the variables x_j for $j \in X'$ in the split cut (10) are all zero. Hence any $\pi \in \mathbb{Z}^n$ that satisfies (12)-(14) defines a zero-coefficient split cut wrt. x'. Conversely, if there exists a valid inequality for $P_I(B)$ which is maximally violated wrt. x', then Theorem 1 shows there exists $\pi \in \mathbb{Z}^n$ that satisfies (12)-(14).

Let $L(x') \subseteq \mathbb{Z}^n$ denote the set of $\pi \in \mathbb{Z}^n$ that satisfy (12) and (13):

$$L(x') := \{\pi \in \mathbb{Z}^n : \pi^T r^j = 0 \text{ for all } j \in X' \text{ and } \pi_j = 0 \text{ for all } j \in N \setminus N_I\}.$$

Observe that $L(x')$ is a lattice, *i.e.*, for any $\pi^1, \pi^2 \in L(x')$ and $k \in \mathbb{Z}$, we have $k\pi^1 \in L(x')$ and $\pi^1 + \pi^2 \in L(x')$. For any lattice it is possible to compute a basis for the lattice in polynomial time. Hence we can find vectors $\pi^1, \ldots, \pi^p \in L(x')$ in polynomial time such that:

$$L(x') = \{\pi \in \mathbb{Z}^n : \pi = \sum_{i=1}^{p} \lambda_i \pi^i \text{ and } \lambda_i \in \mathbb{Z} \text{ for } i = 1, 2, \ldots, p\}.$$

Now, if there exists a lattice basis vector $\pi^{\bar{i}} \in L(x')$ with $\bar{i} \in \{1,2,\dots,p\}$ such that $(\pi^{\bar{i}})^T\bar{x} \notin \mathbb{Z}$, then the split cut derived from $\pi^{\bar{i}}$ is maximally violated wrt. x'. Conversely, if we have $(\pi^i)^T\bar{x} \in \mathbb{Z}$ for all $i \in \{1,2,\dots,p\}$, then $\pi^T\bar{x} \in \mathbb{Z}$ for all $\pi \in L(x')$. We therefore have the following.

Corollary 2. *Let $x' \in P(B)$ be arbitrary. If there exists a valid inequality for $P_I(B)$ that is maximally violated wrt. x', then it is possible to find such an inequality in polynomial time.*

Based on the above results, we have the following implementation of the cutting plane algorithm (InitCPalg) presented earlier:

```
Implementation of (InitCPalg):
  (1) Set $k := 0$. Let $x^k := \bar{x}$ be an optimal solution to (LP).
  (2) Find a lattice basis $\pi^1,\dots,\pi^{p_k}$ for $L(x^k)$.
      Let $I(x^k) := \{i \in \{1,\dots,p_k\} : (\pi^i)^T\bar{x} \notin \mathbb{Z}\}$.
  (3) While $I(x^k) \neq \emptyset$:
      (a) Add all split cuts generated from vectors $\pi^i$
          with $i \in I(x^k)$ to (LP) and re-optimize.
          Let $x^{k+1}$ be an optimal solution.
      (b) Find a lattice basis $\pi^1,\dots,\pi^{p_{k+1}}$ for $L(x^{k+1})$.
          Let $I(x^{k+1}) := \{i \in \{1,\dots,p_{k+1}\} : (\pi^i)^T\bar{x} \notin \mathbb{Z}\}$.
      (c) Set $k := k+1$.
End.
```

We next argue that mixed integer Gomory cuts play a natural role in the above implementation of (InitCPalg). Consider the computation of the lattice basis for $L(x^0)$ in step (2) of (InitCPalg). Observe that, since $x^0 = \bar{x}$, we have $L(x^0) = \mathbb{Z}^n$, and therefore $\pi^1 := e^1,\dots,\pi^n := e^n$ is a lattice basis for $L(x^0)$, where $e^1,\dots,e^n$ denote the unit vectors in $\mathbb{R}^n$. Since a split cut ([10]) obtained from a unit vector is a mixed integer Gomory cut, the first cuts added in step (3).(a) of the above implementation of (InitCPalg) are the mixed integer Gomory cuts. A natural computational question therefore seems to be how much more integrality gap can be closed by continuing (InitCPalg) and generating the remaining zero-coefficient cuts.

4 Zero-Coefficient Cuts from Mixed Integer Polyhedra

In Sect. 3 we considered a fixed basis B and a point $x' \in P(B)$, and we demonstrated how to obtain a zero-coefficient cut wrt. x' from $P_I(B)$ whenever such a cut exists. Given $x' \in P$, we now consider how to obtain an appropriate basis, *i.e.*, we show how to identify a basis B such that a zero-coefficient cut wrt. x' can be derived from $P_I(B)$. For this, we first relate the emptyness of two corner polyhedra $P_I(B)$ and $P_I(B')$ obtained from two adjacent bases B and B' for P.

Lemma 3. *Let B be a basis for P, and let $B' := (B \setminus \{\bar{i}\}) \cup \{\bar{j}\}$ be an adjacent basis to B, where $\bar{i} \in B$ and $\bar{j} \in N \setminus B$. Then $P_I(B) = \emptyset$ if and only if $P_I(B') = \emptyset$.*

Proof. For simplicity let $\bar{x} := \bar{x}^B$ and $\bar{x}' := \bar{x}^{B'}$. Also let $\bar{a}_{.j} := (A_B)^{-1}a_{.j}$ for all $j \in N \setminus B$ and $\bar{a}'_{.j} := (A_{B'})^{-1}a_{.j}$ for all $j \in N \setminus B'$, where A_B and $A_{B'}$ denote the basis matrices associated with the bases B and B' respectively.

Suppose $z \in P_I(B)$. We have $z_i = \bar{x}'_i + \sum_{j \in N \setminus B'} \bar{a}'_{i,j} z_j$ for all $i \in B'$, $z_j \geq 0$ for all $j \in N \setminus (B' \cup \{\bar{i}\})$ and $z_j \in \mathbb{Z}$ for all $j \in N_I$. If $z_{\bar{i}} \geq 0$, we are done, so suppose $z_{\bar{i}} < 0$. Choose an integer $k > 0$ such that $k\bar{a}'_{.\bar{i}} \in \mathbb{Z}^m$ and $z_{\bar{i}} + k \geq 0$. Defining $z'_i := z_i + k\bar{a}'_{i,\bar{i}}$ for all $i \in B'$, $z'_{\bar{i}} := z_{\bar{i}} + k$ and $z'_j := z_j$ for all $j \in N \setminus (B' \cup \{\bar{i}\})$, we have $z' \in P_I(B')$. Hence $P_I(B) \neq \emptyset$ implies $P_I(B') \neq \emptyset$. The opposite direction is symmetric. ■

From Lemma 3 it follows that either all corner polyhedra $P_I(B)$ associated with bases B for P are empty, or they are all non-empty. We next present a pivot operation from a basis B to an adjacent basis B' with the property that, if a zero-coefficient cut wrt. a point $x' \in P$ can be derived from B, then a zero-coefficient cut wrt. x' can also be derived from B'.

Lemma 4. *Let B be a basis for P, let $x' \in P$ and define $X' := \{j \in N : x'_j > 0\}$. Also let $B' := (B \setminus \{\bar{i}\}) \cup \{\bar{j}\}$ be an adjacent basis to B, where $\bar{i} \in B \setminus X'$ and $\bar{j} \in X' \setminus B$. If a zero-coefficient cut wrt. x' can be derived from B, then a zero-coefficient cut wrt. x' can also be derived from B'.*

Proof. Given a set $S \subseteq N$, we will use sets obtained from P, P_I, $P(B)$ and $P_I(B)$ by setting $x_j = 0$ for all $j \in N \setminus S$. For $S \subseteq N$, define $Q(S) := \{x \in P : x_j = 0 \text{ for } j \in N \setminus S\}$ and $Q_I(S) := \{x \in P_I : x_j = 0 \text{ for } j \in N \setminus S\}$. Also, given a basis $B \subseteq S$, define $Q(B,S) := \{x \in P(B) : x_j = 0 \text{ for } j \in N \setminus S\}$ and $Q_I(B,S) := \{x \in P_I(B) : x_j = 0 \text{ for } j \in N \setminus S\}$.

Assume a zero-coefficient cut wrt. x' can be derived from B. Observe that this implies $P_I(B, B \cup X') = \emptyset$. Now, $P_I(B, B \cup X')$ is a corner polyhedron associated with $P_I(B \cup X')$, and $P_I(B', B \cup X')$ is also a corner polyhedron associated with $P_I(B \cup X')$. Since any two bases of $P(B \cup X')$ can be obtained from each other by pivoting, it follows from Lemma 3 that also $P_I(B', B \cup X') = \emptyset$. Corollary 1 now gives a split cut which is a zero-coefficient cut wrt. x' derived from B'. ■

Lemma 4 shows that, for the purpose of identifying zero-coefficient cuts wrt. x', the interesting bases to consider are those bases for which it is *not* possible to pivot a variable x_j with $j \in X'$ into the basis.

Definition 1. *Let $x' \in P$, and define $X' := \{j \in N : x'_j > 0\}$. A basis B for P is called* maximal *wrt. x' if $(B \setminus \{\bar{i}\}) \cup \{\bar{j}\}$ is* not *a basis for P for all $\bar{i} \in B \setminus X'$ and $\bar{j} \in X' \setminus B$.*

From the above results it is not clear whether it is necessary to investigate *all* maximal bases wrt. x' in order to identify a zero-coefficient cut wrt. x'. However, the following lemma shows that it is sufficient to examine just a single arbitrarily chosen maximal basis wrt. x'. In other words, if there exists a basis from which a zero-coefficient cut wrt. x' can be derived, then a zero-coefficient cut wrt. x' can be derived from *every* maximal basis wrt. x'.

Lemma 5. *If there exists a basis B for P from which a zero-coefficient cut wrt. x' can be derived, then a zero-coefficient cut can be derived from every basis for P which is maximal wrt. x'.*

Proof. Suppose B is a basis from which a zero-coefficient cut wrt. x' can be derived. Let $J := N \setminus B$, $B_{x'} := B \cap X'$ and $J_{x'} := J \cap X'$. Also let $\bar{x} := \bar{x}^B$ and $\bar{a}_{.j} := (A_B)^{-1} a_{.j}$ for $j \in J$, where A_B denotes the basis matrix associated with B. Lemma 4 shows that we may assume B is maximal, *i.e.*, we may assume that the simplex tableau associated with B is of the form:

$$x_i = 0 + \sum_{j \in J \setminus J_{x'}} \bar{a}_{i,j} x_j \text{ for all } i \in B \setminus B_{x'}, \tag{15}$$

$$x_i = \bar{x}_i + \sum_{j \in J_{x'}} \bar{a}_{i,j} x_j + \sum_{j \in J \setminus J_{x'}} \bar{a}_{i,j} x_j \quad \text{for all } i \in B_{x'}. \tag{16}$$

$$x_j \geq 0 \quad \text{for all } i \in J. \tag{17}$$

Observe that $\bar{x}_i = 0$ for all $i \in B \setminus B_{x'}$, since x' satisfies (15). The set $P(B)$ is the set of solutions to (15)-(17), and $P_I(B)$ is the set of mixed integer solutions to (15)-(17). Furthermore, from (15)-(17) it follows that a zero-coefficient cut can be derived from B if and only if the following set does not contain mixed integer points:

$$T(B) := \{x \in \mathbb{R}^n : x_i = \bar{x}_i + \sum_{j \in J_{x'}} \bar{a}_{i,j} x_j \text{ for } i \in B_{x'}, \text{ and } x_j \geq 0 \text{ for } j \in J_{x'}\}.$$

Now, $T(B)$ is a basic polyhedron associated with the set:

$$T := \{x \in \mathbb{R}^n : x_i = \bar{x}_i + \sum_{j \in J_{x'}} \bar{a}_{i,j} x_j \text{ for } i \in B_{x'}, \text{ and } x_j \geq 0 \text{ for } j \in J_{x'} \cup B_{x'}\}.$$

Furthermore, from *any* basis B' for P which is maximal wrt. x', a basic polyhedron $T(B')$ of T can be associated of the above form, and a zero-coefficient cut wrt. x' can be derived from B' if and only if $T(B')$ does not contain mixed integer points. Since $T(B)$ does not contain mixed integer points, it follows from Lemma 3 that *every* basic polyhedron $T(B')$ for T does not contain mixed integer points. Hence a zero-coefficient cut can be derived from *every* basis B' for P which is maximal wrt. x'. ■

Since a maximal basis wrt. $x' \in P$ can be obtained in polynomial time, we immediately have our main theorem.

Theorem 2. *Let $x' \in P$ be arbitrary. If there exists basis B, and a valid inequality for $P_I(B)$ which is a zero-coefficient cut wrt. x', then such a zero-coefficient cut can be obtained in polynomial time.*

5 Computational Results

We now test the performance of the cutting plane algorithm (InitCPalg) described in Sect. 3. In our implementation, we use CPLEX 9.1 for solving linear programs, and the open source software NTL for the lattice computations. We use instances from miplib 3.0 [6] and miplib 2003 [1] in our experiments. All instances are minimization problems, and we use the preprocessed version of each instance, *i.e.*, when we refer to an instance, we refer to the instance obtained after applying the preprocessor of CPLEX 9.1.

For each instance, we formulate the optimization problem over the corner polyhedron associated with an optimal basis of the LP relaxation. To distinguish the optimization problem over the corner polyhedron from the original mixed integer program, we use the following notation: The original mixed integer program is denoted (MIP), and the mixed integer program over the corner polyhedron is denoted (MIP^c). The optimal objective of (MIP) is denoted z^{MIP}, and the optimal objective value of (MIP^c) is denoted z^{MIP^c}. The LP relaxation of (MIP) is denoted (LP), and the optimal objective value of (LP) is denoted z^{LP}.

We assume the (original) mixed integer program (MIP) has n variables, and includes slack, surplus and artificial variables in the formulation:

$$\begin{aligned}
&\min c^T x\\
&\text{such that}\\
&\qquad a_{i.}^T x = b_i, \qquad \text{for all } i \in M, && (18)\\
&\qquad l_j \le x_j \le u_j, \qquad \text{for all } j \in N, && (19)\\
&\qquad x_j \in \mathbb{Z}, \qquad \text{for all } j \in N_I. && (20)
\end{aligned}$$

where M is an index set for the constraints, $c \in \mathbb{Q}^{n+|M|}$ denotes the objective coefficients, $N := \{1, 2, \ldots, (n + |M|)\}$ is an index set for the variables, $N_I \subseteq N$ denotes those variables that are integer constrained, l and u are the lower and upper bounds on the variables respectively and $(a_{i.}, b_i) \in \mathbb{Q}^{|N|+1}$ for $i \in M$ denotes the coefficients in the i^{th} constraint. The variables x_{n+i} for $i \in M$ are either slack, surplus or artificial variables.

The problem (MIP^c) is formulated as follows. An optimal basis for (LP) is an $|M|$-subset $B^* \subseteq N$ of basic variables. Let $J^* := N \setminus B^*$ denote the non-basic variables. The problem (MIP^c) can be constructed from (MIP) by eliminating certain bounds on the variables. Let J_A^* denote the non-basic artificial variables, let $J_L^* \subseteq J^* \setminus J_A^*$ denote the non-basic structural variables on lower bound, and let $J_U^* \subseteq J^* \setminus J_A^*$ denote the non-basic structural variables on upper bound. By re-defining the bounds on the variables x_j for $j \in N$ to:

$$l_j(B^*) := \begin{cases} 0 & \text{if } j \in J_A^*,\\ l_j & \text{if } j \in J_L^*,\\ -\infty & \text{otherwise,}\end{cases} \quad \text{and } u_j(B^*) := \begin{cases} 0 & \text{if } j \in J_A^*,\\ u_j & \text{if } j \in J_U^*,\\ +\infty & \text{otherwise,}\end{cases} \tag{21}$$

then the problem (MIP^c) associated with B^* is given by:

$$\begin{aligned}
&\min c^T x\\
&\text{such that}\\
&\qquad a_{i.}^T x = b_i, \qquad \text{for all } i \in M, && (22)
\end{aligned}$$

$$l_j(B^*) \le x_j \le u_j(B^*), \qquad \text{for all } j \in N, \tag{23}$$
$$x_j \in \mathbb{Z}, \qquad \text{for all } j \in N_I. \tag{24}$$

We evaluate the performance of zero-coefficient cuts by calculating how much more of the integrality gap of (MIP^c) can be closed by continuing (InitCPalg) beyond the generation of the mixed integer Gomory (MIG) cuts. For this, we first evaluate the quality of (MIP^c) as an approximation to (MIP). Our measure of quality is the size of the *integrality gap*. The integrality gap of (MIP^c) is the number $\text{Gap}_I(\text{MIP}^c) := z^{\text{MIP}^c} - z^{\text{LP}}$, and the integrality gap of (MIP) is the number $\text{Gap}_I(\text{MIP}) := z^{\text{MIP}} - z^{\text{LP}}$. The relationship between the numbers Gap_I(MIP) and $\text{Gap}_I(\text{MIP}^c)$ give information on the quality of (MIP^c) as an object for cutting plane generation for (MIP).

Table 1 contains our results for evaluating the quality of (MIP^c). The first three columns contain the problem name, the number of constraints and the number of variables for each instance respectively.

There are six instances (not included in Table 1) for which (MIP^c) does not close *any* of the integrality gap between (LP) and (MIP). This means that the bounds deleted from (MIP) to create (MIP^c) are important constraints of (MIP), although these bounds are not active in defining an optimal solution to (LP). This seems to indicate that, for these instances, (MIP^c) is not the right relaxation of (MIP) from which to derive strong cuts.

For the first seven instances in Table 1 the opposite is true, *i.e.*, for these instances the optimal objective of (MIP^c) is the same as the optimal objective of (MIP). Therefore, for these instances, if we can identify all facet defining inequalities associated with (MIP^c), and add these inequalities (LP), then all of the integrality gap between (LP) and (MIP) will be closed. Hence, for these instances, (MIP^c) seems to be the right object from which to derive strong cuts for (MIP). For the remaining instances in Table 1, not all the integrality gap between (LP) and (MIP) is closed by valid cuts from (MIP^c). However, for most instances in Table 1, it is still a large amount of integrality gap between (LP) and (MIP) that can potentially be closed with valid cuts for (MIP^c).

We next evaluate the performance of zero-coefficient cuts. Table 2 contains the main results. Before considering the results in Table 2, we first make a few comments on those instances that are *not* in Table 2.

For three instances, MIG cuts close *all* the integrality gap of (MIP^c). For these instances, zero-coefficient cuts can therefore not close any additional integrality gap, and we did not include these instances in our test of the performance of zero-coefficient cuts. Furthermore, for another nine instances, no further zero-coefficient cuts were generated besides the MIG cuts. Observe that (InitCPalg) does not do much work for these instances, since this is detected after the first lattice basis has been computed.

For the remaining instances, we divided them into those instances where MIG cuts closed less than 80% of the total integrality gap that can be closed with zero-coefficient cuts, and those where MIG cuts closed more than 80% of the total integrality gap that can be closed with zero-coefficient cuts.

Table 1. Strength of corner polyhedron

Problem	# Constr.	# Var.	z^{MIP^c}	z^{MIP}	z^{LP}	$\frac{\text{Gap}_I(\text{MIP}^c)}{\text{Gap}_I(\text{MIP})} \times 100\%$
10teams	210	1600	904	904	897	100.00 %
air04	614	7564	54632	54632	54030.44	100.00 %
egout	35	47	299.01	299.01	242.52	100.00 %
l152lav	97	1988	4722	4722	4656.36	100.00 %
mas76	12	148	40005.05	40005.05	38893.90	100.00 %
mod008	6	319	307	307	290.93	100.00 %
p0282	160	200	258401	258401	179990.30	100.00 %
qnet1	363	1417	15997.04	16029.69	14274.10	98.14 %
flugpl	13	14	759600	760500	726875	97.32 %
nsrand-ipx	76	4162	50880.00	51200.00	49667.89	79.11 %
vpm1	128	188	19	20	16.43	71.99 %
vpm2	128	188	13	13.75	11.14	71.26 %
pp08a	133	234	5870	7350	2748.35	67.84 %
p2756	702	2642	2893	3124	2701.67	45.30 %
swath	482	6260	378.07	467.41	334.50	32.78 %
modglob	286	384	19886358	20099766	19790206	31.06 %
fixnet6	477	877	3357	3981	3190.04	21.11 %
p0201	107	183	7185	7615	7155	6.52 %
rout	290	555	-1388.42	-1297.69	-1393.39	5.19 %

Table 2. Instances where the increase in objective with *all* ZC cuts was substantially larger than the increase in objective with *only* MIG cuts

Problem	# Constr.	# Var.	# MIG cuts	# Additional ZC cuts	$\frac{\Delta\text{Obj. MIG cuts}}{\Delta\text{Obj. All cuts}} \times 100\%$
l152lav	97	1988	53	6	0.00%
mkc*	1286	3230	119	29	0.00%
p0201	107	183	42	27	0.00%
p2756	702	2642	201	7	6.02%
rout	290	555	52	36	6.24%
swath	482	6260	80	26	7.58%
vpm1	114	188	15	40	22.38%
vpm2	128	188	29	29	22.73%
flugpl	13	14	13	5	23.36%
fixnet6	477	877	12	19	27.06%
timtab2*	287	648	237	653	29.85%
timtab1*	166	378	133	342	30.93%
egout	35	47	8	5	41.47%
qnet1	363	1417	55	47	45.05%
p0282	160	200	34	53	49.60%
air04	614	7564	290	30	50.53%
modglob	286	384	60	28	56.77%
mas76	12	148	11	11	65.77%
pp08a	133	234	51	34	66.16%
10teams	210	1600	179	76	66.67%
mod008	6	319	5	10	69.77%
nsrand-ipx	590	4162	226	91	73.17%

Table 2 contains those instances where MIG cuts closed less than 80% of the total integrality gap that can be closed with zero-coefficient cuts. We observe that for the first 16 instances in Table 2, continuing (InitCPalg) beyond MIG cuts closed at least twice as much integrality gap as would have been achieved by using only MIG cuts. For the remaining instances in Table 2, it was not at least a factor of two which was achieved, but still a substantial improvement. The instances marked with an asterisk in Table 2 are instances where we were unable to solve (MIP^c). For those instances, the results are based on the best possible solution we were able to find.

The remaining class of instances are those instances where MIG cuts closed more than 80% of the total integrality gap that can be closed with zero-coefficient cuts. There were 28 of these instances. For these instances, continuing (InitC-Palg) beyond MIG cuts was therefore not beneficial. However, we observe that for all except two of these instances (markshare1 and markshare2), this was because very few zero-coefficient cuts were generated that are not MIG cuts. Detecting that it is not beneficial to continue (InitCPalg) beyond the generation of MIG cuts was therefore done after only very few lattice basis computations.

References

1. Achterberg, T., Koch, T.: MIPLIB 2003. Operations Research Letters 34, 361–372 (2006)
2. Andersen, K., Louveaux, Q., Weismantel, R.: Certificates of linear mixed integer infeasibility. Operations Research Letters 36, 734–738 (2008)
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. Balas, E.: Intersection Cuts - a new type of cutting planes for integer programming. Operations Research 19, 19–39 (1971)
5. Balas, E., Saxena, A.: Optimizing over the split closure. Mathematical Programming, Ser. A 113, 219–240 (2008)
6. Bixby, R.E., Ceria, S., McZeal, C.M., Savelsbergh, M.W.P.: An updated mixed integer programming library: MIPLIB 3. 0. Optima 58, 12–15 (1998)
7. Caprara, A., Letchford, A.: On the separation of split cuts and related inequalities. Mathematical Programming, Ser. A 94, 279–294 (2003)
8. Cook, W.J., Kannan, R., Schrijver, A.: Chvátal closures for mixed integer programming problems. Mathematical Programming 47, 155–174 (1990)
9. Cornuéjols, G., Margot, F.: On the Facets of Mixed Integer Programs with Two Integer Variables and Two Constraints. Mathematical Programming, Ser. A 120, 429–456 (2009)
10. Gomory, R.E.: An algorithm for the mixed integer problem. Technical Report RM-2597, The Rand Corporation (1960a)
11. Nemhauser, G., Wolsey, L.A.: A recursive procedure to generate all cuts for 0-1 mixed integer programs. Mathematical Programming, Ser. A 46, 379–390 (1990)

Prize-Collecting Steiner Network Problems[⋆]

MohammadTaghi Hajiaghayi[1], Rohit Khandekar[2],
Guy Kortsarz[3,⋆⋆], and Zeev Nutov[4]

[1] AT&T Research Lab Research
Hajiagha@research.att.com
[2] IBM T.J.Watson Research Center
rohitk@us.ibm.com
[3] Rutgers University, Camden
guyk@camden.rutgers.edu
[4] The Open University of Israel
nutov@openu.ac.il

Abstract. In the Steiner Network problem we are given a graph G with edge-costs and connectivity requirements r_{uv} between node pairs u, v. The goal is to find a minimum-cost subgraph H of G that contains r_{uv} edge-disjoint paths for all $u, v \in V$. In Prize-Collecting Steiner Network problems we do not need to satisfy all requirements, but are given a *penalty function* for violating the connectivity requirements, and the goal is to find a subgraph H that minimizes the cost plus the penalty. The case when $r_{uv} \in \{0, 1\}$ is the classic Prize-Collecting Steiner Forest problem.

In this paper we present a novel linear programming relaxation for the Prize-Collecting Steiner Network problem, and by rounding it, obtain the first constant-factor approximation algorithm for submodular and monotone non-decreasing penalty functions. In particular, our setting includes all-or-nothing penalty functions, which charge the penalty even if the connectivity requirement is slightly violated; this resolves an open question posed in [SSW07]. We further generalize our results for element-connectivity and node-connectivity.

1 Introduction

Prize-collecting Steiner problems are well-known network design problems with several applications in expanding telecommunications networks (see for example [JMP00, SCRS00]), cost sharing, and Lagrangian relaxation techniques (see e.g. [JV01, CRW01]). A general form of these problems is the Prize-Collecting Steiner Forest problem[1]: given a network (graph) $G = (V, E)$, a set of source-sink pairs $\mathcal{P} = \{\{s_1, t_1\}, \{s_2, t_2\}, \ldots, \{s_k, t_k\}\}$, a non-negative cost function $c : E \to \Re_+$, and a non-negative penalty function $\pi : \mathcal{P} \to \Re_+$, our goal is

⋆ Part of this work was done while the authors were meeting at DIMACS. We would like to thank DIMACS for hospitality.

⋆⋆ Partially supported by NSF Award Grant number 0829959.

[1] In the literature, this problem is also called "prize-collecting generalized Steiner tree".

a minimum-cost way of installing (buying) a set of links (edges) and paying the penalty for those pairs which are not connected via installed links. When all penalties are ∞, the problem is the classic APX-hard Steiner Forest problem, for which the best known approximation ratio is $2 - \frac{2}{n}$ (n is the number of nodes of the graph) due to Agrawal, Klein, and Ravi [AKR95] (see also [GW95] for a more general result and a simpler analysis). The case of Prize-Collecting Steiner Forest problem when all sinks are identical is the classic Prize-Collecting Steiner Tree problem. Bienstock, Goemans, Simchi-Levi, and Williamson [BGSLW93] first considered this problem (based on a problem earlier proposed by Balas [Bal89]) and gave for it a 3-approximation algorithm. The current best ratio for this problem is 1.992 by Archer, Bateni, Hajiaghayi, and Karloff [ABHK09], improving upon a primal-dual $\left(2 - \frac{1}{n-1}\right)$-approximation algorithm of Goemans and Williamson [GW95]. When in addition all penalties are ∞, the problem is the classic Steiner Tree problem, which is known to be APX-hard [BP89] and for which the best approximation ratio is 1.55 [RZ05]. Very recently, Byrka et al. [BGRS10] have announced an improved approximation algorithm for the Steiner tree problem.

The general form of the Prize-Collecting Steiner Forest problem first has been formulated by Hajiaghayi and Jain [HJ06]. They showed how by a primal-dual algorithm to a novel integer programming formulation of the problem with doubly-exponential variables, we can obtain a 3-approximation algorithm for the problem. In addition, they show that the factor 3 in the analysis of their algorithm is tight. However they show how a direct randomized LP-rounding algorithm with approximation factor 2.54 can be obtained for this problem. Their approach has been generalized by Sharma, Swamy, and Williamson [SSW07] for network design problems where violated arbitrary 0-1 connectivity constraints are allowed in exchange for a very general penalty function. The work of Hajiaghayi and Jain has also motivated a game-theoretic version of the problem considered by Gupta et al. [GKL$^+$07].

In this paper, we consider a much more general high-connectivity version of Prize-Collecting Steiner Forest, called Prize-Collecting Steiner Network, in which we are also given connectivity requirements r_{uv} for pairs of nodes u and v and a penalty function in case we do not satisfy all r_{uv}. Our goal is to find a minimum way of constructing a network (graph) in which we connect u and v with $r'_{uv} \leq r_{uv}$ edge-disjoint paths and paying a penalty for all violated connectivity between source-sink pairs. This problem can arise in real-world network design, in which a typical client not only might want to connect to the network but also might want to connect via a few disjoint paths (e.g., to have a higher bandwidth or redundant connections in case of edge failures) and a penalty might be charged if we cannot satisfy its connectivity requirement. When all penalties are ∞, the problem is the classic Steiner Network problem. Improving on a long line of earlier research that applied primal-dual methods, Jain [Jai01] obtained a 2-approximation algorithm for Steiner Network using the iterative rounding method. This algorithm was generalized to so called "element-connectivity" by Fleischer, Jain, and Williamson [FJW01] and by Cheriyan, Vempala, and Vetta

[CVV06]. Recently, some results were obtained for the node-connectivity version; the currently best known ratios for the node-connectivity case are $O(R^3 \log n)$ for general requirements [CK09] and $O(R^2)$ for rooted requirements [Nut09], where $R = \max_{u,v \in V} r_{uv}$ is the maximum requirement. See also the survey by Kortsarz and Nutov [KN07] for various min-cost connectivity problems.

Hajiaghayi and Nasri [HN10] generalize the iterative rounding approach of Jain to Prize-Collecting Steiner Network when there is a separate non-increasing marginal penalty function for each pair u, v whose r_{uv}-connectivity requirement is not satisfied. They obtain an iterative rounding 3-approximation algorithm for this case. For the special case when penalty functions are linear in the violation of the connectivity requirements, Nagarajan, Sharma, and Williamson [NSW08] using Jains iterative rounding algorithm as a black box give a 2.54-factor approximation algorithm. They also generalize the 0-1 requirements of Prize-Collecting Steiner Forest problem introduced by Sharma, Swamy, and Williamson [SSW07] to include general connectivity requirements. Assuming the monotone submodular penalty function of Sharma et al. is generalized to a multiset function that can be decomposed into functions in the same type as that of Sharma et al., they give an $O(\log R)$-approximation algorithm (recall that R is the maximum connectivity requirement). In this algorithm, they assume that we can use each edge possibly many times (without bound). They raise the question whether we can obtain a constant ratio without all these assumptions, when penalty is a submodular multi-set function of the set of disconnected pairs? More importantly they pose as an open problem to design a good approximation algorithm for the all-or-nothing version of penalty functions: penalty functions which charge the penalty even if the connectivity requirement is slightly violated. In this paper, we answer affirmatively all these open problems by proving the first constant factor 2.54-approximation algorithm which is based on a novel LP formulation of the problem. We further generalize our results for element-connectivity and node-connectivity. In fact, for all types of connectivities, we prove a very general result (see Theorem 1) stating that if Steiner Network (the version without penalties) admits an LP-based ρ-approximation algorithm, then the corresponding prize-collecting version admits a $(\rho + 1)$-approximation algorithm.

1.1 Problems We Consider

In this section, we define formally the terms used in the paper. For a subset S of nodes in a graph H, let $\lambda_H^S(u, v)$ denote the *S-connectivity* between u and v in H, namely, the maximum number of edge-disjoint uv-paths in H so that no two of them have a node in $S - \{u, v\}$ in common. In the Generalized Steiner-Network (GSN) problem we are given a graph $G = (V, E)$ with edge-costs $\{c_e \geq 0 \mid e \in E\}$, a node subset $S \subseteq V$, a collection $\{u_1, v_1\}, \ldots, \{u_k, v_k\}$ of node pairs from V, and S-connectivity requirements $r_1, \ldots, r_k$. The goal is to find a minimum cost subgraph H of G so that $\lambda_H^S(u_i, v_i) \geq r_i$ for all i. Extensively studied particular cases of GSN are: the Steiner Network problem, called also Edge-Connectivity GSN ($S = \emptyset$), Node-Connectivity GSN ($S = V$), and Element-Connectivity GSN ($S \cap \{u_i, v_i\} = \emptyset$ for all i). The case of *rooted requirements* is when there is a "root" s that belongs to all pairs $\{u_i, v_i\}$. We consider the following "prize-collecting" version of GSN.

All-or-Nothing Prize Collecting Generalized Steiner Network (PC-GSN):

Instance: A graph $G = (V, E)$ with edge-costs $\{c_e \geq 0 \mid e \in E\}$, $S \subseteq V$, a collection $\{u_1, v_1\}, \ldots, \{u_k, v_k\}$ of node pairs from V, S-connectivity requirements $r_1, \ldots, r_k > 0$, and a penalty function $\pi : 2^{\{1,\ldots,k\}} \rightarrow \Re_+$.

Objective: Find a subgraph H of G that minimizes the *value*

$$\mathtt{val}(H) = c(H) + \pi(\mathtt{unsat}(H))$$

of H, where $\mathtt{unsat}(H) = \{i \mid \lambda^S_H(u_i, v_i) < r_i\}$ is the set of requirements *not* satisfied by H.

We will assume that the penalty function π is given by an evaluation oracle. We will also assume that π is *submodular*, namely, that $\pi(A) + \pi(B) \geq \pi(A \cap B) + \pi(A \cup B)$ for all A, B and that it is *monotone non-decreasing*, namely, $\pi(A) \leq \pi(B)$ for all A, B with $A \subseteq B$. As was mentioned, approximating the edge-connectivity variant of PC-GSN was posed as the main open problem by Nagarajan, Sharma, and Williamson [NSW08]. We resolve this open problem for the submodular function $\mathtt{val}(H)$ considered here.

We next define the second problem we consider.

Generalized Steiner Network with Generalized Penalties (GSN-GP):

Instance: A graph $G = (V, E)$ with edge-costs $\{c_e \geq 0 \mid e \in E\}$, $S \subseteq V$, a collection $\{u_1, v_1\}, \ldots, \{u_k, v_k\}$ of node pairs from V, and non-increasing penalty functions $p_1, \ldots, p_k : \{0, 1, \ldots, n-1\} \rightarrow \Re_+$.

Objective: Find a subgraph H of G that minimizes the *value*

$$\mathtt{val}'(H) = c(H) + \sum_{i=1}^{k} p_i(\lambda^S_H(u_i, v_i)).$$

The above problem captures general penalty functions of the S-connectivity $\lambda^S(u_i, v_i)$ for given pairs $\{u_i, v_i\}$. It is natural to assume that the penalty functions are non-increasing, i.e., we pay less in the objective function if the achieved connectivity is more. This problem was posed as an open question by Nagarajan et al. [NSW08]. In this paper, we use the convention that $p_i(n) = 0$ for all i.

We need some definitions to introduce our results. A pair $T = \{T', T''\}$ of subsets of V is called a *setpair* (of V) if $T' \cap T'' = \emptyset$. Let $K = \{1, \ldots, k\}$. Let $T = \{T', T''\}$ be a setpair of V. We denote by $\delta(T)$ the set of edges in E between T' and T''. For $i \in K$ we use $T \odot (i, S)$ to denote that $|T' \cap \{u_i, v_i\}| = 1$, $|T'' \cap \{u_i, v_i\}| = 1$ and $V \setminus (T' \cup T'') \subseteq S$. While in the case of edge-connectivity a "cut" consists of edges only, in the case of S-connectivity a cut that separates between u and v is "mixed", meaning it may contain both edges in the graph and nodes from S. Note that if $T \odot (i,S)$ then $\delta(T) \cup (V \setminus (T' \cup T''))$ is such a mixed cut that separates between u_i and v_i. Intuitively, Menger's Theorem for S-connectivity (c.f. [KN07]) states that the S-connectivity between u_i and v_i equals the minimum size of such a mixed cut. Formally, for a node pair u_i, v_i of a graph $H = (V,E)$ and $S \subseteq V$ we have:

$$\lambda_H^S(u_i,v_i) = \min_{T \odot (i,S)} (|\delta(T)| + |V \setminus (T' \cup T'')|) = \min_{T \odot (i,S)} (|\delta(T)| + |V| - (|T'| + |T''|))$$

Hence if $\lambda_H^S(u_i,v_i) \geq r_i$ for a graph $H = (V,E)$, then for any setpair T with $T \odot (i,S)$ we must have $|\delta(T)| \geq r_i(T)$, where $r_i(T) = \max\{r_i + |T'| + |T''| - |V|, 0\}$. Consequently, a standard "cut-type" LP-relaxation of the GSN problem is as follows (c.f. [KN07]):

$$\min \left\{ \sum_{e \in E} c_e x_e \mid \sum_{e \in \delta(T)} x_e \geq r_i(T) \ \forall T \odot (i,S), \ \forall i \in K, \ x_e \in [0,1] \ \forall e \right\} . \quad (1)$$

1.2 Our Results

We introduce a novel LP relaxation of the problem which is shown to be better, in terms of the integrality gap, than a "natural" LP relaxation considered in [NSW08]. Using our LP relaxation, we prove the following main result.

Theorem 1. *Suppose that there exists a polynomial time algorithm that computes an integral solution to LP (1) of cost at most ρ times the optimal value of LP (1) for any subset of node pairs. Then* PC-GSN *admits a $(1 - e^{-1/\rho})^{-1}$-approximation algorithm, provided that the penalty function π is submodular and monotone non-decreasing.*

Note that since $1 - \frac{1}{\rho} < e^{-\frac{1}{\rho}} < 1 - \frac{1}{\rho+1}$ holds for $\rho \geq 1$, we have $\rho < (1 - e^{-1/\rho})^{-1} < \rho + 1$.

Let $R = \max_i r_i$ denote the maximum requirement. The best known values of ρ are as follows: 2 for Edge-GSN [Jai01], 2 for Element-GSN [FJW01, CVV06], $O(R^3 \log|V|)$ for Node-GSN [CK09], and $O(R^2)$ for Node-GSN with rooted requirements [Nut09]. Substituting these values in Theorem 1, we obtain:

Corollary 1. PC-GSN *problems admit the following approximation ratios provided that the penalty function π is submodular and monotone non-decreasing:* 2.54 *for edge- and element-connectivity, $O(R^3 \log|V|)$ for node-connectivity, and $O(R^2)$ for node-connectivity with rooted requirements.*

Our results for GSN-GP follow from Corollary 1.

Corollary 2. GSN-GP *problems admit the following approximation ratios:* 2.54 *for edge- and element-connectivity, $O(R^3 \log|V|)$ for node-connectivity, and $O(R^2)$ for node-connectivity with rooted requirements. Here $R = \max_{1 \leq i \leq k} \min\{\lambda \geq 0 \mid p_i(\lambda) = 0\}$.*

Proof. We present an approximation ratio preserving reduction from the GSN-GP problem to the corresponding PC-GSN problem. Given an instance of the GSN-GP problem, we create an instance of the PC-GSN problem as follows. The PC-GSN instance inherits the graph G, its edge-costs, and the set S. Let (u_i, v_i) be a pair in GSN-GP and let $R_i = \min\{\lambda \geq 0 \mid p_i(\lambda) = 0\}$. We introduce R_i copies of this pair, $\{(u_i^1, v_i^1), \ldots, (u_i^{R_i}, v_i^{R_i})\}$, to the set of pairs in the PC-GSN instance. We set the edge-connectivity requirement of a pair (u_i^t, v_i^t) to be t for $1 \leq t \leq R_i$. We also set the penalty function for singleton sets as follows $\pi(\{(u_i^t, v_i^t)\}) = p_i(t-1) - p_i(t)$ for all $1 \leq t \leq R_i$. Finally, we extend this function π to a set of pairs P by *linearity*, i.e., $\pi(P) = \sum_{p \in P} \pi(\{p\})$. Note that such a function π is clearly submodular and monotone non-decreasing.

It is sufficient to show that for any subgraph H of G, its value in the GSN-GP instance equals its value in the PC-GSN instance, i.e., $\texttt{val}(H) = \texttt{val}'(H)$; then we can use the algorithm from Corollary 1 to complete the proof. Fix a pair (u_i, v_i) in the GSN-GP instance. Let $\lambda_H^S(u_i, v_i) = t_i$. Thus the contribution of pair (u_i, v_i) to the objective function $\texttt{val}(H)$ of the GSN-GP instance is $p_i(t_i)$. On the other hand, since π is linear, the total contribution of pairs $\{(u_i^1, v_i^1), \ldots, (u_i^{R_i}, v_i^{R_i})\}$ to the objective function $\texttt{val}'(H)$ of the PC-GSN instance is $\sum_{t=t_i+1}^{R_i} \pi(\{(u_i^t, v_i^t)\}) = \sum_{t=t_i+1}^{R_i} (p_i(t-1) - p_i(t)) = p_i(t_i)$. Note that the pairs (u_i^t, v_i^t) for $1 \leq t \leq t_i$ do not incur any penalty. Summing up over all pairs, we conclude that $\texttt{val}(H) = \texttt{val}'(H)$, as claimed. ∎

2 A New LP Relaxation

We use the following LP-relaxation for the PC-GSN problem. We introduce variables x_e for $e \in E$ ($x_e = 1$ if $e \in H$), $f_{i,e}$ for $i \in K$ and $e \in E$ ($f_{i,e} = 1$ if $i \notin \texttt{unsat}(H)$ and e appears on a chosen set of r_i S-disjoint $\{u_i, v_i\}$-paths in H), and z_I for $I \subseteq K$ ($z_I = 1$ if $I = \texttt{unsat}(H)$).

$$\begin{array}{lrll} \text{Minimize} & \multicolumn{3}{l}{\sum_{e \in E} c_e x_e + \sum_{I \subseteq K} \pi(I) z_I} \\ \\ \text{Subject to} & \sum_{e \in \delta(T)} f_{i,e} \geq (1 - \sum_{I: i \in I} z_I) r_i(T) & \forall i \;\; \forall T \odot (i, S) & \\ & f_{i,e} \leq 1 - \sum_{I: i \in I} z_I & \forall i \;\; \forall e & \\ & x_e \geq f_{i,e} & \forall i \;\; \forall e & (2) \\ & \sum_{I \subseteq K} z_I = 1 & & \\ & x_e, f_{i,e}, z_I \in [0,1] & \forall i \;\; \forall e \;\; \forall I & \end{array}$$

We first prove that (2) is a valid LP-relaxation of the PC-GSN problem.

Lemma 1. *The optimal value of LP (2) is at most the optimal solution value to the* PC-GSN *problem. Moreover, if π is monotone non-decreasing, the optimum solution value to the* PC-GSN *problem is at most the value of the optimum* integral *solution of LP (2).*

Proof. Given a feasible solution H to the PC-GSN problem define a feasible solution to LP (2) as follows. Let $x_e = 1$ if $e \in H$ and $x_e = 0$ otherwise. Let $z_I = 1$ if $I = \texttt{unsat}(H)$ and $z_I = 0$ otherwise. For each $i \in \texttt{unsat}(H)$ set $f_{i,e} = 0$ for all $e \in E$, while for $i \notin \texttt{unsat}(H)$ the variables $f_{i,e}$ take values as follows: fix a set of r_i pairwise S-disjoint $\{u_i, v_i\}$-paths, and let $f_{i,e} = 1$ if e belongs to one of these paths and $f_{i,e} = 0$ otherwise. The defined solution is feasible for LP (2): the first set of constraints are satisfied by Menger's Theorem for S-connectivity, while the remaining constraints are satisfied by the above definition of variables. It is also easy to see that the above solution has value exactly $\texttt{val}(H)$.

If π is monotone non-decreasing, we prove that for any *integral* solution $\{x_e, f_{i,e}, z_I\}$ to (2), the graph H with edge-set $\{e \in E \mid x_e = 1\}$ has $\texttt{val}(H)$ at most the value of the solution $\{x_e, f_{i,e}, z_I\}$. To see this, first note that there is a unique set $I \subseteq K$ with $z_I = 1$, since the variables z_I are integral and $\sum_{I \subseteq K} z_I = 1$. Now consider an index $i \notin I$. Since $\sum_{I: i \in I} z_I = 0$, we have $\sum_{e \in \delta(T)} x_e \geq \sum_{e \in \delta(T)} f_{i,e} \geq r_i(T)$ for all $T \odot (i, S)$. This implies that $i \notin \texttt{unsat}(H)$, by Menger's Theorem for S-connectivity. Consequently, $\texttt{unsat}(H) \subseteq I$, hence $\pi(\texttt{unsat}(H)) \leq \pi(I)$ by the monotonicity of π. Thus $\texttt{val}(H) = c(H) + \pi(\texttt{unsat}(H)) \leq \sum_{e \in E} c_e x_e + \sum_{I \subseteq K} \pi(I) z_I$ and the lemma follows. ∎

2.1 Why Does a "Natural" LP Relaxation Not Work?

One may be tempted to consider a natural LP without using the flow variables $f_{i,e}$, namely, the LP obtained from LP (2) by replacing the the first three sets of constraints by the set of constraints

$$\sum_{e \in \delta(T)} x_e \geq (1 - \sum_{I: i \in I} z_I) r_i(T)$$

for all i and $T \odot (i, S)$. Here is an example demonstrating that the integrality gap of this LP can be as large as $R = \max_i r_i$ even for edge-connectivity. Let G consist of $R - 1$ edge-disjoint paths between two nodes s and t. All the edges have cost 0. There is only one pair $\{u_1, v_1\} = \{s, t\}$ that has requirement $r_1 = R$ and penalty $\pi(\{1\}) = 1$. Let $\pi(\emptyset) = 0$. Clearly, π is submodular and monotone non-decreasing. We have $S = \emptyset$. No integral solution can satisfy the requirement r_1, hence an optimal integral solution pays the penalty $\pi(\{1\})$ and has value 1. A feasible fractional solution (without the flow variables) sets $x_e = 1$ for all e, and sets $z_{\{1\}} = 1/R$, $z_\emptyset = 1 - 1/R$. The new set of constraints is satisfied since $\sum_{e \in \delta(T)} x_e \geq (1 - 1/R) \cdot R = (1 - z_{\{1\}}) r_1(T)$ for any $\{s, t\}$-cut T. Thus the optimal LP-value is at most $1/R$, giving a gap of at least R.

With flow variables, however, we have an upper bound $f_{1,e} \leq 1 - z_{\{1\}}$. Since there is an $\{s, t\}$-cut T with $|\delta(T)| = R - 1$, we cannot satisfy the constraints $\sum_{e \in \delta(T)} f_{1,e} \geq (1 - z_{\{1\}}) r_1(T)$ and $f_{1,e} \leq 1 - z_{\{1\}}$ simultaneously unless we set $z_{\{1\}} = 1$. Thus in this case, our LP (2) with flow variables has the same optimal value of as the integral optimum.

2.2 Some Technical Results Regarding LP (2)

We will prove the following two statements that together imply Theorem 1.

Lemma 2. *Any basic feasible solution to (2) has a polynomial number of non-zero variables. Furthermore, an optimal basic solution to (2) (the non-zero entries) can be computed in polynomial time.*

Lemma 3. *Suppose that there exists a polynomial time algorithm that computes an integral solution to LP (1) of cost at most ρ times the optimal value of LP (1) for any subset of node pairs. Then there exists a polynomial time algorithm that given a feasible solution to (2) computes as a solution to* PC-GSN *a subgraph H of G so that $\texttt{val}(H) = c(H) + \pi(\texttt{unsat}(H))$ is at most $(1 - e^{-1/\rho})^{-1}$ times the value of this solution, assuming π is submodular and monotone non-decreasing.*

Before proving these lemmas, we prove some useful results. The following statement can be deduced from a theorem of Edmonds for polymatroids (c.f. [KV02, Chapter 14.2]), as the dual LP $\textsc{d}(\gamma)$ in the lemma seeks to optimize a linear function over a polymatroid. We provide a direct proof for completeness of exposition.

Lemma 4. *Let $\gamma \in [0,1]^k$ be a vector. Consider a primal LP*

$$\textsc{p}(\gamma) := \min \left\{ \sum_{I \subseteq K} \pi(I) z_I \mid \sum_{I : i \in I} z_I \geq \gamma_i \ \forall i \in K, z_I \geq 0 \ \forall I \subseteq K \right\}$$

and its dual LP

$$\textsc{d}(\gamma) := \max \left\{ \sum_{i \in K} \gamma_i y_i \mid \sum_{i \in I} y_i \leq \pi(I) \ \forall I \subseteq K, y_i \geq 0 \ \forall i \in K \right\}.$$

Let σ be a permutation of K such that $\gamma_{\sigma(1)} \leq \gamma_{\sigma(2)} \leq \cdots \leq \gamma_{\sigma(k)}$. Let us also use the notation that $\gamma_{\sigma(0)} = 0$. The optimum solutions to $\textsc{p}(\gamma)$ and $\textsc{d}(\gamma)$ respectively are given by

$$z_I = \begin{cases} \gamma_{\sigma(i)} - \gamma_{\sigma(i-1)}, & \textit{for } I = \{\sigma(i), \ldots, \sigma(k)\}, i \in K \\ 0, & \textit{otherwise}; \end{cases}$$

and

$$y_{\sigma(i)} = \pi(\{\sigma(i), \ldots, \sigma(k)\}) - \pi(\{\sigma(i+1), \ldots, \sigma(k)\}), \ \textit{for } i \in K.$$

Proof. To simplify the notation, we assume without loss of generality that $\gamma_1 \leq \gamma_2 \leq \cdots \leq \gamma_k$, i.e., that σ is the identity permutation.

We argue that the above defined $\{z_I\}$ and $\{y_i\}$ form feasible solutions to the primal and dual LPs respectively. Note that $z_I \geq 0$ for all I and $\sum_{I : i \in I} z_I = \sum_{j=1}^{i} (\gamma_j - \gamma_{j-1}) = \gamma_i$ for all i. Since π is monotone non-decreasing, the above defined y_i satisfy $y_i \geq 0$ for all i. Now fix $I \subseteq K$. Let $I = \{i_1, \ldots, i_p\}$ where $i_1 < \cdots < i_p$. Therefore

$$\begin{aligned}\sum_{i \in I} y_i \;=\; \sum_{j=1}^{p} y_{i_j} &= \sum_{j=1}^{p} [\pi(\{i_j, \ldots, k\}) - \pi(\{i_j + 1, \ldots, k\})] \\ &\leq \sum_{j=1}^{p} [\pi(\{i_j, i_{j+1}, \ldots, i_p\}) - \pi(\{i_{j+1}, i_{j+2}, \ldots, i_p\})] \\ &= \pi(\{i_1, \ldots, i_p\}) \;=\; \pi(I).\end{aligned}$$

The above inequality holds because of the submodularity of π. Next observe that the solutions $\{z_I\}$ and $\{y_i\}$ satisfy

$$\begin{aligned}\sum_{I} \pi(I) z_I &= \sum_{i=1}^{k} \pi(\{i, \ldots, k\}) \cdot (\gamma_i - \gamma_{i-1}) \\ &= \sum_{i=1}^{k} \gamma_i \cdot (\pi(\{i, \ldots, k\}) - \pi(\{i+1, \ldots, k\})) = \sum_{i=1}^{k} \gamma_i \cdot y_i.\end{aligned}$$

Thus from weak LP duality, they in fact form optimum solutions to primal and dual LPs respectively. ∎

Recall that a sub-gradient of a convex function $g : \Re^k \to \Re$ at a point $\gamma \in \Re^k$ is a vector $d \in \Re^k$ such that for any $\gamma' \in \Re^k$, we have $g(\gamma') - g(\gamma) \geq d \cdot (\gamma' - \gamma)$. For a differentiable convex function g, the sub-gradient corresponds to gradient ∇g. The function $\textsc{p}(\gamma)$ defined in Lemma 4 is essentially Lovasz's continuous extension of the submodular function π. The fact that $\textsc{p}$ is convex and its subgradient can be computed efficiently is given in [Fuj05]. We provide a full proof for completeness of exposition.

Lemma 5. *The function* $\textsc{p}(\gamma)$ *in Lemma 4 is convex and given* $\gamma \in [0,1]^k$*, both* $\textsc{p}(\gamma)$ *and its sub-gradient* $\nabla\textsc{p}(\gamma)$ *can be computed in polynomial time.*

Proof. We first prove that $\textsc{p}$ is convex. Fix $\gamma_1, \gamma_2 \in [0,1]^k$ and $\alpha \in [0,1]$. To show that $\textsc{p}$ is convex, we will show $\textsc{p}(\alpha\gamma_1 + (1-\alpha)\gamma_2) \leq \alpha\textsc{p}(\gamma_1) + (1-\alpha)\textsc{p}(\gamma_2)$. Let $\{z_I^1\}$ and $\{z_I^2\}$ be the optimum solutions of the primal LP defining $\textsc{p}$ for γ_1 and γ_2 respectively. Note that the solution $\{\alpha z_I^1 + (1-\alpha) z_I^2\}$ is feasible for this LP for $\gamma = \alpha\gamma_1 + (1-\alpha)\gamma_2$. Thus the optimum solution has value not greater than the value of this solution which is $\alpha\textsc{p}(\gamma_1) + (1-\alpha)\textsc{p}(\gamma_2)$.

From Lemma 4, it is clear that given $\gamma \in [0,1]^k$, the value $\textsc{p}(\gamma)$ can be computed in polynomial time. Lemma 4 also implies that the optimum dual solution $y^* = (y_1^*, \ldots, y_k^*) \in \Re_+^k$ can be computed in polynomial time. We now argue that y^* is a sub-gradient of $\textsc{p}$ at γ. Fix any $\gamma' \in \Re^k$. First note that, from LP duality, $\textsc{p}(\gamma) = y^* \cdot \gamma$. Thus we have

$$\textsc{p}(\gamma) + y^* \cdot (\gamma' - \gamma) = y^* \cdot \gamma + y^* \cdot (\gamma' - \gamma) = y^* \cdot \gamma' \leq \textsc{p}(\gamma').$$

The last inequality holds from weak LP duality since y^* is a feasible solution for the dual LP $\textsc{d}(\gamma')$ as well. The lemma follows. ∎

3 Proof of Lemma 3

We now describe how to round LP (2) solutions to obtain a $(\rho+1)$-approximation for PC-GSN. Later we show how to improve it to $(1-e^{-1/\rho})^{-1}$. Let $\{x_e^*, f_{i,e}^*, z_I^*\}$ be a feasible solution to LP (2). Let $\alpha \in (0,1)$ be a parameter to be fixed later. We partition the requirements into two classes: we call a requirement $i \in K$ *good* if $\sum_{I:i\in I} z_I^* \leq \alpha$ and *bad* otherwise. Let K_g denote the set of good requirements. The following statement shows how to satisfy the good requirements.

Lemma 6. *There exists a polynomial time algorithm that computes a subgraph H of G of cost $c(H) \leq \frac{\rho}{1-\alpha} \cdot \sum_e c_e x_e^*$ that satisfies all good requirements.*

Proof. Consider the LP-relaxation (1) of the GSN problem with good requirements only, with K replaced by K_g; namely, we seek a minimum cost subgraph H of G that satisfies the set K_g of good requirements. We claim that $x_e^{**} = \min\{1, x_e^*/(1-\alpha)\}$ for each $e \in E$ is a feasible solution to LP (1). Thus the optimum value of LP (1) is at most $\sum_{e\in E} c_e x_e^{**}$. Consequently, using the algorithm that computes an integral solution to LP (1) of cost at most ρ times the optimal value of LP (1), we can construct a subgraph H that satisfies all good requirements and has cost at most $c(H) \leq \rho \sum_{e\in E} c_e x_e^{**} \leq \frac{\rho}{1-\alpha}\sum_e c_e x_e^*$, as desired.

We now show that $\{x_e^{**}\}$ is a feasible solution to LP (1), namely, that $\sum_{e\in\delta(T)} x_e^{**} \geq r_i(T)$ for any $i \in K_g$ and any $T \odot (i,S)$. Let $i \in K_g$ and let $\zeta_i = 1 - \sum_{I:i\in I} z_I^*$. Note that $\zeta_i \geq 1-\alpha$, by the definition of K_g. By the second and the third sets of constraints in LP (2), for every $e \in E$ we have $\min\{\zeta_i, x_e^*\} \geq f_{i,e}^*$. Thus we obtain: $x_e^{**} = \min\left\{1, \frac{x_e^*}{1-\alpha}\right\} = \frac{1}{\zeta_i}\min\left\{\zeta_i, \frac{\zeta_i}{1-\alpha}x_e^*\right\} \geq \frac{1}{\zeta_i}\min\{\zeta_i, x_e^*\} \geq \frac{f_{i,e}^*}{\zeta_i} = \frac{f_{i,e}^*}{1-\sum_{I:i\in I} z_I^*}$. Consequently, combining with the first set of constraints in LP (2), for any $T \odot (i,S)$ we obtain that $\sum_{e\in\delta(T)} x_e^{**} \geq \frac{\sum_{e\in\delta(T)} f_{i,e}^*}{1-\sum_{I:i\in I} z_I^*} \geq r_i(T)$. ∎

Let H be as in Lemma 6, and recall that $\texttt{unsat}(H)$ denotes the set of requirements not satisfied by H. Clearly each requirement $i \in \texttt{unsat}(H)$ is bad. The following lemma bounds the total penalty we pay for $\texttt{unsat}(H)$.

Lemma 7. $\pi(\texttt{unsat}(H)) \leq \frac{1}{\alpha} \cdot \sum_I \pi(I) z_I^*$.

Proof. Define $\gamma \in [0,1]^k$ as follows: $\gamma_i = 1$ if $i \in \texttt{unsat}(H)$ and 0 otherwise. Now consider LP $\mathrm{P}(\gamma)$ defined in Lemma 4. Since each $i \in \texttt{unsat}(H)$ is bad, from the definition of bad requirements, it is clear that $\{z_I^*/\alpha\}$ is a feasible solution to LP $\mathrm{P}(\gamma)$. Furthermore, from Lemma 4, the solution $\{z_I\}$ defined as $z_I = 1$ if $I = \texttt{unsat}(H)$ and 0 otherwise is the optimum solution to $\mathrm{P}(\gamma)$. The cost of this solution, $\pi(\texttt{unsat}(H))$, is therefore at most the cost of the feasible solution $\{z_I^*/\alpha\}$ which is $\frac{1}{\alpha} \cdot \sum_I \pi(I) z_I^*$. The lemma thus follows. ∎

Combining Lemmas 6 and 7, we obtain $\max\{\frac{\rho}{1-\alpha}, \frac{1}{\alpha}\}$-approximation. If we substitute $\alpha = 1/(\rho+1)$, we obtain a $(\rho+1)$-approximation for PC-GSN.

Improving the Approximation to $(1 - e^{-1/\rho})^{-1}$. We use a technique introduced by Goemans as follows. We pick α uniformly at random from the interval $(0, \beta]$ where $\beta = 1 - e^{-1/\rho}$. From Lemmas 6 and 7, the expected cost of the solution is at most

$$\mathbb{E}_\alpha\left[\frac{\rho}{1-\alpha}\right] \cdot \sum_{e\in E} c_e x_e^* + \mathbb{E}_\alpha\left[\pi(\mathtt{unsat}(H))\right]. \tag{3}$$

To complete the proof of $\frac{1}{\beta}$-approximation, we now argue that the above expectation is at most $\frac{1}{\beta} \cdot \sum_{e\in E}(c_e x_e^* + \sum_I \pi(I) z_I^*)$.

Since $\mathbb{E}_\alpha\left[\frac{\rho}{1-\alpha}\right] = \frac{1}{\beta}$, the first term in (3) is at most $\frac{1}{\beta} \cdot \sum_{e\in E} c_e x_e^*$. Since $\mathtt{unsat}(H) \subseteq \{i \mid \sum_{I:i\in I} z_I^* \geq \alpha\}$ and since π is monotone non-decreasing, the second term in (3) is at most $\mathbb{E}_\alpha\left[\pi\left(\{i \mid \sum_{I:i\in I} z_I^* \geq \alpha\}\right)\right]$. Lemma 8 bounds this quantity as follows. The ideas used here are also presented in Sharma et al. [SSW07].

Lemma 8. *We have*

$$\mathbb{E}_\alpha\left[\pi\left(\{i \mid \sum_{I:i\in I} z_I^* \geq \alpha\}\right)\right] \leq \frac{1}{\beta} \cdot \sum_I \pi(I) z_I^*. \tag{4}$$

Proof. Let $\gamma_i = \sum_{I:i\in I} z_I^*$ for all $i \in K$. Let us, without loss of generality, order the elements $i \in K$ such that $\gamma_1 \leq \gamma_2 \leq \cdots \leq \gamma_k$. We also use the notation $\gamma_0 = 0$. Note that $\{z_I^*\}$ forms a feasible solution to the primal LP $\mathrm{P}(\gamma)$ given in Lemma 4. Therefore, from Lemma 4, its objective value is at least that of the optimum solution:

$$\sum_I \pi(I) z_I^* \;\geq\; \sum_{i=1}^{k} \left[(\gamma_i - \gamma_{i-1}) \cdot \pi(\{i, \ldots, k\})\right]. \tag{5}$$

We now observe that the LHS of (4) can be expressed as follows. Since α is picked uniformly at random from $(0, \beta]$, we have that for all $1 \leq i \leq k$, with probability at most $\frac{\gamma_i - \gamma_{i-1}}{\beta}$, the random variable α lies in the interval $(\gamma_{i-1}, \gamma_i]$. When this event happens, we get that $\{i' \mid \sum_{I:i'\in I} z_I^* \geq \alpha\} = \{i' \mid \gamma_{i'} \geq \alpha\} = \{i, \ldots, k\}$. Thus the expectation in LHS of (4) is at most

$$\sum_{i=1}^{k}\left[\frac{\gamma_i - \gamma_{i-1}}{\beta} \cdot \pi(\{i, \ldots, k\})\right]. \tag{6}$$

From expressions (5) and (6), the lemma follows. ∎

Thus the proof of $(1 - e^{-1/\rho})^{-1}$-approximation is complete. It is worth mentioning so far in this section we obtain a solution with a bound on its expected cost. However, the choice of α can be simply derandomized by trying out all the breakpoints where a good demand pair becomes a bad one (plus 0 and β).

4 Proof of Lemma 2

We next show that even if LP (2) has exponential number of variables and constraints, the following lemma holds.

Lemma 9. *Any basic feasible solution to LP (2) has a polynomial number of non-zero variables.*

Proof. Fix a basic feasible solution $\{x_e^*, f_{i,e}^*, z_i^*\}$ to (2). For $i \in K$, let

$$\gamma_i = 1 - \frac{\min_{T:T\odot i} \sum_{e\in\delta(T)} f_{i,e}^*}{r_i} \qquad \text{and} \qquad \gamma_i' = 1 - \max_e f_{i,e}^* \ .$$

Now fix the values of variables $\{x_e, f_{i,e}\}$ to $\{x_e^*, f_{i,e}^*\}$ and project the LP (2) onto variables $\{z_I\}$ as follows.

$$\sum_{e\in E} c_e x_e^* + \min \left\{ \sum_{I\subseteq K} \pi(I) z_I \mid \sum_{I\subseteq K} z_I = 1, \gamma_i \le \sum_{I:i\in I} z_I \le \gamma_i' \ \forall i \in K, z_I \ge 0 \ \forall I \subseteq K \right\}. \qquad (7)$$

Since $\{x_e^*, f_{i,e}^*, z_i^*\}$ is a basic feasible solution to (2), it cannot be written as a convex combination of two distinct feasible solutions to (2). Thus we get that $\{z_I^*\}$ cannot be written as a convex combination of two distinct feasible solutions to (7), and hence it forms a basic feasible solution to (7). Since there are $1+2|K|$ non-trivial constraints in (7), at most $1+2|K|$ variables z_I can be non-zero in any basic feasible solution of (7). Thus the lemma follows. ∎

We prove that LP (2) can be solved in polynomial time. Introduce variables $\gamma \in [0,1]^k$ and obtain the following program (the function P is as in Lemma 4).

$$\begin{array}{lrll}
\text{Minimize} & \sum_{e\in E} c_e x_e + \mathrm{P}(\gamma) & & \\
\text{Subject to} & \sum_{e\in\delta(T)} f_{i,e} \ge (1-\gamma_i) r_i(T) & \forall i & \forall T \odot (i,S) \\
& f_{i,e} \le 1-\gamma_i & \forall i & \forall e \\
& x_e \ge f_{i,e} & \forall i & \forall e \\
& x_e, f_{i,e}, \gamma_i \in [0,1] & \forall i & \forall e
\end{array} \qquad (8)$$

It is clear that solving (8) is enough to solve (2). Now note that this is a convex program since P is a convex function. To solve (8), we convert its objective function into a constraint $\sum_{e\in E} c_e x_e + \mathrm{P}(\gamma) \le \mathrm{OPT}$ where OPT is the target objective value and thus reduce it to a feasibility problem. Now to find a feasible solution using an ellipsoid algorithm, we need to show a polynomial time separation oracle. The separation oracle for the first set of constraints can be reduced to a minimum u-v cut problem using standard techniques. The separation oracle for the remaining constraints is trivial.

The separation oracle for the objective function is as follows. Given a point $(x,\gamma) = \{x_e, \gamma_i\}$ that satisfies $\sum_{e\in E} c_e x_e + \mathrm{P}(\gamma) > \mathrm{OPT}$, we compute a sub-gradient of the function $\sum_{e\in E} c_e x_e + \textsc{p}(\gamma)$ w.r.t. variables $\{x_e, \gamma_i\}$. The sub-gradient of $\sum_{e\in E} c_e x_e$ w.r.t. x is simply the cost vector c. The sub-gradient of $\textsc{p}(\gamma)$ w.r.t. γ is computed using Lemma 5, denote it by $y \in \Re^k_+$. From the definition of sub-gradient, we have that the sub-gradient (c, y) to the objective function at point (x, γ) satisfies

$$\left(\sum_{e\in E} c_e x'_e + \textsc{p}(\gamma')\right) - \left(\sum_{e\in E} c_e x_e + \textsc{p}(\gamma)\right) \geq (c, y) \cdot ((x', \gamma') - (x, \gamma))\,.$$

Now fix any feasible solution (x^*, γ^*), i.e., the one that satisfies $\sum_{e\in E} c_e x^*_e + \textsc{p}(\gamma^*) \leq \textsc{opt}$. Substituting $(x', \gamma') = (x^*, \gamma^*)$ in the above equation we get,

$$\begin{aligned} 0 \;=\; \textsc{opt} - \textsc{opt} &> \left(\sum_{e\in E} c_e x^*_e + \textsc{p}(\gamma^*)\right) - \left(\sum_{e\in E} c_e x_e + \textsc{p}(\gamma)\right) \\ &\geq (c, y) \cdot (x^*, \gamma^*) - (c, y) \cdot (x, \gamma). \end{aligned}$$

Thus (c, y) defines a separating hyperplane between the point (x, γ) and any point (x^*, γ^*) that satisfies $\sum_{e\in E} c_e x^*_e + \textsc{p}(\gamma^*) \leq \textsc{opt}$. Hence we have a polynomial time separation oracle for the objective function as well.

Thus we can solve (8) using the ellipsoid algorithm. The proof of Lemma 2 is hence complete.

References

[ABHK09] Archer, A., Bateni, M., Hajiaghayi, M., Karloff, H.: A technique for improving approximation algorithms for prize-collecting problems. In: Proc. 50th IEEE Symp. on Foundations of Computer Science, FOCS (2009)

[AKR95] Agrawal, A., Klein, P., Ravi, R.: When trees collide: an approximation algorithm for the generalized Steiner problem on networks. SIAM J. Comput. 24(3), 440–456 (1995)

[Bal89] Balas, E.: The prize collecting traveling salesman problem. Networks 19(6), 621–636 (1989)

[BGRS10] Byrka, J., Grandoni, F., Rothvoss, T., Sanita, L.: An improved lp-based approximation for steiner tree. In: Proceedings of the 42nd annual ACM Symposium on Theory of computing, STOC (2010)

[BGSLW93] Bienstock, D., Goemans, M., Simchi-Levi, D., Williamson, D.: A note on the prize collecting traveling salesman problem. Math. Programming 59(3, Ser. A), 413–420 (1993)

[BP89] Bern, M., Plassmann, P.: The Steiner problem with edge lengths 1 and 2. Information Processing Letters 32, 171–176 (1989)

[CK09] Chuzhoy, J., Khanna, S.: An $O(k^3 \log n)$-approximation algorithms for vertex-connectivity network design. In: Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science, FOCS (2009)

[CRW01] Chudak, F., Roughgarden, T., Williamson, D.: Approximate k-MSTs and k-Steiner trees via the primal-dual method and Lagrangean relaxation. In: Aardal, K., Gerards, B. (eds.) IPCO 2001. LNCS, vol. 2081, pp. 60–70. Springer, Heidelberg (2001)

[CVV06] Cheriyan, J., Vempala, S., Vetta, A.: Network design via iterative rounding of setpair relaxations. Combinatorica 26(3), 255–275 (2006)

[FJW01] Fleischer, L., Jain, K., Williamson, D.: An iterative rounding 2-approximation algorithm for the element connectivity problem. In: Proc. of the 42nd IEEE Symp. on Foundations of Computer Science (FOCS), pp. 339–347 (2001)

[Fuj05] Fujishige, S.: Submodular functions and optimization. Elsevier, Amsterdam (2005)

[GKL$^+$07] Gupta, A., Könemann, J., Leonardi, S., Ravi, R., Schäfer, G.: An efficient cost-sharing mechanism for the prize-collecting steiner forest problem. In: Proc. of the 18th ACM-SIAM Symposium on Discrete algorithms (SODA), pp. 1153–1162 (2007)

[GW95] Goemans, M., Williamson, D.: A general approximation technique for constrained forest problems. SIAM J. Comput. 24(2), 296–317 (1995)

[HJ06] Hajiaghayi, M., Jain, K.: The prize-collecting generalized Steiner tree problem via a new approach of primal-dual schema. In: Proc. of the 17th ACM-SIAM Symp. on Discrete Algorithms (SODA), pp. 631–640 (2006)

[HN10] Hajiahayi, M., Nasri, A.: Prize-collecting Steiner networks via iterative rounding. In: LATIN (to appear, 2010)

[Jai01] Jain, K.: A factor 2 approximation algorithm for the generalized Steiner network problem. Combinatorica 21(1), 39–60 (2001)

[JMP00] Johnson, D., Minkoff, M., Phillips, S.: The prize collecting Steiner tree problem: theory and practice. In: Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 760–769 (2000)

[JV01] Jain, K., Vazirani, V.: Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation. J. ACM 48(2), 274–296 (2001)

[KN07] Kortsarz, G., Nutov, Z.: Approximating minimum cost connectivity problems. In: Gonzales, T.F. (ed.) Approximation Algorithms and Metahueristics, ch. 58. CRC Press, Boca Raton (2007)

[KV02] Korte, B., Vygen, J.: Combinatorial Optimization: Theory and Algorithms. Springer, Berlin (2002)

[NSW08] Nagarajan, C., Sharma, Y., Williamson, D.: Approximation algorithms for prize-collecting network design problems with general connectivity requirements. In: Bampis, E., Skutella, M. (eds.) WAOA 2008. LNCS, vol. 5426, pp. 174–187. Springer, Heidelberg (2009)

[Nut09] Nutov, Z.: Approximating minimum cost connectivity problems via uncrossable bifamilies and spider-cover decompositions. In: Proc. of the 50th IEEE Symposium on Foundations of Computer Science, FOCS (2009)

[RZ05] Robins, G., Zelikovsky, A.: Tighter bounds for graph Steiner tree approximation. SIAM J. on Discrete Mathematics 19(1), 122–134 (2005)

[SCRS00] Salman, F., Cheriyan, J., Ravi, R., Subramanian, S.: Approximating the single-sink link-installation problem in network design. SIAM J. on Optimization 11(3), 595–610 (2000)

[SSW07] Sharma, Y., Swamy, C., Williamson, D.: Approximation algorithms for prize collecting forest problems with submodular penalty functions. In: Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1275–1284 (2007)

On Lifting Integer Variables in Minimal Inequalities

Amitabh Basu[1,⋆], Manoel Campelo[2,⋆⋆], Michele Conforti[3], Gérard Cornuéjols[1,4,⋆⋆⋆], and Giacomo Zambelli[3]

[1] Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA 15213
[2] Departamento de Estatística e Matemática Aplicada, Universidade Federal do Ceará, Brazil
[3] Dipartimento di Matematica Pura e Applicata, Universitá di Padova, Via Trieste 63, 35121 Padova, Italy
[4] LIF, Faculté des Sciences de Luminy, Université de Marseille, France

Abstract. This paper contributes to the theory of cutting planes for mixed integer linear programs (MILPs). Minimal valid inequalities are well understood for a relaxation of an MILP in tableau form where all the nonbasic variables are continuous. In this paper we study lifting functions for the nonbasic *integer* variables starting from such minimal valid inequalities. We characterize precisely when the lifted coefficient is equal to the coefficient of the corresponding continuous variable in every minimal lifting. The answer is a nonconvex region that can be obtained as the union of convex polyhedra.

1 Introduction

There has been a renewed interest recently in the study of cutting planes for general mixed integer linear programs (MILPs) that cut off a basic solution of the linear programming relaxation. More precisely, consider a mixed integer linear set in which the variables are partitioned into a basic set B and a nonbasic set N, and $K \subseteq B \cup N$ indexes the integer variables:

$$\begin{array}{ll} x_i = f_i - \sum_{j\in N} a_{ij}x_j & \text{for } i \in B \\ x \geq 0 & \\ x_k \in \mathbb{Z} & \text{for } k \in K. \end{array} \tag{1}$$

Let X be the relaxation of (1) obtained by dropping the nonnegativity restriction on all the basic variables x_i, $i \in B$. The convex hull of X is the *corner polyhedron* introduced by Gomory [11] (see also [12]). Note that, for any $i \in B \setminus K$, the equation $x_i = f_i - \sum_{j\in N} a_{ij}x_j$ can be removed from the formulation of X since it just defines variable x_i. Therefore, throughout the paper, we will assume

⋆ Supported by a Mellon Fellowship and NSF grant CMMI0653419.
⋆⋆ Partially supported by CNPq Brazil.
⋆⋆⋆ Supported by NSF grant CMMI0653419, ONR grant N00014-09-1-0133 and ANR grant ANR06-BLAN-0375.

$B \subseteq K$, i.e. all basic variables are integer. Andersen, Louveaux, Weismantel and Wolsey [1] studied the corner polyhedron when $|B| = 2$ and $B = K$, i.e. all nonbasic variables are continuous. They give a complete characterization of the corner polyhedron using intersection cuts (Balas [2]) arising from splits, triangles and quadrilaterals. This very elegant result has been extended to $|B| > 2$ and $B = K$ by showing a correspondence between minimal valid inequalities and maximal lattice-free convex sets [5], [7]. These results and their extensions [6], [10] are best described in an infinite model, which we motivate next.

A classical family of cutting planes for (1) is that of Gomory mixed integer cuts. For a given row of the tableau, the Gomory mixed integer cut is of the form $\sum_{j \in N \setminus K} \psi(a_{ij})x_j + \sum_{j \in N \cap K} \pi(a_{ij})x_j \geq 1$ where ψ and π are functions given by simple formulas. A nice feature of the Gomory mixed integer cut is that, for fixed f_i, the same functions ψ, π are used for any possible choice of the a_{ij}s in (1). It is well known that the Gomory mixed integer cuts are also valid for X. More generally, let a^j be the vector with entries a_{ij}, $i \in B$; we are interested in pairs (ψ, π) of functions such that the inequality $\sum_{j \in N \setminus K} \psi(a^j)x_j + \sum_{j \in N \cap K} \pi(a^j)x_j \geq 1$ is valid for X for any possible choice of the nonbasic coefficients a_{ij}. Since we are interested in nonredundant inequalities, we can assume that the function (ψ, π) is pointwise minimal. While a general characterization of minimal valid functions seems hopeless (see for example [4]), when $N \cap K = \emptyset$ the minimal valid functions ψ are well understood in terms of maximal lattice-free convex sets, as already mentioned. Starting from such a minimal valid function ψ, an interesting question is how to generate a function π such that (ψ, π) is valid and minimal. Recent papers [8], [9] study when such a function π is unique. Here we prove a theorem that generalizes and unifies results from these two papers.

In order to formalize the concept of valid function (ψ, π), we introduce the following infinite model. In the setting below, we also allow further linear constraints on the basic variables. Let S be the set of integral points in some rational polyhedron in $\mathbb{R}^n$ such that $\dim(S) = n$ (for example, S could be the set of nonnegative integer points). Let $f \in \mathbb{R}^n \setminus S$. Consider the following semi-infinite relaxation of (1), introduced in [10].

$$\begin{aligned} x = f + \sum_{r \in \mathbb{R}^n} r s_r + \sum_{r \in \mathbb{R}^n} r y_r,& \\ x \in S,& \\ s_r \in \mathbb{R}_+, \forall r \in \mathbb{R}^n,& \\ y_r \in \mathbb{Z}_+, \forall r \in \mathbb{R}^n,& \\ s, y \text{ have finite support}& \end{aligned} \tag{2}$$

where the nonbasic continuous variables have been renamed s and the nonbasic integer variables have been renamed y. Given two functions $\psi, \pi : \mathbb{R}^n \to \mathbb{R}$, (ψ, π) is said to be *valid* for (2) if the inequality $\sum_{r \in \mathbb{R}^n} \psi(r)s_r + \sum_{r \in \mathbb{R}^n} \pi(r)y_r \geq 1$ holds for every (x, s, y) satisfying (2). We also consider the semi-infinite model where we only have continuous nonbasic variables.

$$\begin{aligned} x = f + \sum_{r\in\mathbb{R}^n} r s_r \\ x \in S, \\ s_r \in \mathbb{R}_+,\ \forall r \in \mathbb{R}^n, \\ s \text{ has finite support.} \end{aligned} \tag{3}$$

A function $\psi : \mathbb{R}^n \to \mathbb{R}$ is said to be *valid* for (3) if the inequality $\sum_{r\in\mathbb{R}^n} \psi(r)s_r \geq 1$ holds for every (x, s) satisfying (3). Given a valid function ψ for (3), a function π is a *lifting* of ψ if (ψ, π) is valid for (2). One is interested only in (pointwise) *minimal valid functions*, since non-minimal ones are implied by some minimal valid function. If ψ is a minimal valid function for (3) and π is a lifting of ψ such that (ψ, π) is a minimal valid function for (2) then we say that π is a *minimal lifting* of ψ.

While minimal valid functions for (3) have a simple characterization [6], minimal valid functions for (2) are not well understood. A general idea to derive minimal valid functions for (2) is to start from some minimal valid function ψ for (3), and construct a minimal lifting π of ψ. While there is no general technique to compute such minimal lifting π, it is known that there exists a region R_ψ, containing the origin in its interior, where ψ coincides with π for any minimal lifting π. This latter fact was observed by Dey and Wolsey [9] for the case of $S = \mathbb{Z}^2$ and by Conforti, Cornuéjols and Zambelli [8] for the general case. Furthermore, it is remarked in [8] and [10] that, if π is a minimal lifting of ψ, then $\pi(r) = \pi(r')$ for every $r, r' \in \mathbb{R}^n$ such that $r - r' \in \mathbb{Z}^n \cap \operatorname{lin}(\operatorname{conv}(S))$. Therefore the coefficients of any minimal lifting π are uniquely determined in the region $R_\psi + (\mathbb{Z}^n \cap \operatorname{lin}(\operatorname{conv}(S)))$. In particular, whenever translating R_ψ by integer vectors in $\operatorname{lin}(\operatorname{conv}(S))$ covers $\mathbb{R}^n$, ψ has a unique minimal lifting. The purpose of this paper is to give a precise description of the region R_ψ.

To state our main result, we need to explain the characterization of minimal valid functions for (3). We say that a convex set $B \subseteq \mathbb{R}^n$ is S-free if B does not contain any point of S in its interior. A set B is a *maximal S-free* convex set if it is an S-free convex set that is not properly contained in any S-free convex set. It was proved in [6] that maximal S-free convex sets are polyhedra containing a point of S in the relative interior of each facet.

Given an S-free polyhedron $B \subseteq \mathbb{R}^n$ containing f in its interior, B can be uniquely written in the form

$$B = \{x \in \mathbb{R}^n : \ a_i(x - f) \leq 1,\ i \in I\},$$

where I is a finite set of indices and $a_i(x - f) \leq 1$ is facet-defining for B for every $i \in I$.

Let $\psi_B : \mathbb{R}^n \to \mathbb{R}$ be the function defined by

$$\psi_B(r) = \max_{i\in I} a_i r, \quad \forall r \in \mathbb{R}^n.$$

Note in particular that, since maximal S-free convex sets are polyhedra, the above function is defined for all maximal S-free convex sets B.

Theorem 1. *[6] Let ψ be a minimal valid function for (3). Then the set*

$$B_\psi := \{x \in \mathbb{R}^n \,|\, \psi(x-f) \le 1\}$$

is a maximal S-free convex set containing f in its interior, and $\psi = \psi_{B_\psi}$.
Conversely, if B is a maximal S-free convex set containing f in its interior, then ψ_B is a minimal valid function for (3).

We are now ready to state the main result of the paper. Given a minimal valid function ψ for (3), by Theorem 1 B_ψ is a maximal S-free convex set containing f in its interior, thus it can be uniquely written as $B_\psi = \{x \in \mathbb{R}^n \,|\, a_i(x-f) \le 1,\ i \in I\}$. For every $r \in \mathbb{R}^n$, let $I(r) = \{i \in I \,|\, \psi(r) = a_i r\}$. Given $x \in S$, let

$$R(x) := \{r \in \mathbb{R}^n \,|\, I(r) \supseteq I(x-f) \text{ and } I(x-f-r) \supseteq I(x-f)\}.$$

We define

$$R_\psi := \bigcup_{x \in S \cap B_\psi} R(x).$$

Theorem 2. *Let ψ be a minimal valid function for (3). If π is a minimal lifting of ψ, then $\pi(r) = \psi(r)$ for every $r \in R_\psi$.*
Conversely, for every $\bar{r} \notin R_\psi$, there exists a lifting π of ψ such that $\pi(\bar{r}) < \psi(\bar{r})$.

Figure 1 illustrates the region R_ψ for several examples. We conclude the introduction presenting a different characterization of the regions $R(x)$.

Proposition 1. *Let ψ be a minimal valid function for (3), and let $x \in S$. Then $R(x) = \{r \in \mathbb{R}^n \,|\, \psi(r) + \psi(x-f-r) = \psi(x-f)\}$.*

Proof. We can uniquely write $B_\psi = \{x \in \mathbb{R}^n \,|\, a_i(x-f) \le 1,\ i \in I\}$. Let $h \in I(x-f)$. Then

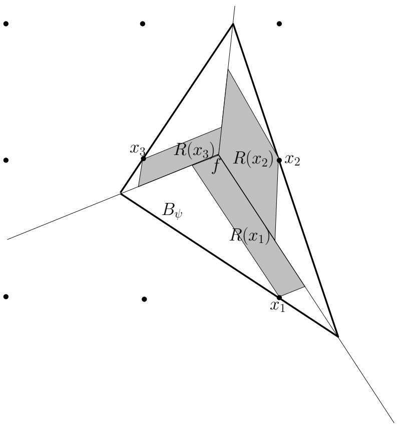
$\psi(x-f) = a_h(x-f) = a_h r + a_h(x-f-r) \le \max_{i\in I} a_i r + \max_{i \in I} a_i(x-f-r) = \psi(r) + \psi(x-f-r)$.

In the above expression, equality holds if and only if $h \in I(r)$ and $h \in I(x-f-r)$.

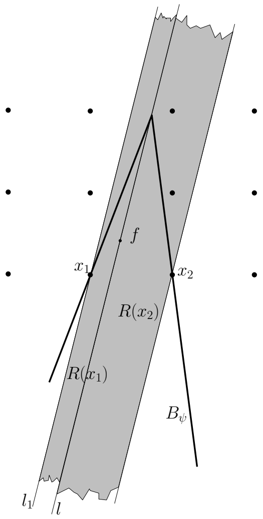
2 Minimum Lifting Coefficient of a Single Variable

Given $r^* \in \mathbb{R}^n$, we consider the set of solutions to

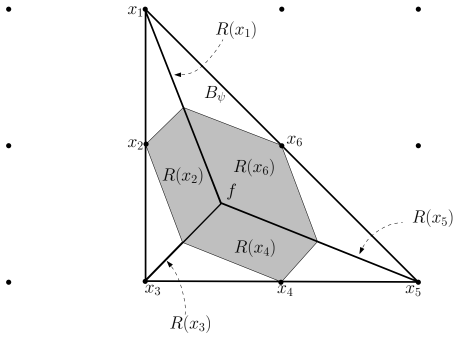
$$\begin{aligned} x = f + \sum_{r \in \mathbb{R}^n} r s_r + r^* y_{r^*} \\ x \in S \\ s \ge 0 \\ y_{r^*} \ge 0,\ y_{r^*} \in \mathbb{Z} \\ s \text{ has finite support.} \end{aligned} \tag{4}$$



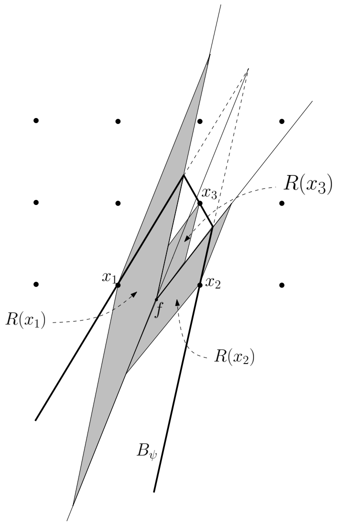
(a) A maximal $\mathbb{Z}^2$-free triangle with three integer points



(b) A wedge



(c) A maximal $\mathbb{Z}^2$-free triangle with integer vertices



(d) A truncated wedge

Fig. 1. Regions $R(x)$ for some maximal S-free convex sets in the plane. The thick dark line indicates the boundary of B_ψ. For a particular x, the dark gray regions denote $R(x)$. The jagged lines in a region indicate that it extends to infinity. For example, in Figure 1(b), $R(x_1)$ is the strip between lines l_1 and l. Figure 1(c) shows an example where $R(x)$ is full-dimensional for x_2, x_4, x_6, but is not full-dimensional for x_1, x_3, x_5.

Given a minimal valid function ψ for (3) and scalar λ, we say that the inequality $\sum_{r\in\mathbb{R}^n} \psi(r)s_r + \lambda y_{r^*} \geq 1$ is valid for (4) if it holds for every (x, s, y_{r^*}) satisfying (4). We denote by $\psi^*(r^*)$ the minimum value of λ for which $\sum_{r\in\mathbb{R}^n} \psi(r)s_r + \lambda y_{r^*} \geq 1$ is valid for (4).

We observe that, for *any* lifting π of ψ, we have

$$\psi^*(r^*) \leq \pi(r^*).$$

Indeed, $\sum_{r\in\mathbb{R}^n} \psi(r)s_r + \pi(r^*)y_{r^*} \geq 1$ is valid for (4), since, for any $(\bar{s}, \bar{y}_{r^*})$ satisfying (4), the vector $(\bar{s}, \bar{y})$, defined by $\bar{y}_r = 0$ for all $r \in \mathbb{R}^n \backslash \{r^*\}$, satisfies (2).

Moreover, the following fact was shown in [8].

Lemma 1. *If ψ is a minimal valid function for* (3) *and π is a minimal lifting of ψ, then $\pi \leq \psi$.*

So we have the following relation for every minimal lifting π of ψ :

$$\psi^*(r) \leq \pi(r) \leq \psi(r) \quad \text{for all } r \in R^n.$$

In general ψ^* is not a lifting of ψ, but if it is, then the above relation implies that it is the unique minimal lifting of ψ.

Remark 1. For any $r \in \mathbb{R}^n$ such that $\psi^*(r) = \psi(r)$, we have $\pi(r) = \psi(r)$ for every minimal lifting π of ψ. Conversely, if $\psi^*(r^*) < \psi(r^*)$ for some $r^* \in \mathbb{R}^n$, then there exists some lifting π of ψ such that $\pi(r^*) < \psi(r^*)$.

Proof. The first part follows from $\psi^* \leq \pi \leq \psi$. For the second part, given $r^* \in \mathbb{R}^n$ such that $\psi^*(r^*) < \psi(r^*)$, we can define π by $\pi(r^*) = \psi^*(r^*)$ and $\pi(r) = \psi(r)$ for all $r \in \mathbb{R}^n$, $r \neq r^*$. Since ψ is valid for (3), it follows by the definition of $\psi^*(r^*)$ that π is a lifting of ψ.

By the above remark, in order to prove Theorem 2 we need to show that $R_\psi = \{r \in \mathbb{R}^n \,|\, \psi(r) = \psi^*(r)\}$. We will need the following results.

Theorem 3. *[6] A full-dimensional convex set B is a maximal S-free convex set if and only if it is a polyhedron such that B does not contain any point of S in its interior and each facet of B contains a point of S in its relative interior. Furthermore if $B \cap \text{conv}(S)$ has nonempty interior, $\text{lin}(B)$ contains $\text{rec}(B \cap \text{conv}(S))$.*

Remark 2. The proof of Theorem 3 in [6] implies the following. Given a maximal S-free convex set B, there exists $\delta > 0$ such that there is no point of $S \setminus B$ at distance less than δ from B.

Let $r^* \in \mathbb{R}^n$. Given a maximal S-free convex set $B = \{x \in \mathbb{R}^n \,|\, a_i(x - f) \leq 1,\ i \in I\}$, for any $\lambda \in \mathbb{R}$, we define the set $B(\lambda) \subset \mathbb{R}^{n+1}$ as follows

$$B(\lambda) = \{\begin{pmatrix} x \\ x_{n+1} \end{pmatrix} \in \mathbb{R}^{n+1} \,|\, a_i(x - f) + (\lambda - a_i r^*)x_{n+1} \leq 1,\ i \in I\}. \tag{5}$$

Theorem 4. [8] *Let $r^* \in \mathbb{R}^n$. Given a maximal S-free convex set B, let $\psi = \psi_B$. Given $\lambda \in \mathbb{R}$, the inequality $\sum_{r\in\mathbb{R}^n} \psi(r)s_r + \lambda y_{r^*} \geq 1$ is valid for (4) if and only if $B(\lambda)$ is $(S \times \mathbb{Z}_+)$-free.*

Remark 3. Let $r^* \in \mathbb{R}^n$, and let B be a maximal S-free convex set. For every λ such that $B(\lambda)$ is $(S \times \mathbb{Z}_+)$-free, $B(\lambda)$ is maximal $(S \times \mathbb{Z}_+)$-free.

Proof. Since B is a maximal S-free convex set, then by Theorem 3 each facet of B contains a point $\bar{x}$ of S in its relative interior. Therefore the corresponding facet of $B(\lambda)$ contains the point $\binom{\bar{x}}{0}$ in its relative interior. If $B(\lambda)$ is $(S \times \mathbb{Z}_+)$-free, by Theorem 3 it is a maximal $(S \times \mathbb{Z}_+)$-free convex set.

3 Characterizing the Region R_ψ

Next we state and prove a theorem that characterizes when $\psi^*(r^*) = \psi(r^*)$. The main result of this paper, Theorem 2, will then follow easily.

Theorem 5. *Given a maximal S-free convex set B, let $\psi = \psi_B$. Given $r^* \in \mathbb{R}^n$, the following are equivalent:*
(i) $\psi^*(r^*) = \psi(r^*)$.
(ii) There exists a point $\bar{x} \in S$ such that $\binom{\bar{x}}{1} \in B(\psi(r^*))$.

Proof. Let $\lambda^* = \psi(r^*)$. Note that the inequality $\sum_{r\in\mathbb{R}^n} \psi(r)s_r + \lambda^* y_{r^*} \geq 1$ is valid for (4). Thus, it follows from Theorem 4 that $B(\lambda^*)$ is $S \times \mathbb{Z}_+$-free .

We first show that (ii) implies (i). Assume there exists $\bar{x} \in S$ such that $\binom{\bar{x}}{1} \in B(\lambda^*)$. Then, for every $\epsilon > 0$, $\binom{\bar{x}}{1}$ is in the interior of $B(\lambda^* - \epsilon)$, because $a_i(\bar{x} - f) + (\lambda - \varepsilon - a_i r^*) \leq 1 - \varepsilon < 1$ for all $i \in I$. Theorem 4 then implies that $\psi^*(r^*) = \lambda^*$.

Next we show that (i) implies (ii). Assume that $\psi^*(r^*) = \psi(r^*) = \lambda^*$. We recall that $\lambda^* = \max_{i\in I} a_i r^*$.

Note that, if $a_i r^* = \lambda^*$ for all $i \in I$, then $B(\lambda^*) = B \times \mathbb{R}$, so given any point $\bar{x}$ in $B \cap S$, $\binom{\bar{x}}{1}$ is in $B(\lambda^*)$. Thus we assume that there exists an index h such that $a_h r^* < \lambda^*$.

By Remark 3, $B(\lambda^*)$ is maximal $(S \times \mathbb{Z}_+)$-free. Theorem 3 implies the following,
a) $\text{rec}(B \cap \text{conv}(S)) \subseteq \text{lin}(B)$,
b) $\text{rec}(B(\lambda^*) \cap \text{conv}(S \times \mathbb{Z}_+)) \subseteq \text{lin}(B(\lambda^*))$.

Lemma 2. $\text{rec}(B(\lambda^*) \cap \text{conv}(S \times \mathbb{Z}_+)) = \text{rec}(B \cap \text{conv}(S)) \times \{0\}$.

Let $\binom{\bar{r}}{\bar{r}_{n+1}} \in \text{rec}(B(\lambda^*) \cap \text{conv}(S \times \mathbb{Z}_+))$. Note that $\text{rec}(\text{conv}(S \times \mathbb{Z}_+)) = \text{rec}(\text{conv}(S)) \times \mathbb{Z}_+$, thus $\bar{r} \in \text{rec}(\text{conv}(S))$ and $\bar{r}_{n+1} \geq 0$. We only need to show that $\bar{r}_{n+1} = 0$.

By b), $\binom{\bar{r}}{\bar{r}_{n+1}}$ satisfies

$$a_i\bar{r} + (\lambda^* - a_i r^*)\bar{r}_{n+1} = 0, \quad i \in I. \tag{6}$$

Since $\lambda^* - a_i r^* \geq 0$ and $\bar{r}_{n+1} \geq 0$,

$$a_i\bar{r} \leq 0, \quad i \in I,$$

therefore $\bar{r} \in \text{rec}(B)$. Thus $\bar{r} \in \text{rec}(B \cap \text{conv}(S))$ which, by a), is contained in $\text{lin}(B)$. This implies

$$a_i\bar{r} = 0, \quad i \in I.$$

It follows from the above and from (6) that $(\lambda^* - a_ir^*)\bar{r}_{n+1} = 0$ for $i \in I$. Since $\lambda^* - a_hr^* > 0$ for some index h, it follows that $\bar{r}_{n+1} = 0$. This concludes the proof of Lemma 2.

Lemma 3. *There exists* $\bar{\varepsilon} > 0$ *such that* $\text{rec}(B(\lambda^* - \varepsilon) \cap \text{conv}(S \times \mathbb{Z}_+)) = \text{rec}(B \cap \text{conv}(S)) \times \{0\}$ *for every* $\varepsilon \in [0, \bar{\varepsilon}]$.

Since $\text{conv}(S)$ is a rational polyhedron, $S = \{x \in \mathbb{R}^n \,|\, Cx \leq d\}$ for some rational matrix (C, d). By Lemma 2, there is no vector $\binom{r}{1}$ in $\text{rec}(B(\lambda^*) \cap \text{conv}(S \times \mathbb{Z}_+))$. Thus the system

$$\begin{gathered} a_ir + (\lambda^* - a_ir^*) \leq 0, i \in I \\ Cr \leq 0 \end{gathered}$$

is infeasible. By Farkas Lemma, there exist scalars $\mu_i \geq 0$, $i \in I$ and a nonnegative vector γ such that

$$\begin{gathered} \sum_{i \in I} \mu_i a_i + \gamma C = 0 \\ \lambda^*(\sum_{i \in I} \mu_i) - (\sum_{i \in I} \mu_i a_i)r^* > 0. \end{gathered}$$

This implies that there exists some $\bar{\varepsilon} > 0$ such that for all $\varepsilon \leq \bar{\varepsilon}$,

$$\begin{gathered} \sum_{i \in I} \mu_i a_i + \gamma C = 0 \\ (\lambda^* - \varepsilon)(\sum_{i \in I} \mu_i) - (\sum_{i \in I} \mu_i a_i)r^* > 0, \end{gathered}$$

thus the system

$$\begin{gathered} a_ir + (\lambda^* - \varepsilon - a_ir^*) \leq 0, i \in I \\ Cr \leq 0 \end{gathered}$$

is infeasible. This implies that $\text{rec}(B(\lambda^* - \varepsilon) \cap \text{conv}(S \times \mathbb{Z}_+)) = \text{rec}(B \cap \text{conv}(S)) \times \{0\}$.

Lemma 4. $B(\lambda^*)$ *contains a point* $\binom{\bar{x}}{\bar{x}_{n+1}} \in S \times \mathbb{Z}_+$ *such that* $\bar{x}_{n+1} > 0$.

By Lemma 3, there exists $\bar{\varepsilon}$ such that, for every $\varepsilon \in [0, \bar{\varepsilon}]$, $\text{rec}(B(\lambda^* - \varepsilon) \cap \text{conv}(S \times \mathbb{Z}_+)) = \text{rec}(B \cap \text{conv}(S)) \times \{0\}$. This implies that there exists a scalar M such that, for every $\varepsilon \in [0, \bar{\varepsilon}]$ and every point $\binom{\bar{x}}{\bar{x}_{n+1}} \in B(\lambda^* - \varepsilon) \cap (S \times \mathbb{Z}_+)$, it follows $\bar{x}_{n+1} \leq M$.

Remark 2 and Remark 3 imply that there exists $\delta > 0$ such that, for every $\binom{\bar{x}}{\bar{x}_{n+1}} \in (S \times \mathbb{Z}_+) \setminus B(\lambda^*)$, there exists $h \in I$ such that $a_h(\bar{x} - f) + (\lambda^* - a_hr^*)\bar{x}_{n+1} \geq 1 + \delta$. Choose $\varepsilon > 0$ such that $\varepsilon \leq \bar{\varepsilon}$ and $\varepsilon M \leq \delta$.

Since $\psi^*(r^*) = \lambda^*$, by Theorem 4, $B(\lambda^* - \varepsilon)$ has a point $\binom{\bar{x}}{\bar{x}_{n+1}} \in S \times \mathbb{Z}_+$ in its interior. Thus $a_i(\bar{x} - f) + (\lambda^* - \varepsilon - a_ir^*)\bar{x}_{n+1} < 1$, $i \in I$.

We show that $\binom{\bar{x}}{\bar{x}_{n+1}}$ is also in $B(\lambda^*)$. Suppose not. Then, by our choice of δ, there exists $h \in I$ such that $a_h(\bar{x}-f) + (\lambda^* - a_h r^*)\bar{x}_{n+1} \geq 1+\delta$. By our choice of M and ε,

$1+\delta \leq a_h(\bar{x}-f)+(\lambda^*-a_h r^*)\bar{x}_{n+1} \leq a_h(\bar{x}-f)+(\lambda^*-\varepsilon-a_h r^*)\bar{x}_{n+1}+\varepsilon M < 1+\varepsilon M \leq 1+\delta,$

a contradiction.

Hence $\binom{\bar{x}}{\bar{x}_{n+1}}$ is in $B(\lambda^*)$. Since B is S-free and $B(\lambda^*-\varepsilon)\cap(\mathbb{R}^n\times\{0\}) = B\times\{0\}$, it follows that $B(\lambda^*-\varepsilon)$ does not contain any point of $S\times\{0\}$ in its interior. Thus $\bar{x}_{n+1} > 0$. This concludes the proof of Lemma 4.

By the previous lemma, $B(\lambda^*)$ contains a point $\binom{\bar{x}}{\bar{x}_{n+1}} \in S \times \mathbb{Z}_+$ such that $\bar{x}_{n+1} > 0$. Note that $B(\lambda^*)$ contains $\binom{\bar{x}}{1}$, since

$$a_i(\bar{x}-f)+(\lambda^*-a_i r^*) \leq a_i(\bar{x}-f)+(\lambda^*-a_i r^*)\bar{x}_{n+1} \leq 1, \quad i \in I$$

since $\lambda^* - a_i r^* \geq 0$, $i \in I$.

Corollary 1. *Let ψ be a minimal valid function for (3). Then $\psi^*(r^*) = \psi(r^*)$ if and only if there exists $\bar{x} \in S$ such that*

$$\psi(r^*) + \psi(\bar{x}-f-r^*) = 1. \tag{7}$$

Proof. We first show that, if there exist $\bar{x} \in S$ satisfying (7), then $\psi^*(r^*) = \psi(r^*)$. Indeed, since $\sum_{r\in\mathbb{R}^n} \psi(r)s_r + \psi^*(r^*)y_{r^*} \geq 1$ is valid for (4),

$$1 \leq \psi^*(r^*) + \psi(\bar{x}-f-r^*) \leq \psi(r^*) + \psi(\bar{x}-f-r^*) = 1.$$

We show the converse. Since ψ is a valid function for (3), $\psi(\bar{x}-f-r^*)+\psi(r^*) \geq 1$. Since ψ is a minimal valid function for (3), by Theorem 1 there exists a maximal S-free convex set B such that $\psi = \psi_B$. Let $B_\psi = \{x \in \mathbb{R}^n \,|\, a_i(x-f) \leq 1, i \in I\}$.

Assume $\psi^*(r^*) = \psi(r^*)$. By Theorem 5, there exists a point $\bar{x} \in S$ such that $\binom{\bar{x}}{1} \in B(\psi(r^*))$. Therefore

$$a_i(\bar{x}-f) + \psi(r^*) - a_i r^* \leq 1, \quad i \in I.$$

Thus

$$\max_{i\in I} a_i(\bar{x}-f-r^*) \leq 1 - \psi(r^*),$$

which implies $\psi(\bar{x}-f-r^*)+\psi(r^*) \leq 1$. Hence $\psi(\bar{x}-f-r^*)+\psi(r^*) = 1$.

Proof (Proof of Theorem 2). By Remark 1, we only need to show that $R_\psi = \{r \in \mathbb{R}^n \,|\, \psi(r) = \psi^*(r)\}$. For every $x \in S$, we have $\psi(x-f) = 1$ if and only if $x \in S \cap B_\psi$. Therefore, by Proposition 1, $R(x) = \{r \in \mathbb{R}^n \,|\, \psi(r) + \psi(x-f-r) = \psi(x-f) = 1\}$ if and only if $x \in S \cap B_\psi$. The latter fact and Corollary 1 imply that a vector $r \in \mathbb{R}^n$ satisfies $\psi^*(r) = \psi(r)$ if and only if $r \in R(x)$ for some $x \in S \cap B_\psi$. The statement now follows from the definition of R_ψ.

4 Conclusion

In this paper we give an exact characterization of the region where a minimal valid inequality ψ and any minimal lifting π of ψ coincide. This was exhibited in Theorem 2, which generalizes results from [8] and [9] about liftings of minimal valid inequalities.

As already mentioned in the introduction, the following theorem was proved in [8].

Theorem 6. *Let ψ be a minimal valid function for (3). If $R_\psi + (\mathbb{Z}^n \cap \text{lin}(\text{conv}(S)))$ covers all of $\mathbb{R}^n$, then there exists a unique minimal lifting π of ψ.*

We conjecture that the converse also holds.

Conjecture 7 *Let ψ be a minimal valid function for (3). There exists a unique minimal lifting π of ψ if and only if $R_\psi + (\mathbb{Z}^n \cap \text{lin}(\text{conv}(S)))$ covers all of $\mathbb{R}^n$.*

Acknowledgements

The authors would like to thank Marco Molinaro for helpful discussions about the results presented in this paper.

References

1. Andersen, K., Louveaux, Q., Weismantel, R., Wolsey, L.A.: Cutting Planes 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)
2. Balas, E.: Intersection Cuts - A New Type of Cutting Planes for Integer Programming. Operations Research 19, 19–39 (1971)
3. Barvinok, A.: A Course in Convexity. In: Graduate Studies in Mathematics, vol. 54. American Mathematical Society, Providence (2002)
4. Basu, A., Conforti, M., Cornuejols, G., Zambelli, G.: A Counterexample to a Conjecture of Gomory and Johnson. Mathematical Programming Ser. A (to appear 2010)
5. Basu, A., Conforti, M., Cornuejols, G., Zambelli, G.: Maximal Lattice-free Convex Sets in Linear Subspaces (2009) (manuscript)
6. Basu, A., Conforti, M., Cornuejols, G., Zambelli, G.: Minimal Inequalities for an Infinite Relaxation of Integer Programs. SIAM Journal of Discrete Mathematics (to appear 2010)
7. Borozan, V., Cornuéjols, G.: Minimal Valid Inequalities for Integer Constraints. Mathematics of Operations Research 34, 538–546 (2009)
8. Conforti, M., Cornuejols, G., Zambelli, G.: A Geometric Perspective on Lifting (May 2009) (manuscript)
9. Dey, S.S., Wolsey, L.A.: Lifting Integer Variables in Minimal Inequalities corresponding to Lattice-Free Triangles. In: Lodi, A., Panconesi, A., Rinaldi, G. (eds.) IPCO 2008. LNCS, vol. 5035, pp. 463–475. Springer, Heidelberg (2008)
10. Dey, S.S., Wolsey, L.A.: Constrained Infinite Group Relaxations of MIPs (March 2009) (manuscript)

11. Gomory, R.E.: Some Polyhedra related to Combinatorial Problems. Linear Algebra and its Applications 2, 451–558 (1969)
12. Gomory, R.E., Johnson, E.L.: Some Continuous Functions Related to Corner Polyhedra, Part I. Mathematical Programming 3, 23–85 (1972)
13. Johnson, E.L.: On the Group Problem for Mixed Integer Programming. In: Mathematical Programming Study, pp. 137–179 (1974)
14. Schrijver, A.: Theory of Linear and Integer Programming. John Wiley & Sons, New York (1986)
15. Meyer, R.R.: On the Existence of Optimal Solutions to Integer and Mixed-Integer Programming Problems. Mathematical Programming 7, 223–235 (1974)

Efficient Edge Splitting-Off Algorithms Maintaining All-Pairs Edge-Connectivities

Lap Chi Lau and Chun Kong Yung

Department of Computer Science and Engineering
The Chinese University of Hong Kong

Abstract. In this paper we present new edge splitting-off results maintaining all-pairs edge-connectivities of a graph. We first give an alternate proof of Mader's theorem, and use it to obtain a deterministic $\tilde{O}({r_{\max}}^2 \cdot n^2)$-time complete edge splitting-off algorithm for unweighted graphs, where $r_{\max}$ denotes the maximum edge-connectivity requirement. This improves upon the best known algorithm by Gabow by a factor of $\tilde{\Omega}(n)$. We then prove a new structural property, and use it to further speedup the algorithm to obtain a randomized $\tilde{O}(m + {r_{\max}}^3 \cdot n)$-time algorithm. These edge splitting-off algorithms can be used directly to speedup various graph algorithms.

1 Introduction

The edge splitting-off operation plays an important role in many basic graph problems, both in proving theorems and obtaining efficient algorithms. Splitting-off a pair of edges (xu, xv) means deleting these two edges and adding a new edge uv if $u \neq v$. This operation is introduced by Lovász [18] who showed that splitting-off can be performed to maintain the *global edge-connectivity* of a graph. Mader extended Lovász's result significantly to prove that splitting-off can be performed to maintain the *local edge-connectivity* for all pairs:

Theorem 1 (Mader [19]). *Let $G = (V, E)$ be an undirected graph that has at least $r(s,t)$ edge-disjoint paths between s and t for all $s, t \in V - x$. If there is no cut edge incident to x and $d(x) \neq 3$, then some edge pair (xu, xv) can be split-off so that in the resulting graph there are still at least $r(s,t)$ edge-disjoint paths between s and t for all $s, t \in V - x$.*

These splitting-off theorems have applications in various graph problems. Lovász [18] and Mader [19] used their splitting-off theorems to derive Nash-Williams' graph orientation theorems [23]. Subsequently these theorems and their extensions have found applications in a number of problems, including edge-connectivity augmentation problems [4, 8, 9], network design problems [7, 13, 16], tree packing problems [1, 6, 17], and graph orientation problems [11].

Efficient splitting-off algorithms have been developed to give fast algorithms for the above problems [4, 6, 12, 20, 22]. However, most of the efficient algorithms are developed only in the global edge-connectivity setting, although there are important applications in the more general local edge-connectivity setting.

In this paper we present new edge splitting-off results maintaining all-pairs edge-connectivities. First we give an alternate proof of Mader's theorem (Theorem 1). Based on this, we develop a faster deterministic algorithm for edge splitting-off maintaining all-pairs edge-connectivities (Theorem 2). Then we prove a new structural property (Theorem 3), and use it to design a randomized procedure to further speedup the splitting-off algorithm (Theorem 2). These algorithms improve the best known algorithm by a factor of $\tilde{\Omega}(n)$, and can be applied directly to speedup various graph algorithms using edge splitting-off.

1.1 Efficient Complete Edge Splitting-Off Algorithm

Mader's theorem can be applied repeatedly until $d(x) = 0$ when $d(x)$ is even and there is no cut edge incident to x. This is called a *complete edge splitting-off* at x, which is a key subroutine in algorithms for connectivity augmentation, graph orientation, and tree packing.

A straightforward algorithm to compute a complete splitting-off sequence is to split-off (xu, xv) for every pair $u, v \in N(x)$ where $N(x)$ is the neighbor set of x, and then check whether the connectivity requirements are violated by computing all-pairs edge-connectivities in the resulting graph, and repeat this procedure until $d(x) = 0$.

Several efficient algorithms are proposed for the complete splitting-off problem, but only Gabow's algorithm [12] can be used in the local edge-connectivity setting, with running time $O({r_{\max}}^2 \cdot n^3)$. Our algorithms improve the running time of Gabow's algorithm by a factor of $\tilde{\Omega}(n)$. In applications where $r_{\max}$ is small, the improvement of the randomized algorithm could be a factor of $\tilde{\Omega}(n^2)$.

Theorem 2. *In the local edge-connectivity setting, there is a deterministic $\tilde{O}({r_{\max}}^2 \cdot n^2)$-time algorithm and a randomized $\tilde{O}(m + {r_{\max}}^3 \cdot n)$-time algorithm for the complete edge splitting-off problem in unweighted graphs.*

These edge splitting-off algorithms can be used directly to improve the running time of various graph algorithms [7, 9, 12, 13, 17, 23]. For instance, using Theorem 2 in Gabow's local edge-connectivity augmentation algorithm [12] in unweighted graphs, the running time can be improved from $\tilde{O}({r_{\max}}^2 n^3)$ to $\tilde{O}({r_{\max}}^2 n^2)$ time. Similarly, using Theorem 2 in Gabow's orientation algorithm [12], one can find a well-balanced orientation in unweighted graphs in $\tilde{O}({r_{\max}}^3 n^2)$ expected time, improving the $O({r_{\max}}^2 n^3)$ result by Gabow [12]. We will not discuss the details of these applications in this paper.

Our edge splitting-off algorithms are conceptually very simple, which can be seen as refinements of the straightforward algorithm. The improvements come from some new structural results, and a recent fast Gomory-Hu tree construction algorithm by Bhalgat, Hariharan, Kavitha, and Panigrahi [5]. First, in Section 3.2, we show how to find a complete edge splitting-off sequence by using at most $O(|N(x)|)$ splitting-off attempts, instead of $O(|N(x)|^2)$ attempts by the straightforward algorithm. This is based on an alternative proof of Mader's theorem in Section 3.1. Then, in Section 3.4, we show how to reduce the problem of checking local edge-connectivities for all pairs, to the problem of checking local edge-connectivities from a particular vertex (i.e. checking at most $O(n)$ pairs instead of checking $O(n^2)$ pairs). This allows us to use the recent fast Gomory-Hu tree algorithm [5] to check connectivities efficiently. Finally, using a new structural property (Theorem 3), we show how to speedup the algorithm by a randomized edge splitting-off procedure in Section 4.

1.2 Structural Property and Randomized Algorithm

Mader's theorem shows the existence of one *admissible* edge pair, whose splitting-off maintains the local edge-connectivity requirements of the graph. Given an edge xv, we say an edge xw is a *non-admissible partner* of xv if (xv, xw) is not admissible. We prove a tight upper bound on the number of non-admissible partners of a given edge xv, which may be of independent interest. In the following $r_{\max} := \max_{s,t \in V-x} r(s,t)$ is the maximum edge-connectivity requirement.

Theorem 3. *Suppose there is no cut edge incident to x and $r_{\max} \geq 2$. Then the number of non-admissible partners for any given edge xv is at most $2r_{\max} - 2$.*

This improves the result of Bang-Jensen and Jordán [2] by a factor of $r_{\max}$, and the bound is best possible as there are examples achieving it. Theorem 3 implies that when $d(x)$ is considerably larger than $r_{\max}$, most of the edge pairs incident to x are admissible. Therefore, we can split-off edge pairs *randomly* to speedup our efficient splitting-off algorithm. The proof of Theorem 3 is based on a new inductive argument and will be presented in Section 4.

2 Preliminaries

Let $G = (V, E)$ be a graph. For $X, Y \subseteq V$, denote by $\delta(X, Y)$ the set of edges with one endpoint in $X - Y$ and the other endpoint in $Y - X$ and $d(X, Y) := |\delta(X, Y)|$, and also define $\bar{d}(X, Y) := d(X \cap Y, V - (X \cup Y))$. For $X \subseteq V$, define $\delta(X) := \delta(X, V - X)$ and the *degree* of X as $d(X) := |\delta(X)|$. Denote the degree of a vertex as $d(v) := d(\{v\})$. Also denote the set of neighbors of v by $N(v)$, and call a vertex in $N(v)$ a *v-neighbor.*

Let $\lambda(s, t)$ be the maximum number of edge-disjoint paths between s and t in V, and let $r(s, t)$ be an *edge-connectivity requirement* for $s, t \in V$. The connectivity requirement is *global* if $r_{s,t} = k$ for all $s, t \in V$, otherwise it is *local.* We say a graph G satisfies the connectivity requirements if $\lambda(s, t) \geq r(s, t)$ for any $s, t \in V$. The requirement $r(X)$ of a set $X \subseteq V$ is the maximum edge-connectivity requirement between u and v with $u \in X$ and $v \in V - X$. By Menger's theorem, to satisfy the requirements, it suffices to guarantee that $d(X) \geq r(X)$ for all $X \subset V$. The *surplus* $s(X)$ of a set $X \subseteq V$ is defined as $d(X) - r(X)$. A graph satisfies the edge-connectivity requirements if $s(X) \geq 0$ for all $\emptyset \neq X \subset V$. For $X \subset V - x$, X is called *dangerous* if $s(X) \leq 1$ and *tight* if $s(X) = 0$. The following proposition will be used throughout our proofs.

Proposition 4 ([10] Proposition 2.3). *For $X, Y \subseteq V$ at least one of the following inequalities holds:*

$$s(X) + s(Y) \geq s(X \cap Y) + s(X \cup Y) + 2d(X, Y) \tag{4a}$$
$$s(X) + s(Y) \geq s(X - Y) + s(Y - X) + 2\bar{d}(X, Y) \tag{4b}$$

In edge splitting-off problems, the objective is to split-off a pair of edges incident to a designated vertex x to maintain the edge-connectivity requirements for all other pairs in $V - x$. For this purpose, we may assume that the edge-connectivity requirements between x and other vertices are zero. In particular, we may assume that $r(V - x) = 0$ and thus the set $V - x$ is not a dangerous set. Two edges xu, xv form an *admissible pair* if the graph after splitting-off (xu, xv) does not violate $s(X) \geq 0$ for all $X \subset V$. Given an edge xv, we say an edge xw is a *non-admissible partner* of xv if (xv, xw) is not admissible. The following simple proposition characterizes when a pair is admissible.

Proposition 5 ([10] Claim 3.1). *A pair xu, xv is not admissible if and only if u, v are contained in a dangerous set.*

A vertex subset $S \subseteq N(x)$ is called a *non-admissible set* if (xu, xv) is non-admissible for every $u, v \in S$. We define the *capacity* of an edge pair to be the number of copies of the edge pair that can be split-off while satisfying edge-connectivity requirements. In our algorithms we will always split-off an edge pair to its capacity (which could be zero), and only attempt at most $O(|N(x)|)$ many pairs. Following the definition of Gabow [12], we say that a splitting-off operation *voids* a vertex u if $d(x, u) = 0$ after the splitting-off.

Throughout the complete splitting-off algorithm, we assume that there is no cut edge incident to x. This holds at the beginning by our assumption, and so the local edge-connectivity between x and v is at least two for each x-neighbor v. Therefore, we can reset the connectivity requirement between u and v as $\max\{r(u, v), 2\}$, and hence splitting-off any admissible pair would maintain the property that there is no cut edge incident to x at each step.

2.1 Some Useful Results

The first lemma is about a reduction step of contracting tight sets. Suppose there is a *non-trivial* tight set T, i.e. T is a tight set and $|T| \geq 2$. Clearly there are no admissible pairs xu, xv with $u, v \in T$. Let G/T be the graph obtained by contracting T into a single vertex t, and define the connectivity requirement $r(t, v)$ as $\max_{u \in T} r(u, v)$, while other connectivity requirements remain the same. The following lemma says that one can consider the admissible pairs in G/T, without losing any information about the admissible pairs in G. This lemma is useful in proofs to assume that every tight set is a singleton, and is useful in algorithms to allow us to make progress by contracting non-trivial tight sets.

Lemma 6 ([19], [10] Claim 3.2). *Let T be a non-trivial tight set. For an x-neighbor w in G/T, let w' be the corresponding vertex in G if $w \neq t$, and let w' be any x-neighbor in T in G if $w = t$. Suppose (xu, xv) is an admissible pair in G/T, then (xu', xv') is an admissible pair in G.*

The next lemma proved in [7] shows that if the conditions in Mader's theorem are satisfied, then there is no "3-dangerous-set structure". This lemma is important in the efficient edge splitting-off algorithm.

Lemma 7 ([7] Lemma 2.7). *If $d(x) \neq 3$ and there is no cut edge incident to x, then there are no maximal dangerous sets X, Y, Z and $u, v, w \in N(x)$ with $u \in X \cap Y$, $v \in X \cap Z$, $w \in Y \cap Z$ and $u, v, w \notin X \cap Y \cap Z$.*

Nagamochi and Ibaraki [21] gave a fast algorithm to find a sparse subgraph that satisfies edge-connectivity requirements, which will be used in Section 3.3 as a preprocessing step.

Theorem 8 ([21] Lemma 2.1). *There is an $O(m)$-time algorithm to construct a subgraph with $O(r_{\max} \cdot n)$ edges that satisfies all the connectivity requirements.*

As a key tool in checking local edge-connectivities, we need to construct a Gomory-Hu tree, which is a compact representation of all pairwise min-cuts of an undirected graph. Let $G = (V, E)$ be an undirected graph, a *Gomory-Hu tree* is a weighted tree $T = (V, F)$ with the following property. Consider any $s, t \in V$, the unique s-t path P in T, an edge $e = uv$ on P with minimum weight, and any component K of $T - e$. Then the local edge-connectivity between s and t in G is equal to the weight of e in T, and $\delta(K)$ is a minimum s-t cut in G. To check whether the connectivity requirements are satisfied, we only need to check the pairs with $\lambda(u, v) \leq r_{\max}$. A *partial Gomory-Hu tree* T_k of G is obtained from a Gomory-Hu tree T of G by contracting all edges with weight at least k. Therefore, each node in T_k represents a subset of vertices S in G, where the local edge-connectivity between each pair of vertices in S is at least k. For vertices $u, v \in G$ in different nodes of T_k, their local edge-connectivity (which is less than k) is determined in the same way as in an ordinary Gomory-Hu tree. Bhalgat et.al. [5] gave a fast randomized algorithm to construct a partial Gomory-Hu tree. We will use the following theorem by setting $k = r_{\max}$. The following result can be obtained by using the algorithm in [15], with the fast tree packing algorithm in [5].

Theorem 9 ([5, 15]). *A partial Gomory-Hu tree T_k can be constructed in $\tilde{O}(km)$ expected time.*

3 Efficient Complete Edge Splitting-Off Algorithm

In this section we present the deterministic splitting-off algorithm as stated in Theorem 2. First we present an alternative proof of Mader's theorem in Section 3.1. Extending the ideas in the alternative proof we show how to find a complete edge splitting-off sequence by only $O(|N(x)|)$ edge splitting-off attempts in Section 3.2. Then, in Section 3.3, we show how to efficiently perform one edge splitting-off attempt, by doing some preprocessing and applying some fast algorithms to check edge-connectivities. Combining these two steps yields an $\tilde{O}({r_{\max}}^2 \cdot n^2)$ randomized algorithm for the complete splitting-off problem. Finally, in Section 3.5, we describe how to modify some steps in Section 3.3 to obtain an $\tilde{O}({r_{\max}}^2 \cdot n^2)$ deterministic algorithm for the problem.

3.1 Mader's Theorem

We present an alternative proof of Mader's theorem, which can be extended to obtain an efficient algorithm. The following lemma about non-admissible sets can be used directly to derive Mader's theorem.

Lemma 10. *Suppose there is no 3-dangerous set structure. Then, for any non-admissible set $U \subseteq N(x)$ with $|U| \geq 2$, there is a dangerous set containing U.*

Proof. We prove the lemma by a simple induction. The statement holds trivially for $|U| = 2$ by Proposition 5. Consider $U = \{u_1, u_2, \ldots, u_{k+1}\} \subseteq N(x)$ where every pair (u_i, u_j) is non-admissible. By induction, since every pair (u_i, u_j) is non-admissible, there are maximal dangerous sets X, Y such that $\{u_1, ..., u_{k-1}, u_k\} \subseteq X$ and $\{u_1, ..., u_{k-1}, u_{k+1}\} \subseteq Y$. Since (u_k, u_{k+1}) is non-admissible, by Proposition 5, there is a dangerous set Z containing u_k and u_{k+1}. If $u_{k+1} \notin X$ and $u_k \notin Y$ and there is some $u_i \notin Z$, then X, Y and Z form a 3-dangerous-set structure with $u = u_i, v = u_k, w = u_{k+1}$. Hence either X, Y or Z contains U. □

To prove Mader's theorem, consider a vertex $x \in V$ with $d(x)$ is even and there is no cut edge incident to it. By Lemma 7, there is no 3-dangerous set structure in G. Suppose that there is no admissible pair incident to x. Then, by Lemma 10, there is a dangerous set D containing all the vertices in $N(x)$. But this is impossible since $r(V-D-x) = r(D) \geq d(D)-1 = d(V-D-x)+d(x)-1 \geq d(V-D-x)+1$, contradicting that the connectivity requirements are satisfied in G. This completes the proof.

3.2 An Upper Bound on Splitting-Off Attempts

Extending the ideas in the proof of Lemma 10, we present an algorithm to find a complete splitting-off sequence by making at most $O(|N(x)|)$ splitting-off attempts (to split-off to capacity). In the algorithm we maintain a non-admissible set C; initially $C = \emptyset$. The algorithm will apply one of the following three operations guaranteed by the following lemma. Here we assume that $\{u\}$ is a non-admissible set for every $u \in N(x)$. This can be achieved by a pre-processing step that split-off every (u, u) to capacity.

Lemma 11. *Suppose that C is a non-admissible set and there is a vertex $u \in N(x) - C$. Then, using at most three splitting-off attempts, at least one of the following operations can be applied:*

1. *Splitting-off an edge pair to capacity that voids an x-neighbor.*
2. *Deducing that every pair in $C \cup \{u\}$ is non-admissible, and add u to C.*
3. *Contracting a tight set T containing at least two x-neighbors.*

Proof. We consider three cases based on the size of C. When $|C| = 0$, we simply assign $C = \{u\}$. When $|C| = 1$, pick the vertex $v \in C$, and split-off (u, v) to capacity. Either case (1) applies when either u or v becomes void, or case (2) applies in the resulting graph after (u, v) is split-off to capacity. Hence, when $|C| \leq 1$, either case (1) or case (2) applies after only one splitting-off attempt.

The interesting case is when $|C| \geq 2$ and let $v_1, v_2 \in C$. Since C is a non-admissible set, by Lemma 10, there is a maximal dangerous set D containing C. First, we split-off (u, v_1) and (u, v_2) to capacity. If case (1) applies then we are done, so we assume that none of the three x-neighbors voids, implying that (u, v_1) and (u, v_2) are non-admissible in the resulting graph G' after splitting-off these edge pairs to capacity. Note that the edge pair (v_1, v_2) is also non-admissible since non-admissible edge pair in G remains non-admissible in G'. By Lemma 10, there exists a maximal dangerous set D' covering the non-admissible set $\{u, v_1, v_2\}$. Then inequality (4b) cannot hold for D and D', since $1 + 1 = s(D) + s(D') \geq s(D - D') + s(D' - D) + 2\bar{d}(D, D') \geq 0 + 0 + 2d(x, \{v_1, v_2\}) \geq 2 \cdot 2$. Therefore inequality (4a) must hold for D and D', hence $1 + 1 = s(D) + s(D') \geq s(D \cap D') + s(D \cup D')$.

This implies that either $D \cup D'$ is a dangerous set for which case (2) applies, since $C \cup \{u\}$ is contained in a dangerous set and hence every pair is a non-admissible pair by Proposition 5, or $D \cap D'$ is a tight set for which case (3) applies since v_1 and v_2 are x-neighbors. Note that v_1, v_2 are contained in a tight set if and only if after splitting-off one copy of (xv_1, xv_2) the connectivity requirement of some pair is violated by two. Hence this can be checked by one splitting-off attempt, and thus we can distinguish between case (2) and case (3), and in case (3) we can find such a tight set efficiently. Therefore, by making at most three splitting-off attempts $((xu, xv_1), (xu, xv_2), (xv_1, xv_2))$, one of the three operations can be applied. □

The following result can be obtained by applying Lemma 11 repeatedly.

Lemma 12. *The algorithm computes a complete edge splitting-off sequence using at most $O(|N(x)|)$ numbers of splitting-off attempts.*

Proof. The algorithm maintains the property that C is a non-admissible set, which holds at the beginning when $C = \emptyset$. It is clear that in case (2) the set C remains non-admissible. In case (1), by splitting-off an admissible pair, every pair of vertices in C remains non-admissible. Also, in case (3), by contracting a tight set, every pair of vertices in C remains non-admissible by Lemma 6.

The algorithm terminates when there is no vertex in $N(x) - C$. At that time, if $C = \emptyset$, then we have found a complete splitting-off sequence; if $C \neq \emptyset$, then by Mader's theorem (or by the proof in Section 3.1), this only happens if $d(x) = 3$ and $d(x)$ is odd at the beginning. In any case, the longest splitting-off sequence is found and the given complete edge splitting-off problem is solved.

It remains to prove that the total number of splitting-off attempts in the whole algorithm is at most $O(|N(x)|)$. To see this, we claim that each of the operations in Lemma 11 will be performed at most $|N(x)|$ times. Indeed, case (1) and (3) will be applied at most $|N(x)|$ times since each application reduces the number of x-neighbors by at least one, and case (2) will be applied at most $|N(x)|$ times since each application reduces the number of x-neighbors in $N(x) - C$ by one. □

3.3 Algorithm Outline

The following is an outline of the whole algorithm for the complete splitting-off problem. First we use the $O(m)$ time algorithm in Theorem 8 to construct a subgraph of G with $O(r_{\max} \cdot n)$ edges satisfying the connectivity requirements. To find a complete splitting-off sequence, we can thus restrict our attention to maintain the local edge-connectivities in this subgraph.

In the next preprocessing step, we will reduce the problem further to an instance where there is a particular *indicator vertex* $t \neq x$, with the property that for any pair of vertices $u, v \in V - x$ with $\lambda(u,v) \leq r_{\max}$, then it holds that $\lambda(u,v) = \min\{\lambda(u,t), \lambda(v,t)\}$. With this indicator vertex, to check the local edge-connectivity for all pairs with $\lambda(u,v) \leq r_{\max}$, we only need to check the local edge-connectivities from t to every vertex v with $\lambda(v,t) \leq r_{\max}$. This allows us to make only $O(n)$ queries (instead of $O(n^2)$ queries) to check the local edge-connectivities. This reduction step can be done by computing a partial Gomory-Hu tree and contracting appropriate tight sets; see the details in Section 3.4. The total preprocessing time is at most $\tilde{O}(m + {r_{\max}}^2 \cdot n)$, by using the fast Gomory-Hu tree algorithm in Theorem 9.

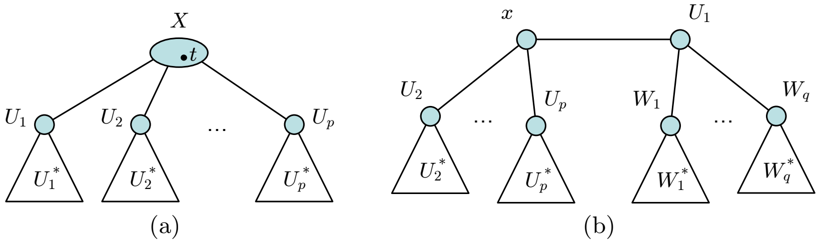
After these two preprocessing steps, we can perform a splitting-off attempt (split-off a pair to capacity) efficiently. For a vertex pair (u,v), we replace $\min\{d(x,u), d(x,v)\}$ copies of xu and xv by copies of uv, and then determine the maximum violation of connectivity requirements by constructing a partial Gomory-Hu tree and checking the local edge-connectivities from the indicator vertex t to every other vertex. If q is the maximum violation of the connectivity requirements, then exactly $\min\{d(x,u), d(x,v)\} - \lceil q/2 \rceil$ copies of (xu, xv) are admissible. Therefore, using Theorem 9, one splitting-off attempt can be performed in $\tilde{O}(r_{\max} \cdot m + n) = \tilde{O}({r_{\max}}^2 \cdot n)$ expected time. By Lemma 12, the complete splitting-off problem can be solved by at most $O(|N(x)|) = O(n)$ splitting-off attempts. Hence we obtain the following result.

Theorem 13. *The complete edge splitting-off problem can be solved in* $\tilde{O}({r_{\max}}^2 \cdot |N(x)| \cdot n) = \tilde{O}({r_{\max}}^2 \cdot n^2)$ *expected time.*

3.4 Indicator Vertex

We show how to reduce the problem into an instance with a particular indicator vertex $t \neq x$, with the property that if $\lambda(u,v) \leq r_{\max}$ for $u, v \neq x$, then $\lambda(u,v) = \min\{\lambda(u,t), \lambda(v,t)\}$. Hence if we could maintain the local edge-connectivity from t to v for every $v \in V - x$ with $\lambda(v,t) \leq r_{\max}$, then the connectivity requirements for every pair in $V - x$ will be satisfied. Furthermore, by maintaining the local edge-connectivity, the indicator vertex t will remain to be an indicator vertex, and therefore this procedure needs to be executed only once. Without loss of generality, we assume that the connectivity requirement for each pair of vertices $u, v \in V - x$ is equal to $\min\{\lambda(u,v), r_{\max}\}$, and $r(x,v) = 0$ for every $v \in V - x$.

First we compute a partial Gomory-Hu tree $T_{r_{\max}}$ in $\tilde{O}(r_{\max} \cdot m)$ time by Theorem 9, which is $\tilde{O}({r_{\max}}^2 \cdot n)$ after applying the sparsifying algorithm in Theorem 8. Recall that each node in $T_{r_{\max}}$ represents a subset of vertices in G. In the following we will use a capital letter (say U) to denote both a node in $T_{r_{\max}}$ and the corresponding subset of vertices in G. If $T_{r_{\max}}$ has only one node, then this means that the local edge-connectivity between every pair of vertices in G is at least $r_{\max}$. In this case, any vertex $t \neq x$ is an indicator vertex. So assume that $T_{r_{\max}}$ has at least two nodes. Let X be the node in $T_{r_{\max}}$ that contains x in G, and $U_1, \ldots, U_p$ be the nodes adjacent to X in $T_{r_{\max}}$, and let XU_1 be the edge in $T_{r_{\max}}$ with largest weight among XU_i for $1 \leq i \leq p$. See Figure (a).



Suppose X contains a vertex $t \neq x$ in G. The idea is to contract tight sets so that t will become an indicator vertex in the resulting graph. For any edge XU_i in $T_{r_{\max}}$, let T_i' be the component of $T_{r_{\max}}$ that contains U_i when XU_i is removed from $T_{r_{\max}}$. We claim that each $U_i^* := \cup_{U \in T_i'} U$ is a tight set in G; see Figure (a). By the definition of a Gomory-Hu tree, the local edge-connectivity between any vertex $u_i \in U_i$ and t is equal to the edge weight of XU_i in $T_{r_{\max}}$. Also, by the definition of a Gomory-Hu tree, $d(U_i^*)$ is equal to the weight of edge XU_i in $T_{r_{\max}}$. Therefore, U_i^* is a tight set in G, because $r(u_i, t) = \lambda(u_i, t) = d(U_i^*)$ for some pair $u_i, t \in V - x$. By Proposition 5, we can contract each U_i^* into a single vertex u_i for $1 \leq i \leq p$ without losing any information about admissible pairs in G. Since each U_i^* becomes a single vertex, the vertex t becomes an indicator vertex in the resulting graph.

Suppose X contains only x in G. Then U_1^* may not be a tight set, since there may not exist a pair $u, v \in V - x$ with $r(u, v) = \lambda(u, v) = d(U_1^*)$ (note that there is a vertex v with $\lambda(x, v) = d(U_1^*)$, but $r(x, v) = 0$ for every vertex v). In this case, we will contract some tight sets so that any vertex in U_1 will become an indicator vertex. Let $W_1 \neq X, \ldots, W_q \neq X$ be the nodes (if any) adjacent to U_1 in $T_{r_{\max}}$; see Figure (b). By using similar arguments as before, it can be shown that each U_i^* is a tight set for $2 \leq i \leq p$ (through $u_i \in U_i$ and $u_1 \in U_1$). Therefore we can contract each U_i^* into a single vertex u_i for $2 \leq i \leq p$. Similarly, we can argue that each W_j^* (defined analogously as U_i^*) is a tight set, and hence we can contract each W_j^* into a single vertex w_j for each $1 \leq j \leq q$. We can see that any vertex $t \in U_1$ is an indicator vertex in the resulting graph, because $\lambda(t, v) \geq \min\{\lambda(w, v), r_{\max}\}$ for any pair of vertices v, w.

Henceforth we can consider this resulting graph instead of G for the purpose of computing a complete splitting-off sequence, and using t as the indicator vertex to check connectivities. The running time of this procedure is dominated by the partial Gomory-Hu tree computation, which is at most $\tilde{O}({r_{\max}}^2 \cdot n)$.

3.5 Deterministic Algorithm

We describe how to modify the randomized algorithm in Theorem 13 to obtain a deterministic algorithm with the same running time. Every step in the algorithm is deterministic except the Gomory-Hu tree construction in Theorem 9. The randomized Gomory-Hu tree construction is used in two places. First it is used in finding an indicator vertex in Section 3.4, and for this purpose it is executed only once. Here we can replace it by a slower deterministic partial Gomory-Hu tree construction algorithm. It is well-known that a Gomory-Hu tree can be computed using at most $n-1$ max-flow computations [14]. By using the Ford-Fulkerson flow algorithm, one can obtain an $O({r_{\max}}^2 \cdot n^2)$-time deterministic algorithm to construct a partial Gomory-Hu tree $T_{r_{\max}}$. The randomized partial Gomory-Hu construction is also used in every splitting-off attempt to check whether the connectivity requirements are satisfied. With the indicator vertex t, this task reduces to checking the local edge-connectivities from t to other vertices, and there is a fast deterministic algorithm for this simpler task by Bhalgat et.al. [5].

Theorem 14 ([5]). *Given an undirected graph G and a vertex t, there is an $\tilde{O}(r_{\max} \cdot m)$-time deterministic algorithm to compute* $\min\{\lambda_G(t,v), r_{\max}\}$ *for all vertices $v \in G$.*

Thus we can replace the randomized partial Gomory-Hu tree algorithm by this algorithm, and so Theorem 13 still holds deterministically. Hence there is a deterministic $\tilde{O}({r_{\max}}^2 \cdot n^2)$ time algorithm for the complete splitting-off problem.

4 Structural Property and Randomized Algorithm

Before we give the proof of Theorem 3, we first show how to use it in a randomized edge splitting-off procedure to speedup the algorithm. By Theorem 3, when the degree of x is much larger than $2r_{\max}$, even a random edge pair will be admissible with probability at least $1 - 2r_{\max}/(d(x)-1)$. Using this observation, we show how to reduce $d(x)$ to $O(r_{\max})$ in $\tilde{O}({r_{\max}}^3 \cdot n)$ time. Then, by Theorem 13, the remaining edges can be split-off in $\tilde{O}({r_{\max}}^2 \cdot d(x) \cdot n) = \tilde{O}({r_{\max}}^3 \cdot n)$ time. So the total running time of the complete splitting-off algorithm is improved to $\tilde{O}(m + {r_{\max}}^3 \cdot n)$, proving Theorem 2.

The idea is to split-off many random edge pairs in parallel, before checking if some connectivity requirement is violated. Suppose that $2^{l+q-1} < d(x) \leq 2^{l+q}$ and $2^{l-1} < r_{\max} \leq 2^l$ for some positive integers l and q. To reduce $d(x)$ to 2^{l+q-1}, we need to split-off at most 2^{l+q-1} x-edges. Since each x-edge has fewer than $2r_{\max}$ non-admissible partners by Theorem 3, the probability that a random edge pair is admissible is at least $\frac{(d(x)-1)-2r_{\max}}{d(x)-1} \geq \frac{2^{l+q-1}-2^{l+1}}{2^{l+q-1}} = \frac{2^{q-2}-1}{2^{q-2}}$. Now, consider a random splitting-off operation that split-off at most 2^{q-2} edge pairs at random in parallel. The operation is successful if all the edge pairs are admissible. The probability for the operation to succeed is at least $(\frac{2^{q-2}-1}{2^{q-2}})^{2^{q-2}} = O(1)$. After each operation, we run the checking algorithm to determine whether this operation is successful or not. Consider an iteration that consists of $c \cdot \log n$ operations, for some constant c. The iteration is successful if it finds a set of 2^{q-2} admissible pairs, i.e. any of its operations succeeds. The probability for an iteration to fail is hence at most $1/n^c$ for $q \geq 3$. The time complexity of an iteration is $\tilde{O}({r_{\max}}^2 \cdot n)$.

Since each iteration reduces the degree of x by 2^{q-2}, with at most $2^{l+1} = O(r_{\max})$ successful iterations, we can then reduce $d(x)$ to 2^{l+q-1}, i.e. reduce $d(x)$ by half. This procedure is applicable as long as $q \geq 3$. Therefore, we can reduce $d(x)$ to 2^{l+2} by using this procedure for $O(\log n)$ times. The total running time is thus $\tilde{O}(2^{l+1} \cdot \log n \cdot {r_{\max}}^2 \cdot n) = \tilde{O}({r_{\max}}^3 \cdot n)$. Note that there are at most $\tilde{O}(r_{\max})$ iterations and the failure probability of each iteration is at most $1/n^c$. By the union bound, the probability for above randomized algorithm to fail is at most $1/n^{c-1}$. Therefore, with high probability, the algorithm succeeds in $\tilde{O}({r_{\max}}^3 \cdot n)$ time to reduce $d(x)$ to $O(r_{\max})$. Since the correctness of solution can be verified by a Gomory-Hu Tree, this also gives a Las Vegas algorithm with the same expected runtime.

4.1 Proof of Theorem 3

In this subsection we will prove that each edge has at most $2r_{\max} - 2$ non-admissible partners. Given an edge pair (xv, xw), if it is a non-admissible pair, then there is a dangerous set D with $\{xv, xw\} \subseteq \delta(D)$ by Proposition 5, and we say such a dangerous set D *covers* xv and xw. Let P be the set of non-admissible partners of xv in the initial graph. Our goal is to show that $|P| \leq 2r_{\max} - 2$.

Proposition 15 ([2] Lemma 5.4). *Suppose there is no cut edge incident to x. For any disjoint vertex sets S_1, S_2 with $d(S_1, S_2) = 0$ and $d(x, S_1) \geq 1$ and $d(x, S_2) \geq 1$, then $S_1 \cup S_2$ is not a dangerous set.*

We first present an outline of the proof. Let $\mathcal{D}_P$ be a minimal set of maximal dangerous sets such that (i) each set $D \in \mathcal{D}_P$ covers the edge xv and (ii) each edge in P is covered by some set $D \in \mathcal{D}_P$. First, we consider the base case with $|\mathcal{D}_P| \leq 2$. The theorem follows immediately if $|\mathcal{D}_P| = 1$, so assume $D_P = \{D_1, D_2\}$. By Proposition 15, $d(D_1 - D_2, D_1 \cap D_2) \geq 1$ as $\mathcal{D}_P$ is minimal. Hence $d(D, V - x - D) \geq 1$ for each $D \in \mathcal{D}_P$. Since $d(D) \leq r_{\max} + 1$ and D covers xv for each $D \in \mathcal{D}_P$, each set in $\mathcal{D}_P$ can cover at most $r_{\max} - 1$ non-admissible partner of xv, proving $|P| \leq 2r_{\max} - 2$.

The next step is to show that $|\mathcal{D}_P| \leq r_{\max} - 1$ when $|\mathcal{D}_P| \geq 3$, where the proofs of this step use very similar ideas as in [2, 24]. When $|\mathcal{D}_P| \geq 3$, we show in Lemma 16 that inequality (4a) must hold for each pair of dangerous sets in $\mathcal{D}_P$. Since each dangerous set is connected by Proposition 15, this allows us to conclude in Lemma 17 that $|\mathcal{D}_P| \leq r_{\max} - 1$. This implies that $|P| < r_{\max}^2$.

To improve this bound, we use a new inductive argument to show that $|P| \leq r_{\max} - 1 + |\mathcal{D}_P| \leq 2r_{\max} - 2$. First we prove in Lemma 18 that there is an admissible pair (xa, xb) in P (so by definition $a, b \neq v$). By splitting-off (xa, xb), let $P' = P - \{xa, xb\}$ with $|P'| = |P| - 2$. In the resulting graph, we prove in Lemma 19 that $|\mathcal{D}_{P'}| \leq |\mathcal{D}_P| - 2$. Hence, by repeating this reduction, we can show that after splitting-off $\lfloor |\mathcal{D}_P|/2 \rfloor$ pairs of edges in P, the remaining edges in P is covered by one dangerous set. Therefore, we can conclude that $|P| \leq r_{\max} - 1 + |\mathcal{D}_P| \leq 2r_{\max} - 2$. In the following we will first prove the upper bound on $|\mathcal{D}_P|$, then we will provide the details of the inductive argument.

An Upper Bound on $|\mathcal{D}_P|$: By contracting non-trivial tight sets, each edge in P is still a non-admissible partner of xv by Lemma 6. Henceforth, we will assume that all tight sets in G are singletons. Also we assume there is no cut edge incident to x and $r_{\max} \geq 2$ as required in the proof by Theorem 3. Recall that $\mathcal{D}_P$ is a minimal set of maximal dangerous sets such that (i) each set $D \in \mathcal{D}_P$ covers the edge xv and (ii) each edge in P is covered by some set $D \in \mathcal{D}_P$. We use the following result.

Lemma 16 ([2] Lemma 5.4, [24] Lemma 2.6). *If $|\mathcal{D}_P| \geq 3$, then inequality (4a) holds for every $X, Y \in \mathcal{D}_P$. Furthermore, $X \cap Y = \{v\}$ and is a tight set for any $X, Y \in \mathcal{D}_P$.*

Lemma 17. $|\mathcal{D}_P| \leq r_{\max} - 1$ *when* $|D_P| \geq 3$.

Proof. By Lemma 16, we have $X \cap Y = \{v\}$ for any $X, Y \in \mathcal{D}_P$. For each set $X \in \mathcal{D}_P$, we have $d(x,v) \geq 1$ and $d(x, X - v) \geq 1$ by the minimality of $\mathcal{D}_P$. Therefore, we must have $d(v, X - v) \geq 1$ by Proposition 15. By Lemma 16, $X - v$ and $Y - v$ are disjoint for each pair $X, Y \in \mathcal{D}_P$. Since $d(v, X - v) \geq 1$ for each $X \in \mathcal{D}_P$ and $d(x,v) \geq 1$, it follows that $|\mathcal{D}_P| \leq d(v) - 1$. By Lemma 16, $\{v\}$ is a tight set, and thus $|\mathcal{D}_P| \leq d(v) - 1 \leq r_{\max} - 1$. □

An Inductive Argument: The goal is to prove that $|P| \leq r_{\max} - 1 + |\mathcal{D}_P|$. By Lemma 17, this holds if $d(x, X - v) = 1$ for every dangerous set $X \in \mathcal{D}_P$. Hence we assume that there is a dangerous set $A \in \mathcal{D}_P$ with $d(x, A - v) \geq 2$; this property will only be used at the very end of the proof. By Lemma 16, inequality (4a) holds for A and B for every $B \in \mathcal{D}_P$. By the minimality of $\mathcal{D}_P$, there exists a x-neighbor $a \in A$ which is not contained in any other set in $\mathcal{D}_P$. Similarly, there exists $b \in B$ which is not contained in any other set in $\mathcal{D}_P$. The following lemma shows that the edge pair (xa, xb) is admissible.

Lemma 18. *For any $A, B \in \mathcal{D}_P$ satisfying inequality (4a), an edge pair (xa, xb) is admissible if $a \in A - B$ and $b \in B - A$.*

Proof. Suppose, by way of contradiction, that (xa, xb) is non-admissible. Then, by Proposition 5, there exists a maximal dangerous set C containing a and b. We claim that $v \in C$; otherwise there exists a 3-dangerous-set structure, contradicting Lemma 7. Then $d(x, A \cap C) \geq d(x, \{v, a\}) \geq 2$, and so inequality (4b) cannot hold for A and C, since $1 + 1 \geq s(A) + s(C) \geq s(A - C) + s(C - A) + 2\bar{d}(A, C) \geq 0 + 0 + 2 \cdot 2$. Therefore, inequality (4a) must hold for A and C. Since A and C are maximal dangerous sets, $A \cup C$ cannot be a dangerous set, and thus $1 + 1 \geq s(A) + s(C) \geq s(A \cup C) + s(A \cap C) + 2d(A, C) \geq 2 + s(A \cap C) + 0$, which implies that $A \cap C$ is a tight set, but this contradicts the assumption that each tight set is a singleton as $\{v, a\} \subseteq A \cap C$. □

After splitting-off (xa, xb), let the resulting graph be G' and $P' = P - \{xa, xb\}$. Clearly, since each edge in P' is a non-admissible partner of xv in G, every edge in P' is still a non-admissible partner of xv in G'. Furthermore, by contracting non-trivial tight sets in G', each edge in P' is still a non-admissible partner of xv by Lemma 6. Hence we assume all tight sets in G' are singletons. Let $\mathcal{D}_{P'}$ be a minimal set of maximal dangerous sets such that (i) each set $D \in \mathcal{D}_{P'}$ covers the edge xv and (ii) each edge in P' is covered by some set $D \in \mathcal{D}_{P'}$. The following lemma shows that there exists $\mathcal{D}_{P'}$ with $|\mathcal{D}_{P'}| \leq |\mathcal{D}_P| - 2$.

Lemma 19. *When $|\mathcal{D}_P| \geq 3$, the edges in P' can be covered by a set $\mathcal{D}_{P'}$ of maximal dangerous sets in G' such that (i) each set in $\mathcal{D}_{P'}$ covers xv, and (ii) each edge in P' is covered by some set in $\mathcal{D}_{P'}$, and (iii) $|\mathcal{D}_{P'}| \leq |\mathcal{D}_P| - 2$.*

Proof. We will use the dangerous sets in $\mathcal{D}_P$ to construct $\mathcal{D}_{P'}$. Since each pair of sets in $\mathcal{D}_P$ satisfies inequality (4a), we have $s(A \cup D) = 2$ before splitting-off (xa, xb) for each $D \in \mathcal{D}_P$. Also, before splitting-off (xa, xb), for $A, B, C \in \mathcal{D}_P$, inequality (4b) cannot hold for $A \cup B$ and C because $2 + 1 = s(A \cup B) + s(C) \geq s((A \cup B) - C) + s(C - (A \cup B)) + 2\bar{d}(A \cup B, C) \geq 2 + 0 + 2 \cdot 1$, where the last inequality follows since $v \in A \cap B \cap C$ and $(A \cup B) - C$ is not dangerous (as it covers the admissible edge pair (xa, xb)). Therefore inequality (4a) must hold for $A \cup B$ and C, which implies that $s(A \cup B \cup C) \leq 3$ since $2 + 1 = s(A \cup B) + s(C) \geq s((A \cup B) \cup C) + s((A \cup B) \cap C)$. For A and B as defined before Lemma 18, since $s(A \cup B) = 2$ before splitting-off (xa, xb), $A \cup B$ becomes a tight set after splitting-off (xa, xb). For any other set $C \in \mathcal{D}_P - A - B$, since $s(A \cup B \cup C) \leq 3$ before splitting-off (xa, xb), $A \cup B \cup C$ becomes a dangerous set after splitting-off (xa, xb). Hence, after splitting-off (xa, xb) and contracting the tight set $A \cup B$ into v, each set in $\mathcal{D}_P - A - B$ becomes a dangerous set. Then $\mathcal{D}_{P'} = \mathcal{D}_P - A - B$ is a set of dangerous sets covering each edge in P', satisfying properties (i)-(iii). By replacing a dangerous set $C \in \mathcal{D}_{P'}$ by a maximal dangerous set $C' \supseteq C$ and removing redundant dangerous sets in $\mathcal{D}_{P'}$ so that it minimally covers P', we have found $\mathcal{D}_{P'}$ as required by the lemma. $\square$

Recall that we chose A with $d(x, A - v) \geq 2$, and hence $d(x, v) \geq 2$ after the splitting-off and contraction of tight sets. Therefore, inequality (4a) holds for every two maximal dangerous sets in $\mathcal{D}_{P'}$. By induction, when $|\mathcal{D}_P| \geq 3$, we have $|P| = |P'| + 2 \leq r_{\max} - 1 + |\mathcal{D}_{P'}| + 2 \leq r_{\max} - 1 + |\mathcal{D}_P|$. In the base case when $|\mathcal{D}_P| = 2$ and $A, B \in \mathcal{D}_P$ satisfy (4a), the same argument in Lemma 19 can be used to show that the edges in P' is covered by one *tight* set after splitting-off (xa, xb), and thus $|P| = |P'| + 2 \leq r_{\max} - 1 + 2 \leq r_{\max} - 1 + |\mathcal{D}_P|$. This completes the proof that $|P| \leq r_{\max} - 1 + |\mathcal{D}_P|$, proving the theorem.

5 Concluding Remarks

Theorem 3 can be applied to constrained edge splitting-off problems, and give additive approximation algorithms for constrained augmentation problems. The efficient algorithms can also be adapted to these problems. We refer the reader to [25] for these results.

References

1. Bang-Jensen, J., Frank, A., Jackson, B.: Preserving and increasing local edge-connectivity in mixed graphs. SIAM J. Disc. Math. 8(2), 155–178 (1995)
2. Bang-Jensen, J., Jordán, T.: Edge-connectivity augmentation preserving simplicity. SIAM Journal on Discrete Mathematics 11(4), 603–623 (1998)

3. Bernáth, A., Király, T.: A new approach to splitting-off. In: Lodi, A., Panconesi, A., Rinaldi, G. (eds.) IPCO 2008. LNCS, vol. 5035, pp. 401–415. Springer, Heidelberg (2008)
4. Benczúr, A.A., Karger, D.R.: Augmenting undirected edge connectivity in $O(n^2)$ time. Journal of Algorithms 37(1), 2–36 (2000)
5. Bhalgat, A., Hariharan, R., Kavitha, T., Panigrahi, D.: An $\tilde{O}(mn)$ Gomory-Hu tree construction algorithm for unweighted graphs. In: STOC 2007, pp. 605–614 (2007)
6. Bhalgat, A., Hariharan, R., Kavitha, T., Panigrahi, D.: Fast edge splitting and Edmonds' arborescence construction for unweighted graphs. In: SODA '08, pp. 455–464 (2008)
7. Chan, Y.H., Fung, W.S., Lau, L.C., Yung, C.K.: Degree Bounded Network Design with Metric Costs. In: FOCS '08, pp. 125–134 (2008)
8. Cheng, E., Jordán, T.: Successive edge-connectivity augmentation problems. Mathematical Programming 84(3), 577–593 (1999)
9. Frank, A.: Augmenting graphs to meet edge-connectivity requirements. SIAM Journal on Discrete Mathematics 5(1), 25–53 (1992)
10. Frank, A.: On a theorem of Mader. Ann. of Disc. Math. 101, 49–57 (1992)
11. Frank, A., Király, Z.: Graph orientations with edge-connection and parity constraints. Combinatorica 22(1), 47–70 (2002)
12. Gabow, H.N.: Efficient splitting off algorithms for graphs. In: STOC '94, pp. 696–705 (1994)
13. Goemans, M.X., Bertsimas, D.J.: Survivable networks, linear programming relaxations and the parsimonious property. Math. Prog. 60(1), 145–166 (1993)
14. Gomory, R.E., Hu, T.C.: Multi-terminal network flows. Journal of the Society for Industrial and Applied Mathematics 9(4), 551–570 (1961)
15. Hariharan, R., Kavitha, T., Panigrahi, D.: Efficient algorithms for computing all low st edge connectivities and related problems. In: SODA '07, pp. 127–136 (2007)
16. Jordán, T.: On minimally k-edge-connected graphs and shortest k-edge-connected Steiner networks. Discrete Applied Mathematics 131(2), 421–432 (2003)
17. Lau, L.C.: An approximate max-Steiner-tree-packing min-Steiner-cut theorem. Combinatorica 27(1), 71–90 (2007)
18. Lovász, L.: Lecture. Conference of Graph Theory, Prague (1974); See also Combinatorial problems and exercises. North-Holland (1979)
19. Mader, W.: A reduction method for edge-connectivity in graphs. Annals of Discrete Mathematics 3, 145–164 (1978)
20. Nagamochi, H.: A fast edge-splitting algorithm in edge-weighted graphs. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 1263–1268 (2006)
21. Nagamochi, H., Ibaraki, T.: Linear time algorithm for finding a sparse k-connected spanning subgraph of a k-connected graph. Algorithmica 7(1), 583–596 (1992)
22. Nagamochi, H., Ibaraki, T.: Deterministic O(nm) time edge-splitting in undirected graphs. Journal of Combinatorial Optimization 1(1), 5–46 (1997)
23. Nash-Williams, C.S.J.A.: On orientations, connectivity and odd vertex pairings in finite graphs. Canadian Journal of Mathematics 12, 555–567 (1960)
24. Szigeti, Z.: Edge-splittings preserving local edge-connectivity of graphs. Discrete Applied Mathematics 156(7), 1011–1018 (2008)
25. Yung, C.K.: Edge splitting-off and network design problems. Master thesis, The Chinese University of Hong Kong (2009)

On Generalizations of Network Design Problems with Degree Bounds

Nikhil Bansal[1], Rohit Khandekar[1], Jochen Könemann[2], Viswanath Nagarajan[1], and Britta Peis[3]

[1] IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA
[2] University of Waterloo
[3] Technische Universität Berlin

Abstract. Iterative rounding and relaxation have arguably become the method of choice in dealing with unconstrained and constrained network design problems. In this paper we extend the scope of the iterative relaxation method in two directions: (1) by handling more complex degree constraints in the minimum spanning tree problem (namely *laminar* crossing spanning tree), and (2) by incorporating 'degree bounds' in other combinatorial optimization problems such as *matroid intersection* and *lattice polyhedra*. We give new or improved approximation algorithms, hardness results, and integrality gaps for these problems.

1 Introduction

Iterative rounding and relaxation have arguably become the method of choice in dealing with unconstrained and constrained network design problems. Starting with Jain's elegant *iterative rounding* scheme for the generalized Steiner network problem in [14], an extension of this technique (iterative *relaxation*) has more recently lead to breakthrough results in the area of constrained network design, where a number of linear constraints are added to a classical network design problem. Such constraints arise naturally in a wide variety of practical applications, and model limitations in processing power, bandwidth or budget. The design of powerful techniques to deal with these problems is therefore an important goal.

The most widely studied constrained network design problem is the *minimum-cost degree-bounded spanning tree* problem. In an instance of this problem, we are given an undirected graph, non-negative costs for the edges, and positive, integral degree-bounds for each of the nodes. The problem is easily seen to be NP-hard, even in the absence of edge-costs, since finding a spanning tree with maximum degree two is equivalent to finding a Hamiltonian Path. A variety of techniques have been applied to this problem [5,6,11,17,18,23,24], culminating in Singh and Lau's breakthrough result in [27]. They presented an algorithm that computes a spanning tree of at most optimum cost whose degree at each vertex v exceeds its bound by at most 1, using the *iterative relaxation* framework developed in [20,27].

The iterative relaxation technique has been applied to several constrained network design problems: spanning tree [27], survivable network design [20,21], directed graphs with intersecting and crossing super-modular connectivity [20,2]. It has also been applied to degree bounded versions of matroids and submodular flow [15].

In this paper we further extend the applicability of iterative relaxation, and obtain new or improved bicriteria approximation results for minimum crossing spanning tree (MCST), crossing matroid intersection, and crossing lattice polyhedra. We also provide hardness results and integrality gaps for these problems.

Notation. As is usual, when dealing with an undirected graph $G = (V, E)$, for any $S \subseteq V$ we let $\delta_G(S) := \{(u, v) \in E \mid u \in S,\ v \notin S\}$. When the graph is clear from context, the subscript is dropped. A collection $\{U_1, \cdots, U_t\}$ of vertex-sets is called *laminar* if for every pair U_i, U_j in this collection, we have $U_i \subseteq U_j$, $U_j \subseteq U_i$, or $U_i \cap U_j = \emptyset$. A $(\rho, f(b))$ approximation for minimum cost degree bounded problems refers to a solution that (1) has cost at most ρ times the optimum that satisfies the degree bounds, and (2) satisfies the relaxed degree constraints in which a bound b is replaced with a bound $f(b)$.

1.1 Our Results, Techniques and Paper Outline

Laminar MCST. Our main result is for a natural generalization of bounded-degree MST (called Laminar Minimum Crossing Spanning Tree or *laminar MCST*), where we are given an edge-weighted undirected graph with a laminar family $\mathcal{L} = \{S_i\}_{i=1}^m$ of vertex-sets having bounds $\{b_i\}_{i=1}^m$; and the goal is to compute a spanning tree of minimum cost that contains at most b_i edges from $\delta(S_i)$ for each $i \in [m]$.

The motivation behind this problem is in designing a network where there is a hierarchy (i.e. laminar family) of service providers that control nodes (i.e. vertices). The number of edges crossing the boundary of any service provider (i.e. its vertex-cut) represents some cost to this provider, and is therefore limited. The laminar MCST problem precisely models the question of connecting all nodes in the network while satisfying bounds imposed by all the service providers.

From a theoretical viewpoint, cut systems induced by laminar families are well studied, and are known to display rich structure. For example, *one-way cut-incidence matrices* are matrices whose rows are incidence vectors of directed cuts induced by the vertex-sets of a laminar family; It is well known (e.g., see [19]) that such matrices are totally unimodular. Using the laminar structure of degree-constraints and the iterative relaxation framework, we obtain the following main result, and present its proof in Section 2.

Theorem 1. *There is a polynomial time $(1, b + O(\log n))$ bicriteria approximation algorithm for laminar MCST. That is, the cost is no more than the optimum cost and the degree violation is at most additive $O(\log n)$. This guarantee is relative to the natural LP relaxation.*

This guarantee is substantially stronger than what follows from known results for the general *minimum crossing spanning tree* (MCST) problem: where the degree bounds could be on arbitrary edge-subsets $E_1, \ldots, E_m$. In particular, for general MCST a $(1, b + \Delta - 1)$ [2,15] is known where Δ is the maximum number of degree-bounds an edge appears in. However, this guarantee is not useful for laminar MCST as Δ can be as large as $\Omega(n)$ in this case. If a multiplicative factor in the degree violation is allowed, Chekuri et al. [8] recently gave a very elegant $\left(1, (1 + \epsilon)b + O(\frac{1}{\epsilon} \log m)\right)$ guarantee (which subsumes the previous best $(O(\log n), O(\log m)\, b)$ [4] result). However, these results also cannot be used to obtain a small additive violation, especially if b is large. In particular, both the results [4,8] for general MCST are based on the natural LP relaxation, for which there is an integrality gap of $b + \Omega(\sqrt{n})$ even without regard to costs and when $m = O(n)$ [26] (see also [3]). On the other hand, Theorem 1 shows that a purely additive $O(\log n)$ guarantee on degree (relative to the LP relaxation and even in presence of costs) is indeed achievable for MCST, when the degree-bounds arise from a laminar cut-family.

The algorithm in Theorem 1 is based on iterative relaxation and uses two main new ideas. Firstly, we drop a carefully chosen *constant fraction of degree-constraints* in each iteration. This is crucial as it can be shown that dropping one constraint at a time as in the usual applications of iterative relaxation can indeed lead to a degree violation of $\Omega(\Delta)$. Secondly, the algorithm does not just drop degree constraints, but in some iterations it also *generates new degree constraints*, by merging existing degree constraints.

All previous applications of iterative relaxation to constrained network design treat connectivity and degree constraints rather asymmetrically. While the structure of the connectivity constraints of the underlying LP is used crucially (e.g., in the ubiquitous uncrossing argument), the handling of degree constraints is remarkably simple. Constraints are dropped one by one, and the final performance of the algorithm is good only if the number of side constraints is small (e.g., in recent work by Grandoni et al. [12]), or if their structure is simple (e.g., if the 'frequency' of each element is small). In contrast, our algorithm for laminar MCST exploits the structure of degree constraints in a non-trivial manner.

Hardness Results. We obtain the following hardness of approximation for the *general MCST* problem (and its matroid counterpart). In particular this rules out any algorithm for MCST that has additive constant degree violation, even without regard to costs.

Theorem 2. *Unless $\mathcal{NP}$ has quasi-polynomial time algorithms, the MCST problem admits no polynomial time $O(\log^{\alpha} m)$ additive approximation for the degree bounds for some constant $\alpha > 0$; this holds even when there are no costs.*

The proof for this theorem is given in Section 3, and uses a a two-step reduction from the well-known *Label Cover* problem. First, we show hardness for a *uniform* matroid instance. In a second step, we then demonstrate how this implies the result for MCST claimed in Theorem 2.

Note that our hardness bound nearly matches the result obtained by Chekuri et al. in [8]. We note however that in terms of *purely* additive degree guarantees, a large gap remains. As noted above, there is a much stronger lower bound of $b + \Omega(\sqrt{n})$ for LP-based algorithms [26] (even without regard to costs), which is based on discrepancy. In light of the small number of known hardness results for discrepancy type problems, it is unclear how our bounds for MCST could be strengthened.

Degree Bounds in More General Settings. We consider crossing versions of other classic combinatorial optimization problems, namely *matroid intersection* and *lattice polyhedra*. We discuss our results briefly and defer the proofs to the full version of the paper [3].

Definition 1 (Minimum crossing matroid intersection problem). *Let $r_1, r_2 : 2^E \to \mathbb{Z}$ be two supermodular functions, $c : E \to \mathbb{R}$ and $\{E_i\}_{i \in I}$ be a collection of subsets of E with corresponding bounds $\{b_i\}_{i \in I}$. Then the goal is to minimize:*

$$\{c^T x \mid \quad x(S) \geq \max\{r_1(S), r_2(S)\}, \forall\, S \subseteq E; \quad x(E_i) \leq b_i,\ \ \forall\, i \in [m]; \quad x \in \{0,1\}^E\}.$$

We remark that there are alternate definitions of matroid intersection (e.g., see Schrijver [25]) and that our result below extends to those as well.

Let $\Delta = \max_{e \in E} |\{i \in [m] \mid e \in E_i\}|$ be the largest number of sets E_i that any element of E belongs to, and refer to it as *frequency*.

Theorem 3. *Any optimal basic solution x^* of the linear relaxation of the minimum crossing matroid intersection problem can be rounded into an integral solution $\hat{x}$ such that $\hat{x}(S) \geq \max\{r_1(S), r_2(S)\}$ for all $S \subseteq E$ and*

$$c^T \hat{x} \leq 2c^T x^* \quad \textit{and} \quad \hat{x}(E_i) \leq 2b_i + \Delta - 1 \quad \forall i \in I.$$

The algorithm for this theorem again uses iterative relaxation, and its proof is based on a ‘fractional token’ counting argument similar to the one used in [2].

An interesting special case is for the *bounded-degree arborescence* problem (where $\Delta = 1$). As the set of arborescences in a digraph can be expressed as the intersection of partition and graphic matroids, Theorem 3 readily implies a $(2, 2b)$ approximation for this problem. This is an improvement over the previously best-known $(2, 2b + 2)$ bound [20] for this problem.

The bounded-degree arborescence problem is potentially of wider interest since it is a relaxation of ATSP, and it is hoped that ideas from this problem lead to new ideas for ATSP. In fact Theorem 3 also implies an improved $(2, 2b)$-approximation for the *bounded-degree arborescence packing* problem, where the goal is to pack a given number of arc-disjoint arborescences while satisfying degree-bounds on vertices (arborescence packing can again be phrased as matroid intersection). The previously best known bound for this problem was $(2, 2b + 4)$ [2]. We also give the following integrality gap.

Theorem 4. *For any $\epsilon > 0$, there exists an instance of unweighted minimum crossing arborescence for which the LP is feasible, and any integral solution must violate the bound on some set $\{E_i\}_{i=1}^m$ by a multiplicative factor of at least $2 - \epsilon$. Moreover, this instance has $\Delta = 1$, and just one non-degree constraint.*

Thus Theorem 3 is the best one can hope for, relative to the LP relaxation. First, Theorem 4 implies that the multiplicative factor in the degree cannot be improved beyond 2 (even without regard to costs). Second, the lower bound for arborescences with costs presented in [2] implies that no cost-approximation ratio better than 2 is possible, without violating degrees by a factor greater than 2.

Crossing Lattice Polyhedra. Classical *lattice polyhedra* form a unified framework for various discrete optimization problems and go back to Hoffman and Schwartz [13] who proved their integrality. They are polyhedra of type

$$\{x \in [0,1]^E \mid x(\rho(S)) \geq r(S), \quad \forall S \in \mathcal{F}\}$$

where $\mathcal{F}$ is a *consecutive submodular* lattice, $\rho : \mathcal{F} \to 2^E$ is a mapping from $\mathcal{F}$ to subsets of the ground-set E, and $r \in \mathbb{R}^{\mathcal{F}}$ is supermodular. A key property of lattice polyhedra is that the uncrossing technique can be applied which turns out to be crucial in almost all iterative relaxation approaches for optimization problems with degree bounds. We refer the reader to [25] for a more comprehensive treatment of this subject.

We generalize our work further to *crossing lattice polyhedra* which arise from classical lattice polyhedra by adding "degree-constraints" of the form $a_i \leq x(E_i) \leq b_i$ for a given collection $\{E_i \subseteq E \mid i \in I\}$ and lower and upper bounds $a, b \in \mathbb{R}^I$. We mention that this model covers several important applications including the crossing matroid basis and crossing planar mincut problems, among others.

We can show that the standard LP relaxation for the general crossing lattice polyhedron problem is weak; details are deferred to the full version of the paper in [3]. For this reason, we henceforth focus on a restricted class of crossing lattice polyhedra in which the underlying lattice $(\mathcal{F}, \leq)$ satisfies the following monotonicity property

$$(*) \quad S < T \implies |\rho(S)| < |\rho(T)| \qquad \forall\, S, T \in \mathcal{F}.$$

We obtain the following theorem whose proof is given in [3].

Theorem 5. *For any instance of the crossing lattice polyhedron problem in which $\mathcal{F}$ satisfies property $(*)$, there exists an algorithm that computes an integral solution of cost at most the optimal, where all rank constraints are satisfied, and each degree bound is violated by at most an additive $2\Delta - 1$.*

We note that the above property $(*)$ is satisfied for matroids, and hence Theorem 5 matches the previously best-known bound [15] for degree bounded matroids (with both upper/lower bounds). Also note that property $(*)$ holds whenever $\mathcal{F}$ is ordered by inclusion. In this special case, we can improve the result to an additive $\Delta - 1$ approximation if only upper bounds are given.

1.2 Related Work

As mentioned earlier, the basic bounded-degree MST problem has been extensively studied [5,6,11,17,18,23,24,27]. The iterative relaxation technique for degree-constrained problems was developed in [20,27].

MCST was first introduced by Bilo et al. [4], who presented a randomized-rounding algorithm that computes a tree of cost $O(\log n)$ times the optimum where each degree constraint is violated by a multiplicative $O(\log n)$ factor and an additive $O(\log m)$ term. Subsequently, Bansal et al. [2] gave an algorithm that attains an optimal cost guarantee and an additive $\Delta - 1$ guarantee on degree; recall that Δ is the maximum number of degree constraints that an edge lies in. This algorithm used iterative relaxation as its main tool. Recently, Chekuri et al. [8] obtained an improved $\big(1, (1+\epsilon)b + O(\frac{1}{\epsilon}\log m)\big)$ approximation algorithm for MCST, for any $\epsilon > 0$; this algorithm is based on pipage rounding.

The minimum crossing matroid basis problem was introduced in [15], where the authors used iterative relaxation to obtain (1) $(1, b + \Delta - 1)$-approximation when there are only upper bounds on degree, and (2) $(1, b + 2\Delta - 1)$-approximation in the presence of both upper and lowed degree-bounds. The [8] result also holds in this matroid setting. [15] also considered a degree-bounded version of the *submodular flow* problem and gave a $(1, b+1)$ approximation guarantee.

The bounded-degree arborescence problem was considered in Lau et al. [20], where a $(2, 2b+2)$ approximation guarantee was obtained. Subsequently Bansal et al. [2] designed an algorithm that for any $0 < \epsilon \leq 1/2$, achieves a $(1/\epsilon, b_v/(1-\epsilon)+4)$ approximation guarantee. They also showed that this guarantee is the best one can hope for via the natural LP relaxation (for every $0 < \epsilon \leq 1/2$). In the absence of edge-costs, [2] gave an algorithm that violates degree bounds by at most an additive two. Recently Nutov [22] studied the arborescence problem under *weighted* degree constraints, and gave a $(2, 5b)$ approximation for it.

Lattice polyhedra were first investigated by Hoffman and Schwartz [13] and the natural LP relaxation was shown to be totally dual integral. Even though greedy-type algorithms are known for all examples mentioned earlier, so far no combinatorial algorithm has been found for lattice polyhedra in general. Two-phase greedy algorithms have been established only in cases where an underlying rank function satisfies a monotonicity property [10], [9].

2 Crossing Spanning Tree with Laminar Degree Bounds

In this section we prove Theorem 1 by presenting an iterative relaxation-based algorithm with the stated performance guarantee. During its execution, the algorithm selects and deletes edges, and it modifies the given laminar family of degree bounds. A generic iteration starts with a subset F of edges already picked in the solution, a subset E of *undecided* edges, i.e., the edges not yet picked or dropped from the solution, a laminar family $\mathcal{L}$ on V, and residual degree bounds $b(S)$ for each $S \in \mathcal{L}$.

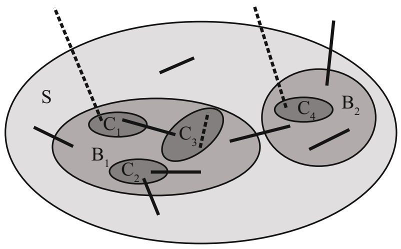
The laminar family $\mathcal{L}$ has a natural forest-like structure with *nodes* corresponding to each element of $\mathcal{L}$. A node $S \in \mathcal{L}$ is called the *parent* of node $C \in \mathcal{L}$ if S is the inclusion-wise minimal set in $\mathcal{L} \setminus \{C\}$ that contains C; and C is called a *child* of S. Node $D \in \mathcal{L}$ is called a *grandchild* of node $S \in \mathcal{L}$ if S is the parent of D's parent. Nodes $S, T \in \mathcal{L}$ are *siblings* if they have the same parent node. A node that has no parent is called *root*. The *level* of any node $S \in \mathcal{L}$ is the length of the path in this forest from S to the root of its tree. We also maintain a *linear ordering* of the children of each $\mathcal{L}$-node. A subset $\mathcal{B} \subseteq \mathcal{L}$ is called *consecutive* if all nodes in $\mathcal{B}$ are siblings (with parent S) and they appear consecutively in the ordering of S's children. In any iteration $(F, E, \mathcal{L}, b)$, the algorithm solves the following LP relaxation of the residual problem.

$$\begin{aligned}
\min \quad & \sum_{e \in E} c_e x_e & & (1)\\
\text{s.t.} \quad & x(E(V)) = |V| - |F| - 1 & & \\
& x(E(U)) \leq |U| - |F(U)| - 1 & \forall U \subset V & \\
& x(\delta_E(S)) \leq b(S) & \forall S \in \mathcal{L} & \\
& x_e \geq 0 & \forall e \in E &
\end{aligned}$$

For any vertex-subset $W \subseteq V$ and edge-set H, we let $H(W) := \{(u,v) \in H \mid u, v \in W\}$ denote the edges induced on W; and $\delta_H(W) := \{(u,v) \in H \mid u \in W,\ v \notin W\}$ the set of edges crossing W. The first two sets of constraints are spanning tree constraints while the third set corresponds to the degree bounds. Let x denote an optimal *extreme point solution* to this LP. By reducing degree bounds $b(S)$, if needed, we assume that x *satisfies all degree bounds at equality* (the degree bounds may therefore be fractional-valued). Let $\alpha := 24$.

Definition 2. *An edge $e \in E$ is said to be* local *for $S \in \mathcal{L}$ if e has at least one end-point in S but is neither in $E(C)$ nor in $\delta(C) \cap \delta(S)$ for any grandchild C of S. Let* $\mathrm{local}(S)$ *denote the set of local edges for S. A node $S \in \mathcal{L}$ is said to be* good *if* $|\mathrm{local}(S)| \leq \alpha$.

The figure on the left shows a set S, its children B_1 and B_2, and grand-children $C_1, \ldots, C_4$; edges in $\mathrm{local}(S)$ are drawn solid, non-local ones are shown dashed.



Initially, E is the set of edges in the given graph, $F \leftarrow \emptyset$, $\mathcal{L}$ is the original laminar family of vertex sets for which there are degree bounds, and an arbitrary linear ordering is chosen on the children of each node in $\mathcal{L}$. In a generic iteration $(F, E, \mathcal{L}, b)$, the algorithm performs one of the following steps (see also Figure 1):

1. If $x_e = 1$ for some edge $e \in E$ then $F \leftarrow F \cup \{e\}$, $E \leftarrow E \setminus \{e\}$, and set $b(S) \leftarrow b(S) - 1$ for all $S \in \mathcal{L}$ with $e \in \delta(S)$.
2. If $x_e = 0$ for some edge $e \in E$ then $E \leftarrow E \setminus \{e\}$.
3. **DropN:** Suppose there at least $|\mathcal{L}|/4$ good non-leaf nodes in $\mathcal{L}$. Then either odd-levels or even-levels contain a set $\mathcal{M} \subseteq \mathcal{L}$ of $|\mathcal{L}|/8$ good non-leaf nodes. Drop the degree bounds of all *children* of $\mathcal{M}$ and modify $\mathcal{L}$ accordingly. The ordering of siblings also extends naturally.
4. **DropL:** Suppose there are more than $|\mathcal{L}|/4$ good leaf nodes in $\mathcal{L}$, denoted by $\mathcal{N}$. Then partition $\mathcal{N}$ into parts corresponding to siblings in $\mathcal{L}$. For any part $\{N_1, \cdots, N_k\} \subseteq \mathcal{N}$ consisting of ordered (not necessarily contiguous) children of some node S:
   (a) Define $M_i = N_{2i-1} \cup N_{2i}$ for all $1 \leq i \leq \lfloor k/2 \rfloor$ (if k is odd N_k is not used).
   (b) Modify $\mathcal{L}$ by removing leaves $\{N_1, \cdots, N_k\}$ and adding new leaf-nodes $\{M_1, \cdots, M_{\lfloor k/2 \rfloor}\}$ as children of S (if k is odd N_k is removed). The children of S in the new laminar family are ordered as follows: each node M_i takes the position of either N_{2i-1} or N_{2i}, and other children of S are unaffected.
   (c) Set the degree bound of each M_i to $b(M_i) = b(N_{2i-1}) + b(N_{2i})$.

Assuming that one of the above steps applies at each iteration, the algorithm terminates when $E = \emptyset$ and outputs the final set F as a solution. It is clear that the algorithm outputs a spanning tree of G. An inductive argument (see e.g. [20]) can be used to show that the LP (1) is feasible at each each iteration and $c(F) + z_{cur} \leq z_o$ where z_o is the original LP value, z_{cur} is the current LP value, and F is the chosen edge-set at the current iteration. Thus the cost of the final solution is at most the initial LP optimum z_o. Next we show that one of the four iterative steps always applies.

Lemma 1. *In each iteration, one of the four steps above applies.*

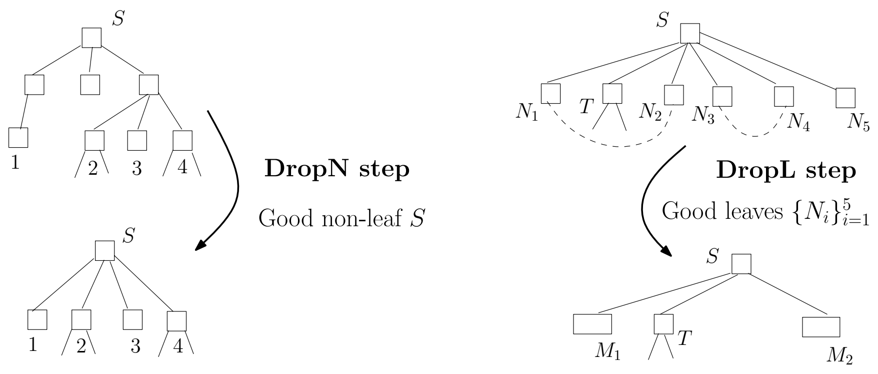


Fig. 1. Examples of the degree constraint modifications DropN and DropL

Proof. Let x^* be the optimal basic solution of (1), and suppose that the first two steps do not apply. Hence, we have $0 < x^*_e < 1$ for all $e \in E$. The fact that x^* is a basic solution together with a standard uncrossing argument (e.g., see [14]) implies that x^* is uniquely defined by

$$x(E(U)) = |U| - |F(U)| - 1 \quad \forall\, U \in \mathcal{S}, \qquad \text{and} \qquad x(\delta_E(S)) = b(S), \quad \forall\, S \in \mathcal{L}',$$

where $\mathcal{S}$ is a laminar subset of the tight spanning tree constraints, and $\mathcal{L}'$ is a subset of tight degree constraints, and where $|E| = |\mathcal{S}| + |\mathcal{L}'|$.

A simple counting argument (see, e.g., [27]) shows that there are at least 2 edges induced on each $S \in \mathcal{S}$ that are not induced on any of its children; so $2|\mathcal{S}| \le |E|$. Thus we obtain $|E| \le 2|\mathcal{L}'| \le 2|\mathcal{L}|$.

From the definition of local edges, we get that any edge $e = (u, v)$ is local to at most the following six sets: the smallest set $S_1 \in \mathcal{L}$ containing u, the smallest set $S_2 \in \mathcal{L}$ containing v, the parents P_1 and P_2 of S_1 and S_2 resp., the least-common-ancestor L of P_1 and P_2, and the parent of L. Thus $\sum_{S\in\mathcal{L}} |\mathrm{local}(S)| \le 6|E|$. From the above, we conclude that $\sum_{S\in\mathcal{L}} |\mathrm{local}(S)| \le 12|\mathcal{L}|$. Thus at least $|\mathcal{L}|/2$ sets $S \in \mathcal{L}$ must have $|\mathrm{local}(S)| \le \alpha = 24$, i.e., must be good. Now either at least $|\mathcal{L}|/4$ of them must be non-leaves or at least $|\mathcal{L}|/4$ of them must be leaves. In the first case, step 3 holds and in the second case, step 4 holds. ■

It remains to bound the violation in the degree constraints, which turns out to be rather challenging. We note that this is unlike usual applications of iterative rounding/relaxation, where the harder part is in showing that one of the iterative steps applies.

It is clear that the algorithm reduces the size of $\mathcal{L}$ by at least $|\mathcal{L}|/8$ in each DropN or DropL iteration. Since the initial number of degree constraints is at most $2n-1$, we get the following lemma.

Lemma 2. *The number of drop iterations (DropN and DropL) is* $T := O(\log n)$.

Performance guarantee for degree constraints. We begin with some notation. The iterations of the algorithm are broken into periods between successive drop iterations: there are exactly T drop-iterations (Lemma 2). In what follows, the t-th drop iteration is called *round* t. The *time* t refers to the instant just after round t; time 0 refers to the start of the algorithm. At any time t, consider the following parameters.

- $\mathcal{L}_t$ denotes the laminar family of degree constraints.
- E_t denotes the undecided edge set, i.e., support of the current LP optimal solution.
- For any set $\mathcal{B}$ of *consecutive siblings* in $\mathcal{L}_t$, $\mathsf{Bnd}(\mathcal{B}, t) = \sum_{N \in \mathcal{B}} b(N)$ equals the sum of the residual degree bounds on nodes of $\mathcal{B}$.
- For any set $\mathcal{B}$ of *consecutive siblings* in $\mathcal{L}_t$, $\mathsf{Inc}(\mathcal{B}, t)$ equals the number of edges from $\delta_{E_t}(\cup_{N \in \mathcal{B}} N)$ included in the final solution.

Recall that b denotes the *residual* degree bounds at any point in the algorithm. The following lemma is the main ingredient in bounding the degree violation.

Lemma 3. *For any set $\mathcal{B}$ of consecutive siblings in $\mathcal{L}_t$ (at any time t),* $\mathsf{Inc}(\mathcal{B}, t) \leq \mathsf{Bnd}(\mathcal{B}, t) + 4\alpha \cdot (T - t)$.

Observe that this implies the desired bound on each original degree constraint S: using $t = 0$ and $\mathcal{B} = \{S\}$, the violation is bounded by an additive $4\alpha \cdot T$ term.

Proof. The proof of this lemma is by induction on $T - t$. The base case $t = T$ is trivial since the only iterations after this correspond to including 1-edges: hence there is no violation in *any* degree bound, i.e. $\mathsf{Inc}(\{N\}, T) \leq b(N)$ for all $N \in \mathcal{L}_T$. Hence for *any* $\mathcal{B} \subseteq \mathcal{L}$, $\mathsf{Inc}(\mathcal{B}, T) \leq \sum_{N \in \mathcal{B}} \mathsf{Inc}(\{N\}, T) \leq \sum_{N \in \mathcal{B}} b(N) = \mathsf{Bnd}(\mathcal{B}, T)$.

Now suppose $t < T$, and assume the lemma for $t + 1$. Fix a consecutive $\mathcal{B} \subseteq \mathcal{L}_t$. We consider different cases depending on what kind of drop occurs in round $t + 1$.

DropN round. Here either all nodes in $\mathcal{B}$ get dropped or none gets dropped.

Case 1: *None of $\mathcal{B}$ is dropped.* Then observe that $\mathcal{B}$ is consecutive in $\mathcal{L}_{t+1}$ as well; so the inductive hypothesis implies $\mathsf{Inc}(\mathcal{B}, t + 1) \leq \mathsf{Bnd}(\mathcal{B}, t + 1) + 4\alpha \cdot (T - t - 1)$. Since the only iterations between round t and round $t + 1$ involve edge-fixing, we have $\mathsf{Inc}(\mathcal{B}, t) \leq \mathsf{Bnd}(\mathcal{B}, t) - \mathsf{Bnd}(\mathcal{B}, t+1) + \mathsf{Inc}(\mathcal{B}, t+1) \leq \mathsf{Bnd}(\mathcal{B}, t) + 4\alpha \cdot (T - t - 1) \leq \mathsf{Bnd}(\mathcal{B}, t) + 4\alpha \cdot (T - t)$.

Case 2: *All of $\mathcal{B}$ is dropped.* Let $\mathcal{C}$ denote the set of all children (in $\mathcal{L}_t$) of nodes in $\mathcal{B}$. Note that $\mathcal{C}$ consists of consecutive siblings in $\mathcal{L}_{t+1}$, and inductively $\mathsf{Inc}(\mathcal{C}, t + 1) \leq \mathsf{Bnd}(\mathcal{C}, t + 1) + 4\alpha \cdot (T - t - 1)$. Let $S \in \mathcal{L}_t$ denote the parent of the $\mathcal{B}$-nodes; so $\mathcal{C}$ are grand-children of S in $\mathcal{L}_t$. Let x denote the optimal LP solution *just before* round $t + 1$ (when the degree bounds are still given by $\mathcal{L}_t$), and $H = E_{t+1}$ the support edges of x. At that point, we have $b(N) = x(\delta(N))$ for all $N \in \mathcal{B} \cup \mathcal{C}$. Also let $\mathsf{Bnd}'(\mathcal{B}, t + 1) := \sum_{N \in \mathcal{B}} b(N)$ be the sum of bounds on $\mathcal{B}$-nodes just before round $t + 1$. Since S is a good node in round $t + 1$, $|\mathsf{Bnd}'(\mathcal{B}, t + 1) - \mathsf{Bnd}(\mathcal{C}, t + 1)| = |\sum_{N \in \mathcal{B}} b(N) - \sum_{M \in \mathcal{C}} b(M)| = |\sum_{N \in \mathcal{B}} x(\delta(N)) - \sum_{M \in \mathcal{C}} x(\delta(M))| \leq 2\alpha$. The last inequality follows since S is good; the factor of 2 appears since some edges, e.g., the edges between two children or two grandchildren of S, may get counted twice. Note also that the symmetric difference of $\delta_H(\cup_{N \in \mathcal{B}} N)$ and $\delta_H(\cup_{M \in \mathcal{C}} M)$ is contained in $\text{local}(S)$. Thus $\delta_H(\cup_{N \in \mathcal{B}} N)$ and $\delta_H(\cup_{M \in \mathcal{C}} M)$ differ in at most α edges.

Again since all iterations between time t and $t + 1$ are edge-fixing:

$$\begin{aligned}\mathsf{Inc}(\mathcal{B}, t) \leq\ & \mathsf{Bnd}(\mathcal{B}, t) - \mathsf{Bnd}'(\mathcal{B}, t + 1) + |\delta_H(\cup_{N \in \mathcal{B}} N) \setminus \delta_H(\cup_{M \in \mathcal{C}} M)| \\ & + \mathsf{Inc}(\mathcal{C}, t + 1)\end{aligned}$$

$$\begin{aligned}
&\leq \mathsf{Bnd}(\mathcal{B},t) - \mathsf{Bnd}'(\mathcal{B},t+1) + \alpha + \mathsf{Inc}(\mathcal{C},t+1)\\
&\leq \mathsf{Bnd}(\mathcal{B},t) - \mathsf{Bnd}'(\mathcal{B},t+1) + \alpha + \mathsf{Bnd}(\mathcal{C},t+1) + 4\alpha\cdot(T-t-1)\\
&\leq \mathsf{Bnd}(\mathcal{B},t) - \mathsf{Bnd}'(\mathcal{B},t+1) + \alpha + \mathsf{Bnd}'(\mathcal{B},t+1) + 2\alpha + 4\alpha\cdot(T-t-1)\\
&\leq \mathsf{Bnd}(\mathcal{B},t) + 4\alpha\cdot(T-t)
\end{aligned}$$

The first inequality above follows from simple counting; the second follows since $\delta_H(\cup_{N\in\mathcal{B}}N)$ and $\delta_H(\cup_{M\in\mathcal{C}}M)$ differ in at most α edges; the third is the induction hypothesis, and the fourth is $\mathsf{Bnd}(\mathcal{C},t+1) \leq \mathsf{Bnd}'(\mathcal{B},t+1) + 2\alpha$ (as shown above).

DropL round. In this case, let S be the parent of $\mathcal{B}$-nodes in $\mathcal{L}_t$, and $\mathcal{N} = \{N_1,\cdots,N_p\}$ be all the ordered children of S, of which $\mathcal{B}$ is a subsequence (since it is consecutive). Suppose indices $1 \leq \pi(1) < \pi(2) < \cdots < \pi(k) \leq p$ correspond to good leaf-nodes in $\mathcal{N}$. Then for each $1 \leq i \leq \lfloor k/2 \rfloor$, nodes $N_{\pi(2i-1)}$ and $N_{\pi(2i)}$ are merged in this round. Let $\{\pi(i) \mid e \leq i \leq f\}$ (possibly empty) denote the indices of good leaf-nodes in $\mathcal{B}$. Then it is clear that the only nodes of $\mathcal{B}$ that may be merged with nodes outside $\mathcal{B}$ are $N_{\pi(e)}$ and $N_{\pi(f)}$; all other $\mathcal{B}$-nodes are either not merged or merged with another $\mathcal{B}$-node. Let $\mathcal{C}$ be the inclusion-wise minimal set of *children of S in $\mathcal{L}_{t+1}$* s.t.

- $\mathcal{C}$ is consecutive in $\mathcal{L}_{t+1}$,
- $\mathcal{C}$ contains all nodes of $\mathcal{B} \setminus \{N_{\pi(i)}\}_{i=1}^{k}$, and
- $\mathcal{C}$ contains all new leaf nodes resulting from merging *two good leaf nodes* of $\mathcal{B}$.

Note that $\cup_{M\in\mathcal{C}}M$ consists of some subset of $\mathcal{B}$ and at most two good leaf-nodes in $\mathcal{N}\setminus\mathcal{B}$. These two extra nodes (if any) are those merged with the good leaf-nodes $N_{\pi(e)}$ and $N_{\pi(f)}$ of $\mathcal{B}$. Again let $\mathsf{Bnd}'(\mathcal{B},t+1) := \sum_{N\in\mathcal{B}} b(N)$ denote the sum of bounds on $\mathcal{B}$ just before drop round $t+1$, when degree constraints are $\mathcal{L}_t$. Let $H = E_{t+1}$ be the undecided edges in round $t+1$. By the definition of bounds on merged leaves, we have $\mathsf{Bnd}(\mathcal{C},t+1) \leq \mathsf{Bnd}'(\mathcal{B},t+1) + 2\alpha$. The term 2α is present due to the two extra good leaf-nodes described above.

Claim 6. *We have* $|\delta_H(\cup_{N\in\mathcal{B}}N) \setminus \delta_H(\cup_{M\in\mathcal{C}}M)| \leq 2\alpha$.

Proof. We say that $N \in \mathcal{N}$ is represented in $\mathcal{C}$ if either $N \in \mathcal{C}$ or N is contained in some node of $\mathcal{C}$. Let $\mathcal{D}$ be set of nodes of $\mathcal{B}$ that are *not* represented in $\mathcal{C}$ and the nodes of $\mathcal{N}\setminus\mathcal{B}$ that are represented in $\mathcal{C}$. Observe that by definition of $\mathcal{C}$, the set $\mathcal{D} \subseteq \{N_{\pi(e-1)}, N_{\pi(e)}, N_{\pi(f)}, N_{\pi(f+1)}\}$; in fact it can be easily seen that $|\mathcal{D}| \leq 2$. Moreover $\mathcal{D}$ consists of only good leaf nodes. Thus, we have $|\cup_{L\in\mathcal{D}}\delta_H(L)| \leq 2\alpha$. Now note that the edges in $\delta_H(\cup_{N\in\mathcal{B}}N) \setminus \delta_H(\cup_{M\in\mathcal{C}}M)$ must be in $\cup_{L\in\mathcal{D}}\delta_H(L)$. This completes the proof. ■

As in the previous case, we have:

$$\begin{aligned}
\mathsf{Inc}(\mathcal{B},t) &\leq \mathsf{Bnd}(\mathcal{B},t) - \mathsf{Bnd}'(\mathcal{B},t+1) + |\delta_H(\cup_{N\in\mathcal{B}}N) \setminus \delta_H(\cup_{M\in\mathcal{C}}M)|\\
&\qquad + \mathsf{Inc}(\mathcal{C},t+1)\\
&\leq \mathsf{Bnd}(\mathcal{B},t) - \mathsf{Bnd}'(\mathcal{B},t+1) + 2\alpha + \mathsf{Inc}(\mathcal{C},t+1)\\
&\leq \mathsf{Bnd}(\mathcal{B},t) - \mathsf{Bnd}'(\mathcal{B},t+1) + 2\alpha + \mathsf{Bnd}(\mathcal{C},t+1) + 4\alpha\cdot(T-t-1)\\
&\leq \mathsf{Bnd}(\mathcal{B},t) - \mathsf{Bnd}'(\mathcal{B},t+1) + 2\alpha + \mathsf{Bnd}'(\mathcal{B},t+1) + 2\alpha + 4\alpha\cdot(T-t-1)\\
&= \mathsf{Bnd}(\mathcal{B},t) + 4\alpha\cdot(T-t)
\end{aligned}$$

The first inequality follows from simple counting; the second uses Claim 6, the third is the induction hypothesis (since $\mathcal{C}$ is consecutive), and the fourth is $\mathsf{Bnd}(\mathcal{C}, t+1) \leq \mathsf{Bnd}'(\mathcal{B}, t+1) + 2\alpha$ (from above).

This completes the proof of the inductive step and hence Lemma 3. ■

3 Hardness Results

We now prove Theorem 2. The first step to proving this result is a hardness for the more general minimum crossing matroid basis problem: given a matroid $\mathcal{M}$ on a ground set V of elements, a cost function $c : V \to \mathbb{R}_+$, and degree bounds specified by pairs $\{(E_i, b_i)\}_{i=1}^m$ (where each $E_i \subseteq V$ and $b_i \in \mathbb{N}$), find a minimum cost basis I in $\mathcal{M}$ such that $|I \cap E_i| \leq b_i$ for all $i \in [m]$.

Theorem 7. *Unless $\mathcal{NP}$ has quasi-polynomial time algorithms, the unweighted minimum crossing matroid basis problem admits no polynomial time $O(\log^c m)$ additive approximation for the degree bounds for some fixed constant $c > 0$.*

Proof. We reduce from the label cover problem [1]. The input is a graph $G = (U, E)$ where the vertex set U is partitioned into pieces $U_1, \cdots, U_n$ each having size q, and all edges in E are between distinct pieces. We say that there is a *superedge* between U_i and U_j if there is an edge connecting some vertex in U_i to some vertex in U_j. Let t denote the total number of superedges; i.e.,

$$t = \left| \left\{ (i,j) \in \binom{[n]}{2} : \text{ there is an edge in } E \text{ between } U_i \text{ and } U_j \right\} \right|$$

The goal is to pick one vertex from each part $\{U_i\}_{i=1}^n$ so as to maximize the number of induced edges. This is called the value of the label cover instance and is at most t.

It is well known that there exists a universal constant $\gamma > 1$ such that for every $k \in \mathbb{N}$, there is a reduction from any instance of SAT (having size N) to a label cover instance $\langle G = (U, E), q, t \rangle$ such that:

– If the SAT instance is satisfiable, the label cover instance has optimal value t.
– If the SAT instance is not satisfiable, the label cover instance has optimal value $< t/\gamma^k$.
– $|G| = N^{O(k)}$, $q = 2^k$, $|E| \leq t^2$, and the reduction runs in time $N^{O(k)}$.

We consider a uniform matroid $\mathcal{M}$ with rank t on ground set E (recall that any subset of t edges is a basis in a uniform matroid). We now construct a crossing matroid basis instance $\mathcal{I}$ on $\mathcal{M}$. There is a set of degree bounds corresponding to each $i \in [n]$: for every collection C of edges incident to vertices in U_i such that no two edges in C are incident to the same vertex in U_i, there is a degree bound in $\mathcal{I}$ requiring *at most one* element to be chosen from C. Note that the number of degree bounds m is at most $|E|^q \leq N^{O(k\,2^k)}$. The following claim links the SAT and crossing matroid instances. Its proof is deferred to the full version of this paper.

Claim 8. [Yes instance] *If the SAT instance is satisfiable, there is a basis (i.e. subset $B \subseteq E$ with $|B| = t$) satisfying all degree bounds.*

[No instance] *If the SAT instance is unsatisfiable, every subset $B' \subseteq E$ with $|B'| \geq t/2$ violates some degree bound by an additive $\rho = \gamma^{k/2}/\sqrt{2}$.*

The steps described in the above reduction can be done in time polynomial in m and $|G|$. Also, instead of randomly choosing vertices from the sets W_i, we can use conditional expectations to derive a deterministic algorithm that recovers at least t/ρ^2 edges. Setting $k = \Theta(\log\log N)$ (recall that N is the size of the original SAT instance), we obtain an instance of bounded-degree matroid basis of size $\max\{m, |G|\} = N^{\log^a N}$ and $\rho = \log^b N$, where $a, b > 0$ are constants. Note that $\log m = \log^{a+1} N$, which implies $\rho = \log^c m$ for $c = \frac{b}{a+1} > 0$, a constant. Thus it follows that for this constant $c > 0$ the bounded-degree matroid basis problem has no polynomial time $O(\log^c m)$ *additive approximation* for the degree bounds, unless $\mathcal{NP}$ has quasi-polynomial time algorithms. ■

We now prove Theorem 2.

Proof. [Proof of Theorem 2] We show how the bases of a uniform matroid can be represented in a suitable instance of the crossing spanning tree problem. Let the uniform matroid from Theorem 7 consist of e elements and have rank $t \le e$; recall that $t \ge \sqrt{e}$ and clearly $m \le 2^e$. We construct a graph as in Figure 2, with vertices $v_1, \cdots, v_e$ corresponding to elements in the uniform matroid. Each vertex v_i is connected to the root r by two vertex-disjoint paths: $\langle v_i, u_i, r\rangle$ and $\langle v_i, w_i, r\rangle$. There are no costs in this instance. Corresponding to each degree bound (in the uniform matroid) of $b(C)$ on a subset $C \subseteq [e]$, there is a constraint to pick at most $|C| + b(C)$ edges from $\delta(\{u_i \mid i \in C\})$. Additionally, there is a *special degree bound* of $2e - t$ on the edge-set $E' = \bigcup_{i=1}^{e} \delta(w_i)$; this corresponds to picking a basis in the uniform matroid.

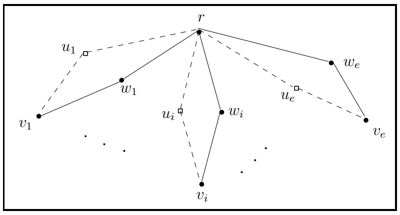


Fig. 2. The crossing spanning tree instance used in the reduction

Observe that for each $i \in [e]$, any spanning tree must choose exactly three edges amongst $\{(r, u_i), (u_i, v_i), (r, w_i), (w_i, v_i)\}$, in fact any three edges suffice. Hence every spanning tree T in this graph corresponds to a subset $X \subseteq [e]$ such that: (I) T contains both edges in $\delta(u_i)$ and one edge from $\delta(w_i)$, for each $i \in X$, and (II) T contains both edges in $\delta(w_i)$ and one edge from $\delta(u_i)$ for each $i \in [e] \setminus X$.

From Theorem 7, for the crossing matroid problem, we obtain the two cases:

Yes instance. There is a basis B^* (i.e. $B^* \subseteq [e]$, $|B^*| = t$) satisfying all degree bounds. Consider the spanning tree

$$T^* = \{(r, u_i), (u_i, v_i), (r, w_i) \mid i \in B^*\} \bigcup \{(r, w_i), (u_i, w_i), (r, u_i) \mid i \in [e] \setminus B^*\}.$$

Since B^* satisfies its degree-bounds, T^* satisfies all degree bounds derived from the crossing matroid instance. For the special degree bound on E', note that $|T^* \cap E'| = 2e - |B^*| = 2e - t$; so this is also satisfied. Thus there is a spanning tree satisfying all the degree bounds.

No instance. Every subset $B' \subseteq [e]$ with $|B'| \ge t/2$ (i.e. near basis) violates some degree bound by an additive $\rho = \Omega(\log^c m)$ term, where $c > 0$ is a fixed constant. Consider any spanning tree T that corresponds to subset $X \subseteq [e]$ as described above.

1. Suppose that $|X| \leq t/2$; then we have $|T \cap E'| = 2e - |X| \geq 2e - t + \frac{t}{2}$, i.e. the special degree bound is violated by $t/2 \geq \Omega(\sqrt{e}) = \Omega(\log^{1/2} m)$.
2. Now suppose that $|X| \geq t/2$. Then by the guarantee on the no-instance, T violates some degree-bound derived from the crossing matroid instance by additive ρ.

Thus in either case, every spanning tree violates some degree bound by additive $\rho = \Omega(\log^c m)$.

By Theorem 7, it is hard to distinguish the above cases and we obtain the corresponding hardness result for crossing spanning tree, as claimed in Theorem 2. ■

References

1. Arora, S., Babai, L., Stern, J., Sweedyk, Z.: The hardness of approximate optima in lattices, codes, and systems of linear equations. J. Comput. Syst. Sci. 54(2), 317–331 (1997)
2. Bansal, N., Khandekar, R., Nagarajan, V.: Additive guarantees for degree bounded network design. In: STOC, pp. 769–778 (2008)
3. Bansal, N., Khandekar, R., Könemann, J., Nagarajan, V., Peis, B.: On Generalizations of Network Design Problems with Degree Bounds (full version),Technical Report (2010)
4. Bilo, V., Goyal, V., Ravi, R., Singh, M.: On the crossing spanning tree problem. In: Jansen, K., Khanna, S., Rolim, J.D.P., Ron, D. (eds.) RANDOM 2004 and APPROX 2004. LNCS, vol. 3122, pp. 51–60. Springer, Heidelberg (2004)
5. Chaudhuri, K., Rao, S., Riesenfeld, S., Talwar, K.: What would Edmonds do? Augmenting paths and witnesses for degree-bounded MSTs. In: Chekuri, C., Jansen, K., Rolim, J.D.P., Trevisan, L. (eds.) APPROX 2005 and RANDOM 2005. LNCS, vol. 3624, pp. 26–39. Springer, Heidelberg (2005)
6. Chaudhuri, K., Rao, S., Riesenfeld, S., Talwar, K.: Push relabel and an improved approximation algorithm for the bounded-degree MST problem. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4051, pp. 191–201. Springer, Heidelberg (2006)
7. Chazelle, B.: The Discrepancy Method: Randomness and Complexity. Cambridge University Press, Cambridge (2000)
8. Chekuri, C., Vondrák, J., Zenklusen, R.: Dependent Randomized Rounding for Matroid Polytopes and Applications (2009), http://arxiv.org/abs/0909.4348
9. Faigle, U., Peis, B.: Two-phase greedy algorithms for some classes of combinatorial linear programs. In: SODA, pp. 161–166 (2008)
10. Frank, A.: Increasing the rooted connectivity of a digraph by one. Math. Programming 84, 565–576 (1999)
11. Goemans, M.X.: Minimum Bounded-Degree Spanning Trees. In: FOCS, pp. 273–282 (2006)
12. Grandoni, F., Ravi, R., Singh, M.: Iterative Rounding for Multiobjective Optimization Problems. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 95–106. Springer, Heidelberg (2009)
13. Hoffman, A., Schwartz, D.E.: On lattice polyhedra. In: Hajnal, A., Sos, V.T. (eds.) Proceedings of Fifth Hungarian Combinatorial Coll, pp. 593–598. North-Holland, Amsterdam (1978)
14. Jain, K.: A factor 2 approximation algorithm for the generalized Steiner network problem. In: Combinatorica, pp. 39–61 (2001)
15. Király, T., Lau, L.C., Singh, M.: Degree bounded matroids and submodular flows. In: Lodi, A., Panconesi, A., Rinaldi, G. (eds.) IPCO 2008. LNCS, vol. 5035, pp. 259–272. Springer, Heidelberg (2008)

16. Klein, P.N., Krishnan, R., Raghavachari, B., Ravi, R.: Approximation algorithms for finding low degree subgraphs. Networks 44(3), 203–215 (2004)
17. Könemann, J., Ravi, R.: A matter of degree: Improved approximation algorithms for degree bounded minimum spanning trees. SIAM J. on Computing 31, 1783–1793 (2002)
18. Könemann, J., Ravi, R.: Primal-Dual meets local search: approximating MSTs with nonuniform degree bounds. SIAM J. on Computing 34(3), 763–773 (2005)
19. Korte, B., Vygen, J.: Combinatorial Optimization, 4th edn. Springer, New York (2008)
20. Lau, L.C., Naor, J., Salavatipour, M.R., Singh, M.: Survivable network design with degree or order constraints (full version). In: STOC, pp. 651–660 (2007)
21. Lau, L.C., Singh, M.: Additive Approximation for Bounded Degree Survivable Network Design. In: STOC, pp. 759–768 (2008)
22. Nutov, Z.: Approximating Directed Weighted-Degree Constrained Networks. In: Goel, A., Jansen, K., Rolim, J.D.P., Rubinfeld, R. (eds.) APPROX 2008 and RANDOM 2008. LNCS, vol. 5171, pp. 219–232. Springer, Heidelberg (2008)
23. Ravi, R., Marathe, M.V., Ravi, S.S., Rosenkrantz, D.J., Hunt, H.B.: Many birds with one stone: Multi-objective approximation algorithms. In: STOC, pp. 438–447 (1993)
24. Ravi, R., Singh, M.: Delegate and Conquer: An LP-based approximation algorithm for Minimum Degree MSTs. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4051, pp. 169–180. Springer, Heidelberg (2006)
25. Schrijver, A.: Combinatorial Optimization. Springer, Heidelberg (2003)
26. Singh, M.: Personal Communication (2008)
27. Singh, M., Lau, L.C.: Approximating minimum bounded degree spanning trees to within one of optimal. In: STOC, pp. 661–670 (2007)

A Polyhedral Study of the Mixed Integer Cut

Steve Tyber and Ellis L. Johnson

H. Milton Stewart School of Industrial and Systems Engineering,
Georgia Institute of Technology, Atlanta, GA USA
{styber,ejohnson}@isye.gatech.edu

Abstract. General purpose cutting planes have played a central role in modern IP solvers. In practice, the Gomory mixed integer cut has proven to be among the most useful general purpose cuts. One may obtain this inequality from the group relaxation of an IP, which arises by relaxing non-negativity on the basic variables. We study the mixed integer cut as a facet of the master cyclic group polyhedron and characterize its extreme points and adjacent facets in this setting. Extensions are provided under automorphic and homomorphic mappings.

Keywords: Integer Programming, Group Relaxation, Master Cyclic Group Polyhedron, Master Knapsack Polytope, Cutting Planes.

1 Introduction

Consider the integer program

$$\min(cx : Ax = b, x \in \mathbb{Z}^n_+), \tag{1}$$

where $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, and $c \in \mathbb{R}^n$. Given a basis B of the LP relaxation of (1), the *group relaxation* of X, is obtained by relaxing non-negativity on x_B, i.e.

$$X_{\text{GR}} = \{x : Bx_B + Nx_N = b, x_B \in \mathbb{Z}^m, x_N \in \mathbb{Z}^{n-m}_+\}.$$

It follows that for an integer vector x_N, x_B is integral if and only if $Nx_N \equiv b \pmod{B}$; that is, $Nx_N - b$ belongs to the lattice generated by the columns of B.

Consider the group $\mathcal{G}$ of equivalence classes of $\mathbb{Z}^n$ modulo B. Let $\mathcal{N}$ be the set of distinct equivalence classes represented by the columns of N, and let g_0 be the equivalence class represented by b. The *group polyhedron* is given by

$$P(\mathcal{N}, g_0) = \text{conv}\left\{ t \in \mathbb{Z}^{|\mathcal{N}|}_+ : \sum_{g \in \mathcal{N}} gt(g) = g_0 \right\},$$

where equality is taken modulo B. Letting $\mathcal{G}_+ = \mathcal{G} \setminus 0$, i.e. the set of equivalence classes distinct from the lattice generated by B, the *master group polyhedron* is given by

$$P(\mathcal{G}, g_0) = \text{conv}\left\{ t \in \mathbb{Z}^{|\mathcal{G}|-1}_+ : \sum_{g \in \mathcal{G}_+} gt(g) = g_0 \right\}.$$

When A consists of a single row, the master group polyhedron is of the form

$$P(\mathcal{C}_D, r) = \operatorname{conv}\left\{ t \in \mathbb{Z}_+^{|D|-1} : \sum_{i=1}^{|D|-1} i t_i \equiv r \pmod{D} \right\}$$

and is called the *master cyclic group polyhedron.*

In [3], Gomory introduces the group polyhedron and studies its facets. In particular, he shows that one can obtain facets of the group polyhedron from facets of its corresponding master polyhedron, and that these can be used to obtain valid inequalities for P. Further, Gomory identifies the *mixed integer cut* as a facet of the master cyclic group polyhedron $P(\mathcal{C}_n, r)$ where $r \neq 0$:

$$\frac{1}{r}t_1 + \cdots + \frac{r-1}{r}t_{r-1} + t_r + \frac{n-r-1}{n-r}t_{r+1} + \cdots + \frac{1}{n-r}t_{n-1} \geq 1.$$

Indeed, by dropping the coefficients in the above inequality for elements not appearing in the group problem for a tableau row, one readily obtains the familiar Gomory mixed integer cut.

Empirically, this cut has been effective in solving integer programs [2], and shooting experiments indicate that this facet describes a large portion of the master cyclic group polyhedron [4].

We continue this investigation of the mixed integer cut. In Section 2, we characterize its extreme points and identify the adjacent facets of the master cyclic group polyhedron; in Section 3, we extend our characterization of extreme points to all integer points of the mixed integer cut; and in Section 4, we discuss mappings of the mixed integer cut under automorphisms and homomorphisms of groups and provide extensions of our results. We conclude with future research directions.

2 Facets and Extreme Points of the Mixed Integer Cut

Throughout, we consider the master cyclic group polyhedron:

$$P(\mathcal{C}_n, r) = \operatorname{conv}\left\{ t \in \mathbb{Z}_+^{n-1} : \sum_{i=1}^{n-1} i t_i \equiv r \pmod{n} \right\}.$$

We will also frequently refer to the *master knapsack polyhedron*,

$$P(K_m) = \operatorname{conv}\left\{ x \in \mathbb{Z}_+^{m} : \sum_{i=1}^{m} i x_i = m \right\}.$$

Further, we will always assume that $r > 0$ and that $n \geq 3$. By observing that the recession cone of $P(\mathcal{C}_n, r)$ is the non-negative orthant, one notes that $P(\mathcal{C}_n, r)$ is of dimension $n-1$. It is also easily observed that $P(K_m)$ is of dimension $m-1$.

By the assumption that $n \geq 3$, it follows that the non-negativity constraints are facet defining. In our discussion, these shall be referred to as the *trivial facets.*

Let (π, π_0), denote the inequality

$$\sum_{t=1}^{n-1} \pi_i t_i \geq \pi_0.$$

When speaking of valid inequalities for the master knapsack polyhedra, we shall use the same notation where entries are understood to be of appropriate dimension. Denote the mixed integer cut by $(\mu, 1)$, where

$$\mu_i = \begin{cases} \frac{i}{r} & i \leq r \\ \frac{n-i}{n-r} & i > r \end{cases}.$$

For completeness, we include the following theorem to which we have already referred:

Theorem 1 (Gomory [3]). *$(\mu, 1)$ is a facet of $P(\mathcal{C}_n, r)$.*

We consider the mixed integer cut as the polytope

$$P_{\text{MIC}}(n, r) = P(\mathcal{C}_n, r) \cap \{t : \mu t = 1\}.$$

Since $(\mu, 1)$ is a facet of $P(\mathcal{C}_n, r)$ and $P(\mathcal{C}_n, r)$ is integral, $P_{\text{MIC}}(n, r)$ is also integral. Note that a facet (π, π_0) is adjacent to $(\mu, 1)$ if and only if it is a facet of $P_{\text{MIC}}(n, r)$. We assume that $1 < r < n - 1$, since otherwise the non-trivial facets of $P_{\text{MIC}}(n, r)$ are clearly knapsack facets.

We shall now discuss the connection between $P_{\text{MIC}}(n, r)$ and the master knapsack polytopes $P(K_r)$ and $P(K_{n-r})$. The following proposition highlights an operation that we will call *extending* a knapsack solution.

Proposition 1. *If $x \in P(K_r)$, $x = (x_1, \ldots, x_r)$, then $t = (x_1, \ldots, x_r, 0, \ldots, 0)$ belongs to $P_{\text{MIC}}(n, r)$. Likewise, if $x \in P(K_{n-r})$, $x = (x_1, \ldots, x_{n-r})$, then $t = (0, \ldots, 0, x_{n-r}, \ldots, x_1)$ belongs to $P_{\text{MIC}}(n, r)$.*

Proof. For $x \in P(K_r)$, the result is trivial. So take $x \in P(K_{n-r})$. Since $P(K_{n-r})$ is convex and integral, we may assume that x is integral. Rewriting $i = n-(n-i)$ for $i = 1, \ldots, r$ and applying the assumption that x is an integral knapsack solution, the proposition follows.

In terms of facets, we shall focus on a family of facets introduced in [1]. Before stating the theorem, we note that for any non-trivial knapsack facet, by taking an appropriate linear combination with the knapsack equation, we may assume the following:

Proposition 2. *Let (ρ, ρ_0) be a non-trivial facet of $P(K_m)$. Without loss of generality we may assume that $(\rho, \rho_0) \geq 0$, $\rho_0 = \rho_m = 1$. Moreover, we may assume there exists some $i \neq m$ such that $\rho_i = 0$.*

Theorem 2 (Aráoz et. al. [1]). *Let (ρ, ρ_r) be a non-trivial facet of $P(K_r)$ such that $\rho \geq 0$, $\rho_i = 0$ for at least one i, and $\rho_r = 1$. Let*

$$\overline{\rho} = \left(\rho_1, \ldots, \rho_r = 1, \frac{n-r-1}{n-r}, \ldots, \frac{1}{n-r}\right).$$

Then there exists some $\alpha \in \mathbb{R}$ such that $(\pi, \pi_0) = (\overline{\rho} + \alpha\mu, 1 + \alpha)$ is a facet of $P(\mathcal{C}_n, r)$.

Although not explicitly stated in [1], as an easy consequence of Theorem 2 and Theorem 6 (Section 4), this operation can also be performed using non-trivial facets of $P(K_{n-r})$.

Proposition 3. *Let (ρ, ρ_{n-r}) be a non-trivial facet of $P(K_{n-r})$ such that $\rho \geq 0$, $\rho_i = 0$ for at least one i, and $\rho_{n-r} = 1$. Let*

$$\overline{\rho} = \left(\frac{1}{r}, \ldots, \frac{r-1}{r}, 1 = \rho_{n-r}, \rho_{n-r-1}, \ldots, \rho_1\right).$$

Then there exists some $\alpha \in \mathbb{R}$ such that $(\pi, \pi_0) = (\overline{\rho} + \alpha\mu, 1 + \alpha)$ is a facet of $P(\mathcal{C}_n, r)$.

In particular, given any non-trivial facet of $P(K_r)$ or $P(K_{n-r})$ we can construct a facet of $P(\mathcal{C}_n, r)$. Such facets are called *tilted knapsack facets* and the α is called the *tilting coefficient*. Details for calculating α are given in [1]. Applying Proposition 1 we arrive at the following:

Lemma 1. *The tilted knapsack facets are facets of $P_{\text{MIC}}(n, r)$.*

Proof. We argue for facets tilted from $P(K_r)$; an analogous argument proves the result for facets tilted from $P(K_{n-r})$.

Let (π, π_0) be tilted from $(\rho, 1)$, and let $\overline{\rho}$ be as described in Theorem 2 and α be the corresponding tilting coefficient. Since $(\rho, 1)$ is a facet of $P(K_r)$, there exist $r - 1$ affinely independent extreme points $x^1, \ldots, x^{r-1}$ satisfying $(\rho, 1)$ at equality. As described in Proposition 1, these points may be extended to points $t^1, \ldots, t^{r-1} \in P_{\text{MIC}}(n, r)$, and clearly this preserves affine independence. Moreover, for $i = 1, \ldots, r-1$, $\mu t^i = 1$ and $\overline{\rho} t^i = \rho x = 1$, thus

$$\pi t^i = (\overline{\rho} + \alpha\mu)t^i = \overline{\rho} t^i + \alpha \cdot \mu t^i = 1 + \alpha = \pi_r.$$

Now consider $n-r$ affinely independent extreme points $y^1, \ldots, y^{n-r}$ of $P(K_{n-r})$, and again as in Proposition 1, extend them to points $s^1, \ldots, s^{n-r} \in P_{\text{MIC}}(n, r)$.

$$\pi s^i = (\overline{\rho} + \alpha\mu)s^i = \overline{\rho} s^i + \alpha \cdot \mu s^i = 1 + \alpha = \pi_r.$$

It is easily seen that $\{t^1, \ldots, t^{r-1}\} \cap \{s^1, \ldots, s^{n-r}\} = e_r$. Therefore we have produced $n - 2$ affinely independent points, proving the claim.

Consider a tilted knapsack facet (π, π_0) arising from the facet $(\rho, 1)$ of $P(K_r)$ with tilting coefficient α. Letting μ' denote first r coefficients of μ, the same facet of $P(K_r)$ is described by $(\gamma, 0) = (\rho, 1) - (\mu', 1)$. In particular letting,

$$(\bar{\gamma}, 0) = (\gamma_1, \ldots, \gamma_r = 0, 0, \ldots, 0),$$

it follows that $(\pi, \pi_0) = (\bar{\gamma}, 0) + (1+\alpha)(\mu, 1)$. The same applies to tilted knapsack facets arising from $P(K_{n-r})$.

Therefore we will think of tilted knapsack facets as arising from facets of the form $(\rho, 0)$, and by subtracting off the mixed integer cut we think of tilted knapsack facets in the form $(\bar{\rho}, 0)$.

We now prove our main result.

Theorem 3. *The convex hull of $P_{\mathrm{MIC}}(n,r)$ is given by the tilted knapsack facets and the non-negativity constraints.*

Proof. For convenience, say that $P(K_r)$ has non-trivial facets $(\rho^1, 0), \ldots, (\rho^M, 0)$ and that $P(K_{n-r})$ has non-trivial facets $(\gamma^1, 0), \ldots, (\gamma^N, 0)$. Let $(\bar{\rho}^i, 0)$ and $(\bar{\gamma}^i, 0)$ denote the tilted knapsack facets from $(\rho^i, 0)$ and $(\gamma^i, 0)$ respectively.

We shall show that the system

$$\begin{array}{lll} \min & c \cdot t & \\ \text{s.t.} & \mu \cdot t & = 1 \\ & \bar{\rho}^i \cdot t & \geq 0 \; i = 1, \ldots M \\ & \bar{\gamma}^i \cdot t & \geq 0 \; i = 1, \ldots N \\ & t & \geq 0 \end{array} \tag{2}$$

attains an integer optimum that belongs to $P_{\mathrm{MIC}}(n,r)$ for every c.

Let $c' = (c_1, \ldots, c_r)$ and $c'' = (c_{n-1}, \ldots, c_r)$, $\mu' = \left(\frac{1}{r}, \ldots, \frac{r-1}{r}, 1\right)$, and $\mu'' = \left(\frac{1}{n-r}, \ldots, \frac{n-r-1}{n-r}, 1\right)$. Consider the systems

$$\begin{array}{lll} \min & c' \cdot x' & \\ \text{s.t.} & \mu' \cdot x' & = 1 \\ & \rho^i \cdot x' & \geq 0 \; i = 1, \ldots M \\ & x' & \geq 0 \end{array} \tag{3}$$

and

$$\begin{array}{lll} \min & c'' \cdot x'' & \\ \text{s.t.} & \mu'' \cdot x'' & = 1 \\ & \gamma^i \cdot x'' & \geq 0 \; i = 1, \ldots N \\ & x'' & \geq 0 \end{array} \tag{4}$$

representing $P(K_r)$ and $P(K_{n-r})$ respectively. Since both systems are integral, the minima are obtained at integer extreme points $\widehat{x'}$ and $\widehat{x''}$ respectively. Now let t^* be obtained by extending the solution achieving the smaller objective value to a feasible point of $P_{\mathrm{MIC}}(n,r)$. Indeed this t^* is feasible and integral; it remains to show that it is optimal.

We now consider the duals. The dual of (3) is given by

$$\max_{\lambda_1, \alpha'} (\lambda_1 : \lambda_1 \mu' + \alpha'_1 \rho^1 + \cdots + \alpha'_M \rho^M \leq c', \alpha' \geq 0), \tag{5}$$

and the dual of (4) is given by

$$\max_{\lambda_2, \beta'} (\lambda_2 : \lambda_2 \mu'' + \beta'_1 \gamma^1 + \cdots + \beta'_N \gamma^N \leq c'', \beta' \geq 0). \tag{6}$$

Lastly the dual of (2) is given by

$$\max_{\lambda,\alpha,\beta} \left(\lambda : \begin{array}{l} \lambda\mu + \alpha_1 \bar{\rho}^1 + \cdots + \alpha_M \bar{\rho}^M + \beta_1 \bar{\gamma}^1 + \cdots + \beta_N \bar{\gamma}^N \le c \\ \alpha, \beta \ge 0 \end{array} \right). \qquad (7)$$

Let $(\widehat{\lambda_1}, \widehat{\alpha'})$ and $(\widehat{\lambda_2}, \widehat{\beta'})$ attain the maxima in (5) and (6) respectively. Setting $\widehat{\lambda} = \min(\widehat{\lambda_1}, \widehat{\lambda_2})$, it easily follows from the zero pattern of (2) and non-negativity of μ that $(\widehat{\lambda}, \widehat{\alpha'}, \widehat{\beta'})$ is feasible to (7). Moreover $\widehat{\lambda} = c \cdot t^*$, proving optimality.

Further observe that $P_{\text{MIC}}(n, r)$ is pointed, and so from this same proof we get the following characterization of extreme points:

Theorem 4. *A point t is an extreme point of $P_{\text{MIC}}(n, r)$ if and only if it can be obtained by extending an extreme point of $P(K_r)$ or $P(K_{n-r})$.*

3 Integer Points of the Mixed Integer Cut

In this section we highlight a noteworthy extension of Theorem 4 to all integer points of $P_{\text{MIC}}(n, r)$.

Theorem 5. *$t \in P_{\text{MIC}}(n, r) \cap \mathbb{Z}^{n-1}$ if and only if t can be obtained by extending an integer solution of $P(K_r)$ or $P(K_{n-r})$.*

Proof. If $t = e_r$ the claim is obvious. So we suppose that $t_r = 0$. We shall show that if $t \in P_{\text{MIC}}(n, r) \cap \mathbb{Z}^{n-1}$, $t_r = 0$, then either (I) $(t_1, \ldots, t_r) > 0$ or (II) $(t_r, \ldots, t_{n-1}) > 0$ but not both.

Since $t \in P(\mathcal{C}_n, r)$,

$$t_1 + \cdots + (r-1)t_{r-1} + rt_r + (r+1)t_{r+1} + \cdots + (n-1)t_{n-1} \equiv r \pmod{n}.$$

Thus there exists some $\beta \in \mathbb{Z}$ such that

$$t_1 + \cdots + (r-1)t_{r-1} + rt_r + (r+1)t_{r+1} + \cdots + (n-1)t_{n-1} = r + \beta n$$

, and since $r > 0$, we may rewrite this

$$\frac{1}{r}t_1 + \cdots + \frac{r-1}{r}t_{r-1} + t_r + \frac{r+1}{r}t_{r+1} + \cdots + \frac{n-1}{r}t_{n-1} = 1 + \beta\frac{n}{r}. \qquad (8)$$

Now, $t \in P_{\text{MIC}}(n, r)$ therefore

$$\frac{1}{r}t_1 + \cdots + \frac{r-1}{r}t_{r-1} + t_r + \frac{n-r-1}{n-r}t_{r+1} + \cdots + \frac{1}{n-r}t_{n-1} = 1$$

or

$$\frac{1}{r}t_1 + \cdots + \frac{r-1}{r}t_{r-1} + t_r = 1 - \left[\frac{n-r-1}{n-r}t_{r+1} + \cdots + \frac{1}{n-r}t_{n-1}\right]. \qquad (9)$$

Substituting (9) into (8), we obtain

$$\begin{aligned}
& 1-\left[\tfrac{n-r-1}{n-r}t_{r+1}+\cdots+\tfrac{1}{n-r}t_{n-1}\right]+\tfrac{r+1}{r}t_{r+1}+\cdots+\tfrac{n-1}{r}t_{n-1}=1+\beta\tfrac{n}{r}\\
\Rightarrow\quad & \left(\tfrac{r+1}{r}-\tfrac{n-r-1}{n-r}\right)t_{r+1}+\cdots+\left(\tfrac{n-1}{r}-\tfrac{1}{n-r}\right)t_{n-1}=\beta\tfrac{n}{r}\\
\Rightarrow\quad & \tfrac{n}{r}\cdot\tfrac{1}{n-r}t_{r+1}+\cdots+\tfrac{n}{r}\cdot\tfrac{n-r-1}{n-r}t_{n-1}=\beta\tfrac{n}{r}\\
\Rightarrow\quad & \tfrac{1}{n-r}t_{r+1}+\cdots+\tfrac{n-r-1}{n-r}t_{n-1}=\beta\\
\Rightarrow\quad & \underbrace{[t_{r+1}+\cdots+t_{n-1}]}_{(*)}-\underbrace{\left[\frac{n-r-1}{n-r}t_{r+1}+\cdots+\frac{1}{n-r}t_{n-1}\right]}_{(**)}=\beta.
\end{aligned}$$

Because t was assumed to be integral $(*)$ is necessarily integral. Suppose conversely that both (I) and (II) hold; by the assumption that $t_r = 0$ and because t is necessarily non-negative, the relation

$$\frac{n-r-1}{n-r}t_{r+1}+\cdots+\frac{1}{n-r}t_{n-1}=1-\left[\frac{1}{r}t_1+\cdots+\frac{r-1}{r}t_{r-1}+t_r\right]$$

implies that $(**)$ must be fractional. But this contradicts that β is integral. Therefore (I) and (II) cannot simultaneously hold.

4 Extensions under Automorphisms and Homomorphisms

Here we review some general properties of facets of the master group polyhedra and discuss extensions of our previous results. Throughout, some basic knowledge of algebra is assumed.

Let $\mathcal{G}$ be an abelian group with identity 0, $\mathcal{G}_+ = \mathcal{G} \setminus 0$, and $g_0 \in \mathcal{G}_+$. The master group polyhedron, $P(\mathcal{G}, g_0)$ is defined by

$$P(\mathcal{G},g_0)=\text{conv}\left\{t\in\mathbb{Z}_+^{|\mathcal{G}|-1}:\sum_{g\in\mathcal{G}_+}gt(g)=g_0\right\}.$$

Because $|\mathcal{G}|g = 0$ for all $g \in \mathcal{G}_+$, the recession cone of $P(\mathcal{G}, g_0)$ is the non-negative orthant, and since $P(\mathcal{G}, g_0)$ is nonempty, the polyhedron is of full dimension.

As before, let (π, π_0) denote the inequality

$$\sum_{g\in\mathcal{G}_+}\pi(g)t(g)\geq\pi_0.$$

If $|\mathcal{G}| - 1 \geq 2$, then the inequality $t(g) \geq 0$ is facet defining for all $g \in \mathcal{G}_+$, and it is easily verified that these are the only facets with $\pi_0 = 0$. Likewise, we call these the trivial facets of $P(\mathcal{G}_+, g_0)$.

4.1 Automorphisms

We are able to use automorphisms of $\mathcal{G}$ to obtain facets of $P(\mathcal{G}, g_0)$ from other master group polyhedra. Throughout, let ϕ be an automorphism of $\mathcal{G}$.

Theorem 6 (Gomory [3], Theorem 14). *If (π, π_0) is a facet of $P(\mathcal{G}, g_0)$, with components, $\pi(g)$, then (π', π_0) with components $\pi'(g) = \pi(\phi^{-1}(g))$ is a facet of $P(\mathcal{G}, \phi(g_0))$.*

Similarly if t satisfies (π, π_0) at equality, then t' with components $t'(g) = t(\phi^{-1}(g))$ satisfies (π', π_0) at equality, and since ϕ is an automorphism of $\mathcal{G}$, t' necessarily satisfies the group equation for $P(\mathcal{G}, \phi(g_0))$. As an obvious consequence, a point t lies on the facet (π, π_0) of $P(\mathcal{G}, g_0)$ if and only if the corresponding point t' lies on the facet (π', π_0) of $P(\mathcal{G}, \phi(g_0))$. Hence we obtain the following:

Proposition 4. *If (π, π_0) and (γ, γ_0) are facets of $P(\mathcal{G}, g_0)$, then (γ, γ_0) is adjacent to (π, π_0) if and only if (π', π_0) and (γ', γ_0) are adjacent facets of $P(\mathcal{G}, \phi(g_0))$, where $\pi'(g) = \pi(\phi^{-1}(g))$ and $\gamma'(g) = \gamma(\phi^{-1}(g))$.*

Proof. Since (γ, γ_0) and (π, π_0) define adjacent facets, there exist affinely independent points $t^1, \ldots, t^{(|\mathcal{G}|-2)}$ satisfying both at equality. By the previous remarks, we may define points $(t^1)'$, …, $(t^{(|\mathcal{G}|-2)})'$ satisfying both (π', π_0) and (γ', γ_0) at equality. Since these are all defined by the same permutation of the indices of $t^1, \ldots, t^{(|\mathcal{G}|-2)}$, affine independence is preserved.

Now consider the case when $\mathcal{G} = \mathcal{C}_n$, $g_0 = r$. Let $(\mu', 1)$ be obtained by applying ϕ to $(\mu, 1)$. Our previous results extend in the following sense:

Theorem 7. *The non-trivial facets of $P(\mathcal{C}_n, \phi(r))$ adjacent to $(\mu', 1)$ are exactly those obtained by applying ϕ to tilted knapsack facets.*

Theorem 8. *An integer point $t \in P(\mathcal{C}_n, \phi(r))$ satisfies $(\mu', 1)$ at equality if and only if t is obtained by extending a knapsack solution of $P(K_r)$ or $P(K_{n-r})$ and applying ϕ to the indices of t.*

4.2 Homomorphisms

Additionally one can obtain facets from homomorphisms of $\mathcal{G}$ by *homomorphic lifting.* Let $\psi : \mathcal{G} \to \mathcal{H}$ be a homomorphism with kernel $\mathcal{K}$ such that $g_0 \notin \mathcal{K}$. For convenience let $h_0 = \psi(g_0)$.

Theorem 9 (Gomory [3], Theorem 19). *Let (π, π_0) be a non-trivial facet of $P(\mathcal{H}, h_0)$. Then (π', π_0) is a facet of $P(\mathcal{G}, g_0)$ where $\pi'(g) = \pi(\psi(g))$ for all $g \in \mathcal{G} \setminus \mathcal{K}$, and $\pi'(k) = 0$ for all $k \in \mathcal{K}$.*

Unlike automorphisms, it is not clear that homomorphic lifting preserves the adjacency of facets. We show next that it in fact does preserve adjacency.

First we prove the following useful proposition:

Proposition 5. *Let (π, π_0) and (γ, γ_0) be adjacent non-trivial facets in $P(\mathcal{H}, h_0)$ $(h_0 \neq 0)$. Then the affine subspace*

$$T = P(\mathcal{H}, h_0) \cap \{t \in \mathbb{R}^{|\mathcal{H}|-1} : \pi t = \pi_0, \gamma t = \gamma_0\}$$

does not lie in the hyperplane $H(h) = \{t \in \mathbb{R}^{|\mathcal{H}|-1} : t(h) = 0\}$ for any $h \in \mathcal{H}_+$.

Proof. In [3], Gomory shows that every non-trivial facet (π, π_0) of $P(\mathcal{H}, h_0)$ satisfies $\pi(h) + \pi(h_0 - h) = \pi(h_0) = \pi_0$. In particular for all $h \in \mathcal{H}_+ \setminus h_0$, the point $t = e_h + e_{h_0-h}$ belongs to T, and has $t(h) > 0$. Similarly, the point $t = e_{h_0}$ belongs to T and has $t(h_0) > 0$.

Using this proposition we obtain the following:

Lemma 2. *Let (π, π_0) and (γ, γ_0) be adjacent non-trivial facets of $P(\mathcal{H}, h_0)$, and let (π', π_0) and (γ', γ_0) be facets of $P(\mathcal{G}, g_0)$ obtained by homomorphic lifting using the homomorphism ψ. Then (π', π_0) and (γ', γ_0) are adjacent.*

Proof. Let $\mathcal{K} = \ker(\psi)$. Let φ be a function selecting one element from each coset of $\mathcal{G}/\mathcal{K}$ distinct from $\mathcal{K}$, and let $\varphi(\mathcal{H})$ denote the set of coset representatives chosen by φ.

Since we are assuming that (π', π_0) and (γ', γ_0) are obtained by homomorphic lifting, $h_0 \neq 0$. Since (π, π_0) and (γ, γ_0) are adjacent there exist affinely independent points $t^1, \ldots, t^{(|\mathcal{H}|-2)}$ in $P(\mathcal{H}, h_0)$ satisfying (π, π_0) and (γ, γ_0) at equality. By Proposition 5, for all $h \in \mathcal{H}_+$, there exists an $i \in \{1, \ldots, |\mathcal{H}| - 2\}$ such that $t^i(h) > 0$.

Using these points, we will construct $|\mathcal{G}| - 2$ affinely independent points belonging to $P(\mathcal{G}, g_0)$ that satisfy both (π', π_0) and (γ', γ_0) at equality. We proceed as follows:

1. Set $\mathcal{N} = \mathcal{H}_+$
2. For $i = \{1, \ldots, |\mathcal{H}| - 2\}$
   - Set $\mathcal{N}(i) = \{h \in \mathcal{H}_+ : t^i(h) > 0\} \cap \mathcal{N}$
   - Define s^i as follows:

$$\begin{array}{ll} s^i(\varphi(h)) = t^i(h), & \forall h \in \mathcal{H} \\ s^i(g) = 0, & g \in \mathcal{G} \setminus (\mathcal{K} \cup \varphi(\mathcal{H})) \\ s^i(k) = 1, & k = g_0 - \sum_{g \in \mathcal{G}_+ \setminus \mathcal{K}} s^i(g) \cdot g \\ s^i(k) = 0, & k \neq g_0 - \sum_{g \in \mathcal{G}_+ \setminus \mathcal{K}} s^i(g) \cdot g \end{array}$$

   - For each $h' \in \mathcal{N}(i)$, $k' \in \mathcal{K}_+$, define the point $s^i_{k',h'}$ as follows:

$$\begin{array}{ll} s^i_{k',h'}(\varphi(h')) + k') = t^i(h') & \\ s^i_{k',h'}(\varphi(h')) = 0 & \\ s^i_{k',h'}(g) = s^i(g), & g \in \mathcal{G}_+ \setminus \mathcal{K}, g \neq \varphi(h'), g \neq \varphi(h') + k' \\ s^i_{k',h'}(k) = 1, & k = g_0 - \sum_{g \in \mathcal{G}_+ \setminus \mathcal{K}} s^i_{k',h'}(g) \cdot g \\ s^i_{k',h'}(k) = 0, & k \neq g_0 - \sum_{g \in \mathcal{G}_+ \setminus \mathcal{K}} s^i_{k',h'}(g) \cdot g \end{array}$$

   - Set $\mathcal{N} = \mathcal{N} \setminus \mathcal{N}(i)$
3. For each $k \in \mathcal{K}_+$, define s_k by $s_k = s^1 + |\mathcal{G}| e_k$

By construction these points satisfy (π', π_0) and (γ', γ_0) at equality. It remains to verify that the above procedure indeed produces $|\mathcal{G}| - 2$ affinely independent points belonging to $P(\mathcal{G}, g_0)$.

First we show that the above points belong to $P(\mathcal{G}, g_0)$. Let s be one of the above points. Then

$$\psi \left(\sum_{g \in \mathcal{G}_+ \setminus \mathcal{K}} g s(g) \right) = \sum_{g \in \mathcal{G}_+ \setminus \mathcal{K}} \psi(g) s(g) = \sum_{h \in \mathcal{H}_+} h \cdot \left(\sum_{g \in \mathcal{G}_+ : \psi(g) = h} s(g) \right) = h_0,$$

where the first equality comes from the fact that ψ is a homomorphism and the second equality follows by how we defined the above points. Therefore,

$$\sum_{g \in \mathcal{G}_+ \setminus \mathcal{K}} g s(g) \in g_0 \mathcal{K},$$

and by construction,

$$\sum_{k \in \mathcal{K}} k s(k) = g_0 - \left(\sum_{g \in \mathcal{G}_+ \setminus \mathcal{K}} g s(g) \right).$$

Thus $s \in P(\mathcal{G}, g_0)$.

Note that we have the $|\mathcal{H}| - 2$ points $s^1, \ldots, s^{|\mathcal{H}|-2}$. By Proposition 5, we obtain $(|\mathcal{H}| - 1)(|\mathcal{K}| - 1)$ points of the form $s_{k,h}$ for $k \in \mathcal{K}_+$ and $h \in \mathcal{H}_+$, and lastly, we obtain $|K| - 1$ points, s_k for $k \in \mathcal{K}_+$. Using the identity $|\mathcal{G}| = |\mathcal{K}||\mathcal{H}|$, it immediately follows that we have $|\mathcal{G}| - 2$ points.

The affine independence of these points is easily verified. By constructing a matrix for which the first $|\mathcal{K}| - 1$ columns correspond to $\mathcal{K}$, the next $|\mathcal{H}| - 1$ columns corresponding to $\varphi(\mathcal{H})$, and the remaining columns are arranged in blocks by the cosets, it is readily observed by letting each row be one of the above points and using the affine independence of $t^1, \ldots, t^{|\mathcal{H}|-2}$ that the newly defined points are affinely independent.

Given a point $s \in P(\mathcal{G}, g_0)$ that satisfies the lifted facets at equality, we can obtain a point $t \in P(\mathcal{H}, h_0)$ that satisfies (π, π_0) and (γ, γ_0) at equality under the mapping $t(h) = \sum_{g \in G : \psi(g) = h} s(g)$. By a fairly routine exercise in linear algebra, one can use this to verify that s is in the affine hull of the points described above. Hence we obtain the following theorem:

Theorem 10. *Let (π, π_0) and (γ, γ_0) be non-trivial facets of $P(\mathcal{H}, h_0)$, and let (π', π_0) and (γ', γ_0) be facets of $P(\mathcal{G}, g_0)$ obtained by homomorphic lifting using the homomorphism ψ. Then (π', π_0) and (γ', γ_0) are adjacent if and only if (π, π_0) and (γ, γ_0) are adjacent.*

Now consider $\mathcal{G} = \mathcal{C}_{n'}$, $g_0 = r'$, a homomorphism $\psi : \mathcal{C}_{n'} \to \mathcal{C}_n$, $\psi(r') = r \neq 0$, and let $(\mu', 1)$ be obtained by applying homomorphic lifting to $(\mu, 1)$. Similarly by applying Theorem 10, we know that the only lifted facets under ψ that are adjacent to $(\mu', 1)$ come from tilted knapsack facets. Stated precisely:

Theorem 11. *Let (π', π_0) be obtained by homomorphic lifting using ψ applied to (π, π_0). Then (π', π_0) is adjacent to $(\mu', 1)$ if and only if (π, π_0) is a tilted knapsack facet.*

Moreover, for the integer points we obtain the following:

Theorem 12. *If an integer point $s \in P(\mathcal{C}_{n'}, r')$ satisfies $(\mu', 1)$ at equality. Then the point t defined by the mapping*

$$t_i = \sum_{j:\psi(j)=i} s_j$$

is an integer point of $P(\mathcal{C}_n, r)$ and satisfies $(\mu, 1)$ at equality. In particular it is obtained from extending a knapsack solution of $P(K_r)$ or $P(K_{n-r})$.

5 Future Work and Conclusions

Several questions remain for both the group polyhedron and knapsack polytope. One worthy avenue of research is to expand the existing library of knapsack facets, which in turn will provide even more information about the mixed integer cut.

Another interesting problem is to obtain non-trivial necessary and sufficient conditions to describe the extreme points of the master knapsack polytope and the master group polyhedron. A natural idea was considered for the group polyhedron in terms of irreducibility. This condition is necessary for all vertices, but insufficient. One might hope that this condition becomes sufficient for the master knapsack polytope; however, it again fails.

Lastly, a closer inspection will reveal that in homomorphic lifting we gain no information about the kernel of our homomorphism. If we consider the lifted mixed integer cut as a polyhedron, it is no longer sufficient to characterize its extreme points in terms of two related knapsacks. Similarly, it is easy to see that lifted tilted knapsack facets are not the only adjacent non-trivial facets of the lifted mixed integer cut. One might address whether there exists a family of facets that when added to the lifted tilted knapsack facets completely characterizes the adjacent facets of the lifted mixed integer cut.

References

1. Aráoz, J., Evans, L., Gomory, R.E., Johnson, E.L.: Cyclic group and knapsack facets. Math. Program. 96(2), 377–408 (2003)
2. Dash, S., Günlük, O.: On the strength of gomory mixed-integer cuts as group cuts. Math. Program. 115(2), 387–407 (2008)
3. Gomory, R.E.: Some polyhedra related to combinatorial problems. Linear Algebra and Its Applications (2), 451–558 (1969)
4. Gomory, R.E., Johnson, E.L., Evans, L.: Corner polyhedra and their connection with cutting planes. Math. Program. 96(2), 321–339 (2003)

Symmetry Matters for the Sizes of Extended Formulations

Volker Kaibel, Kanstantsin Pashkovich, and Dirk O. Theis

Otto-von-Guericke-Universität Magdeburg, Institut für Mathematische Optimierung
Universitätsplatz 2, 39108 Magdeburg, Germany
{kaibel,pashkovich,theis}@ovgu.de

Abstract. In 1991, Yannakakis [17] proved that no symmetric extended formulation for the matching polytope of the complete graph K_n with n nodes has a number of variables and constraints that is bounded subexponentially in n. Here, symmetric means that the formulation remains invariant under all permutations of the nodes of K_n. It was also conjectured in [17] that "asymmetry does not help much," but no corresponding result for general extended formulations has been found so far. In this paper we show that for the polytopes associated with the matchings in K_n with $\lfloor \log n \rfloor$ edges there are non-symmetric extended formulations of polynomial size, while nevertheless no symmetric extended formulation of polynomial size exists. We furthermore prove similar statements for the polytopes associated with cycles of length $\lfloor \log n \rfloor$. Thus, with respect to the question for smallest possible extended formulations, in general symmetry requirements may matter a lot.

1 Introduction

Linear Programming techniques have proven to be extremely fruitful for combinatorial optimization problems with respect to both structural analysis and the design of algorithms. In this context, the paradigm is to represent the problem by a polytope $P \subseteq \mathbb{R}^m$ whose vertices correspond to the feasible solutions of the problem in such a way that the objective function can be expressed by a linear functional $x \mapsto \langle c, x\rangle$ on $\mathbb{R}^m$ (with some $c \in \mathbb{R}^m$). If one succeeds in finding a description of P by means of linear constraints, then algorithms as well as structural results from Linear Programming can be exploited. In many cases, however, the polytope P has exponentially (in m) many facets, thus P can only be described by exponentially many inequalities. Also it may be that the inequalities needed to describe P are too complicated to be identified.

In some of these cases one may find an *extended formulation for P*, i.e., a (preferably small and simple) description by linear constraints of another polyhedron $Q \subseteq \mathbb{R}^d$ in some higher dimensional space that projects to P via some (simple) linear map $p : \mathbb{R}^d \to \mathbb{R}^m$ with $p(y) = Ty$ for all $y \in \mathbb{R}^d$ (and some matrix $T \in \mathbb{R}^{m\times d}$). Indeed, if $p^\star : \mathbb{R}^m \to \mathbb{R}^d$ with $p^\star(x) = T^{\mathrm{t}}x$ for all $x \in \mathbb{R}^m$ denotes the linear map that is adjoint to p (with respect to the standard bases), then we have $\max\{\langle c, x\rangle : x \in P\} = \max\{\langle p^\star(c), y\rangle : y \in Q\}$.

As for an example, let us consider the spanning tree polytope $\mathrm{P}_{\mathrm{spt}}(n) = \mathrm{conv}\{\chi(T) \in \{0,1\}^{E_n} : T \subseteq E_n \text{ spanning tree of } K_n\}$, where $K_n = ([n], E_n)$ denotes the complete graph with node set $[n] = \{1, \ldots, n\}$ and edge set $E_n = \{\{v,w\} : v,w \in [n], v \neq w\}$, and $\chi(A) \in \{0,1\}^B$ is the *characteristic vector* of the subset $A \subseteq B$ of B, i.e., for all $b \in B$, we have $\chi(A)_b = 1$ if and only if $b \in A$. Thus, $\mathrm{P}_{\mathrm{spt}}(n)$ is the polytope associated with the bases of the graphical matroid of K_n, and hence (see [7]), it consists of all $x \in \mathbb{R}_+^{E_n}$ satisfying $x(E_n) = n-1$ and $x(E_n(S)) \leq |S| - 1$ for all $\subseteq [n]$ with $2 \leq |S| \leq n-1$, where $\mathbb{R}_+^E$ is the nonnegative orthant of $\mathbb{R}^E$, we denote by $E_n(S)$ the subset of all edges with both nodes in S, and $x(F) = \sum_{e \in F} x_e$ for $F \subseteq E_n$. This linear description of $\mathrm{P}_{\mathrm{spt}}(n)$ has an exponential (in n) number of constraints, and as all the inequalities define pairwise disjoint facets, none of them is redundant.

The following much smaller exended formulation for $\mathrm{P}_{\mathrm{spt}}(n)$ (with $O(n^3)$ variables and constraints) appears in [5] (and a similar one in [17], who attributes it to [13]). Let us introduce additional 0/1-variables $z_{e,v,u}$ for all $e \in E_n$, $v \in e$, and $u \in [n] \setminus e$. While each spanning tree $T \subseteq E_n$ is represented by its characteristic vector $x^{(T)} = \chi(T)$ in $\mathrm{P}_{\mathrm{spt}}(n)$, in the extended formulation it will be represented by the vector $y^{(T)} = (x^{(T)}, z^{(T)})$ with $z^{(T)}_{e,v,u} = 1$ (for $e \in E_n$, $v \in e$, $u \in [n] \setminus e$) if and only if $e \in T$ and u is contained in the component of v in $T \setminus e$. The polyhedron $Q_{\mathrm{spt}}(n) \subseteq \mathbb{R}^d$ defined by the nonnegativity constraints $x \geq \mathbf{0}$, $z \geq \mathbf{0}$, the equations $x(E_n) = n-1$, $x_{\{v,w\}} - z_{\{v,w\},v,u} - z_{\{v,w\},w,u} = 0$ for all pairwise distinct $v, w, u \in [n]$, as well as $x_{\{v,w\}} + \sum_{u \in [n] \setminus \{v,w\}} z_{\{v,u\},u,w} = 1$ for all distinct $v, w \in [n]$, satisfies $p(Q_{\mathrm{spt}}(n)) = \mathrm{P}_{\mathrm{spt}}(n)$, where $p : \mathbb{R}^d \to \mathbb{R}^E$ is the orthogonal projection onto the x-variables.

For many other polytopes (with exponentially many facets) associated with polynomial time solvable combinatorial optimization problems polynomially sized extended formulations can be constructed as well (see, e.g., the recent survey [5]). Probably the most prominent problem in this class for which, however, no such small formulation is known, is the matching problem. In fact, Yannakakis [17] proved that no *symmetric* polynomially sized extended formulation of the matching polytope exists.

Here, *symmetric* refers to the symmetric group $\mathfrak{S}(n)$ of all permutations $\pi : [n] \to [n]$ of the node set $[n]$ of K_n acting on E_n via $\pi.\{v,w\} = \{\pi(v), \pi(w)\}$ for all $\pi \in \mathfrak{S}(n)$ and $\{v,w\} \in E_n$. Clearly, this action of $\mathfrak{S}(n)$ on E_n induces an action on the set of all subsets of E_n. For instance, this yields an action on the spanning trees of K_n, and thus, on the vertices of $\mathrm{P}_{\mathrm{spt}}(n)$. The extended formulation of $\mathrm{P}_{\mathrm{spt}}(n)$ discussed above is *symmetric* in the sense that, for every $\pi \in \mathfrak{S}(n)$, replacing all indices associated with edges $e \in E_n$ and nodes $v \in [n]$ by $\pi.e$ and $\pi.v$, respectively, does not change the set of constraints in the formulation. Phrased informally, all subsets of nodes of K_n of equal cardinality play the same role in the formulation. For a general definition of symmetric extended formulations see Section 2.

In order to describe the main results of Yannakakis paper [17] and the contributions of the present paper, let us denote by $\mathcal{M}^\ell(n) = \{M \subseteq E_n : M \text{ matching in } K_n, |M| = \ell\}$ the set of all matchings of size ℓ (a matching being a subset of edges no two of which share a node), and by $\mathrm{P}^{\ell}_{\mathrm{match}}(n) = \mathrm{conv}\{\chi(M) \in \{0,1\}^{E_n} : M \in \mathcal{M}^{\ell}(n)\}$ the associated polytope. According to Edmonds [6] the *perfect matching polytope* $\mathrm{P}^{n/2}_{\mathrm{match}}(n)$ (for even n) is described by

$$\mathrm{P}^{n/2}_{\mathrm{match}}(n) = \{x \in \mathbb{R}^{E_n}_{+} : x(\delta(v)) = 1 \text{ for all } v \in [n], \\ x(E(S)) \leq (|S|-1)/2 \text{ for all } S \subseteq [n], 3 \leq |S| \text{ odd}\} \quad (1)$$

(with $\delta(v) = \{e \in E_n : v \in e\}$). Yannakakis [17, Thm.1 and its proof] shows that there is a constant $C > 0$ such that, for every extended formulation for $\mathrm{P}^{n/2}_{\mathrm{match}}(n)$ (with n even) that is symmetric in the sense above, the number of variables and constraints is at least $C \cdot \binom{n}{\lfloor n/4 \rfloor} = 2^{\Omega(n)}$. This in particular implies that there is no polynomial size symmetric extended formulation for the *matching polytope* of K_n (the convex hulls of characteristic vectors of *all* matchings in K_n), of which the *perfect* matching polytope is a face.

Yannakakis [17] also obtains a similar (maybe less surprising) result on traveling salesman polytopes. Denoting the set of all (simple) cycles of length ℓ in K_n by $\mathcal{C}^{\ell}(n) = \{C \subseteq E_n : C \text{ cycle in } K_n, |C| = \ell\}$, and the associated polytopes by $\mathrm{P}^{\ell}_{\mathrm{cycl}}(n) = \mathrm{conv}\{\chi(C) \in \{0,1\}^{E_n} : C \in \mathcal{C}^{\ell}(n)\}$, the *traveling salesman polytope* is $\mathrm{P}^{n}_{\mathrm{cycl}}(n)$. Identifying $\mathrm{P}^{n/2}_{\mathrm{match}}(n)$ (for even n) with a suitable face of $\mathrm{P}^{3n}_{\mathrm{cycl}}(3n)$, Yannakakis concludes that all symmetric extended formulations for $\mathrm{P}^{n}_{\mathrm{cycl}}(n)$ have size at least $2^{\Omega(n)}$ as well [17, Thm. 2 and its proof].

Yannakakis' results in a fascinating way illuminate the borders of our principal abilities to express combinatorial optimization problems like the matching or the traveling salesman problem by means of linear constraints. However, they only refer to linear descriptions that respect the inherent symmetries in the problems. In fact, the second open problem mentioned in the concluding section of [17] is described as follows: "We do not think that asymmetry helps much. Thus, prove that the matching and TSP polytopes cannot be expressed by polynomial size LP's without the asymmetry assumption."

The contribution of our paper is to show that, in contrast to the assumption expressed in the quotation above, asymmetry can help much, or, phrased differently, that symmetry requirements on extended formulations indeed can matter significantly with respect to the minimal sizes of extended formulations. Our main results are that both $\mathrm{P}^{\lfloor \log n \rfloor}_{\mathrm{match}}(n)$ and $\mathrm{P}^{\lfloor \log n \rfloor}_{\mathrm{cycl}}(n)$ do not admit symmetric extended formulations of polynomial size, while they have non-symmetric extended formulations of polynomial size (see Cor. 1 and 2 for matchings, as well as Cor. 3 and 4 for cycles). The corresponding theorems from which these corollaries are derived provide some more general and more precise results for $\mathrm{P}^{\ell}_{\mathrm{match}}(n)$ and $\mathrm{P}^{\ell}_{\mathrm{cycl}}(n)$. In order to establish the lower bounds for symmetric extensions, we generalize the techniques developed by Yannakakis [17]. The constructions of the compact non-symmetric extended formulations rely on small families of perfect hash functions [1,8,15].

The paper is organized as follows. In Section 2, we provide definitions of extensions, extended formulations, their sizes, the crucial notion of a section of an extension, and we give some auxilliary results. In Section 3, we present Yannakakis' method to derive lower bounds on the sizes of symmetric extended formulations for perfect matching polytopes in a general setting, which we then exploit in Section 4 in order to derive lower bounds on the sizes of symmetric extended formulations for the polytopes $\mathrm{P}^{\ell}_{\mathrm{match}}(n)$ associated with cardinality restricted matchings. In Section 5, we describe our non-symmetric extended formultions for these polytopes. Finally, in Section 6 we present the results on $\mathrm{P}^{\ell}_{\mathrm{cycl}}(n)$. Some remarks conclude the paper in Section 7.

2 Extended Formulations, Extensions, and Symmetry

An *extension* of a polytope $P \subseteq \mathbb{R}^m$ is a polyhedron $Q \subseteq \mathbb{R}^d$ together with a projection (i.e., a linear map) $p : \mathbb{R}^d \to \mathbb{R}^m$ with $p(Q) = P$; it is called a *subspace extension* if Q is the intersection of an affine subspace of $\mathbb{R}^d$ and the nonnegative orthant $\mathbb{R}^d_+$. For instance, the polyhedron $Q_{\mathrm{spt}}(n)$ defined in the Introduction is a subspace extension of the spanning tree polytope $\mathrm{P}_{\mathrm{spt}}(n)$. A (finite) system of linear equations and inequalities whose solutions are the points in an extension Q of P is an *extended formulation* for P. The *size* of an extension is the number of its facets plus the dimension of the space it lies in. The *size* of an extended formulation is its number of inequalities (including nonnegativity constraints, but not equations) plus its number of variables. Clearly, the size of an extended formulation is at least as large as the size of the extension it describes. Conversely, every extension is described by an extended formulation of at most its size.

Extensions or extended formulations of a family of polytopes $P \subseteq \mathbb{R}^m$ (for varying m) are *compact* if their sizes and the encoding lengths of the coefficients needed to describe them can be bounded by a polynomial in m and the maximal encoding length of all components of all vertices of P. Clearly, the extension $Q_{\mathrm{spt}}(n)$ of $\mathrm{P}_{\mathrm{spt}}(n)$ from the Introduction is compact.

In our context, *sections* $s : X \to Q$ play a crucial role, i.e., maps that assign to every vertex $x \in X$ of P some point $s(x) \in Q \cap p^{-1}(x)$ in the intersection of the polyhedron Q and the *fiber* $p^{-1}(x) = \{y \in \mathbb{R}^d \ : \ p(y) = x\}$ of x under the projection p. Such a section induces a bijection between X and its image $s(X) \subseteq Q$, whose inverse is given by p. In the spanning tree example from the Introduction, the assignment $\chi(T) \mapsto y^{(T)} = (x^{(T)}, z^{(T)})$ defined such a section. Note that, in general, sections will not be induced by linear maps. In fact, if a section is induced by a linear map $s : \mathbb{R}^m \to \mathbb{R}^d$, then the intersection of Q with the affine subspace of $\mathbb{R}^d$ generated by $s(X)$ is isomorphic to P, thus Q has at least as many facets as P.

For a family $\mathcal{F}$ of subsets of X, an extension $Q \subseteq \mathbb{R}^d$ is said to be *indexed by* $\mathcal{F}$ if there is a bijection between $\mathcal{F}$ and $[d]$ such that (identifying $\mathbb{R}^{\mathcal{F}}$ with $\mathbb{R}^d$ via this bijection) the map $\mathbf{1}_{\mathcal{F}} = (\mathbf{1}_F)_{F \in \mathcal{F}} : X \to \{0,1\}^{\mathcal{F}}$ whose component functions are the characteristic functions $\mathbf{1}_F : X \to \{0,1\}$ (with $\mathbf{1}_F(x) = 1$ if and only if $x \in F$), is a section for the extension, i.e., $\mathbf{1}_{\mathcal{F}}(X) \subseteq Q$ and $p(\mathbf{1}_{\mathcal{F}}(x)) = x$ hold for all $x \in X$. For instance, the extension $Q_{\mathrm{spt}}(n)$ of $\mathrm{P}_{\mathrm{spt}}(n)$ is indexed by the family $\{\mathcal{T}(e) : e \in E_n\} \cup \{\mathcal{T}(e, v, u) : e \in E_n, v \in e, u \in [n] \setminus e\}$, where $\mathcal{T}(e)$ contains all spanning trees using edge e, and $\mathcal{T}(e, v, u)$ consists of all spanning trees in $\mathcal{T}(e)$ for which u and v are in the same component of $T \setminus \{e\}$.

In order to define the notion of symmetry of an extension precisely, let the group $\mathfrak{S}(d)$ of all permutations of $[d] = \{1, \ldots, d\}$ act on $\mathbb{R}^d$ by coordinate permutations. Thus we have $(\sigma.y)_j = y_{\sigma^{-1}(j)}$ for all $y \in \mathbb{R}^d$, $\sigma \in \mathfrak{S}(d)$, and $j \in [d]$.

Let $P \subseteq \mathbb{R}^m$ be a polytope and G be a group acting on $\mathbb{R}^m$ with $\pi.P = P$ for all $\pi \in G$, i.e., the action of G on $\mathbb{R}^m$ induces an action of G on the set X of vertices of P. An extension $Q \subseteq \mathbb{R}^d$ of P with projection $p : \mathbb{R}^d \to \mathbb{R}^m$ is *symmetric* (with respect to the action of G), if for every $\pi \in G$ there is a permutation $\kappa_\pi \in \mathfrak{S}(d)$ with $\kappa_\pi.Q = Q$ and

$$p(\kappa_\pi.y) = \pi.p(y) \quad \text{for all } y \in \mathbb{R}^d. \tag{2}$$

The prime examples of symmetric extensions arise from extended formulations that "look symmetric". To be more precise, we define an extended formulation $A^{=}y = b^{=}$, $A^{\leq}y \leq b^{\leq}$ describing the polyhedron $Q = \{y \in \mathbb{R}^d : A^{=}y = b^{=}, A^{\leq}y \leq b^{\leq}\}$ extending $P \subseteq \mathbb{R}^m$ as above to be *symmetric* (with respect to the action of G on the set X of vertices of P), if for every $\pi \in G$ there is a permutation $\kappa_\pi \in \mathfrak{S}(d)$ satisfying (2) and there are two permutations $\varrho^{=}_\pi$ and $\varrho^{\leq}_\pi$ of the rows of $(A^{=}, b^{=})$ and $(A^{\leq}, b^{\leq})$, respectively, such that the corresponding simultaneous permutations of the columns and the rows of the matrices $(A^{=}, b^{=})$ and $(A^{\leq}, b^{\leq})$ leaves them unchanged. Clearly, in this situation the permutations κ_π satisfy $\kappa_\pi.Q = Q$, which implies the following.

Lemma 1. *Every symmetric extended formulation describes a symmetric extension.*

One example of a symmetric extended formulation is the extended formulation for the spanning tree polytope described in the Introduction (with respect to the group G of all permutations of the nodes of the complete graph).

For the proof of the central result on the non-existence of certain symmetric subspace extensions (Theorem 1), a weaker notion of symmetry will be sufficient. We call an extension as above *weakly symmetric* (with respect to the action of G) if there is a section $s : X \to Q$ for which the action of G on $s(X)$ induced by the bijection s works by permutation of variables, i.e., for every $\pi \in G$ there is a permutation $\kappa_\pi \in \mathfrak{S}(d)$ with $s(\pi.x) = \kappa_\pi.s(x)$ for all $x \in X$. The following statement (and its proof, for which we refer to [12]) generalizes the construction of sections for symmetric extensions of matching polytopes described in Yannakakis' paper [17, Claim 1 in the proof of Thm. 1].

Lemma 2. *Every symmetric extension is weakly symmetric.*

Finally, the following result (again, we refer to [12] for a proof) will turn out to be useful in order to derive lower bounds on the sizes of symmetric extensions for one polytope from bounds for another one.

Lemma 3. *Let $Q \subseteq \mathbb{R}^d$ be an extension of the polytope $P \subseteq \mathbb{R}^m$ with projection $p : \mathbb{R}^d \to \mathbb{R}^m$, and let the face P' of P be an extension of a polytope $R \subseteq \mathbb{R}^k$ with projection $q : \mathbb{R}^m \to \mathbb{R}^k$. Then the face $Q' = p^{-1}(P') \cap Q \subseteq \mathbb{R}^d$ of Q is an extension of R via the composed projection $q \circ p : \mathbb{R}^d \to \mathbb{R}^k$.*

If the extension Q of P is symmetric with respect to an action of a group G on $\mathbb{R}^m$ (with $\pi.P = P$ for all $\pi \in G$), and a group H acts on $\mathbb{R}^k$ such that, for every $\tau \in H$, we have $\tau.R = R$, and there is some $\pi_\tau \in G$ with $\pi_\tau.P' = P'$ and $q(\pi_\tau.x) = \tau.q(x)$ for all $x \in \mathbb{R}^m$, then the extension Q' of R is symmetric (with respect to the action of the group H).

3 Yannakakis' Method

Here, we provide an abstract view on the method used by Yannakakis [17] in order to bound from below the sizes of symmetric extensions for perfect matching polytopes, without referring to these concrete poytopes. That method is capable of establishing lower bounds on the number of variables of weakly symmetric subspace extensions of certain polytopes. By the following lemma, which is basically Step 1 in the proof of [17, Theorem 1], such bounds imply similar lower bounds on the dimension of the ambient space and the number of facets for general symmetric extensions (that are not necessarily subspace extensions).

Lemma 4. *If, for a polytope P, there is a symmetric extension in $\mathbb{R}^{\tilde{d}}$ with f facets, then P has also a symmetric subspace extension in $\mathbb{R}^d$ with $d \leq 2\tilde{d} + f$.*

The following simple lemma provides the strategy for Yannakakis' method, which we need to extend slightly by allowing restrictions to affine subspaces.

Lemma 5. *Let $Q \subseteq \mathbb{R}^d$ be a subspace extension of the polytope $P \subseteq \mathbb{R}^m$ with vertex set $X \subseteq \mathbb{R}^m$, and let $s : X \to Q$ be a section for the extension. If $S \subseteq \mathbb{R}^m$ is an affine subspace, and, for some $X^\star \subseteq X \cap S$, the coefficients $c_x \in \mathbb{R}$ ($x \in X^\star$) yield an affine combination of a nonnegative vector*

$$\sum_{x \in X^\star} c_x s(x) \geq \mathbf{0}_d \quad \text{with} \quad \sum_{x \in X^\star} c_x = 1\,, \tag{3}$$

from the section images of the vertices in $X^\star$, then $\sum_{x \in X^\star} c_x x \in P \cap S$ holds.

Proof. Since Q is a subspace extension, we obtain $\sum_{x \in X^\star} c_x s(x) \in Q$ from $s(x) \in Q$ (for all $x \in X^\star$). Thus, if $p : \mathbb{R}^d \to \mathbb{R}^m$ is the projection of the extension, we derive

$$P \ni p(\sum_{x \in X^\star} c_x s(x)) = \sum_{x \in X^\star} c_x p(s(x)) = \sum_{x \in X^\star} c_x x\,. \tag{4}$$

As S is an affine subspace containing $X^\star$, we also have $\sum_{x \in X^\star} c_x x \in S$.

Due to Lemma 5 one can prove that subspace extensions of some polytope P with certain properties do not exist by finding, for such a hypothetical extension, a subset $X^\star$ of vertices of P and an affine subspace S containing $X^\star$, for which one can construct coefficients $c_x \in \mathbb{R}$ satisying (3) such that $\sum_{x\in X^\star} c_x x$ violates some inequality that is valid for $P \cap S$.

Actually, following Yannakakis, we will not apply Lemma 5 directly to a hypothetical small weakly symmetric subspace extension, but we will rather first construct another subspace extension from the one assumed to exist that is indexed by some convenient family $\mathcal{F}$. We say that an extension Q of a polytope P is *consistent* with a family $\mathcal{F}$ of subsets of the vertex set X of P if there is a section $s : X \to Q$ for the extension such that, for every component function s_j of s, there is a subfamily $\mathcal{F}_j$ of $\mathcal{F}$ such that s_j is constant on every set in $\mathcal{F}_j$, and the sets in $\mathcal{F}_j$ partition X. In this situation, we also call the section s *consistent* with $\mathcal{F}$. The proof of the following lemma can be found in [12].

Lemma 6. *If $P \subseteq \mathbb{R}^m$ is a polytope and $\mathcal{F}$ is a family of vertex sets of P for which there is some extension Q of P that is consistent with $\mathcal{F}$, then there is some extension Q' for P that is indexed by $\mathcal{F}$. If Q is a subspace extension, then Q' can be chosen to be a subspace extension as well.*

Lemmas 5 and 6 suggest the following strategy for proving that subspace extensions of some polytope P with certain properties (e.g., being weakly symmetric and using at most B variables) do not exist by (a) exhibiting a family $\mathcal{F}$ of subsets of the vertex set X of P with which such an extension would be consistent and (b) determining a subset $X^\star \subset X$ of vertices and an affine subspace S containing $X^\star$, for which one can construct coefficients $c_x \in \mathbb{R}$ satisying

$$\sum_{x\in X^\star} c_x \mathbf{1}_{\mathcal{F}}(x) \geq \mathbf{0}_{\mathcal{F}} \quad \text{with} \quad \sum_{x\in X^\star} c_x = 1\,, \tag{5}$$

such that $\sum_{x\in X^\star} c_x x$ violates some inequality that is valid for $P \cap S$.

Let us finally investigate more closely the sections that come with weakly symmetric extensions. In particular, we will discuss an approach to find suitable families $\mathcal{F}$ within the strategy mentioned above in the following setting. Let $Q \subseteq \mathbb{R}^d$ be a weakly symmetric extension of the polytope $P \subseteq \mathbb{R}^m$ (with respect to an action of the group G on the vertex set X of P) along with a section $s : X \to Q$ such that for every $\pi \in G$ there is a permutation $\kappa_\pi \in \mathfrak{S}(d)$ that satisfies $s(\pi.x) = \kappa_\pi.s(x)$ for all $x \in X$ (with $(\kappa_\pi.s(x))_j = s_{\kappa_\pi^{-1}(j)}(x)$).

In this setting, we can define an action of G on the set $\mathcal{S} = \{s_1, \ldots, s_d\}$ of the component functions of the section $s : X \to Q$ with $\pi.s_j = s_{\kappa^{-1}_{\pi^{-1}}(j)} \in \mathcal{S}$ for each $j \in [d]$. In order to see that this definition indeed is well-defined (note that $s_1, \ldots, s_d$ need not be pairwise distinct functions) and yields a group action, observe that, for each $j \in [d]$ and $\pi \in G$, we have

$$(\pi.s_j)(x) = s_{\kappa^{-1}_{\pi^{-1}}(j)}(x) = (\kappa_{\pi^{-1}}.s(x))_j = s_j(\pi^{-1}.x) \quad \text{for all } x \in X\,, \tag{6}$$

from which one deduces $1.s_j = s_j$ for the one-element 1 in G as well as $(\pi\pi').s_j = \pi.(\pi'.s_j)$ for all $\pi, \pi' \in G$. The *isotropy* group of $s_j \in \mathcal{S}$ under this action is $\mathrm{iso}_G(s_j) = \{\pi \in G \ : \ \pi.s_j = s_j\}$. From (6) one sees that, for all $x \in X$ and

$\pi \in \mathrm{iso}_G(s_j)$, we have $s_j(x) = s_j(\pi^{-1}.x)$. Thus, s_j is constant on every orbit of the action of the subgroup $\mathrm{iso}_G(s_j)$ of G on X. We conclude the following.

Remark 1. In the setting described above, if $\mathcal{F}$ is a family of subsets of X such that, for each $j \in [d]$, there is a sub-family $\mathcal{F}_j$ partitioning X and consisting of vertex sets each of which is contained in an orbit under the action of $\mathrm{iso}_G(s_j)$ on X, then s is consistent with $\mathcal{F}$.

In general, it will be impossible to identify the isotropy groups $\mathrm{iso}_G(s_j)$ without more knowledge on the section s. However, for each isotropy group $\mathrm{iso}_G(s_j)$, one can at least bound its index $(G : \mathrm{iso}_G(s_j))$ in G.

Lemma 7. *In the setting described above, we have* $(G : \mathrm{iso}_G(s_j)) \leq d$.

Proof. This follows readily from the fact that the index $(G : \mathrm{iso}_G(s_j))$ of the isotropy group of the element $s_j \in \mathcal{S}$ under the action of G on $\mathcal{S}$ equals the cardinality of the orbit of s_j under that action, which due to $|\mathcal{S}| \leq d$, clearly is bounded from above by d.

The bound provided in Lemma 7 can become useful, in case one is able to establish a statement like "if $\mathrm{iso}_G(s_j)$ has index less than τ in G then it contains a certain subgroup H_j". Choosing $\mathcal{F}_j$ as the family of orbits of X under the action of the subgroup H_j of G, then $\mathcal{F} = \mathcal{F}_1 \cup \cdots \cup \mathcal{F}_d$ is a familiy as in Remark 1. If this family (or any refinement of it) can be used to perform Step (b) in the strategy outlined in the paragraph right after the statement of Lemma 6, then one can conclude the lower bound $d \geq \tau$ on the number of variables d in an extension as above.

4 Bounds on Symmetric Extensions of $\mathrm{P}^{\ell}_{\mathrm{match}}(n)$

In this section, we use Yannakakis' method described in Section 3 to prove the following result.

Theorem 1. *For every $n \geq 3$ and odd ℓ with $\ell \leq \frac{n}{2}$, there exists no weakly symmetric subspace extension for $\mathrm{P}^{\ell}_{\mathrm{match}}(n)$ with at most $\binom{n}{(\ell-1)/2}$ variables (with respect to the group $\mathfrak{S}(n)$ acting via permuting the nodes of K_n as described in the Introduction).*

From Theorem 1, we can derive the following more general lower bounds. Since we need it in the proof of the next result, and also for later reference, we state a simple fact on binomial coefficients first.

Lemma 8. *For each constant $b \in \mathbb{N}$ there is some constant $\beta > 0$ with $\binom{M-b}{N} \geq \beta\binom{M}{N}$ for all large enough $M \in \mathbb{N}$ and $N \leq \frac{M}{2}$.*

Theorem 2. *There is a constant $C > 0$ such that, for all n and $1 \leq \ell \leq \frac{n}{2}$, the size of every extension for $\mathrm{P}^{\ell}_{\mathrm{match}}(n)$ that is symmetric (with respect to the group $\mathfrak{S}(n)$ acting via permuting the nodes of K_n as described in the Introduction) is bounded from below by $C \cdot \binom{n}{\lfloor(\ell-1)/2\rfloor}$.*

Proof. For odd ℓ, this follows from Theorem 1 using Lemmas 1, 2, and 4. For even ℓ, the polytope $\mathrm{P}^{\ell-1}_{\mathrm{match}}(n-2)$ is (isomorphic to) a face of $\mathrm{P}^{\ell-1}_{\mathrm{match}}(n)$ defined by $x_e = 1$ for an arbitrary edge e of K_n. From this, as $\ell - 1$ is odd (and not larger than $(n-2)/2$) with $\lfloor(\ell-2)/2\rfloor = \lfloor(\ell-1)/2\rfloor$, and due to Lemma 8, the theorem follows by Lemma 3.

For even n and $\ell = n/2$, Theorem 2 provides a similar bound to Yannakakis result (see Step 2 in the proof of [17, Theorem 1]) that no weakly symmetric subspace extension of the perfect matching polytope of K_n has a number of variables that is bounded by $\binom{n}{k}$ for any $k < n/4$.

Theorem 2 in particular implies that the size of every symmetric extension for $\mathrm{P}^{\ell}_{\mathrm{match}}(n)$ with $\Omega(\log n) \le \ell \le n/2$ is bounded from below by $n^{\Omega(\log n)}$, which has the following consequence.

Corollary 1. *For $\Omega(\log n) \le \ell \le n/2$, there is no compact extended formulation for $\mathrm{P}^{\ell}_{\mathrm{match}}(n)$ that is symmetric (with respect to the group $G = \mathfrak{S}(n)$ acting via permuting the nodes of K_n as described in the Introduction).*

The rest of this section is devoted to indicate the proof of Theorem 1. Throughout, with $\ell = 2k+1$, we assume that $Q \subseteq \mathbb{R}^d$ with $d \le \binom{n}{k}$ is a weakly symmetric subspace extension of $\mathrm{P}^{2k+1}_{\mathrm{match}}(n)$ for $4k+2 \le n$. We will only consider the case $k \ge 1$, as for $\ell = 1$ the theorem trivially is true (note that we restrict to $n \ge 3$). Weak symmetry is meant with respect to the action of $G = \mathfrak{S}(n)$ on the set X of vertices of $\mathrm{P}^{2k+1}_{\mathrm{match}}(n)$ as described in the Introduction, and we assume $s : X \to Q$ to be a section as required in the definition of weak symmetry. Thus, we have $X = \{\chi(M) \in \{0,1\}^{E_n} : M \in \mathcal{M}^{2k+1}(n)\}$, where $\mathcal{M}^{2k+1}(n)$ is the set of all matchings $M \subseteq E_n$ with $|M| = 2k+1$ in the complete graph $K_n = (V, E)$ (with $V = [n]$), and $(\pi.\chi(M))_{\{v,w\}} = \chi(M)_{\{\pi^{-1}(v),\pi^{-1}(w)\}}$ holds for all $\pi \in \mathfrak{S}(n)$, $M \in \mathcal{M}^{2k+1}(n)$, and $\{v,w\} \in E$.

In order to identify suitable subgroups of the isotropy groups $\mathrm{iso}_{\mathfrak{S}(n)}(s_j)$ (see the remarks at the end of Section 3), we use the following result on subgroups of the symmetric group $\mathfrak{S}(n)$, where $\mathfrak{A}(n) \subseteq \mathfrak{S}(n)$ is the *alternating group* formed by all even permutations of $[n]$. This result is Claim 2 in the proof of Thm. 1 of Yannakakis paper [17]. Its proof relies on a theorem of Bochert's [3] stating that any subgroup of $\mathfrak{S}(m)$ that acts primitively on $[m]$ and does not contain $\mathfrak{A}(m)$ has index at least $\lfloor(m+1)/2\rfloor!$ in $\mathfrak{S}(m)$ (see [16, Thm. 14.2]).

Lemma 9. *For each subgroup U of $\mathfrak{S}(n)$ with $(\mathfrak{S}(n) : U) \le \binom{n}{k}$ for $k < \frac{n}{4}$, there is a $W \subseteq [n]$ with $|W| \le k$ and $H_j = \{\pi \in \mathfrak{A}(n) : \pi(v) = v$ for all $v \in W\} \subseteq U$.*

As we assumed $d \le \binom{n}{k}$ (with $k < \frac{n}{4}$ due to $4k+2 \le n$), Lemmas 7 and 9 imply $H_j \subseteq \mathrm{iso}_{\mathfrak{S}(n)}(s_j)$ for all $j \in [d]$. For each $j \in [d]$, two vertices $\chi(M)$ and $\chi(M')$ of $\mathrm{P}^{2k+1}_{\mathrm{match}}(n)$ (with $M, M' \in \mathcal{M}^{2k+1}(n)$) are in the same orbit under the action of the group H_j if and only if we have

$$M \cap E(V_j) = M' \cap E(V_j) \quad \text{and} \quad V_j \setminus M = V_j \setminus M' . \tag{7}$$

Indeed, it is clear that (7) holds if we have $\chi(M') = \pi.\chi(M)$ for some permutation $\pi \in H_j$. In turn, if (7) holds, then there clearly is some permutation $\pi \in \mathfrak{S}(n)$ with $\pi(v) = v$ for all $v \in V_j$ and $M' = \pi.M$. Due to

$|M| = 2k+1 > 2|V_j|$ there is some edge $\{u,w\} \in M$ with $u,w \notin V_j$. Denoting by $\tau \in \mathfrak{S}(n)$ the transposition of u and w, we thus also have $\pi\tau(v) = v$ for all $v \in V_j$ and $M' = \pi\tau.M$. As one of the permutations π and $\pi\tau$ is even, say π', we find $\pi' \in H_j$ and $M' = \pi'.M$, proving that M and M' are contained in the same orbit under the action of H_j.

As it will be convenient for Step (b) (referring to the strategy described after the statement of Lemma 6), we will use the following refinements of the partitionings of X into orbits of H_j (as mentioned at the end of Section 3). Clearly, for $j \in [d]$ and $M, M' \in \mathcal{M}^{2k+1}(n)$,

$$M \setminus E(V \setminus V_j) = M' \setminus E(V \setminus V_j) \tag{8}$$

implies (7). Thus, for each $j \in [d]$, the equivalence classes of the equivalence relation defined by (8) refine the partitioning of X into orbits under H_j, and we may use the collection of all these equivalence classes (for all $j \in [d]$) as the family $\mathcal{F}$ in Remark 1. With

$$\Lambda = \{(A,B) : A \subseteq E \text{ matching and there is some } j \in [d] \text{ with } A \subseteq E \setminus E(V \setminus V_j), B = V_j \setminus V(A)\},$$

(with $V(A) = \bigcup_{a \in A} a$) we hence have $\mathcal{F} = \{F(A,B) : (A,B) \in \Lambda\}$, where

$$F(A,B) = \{\chi(M) : M \in \mathcal{M}^{2k+1}(n), A \subseteq M \subseteq E(V \setminus B)\}.$$

In order to construct a subset $X^\star \subseteq X$ which will be used to derive a contradiction as mentioned after Equation (5), we choose two arbitrary disjoint subsets $V_\star, V^\star \subset V$ of nodes with $|V_\star| = |V^\star| = 2k+1$, and define $\mathcal{M}^\star = \{M \in \mathcal{M}^{2k+1}(n) : M \subseteq E(V_\star \cup V^\star)\}$ as well as $X^\star = \{\chi(M) : M \in \mathcal{M}^\star\}$. Thus, $\mathcal{M}^\star$ is the set of perfect matchings on $K(V_\star \cup V^\star)$. Clearly, $X^\star$ is contained in the affine subspace S of $\mathbb{R}^E$ defined by $x_e = 0$ for all $e \in E \setminus E(V_\star \cup V^\star)$. In fact, $X^\star$ is the vertex set of the face $\mathrm{P}^{2k+1}_{\mathrm{match}}(n) \cap S$ of $\mathrm{P}^{2k+1}_{\mathrm{match}}(n)$, and for this face the inequality $x(V_\star : V^\star) \geq 1$ is valid (where $(V_\star : V^\star)$ is the set of all edges having one node in $V_\star$ and the other one in $V^\star$), since every matching $M \in \mathcal{M}^\star$ intersects $(V_\star : V^\star)$ in an odd number of edges. Therefore, in order to derive the desired contradiction, it suffices to find $c_x \in \mathbb{R}$ (for all $x \in X^\star$) with

$\sum_{x \in X^\star} c_x = 1$, $\sum_{x \in X^\star} c_x \cdot \mathbf{1}_{\mathcal{F}}(x) \geq \mathbf{0}_{\mathcal{F}}$, and $\sum_{x \in X^\star} c_x \sum_{e \in (V_\star : V^\star)} x_e = 0$. For the details on how this can be done we refer to [12].

5 A Non-symmetric Extension for $\mathrm{P}^{\ell}_{\mathrm{match}}(n)$

We shall establish the following result on the existence of extensions for cardinality restricted matching polytopes in this section.

Theorem 3. *For all n and ℓ, there are extensions for $\mathrm{P}^{\ell}_{\mathrm{match}}(n)$ whose sizes can be bounded by $2^{O(\ell)} n^2 \log n$ (and for which the encoding lengths of the coefficients needed to describe the extensions by linear systems can be bounded by a constant).*

In particular, Theorem 3 implies the following, although, according to Corollary 1, no compact symmetric extended formulations exist for $\mathrm{P}^{\ell}_{\mathrm{match}}(n)$ with $\ell = \Theta(\log n)$.

Corollary 2. *For all n and $\ell \le O(\log n)$, there are compact extended formulations for $\mathrm{P}^{\ell}_{\mathrm{match}}(n)$.*

The proof of Theorem 3 relies on the following result on the existence of small families of *perfect-hash functions*, which is from [1, Sect. 4]. Its proof is based on results from [8,15].

Theorem 4 (Alon, Yuster, Zwick [1]). *There are maps $\phi_1, \dots, \phi_{q(n,r)} : [n] \to [r]$ with $q(n,r) \le 2^{O(r)} \log n$ such that, for every $W \subseteq [n]$ with $|W| = r$, there is some $i \in [q(n,r)]$ for which the map ϕ_i is bijective on W.*

Furthermore, we will use the following two auxilliary results that can be derived from general results on *polyhedral branching systems* [11, see Cor. 3 and Sect. 4.4]. The first one (Lemma 10) provides a construction of an extension of a polytope that is specified as the convex hull of some polytopes of which extensions are already available. In fact, in this section it will be needed only for the case that these extensions are the polytopes themselves (this is a special case of a result of Balas', see [2, Thm.2.1]). However, we will face the slightly more general situation in our treatment of cycle polytopes in Section 6.

Lemma 10. *If the polytopes $P_i \subseteq \mathbb{R}^m$ (for $i \in [q]$) have extensions Q_i of size s_i, respectively, then $P = \mathrm{conv}(P_1 \cup \dots \cup P_q)$ has an extension of size $\sum_{i=1}^{q}(s_i+2)+1$.*

The second auxilliary result that we need deals with describing a 0/1-polytope that is obtained by splitting variables of a 0/1-polytope of which a linear description is already available.

Lemma 11. *Let $\mathcal{S}$ be a set of subsets of $[t]$, $P = \mathrm{conv}\{\chi(S) \in \{0,1\}^t : S \in \mathcal{S}\} \subseteq \mathbb{R}^t$, the corresponding 0/1-polytope, $J = J(1) \uplus \dots \uplus J(t)$ a disjoint union of finite sets $J(i)$,*

$$\mathcal{S}^{\star} = \{S^{\star} \subseteq J : \text{There is some } S \in \mathcal{S} \text{ with } \\ |S^{\star} \cap J(i)| = 1 \text{ for all } i \in S, |S^{\star} \cap J(i)| = 0 \text{ for all } i \notin S\}\,, \quad (9)$$

and $P^{\star} = \mathrm{conv}\{\chi(S^{\star}) \in \{0,1\}^J : S^{\star} \in \mathcal{S}^{\star}\}$. If $P = \{y \in [0,1]^t : Ay \le b\}$ for some $A \in \mathbb{R}^{s \times t}$ and $b \in \mathbb{R}^s$, then

$$P^{\star} = \{x \in [0,1]^J : \sum_{i=1}^{t} A_{\star,i} \cdot \sum_{j \in J(i)} x_j \le b_i \text{ for all } i \in [t]\}\,. \quad (10)$$

In order to prove Theorem 3, let $\phi_1, \dots, \phi_q$ be maps as guaranteed to exist by Theorem 4 with $r = 2\ell$ and $q = q(n, 2\ell) \le 2^{O(\ell)} \log n$, and denote $\mathcal{M}_i = \{M \in \mathcal{M}^{\ell}(n) : \phi_i \text{ is bijective on } V(M)\}$ for each $i \in [q]$. By Theorem 4, we have $\mathcal{M}^{\ell}(n) = \mathcal{M}_1 \cup \dots \cup \mathcal{M}_q$. Consequently,

$$\mathrm{P}^{\ell}_{\mathrm{match}}(n) = \mathrm{conv}(P_1 \cup \dots \cup P_q) \quad (11)$$

with $P_i = \operatorname{conv}\{\chi(M) : M \in \mathcal{M}_i\}$ for all $i \in [q]$, where we have

$$P_i = \{x \in \mathbb{R}^E_+ : x_{E\setminus E_i} = \mathbf{0}, x(\delta(\phi_i^{-1}(s))) = 1 \text{ for all } s \in [2\ell], \\ x(E_i(\phi_i^{-1}(S))) \le (|S|-1)/2 \text{ for all } S \subseteq [2\ell], |S| \text{ odd}\}\,,$$

where $E_i = E \setminus \bigcup_{j\in[2\ell]} E(\phi_i^{-1}(j))$. This follows by Lemma 11 from Edmonds' linear description (1) of the perfect matching polytope $\mathrm{P}^\ell_{\mathrm{match}}(2\ell)$ of $K_{2\ell}$. As the sum of the number of variables and the number of inequalities in the description of P_i is at most $2^{O(\ell)} + n^2$ (the summand n^2 comes from the nonnegativity constraints on $x \in \mathbb{R}^E_+$ and the constant in $O(\ell)$ is independent of i), we obtain an extension of $\mathrm{P}^\ell_{\mathrm{match}}(n)$ of size $2^{O(\ell)}n^2 \log n$ by Lemma 10. This proves Theorem 3.

6 Extensions for Cycle Polytopes

By a modification of Yannakakis' construction for the derivation of lower bounds on the sizes of symmetric extensions for traveling salesman polytopes from the corresponding lower bounds for matching polytopes [17, Thm. 2], we obtain lower bounds on the sizes of symmetric extensions for $\mathrm{P}^\ell_{\mathrm{cycl}}(n)$. The lower bound $\ell \ge 42$ in the statement of the theorem (whose proof can be found in [12]) is convenient with respect to both formulating the bound and proving its validity.

Theorem 5. *There is a constant $C' > 0$ such that, for all n and $42 \le \ell \le n$, the size of every extension for $\mathrm{P}^\ell_{\mathrm{cycl}}(n)$ that is symmetric (with respect to the group $\mathfrak{S}(n)$ acting via permuting the nodes of K_n as described in the Introduction) is bounded from below by $C' \cdot \binom{\lfloor \frac{n}{3} \rfloor}{\lfloor (\lfloor \frac{\ell}{6} \rfloor - 1)/2 \rfloor}$.*

Corollary 3. *For $\Omega(\log n) \le \ell \le n$, there is no compact extended formulation for $\mathrm{P}^\ell_{\mathrm{cycl}}(n)$ that is symmetric (with respect to the group $\mathfrak{S}(n)$ acting via permuting the nodes of K_n as described in the Introduction).*

On the other hand, if we drop the symmetry requirement, we find extensions of the following size.

Theorem 6. *For all n and ℓ, there are extensions for $\mathrm{P}^\ell_{\mathrm{cycl}}(n)$ whose sizes can be bounded by $2^{O(\ell)}n^3 \log n$ (and for which the encoding lengths of the coefficients needed to describe the extensions by linear systems can be bounded by a constant).*

Before we prove Theorem 6, we state a consequence that is similar to Corollary 1 for matching polytopes. It shows that, despite the non-existence of symmetric extensions for the polytopes associated with cycles of length $\Theta(\log n)$ (Corollary 3), there are non-symmetric compact extensions of these polytopes.

Corollary 4. *For all n and $\ell \le O(\log n)$, there are compact extended formulations for $\mathrm{P}^\ell_{\mathrm{cycl}}(n)$.*

The rest of the section is devoted to prove Theorem 6, i.e., to construct an extension of $\mathrm{P}^\ell_{\mathrm{cycl}}(n)$ whose size is bounded by $2^{O(\ell)}n^3 \log n$. We proceed similarly to the proof of Theorem 3 (the construction of extensions for matching polytopes), this time starting with maps $\phi_1, \ldots, \phi_q$ as guaranteed to exist by Theorem 4 with $r = \ell$ and $q = q(n, \ell) \leq 2^{O(\ell)} \log n$, and defining $\mathcal{C}_i = \{C \in \mathcal{C}^\ell(n) : \phi_i \text{ is bijective on } V(C)\}$ for each $i \in [q]$. Thus, we have $\mathcal{C}^\ell(n) = \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_q$, and hence, $\mathrm{P}^\ell_{\mathrm{cycl}}(n) = \mathrm{conv}(P_1 \cup \cdots \cup P_q)$ with $P_i = \mathrm{conv}\{\chi(C) : C \in \mathcal{C}_i\}$ for all $i \in [q]$. Due to Lemma 10, it suffices to exhibit, for each $i \in [q]$, an extension of P_i of size bounded by $O(2^\ell \cdot n^3)$ (with the constant independent of i). Towards this end, let, for $i \in [q]$, $V_c = \phi_i^{-1}(c)$ for all $c \in [\ell]$, and define $P_i(v^\star) = \mathrm{conv}\{\chi(C) : C \in \mathcal{C}_i, v^\star \in V(C)\}$ for each $v^\star \in V_\ell$. Thus, we have $P_i = \mathrm{conv} \bigcup_{v^\star \in V_\ell} P_i(v^\star)$, and hence, due to Lemma 10, it suffices to construct extensions of the $P_i(v^\star)$, whose sizes are bounded by $O(2^\ell \cdot n^2)$.

In order to derive such extensions define, for each $i \in [q]$ and $v^\star \in V_\ell$, a directed acyclic graph D with nodes (A, v) for all $A \subseteq [\ell - 1]$ and $v \in \phi_i^{-1}(A)$, as well as two additional nodes s and t, and arcs $\big(s, (\{\phi_i(v)\}, v)\big)$ and $\big(([\ell-1], v), t\big)$ for all $v \in \phi_i^{-1}([\ell - 1])$, as well as $\big((A, v), (A \cup \{\phi_i(w)\}, w)\big)$ for all $A \subseteq [\ell - 1]$, $v \in \phi_i^{-1}(A)$, and $w \in \phi_i^{-1}([\ell - 1] \setminus A)$. This is basically the dynamic programming digraph (using an idea going back to [10]) from the color-coding method for finding paths of prescribed lengths described in [1]. Each s-t-path in D corresponds to a cycle in $\mathcal{C}_i$ that visits $v^\star$, and each such cycle, in turn, corresponds to two s-t-paths in D (one for each of the two directions of transversal).

Defining $Q_i(v^\star)$ as the convex hull of the characteristic vectors of all s-t-paths in D in the arc space of D, we find that $P_i(v^\star)$ is the image of $Q_i(v^\star))$ under the projection whose component function corresponding to the edge $\{v, w\}$ of K_n is given by the sum of all arc variables corresponding to arcs $((A, v), (A', w))$ (for $A, A' \subseteq [\ell - 1]$) if $v^\star \notin \{v, w\}$, and by the sum of the two arc variables corresponding to $(s, (\{\phi_i(w)\}, w))$ and $(([\ell - 1], w), t)$ in case of $v = v^\star$. Clearly, $Q_i(v^\star)$ can be described by nonnegativity constraints, flow conservation constraints for all nodes in D different from s and t, and by the equation stating that there must be exactly one flow-unit leaving s. As the number of arcs of D is in $O(2^\ell \cdot n^2)$, we thus have found an extension of $P_i(v^\star)$ of the desired size.

7 Conclusions

The results presented in this paper demonstrate that there are polytopes which have compact extended formulations though they do not admit symmetric ones. These polytopes are associated with matchings (or cycles) of some prescribed cardinalities (see [4] for a recent survey on general cardinality restricted combinatorial optimization problems). Similarly, for the permutahedron associated with $[n]$ there is a gap between the smallest sizes $\Theta(n \log n)$ of a non-symmetric extension [9] and $\Theta(n^2)$ of a symmetric extension [14].

Nevertheless, the question whether there are compact extended formulations for general matching polytopes (or for perfect matching polytopes), remains one of the most interesting open question here. In fact, it is even unknown whether there are (non-symmetric) extended formulations of these polytopes of size $2^{o(n)}$.

Actually, it seems that there are almost no lower bounds known on the sizes of (not necessarily symmetric) extensions, except for the one obtained by the observation that every extension Q of a polytope P with f faces has at least f faces itself, thus Q has at least $\log f$ facets (since a face is uniquely determined by the subset of facets it is contained in) [9]. It would be most interesting to obtain other lower bounds, including special ones for 0/1-polytopes.

Acknowledgements. We thank Christian Bey for useful discussions on subspaces that are invariant under coordinate permutations.

References

1. Alon, N., Yuster, R., Zwick, U.: Color-coding. J. Assoc. Comput. Mach. 42(4), 844–856 (1995)
2. Balas, E.: Disjunctive programming and a hierarchy of relaxations for discrete optimization problems. SIAM J. Algebraic Discrete Methods 6(3), 466–486 (1985)
3. Bochert, A.: Ueber die Zahl der verschiedenen Werthe, die eine Function gegebener Buchstaben durch Vertauschung derselben erlangen kann. Math. Ann. 33(4), 584–590 (1889)
4. Bruglieri, M., Ehrgott, M., Hamacher, H.W., Maffioli, F.: An annotated bibliography of combinatorial optimization problems with fixed cardinality constraints. Discrete Appl. Math. 154(9), 1344–1357 (2006)
5. Conforti, M., Cornuéjols, G., Zambelli, G.: Extended formulations in combinatorial optimization. Tech. Rep., Università di Padova (2009)
6. Edmonds, J.: Maximum matching and a polyhedron with 0, 1-vertices. J. Res. Nat. Bur. Standards Sect. B 69B, 125–130 (1965)
7. Edmonds, J.: Matroids and the greedy algorithm. Math. Programming 1, 127–136 (1971)
8. Fredman, M.L., Komlós, J., Szemerédi, E.: Storing a sparse table with $O(1)$ worst case access time. J. Assoc. Comput. Mach. 31(3), 538–544 (1984)
9. Goemans, M.: Smallest compact formulation for the permutahedron, `http://www-math.mit.edu/~goemans/publ.html`
10. Held, M., Karp, R.M.: A dynamic programming approach to sequencing problems. J. Soc. Indust. Appl. Math. 10, 196–210 (1962)
11. Kaibel, V., Loos, A.: Branched polyhedral systems. In: Eisenbrand, F., Shepherd, B. (eds.) IPCO 2010. LNCS, vol. 6080, pp. 177–190. Springer, Heidelberg (2010)
12. Kaibel, V., Pashkovich, K., Theis, D.O.: Symmetry matters for the sizes of extended formulations. arXiv:0911.3712v1 [math.CO]
13. Kipp Martin, R.: Using separation algorithms to generate mixed integer model reformulations. Tech. Rep., University of Chicago (1987)
14. Pashkovich, K.: Tight lower bounds on the sizes of symmetric extensions of permutahedra and similar results (in preparation)
15. Schmidt, J.P., Siegel, A.: The spatial complexity of oblivious k-probe hash functions. SIAM J. Comput. 19(5), 775–786 (1990)
16. Wielandt, H.: Finite permutation groups. Translated from the German by Bercov, R. Academic Press, New York (1964)
17. Yannakakis, M.: Expressing combinatorial optimization problems by linear programs. J. Comput. System Sci. 43(3), 441–466 (1991)

A 3-Approximation for Facility Location with Uniform Capacities

Ankit Aggarwal[1], L. Anand[2], Manisha Bansal[3], Naveen Garg[4,⋆], Neelima Gupta[3], Shubham Gupta[5], and Surabhi Jain[6]

[1] Tower Research Capital LLC, Gurgaon
[2] Georgia Tech
[3] University of Delhi
[4] Indian Institute of Technology Delhi
[5] U. of Waterloo
[6] Morgan Stanley, Mumbai

Abstract. We consider the facility location problem where each facility can serve at most U clients. We analyze a local search algorithm for this problem which uses only the operations of add, delete and swap and prove that any locally optimum solution is no more than 3 times the global optimum. This improves on a result of Chudak and Williamson who proved an approximation ratio of $3 + 2\sqrt{2}$ for the same algorithm. We also provide an example which shows that our analysis is tight.

1 Introduction

In a facility location problem we are given a set of clients C and facility locations F. Opening a facility at location $i \in F$ costs f_i (the *facility cost*). The cost of servicing a client j by a facility i is given by $c_{i,j}$ (the *service cost*) and these costs form a metric — for facilities i, i' and clients j, j', $c_{i',j'} \leq c_{i',j} + c_{i,j} + c_{i,j'}$. The objective is to determine which locations to open facilities in, so that the total cost for opening the facilities and for serving all the clients is minimized. Note that in this setting each client would be served by the open facility which offers the smallest service cost.

When the number of clients that a facility can serve is bounded, we have a *capacitated facility location problem*. In this paper we assume that these capacities are the same, U, for all facilities. For this problem of uniform capacities the first approximation algorithm was due to Chudak and Williamson [2] who analyzed a local search algorithm and proved that any locally optimum solution has cost no more than 6 times the facility cost plus 5 times the service cost of an (global) optimum solution. In this paper we refer to such a guarantee as a (6,5)-approximation; note that this is different from the bi-criterion guarantees for which this notation is typically used. The result of Chudak and Williamson

⋆ Work done as part of the "Approximation Algorithms" partner group of MPI-Informatik, Germany.

built on earlier work of Korupolu, Plaxton and Rajaraman [3] who were the first to analyze local search algorithms for facility location problems.

Given the set of open facilities, the best way of serving the clients, can be determined by solving an assignment problem. Thus any solution is completely determined by the set of open facilities. The local search procedure analyzed by Chudak and Williamson, starts with an arbitrary set of open facilities and then updates this set, using one of the operations `add`, `delete`, `swap`, whenever that operation reduces the total costs of the solution. We show that a solution which is locally optimum with respect to this same set of operations is a (3,3)-approximation. We then show that our analysis of this local search algorithm is best possible by demonstrating an instance where the locally optimum solution is three times the optimum solution.

When facilities have different capacities, the best result known is a (6,5)-approximation by Zhang, Chen and Ye [7]. The local search in this case relies on a multi-exchange operation, in which, loosely speaking, a subset of facilities from the current solution is exchanged with a subset not in the solution. This result improves on a (8,4)-approximation by Mahdian and Pal [4] and a (9,5) approximation by Pal, Tardos and Wexler [6].

For capacitated facility location, the only algorithms known are based on local search. One version of capacitated facility location arises when we are allowed to make multiple copies of the facilities. Thus if facility i has capacity U_i and opening cost f_i, then to serve $k > U_i$ clients by facility i we need to open $\lceil k/U_i \rceil$ copies of i and incur an opening cost $f_i \lceil k/U_i \rceil$. This version is usually referred to as "facility location with soft capacities" and the best known algorithm for this problem is a 2-approximation [5].

All earlier work for capacitated facility location (uniform or non-uniform) reroutes all clients in a swap operation from the facility which is closing to one of the facilities being opened. This however can be quite expensive and cannot lead to the tight bounds that we achieve in this paper. We use the idea of Arya et.al. [1] to reassign clients of the facility being closed in a swap operation to other facilities in our current solution. However, to be able to handle the capacity constraints in this reassignment we need to extend the notion of the mapping between clients used in [1] to a fractional assignment. As in earlier work, we use the fact that when we have a local optimum, no operation leads to an improvement in cost. However, we now take carefully defined *linear combinations* of the inequalities capturing this local optimality. All previous work that we are aware of seems to only use the *sum* of such inequalities and therefore requires additional properties like the integrality of the assignment polytope to carry the argument through [2]. Our approach is therefore more general and amenable to better analysis. The idea of doing things fractionally appears more often in our analysis. Thus, when analyzing the cost of an operation we assign clients fractionally to the facilities and rely on the fact that such a fractional assignment cannot be better than the optimum assignment.

In Section 4 we give a tight example that relies on the construction of a suitable triangle free set-system. While this construction itself is quite straightforward, this is the first instance we know of, where such an idea being applied to prove a large locality gap.

2 Preliminaries

Let C be the set of clients and F denote the facility locations. Let S (resp. O) be the set of open facilities in our solution (resp. optimum solution). We abuse notation and use S (resp O) to denote our solution (resp. optimum solution). Initially S is an arbitrary set of facilities which can serve all the clients. Let $\text{cost}(S)$ denote the total cost (facility plus service) of solution S. The three operations that make up our local search algorithm are

Add. For $s \notin S$, if $\text{cost}(S + \{s\}) < \text{cost}(S)$ then $S \leftarrow S + \{s\}$.
Delete. For $s \in S$, if $\text{cost}(S - \{s\}) < \text{cost}(S)$ then $S \leftarrow S - \{s\}$.
Swap. For $s \in S$ and $s' \notin S$, if $\text{cost}(S - \{s\} + \{s'\}) < \text{cost}(S)$ then $S \leftarrow S - \{s\} + \{s'\}$.

S is locally optimum if none of the three operations are possible and at this point our algorithm stops. Polynomial running time can be ensured at the expense of an additive ϵ in the approximation guarantee by doing a local step only if the cost reduces by more than an $1 - \epsilon/n$ factor, for $\epsilon > 0$.

We use $f_i, i \in F$ to denote the cost of opening a facility at location i. Let S_j, O_j denote the service-cost of client j in the solutions S and O respectively. $N_S(s)$ denotes the clients served by facility s in the solution S. Similarly $N_O(o)$ denotes the clients served by facility o in solution O. N_s^o denotes the set of clients served by facility s in solution S and by facility o in solution O.

The presence of the `add` operation ensures that the total service cost of the clients in any locally optimum solution is at most the total cost of the optimum solution [2]. Hence in this paper we only consider the problem of bounding the facility cost of a locally optimum solution which we show is no more than 2 times the cost of an optimum solution.

3 Bounding the Facility Costs

Let S denote the locally optimum solution obtained. For the rest of this section we assume that the sets S and O are disjoint.

We will associate a weight, $\mathtt{wt} : C \to [0..1]$, with each client which satisfies the following properties.

1. For a client $j \in C$ let $\sigma(j)$ be the facility which serves j in solution S. Then
$$\mathtt{wt}(j) \leq \min\left(1, \frac{U - |N_S(\sigma(j))|}{|N_S(\sigma(j))|}\right).$$
Let $\mathtt{init\text{-}wt}(j)$ denote the quantity on the right of the above inequality. Since $|N_S(\sigma(j))| \leq U$, we have that $0 \leq \mathtt{init\text{-}wt}(j) \leq 1$.
2. For all $o \in O$ and $s \in S$, $\mathtt{wt}(N_s^o) \leq \mathtt{wt}(N_O(o))/2$. Here for $X \subseteq C$, $\mathtt{wt}(X)$ denotes the sum of the weights of the clients in X.

To determine $\texttt{wt}(j)$ so that these two properties are satisfied we start by assigning $\texttt{wt}(j) = \texttt{init-wt}(j)$. However, this assignment might violate the second property. A facility $s \in S$ *captures* a facility $o \in O$ if $\texttt{init-wt}(N_s^o) > \texttt{init-wt}(N_O(o))/2$. Note that at most one facility in S can capture a facility o. If s does not capture o then for all $j \in N_s^o$ define $\texttt{wt}(j) = \texttt{init-wt}(j)$. However if s captures o then for all $j \in N_s^o$ define $\texttt{wt}(j) = \alpha \cdot \texttt{init-wt}(j)$ where $\alpha < 1$ is such that $\texttt{wt}(N_s^o) = \texttt{wt}(N_O(o))/2$.

For a facility $o \in O$ we define a fractional assignment $\pi_o : N_O(o) \times N_O(o) \rightarrow \Re^+$ with the following properties.

separation. $\pi_o(j, j') > 0$ only if j and j' are served by different facilities in S.
balance. $\sum_{j' \in N_O(o)} \pi_o(j', j) = \sum_{j' \in N_O(o)} \pi_o(j, j') = \texttt{wt}_o(j)$ for all $j \in N_O(o)$.

The fractional assignment π_o can be obtained along the same lines as the mapping in [1]. The individual fractional assignments π_o are extended to a fractional assignment over all clients, $\pi : C \times C \rightarrow \Re^+$ in the obvious way — $\pi(j, j') = \pi_o(j, j')$ if $j, j' \in N_O(o)$ and is 0 otherwise.

To bound the facility cost of a facility $s \in S$ we will close the facility and assign the clients served by s to other facilities in S and, may be, some facility in O. The reassignment of the clients served by s to the facilities in S is done using the fractional assignment π. Thus if client j is served by s in the solution S and $\pi(j, j') > 0$ then we assign a $\pi(j, j')$ fraction of j to the facility $\sigma(j')$. Note that

1. $\sigma(j') \neq s$ and this follows from the separation property of π.
2. j is reassigned to the facilities in S to a total extent of $\texttt{wt}(j)$ (balance property).
3. A facility $s' \in S, s' \neq s$ would get some additional clients. The total extent to which these additional clients are assigned to s' is at most $\texttt{wt}(N_S(s'))$ (balance property). Since

$$\texttt{wt}(N_S(s')) \leq \texttt{init-wt}(N_S(s')) \leq U - |N_S(s')|,$$

   the total extent to which clients are assigned to s' is at most U.

Let $\Delta(s)$ denote the increase in the service-cost of the clients served by s due to the above reassignment.

Lemma 1. $\sum_{s \in S} \Delta(s) \leq \sum_{j \in C} 2 O_j \texttt{wt}(j)$

Proof. Let $\pi(j, j') > 0$. When the facility $\sigma(j)$ is closed and $\pi(j, j')$ fraction of client j assigned to facility $\sigma(j')$, the increase in service cost is $\pi(j, j')(c_{j,\sigma(j')} - c_{j,\sigma(j)})$. Since $c_{j,\sigma(j')} \leq O_j + O_{j'} + S_{j'}$ we have

$$\begin{aligned}
\sum_{s \in S} \Delta(s) &= \sum_{j,j' \in C} \pi(j, j')(c_{j,\sigma(j')} - c_{j,\sigma(j)}) \\
&\leq \sum_{j,j' \in C} \pi(j, j')(O_j + O_{j'} + S_{j'} - S_j) \\
&= 2 \sum_{j \in C} O_j \texttt{wt}(j)
\end{aligned}$$

where the last equality follows from the balance property. □

If $\texttt{wt}(j) < 1$ then some part of j remains unassigned. The quantity $1-\texttt{wt}(j)$ is the *residual weight* of client j and is denoted by $\texttt{res-wt}(j)$. Clearly $0 \leq \texttt{res-wt}(j) \leq 1$. Note that

1. If we close facility $s \in S$ and assign the residual weight of all clients served by s to a facility $o \in O - S$ then the total extent to which clients are assigned to o equals $\texttt{res-wt}(N_S(s))$ which is less than U.
2. The service cost of a client, j, which is assigned to o would increase by $c_{j,o} - c_{j,s}$. Let
$$c_{s,o} = \max_{j \in C}(c_{j,o} - c_{j,s})$$
denote the maximum possible increase in service cost of a client when it is assigned to o instead of s. Since service costs satisfy the metric property we have
$$c_{s,o} \leq \min_{j \in C}(S_j + O_j).$$
3. The total increase in service cost of all clients in $N_S(s)$ which are assigned (partly) to o is at most $c_{s,o}\texttt{res-wt}(N_S(s))$.

Let $\langle s, o \rangle$ denote the swapping of facilities s, o and the reassignment of clients served by s to facilities in $S \cup \{o\}$ as discussed above. Since S is a locally optimum we have
$$f_o - f_s + c_{s,o}\texttt{res-wt}(N_S(s)) + \Delta(s) \geq 0. \tag{1}$$
The above inequalities are written for every pair $(s, o), s \in S, o \in O$. We take a linear combination of these inequalities with the inequality corresponding to $\langle s, o \rangle$ having a weight $\lambda_{s,o}$ in the combination to get
$$\sum_{s,o} \lambda_{s,o} f_o - \sum_{s,o} \lambda_{s,o} f_s + \sum_{s,o} \lambda_{s,o} c_{s,o}\texttt{res-wt}(N_S(s)) + \sum_{s,o} \lambda_{s,o} \Delta(s) \geq 0. \tag{2}$$
where
$$\lambda_{s,o} = \frac{\texttt{res-wt}(N_s^o)}{\texttt{res-wt}(N_S(s))}$$
and is 0 if $\texttt{res-wt}(N_S(s)) = 0$. Let S' be the subset of facilities in the solution S for which $\texttt{res-wt}(N_S(s)) = 0$. A facility $s \in S'$ can be deleted from S and its clients reassigned completely to the other facilities in S. This implies
$$-f_s + \Delta(s) \geq 0$$
We write such an inequality for each $s \in S'$ and add them to inequality 2.

Note that for all $s \in S - S'$, $\sum_o \lambda_{s,o} = 1$. This implies that
$$\sum_{s \in S'} f_s + \sum_{s,o} \lambda_{s,o} f_s = \sum_s f_s \tag{3}$$
and
$$\sum_{s \in S'} \Delta(s) + \sum_{s,o} \lambda_{s,o} \Delta(s) = \sum_s \Delta(s) \leq \sum_{j \in C} 2O_j\texttt{wt}(j) \tag{4}$$
However, the reason for defining $\lambda_{s,o}$ as above is to ensure the following property.

Lemma 2. $\sum_{s,o} \lambda_{s,o} c_{s,o} \texttt{res-wt}(N_S(s)) \leq \sum_{j \in C} \texttt{res-wt}(j)(O_j + S_j)$

Proof. The left hand side in the inequality is $\sum_{s,o} c_{s,o} \texttt{res-wt}(N_s^o)$. Since for each client $j \in N_s^o$, $c_{s,o} \leq O_j + S_j$ we have

$$\begin{aligned} c_{s,o}\texttt{res-wt}(N_s^o) &= \sum_{j \in N_S^o} c_{s,o}\texttt{res-wt}(j) \\ &\leq \sum_{j \in N_S^o} \texttt{res-wt}(j)(O_j + S_j) \end{aligned}$$

which, when summed over all s and o implies the Lemma. □

Incorporating equations (3), (4) and Lemma 2 into inequality (2) we get

$$\begin{aligned} \sum_s f_s &\leq \sum_{s,o} \lambda_{s,o} f_o + \sum_{j \in C} \texttt{res-wt}(j)(O_j + S_j) + \sum_{j \in C} 2O_j \texttt{wt}(j) \\ &= \sum_{s,o} \lambda_{s,o} f_o + 2\sum_{j \in C} O_j + \sum_{j \in C} \texttt{res-wt}(j)(S_j - O_j) \qquad (5) \end{aligned}$$

We now need to bound the number of times a facility of the optimum solution may be opened.

Lemma 3. *For all $o \in O$, $\sum_s \lambda_{s,o} \leq 2$.*

Proof. We begin with the following observations.

1. For all s, o, $\lambda_{s,o} \leq 1$.
2. Let $I \subseteq S$ be the facilities s such that $|N_S(s)| \leq U/2$ and s does not capture o. Let $s \in I$ and $j \in N_s^o$. Note that $\texttt{wt}(j) = \texttt{init-wt}(j) = 1$ and so $\texttt{res-wt}(j) = 0$. This implies that $\texttt{res-wt}(N_s^o) = 0$ and so for all $s \in I$, $\lambda_{s,o} = 0$.

Thus we only need to show that $\sum_{s \notin I} \lambda_{s,o} \leq 2$.

We first consider the case when o is not captured by any $s \in S$. Let s be a facility not in I which does not capture o. For $j \in N_s^o$,

$$\texttt{res-wt}(j) = 1 - \texttt{wt}(j) = 1 - \texttt{init-wt}(j) = 2 - \frac{U}{|N_S(s)|}.$$

However, for $j \in N_S(s)$ we have that

$$\texttt{res-wt}(j) = 1 - \texttt{wt}(j) \geq 1 - \texttt{init-wt}(j) = 2 - \frac{U}{|N_S(s)|}.$$

Therefore $\lambda_{s,o} \leq |N_s^o|/|N_S(s)|$ and hence

$$\sum_s \lambda_{s,o} = \sum_{s \notin I} \lambda_{s,o} \leq \sum_{s \notin I} \frac{|N_s^o|}{|N_S(s)|} \leq \sum_{s \notin I} \frac{|N_s^o|}{U/2} \leq \frac{|N_O(o)|}{U/2} \leq 2.$$

We next consider the case when o is captured by $s' \in S$. This implies

$$\begin{aligned}
\texttt{init-wt}(N^o_{s'}) &\geq \sum_{s \neq s'} \texttt{init-wt}(N^o_s) \\
&\geq \sum_{s \notin I \cup \{s'\}} \texttt{init-wt}(N^o_s) \\
&= \sum_{s \notin I \cup \{s'\}} |N^o_s| \frac{U - |N_S(s)|}{|N_S(s)|} \\
&= \sum_{s \notin I \cup \{s'\}} \left(U \frac{|N^o_s|}{|N_S(s)|} - |N^o_s| \right)
\end{aligned}$$

Since $\texttt{init-wt}(N^o_{s'}) \leq |N^o_{s'}|$ rearranging we get,

$$\sum_{s \notin I \cup \{s'\}} \frac{|N^o_s|}{|N_S(s)|} \leq \sum_{s \notin I} \frac{|N^o_s|}{U} \leq 1.$$

Now

$$\sum_{s \notin I \cup \{s'\}} \lambda_{s,o} \leq \sum_{s \notin I \cup \{s'\}} \frac{|N^o_s|}{|N_S(s)|} \leq 1$$

and since $\lambda_{s',o} \leq 1$ we have

$$\sum_s \lambda_{s,o} = \sum_{s \notin I} \lambda_{s,o} \leq 2.$$

This completes the proof. □

Incorporating Lemma 3 into inequality (5) we get

$$\sum_s f_s \leq 2 \left(\sum_o f_o + \sum_{j \in C} O_j \right) + \sum_{j \in C} \texttt{res-wt}(j)(S_j - O_j)$$

Note that $\sum_{j \in C} \texttt{res-wt}(j)(S_j - O_j)$ is at most $\sum_{j \in C}(S_j - O_j)$ which in turn can be bounded by $\sum_o f_o$ by considering the operation of adding facilities in the optimum solution. This, however, would lead to a bound of $3\sum_o f_o + 2\sum_{j \in C} O_j$ on the facility cost of our solution.

The key to obtaining a sharper bound on the facility cost of our solution is the observation that in the swap $\langle s, o \rangle$ facility o gets only $\texttt{res-wt}(N_S(s))$ clients and so can accommodate an additional $U - \texttt{res-wt}(N_S(s))$ clients. Since we need to bound $\sum_{j \in C} \texttt{res-wt}(j)(S_j - O_j)$, we assign, the clients in $N_O(o)$ to facility o in the ratio of their residual weights. Thus client j would be assigned to an extent $\beta_{s,o}\texttt{res-wt}(j)$ where

$$\beta_{s,o} = \min \left(1, \frac{U - \texttt{res-wt}(N_S(s))}{\texttt{res-wt}(N_O(o))} \right).$$

$\beta_{s,o}$ is defined so that o gets at most U clients. Let $\Delta'(s,o)$ denote the increase in service cost of the clients of $N_O(o)$ due to this reassignment. Hence

$$\Delta'(s,o) = \beta_{s,o} \sum_{j \in N_O(o)} \texttt{res-wt}(j)(O_j - S_j).$$

The inequality (1) corresponding to the swap $\langle s, o\rangle$ would now get an additional term $\Delta'(s,o)$ on the left. Hence the term $\sum_{s,o} \lambda_{s,o}\Delta'(s,o)$ would appear on the left in inequality (2) and on the right in inequality (5). To bound this term note that

$$\begin{aligned}\sum_s \lambda_{s,o}\Delta'(s,o) &= \sum_s \lambda_{s,o}\beta_{s,o} \sum_{j \in N_O(o)} \texttt{res-wt}(j)(O_j - S_j)\\ &= \left(\sum_s \lambda_{s,o}\beta_{s,o}\right) \sum_{j \in N_O(o)} \texttt{res-wt}(j)(O_j - S_j).\end{aligned}$$

If $\sum_s \lambda_{s,o}\beta_{s,o} > 1$ then we reduce some $\beta_{s,o}$ so that the sum is exactly 1. On the other hand if $\sum_s \lambda_{s,o}\beta_{s,o} = 1 - \gamma_o$, $\gamma_o > 0$, then we take the inequalities corresponding to the operation of adding the facility $o \in O$

$$f_o + \sum_{j \in N_O(o)} \texttt{res-wt}(j)(O_j - S_j) \geq 0 \tag{6}$$

and add these to inequality (2) with a weight γ_o. Hence the total increase in the left hand side of inequality (2) is

$$\begin{aligned}&\sum_{s,o} \lambda_{s,o}\Delta'(s,o) + \sum_o \gamma_o \left(f_o + \sum_{j \in N_O(o)} \texttt{res-wt}(j)(O_j - S_j)\right)\\ &= \sum_o \sum_{j \in N_O(o)} (1-\gamma_o)\texttt{res-wt}(j)(O_j - S_j)\\ &+ \sum_o \gamma_o f_o + \sum_o \sum_{j \in N_O(o)} \gamma_o \texttt{res-wt}(j)(O_j - S_j)\\ &= \sum_o \sum_{j \in N_O(o)} \texttt{res-wt}(j)(O_j - S_j) + \sum_o \gamma_o f_o\\ &= \sum_{j \in C} \texttt{res-wt}(j)(O_j - S_j) + \sum_o \gamma_o f_o\end{aligned}$$

and so inequality (5) now becomes

$$\begin{aligned}\sum_s f_s \leq \sum_o \sum_s \lambda_{s,o} f_o + 2\sum_{j \in C} O_j + \sum_o \gamma_o f_o\\ + \sum_{j \in C} \texttt{res-wt}(j)(S_j - O_j) + \sum_{j \in C} \texttt{res-wt}(j)(O_j - S_j)\end{aligned}$$

$$
\begin{aligned}
&= \sum_o \left(\gamma_o + \sum_s \lambda_{s,o}\right) f_o + 2\sum_{j\in C} O_j \\
&= \sum_o \left(1 + \sum_s \lambda_{s,o}(1-\beta_{s,o})\right) f_o + 2\sum_{j\in C} O_j \\
&\le 2\left(\sum_o f_o + \sum_{j\in C} O_j\right)
\end{aligned}
$$

where the last inequality follows from the following Lemma.

Lemma 4. $\sum_s \lambda_{s,o}(1-\beta_{s,o}) \le 1$.

Proof. Since $\texttt{res-wt}(N_O(o)) \le |N_O(o)| \le U$ we have

$$
\begin{aligned}
\beta_{s,o} &= \min\left(1, \frac{U - \texttt{res-wt}(N_S(s))}{\texttt{res-wt}(N_O(o))}\right) \\
&\ge \min\left(1, 1 - \frac{\texttt{res-wt}(N_S(s))}{\texttt{res-wt}(N_O(o))}\right) \\
&= 1 - \frac{\texttt{res-wt}(N_S(s))}{\texttt{res-wt}(N_O(o))}
\end{aligned}
$$

Hence

$$\sum_s \lambda_{s,o}(1-\beta_{s,o}) \le \sum_s \frac{\texttt{res-wt}(N_s^o)}{\texttt{res-wt}(N_O(o))} = 1.$$

□

This completes the proof of the following theorem.

Theorem 1. *The total cost of open facilities in any locally optimum solution is at most twice the cost of an optimum solution.*

4 When $S \cap O \neq \phi$

We now consider the case when $S \cap O \neq \phi$. We construct a bipartite graph, G, on the vertex set $C \cup F$ as in [2]. Every client $j \in C$ has an edge from the facility $\sigma(j) \in S$ and an edge to the facility $\tau(j) \in O$, where $\tau(j)$ is the facility in O serving client j. Thus each client has one incoming and one outgoing edge. A facility $s \in S$ has $|N_S(s)|$ outgoing edges and a facility $o \in O$ has $|N_O(o)|$ incoming edges.

Decompose the edges of G into a set of maximal paths, $\mathcal{P}$, and cycles, $\mathcal{C}$. Note that all facilities on a cycle are from $S \cap O$. Consider a maximal path, $p \in \mathcal{P}$ which starts at a vertex $s \in S$ and ends at a vertex $o \in O$. Let head(p) denote the client served by s on this path and tail(p) be the client served by o on this path. Let $s = s_0, j_0, s_1, j_1, \ldots, s_k, j_k, o$ be the sequence of vertices on this path.

Note that $\{s_1, s_2, \ldots, s_k\} \subseteq S \cap O$. A *shift* along this path is a reassignment of clients so that j_i which was earlier assigned to s_i is now assigned to s_{i+1} where $s_{k+1} = o$. As a consequence of this shift, facility s serves one less client while facility o serves one more client. Let shift(p) denote the increase in service cost due to a shift along the path p. Then

$$\text{shift}(p) = \sum_{c \in C \cap p} O_c - S_c.$$

We can similarly define a shift along a cycle. The increase in service cost equals the sum of $O_j - S_j$ for all clients j in the cycle and since the assignment of clients to facilities is done optimally in our solution and in the global optimum this sum is zero. Thus

$$\sum_{j \in \mathcal{C}} O_j - S_j = 0.$$

Consider the operation of adding a facility $o \in O$. We shift along all paths which end at o. The increase in service cost due to these shifts equals the sum of $O_j - S_j$ for all clients j on these paths and this quantity is at least $-f_o$.

$$\sum_{j \in \mathcal{P}} O_j - S_j \geq - \sum_{o \in O} f_o.$$

Thus

$$\sum_{j \in C} O_j - S_j = \sum_{j \in \mathcal{P}} O_j - S_j + \sum_{j \in \mathcal{C}} O_j - S_j \geq - \sum_{o \in O} f_o$$

which implies that the service cost of S is bounded by $\sum_{o \in P} f_o + \sum_{j \in C} O_j$.

To bound the cost of facilities in $S - O$ we only need the paths that start from a vertex in $S - O$. Hence we throw away all cycles and all paths that start at a facility in $S \cap O$; this is done by removing all clients on these cycles and paths. Let $\mathcal{P}$ denote the remaining paths and C the remaining clients. Every client in C either belongs to a path which ends in $S \cap O$ (*transfer* path) or to a path which ends in $O - S$ (*swap* path). Let $\mathcal{T}$ denote the set of transfer paths and $\mathcal{S}$ the set of swap paths.

Let N_s^o be the set of paths that start at $s \in S$ and end at $o \in O$. Define

$$N_S(s) = \cup_{o \in O - S} N_s^o.$$

Note that we do not include the transfer paths in the above definition. Similarly for all $o \in O$ define

$$N_O(o) = \cup_{s \in S - O} N_s^o.$$

Just as we defined the `init-wt`, `wt` and `res-wt` of a client, we can define the `init-wt`, `wt` and `res-wt` of a swap path. Thus for a path p which starts from $s \in S - O$ we define

$$\texttt{init-wt}(p) = \min\left(1, \frac{U - |N_S(s)|}{|N_S(s)|}\right).$$

The notion of *capture* remains the same and we reduce the initial weights on the paths to obtain their weights. Thus $\texttt{wt}(p) \leq \texttt{init-wt}(p)$ and for every $s \in S$ and $o \in O$, $\texttt{wt}(N_s^o) \leq \texttt{wt}(N_O(o))/2$. For every $o \in O - S$ we define a fractional mapping $\pi_o : N_O(o) \times N_O(o) \rightarrow \Re^+$ such that

separation. $\pi_o(p, p') > 0$ only if p and p' start at different facilities in $S - O$.
balance. $\sum_{p' \in N_O(o)} \pi_o(p', p) = \sum_{p' \in N_O(o)} \pi_o(p, p') = \texttt{wt}_o(p)$ for all $p \in N_O(o)$.

This fractional mapping can be constructed in the same way as done earlier. The way we use this fractional mapping, π, will differ slightly. When facility s is closed, we will use π to partly reassign the clients served by s in the solution S to other facilities in S. If p is a path starting from s and $\pi(p, p') > 0$, then we shift along p and the client tail(p) is assigned to s', where s' is the facility from which p' starts. This whole operation is done to an extent of $\pi(p, p')$.

Let $\Delta(s)$ denote the total increase in service cost due to the reassignment of clients on all swap paths starting from s. Define the *length* of the path p as

$$\text{length}(p) = \sum_{c \in C \cap p} O_c + S_c.$$

Then

$$\begin{aligned}\sum_s \Delta(s) &\leq \sum_s \sum_{p \in N_S(s)} \sum_{p' \in \mathcal{P}} \pi(p, p')(\text{shift}(p) + \text{length}(p')) \\ &= \sum_{p \in \mathcal{S}} \texttt{wt}(p)(\text{shift}(p) + \text{length}(p))\end{aligned}$$

As a result of the above reassignment a facility $s' \in S - O, s' \neq s$ might get additional clients whose "number" is at most $\texttt{wt}(N_S(s'))$. Note that this is less than $\texttt{init-wt}(N_S(s'))$ which is at most $U - |N_S(s')|$. The number of clients s' was serving equals $|N_S(s')| + |T(s')|$ where $T(s')$ is the set of transfer paths starting from s'. This implies that the total number of clients s' would have after the reassignment could exceed U. To prevent this violation of our capacity constraint, we also perform a shift along these transfer paths. To determine when the shift should be done, we define a mapping $t : N_S(s') \times T(s') \rightarrow \Re^+$ such that

1. For all $q \in T(s')$, $\sum_{p \in N_S(s')} t(p, q) \leq 1$.
2. For all $p \in N_S(s')$, $\sum_{q \in T(s')} t(p, q) \leq \texttt{wt}(p)$.
3. $\sum_{p,q} t(p, q) = \min\left(|T(s')|, \texttt{wt}(N_S(s'))\right)$.

Now suppose s' gets an additional client, say tail(p), to an extent of $\pi(p, p')$, where $p' \in N_S(s')$. Then for all paths $q \in T(s')$ for which $t(p', q) > 0$, we would shift along path q to an extent $\pi(p, p')t(p', q)/\sum_{q \in T(s')} t(p', q)$. The mapping ensures that

1. The total extent to which we will shift along a path $q \in T(s')$ is at most 1. This in turn implies that we do not violate the capacity of any facility

in $S \cap O$. This is because, if there are t transfer paths ending at a facility $o \in S \cap O$ then o serves t more clients in solution O than in S. Hence, in solution S, o serves at most $U - t$ clients. Since the total extent to which we could shift along a transfer path ending at o is 1, even if we were to perform shift along all transfer paths ending in o, the capacity of o in our solution S would not be violated.

2. The capacity constraint of no facility in $S - O$ is violated. If a facility $s' \in S - O$ gets an additional x clients as a result of reassigning the clients of some facility $s \neq s'$, then it would also lose some clients, say y, due to the shifts along the transfer paths. From property 3 of the mapping t it follows that
$$x - y \leq \texttt{wt}(N_S(s')) - |T(s')| \leq U - |N_S(s')| - |T(s')|.$$
Since, initially, s' was serving $|N_S(s')| + |T(s')|$ clients, the total number of clients that s' is serving after the reassignment is at most U.

Consider a transfer path, q, starting from s. We would shift once along path q when we close facility s. We would also be shifting along q to an extent of $\sum_{p' \in N_S(s)} t(p', q)$ which is at most 1. Let $\Delta'(s)$ denote the total increase in service cost due to shifts on all transfer paths starting from s. Then

$$\Delta'(s) \leq 2 \sum_{q \in T(s)} \text{shift}(q) \tag{7}$$

For a swap path p, define $\texttt{res-wt}(p) = 1 - \texttt{wt}(p)$. If a client j belongs to a swap path p then define $\texttt{wt}(j) = \texttt{wt}(p)$ and $\texttt{res-wt}(j) = \texttt{res-wt}(p)$. When facility s is closed, a client j served by s has been assigned to an extent $\texttt{wt}(j)$ to other facilities in S. We will be assigning the remaining part of j to a facility $o \in O - S$ that will be opened when s is closed. Hence the total number of clients that will be assigned to o is $\texttt{res-wt}(N_S(s))$ which is less than U. The increase in service cost due to this reassignment is at most $c_{s,o}\texttt{res-wt}(N_S(s))$. The remaining available capacity of o is utilized by assigning each client $j \in N_O(o)$ to an extent $\beta_{s,o}\texttt{res-wt}(j)$, where $\beta_{s,o}$ is defined as before. This assignment is actually done by shifting along each path, $p \in N_O(o)$, by an extent $\beta_{s,o}\texttt{res-wt}(p)$. As done earlier, the inequality corresponding to the swap $\langle s, o \rangle$ is counted to an extent $\lambda_{s,o}$ in the linear combination.

Just as in Lemma 2 we have

$$\sum_{s,o} \lambda_{s,o} c_{s,o} \texttt{res-wt}(N_S(s)) \leq \sum_{p \in \mathcal{S}} \texttt{res-wt}(p) \text{length}(p).$$

Lemma 3 continues to hold and so does Lemma 4. As before, we might have to add facility $o \in O - S$, shift each path $p \in N_O(o)$ by an extent $\texttt{res-wt}(p)$ and add a γ_o multiple of this inequality to the linear combination. Putting everything together we get

$$\begin{aligned}
\sum_{s\in S-O} f_s &\le 2\sum_{o\in O-S} f_o + \sum_{p\in\mathcal{S}} \mathtt{wt}(p)(\mathrm{shift}(p)+\mathrm{length}(p)) \\
&\quad + \sum_{p\in\mathcal{S}} \mathtt{res\text{-}wt}(p)(\mathrm{length}(p)+\mathrm{shift}(p)) + 2\sum_{p\in\mathcal{T}} \mathrm{shift}(p) \\
&= 2\sum_{o\in O-S} f_o + \sum_{p\in\mathcal{S}} (\mathrm{shift}(p)+\mathrm{length}(p)) + 2\sum_{p\in\mathcal{T}} \mathrm{shift}(p) \\
&\le 2\left(\sum_{o\in O-S} f_o + \sum_{j\in C} O_j\right)
\end{aligned}$$

5 A Tight Example

Our tight example consists of r facilities in the optimum solution O, r facilities in the locally optimum solution S and rU clients. The facilities are $F = O \cup S$. Since, no facility can serve more than U clients, each facility in S and O serves exactly U clients. Our instance has the property that a facility in O and a facility in S share at most one client.

We can view our instance as a set-system — the set of facilities O is the ground set and for every facility $s \in S$ we have a subset X_s of this ground set. $o \in X_s$ iff there is a client which is served by s in the solution S and by o in the solution O. This immediately implies that each element of the ground set is in exactly U sets and that each set is of size exactly U.

A *triangle* in the set-system is a collection of 3 elements, o_1, o_2, o_3 and 3 sets $X_{s_1}, X_{s_2}, X_{s_3}$ such that o_i is not in X_{s_i} but belongs to the other two sets. An important property of our instance is that the corresponding set-system has no triangles.

We now show how to construct a set system with the three properties mentioned above. With every $o \in O$ we associate a distinct point $x^o = (x_1^o, x_2^o, \ldots x_U^o)$ in a U-dimensional space where for all i, $x_i^o \in \{1, 2, 3, \ldots, U\}$. For every choice of coordinate i, $1 \le i \le U$ we form U^{U-1} sets, each of which contains all points differing only in coordinate i. Thus the total number of sets we form is $r = U^U$ which is the same as the number of points. Each set can be viewed as a line in U-dimensional space. To see that this set system satisfies all the three properties note that each line contains U points and each point is on exactly U lines. Further, we do not have 3 points and 3 lines such that each line includes two points and excludes one.

We now define the facility and the service costs. For a facility $o \in O$, $f_o = 2U$ while for facility $s \in S$, $f_s = 6U - 6$. For a client $j \in N_s^o$, we have $c_{s,j} = 3$ and $c_{o,j} = 1$. All other service costs are given by the metric property.

Since the service cost of each client in O is 1 and the facility cost of each facility in O is $2U$, we have $\mathrm{cost}(O) = 3U^{U+1}$. Similarly, $\mathrm{cost}(S) = (3 - 2/U)3U^{U+1}$ and hence $\mathrm{cost}(S) = (3 - 2/U)\mathrm{cost}(O)$. We now need to prove that S is indeed a locally optimum solution with respect to the local search operations of add, delete and swap.

Adding a facility $o \in O$ to the solution S, would incur an opening cost of $2U$. The optimum assignment would reassign only the clients in $N_o(O)$, and all these are assigned to o. The reduction in the service cost due to this is exactly $2U$ which is offset by the increase in the facility cost. Hence the cost of the solution does not improve.

If we delete a facility in the solution S, the solution is no longer feasible since the total capacity of the facilities is now $U^{U+1} - U$ and the number of clients is U^{U+1}.

Now, consider swapping a facility $s \in S$ with a facility $o \in O$. The net decrease in the facility cost is $4U - 6$. One can show that the new optimum assignment of clients to facilities would reassign only clients in $N_s(S)$ and all these are assigned to o. Since $N_s^o \leq 1$, we have $U - 1$ clients in $N_s(S)$ whose service cost would increase from 3 to 7. The client in N_s^o would see a decrease in service cost by 2. The net increase in service cost is $4U - 6$ which is exactly equal to the decrease in facility cost. Hence, swapping any pair of facilities $s \in S$ and $o \in O$ does not improve the solution.

References

1. Arya, V., Garg, N., Khandekar, R., Meyerson, A., Munagala, K., Pandit, V.: Local search heuristics for k-median and facility location problems. SIAM J. Comput. 33(3), 544–562 (2004)
2. Chudak, F., Williamson, D.P.: Improved approximation algorithms for capacitated facility location problems. Math. Program. 102(2), 207–222 (2005)
3. Korupolu, M.R., Plaxton, C.G., Rajaraman, R.: Analysis of a local search heuristic for facility location problems. J. Algorithms 37(1), 146–188 (2000)
4. Mahdian, M., Pál, M.: Universal facility location. In: Di Battista, G., Zwick, U. (eds.) ESA 2003. LNCS, vol. 2832, pp. 409–421. Springer, Heidelberg (2003)
5. Mahdian, M., Ye, Y., Zhang, J.: A 2-approximation algorithm for the soft-capacitated facility location problem. In: Arora, S., Jansen, K., Rolim, J.D.P., Sahai, A. (eds.) RANDOM 2003 and APPROX 2003. LNCS, vol. 2764, pp. 129–140. Springer, Heidelberg (2003)
6. Pál, M., Tardos, É., Wexler, T.: Facility location with nonuniform hard capacities. In: FOCS '01: Proceedings of the 42nd IEEE symposium on Foundations of Computer Science, Washington, DC, USA, p. 329. IEEE Computer Society, Los Alamitos (2001)
7. Zhang, J., Chen, B., Ye, Y.: A multiexchange local search algorithm for the capacitated facility location problem. Math. Oper. Res. 30(2), 389–403 (2005)

Secretary Problems via Linear Programming

Niv Buchbinder[1], Kamal Jain[2], and Mohit Singh[3]

[1] Microsoft Research, New England, Cambridge, MA
[2] Microsoft Research, Redmond, WA, USA
[3] McGill University, Montreal, Canada

Abstract. In the classical secretary problem an employer would like to choose the best candidate among n competing candidates that arrive in a random order. This basic concept of n elements arriving in a random order and irrevocable decisions made by an algorithm have been explored extensively over the years, and used for modeling the behavior of many processes. Our main contribution is a new linear programming technique that we introduce as a tool for obtaining and analyzing mechanisms for the secretary problem and its variants. The linear program is formulated using judiciously chosen variables and constraints and we show a one-to-one correspondence between mechanisms for the secretary problem and feasible solutions to the linear program. Capturing the set of mechanisms as a linear polytope holds the following immediate advantages.

- Computing the optimal mechanism reduces to solving a linear program.
- Proving an upper bound on the performance of any mechanism reduces to finding a feasible solution to the dual program.
- Exploring variants of the problem is as simple as adding new constraints, or manipulating the objective function of the linear program.

We demonstrate these ideas by exploring some natural variants of the secretary problem. In particular, using our approach, we design optimal secretary mechanisms in which the probability of selecting a candidate at any position is equal. We refer to such mechanisms as *incentive compatible* and these mechanisms are motivated by the recent applications of secretary problems to online auctions. We also show a family of linear programs which characterize all mechanisms that are allowed to choose J candidates and gain profit from the K best candidates. We believe that linear programming based approach may be very helpful in the context of other variants of the secretary problem.

1 Introduction

In the classical secretary problem an employer would like to choose the best candidate among n competing candidates. The candidates are assumed to arrive in a random order. After each interview, the position of the interviewee in the total order is revealed vis-á-vis already interviewed candidates. The interviewer has to decide, irrevocably, whether to accept the candidate for the position or to reject the candidate. The objective in the basic problem is to accept the best candidate with high probability. A mechanism used for choosing the best candidate is to interview the first n/e candidates for the purpose of evaluation, and then hire the first candidate that is better than all previous candidates. Analysis of the mechanism shows that it hires the best candidate with probability $1/e$ and that it is optimal [8,18].

This basic concept of n elements arriving in a random order and irrevocable decisions made by an algorithm have been explored extensively over the years. We refer the reader to the survey by Ferguson [9] on the historical and extensive work on different variants of the secretary problem. Recently, there has been a interest in the secretary problem with its application to the online auction problem [13,3]. This has led to the study of variants of the secretary problem which are motivated by this application. For example, [15] studied a setting in which the mechanism is allowed to select multiple candidates and the goal is to maximize the expected profit. Imposing other combinatorial structure on the set of selected candidates, for example, selecting elements which form an independent set of a matroid [4], selecting elements that satisfy a given knapsack constraint [2], selecting elements that form a matching in a graph or hypergraph [16], have also been studied. Other variants include when the profit of selecting a secretary is discounted with time [5]. Therefore, finding new ways of abstracting, as well as analyzing and designing algorithms, for secretary type problems is of major interest.

1.1 Our Contributions

Our main contribution is a new linear programming technique that we introduce as a tool for obtaining and analyzing mechanisms for various secretary problems. We introduce a linear program with judiciously chosen variables and constraints and show a one-to-one correspondence between mechanisms for the secretary problem and feasible solutions to the linear program. Obtaining a mechanism which maximizes a certain objective therefore reduces to finding an optimal solution to the linear program. We use linear programming duality to give a simple proof that the mechanism obtained is optimal. We illustrate our technique by applying it to the classical secretary problem and obtaining a simple proof of optimality of the $\frac{1}{e}$ mechanism [8] in Section 2.

Our linear program for the classical secretary problem consists of a single constraint for each position i, bounding the probability that the mechanism may select the ith candidate. Despite its simplicity, we show that such a set of constraints suffices to correctly capture all possible mechanisms. Thus, optimizing over this polytope results in the optimal mechanism. The simplicity and the tightness of the linear programming formulation makes it flexible and applicable to many other variants. Capturing the set of mechanisms as a linear polytope holds the following immediate advantages.

- Computing the optimal mechanism reduces to solving a linear program.
- Proving an upper bound on the performance of any mechanism reduces to finding a feasible solution to the dual program.
- Exploring variants of the problem is as simple as adding new constraints, or manipulating the objective function of the linear program.

We next demonstrate these ideas by exploring some natural variants of the secretary problem.

Incentive Compatibility. As discussed earlier, the optimal mechanism for the classical secretary problem is to interview the first n/e candidates for the purpose of evaluation, and then hire the first candidate that is better than all previous candidates. This mechanism suffers from a crucial drawback. The candidates arriving early have an incentive to delay their interview and candidates arriving after the position $\frac{n}{e}+1$ have an incentive to advance their interview. Such a behavior challenges the main assumption of the model that interviewees arrive in a random order. This issue of incentives is of major importance especially since secretary problems have been used recently in the context of online auctions [13,3].

Using the linear programming technique, we study mechanisms that are *incentive compatible.* We call a mechanism for the secretary problem *incentive compatible* if the probability of selecting a candidate at i^{th} position is equal for each position $1 \le i \le n$. Since the probability of being selected in each position is the same, there is no incentive for any interviewee to change his or her position and therefore the interviewee arrives at the randomly assigned position. We show that there exists an incentive compatible mechanism which selects the best candidate with probability $1 - \frac{1}{\sqrt{2}} \approx 0.29$ and that this mechanism is optimal. Incentive compatibility is captured in the linear program by introducing a set of very simple constraints.

Surprisingly, we find that the optimal incentive compatible mechanism sometime selects a candidate who is worse than a previous candidate. To deal with this issue, we call a mechanism *regret-free* if the mechanism only selects candidates which are better than all previous candidates. We show that the best incentive compatible mechanism which is regret free accepts the best candidate with probability $\frac{1}{4}$. Another issue with the optimal incentive compatible mechanism is that it does not always select a candidate. In the classical secretary problem, the mechanism can always pick the last candidate but this solution is unacceptable when considering incentive compatibility. We call a mechanism *must-hire* if it always hires a candidate. We show that there is a must-hire incentive compatible mechanism which hires the best candidate with probability $\frac{1}{4}$. All the above results are optimal and we use the linear programming technique to derive the mechanisms as well as prove their optimality.

In subsequent work [6], we further explore the importance of incentive compatibility in the context of online auctions. In this context, bidders are bidding for an item and may have an incentive to change their position if this may increase their utility. We show how to obtain truthful mechanisms for such settings using underlying mechanisms for secretary type problems. While there are inherent differences in the auction model and the secretary problem, a mechanism for the secretary problem is used as a building block for obtaining an incentive compatible mechanism for the online auction problem.

The J-choice, K-best Secretary Problem. Our LP formulation approach is able to capture a much broader class of secretary problems. We define a most general problem that we call the J-Choice, K-best secretary problem, referred to as the (J, K)-secretary problem. Here, n candidates arrive randomly. The mechanism is allowed to pick up to J different candidates and the objective is to pick as many from the top K ranked candidates. The $(1, 1)$-secretary problem is the classical secretary problem. For any J, K, we provide a linear program which characterizes all mechanisms for the problem by generalizing the linear program for the classical secretary problem.

A sub-class that is especially interesting is the (K, K)-secretary problem, since it is closely related to the problem of maximizing the expected profit in a cardinal version of the problem. In the cardinal version of the problem, n elements that have arbitrary non-negative values arrive in a random order. The mechanism is allowed to pick at most k elements and its goal is to maximize its expected profit. We define a *monotone* mechanism to be an mechanism that, at any position, does not select an element that is t best so far with probability higher than an element that is $t' < t$ best so far. We note that any reasonable mechanism (and in particular the optimal mechanism) is monotone. The following is a simple observation. We omit the proof due to lack of space.

Observation 1. *Let Alg be a monotone mechanism for the (K, K)-secretary problem that is c-competitive. Then the mechanism is also c-competitive for maximizing the expected profit in the cardinal version of the problem.*

Kleinberg [15] gave an asymptotically tight mechanism for the cardinal version of the problem. However, this mechanism is randomized, and also not tight for small values of k. Better mechanisms, even restricted to small values of k, are helpful not only for solving the original problem, but also for improving mechanisms that are based upon them. For example, a mechanism for the secretary knapsack [2] uses a mechanism that is $1/e$ competitive for maximizing the expected profit for small values of k ($k \leq 27$). Analyzing the LP asymptotically for any value n is a challenge even for small value k. However, using our characterization we solve the problem easily for small values k and n which gives an idea on how competitive ratio behaves for small values of k. Our results appear in Table 1. We also give complete asymptotic analysis for the cases of $(1, 2)$, $(2, 1)$-secretary problems.

Table 1. Competitive ratio for Maximizing expected profit. Experimental results for $n = 100$

Number of elements allowed to be picked by the mechanism	Competitive ratio
1	$1/e = 0.368$
2	0.474
3	0.565
4	0.613

1.2 Related Work

The basic secretary problem was introduced in a puzzle by Martin Gardner [11]. Dynkin [8] and Lindley [18] gave the optimal solution and showed that no other strategy can do better (see the historical survey by Ferguson [9] on the history of the problem). Subsequently, various variants of the secretary problem have been studied with different assumptions and requirements [20](see the survey [10]).

More recently, there has been significant work using generalizations of secretary problems as a framework for online auctions [2,3,4,13,15]. Incentives issues in online mechanisms have been studied in several models [1,13,17]. These works designed mechanisms where incentive issues were considered for both value and time strategies. For example, Hajiaghayi et. al. [13] studied a limited supply online auction problem, in which an auctioneer has a limited supply of identical goods and bidders arrive and depart dynamically. In their problem bidders also have a time window which they can lie about.

Our linear programming technique is similar to the technique of factor revealing linear programs that have been used successfully in many different settings [7,12,14,19]. Factor revealing linear program formulates the performance of an algorithm for a problem as a linear program (or sometimes, a more general convex program). The objective function is the approximation factor of the algorithm on the problem. Thus solving the linear program gives an upper bound on the worst case instance which an adversary could choose to maximize/minimize the approximation factor. Our technique, in contrast, captures the information structure of the problem itself by a linear program. We do not apriori assume any algorithm but formulate a linear program which captures every possible algorithm. Thus optimizing our linear program not only gives us an optimal algorithm, but it also proves that the algorithm itself is the best possible.

2 Introducing the Technique: Classical Secretary (and Variants)

In this section, we give a simple linear program which we show characterizes *all* possible mechanisms for the secretary problem. We stress that the LP captures not only thresholding mechanisms, but any mechanism including probabilistic mechanisms. Hence, finding the best mechanism for the secretary problem is equivalent to finding the optimal solution to the linear program. The linear program and its dual appear in Figure 1. The following two lemmas show that

$$
\begin{array}{l|l}
\text{(P)}\ \max\ \frac{1}{n}\cdot\sum_{i=1}^{n} i p_i & \text{(D)}\ \min\ \sum_{i=1}^{n} x_i \\
\text{s.t.} & \text{s.t.} \\
\quad \forall\, 1\le i\le n\ \ i\cdot p_i \le 1-\sum_{j=1}^{i-1} p_j & \quad \forall\, 1\le i\le n\ \ \sum_{j=i+1}^{n} x_j + i x_i \ge i/n \\
\quad \forall\, 1\le i\le n\ \ p_i\ge 0 & \quad \forall\, 1\le i\le n\ \ x_i \ge 0
\end{array}
$$

Fig. 1. Linear program and its Dual for the secretary problem

the linear program exactly characterizes all feasible mechanisms for the secretary problem.

Lemma 1. ***(Mechanism to LP solution)*** *Let π be any mechanism for selecting the best candidate. Let p_i^π denote the probability of selecting the candidate at position i. Then p^π is a feasible solution to the linear program (P), i.e, it satisfies the constraints $p_i^\pi \leq \frac{1}{i}\left(1 - \sum_{j<i} p_j^\pi\right)$ for each $1 \leq i \leq n$. Moreover the objective value $\frac{1}{n}\sum_{i=1}^n ip_i^\pi$ is at least the probability of selecting the best candidate by π.*

Proof. Let p_i^π be the probability in which mechanism π selects candidate i. Any mechanism cannot increase its chances of hiring the best candidate by selecting a candidate that is not the best so far, therefore we may consider only such mechanisms. We now show that p^π satisfies the constraints of linear program.

$$\begin{aligned} p_i^\pi &= Pr[\pi \text{ selects candidate } i| \text{ candidate } i \text{ is best so far}] \\ &\qquad\qquad\qquad\qquad\qquad \cdot Pr[\text{candidate } i \text{ is best so far}] \\ &\leq Pr[\pi \text{ did not select candidates } \{1,\ldots,i-1\}]| \text{ candidate } i \text{ is best so far}] \cdot \frac{1}{i} \end{aligned}$$

However, the probability of selecting candidates 1 to $i-1$ depends only on the relative ranks of these candidates and is independent on whether candidate i is best so far (which can be determined after the mechanism have done its choices regarding candidates 1 to $i-1$). Therefore, we obtain $p_i^\pi \leq \frac{1}{i}(1 - \sum_{j<i} p_j^\pi)$, which proves our claim.

Now we show that the objective function of the linear program is at least the probability with which π accepts the best candidate. Since the mechanism cannot distinguish whether the i^{th} candidate is the best candidate so far or best candidate over all, the probability that the mechanism hires candidate i given that the best candidate is in the ith position equals the probability the mechanism hires candidate i given that the best candidate among candidates 1 to i is in the ith position. Since the ith candidate is best so far with probability $1/i$, the latter probability is at least ip_i^π. Summing over all n positions we get that π hires the best candidate with probability at least $\frac{1}{n}\sum_{i=1}^n ip_i^\pi$.

Lemma 1 shows that the optimal solution to (P) is an upper-bound on the performance of the mechanism. The following lemma shows that every LP solution actually corresponds to a mechanism which performs as well as the objective value of the solution.

Lemma 2. ***(LP solution to Mechanism)*** *Let p_i for $1 \leq i \leq n$ be any feasible LP solution to (P). Then consider the mechanism π which selects the candidate i with probability $\frac{ip_i}{(1-\sum_{j<i} p_j)}$ if candidate i is the best candidate so far and candidate $1,\ldots,i-1$ have not been selected, i.e., the mechanism reaches candidate i. Then π is a mechanism which selects the best candidate with probability $\frac{1}{n}\sum_{i=1}^n ip_i$.*

$$\begin{array}{l|l}
\max\ \frac{1}{n}\cdot\sum_{i=1}^{n} ip_i + q(1-\sum_{i=1}^{n} p_i) & \text{(D)}\ \min\ \sum_{i=1}^{n} x_i + q \\
\text{s.t.} & \text{s.t.} \\
\forall\, 1\le i\le n\ \ i\cdot p_i \le 1-\sum_{j=1}^{i-1} p_j & \forall\, 1\le i\le n\ \ \sum_{j=i+1}^{n} x_j + ix_i \ge i/n - q \\
\forall\, 1\le i\le n\ \ p_i\ge 0 & \forall\, 1\le i\le n\ \ x_i\ge 0
\end{array}$$

Fig. 2. Linear program and its Dual for the rehiring secretary problem

Proof. First, notice that the mechanism is well defined since for any i, $\frac{ip_i}{(1-\sum_{j<i} p_j)} \le 1$. We prove by induction that the probability that the mechanism selects candidate at position i is exactly p_i. The base case is trivial. Assume this is true until $i-1$. At step i, the probability we choose i is the probability that we didn't choose candidates 1 to $i-1$ which is $1-\sum_{j<i} p_j$ times the probability that the current candidate is best so far which is $1/i$ times $\frac{ip_i}{(1-\sum_{j<i} p_j)}$ which is exactly p_i.

The probability of hiring the ith candidate given that the ith candidate is the best candidate is equal the probability of hiring the ith candidate given the ith candidate is the best candidate among candidates 1 to i. Otherwise, it means that the mechanism is able to distinguish between the event of seeing the relative ranks and the absolute ranks which is a contradiction to the definition of the secretary problem. Since the ith candidate is best so far with probability $1/i$, the latter probability equals ip_i (the mechanism hires only the best candidate so far). Summing over all possible position n we get that the mechanism π hires the best candidate with probability $\frac{1}{n}\sum_{i=1}^{n} ip_i$.

Using the above equivalence between LP solutions and the mechanisms, it is easy to show that the optimal mechanism can hire the best candidate with probability of no more than $1/e$. The proof is simply by constructing a feasible solution to the dual linear program.

Lemma 3 ([8]). *No mechanism can hire the best candidate with probability better than* $1/e + o(1)$.

Proof. To prove an upper bound of $1/e+o(1)$ we only need to construct a feasible dual solution to program (D) with value $1/e+o(1)$. Set $x_i = 0$ for each $1 \le i \le \frac{n}{e}$ and $x_i = \frac{1}{n}(1-\sum_{j=i}^{n-1}\frac{1}{j})$ for $\frac{n}{e} < i \le n$. A simple calculation shows that x is feasible and has objective value at most $\frac{1}{e}+o(1)$.

2.1 Allowed to Rehire

One natural extension of the secretary problem is the case when one is allowed to rehire the best secretary at the end with certain probability. That is, suppose that after the interviewer has seen all n candidates, he is allowed to hire the best candidate with certain probability q if no other candidate has been hired. Observe that if $q = 0$, the problem reduces to the classical secretary problem

(Incentive compatible)	(Regret free)	(Must-hire)
$(P1)$ max $\frac{1}{n}\sum_{i=1}^{n} f_i$	$(P2)$ max $\frac{1}{n}\sum_{i=1}^{n} f_i$	$(P3)$ max $\frac{1}{n}\sum_{i=1}^{n} f_i$
s.t.	s.t.	s.t.
$p \le 1/n$	$p \le 1/n$	$p = 1/n$
$\forall\, i \quad f_i + (i-1)\cdot p \le 1$	$\forall\, i \quad f_i + (i-1)\cdot p \le 1$	$\forall\, i \quad f_i + (i-1)\cdot p \le 1$
$\forall\, i \quad f_i \le i \cdot p$	$\forall\, i \quad f_i = i \cdot p$	$\forall\, i \quad f_i \le i \cdot p$
$\forall\, i \quad p, f_i \ge 0$	$\forall\, i \quad p, f_i \ge 0$	$\forall\, i \quad p, f_i \ge 0$

Fig. 3. (P1): Characterizes any incentive compatible mechanism. (P2) characterizes mechanisms that are regret free. (P3) characterizes mechanisms that are must-hire mechanisms.

while if $q = 1$, then the optimal strategy is to wait till the end and then hire the best candidate. We give a tight description of strategies as q changes. This can be achieved simply by modifying the linear program: simply add in the objective function $q(1 - \sum_{i=1}^{n} p_i)$. That is, if the mechanism did not hire any candidate you may hire the best candidate with probability q. Solving the primal and the corresponding dual (see Figure 2) give the following tight result. The proof is omitted.

Theorem 2. *There is a mechanism for the rehire variant that selects the best secretary with probability $e^{-(1-q)} + o(1)$ and it is optimal.*

3 Incentive Compatibility

In this section we study incentive compatible mechanisms for the secretary problem. We design a set of mechanisms $\mathcal{M}_p$ and show that with certain parameters these mechanisms are the optimal mechanisms for certain secretary problems. To this end, we derive linear formulations that characterize the set of possible incentive compatible mechanisms and also analyze the dual linear programs.

The basic linear formulation that characterizes **all** incentive compatible mechanisms appears in Figure 3. We give a set of three linear formulations. The formulation $(P1)$ characterizes all mechanisms that are incentive compatible, $(P2)$ captures mechanisms that are also regret free and $(P3)$ captures mechanisms that are must-hire mechanisms. This is formalized in the following two lemmas.

Lemma 4. ***(Mechanism to LP solution)*** *Let π be any mechanism for selecting the best candidate that is incentive compatible. Let p^π denote the probability the mechanism selects a candidate at each position i, and let f_i^π be the probability the mechanism selects the candidate at position i given that the candidate at position i is the best candidate. Then:*

- *p^π, f_i^π is a feasible solution to the linear program $(P1)$.*
- *If the mechanism is also regret free then p^π, f_i^π is a feasible solution to the linear program $(P2)$.*

– *If the mechanism is also must-hire then p^π, f_i^π is a feasible solution to the linear program $(P3)$.*
– *The objective value $\frac{1}{n}\sum_{i=1}^n f_i^\pi$ is at least the probability of selecting the best candidate by π.*

Proof. The proof follows the same ideas as in the proof of Lemma 1. The condition of incentive compatibility implies that $p_i = p_j = p$ for any two positions i and j.

Also, in the original secretary problem, every mechanism could be modified to be a regret free mechanism. This is not true for an incentive compatible mechanism. Indeed, we have the following constraint, $f_i \leq ip_i$ since the probability of hiring in the ith position is at least the probability of hiring in the ith position given that the candidate is best so far times $1/i$. If the mechanism is also supposed to be regret free then equality must hold for each i. In the must-hire part we demand that the sum of p_i is 1. The resulting formulation given in Figure 3 is after simplification.

Lemma 4 shows that the optimal solution to the linear formulations is an upper-bound on the performance of the mechanism. To show the converse we define a family of mechanisms that are defined by their probability of selecting a candidate at each position $0 \leq p \leq 1/n$, we show that the set of feasible solutions to $(P1)$ corresponds to the set of mechanisms $\mathcal{M}_p$ defined here.

Incentive Compatible Mechanism $\mathcal{M}_p$:

– Let $0 \leq p \leq 1/n$. For each $1 \leq i \leq n$, while no candidate is selected, do
  - If $1 \leq i \leq \frac{1}{2p}$, select the i^{th} candidate with probability $\frac{i}{1/p-i+1}$ if she is the best candidate so far.
  - If $\frac{1}{2p} < i \leq n$, let $r = \frac{i}{1/p-i+1}$. Select the i^{th} candidate with probability 1 if her rank is in top $\lfloor r \rfloor$ and with probability $r - \lfloor r \rfloor$ if her rank is $\lfloor r \rfloor + 1$.

The following lemma shows that every LP solution to $(P1)$ corresponds to a mechanism which performs as well as the objective value of the solution.

Lemma 5. ***(LP solution to Mechanism)*** *Let p, f_i for $1 \leq i \leq n$ be a feasible LP solution to $(P1)$. Then the mechanism $\mathcal{M}_p$ selects the best candidate with probability which is at least $\frac{1}{n}\sum_{i=1}^n f_i$.*

Proof. For any p, the optimal values of f_i are given by the following. $f_i = ip$ for $1 \leq i \leq \frac{1}{2p}$ and $f_i = 1-(i-1)p$ for $i > \frac{1}{2p}$. For ease of calculations, we ignore the fact the fractions need not be integers. These are exactly the values achieved by the mechanism $\mathcal{M}_p$ for any value p.

Lemma 6. *The mechanism $\mathcal{M}_p$ is incentive compatible for each $0 \leq p \leq 1/n$ and has efficiency of $1 - \left(\frac{1}{4pn} + \frac{pn}{2}\right)$.*

Proof. We prove by induction that the mechanism $\mathcal{M}_p$ selects each position i with probability p. It is easy to verify that for $i = 1$ this is true. For $i > 1$. The probability the mechanism chooses position i is by our induction hypothesis:

$$r \cdot \frac{1}{i} \cdot (1-(i-1)p) = \frac{i}{1/p+i-1}\frac{1}{i}(1-(i-1)p) = \frac{1-(i-1)p}{1/p+i-1} = p$$

The probability the mechanism selects the best candidate is related to f_i. $f_i = ip$ for $1 \le i \le 1/2p$, and $f_i = 1-(i-1)p$ for $1/2p < i \le n$. Thus, we get:

$$\frac{1}{n}\sum_{i=1}^{n} f_i = \frac{1}{n}\left(\sum_{i=1}^{1/2p} ip + \sum_{i=1/2p+1}^{n}(1-(i-1)p)\right) = 1 - \left(\frac{1}{4pn} + \frac{pn}{2}\right)$$

Optimizing the linear programs $(P1)$, $(P2)$ and $(P3)$ exactly, we get the following theorem. The optimality of the mechanisms can also be shown by exhibiting an optimal dual solution.

Theorem 3. *The family of mechanisms $\mathcal{M}_p$ achieves the following.*

1. *Mechanism $\mathcal{M}_{1/\sqrt{2}n}$ is incentive compatible with efficiency of* $1 - \frac{1}{\sqrt{2}} \approx 0.29$.
2. *Mechanism $\mathcal{M}_{1/2n}$ is incentive compatible and regret free with efficiency* $\frac{1}{4}$.
3. *Mechanism $\mathcal{M}_{1/n}$ is incentive compatible and must-hire with efficiency* $\frac{1}{4}$.

Moreover, all these mechanism are optimal for efficiency along with the additional property.

4 The *J*-Choice *K*-Best Secretary Problem

In this section we study a general problem of selecting as many of the top $1, \ldots, K$ ranked secretaries given J rounds to select. The mechanism is given J possible rounds in which it may select a candidate, and it gains from selecting any of the first K ranked candidates. The classical secretary problem is exactly 1-choice 1-best secretary problem. Other special cases include cases in which the mechanism is given J rounds and get profit only for the best candidate, or getting a single round, but receive profit for any of the best K candidates. Our result is a simple linear formulation that characterize all strategies for selecting the candidates.

The following two lemmas show that the above linear program exactly characterizes all feasible mechanisms for the (J, K)-secretary problem.

Lemma 7. ***(Mechanism to LP solution)*** *Let π be any mechanism for selecting the (J, K)-secretary problem. Let*

– $p_i^j(\pi)$*: The probability of accepting the candidate at ith position in the jth round for each $1 \le i \le n$ and each $1 \le j \le J$.*
– $q_i^{j|k}(\pi)$*: The probability of accepting the candidate ith position in the jth round given that the candidate is the kth best candidate* **among the *i* first candidates** *for each $1 \le i \le n$, $1 \le j \le J$ and $1 \le k \le K$.*

$$
\begin{array}{lll}
\multicolumn{3}{l}{\max F(q) = \frac{1}{n} \cdot \sum_{i=1}^{n} \sum_{j=1}^{J} \sum_{k=1}^{K} \sum_{\ell=1}^{k} \frac{\binom{n-i}{k-\ell}\binom{i-1}{\ell-1}}{\binom{n-1}{k-1}} q_i^{j|\ell}} \\
\text{s.t.} & & \\
\forall\, 1 \le i \le n,\, 1 \le j \le J & p_i^j & = \frac{1}{i} \sum_{k=1}^{\min\{i,K\}} q_i^{j|k} \\
\forall\, 1 \le i \le n,\, 1 \le k \le K & q_i^{1|k} & \le 1 - \sum_{\ell<i} p_i^1 \\
\forall\, 1 \le i \le n,\, 1 \le k \le K,\, 2 \le j \le J & q_i^{j|k} & \le \sum_{\ell<i} p_i^{j-1} - \sum_{\ell<i} p_i^j \\
\forall\, 1 \le i \le n,\, 1 \le k \le K,\, 1 \le j \le J & p_i^j, q_i^{j|k} & \ge 0
\end{array}
$$

Fig. 4. Linear program for the (J, K)-secretary problem

Then $(p(\pi), q(\pi))$ is a feasible solution and expected number of top K candidates selected is at most $F(p(\pi), q(\pi))$.

Proof. Let us prove the first type of constraints of the form: $p_i^j = \frac{1}{i} \sum_{k=1}^{\min\{i,K\}} q_i^{j|k}$

It is clear that there is no reason for any mechanism to select a candidate which is not at least the K best so far. Such a candidate cannot be even potentially one of the K best globally and therefore is not profitable for the mechanism. Thus, the probability any mechanism selects the ith candidate in the jth round is the sum of the probability of selecting the ith candidate in the jth round given that the candidate is the kth best candidate so far times $1/i$, which is the probability that the candidate is the kth best so far. We sum until the minimum between i and K to get the desired equality which holds for every mechanism.Let us now prove the third type of constraints (the second type follows by the same arguments). Consider any mechanism and some position i and some rounds j.

$$
\begin{aligned}
q_i^{j|k} &= Pr[\pi \text{ selects candidate } i \text{ in round j}| \text{ candidate } i \text{ is } k\text{th best so far}] \\
&\le Pr[\pi \text{ selects exactly } j-1 \text{ candidates out of cand. } \{1, \ldots, i-1\}] \\
&\qquad\qquad\qquad\qquad\qquad\qquad | \text{ candidate } i \text{ is } k\text{th best so far}] \\
&= Pr[\pi \text{ selects exactly } j-1 \text{ candidates out of cand. } \{1, \ldots, i-1\}] \\
&= \sum_{\ell<i} p_i^{j-1}(\pi) - \sum_{\ell<i} p_i^j(\pi)
\end{aligned}
$$

The inequality follows since in order to select candidate i in round j the mechanism must have selected exactly $j-1$ candidates out of the previous $i-1$ candidates. The following equality then follows since the decisions made by the policy with respect to the $i-1$ candidates depend only on the relative ranks of the $i-1$ candidates, and is independent of the rank of the ith candidate with respect to these candidates. The final equality follows since the event of selecting $j-1$ candidates contains the event of selecting j candidates, which concludes our proof.

Finally, let us consider the objective function and prove that it upper bounds the performance of the mechanism. For analysis purpose let us consider the probabilities $f_i^{j|k}$ that are defined as probability of selecting the ith candidate in the jth round given that the kth best candidate is in the ith position. Note that the main difference between $f_i^{j|k}$ and $q_i^{j|k}$ is that while the former consider the kth best candidate overall, the latter only looks from the mechanism's perspective and therefore looks at the event of the kth best candidate among the first i candidates. It is easy to state the objective function using the first set of variables as simply the sum over all values of i, j and k of $f_i^{j|k}$ divided by $1/n$.

To finish we simply define each $f_i^{j|k}$ in terms of $q_i^{j|k}$ which proves the lemma.

Claim. For each $1 \le i \le n$, $1 \le j \le J$ and $1 \le k \le K$, we must have

$$f_i^{j|k} = \sum_{\ell=1}^{k} \frac{\binom{i-1}{\ell-1}\binom{n-i}{k-\ell}}{\binom{n-1}{k-1}} q_i^{j|\ell}$$

The proof is omitted due to lack of space. The proof of Lemma 7 follows directly from the claim.

Lemma 7 shows that the optimal solution to (P) is an upper-bound on the performance of the mechanism. The following lemma shows that every LP solution actually corresponds to a mechanism which performs as well as the objective value of the solution.

Lemma 8. ***(LP solution to Mechanism)*** *Let (p, q) be any feasible LP solution to (P). Then consider the mechanism π defined inductively as follows. For each position $1 \le i \le n$,*

- *If the mechanism has not selected any candidate among position $\{1, \dots, i-1\}$ and the rank of candidate i among $\{1, \dots, i\}$ is k for some $1 \le k \le K$, then select candidate i with probability $\frac{q_i^{1|k}}{1-\sum_{\ell<i} p_i^1}$.*
- *If the mechanism has selected $j-1$ candidates in positions $1, \dots, i-1$ for some $2 \le j \le J$ and the rank of candidate i among $\{1, \dots, i\}$ is k for some $1 \le k \le K$, then select candidate i with probability $\frac{q_i^{j|k}}{\sum_{\ell<i} p_i^{j-1} - \sum_{\ell<i} p_i^j}$.*
- *Else do not select candidate i.*

Then expected number of top k candidates selected by π is exactly $F(p, q)$.

Proof (Sketch). The proof is by induction on the steps of the mechanism. It can be verified easily that the procedure above keeps by induction that $p_i^j(\pi) = p_i^j, q_i^{j|k}(\pi) = q_i^{j|k}$. That is, the probability the mechanism selects the ith candidate in the jth round is the same as the LP. As stated in Lemma 7 there is a correspondence between the values of $q_i^{j|k}(\pi)$ and $f_i^{j|k}(\pi)$ which is the probabilities of hiring the ith candidate in the jth round given that the candidate is the kth best. Thus, the objective function of π is exactly $F(p, q)$.

We now give optimal mechanism for the $(1, 2)$ and $(2, 1)$-secretary problem. Observe that $(1, 1)$-secretary problem is the traditional secretary problem.

Theorem 4. *There exists mechanisms which achieve a performance of*

1. $\frac{1}{e} + \frac{1}{e^{1.5}} \simeq 0.591$ *for* $(2, 1)$*-secretary problem.*
2. $\simeq 0.572284$ *for the* $(1, 2)$ *secretary problem.*

Moreover all these mechanisms are (nearly) optimal.

Proof. (Sketch) To give a mechanism, we will give a primal solution to $LP(J, K)$. The optimality is shown by exhibiting a dual solution of the same value. Due to lack of space we only prove the $(2, 1)$ case.

(2,1)-secretary. Let $t_1 = \frac{n}{e^{3/2}}$ and $t_2 = \frac{n}{e}$. Consider the following mechanism that selects the i^{th} candidate if i^{th} candidate is best so far and $t_1 \leq i < t_2$ and no other candidate has been selected or if $t_2 \leq i \leq n$ and i^{th} candidate is best so far and at most one candidate has been selected. The performance of this mechanism is $\frac{1}{e} + \frac{1}{e^{\frac{3}{2}}}$. The mechanism corresponds to the primal LP solution where $p_i^1 = 0$ for $1 \leq i < t_1$ and $p_i^1 = \frac{t_1 - 1}{i(i-1)}$ for $t_1 \leq i \leq n$, $p_i^2 = 0$ for $1 \leq i < t_2$ and $p_i^2 = \frac{t_2 - t_1}{i(i-1)} - \frac{1}{i(i-1)} \sum_{j=1}^{i-1} \frac{t_1 - 1}{i-1}$ for $t_2 \leq i \leq n$, $q_i^{j|1} = i \cdot p_i^j$ for each $1 \leq j \leq 2$ and $1 \leq i \leq n$.

Dual Solution. We first simplify the primal linear program by eliminating the $q_i^{j|k}$ variables using the first set of constraints. Let y_i denote the dual variables corresponding to the second set of constraints and z_i the variables corresponding to the third set of constraints. Then the following dual solution is of value $\frac{1}{e} + \frac{1}{e^{\frac{3}{2}}} - o(1)$. Set $z_i = 0$ for $1 \leq i < t_2$ and $z_i = \frac{1}{n}(1 - \sum_{j=i+1}^{n} \frac{1}{j})$ for $t_2 \leq i \leq n$. Set $y_i = 0$ for $1 \leq i < t_1$, $y_i = \frac{1}{n}(1 - \sum_{j=i+1}^{n} \frac{1}{j}) + \sum_{j=t_2}^{n} \frac{1}{in}(1 - \sum_{k=j+1}^{n} \frac{1}{k})$ for $t_1 \leq i < t_2$ and $y_i = \frac{1}{n}(1 - \sum_{j=i+1}^{n} \frac{1}{j}) + \sum_{j=i}^{n} \frac{1}{in}(1 - \sum_{k=j+1}^{n} \frac{1}{k})$ for $t_2 \leq i \leq n$.

5 Further Discussion

Characterizing the set of mechanisms in secretary type problems as a linear polytope possesses many advantages. In contrast to methods of factor revealing LPs in which linear programs are used to analyze a single algorithm, here we characterize **all** mechanisms by a linear program. One direction for future research is trying to capture more complex settings of a more combinatorial nature. One such example is the clean problem studied in [4] in which elements of a matroid arrive one-by-one. This problem seems extremely appealing since matroid constraints are exactly captured by a linear program. Another promising direction is obtaining upper bounds. While the linear program which characterizes the performance may be too complex to obtain a simple mechanism, the dual linear may still be used for obtaining upper bounds on the performance of any mechanism. We believe that linear programming and duality is a powerful approach for studying secretary problems and will be applicable in more generality.

References

1. Awerbuch, B., Azar, Y., Meyerson, A.: Reducing Truth-Telling Online Mechanisms to Online Optimization. In: Proceedings of ACM Symposium on Theory of Computing, pp. 503–510 (2003)
2. Babaioff, M., Immorlica, N., Kempe, D., Kleinberg, R.: A Knapsack Secretary Problem with Applications. In: Charikar, M., Jansen, K., Reingold, O., Rolim, J.D.P. (eds.) RANDOM 2007 and APPROX 2007. LNCS, vol. 4627, pp. 16–28. Springer, Heidelberg (2007)

3. Babaioff, M., Immorlica, N., Kempe, D., Kleinberg, R.: Online Auctions and Generalized Secretary Problems. SIGecom Exchange 7, 1–11 (2008)
4. Babaioff, M., Immorlica, N., Kleinberg, R.: Matroids, Secretary Problems, and Online Mechanisms. In: Proceedings 18th ACM-SIAM Symposium on Discrete Algorithms (2007)
5. Babaioff, M., Dinitz, M., Gupta, A., Immorlica, N., Talwar, K.: Secretary problems: weights and discounts. In: SODA '09: Proceedings of the Nineteenth Annual ACM -SIAM Symposium on Discrete Algorithms, pp. 1245–1254. Society for Industrial and Applied Mathematics, Philadelphia (2009)
6. Buchbinder, N., Singh, M., Jain, K.: Incentives in Online Auctions and Secretary Problems via Linear Programming (2009) (manuscript)
7. Buchbinder, N., Jain, K., Naor, J(S.): Online primal-dual algorithms for maximizing ad-auctions revenue. In: Arge, L., Hoffmann, M., Welzl, E. (eds.) ESA 2007. LNCS, vol. 4698, pp. 253–264. Springer, Heidelberg (2007)
8. Dynkin, E.B.: The Optimum Choice of the Instant for Stopping a Markov Process. Sov. Math. Dokl. 4 (1963)
9. Ferguson, T.S.: Who Solved the Secretary Problem? Statist. Sci. 4, 282–289 (1989)
10. Freeman, P.R.: The Secretary Problem and its Extensions: A Review. International Statistical Review 51, 189–206 (1983)
11. Gardner, M.: Mathematical Games. Scientific American, 150–153 (1960)
12. Goemans, M., Kleinberg, J.: An improved approximation ratio for the minimum latency problem. In: SODA '96: Proceedings of the seventh annual ACM-SIAM symposium on Discrete algorithms, pp. 152–158 (1996)
13. Hajiaghayi, M.T., Kleinberg, R., Parkes, D.C.: Adaptive Limited-Supply Online Auctions. In: Proceedings of the 5th ACM Conference on Electronic Commerce (2004)
14. Jain, K., Mahdian, M., Markakis, E., Saberi, A., Vazirani, V.V.: Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp. J. ACM 50(6), 795–824 (2003)
15. Kleinberg, R.: A Multiple-Choice Secretary Algorithm with Applications to Online Auctions. In: Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete algorithms (2005)
16. 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. LNCS, vol. 5556, pp. 508–520. Springer, Heidelberg (2009)
17. Lavi, R., Nisan, N.: Competitive Analysis of Incentive Compatible On-line Auctions. In: Proceedings of 2nd ACM Conf. on Electronic Commerce, pp. 233–241 (2000)
18. Lindley, D.V.: Dynamic Programming and Decision Theory. Applied Statistics 10, 39–51 (1961)
19. Mehta, A., Saberi, A., Vazirani, U., Vazirani, V.: Adwords and generalized online matching. J. ACM 54(5), 22 (2007)
20. Samuels, S.M.: Secretary Problems. In: Handbook of Sequential Analysis, vol. 118, pp. 381–405 (1991)

Branched Polyhedral Systems

Volker Kaibel and Andreas Loos

Otto-von-Guericke-Universität Magdeburg, Institut für Mathematische Optimierung
Universitätsplatz 2, 39108 Magdeburg, Germany
{kaibel,loos}@ovgu.de

Abstract. We introduce the framework of branched polyhedral systems that can be used in order to construct extended formulations for polyhedra by combining extended formulations for other polyhedra. The framework, for instance, simultaneously generalizes extended formulations like the well-known ones (see Balas [1]) for the convex hulls of unions of polyhedra (disjunctive programming) and like those obtained from dynamic programming algorithms for combinatorial optimization problems (due to Martin, Rardin, and Campbell [11]). Using the framework, we construct extended formulations for full orbitopes (the convex hulls of all 0/1-matrices with lexicographically sorted columns), we show for two special matching problems, how branched polyhedral systems can be exploited in order to construct formulations for certain nested combinatorial problems, and we indicate how one can build extended formulations for stable set polytopes using the framework of branched polyhedral systems.

1 Introduction

An extended formulation for a polyhedron $P \subseteq \mathbb{R}^n$ is a linear system $Ay \leq b$ defining a polyhedron $Q = \{y \in \mathbb{R}^d \mid Ay \leq b\}$ such that there is a projection (linear map) $p : \mathbb{R}^d \to \mathbb{R}^n$ with $p(Q) = P$. With respect to optimization of linear functionals $x \mapsto \langle c, x \rangle$ over P, such an extended formulation is similarly useful as a description of P by means of linear inequalities in $\mathbb{R}^n$, since we have $\max\{\langle c, x \rangle \mid x \in P\} = \max\{\langle p^\star(c), y \rangle \mid Ay \leq b\}$ with the map $p^\star : \mathbb{R}^n \to \mathbb{R}^d$ that is adjoint to p, i.e., $p^\star(x) = T^t x$ for all $x \in \mathbb{R}^n$ if $T \in \mathbb{R}^{n \times d}$ is the matrix with $p(y) = Ty$ for all $y \in \mathbb{R}^d$.

Extended formulations play an increasingly important role in polyhedral combinatorics and mixed integer programming. For a survey on extended formulations in combinatorial optimization we refer to [2], and for examples of recent interesting results on extended formulations for mixed integer problems to, e.g., [3,4]. The reason for the importance of extended formulations lies in the fact that they can have much less constraints than a description in the original space. Moreover, in many cases the extensions reflect much better the structure of the underlying problem, because not all relevant aspects can be expressed linearly in the original variables. In fact, in several cases, (small) extended formulations are rather easy to derive, while it seems extremely difficult to come up with a linear description in the original space.

A trivial extended formulation for a polytope $P \subseteq \mathbb{R}^n$ with vertex set X is given by the simplex $Q \in \mathbb{R}^X$ that is the convex hull of the standard unit vectors in $\mathbb{R}^X$ with the projection $p : \mathbb{R}^X \to \mathbb{R}^n$ defined via $p(y) = \sum_{x \in X} y_x \cdot x$. Of course, this extended formulation usually neither helps to solve an optimization problem nor does it provide much insight, because it is just trivially combined from many trivial building blocks, the vertices of P. In many cases, however, one can construct much better (e.g., smaller) extended formulations by combining less non-trivial building blocks that are still simple enough. For instance, one may be able to cover the vertex set X by $X = X_1 \cup \cdots \cup X_q$ such that one has available nice linear descriptions of the polytopes $P^{(i)}$ with vertex sets X_i, respectively, which one wants to combine to a linear description or an extended formulation of the whole polytope P (see Section 4.1). This is known as disjunctive programming (see, e.g., [1]). As a simple combinatorial example in which the interplay of the non-trivial building blocks is slightly more involved, let us consider those perfect matchings in a 2ℓ-partite graph with corresponding partitioning of the node set into 2ℓ parts, each of size p, that induce, on each of the bipartite subgraphs defined by any pair of parts, either an empty or a perfect matching. The question is: How can one derive a linear description or an extended formulation for the polytope associated with these matchings from linear descriptions of the perfect matching polytope on the complete graph on 2ℓ nodes and from the perfect matching polytopes on the bipartite subgraphs induced by pairs of parts (see Subsection 4.4)? Or suppose that we can partition the node set of a graph in such a way that the stable set polyopes on each of the parts can be described by linear inequalities (or by extended formulations). How can one construct extended formulations for the stable set polytope of the whole graph from descriptions of the stable set polytopes of the parts and some polytope that describes the interplay of the parts (see Subsection 4.5)?

The framework of *branched polyhedral systems* that we introduce in this paper can be used for combining knowledge on several polyhedra in order to derive extended formulations (or, in some cases, even linear descriptions in the original space) for other polyhedra. We demonstrate the capabilities of the framework on the examples mentioned in the previous paragraph, as well as by showing how the framework generalizes the directed hypergraph framework developed in [11] in order to obtain extended formulations from certain dynamic programming algorithms (see Subsection 4.3). The relation between branched polyhedral systems and the composition methods developed by Margot [10] and Schaffers [12] will be discussed in the journal version of this paper. We also provide the first polynomial size extended formulations for full orbitopes, i.e., the convex hulls of all 0/1-matrices with lexicographically sorted columns (see Section 4.2).

The interplay between the polyhedra in our framework is governed by acyclic directed graphs, where the polyhedra that form the building blocks are assigned to the (non-sink) nodes of the digraph. The central result of the paper is Theorem 1 (stated and proved in Section 3) that establishes how a polyhedron defined by an inner description constructed from inner descriptions of the building block polyhedra as described in Section 2 can be described by means of an extended formulation derived from extended formulations for those building blocks. The applications mentioned above are treated in Section 4.

For easier reference, we define here most of the notions and notations used throughout the paper. For a directed graph $D = (V, A)$ (with $A \subseteq V \times V \setminus \{(v, v) \mid v \in V\}$) and some node $v \in V$, we denote by $\mathrm{N}^{\mathrm{in}}(v) = \{u \in V \mid (u, v) \in A\}$ and $\mathrm{N}^{\mathrm{out}}(v) = \{w \in V \mid (v, w) \in A\}$ the sets of *in-neighbors* and *out-neighbors* of v, respectively, by $\delta^{\mathrm{in}}(v) = \{(u, v) \mid u \in \mathrm{N}^{\mathrm{in}}(v)\}$ and $\delta^{\mathrm{out}}(v) = \{(v, w) \mid w \in \mathrm{N}^{\mathrm{out}}(v)\}$ the *in-star* and the *out-star* of v, respectively, and by $\mathrm{R}(v)$ (the *reach* of v) the set of nodes that can be reached from v by some directed path in D. A digraph $D = (V, A)$ is *acyclic* if it has no directed cycle. We denote the set of sinks of $D = (V, A)$ by $V_T \subseteq V$.

An acyclic subset $\varnothing \neq B \subseteq A$ of arcs of a directed graph $D = (V, A)$ is called a *branching* in D if, viewed undirectedly, B forms a tree, and $|B \cap \delta^{\mathrm{in}}(v)| = 1$ holds for every v in the set $V(B)$ of nodes incident to arcs in B (the nodes *covered* by B), except for one node, called the *root* $r \in V(B)$ of B (for which $B \cap \delta^{\mathrm{in}}(r) = \varnothing$ holds). For $W \subset V$, we denote by $G[W]$ the subgraph of G induced by W.

For an undirected graph $G = (V, E)$, we denote $\delta(v) = \{e \in E \mid v \in E\}$ for all $v \in V$, and $\delta(W) = \{e \in E \mid |e \cap W| = 1\}$ for all $W \subseteq V$. For $W, W' \subseteq V$ with $W \cap W' = \varnothing$, we define $(W : W') = \{e \in E \mid e \cap W \neq \varnothing, e \cap W' \neq \varnothing\}$ and $(w : W') = (\{w\} : W')$ for $w \in V \setminus W'$.

For a vector $x \in \mathbb{R}^M$, the *support* $\mathrm{supp}(x) \subset M$ of x is the set of all $i \in M$ with $x_i \neq 0$. For a subset $N \subseteq M$ of indices of x, we define $x(N) = \sum_{i \in N} x_i$. Furthermore, we denote $[n] = \{1, 2, \ldots, n\}$. For a set $X \subseteq \mathbb{R}^n$, $\mathrm{conv}(X) = \{\sum_{x \in \tilde{X}} \lambda_x \cdot x \mid \tilde{X} \subseteq X \text{ finite}, \sum_{x \in \tilde{X}} \lambda_x = 1, \lambda_x \geq 0 \text{ for all } x \in \tilde{X}\}$ denotes the *convex hull*, $\mathrm{ccone}(X) = \{\sum_{x \in \tilde{X}} \lambda_x \cdot x \mid \tilde{X} \subseteq X \text{ finite}, \lambda_x \geq 0 \text{ for all } x \in \tilde{X}\}$ is the *convex-conic hull* of X, and $\mathrm{cone}(X) = \{\lambda \cdot x \mid x \in X, \lambda \in \mathbb{R}_+\}$ is the *conic hull* of X.

If $N \subseteq M$ is a subset of the finite set M, then $\chi(N) \in \{0, 1\}^M$ with $\chi(N)_i = 1$ if and only if $i \in N$ is the *characteristic vector* of $N \subseteq M$. For $x \in \mathbb{R}^M$, $x_N \in \mathbb{R}^N$ is the vector formed by the components of x corresponding to N. The zero-vector and the standard unit vectors in $\mathbb{R}^M$ are denoted by $\mathbb{O}_M$ and $\mathbb{e}_i$ (for $i \in M$).

The symbol $\uplus$ is used to represent disjoint unions of sets.

2 The Concept

A *branched polyhedral system* (*BPS*) is a pair $\mathcal{B} = (D, (P^{(v)}))$ of an acyclic directed graph $D = (V, A)$ with a unique source $s \in V$ and a family $P^{(v)} \subseteq \mathbb{R}^{\mathrm{N}^{\mathrm{out}}(v)}$ $(v \in V \setminus V_T)$ of non-empty polyhedra, such that, for every $v \in V \setminus V_T$, there is an *admissable pair* $(G^{(v)}_{\mathrm{cv}}, G^{(v)}_{\mathrm{cc}})$ *of generating sets*, i.e., finite sets $\varnothing \neq G^{(v)}_{\mathrm{cv}} \subseteq \mathbb{R}^{\mathrm{N}^{\mathrm{out}}(v)}$ and $G^{(v)}_{\mathrm{cc}} \subseteq \mathbb{R}^{\mathrm{N}^{\mathrm{out}}(v)}$ with

$$P^{(v)} = \mathrm{conv}(G^{(v)}_{\mathrm{cv}}) + \mathrm{ccone}(G^{(v)}_{\mathrm{cc}})$$

satisfying the following for all $\tilde{x} \in G_{\mathrm{cv}}^{(v)} \cup G_{\mathrm{cc}}^{(v)}$:

(S.1) $\tilde{x}_w > 0$ for all $w \in \operatorname{supp}(\tilde{x}) \setminus V_T$
(S.2) $\mathrm{R}(w) \cap \mathrm{R}(w') = \varnothing$ for all $w, w' \in \operatorname{supp}(\tilde{x})$, $w \neq w'$

If $P^{(v)}$ is pointed (i.e., it has vertices), then we only need to consider the vertex set $G_{\mathrm{cv}}^{(v)}$ of $P^{(v)}$ and some set $G_{\mathrm{cc}}^{(v)}$ that contains exactly one non-zero vector from each extreme ray of the recession cone of $P^{(v)}$. As we can identify $\mathrm{N}^{\mathrm{out}}(v)$ with $\delta^{\mathrm{out}}(v)$, we will also consider the polyhedra $P^{(v)}$ as subsets of $\mathbb{R}^{\delta^{\mathrm{out}}(v)}$.

For the remainder of this section, let $\mathcal{B} = (D, (P^{(v)}))$ be a BPS with $D = (V, A)$ and source node $s \in V$. We fix one family $(G_{\mathrm{cv}}^{(v)}, G_{\mathrm{cc}}^{(v)})$ of admissable pairs of generating sets.

With respect to the fixed family of admissable pairs of generating sets, we define two finite sets $\mathcal{G}_{\mathrm{cv}}(\mathcal{B}), \mathcal{G}_{\mathrm{cc}}(\mathcal{B}) \subseteq \mathbb{R}^V$ that will be used to define a polyhedron associated with the BPS later (it will turn out that this polyhedron does not depend on the particular family, but on the BPS only).

We start by constructing $\mathcal{G}_{\mathrm{cv}}(\mathcal{B}) \subseteq \mathbb{R}^V$ as the set that contains all points $x \in \mathbb{R}^V$ for which the following holds:

(V.1) $x_s = 1$
(V.2) For each $v \in \operatorname{supp}(x) \setminus V_T$, we have $\frac{1}{x_v} x_{\mathrm{N}^{\mathrm{out}}(v)} \in G_{\mathrm{cv}}^{(v)}$.
(V.3) For each $v \in \operatorname{supp}(x) \setminus \{s\}$, we have $x_{\mathrm{N}^{\mathrm{in}}(v)} \neq \mathbb{O}_{\mathrm{N}^{\mathrm{in}}(v)}$.

Note that $\mathcal{G}_{\mathrm{cv}}(\mathcal{B}) \neq \varnothing$ holds, since we required the polyhedra $P^{(v)}$ to be non-empty. Furthermore, looking at the nodes in order of a topological ordering of the acyclic digraph D (i.e., v appears before w in the ordering for all $(v, w) \in A$), one finds $|\mathcal{G}_{\mathrm{cv}}(\mathcal{B})| < \infty$.

For a node $v \in V$, the *truncation* of $\mathcal{B}$ at v is the BPS $\mathcal{B}_v$ induced by $\mathcal{B}$ on the reach $\mathrm{R}(v)$ of v in D (with v as its source node). Clearly, a family of admissable pairs of generating sets for $\mathcal{B}$ induces such a family for $\mathcal{B}_v$, to which we refer in the following characterization of $\mathcal{G}_{\mathrm{cv}}(\mathcal{B})$ that in particular clarifies the name *branched* polyhedral system. For a vector $x \in \mathbb{R}^V$, we denote by $A[x]$ the set of arcs $(v, w) \in A$ with $x_v, x_w \neq 0$. We omit the proof of the following proposition in this extended abstract.

Proposition 1. *For each $x \in \mathbb{R}^V$, we have $x \in \mathcal{G}_{cv}(\mathcal{B})$ if and only if*

(a) $x_s = 1$,
(b) $\frac{1}{x_v} \cdot x_{\mathrm{R}(v)} \in \mathcal{G}_{cv}(\mathcal{B}_v)$ *for each* $v \in \operatorname{supp}(x) \setminus V_T$*, and*
(c) $A[x]$ *is a branching in* D *with root* s, $\operatorname{supp}(x)$ *being the set of covered nodes.*

We continue by constructing $\mathcal{G}_{\mathrm{cc}}(\mathcal{B})$ as the set of all $x \in \mathbb{R}^V$ for which there is some node $v_x \in V \setminus V_T$ (the *root node* of x) with:

(R.1) $x_{\mathrm{N}^{\mathrm{out}}(v_x)} \in G_{\mathrm{cc}}^{(v_x)}$
(R.2) $x_{\bar{V}} = \mathbb{O}$ for $\bar{V} = V \setminus \cup\{\mathrm{R}(w) \mid w \in \mathrm{N}^{\mathrm{out}}(v_x) \cap \operatorname{supp}(x)\}$
(R.3) $\frac{1}{x_w} x_{\mathrm{R}(w)} \in \mathcal{G}_{\mathrm{cv}}(\mathcal{B}_w)$ for all $w \in \mathrm{N}^{\mathrm{out}}(v_x) \cap \operatorname{supp}(x)$

Note that, since $\mathcal{G}_{\mathrm{cv}}(\mathcal{B}_w)$ is finite for all $w \in V \setminus V_T$, we also have $|\mathcal{G}_{\mathrm{cc}}(\mathcal{B})| < \infty$ (we already observed $|\mathcal{G}_{\mathrm{cv}}(\mathcal{B})| < \infty$).

Finally, the *polyhedron defined by the branched polyhedral system* $\mathcal{B}$ is

$$\mathcal{P}(\mathcal{B}) = \operatorname{conv}(\mathcal{G}_{\mathrm{cv}}(\mathcal{B})) + \operatorname{ccone}(\mathcal{G}_{\mathrm{cc}}(\mathcal{B}))\,,$$

where Theorem 1 will imply that this definition does not depend on the particular choice of a family of admissable pairs of generating sets for the polyhedra $P^{(v)}$.

Before concluding the section with considering a special class of branched polyhedral systems, we state two remarks on nonnegativity (following readily from (S.1)) and integrality (following readily from (V.1), (V.2), (V.3)).

Remark 1

1. For $v \in V \setminus V_T$, we have $x_v \geq 0$ for all $x \in \mathcal{G}_{\mathrm{cv}}(\mathcal{B}) \cup \mathcal{G}_{\mathrm{cc}}(\mathcal{B})$ (thus for all $x \in \mathcal{P}(\mathcal{B})$).
2. If $P^{(v)}$ is a *pointed* integral polyhedron for each $v \in V \setminus V_T$, then $\mathcal{P}(\mathcal{B})$ is integral as well.

A *branched combinatorial system* (*BCS*) is a pair $\mathcal{C} = (D, (\mathcal{S}^{(v)}))$ of an acyclic directed graph $D = (V, A)$ with a unique source $s \in V$ (and set $V_T \subseteq V$ of sinks) and a non-empty family $\mathcal{S}^{(v)}$ of subsets of $\mathrm{N}^{\mathrm{out}}(v)$ for every node $v \in V \setminus V_T$ such that, for each $v \in V \setminus V_T$ and $S \in \mathcal{S}^{(v)}$ and for every pair $w, w' \in S$ with $w \neq w'$, we have $\mathrm{R}(w) \cap \mathrm{R}(w') = \varnothing$. A subset F of nodes is *feasible* for the BCS $\mathcal{C}$ if it satisfies

(F.1) $s \in F$,
(F.2) $F \cap \mathrm{N}^{\mathrm{out}}(v) \in \mathcal{S}(v)$ for all $v \in F \setminus V_T$, and
(F.3) $F \cap \mathrm{N}^{\mathrm{in}}(v) \neq \varnothing$ for all $v \in F \setminus \{s\}$.

With $P^{(v)} = \operatorname{conv}\{\chi(S) \in \{0,1\}^{\mathrm{N}^{\mathrm{out}}(v)} \mid S \in \mathcal{S}^{(v)}\}$ for all $v \in V \setminus V_T$, we denote by $\mathcal{B}(\mathcal{C}) = (D, (P^{(v)}))$ the BPS defined by the BCS $\mathcal{C}$. Clearly, the admissible pairs of generating sets that we consider in this context are $G_{\mathrm{cv}}^{(v)} = \{\chi(S) \in \{0,1\}^{\mathrm{N}^{\mathrm{out}}(v)} \mid S \in \mathcal{S}^{(v)}\}$ and $G_{\mathrm{cc}}^{(v)} = \varnothing$ for all $v \in V \setminus V_T$, for which we find $\mathcal{G}_{\mathrm{cv}}(\mathcal{B}(\mathcal{C})) = \{\chi(F) \in \{0,1\}^V \mid F \subseteq V \text{ feasible for } \mathcal{C}\}$ (and, clearly, $\mathcal{G}_{\mathrm{cc}}(\mathcal{B}(\mathcal{C})) = \{\mathbb{O}\}$). In particular, Proposition 1 (c) implies that $A \cap (F \times F)$ is a branching with root s, and, with $\mathcal{P}(\mathcal{C}) = \mathcal{P}(\mathcal{B}(\mathcal{C}))$, we have $\mathcal{P}(\mathcal{C}) = \operatorname{conv}\{\chi(F) \in \{0,1\}^V \mid F \subseteq V \text{ feasible for } \mathcal{C}\}$.

3 Inequality Descriptions

For a non-empty polyhedron $\varnothing \neq Q \subseteq \mathbb{R}^d$ with *recession cone* $\operatorname{rec}(Q) = \{z \in \mathbb{R}^d \mid \tilde{z} + z \in Q \text{ for all } \tilde{z} \in Q\}$, the *homogenization*

$$\operatorname{homog}(Q) = \operatorname{cone}\{(z,1) \mid z \in Q\} + \{(z,0) \mid z \in \operatorname{rec}(Q)\} \subseteq \mathbb{R}^d \oplus \mathbb{R}$$

of $Q = \{z \in \mathbb{R}^d \mid Az \leq b\}$ with $A \in \mathbb{R}^{m \times d}$, $b \in \mathbb{R}^m$ is the polyhedral cone

$$\operatorname{homog}(Q) = \{(z,\xi) \in \mathbb{R}^d \oplus \mathbb{R} \mid Az - \xi b \leq 0, \xi \geq 0\}\,.$$

The following result, showing how to obtain, for a BPS $\mathcal{B}$, an extended formulation for $\mathcal{P}(\mathcal{B})$ from extended formulations for the polyhedra $P^{(v)}$, makes up the core of the paper.

Theorem 1. *Let $\mathcal{B} = (D, (P^{(v)}))$ be a branched polyhedral system with a digraph $D = (V, A)$ with source node $s \in V$, let $V_T \subseteq V$ be the set of sinks of D, and suppose that $P^{(v)} = \pi^{(v)}(Q^{(v)})$ with a polyhedron $Q^{(v)} \subseteq \mathbb{R}^{d(v)}$ and a projection (linear map) $\pi^{(v)} : \mathbb{R}^{d(v)} \to \mathbb{R}^{\delta^{\text{out}}(v)}$ for all $v \in V \setminus V_T$.*

Then, with the polyhedron $\mathcal{Q}(\mathcal{B}) \subseteq \mathbb{R}^V \oplus \mathbb{R}^A \oplus (\bigoplus_{v \in V \setminus V_T} \mathbb{R}^{d(v)})$ defined by

$$x_s = 1 \tag{1}$$

$$x_v = y(\delta^{\text{in}}(v)) \qquad \textit{for all } v \in V \setminus \{s\} \tag{2}$$

$$y_{\delta^{\text{out}}(v)} = \pi^{(v)}(z^{(v)}) \qquad \textit{for all } v \in V \setminus V_T \tag{3}$$

$$(z^{(v)}, x_v) \in \text{homog}(Q^{(v)}) \qquad \textit{for all } v \in V \setminus V_T \tag{4}$$

and the orthogonal projection $\pi : \mathbb{R}^V \oplus \mathbb{R}^A \oplus (\bigoplus_{v \in V \setminus V_T} \mathbb{R}^{d(v)}) \to \mathbb{R}^V$, we have

$$\pi(\mathcal{Q}(\mathcal{B})) = \text{conv}(\mathcal{G}_{cv}(\mathcal{B})) + \text{ccone}(\mathcal{G}_{cc}(\mathcal{B})) = \mathcal{P}(\mathcal{B})$$

with $\mathcal{G}_{cv}(\mathcal{B})$ and $\mathcal{G}_{cc}(\mathcal{B})$ defined with respect to any family $(G_{cv}^{(v)}, G_{cc}^{(v)})$ of admissable pairs of generating sets for the polyhedra $P^{(v)}$ (for all $v \in V \setminus V_T$).

Proof. We start by establishing $\mathcal{P}(\mathcal{B}) \subseteq \pi(\mathcal{Q}(\mathcal{B}))$.

First, let $x \in \mathcal{G}_{\text{cv}}(\mathcal{B})$ be arbitrary. For all $v \in (V \setminus V_T) \setminus \text{supp}(x)$, we define $y_{(v,w)} = 0$ for all $w \in \text{N}^{\text{out}}(v)$ and $z^{(v)} = \mathbb{O}$, and for all $v \in (V \setminus V_T) \cap \text{supp}(x)$ (in particular: $x_v > 0$ due to Rem. 1(1)), we set $y_{(v,w)} = x_w$ for all $w \in \text{N}^{\text{out}}(v)$ and $z^{(v)} = x_v \cdot \tilde{z}^{(v)}$ for an arbitrary $\tilde{z}^{(v)} \in Q^{(v)}$ with $\pi^{(v)}(\tilde{z}^{(v)}) = \frac{1}{x_v} y_{\delta^{\text{out}}(v)}$ (such a $\tilde{z}^{(v)}$ exists, as we have $\frac{1}{x_v} x_{\text{N}^{\text{out}}(v)} \in P^{(v)}$ due to (V.2) and $y_{\delta^{\text{out}}(v)} = x_{\text{N}^{\text{out}}(v)}$ due to the settings done just before). Thus, conditions (3) and (4) hold for this vector (x, y, z). Furthermore, (1) (due to (V.1)) and (2) (due to Proposition 1 implying that $y_{(u,v)} = x_v$ holds for exactly one $u \in \text{N}^{\text{in}}(v)$ and $y_{(u,v)} = 0$ for all others) are satisfied as well, which shows $x \in \pi(\mathcal{Q}(\mathcal{B}))$, and thus, $\text{conv}(\mathcal{G}_{\text{cv}}(\mathcal{B})) \subseteq \pi(\mathcal{Q}(\mathcal{B}))$. Similarly, one can show $\text{ccone}(\mathcal{G}_{\text{cc}}(\mathcal{B})) \subseteq \pi(\text{rec}(\mathcal{Q}(\mathcal{B})))$ (which we omit in this extended abstract). Hence, we have $\mathcal{P}(\mathcal{B}) \subseteq \pi(\mathcal{Q}(\mathcal{B}))$.

In order to prove $\pi(\mathcal{Q}(\mathcal{B})) \subseteq \mathcal{P}(\mathcal{B})$, we show that for all $c \in \mathbb{R}^V$ and $\omega = \max\{\langle c, x\rangle \mid (x, y, z) \in \mathcal{Q}(\mathcal{B})\} \in \mathbb{R} \cup \{\infty\}$ there is some $x^\star \in \mathcal{G}_{\text{cv}}(\mathcal{B})$ with $\langle c, x^\star\rangle = \omega$ if $\omega < \infty$, and that there is some $x^\star \in \mathcal{G}_{\text{cc}}(\mathcal{B})$ with $\langle c, x^\star\rangle > 0$ if $\omega = \infty$. Towards this end, we reformulate (3) and (4) by inequality systems in order to exploit linear programming duality. For each $v \in V \setminus V_T$, let $T^{(v)} \in \mathbb{R}^{\delta^{\text{out}}(v) \times [d(v)]}$ be a matrix with $\pi^{(v)}(z^{(v)}) = T^{(v)} z^{(v)}$ for all $z^{(v)} \in \mathbb{R}^{d(v)}$, and let $A^{(v)} \in \mathbb{R}^{m(v) \times d(v)}$, $b^{(v)} \in \mathbb{R}^{m(v)}$ with $Q^{(v)} = \{z^{(v)} \in \mathbb{R}^{d(v)} \mid A^{(v)} z^{(v)} \leq b^{(v)}\}$. Thus, $\mathcal{Q}(\mathcal{B}) \subseteq \mathbb{R}^V \oplus \mathbb{R}^A$ is the polyhedron defined by the system

$$x_s = 1 \tag{5}$$

$$x_v - y(\delta^{\text{in}}(v)) = 0 \qquad \text{for all } v \in V \setminus \{s\} \tag{6}$$

$$y_{\delta^{\text{out}}(v)} - T^{(v)} z^{(v)} = 0 \qquad \text{for all } v \in V \setminus V_T \tag{7}$$

$$A^{(v)} z^{(v)} - x_v b^{(v)} \leq 0 \qquad \text{for all } v \in V \setminus V_T \tag{8}$$

$$x_v \geq 0 \qquad \text{for all } v \in V \setminus V_T \tag{9}$$

(where (9) turns out to be redundant).

In order to construct an $x^\star$ as required above, let us initialize vectors $x^{(v)} = \mathbb{O} \in \mathbb{R}^V$ for all $v \in V \setminus V_T$ and $x^{(v)} = \mathbb{e}_v \in \mathbb{R}^V$ (the standard unit vector associated with $v \in V$) for all $v \in V_T$ as well as some auxilliary objective function vector $c^\star \in \mathbb{R}^V$ with $c^\star_{V_T} = c_{V_T}$. We process the nodes $v \in V \setminus V_T$ in the reverse order of some topological ordering of the acyclic digraph D (i.e., for each $(v, w) \in A$, node v is processed after node w) in the following way:

1. Let $\zeta^{(v)} = \max\{\langle c^\star_{\mathrm{N}^{\mathrm{out}}(v)}, \tilde{x}\rangle \mid \tilde{x} \in P^{(v)}\} \in \mathbb{R} \cup \{\infty\}$ (recall $P^{(v)} \neq \varnothing$).
2. If $\zeta^{(v)} = \infty$:
   (a) Let $\tilde{x} \in G^{(v)}_{\mathrm{cc}}$ with $\langle c^\star_{\mathrm{N}^{\mathrm{out}}(v)}, \tilde{x}\rangle > 0$.
   (b) Set $x^\star = \sum_{w \in \mathrm{supp}(\tilde{x})} \tilde{x}_w \cdot x^{(w)}$, and terminate the construction of $x^\star$.
3. If $\zeta^{(v)} < \infty$:
   (a) Let $\tilde{x} \in G^{(v)}_{\mathrm{cv}}$ with $\langle c^\star_{\mathrm{N}^{\mathrm{out}}(v)}, \tilde{x}\rangle = \zeta^{(v)}$, and let $\lambda^{(v)} \in \mathbb{R}^{m(v)}_+$ be an optimal dual solution to $\max\{\langle (T^{(v)})^t c^\star_{\mathrm{N}^{\mathrm{out}}(v)}, \tilde{z}\rangle \mid A^{(v)}\tilde{z} \leq b^{(v)}\}$ $(= \zeta^{(v)})$.
   (b) Set $x^{(v)} = \mathbb{e}_v + \sum_{w \in \mathrm{supp}(\tilde{x})} \tilde{x}_w \cdot x^{(w)}$.
   (c) Set $c^\star_v = c_v + \zeta^{(v)}$.

If we did not terminate in Step (2b), then we finally set $x^\star = x^{(s)}$.

From Proposition 1 (and exploiting (S.2)), one deduces by induction that after processing a node $v \in V \setminus V_T$, we have $x^{(v)} \in \mathcal{G}_{\mathrm{cv}}(\mathcal{B}_v)$ and $c^\star_v = \langle c, x^{(v)}\rangle$, if $\zeta^{(v)} < \infty$, and $x^\star \in \mathcal{G}_{\mathrm{cc}}(\mathcal{B})$ with $\langle c, x^\star\rangle > 0$ in case of $\zeta^{(v)} = \infty$. Thus, it suffices to exhibit, in case that we did not terminate in Step (2b), a feasible dual solution to $\max\{\langle c, x\rangle \mid (x, y, z)$ satisfies (5), ..., (9)$\}$ of value $\langle c, x^\star\rangle = c^\star_s$. We have already defined dual variables $\lambda^{(v)} \in \mathbb{R}^{\mathrm{N}^{\mathrm{out}}(v)}_+$ for all inequalities (8). Let us complete these dual variables by setting $\mu^{(v)} = -c^\star_{\mathrm{N}^{\mathrm{out}}(v)}$ for all $v \in V \setminus V_T$ (for equations (7)) and $\nu_v = c^\star_v$ for all $v \in V$ (for equations (5) and (6)). Obviously, the vector (ν, μ, λ) has dual objective function value $\nu_s = c^\star_s$. We omit the calculation showing that (ν, μ, λ) is dually feasible (and revealing the redundancy of (9)) in this extended abstract.

Corollary 1. *Let $\mathcal{B} = (D, (P^{(v)}))$ be a branched polyhedral system with a digraph $D = (V, A)$ with source node $s \in V$, and let $V_T \subseteq V$ be the set of sinks of D. Then, with the polyhedron $\mathcal{Q}(\mathcal{B}) \subseteq \mathbb{R}^V \oplus \mathbb{R}^A$ defined by the system*

$$x_s = 1 \tag{10}$$

$$x_v = y(\delta^{\mathrm{in}}(v)) \qquad \textit{for all } v \in V \setminus \{s\} \tag{11}$$

$$(y_{\delta^{\mathrm{out}}(v)}, x_v) \in \mathrm{homog}(P^{(v)}) \qquad \textit{for all } v \in V \setminus V_T \tag{12}$$

and the orthogonal projection $\pi : \mathbb{R}^V \oplus \mathbb{R}^A \to \mathbb{R}^V$, we have $\mathcal{P}(\mathcal{B}) = \pi(\mathcal{Q}(\mathcal{B}))$.

Remark 2. For each BPS $\mathcal{B}$, the orthogonal projection of $\mathcal{Q}(\mathcal{B})$ (defined by (10), (11), (12)) to the y-space is isomorphic to $\mathcal{Q}(\mathcal{B})$ (due to (10), (11)).

In case that the digraph D is a branching itself, the projection $\mathcal{P}(\mathcal{B})$ of $\mathcal{Q}(\mathcal{B})$ to the x-space is isomorphic to $\mathcal{Q}(\mathcal{B})$ as well. Thus, in this case, from descriptions of the polyhedra $P^{(v)}$ $(v \in V \setminus V_T)$ we even obtain a description of $\mathcal{P}(\mathcal{B})$ in the original space.

Corollary 2. *Let $\mathcal{B} = (D, (P^{(v)}))$ be a branched polyhedral system, whose digraph $D = (V, A)$ itself is a branching rooted at $s \in V$, and let $V_T \subseteq V$ be the set of sinks of D. Then $\mathcal{P}(\mathcal{B})$ is the polyhedron defined by*

$$x_{\mathrm{N}^{\mathrm{out}}(s)} \in P^{(s)} \tag{13}$$

$$(x_{\mathrm{N}^{\mathrm{out}}(v)}, x_v) \in \mathrm{homog}(P^{(v)}) \qquad \textit{for all } v \in V \setminus (\{s\} \cup V_T)\,. \tag{14}$$

Proof. This follows readily from Corollary 1, because for branchings $D = (V, A)$, (11) means $x_w = y_{(v,w)}$ for all $(v, w) \in A$.

4 Applications

4.1 Unions of Polyhedra

The following extended formulation for the convex hull of the union of finitely many polyhedra is basically due to Balas [1] (see also [2, Thm. 5.1]). We show how Balas' result can be established as a consequence of Theorem 1. The slight generalization to the case that the given polyhedra themselves are specified by extended formulations is used, e.g., in [8] in order to construct compact extended formulations of the polytopes associated with the cycles of length $\lfloor \log n \rfloor$ in a complete graph with n nodes. We use $\overline{\mathrm{conv}}(S)$ to denote the topological closure of the convex hull of S.

Corollary 3. *If the non-empty polyhedra $\varnothing \neq P^{(i)} \subseteq \mathbb{R}^n$ (for $i \in [q]$) are projections $P^{(i)} = \pi^{(i)}(Q^{(i)})$ of polyhedra $Q^{(i)} = \{z^{(i)} \in \mathbb{R}^{d(i)} \mid A^{(i)} z^{(i)} \leq b^{(i)}\}$ (with $A^{(i)} \in \mathbb{R}^{m_i \times d(i)}$, $b^{(i)} \in \mathbb{R}^{m_i}$ and linear maps $\pi^{(i)} : \mathbb{R}^{d(i)} \to \mathbb{R}^n$), then the topological closure $P = \overline{\mathrm{conv}}(P^{(1)} \cup \cdots \cup P^{(q)})$ of the convex hull of the union of the polyhedra $P^{(i)}$ is the projection $P = p(Q)$ (with $p(z^{(1)}, \ldots, z^{(q)}, x) = \pi^{(1)}(z^{(1)}) + \cdots + \pi^{(q)}(z^{(q)})$) of the polyhedron $Q \subseteq \mathbb{R}^{d(1)} \times \cdots \times \mathbb{R}^{d(q)} \times \mathbb{R}^q$ defined by $A^{(i)} z^{(i)} \leq x_i b^{(i)}$ for all $i \in [q]$, $\sum_{i \in [q]} x_i = 1$, and $x \geq \mathbb{O}$.*

Proof. We define a BPS $\mathcal{B}$ on the digraph $D = (V, A)$ with node set $V = \{s\} \uplus [q] \uplus \{t_1, \ldots, t_n\}$ and arc set $A = (\{s\} \times [q]) \cup ([q] \times \{t_1, \ldots, t_n\})$, thus, $V_T = \{t_1, \ldots, t_n\}$ is the set of sinks of D. Identifying $\mathbb{R}^n$ with $\mathbb{R}^{V_T}$, the polyhedra associated with the non-sink nodes are $P^{(i)} \subseteq \mathbb{R}^{V_T}$ (for $i \in [q]$) and $P^{(s)} = \mathrm{conv}(G^{(s)}_{\mathrm{cv}}) \subseteq \mathbb{R}^{[q]}$ with $G^{(s)}_{\mathrm{cv}} = \{\mathbb{e}_1, \ldots, \mathbb{e}_q\}$ (and $G^{(s)}_{\mathrm{cc}} = \varnothing$). We choose any finite sets $G^{(i)}_{\mathrm{cv}}, G^{(i)}_{\mathrm{cc}} \subseteq \mathbb{R}^{V_T}$ with $P^{(i)} = \mathrm{conv}(G^{(i)}_{\mathrm{cv}}) + \mathrm{ccone}(G^{(i)}_{\mathrm{cc}})$ for all $i \in [q]$.

For each $x \in \mathbb{R}^V$ we have $x \in \mathcal{G}_{\mathrm{cv}}(\mathcal{B})$ if and only if $x_s = 1$ and there is some $i \in [q]$ with $x_{[q]} = \mathbb{e}_i \in \mathbb{R}^{[q]}$ as well as $x_{V_T} \in G^{(i)}_{\mathrm{cv}}$; similarly, $x \in \mathcal{G}_{\mathrm{cc}}(\mathcal{B})$ if and only if $x_s = 0$, $x_{[q]} = \mathbb{O}$, and $x_{V_T} \in G^{(i)}_{\mathrm{cc}}$ for some $i \in [q]$. Denoting by $[\ldots]_{V_T}$ the orthogonal projection of a set of vectors to the x_{V_T}-space, we thus find $[\mathrm{conv}(\mathcal{G}_{\mathrm{cv}}(\mathcal{B}))]_{V_T} = \mathrm{conv}(\cup_{i=1}^q G^{(i)}_{\mathrm{cv}})$ and $[\mathrm{ccone}(\mathcal{G}_{\mathrm{cc}}(\mathcal{B}))]_{V_T} = \mathrm{ccone}(\cup_{i=1}^q G^{(i)}_{\mathrm{cc}})$, from which $\mathrm{conv}(P^{(1)} \cup \cdots \cup P^{(q)}) \subseteq [\mathcal{P}(\mathcal{B})]_{V_T}$ follows, which implies $\overline{\mathrm{conv}}(P^{(1)} \cup \cdots \cup P^{(q)}) \subseteq [\mathcal{P}(\mathcal{B})]_{V_T}$, because as a (projection of a) polyhedron, $[\mathcal{P}(\mathcal{B})]_{V_T}$ is closed. We further derive $[\mathcal{P}(\mathcal{B})]_{V_T} = [\mathcal{Q}(\mathcal{B})]_{V_T} \subseteq p(Q)$, the equation following from Theorem 1 und the inclusion from the fact that (2) and (3) imply $x_{V_T} = \sum_{i=1}^{q} \pi^{(i)}(z^{(i)})$. Thus, we have $\overline{\text{conv}}(P^{(1)} \cup \cdots \cup P^{(q)}) \subseteq p(Q)$.

To see the reverse inclusion, let $(z^{(1)}, \ldots, z^{(q)}, x) \in Q$. With $I = \{i \in [q] \mid x_i > 0\}$ we find $\frac{1}{x_i} A^{(i)} z^{(i)} \leq b^{(i)}$ from (4) or (8), hence $\frac{1}{x_i} z^{(i)} \in Q^{(i)}$ for all $i \in I$, and thus (as we have $\sum_{i \in I}^{q} x_i = 1$ due to (4) or (8) as well as $x \geq \mathbb{O}$)

$$v = \sum_{i \in I} \pi^{(i)}(z^{(i)}) \in \text{conv}(P^{(1)} \cup \cdots \cup P^{(q)}) \,. \tag{15}$$

If $I = [q]$, we have $p(z^{(1)}, \ldots, z^{(q)}, x) = v$, and (15) proves the claim. Otherwise, for each $i \in [q] \setminus I$ (again from (4) resp. (8)) we find $A^{(i)} z^{(i)} \leq \mathbb{O}$, hence $z^{(i)} \in \text{rec}\,(Q^{(i)})$, and thus $\pi^{(i)}(z^{(i)}) \in \text{rec}\,(P^{(i)})$. Therefore, choosing $\overline{x}^{(i)} \in P^{(i)} \neq \varnothing$ arbitrarily for all $i \in [q] \setminus I$, we have

$$w^{(\varepsilon)} = \sum_{i \in [q] \setminus I} \tfrac{1}{q - |I|} \big(\overline{x}^{(i)} + \tfrac{q - |I|}{\varepsilon} \pi^{(i)}(z^{(i)}) \big) \in \text{conv}(P^{(1)} \cup \cdots \cup P^{(q)}) \tag{16}$$

for all $\varepsilon > 0$. By (15) and (16) we obtain $(1-\varepsilon)v + \varepsilon w^{(\varepsilon)} \in \text{conv}(P^{(1)} \cup \cdots \cup P^{(q)})$ for all $0 < \varepsilon \leq 1$. Due to $p(z^{(1)}, \ldots, z^{(q)}, x) = \lim_{\varepsilon \to 0} \big((1-\varepsilon)v + \varepsilon w^{(\varepsilon)} \big)$, this proves the claim.

4.2 Full Orbitopes

We denote by $\mathcal{M}_{p,q}^{\max}$ the set of 0/1-matrices with p rows and q columns whose columns are sorted in lexicographically non-increasing order. The *full orbitope* is the polytope $\text{O}_{p,q} = \text{conv}(\mathcal{M}_{p,q}^{\max})$. While for the related packing- and partitioning orbitopes (i.e., the convex hulls of those matrices in $\mathcal{M}_{p,q}^{\max}$ with at most and exactly, respectively, one 1-entry in every row) descriptions by (exponentially many) linear inequalities are known [9,6], no such description is available for $\text{O}_{p,q}$. In fact, computer experiments indicate that the facet defining inequalities for $\text{O}_{p,q}$ are extremely complicated. However, applying Corollary 1 to a suitable BCS, we can easily provide a compact extended formulation for $\text{O}_{p,q}$.

Let us define a branched combinatorial system on a digraph $D = (V, A)$ whose node set consists, next to a source node s, of nodes $a(i,j,\ell)$ and $b(i,j,\ell)$ for all $i \in [p]$ and $j, \ell \in [q]$ with $j \leq \ell$. The nodes $a(i,j,\ell)$ and $b(i,j,\ell)$ represent intervals of ones and intervals of zeroes, respectively, to be inserted at positions $(i,j), (i,j+1), \ldots, (i,\ell)$ in a rowwise construction of a matrix $M \in \mathcal{M}_{p,q}^{\max}$ from top to bottom. The arc set A consists of all arcs connecting s to the nodes $a(1,j,\ell)$ and $b(1,j,\ell)$ (with $j, \ell \in [q]$, $j \leq \ell$), as well as all pairs (v, w) with $v \in \{a(i,j,\ell), b(i,j,\ell)\}$ and $w \in \{a(i+1,j,k), b(i+1,k,\ell)\}$ for $i \in [p-1]$ and $j, k, \ell \in [q]$ with $j \leq k \leq \ell$. The set V_T of sinks of D thus consists of all nodes $a(p,j,\ell)$ and $b(p,j,\ell)$ (with $j, \ell \in [q]$, $j \leq \ell$). We define

$$\mathcal{S}^{(s)} = \big\{ \{a(1,1,q)\}, \{b(1,1,q)\} \big\} \cup \big\{ \{a(1,1,k), b(1,k+1,q)\} \mid 1 \leq k \leq q-1 \big\}$$

and, for all $i \in [p-1]$, $j, \ell \in [q]$, $j \leq \ell$, and $v \in \{a(i,j,\ell), b(i,j,\ell)\}$,

$$\mathcal{S}^{(v)} = \big\{ \{a(i+1,j,\ell)\}, \{b(i+1,j,\ell)\} \big\} \\ \cup \big\{ \{a(i+1,j,k), b(i+1,k+1,\ell)\} \mid j \leq k \leq \ell - 1 \big\} \,.$$

Then $\mathcal{C} = (D, (\mathcal{S}^{(v)}))$ is a BCS, and the projection $p : \mathbb{R}^V \to \mathbb{R}^{p\times q}$ with $p(x)_{i,k} = \sum_{1\le j\le k\le \ell\le q} x_{a(i,j,\ell)}$ maps $\mathcal{P}(\mathcal{C}) = \operatorname{conv}\{\chi(F) \in \{0,1\}^V \mid F \subseteq V \text{ feasible for } \mathcal{C}\}$ to $\mathrm{O}_{p,q}$. Because $\operatorname{conv}\left(\{\mathbb{e}_1, \mathbb{e}_2\} \cup \{\mathbb{e}_{2r-1} + \mathbb{e}_{2r} \mid r \in \{2,\dots,n\}\}\right)$ equals $\{x \in \mathbb{R}^{2n}_+ \mid x_{2r-1} - x_{2r} = 0 \text{ for all } r \in \{2,\dots,n\}, 2x_1 + 2x_2 + \sum_{r=2}^{2n} x_r = 2\}$, one obtains linear descriptions of the polytopes $P^{(s)}$ and $P^{(v)}$ that yield the following extended formulation for $\mathrm{O}_{p,q}$ via Corollary 1.

Theorem 2. *The full orbitope $\mathrm{O}_{p,q}$ is a projection (obtained by first projecting orthogonally to the x-space and then applying the projection p defined above) of the polyhedron defined by $y \geq \mathbb{O}$ and*

$$x_v - y(\delta^{\text{in}}(v)) = 0 \qquad (17)$$

$$2y_{(s,a(1,1,q))} + 2y_{(s,b(1,1,q))} + \textstyle\sum_{k=1}^{q-1}\left(y_{(s,a(1,1,k))} + y_{(s,b(1,k+1,q))}\right) = 2$$

$$y_{(s,a(1,1,k))} - y_{(s,b(1,k+1,q))} = 0 \qquad (18)$$

$$2y_{v,i+1,j,\ell} + \textstyle\sum_{k=j}^{\ell-1}\left(y_{(v,a(i+1,j,k))} + y_{(v,b(i+1,k+1,\ell))}\right) - 2x_v = 0 \qquad (19)$$

$$y_{(v,a(i+1,j,k))} - y_{(s,b(i+1,k+1,\ell))} = 0\,, \qquad (20)$$

where, for layout reasons, we use $y_{v,i+1,j,\ell} = y_{(v,a(i+1,j,\ell))} + y_{(v,b(i+1,j,\ell))}$, (17) has to be included into the system for all $v \in V\setminus\{s\}$, (18) for all $1 \le k \le q-1$, as well as (19) and (20) for all $i \in [p-1]$, $j, \ell \in [q]$ with $j \le \ell$, $v \in \{a(i,j,\ell), b(i,j,\ell)\}$, and $1 \le k \le \ell - 1$ (where the latter only refers to (20)).

4.3 Dynamic Programming

We briefly show how the framework developed in Martin, Rardin, and Campbell [11] for deriving extended formulations for combinatorial optimization problems from dynamic programming algorithms is related to a special case of Theorem 1 for certain branched combinatorial systems.

The dynamic programing algorithms considered in [11] work by searching hyperpaths in a directed hypergraph $\mathcal{H} = (V, H)$ on a (finite) *state space* $V = [n]$, where the directed hyperarcs in H are of the form (v, S) with $v \in V$ and $S \subseteq \{w \in V \mid w > v\}$. Furthermore, it is assumed that, for each $w \in V \setminus \{1\}$, there is at least one hyperarc $(v, S) \in H$ with $w \in S$, i.e., $s = 1$ is the only source of the directed hypergraph. The set $V_T \subseteq V$ of *boundary states* is the set of all $v \in V$ for which there is no hyperarc $(v, S) \in H$. And finally, there needs to be a (finite) *reference set* $R(v) \neq \varnothing$ for each state $v \in V$ such that $R(w) \subseteq R(v)$ for all $w \in S$ and $R(w) \cap R(w') = \varnothing$ for all $w, w' \in S$ with $w \neq w'$ is satisfied for all $(v, S) \in H$. A *hyperpath* in the hypergraph is a subset L of hyperarcs that contains exactly one hyperarc $(1, S)$ and for each $v \in V \setminus \{1\}$ incident to some hyperarc in L, there is exactly one hyperarc $(u, S) \in L$ with $v \in S$, and, if $v \notin V_T$, there is exactly one $(v, S) \in L$.

In this setting, it is proved in [11] that the convex hull of the characteristic vectors (in the hyperarc space) of hyperpaths is described by the system

$$\textstyle\sum_{(1,S)\in H} z_{(1,S)} = 1 \tag{21}$$

$$\textstyle\sum_{(v,S)\in H} z_{(v,S)} = \sum_{(u,S')\in H\,:\,v\in S'} z_{(u,S')} \qquad \text{for all } v \in V \setminus (\{s\} \cup V_T) \tag{22}$$

$$z \geq \mathbb{O}\,. \tag{23}$$

System (21), (22), (23) arises from Theorem 1 in the following way. We construct a BCS on the acyclic digraph $D = (V, A)$ on the node set V of the hypergraph $\mathcal{H}$, where A consists of all arcs (v, w) for which there is some $(v, S) \in H$ with $w \in S$. Clearly, V_T (the set of boundary states) is the set of all sinks of D. Defining $\mathcal{S}^{(v)} = \{S \subseteq V \mid (v, S) \in H\}$ for every $v \in V \setminus V_T$, we obtain a BCS $\mathcal{C}$ (for this, the existence of the reference sets $R(v)$ is crucial) whose feasible sets are exactly the node sets of hyperpaths in $\mathcal{H}$. Describing the polytopes $P^{(v)} = \operatorname{conv}\{\chi(S) \mid S \in \mathcal{S}^{(v)}\}$, for all $v \in V \setminus V_T$, by their trivial extended formulations $P^{(v)} = \pi^{(v)}(Q^{(v)})$ with $Q^{(v)} = \{z^{(v)} \in \mathbb{R}_+^{\mathcal{S}^{(v)}} \mid \sum_{S\in\mathcal{S}^{(v)}} z_S^{(v)} = 1\}$ and $\pi^{(v)}(z^{(v)}) = \sum_{S\in\mathcal{S}^{(v)}} z_S^{(v)} \cdot \chi(S)$, Theorem 1 yields an extended formulation of the convex hull $\mathcal{P}(\mathcal{C})$ of all node sets of hyperpaths in $\mathcal{H}$ that is isomorphic to (21), (22), (23).

As the feasible sets for the BCS $\mathcal{C}$ are the node sets of the hyperpaths in $\mathcal{H}$ (representing the set of states used in order to construct a solution during the dynamic programming), quite often, the polytope associated with the combinatorial optimization problem solved by the dynamic programming algorithm is a projection of $\mathcal{P}(\mathcal{C})$, and thus, (21), (22), (23) provides an extended formulation for that polytope. In principle, the concept of BCS allows more flexibility here, because we can choose other representations of the polytopes $P^{(v)}$ than their trivial extended formulations. This might be advantageous, e.g, if there are states v for which the number of hyperedges (v, S) is much larger than the number of states. For the full orbitope example in Section 4.2 this is not the case. In fact, one could derive the extended formulation in Theorem 2 also within the hypergraph framework (and eliminating some redundant variables afterwards).

4.4 Nested Combinatorial Problems

Let $N_1, \ldots, N_q$ be pairwise disjoint finite sets, $\mathcal{S}^{(i)} \neq \varnothing$ a set of subsets of N_i for each $i \in [q]$, and $\mathcal{S}^{(0)} \neq \varnothing$ a set of subsets of $[q]$. Denoting $N = N_1 \cup \cdots \cup N_q$, we define $\mathcal{S} = \big\{S \subseteq N \mid S \cap N_i \in \mathcal{S}^{(i)} \cup \{\varnothing\}$ for all $i \in [q]$, $\{i \in [q] \mid S \cap N_i \neq \varnothing\} \in \mathcal{S}^{(0)}\big\}$. In order to derive linear formulations of the polytope $P = \operatorname{conv}\{\chi(S) \mid S \in \mathcal{S}\} \subseteq \mathbb{R}^N$, let us construct a BCS on the digraph $D = (V, A)$ with $V = \{0\} \uplus [q] \uplus N$ and A consisting of all arcs $(0, i)$ $(i \in [q])$ and (i, t_j) $(i \in [q], j \in N_i)$. Thus D is a branching rooted at $s = 0$, the set of sinks is $V_T = N$, and $\mathcal{C} = (D, (\mathcal{S}^{(v)}))$ indeed forms a BCS. From linear descriptions $P^{(0)} = \operatorname{conv}\{\chi(S) \mid S \in \mathcal{S}^{(0)}\} = \{x \in \mathbb{R}^q \mid A^{(0)}x \leq b^{(0)}\}$ and $P^{(i)} = \operatorname{conv}\{\chi(S) \mid S \in \mathcal{S}^{(i)}\} = \{x \in \mathbb{R}^{N_i} \mid A^{(i)}x \leq b^{(i)}\}$ for all $i \in [q]$ of the polytopes associated with the given set systems, we conclude via Corollary 2 that

$$A^{(0)}x \le b^{(0)} \tag{24}$$
$$A^{(i)}x_{N_i} - x_i b^{(i)} \le \mathbb{O} \qquad \text{for all } i \in [q] \tag{25}$$
$$x \ge \mathbb{O} \tag{26}$$

is an extended formulation in $\mathbb{R}^{V\setminus\{0\}}$ for P (via orthogonal projection to the x_N-space).

Let us further consider the case that for each $i \in [q]$, a linear equation $\langle a^{(i)}, x\rangle = 1$ is valid for the polytope $P^{(i)}$. In this case, (25) implies $x_i = \langle a^{(i)}, x_{N_i}\rangle$ for all $i \in [q]$. Thus, we find that $P \subseteq \mathbb{R}^N$ is defined by

$$\textstyle\sum_{i=1}^{q} \langle a^{(i)}, x_{N_i}\rangle \cdot A^{(0)}_{\star,i} \le b \tag{27}$$
$$A^{(i)}x_{N_i} - \langle a^{(i)}, x_{N_i}\rangle b^{(i)} \le \mathbb{O} \qquad \text{for all } i \in [q] \tag{28}$$
$$\langle a^{(i)}, x_{N_i}\rangle \ge 0 \qquad \text{for all } i \in [q]\,. \tag{29}$$

We consider two particular examples of such nested combinatorial systems. The first one is used by [8] in the construction of compact extended formulations of the polytopes associated with the matchings of size $\lfloor \log n \rfloor$ in a complete graph with n nodes. Let $G = (W, E)$ be a 2ℓ-partite graph with $W = W_1 \uplus \cdots \uplus W_{2\ell}$ and $e \not\subseteq W_i$ for all $e \in E$ and $i \in [2\ell]$, and denote by $\mathcal{M}$ the set of all matchings $M \subseteq E$ with $|M| = \ell$ and $|W(M) \cap W_i| = 1$ for all $i \in [2\ell]$ (where $W(M) = \cup_{e\in M} e$ is the set of nodes matched by M).

In order to obtain a linear description of the polytope $P = \operatorname{conv}\{\chi(M) \mid M \in \mathcal{M}\}$ from a BCS as above, we identify $[q]$ with the set $E_{2\ell}$ of edges of the complete graph $K_{2\ell}$ with node set $[2\ell]$, and let $N_{\{i,j\}} = (W_i : W_j)$ for all $\{i,j\} \in E_{2\ell}$. Choosing $\mathcal{S}^{(0)}$ as the set of all perfect matchings in $K_{2\ell}$ and $\mathcal{S}^{(\{i,j\})} = \big\{\{e\} \mid e \in N_{\{i,j\}}\big\}$ for all $\{i,j\} \in E_{2\ell}$, we thus obtain a BCS as above with $\mathcal{S} = \mathcal{M}$. Since we have

$$P^{(0)} = \{x \in \mathbb{R}_+^{E_{2\ell}} \mid x(\delta(i)) = 1 (i \in [2\ell]), x(\delta(I)) \ge 1 (I \subseteq [2\ell], |I| \text{ odd})\}$$

(due to Edmonds [5]) and $P^{(\{i,j\})} = \{x \in \mathbb{R}_+^{N_{\{i,j\}}} \mid x(N_{\{i,j\}}) = 1\}$ for all $\{i,j\} \in E_{2\ell}$, the system (27), (28), (29) provides the linear description

$$x(\delta(W_i)) = 1 \qquad \text{for all } i \in [2\ell]$$
$$x(\delta(\cup_{i\in I} W_i)) \ge 1 \qquad \text{for all } I \subseteq [2\ell], |I| \text{ odd}$$
$$x \ge \mathbb{O}$$

of the polytope $P \subseteq \mathbb{R}^E$ associated with the matchings in $\mathcal{M}$.

In order to modify this example to a second one (mentioned in the Introduction), we assume that there is a number p with $|W_i| = p$ for all $i \in [2\ell]$, and we replace the sets $\mathcal{S}^{(\{i,j\})}$ (which one can consider as the sets of matchings of size one in the bipartite subgraph $G[W_i \cup W_j]$ of G induced by $W_i \cup W_j$) by the sets $\mathcal{S}'^{(\{i,j\})}$ of all perfect matchings in $G[W_i \cup W_j]$. Clearly, instead of $P^{(\{i,j\})}$ we now use $P'^{(\{i,j\})} = \{x \in \mathbb{R}_+^{N_{\{i,j\}}} \mid x(\delta(k)) = 1 \text{ for all } k \in W_i \cup W_j\}$

for all $\{i,j\} \in E_{2\ell}$. As for each $P'^{(\{i,j\})}$ the equation $x(W_i : W_j) = p$ holds, the system (27), (28), (29) for the modified example yields the linear description

$$\begin{aligned} x(\delta(W_i)) &= p && \text{for all } i \in [2\ell] \\ x(\delta(\cup_{i\in I} W_i)) &\geq p && \text{for all } I \subseteq [2\ell], |I| \text{ odd} \\ p \cdot x(v : W_j) - x(W_i : W_j) &= 0 && \text{for all } i,j \in [2\ell], \{i,j\} \in E, v \in W_i \\ x &\geq \mathbb{O} \end{aligned}$$

of the polytope $P' = \operatorname{conv}\{\chi(M) \mid M \in \mathcal{M}'\}$, where $\mathcal{M}'$ is the set of all perfect matchings in G that induce, for each $i,j \in [q]$ with $i \neq j$, an empty or a perfect matching in $G[W_i \cup W_j]$.

4.5 Stable Set Polytopes

Let $G = (W,E)$ be an undirected graph, and $\operatorname{stab}(G) = \operatorname{conv}\{\chi(S) \mid S \subseteq W \text{ stable in } G\} \subseteq \mathbb{R}^W$ the *stable set polytope* of G (where *stable* means that no two nodes in S are adjacent). Let $W = W_1 \uplus \cdots \uplus W_k$ be a partitioning of the node set W into nonempty subsets. We define, for each $i \in [k]$, the *boundary* $\partial W_i = \{w \in W_i \mid \{w,w'\} \in E \text{ for some } w' \in W \setminus W_i\}$ of W_i, as well as $\mathcal{U} = \{\varnothing \neq U \subseteq W \mid U \text{ stable in } G, U \subseteq \partial W_i \text{ for some } i \in [k]\} \uplus \{u_1,\ldots,u_k\}$. For $U \in \mathcal{U}$, let $i(u_i) = i$ for all $i \in [k]$ and let $i(U)$ be the index with $U \subseteq W_{i(U)}$ for all $U \in \mathcal{U} \setminus \{u_1,\ldots,u_k\}$. Moreover, let $\mathcal{H} = (\mathcal{U},\mathcal{K})$ be the undirected graph on $\mathcal{U}$ with, for all $U,U' \in \mathcal{U}$ with $U \neq U'$, $\{U,U'\} \in \mathcal{K}$ if and only if $i(U) = i(U')$ or $U,U' \notin \{u_1,\ldots,u_k\}$ and $(U : U') \neq \varnothing$).

We construct a BCS on the acyclic digraph $D = (V,A)$ with $V = \{s\} \uplus \mathcal{U} \uplus W$, and A containing the arcs (s,U) for all $U \in \mathcal{U}$, as well as the arcs (U,w) for all $U \in \mathcal{U}$, $w \in W_{i(U)}$. Thus, s is the unique source of D, and the set of sinks of D is $V_T = W$. With $\mathcal{S}^{(s)} = \{S \subseteq \mathcal{U} \mid S \text{ stable in } \mathcal{H}\}$ and $\mathcal{S}^{(U)} = \{S \subseteq W_{i(U)} \mid S \text{ stable in } G[W_{i(U)}], S \cap \partial W_{i(U)} = U\}$ for all $U \in \mathcal{U} \setminus \{u_1,\ldots,u_k\}$, as well as $\mathcal{S}^{(u_i)} = \{S \subset W_i \mid S \text{ stable in } G[W_i], S \cap \partial W_i = \varnothing\}$ for all $i \in [k]$, we obtain a BCS for which the intersections of the feasible sets with W are the stable sets in G. Thus, from extended formulations for $\operatorname{stab}(\mathcal{H})$ and $\operatorname{stab}(G[W_i])$ for all $i \in [k]$, one obtains via Theorem 1 an extended formulation for $\operatorname{stab}(G)$ (note that $P^{(U)} = \operatorname{conv}\{\chi(S) \mid S \in \mathcal{S}^{(U)}\}$ is a face of $\operatorname{stab}(G[W_{i(U)}])$ for each $U \in \mathcal{U}$). If $|\mathcal{U}|$ is bounded polynomially in the size of G, then the size of the constructed extended formulation for $\operatorname{stab}(G)$ is bounded by a polynomial in the size of G plus, of course, the sum of the sizes of the used extended formulations for $\operatorname{stab}(\mathcal{H})$ and $\operatorname{stab}(G[W_i])$ $(i \in [k])$. This is, e.g., the case if every boundary ∂W_i $(i \in [k])$ can be covered by a constant number of cliques.

The basic idea in the above construction is to consider two stable sets in W_i equivalent if they agree on the boundary ∂W_i of W_i. One can also work with a weaker equivalence relation by first partitioning the boundaries ∂W_i into cliques $\partial W_i = W_i(1) \uplus \cdots \uplus W_i(\ell_i)$ such that every two nodes in the same clique are adjacent to the same nodes outside W_i, and then considering two stable sets $S,S' \subseteq W_i$ equivalent in case they satisfy $|S \cap W_i(j)| = |S' \cap W_i(j)|$ for all $j \in [\ell_i]$.

This can result in a significantly smaller BCS. It provides, e.g., an alternative construction for the final step in the derivation of an extended formulation for the stable set polytopes of claw-free graphs due to Faenza, Oriolo, and Stauffer [7].

Acknowledgements. We are greatful to the referees for careful reading and some very helpful remarks.

References

1. Balas, E.: Disjunctive programming and a hierarchy of relaxations for discrete optimization problems. SIAM J. Algebraic Discrete Methods 6(3), 466–486 (1985)
2. Conforti, M., Cornuéjols, G., Zambelli, G.: Extended Formulations in Combinatorial Optimization. Technical Report (2009)
3. Conforti, M., Cornuéjols, G., Zambelli, G.: Polyhedral approaches to mixed integer linear programming. In: Jünger, M., Liebling, T., Naddef, D., Nemhauser, G., Pulleyblank, W., Reinelt, G., Rinaldi, G., Wolsey, L. (eds.) 50 Years of Integer Programming 1958-2008. Springer, Heidelberg (2010)
4. Conforti, M., Di Summa, M., Eisenbrand, F., Wolsey, L.: Network formulations of mixed-integer programs. Math. Oper. Res. 34, 194–209 (2009)
5. Edmonds, J.: Maximum matching and a polyhedron with $0, 1$ vertices. Journal of Research of the National Bureau of Standards 69B, 125–130 (1965)
6. Faenza, Y., Kaibel, V.: Extended formulations for packing and partitioning orbitopes. Math. Oper. Res. 34(3), 686–697 (2009)
7. Faenza, Y., Oriolo, G., Stauffer, G.: The hidden matching structure of the composition of strips: a polyhedral perspective. In: 14th Aussois Workshop on Combinatorial Optimization, Aussois (January 2010)
8. Kaibel, V., Pashkovich, K., Theis, D.O.: Symmetry matters for the sizes of extended formulations. In: Eisenbrand, F., Shepherd, B. (eds.) IPCO 2010. LNCS, vol. 6080, pp. 135–148. Springer, Heidelberg (2010)
9. Kaibel, V., Pfetsch, M.: Packing and partitioning orbitopes. Math. Program. 114(1, Ser. A), 1–36 (2008)
10. Margot, F.: Composition de Polytopes Combinatoires: Une Approche par Projection. Ph.D. thesis, École Polytechnique Fédérale de Lausanne (1994)
11. Martin, R.K., Rardin, R.L., Campbell, B.A.: Polyhedral characterization of discrete dynamic programming. Oper. Res. 38(1), 127–138 (1990)
12. Schaffers, M.: On Links Between Graphs with Bounded Decomposability, Existence of Efficient Algorithms, and Existence of Polyhedral Characterizations. Ph.D. thesis, Université Catholique de Louvain (1994)

Hitting Diamonds and Growing Cacti[⋆]

Samuel Fiorini[1], Gwenaël Joret[2,⋆⋆], and Ugo Pietropaoli[3,⋆⋆⋆]

[1] Université Libre de Bruxelles (ULB), Département de Mathématique, CP 216, B-1050 Brussels, Belgium
`sfiorini@ulb.ac.be`

[2] Université Libre de Bruxelles (ULB), Département d'Informatique, CP 212, B-1050 Brussels, Belgium
`gjoret@ulb.ac.be`

[3] Università di Roma "Tor Vergata", Dipartimento di Ingegneria dell'Impresa, Rome, Italy
`pietropaoli@disp.uniroma2.it`

Abstract. We consider the following NP-hard problem: in a weighted graph, find a minimum cost set of vertices whose removal leaves a graph in which no two cycles share an edge. We obtain a constant-factor approximation algorithm, based on the primal-dual method. Moreover, we show that the integrality gap of the natural LP relaxation of the problem is $\Theta(\log n)$, where n denotes the number of vertices in the graph.

1 Introduction

Graphs in this paper are finite, undirected, and may contain parallel edges but no loops. We study the following combinatorial optimization problem: given a vertex-weighted graph, remove a minimum cost subset of vertices so that all the cycles in the resulting graph are edge-disjoint. We call this problem the *diamond hitting set problem*, because it is equivalent to covering all subgraphs which are diamonds with a minimum cost subset of vertices, where a *diamond* is any subdivision of the graph consisting of three parallel edges.

The diamond hitting set problem can be thought of as a generalization of the vertex cover and feedback vertex set problems: Suppose you wish to remove a minimum cost subset of vertices so that the resulting graph has no pair of vertices linked by k internally disjoint paths. Then, for $k = 1$ and $k = 2$, this is respectively the vertex cover problem and feedback vertex set problem, while for $k = 3$ this corresponds to the diamond hitting set problem.

⋆ This work was supported by the "Actions de Recherche Concertées" (ARC) fund of the "Communauté française de Belgique".

⋆⋆ G. Joret is a Postdoctoral Researcher of the Fonds National de la Recherche Scientifique (F.R.S.–FNRS).

⋆⋆⋆ This work was done while U.P. was at Département de Mathématique - Université Libre de Bruxelles as a Postdoctoral Researcher of the F.R.S.–FNRS.

It is well-known that both the vertex cover and feedback vertex set problems admit constant-factor approximation algorithms[1]. Hence, it is natural to ask whether the same is true for the diamond hitting set problem. Our main contribution is a positive answer to this question.

1.1 Background and Related Work

Although there exists a simple 2-approximation algorithm for the vertex cover problem, there is strong evidence that approximating the problem with a factor of $2 - \varepsilon$ might be hard, for every $\varepsilon > 0$ [8]. It should be noted that the feedback vertex set and diamond hitting set problems are at least as hard to approximate as the vertex cover problem, in the sense that the existence of a ρ-approximation algorithm for one of these two problems implies the existence of a ρ-approximation algorithm for the vertex cover problem, where ρ is a constant.

Concerning the feedback vertex set problem, the first approximation algorithm is due to Bar-Yehuda, Geiger, Naor, and Roth [2] and its approximation factor is $O(\log n)$. Later, 2-approximation algorithms have been proposed by Bafna, Berman, and Fujito [1], and Becker and Geiger [3]. Chudak, Goemans, Hochbaum and Williamson [4] showed that these algorithms can be seen as deriving from the primal-dual method (see for instance [9,7]). Starting with an integer programming formulation of the problem, these algorithms simultaneously construct a feasible integral solution and a feasible dual solution of the linear programming relaxation, such that the values of these two solutions are within a constant factor of each other.

These algorithms also lead to a characterization of the integrality gap[2] of two different integer programming formulations of the problem, as we now explain. Let $\mathcal{C}(G)$ denote the collection of all the cycles C of G. A natural integer programming formulation for the feedback vertex set problem is as follows:

$$\begin{aligned} \text{Min} \quad & \sum_{v \in V(G)} c_v \, x_v \\ \text{s.t.} \quad & \sum_{v \in V(C)} x_v \geqslant 1 \qquad \forall C \in \mathcal{C}(G) \\ & x_v \in \{0,1\} \qquad \forall v \in V(G). \end{aligned} \tag{1}$$

(Throughout, c_v denotes the (non-negative) cost of vertex v.) The algorithm of Bar-Yehuda et al. [2] implies that the integrality gap of this integer program is $O(\log n)$. Later, Even, Naor, Schieber, and Zosin [5] proved that its integrality gap is also $\Omega(\log n)$.

[1] A ρ-approximation algorithm for a minimization problem is an algorithm that runs in polynomial time and outputs a feasible solution whose cost is no more than ρ times the cost of the optimal solution. The number ρ is called the approximation factor.

[2] The integrality gap of an integer programming formulation is the worst-case ratio between the optimum value of the integer program and the optimum value of its linear relaxation.

A better formulation has been introduced by Chudak et al. [4]. For $S \subseteq V(G)$, denote by $E(S)$ the set of the edges of G having both ends in S, by $G[S]$ the subgraph of G induced by S, and by $d_S(v)$ the degree of v in $G[S]$. Then, the following is a formulation for the feedback vertex set problem:

$$\begin{aligned} \text{Min} \quad & \sum_{v \in V(G)} c_v \, x_v \\ \text{s.t.} \quad & \sum_{v \in S} (d_S(v) - 1) x_v \geqslant |E(S)| - |S| + 1 \qquad \forall S \subseteq V(G) : E(S) \neq \varnothing \qquad (2) \\ & x_v \in \{0,1\} \qquad \forall v \in V(G). \end{aligned}$$

Chudak et al. [4] showed that the integrality gap of this integer program asymptotically equals 2. Constraints (2) derive from the simple observation that the removal of a feedback vertex set X from G generates a forest having at most $|G| - |X| - 1$ edges. Notice that the covering inequalities (1) are implied by (2).

1.2 Contribution and Key Ideas

First, we obtain a $O(\log n)$-approximation algorithm for the diamond hitting set problem, leading to a proof that the integrality gap of the natural LP formulation is $\Theta(\log n)$. Then, we develop a 9-approximation algorithm. Both the $O(\log n)$- and 9-approximation algorithm are based on the primal-dual method.

Our first key idea is contained in the following observation: every simple graph of order n and minimum degree at least 3 contains a $O(\log n)$-size diamond. This directly yields a $O(\log n)$-approximation algorithm for the diamond hitting set problem, in the unweighted case. However, the weighted case requires more work.

Our second key idea is to generalize constraints (2) by introducing 'sparsity inequalities', that enable us to derive a constant-factor approximation algorithm for the diamond hitting set problem: First, by using reduction operations, we ensure that every vertex of G has at least three neighbors. Then, if G contains a diamond with at most 9 edges, we raise the dual variable of the corresponding covering constraint. Otherwise, no such small diamond exists in G, and we can use this information to select the right sparsity inequality, and raise its dual variable. This inequality would not be valid in case G contained a small diamond.

The way we use the non-existence of small diamonds is perhaps best explained via an analogy with planar graphs: An n-vertex planar simple graph G has at most $3n - 6$ edges. However, if we know that G has no small cycle, then this upper bound can be much strengthened. (For instance, if G is triangle-free then G has at most $2n - 4$ edges.)

We remark that this kind of local/global trade-off did not appear in the work of Chudak et al. [4] on the feedback vertex set problem, because the cycle covering inequalities are implied by their more general inequalities. In our case, the covering inequalities and the sparsity inequalities form two incomparable classes of inequalities, and examples show that the sparsity inequalities alone are not enough to derive a constant-factor approximation algorithm.

This extended abstract is organized as follows. Preliminaries are given in Section 2. Then, in Section 3, we define some reduction operations that allow us to work mainly with graphs where each vertex has at least three distinct neighbors. Next, in Section 4, we deal with the unweighted version of the diamond hitting set problem and provide a simple $O(\log n)$-approximation algorithm. In Section 5, we turn to the weighted version of the problem. We sketch a $O(\log n)$-approximation algorithm. It turns out that the integrality gap of the natural formulation of the problem is $\Theta(\log n)$. Finally, in Section 6, we introduce the sparsity inequalities, prove their validity and sketch a 9-approximation algorithm. Due to length restrictions, most proofs and details are not included in this extended abstract. A full version of the paper can be found in [6].

2 Preliminaries

A *cactus* is a connected graph where each edge belongs to at most one cycle. Equivalently, a connected graph is a cactus if and only if each of its blocks is isomorphic to either K_2 or a cycle. Thus, a connected graph is a cactus if and only if it does not contain a diamond as a subgraph. A graph without diamonds is called a *forest of cacti* (see Figure 1).

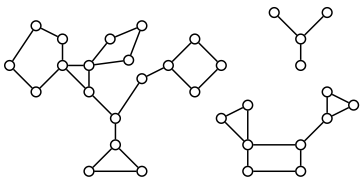


Fig. 1. A forest of cacti

A *diamond hitting set* (or simply *hitting set*) of a graph is a subset of vertices that hits every diamond of the graph. A *minimum (diamond) hitting set* of a weighted graph is a hitting set of minimum total cost, and its cost is denoted by OPT.

Let $\mathcal{D}(G)$ denote the collection of all diamonds contained in G. From the standard IP formulation for a covering problem, we obtain the following LP relaxation for the diamond hitting set problem:

$$\begin{aligned} \text{Min} \quad & \sum_{v \in V(G)} c_v \, x_v \\ \text{s.t.} \quad & \sum_{v \in V(D)} x_v \geqslant 1 \qquad \forall D \in \mathcal{D}(G) \\ & x_v \geqslant 0 \qquad \forall v \in V(G). \end{aligned} \tag{3}$$

We call inequalities (3) *diamond inequalities*.

3 Reductions

In this section, we define two reduction operations on graphs: First, we define the 'shaving' of an arbitrary graph, and then introduce a 'bond reduction' operation for shaved graphs.

The aim of these two operations is to modify a given graph so that the following useful property holds: each vertex either has at least three distinct neighbors, or is incident to at least three parallel edges.

3.1 Shaving a Graph

Let G be a graph. Every block of G is either isomorphic to K_1, K_2, a cycle, or contains a diamond. Mark every vertex of G that is included in a block containing a diamond. The *shaving* of G is the graph obtained by removing every unmarked vertex from G. A graph is *shaved* if all its vertices belong to a block containing a diamond. Observe that, in particular, every endblock[3] of a shaved graph contains a diamond.

3.2 Reducing a Bond

A *bond* of a graph G is a connected subgraph $Q \subseteq G$ equipped with two distinguished vertices v, w (called *ends*) satisfying the following requirements:

- Q is a cactus with at least two blocks;
- the block-graph of Q is a path;
- v and w belong to distinct endblocks of Q;
- v and w are not adjacent in Q;
- $Q - \{v, w\}$ is a non-empty component of $G - \{v, w\}$, and
- Q contains all the edges in G between $\{v, w\}$ and $V(Q) - \{v, w\}$.

Observe that Q is "almost" an induced subgraph of G, since Q includes every edge of G between vertices of Q, except those between v and w (if any). The vertices in $V(Q) - \{v, w\}$ are said to be the *internal vertices* of Q. The bond Q is *simple* if Q is isomorphic to a path, *double* otherwise.

Let G be a shaved graph. A vertex u of G is *reducible* if u has exactly two neighbors in G, and there are at most two parallel edges connecting u to each of its neighbors. The *bond reduction* operation is defined as follows. Let u be a reducible vertex and let Q_u be an inclusion-wise maximal bond of G containing u, with ends v and w. (Observe that such a bond exists by our hypothesis on u; moreover, it might not be unique.) Then, remove from G every internal vertex of Q_u, and add one or two edges between v and w, depending on whether Q_u is simple or double. In the latter case, the two new parallel edges are said to be *twins*. See Figure 2 for an illustration of the operation. Observe that the resulting graph is also a shaved graph.

[3] We recall that the block-graph of G has the blocks of G and the cutvertices of G as vertices, a block and a cutvertex are adjacent if the former contains the latter. This graph is always acyclic. An endblock of G is a vertex of the block-graph with degree at most one.

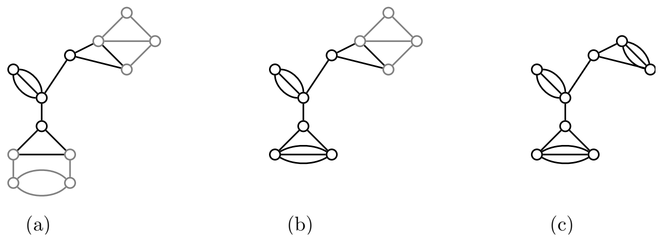


Fig. 2. (a) A shaved graph G with two maximal bonds (in grey). (b) Reduction of the first bond. (c) Reduction of the second bond. The graph is now reduced.

A crucial property of the bond reduction operation is that, when applying it iteratively, we never include in the bond to be reduced any edge coming from previous bond reductions [6, Lemma 3.1].

A *reduced graph* $\widetilde{G}$ of G is any graph obtained from G by iteratively applying a bond reduction, as long as there is a reducible vertex (see Figure 2). We remark that there is not necessarily a unique reduced graph of G (consider for instance K_3 where two edges are doubled).

4 A $O(\log n)$-Approximation Algorithm in the Unweighted Case

As a first step, we show that every reduced graph contains a diamond of size $O(\log n)$ [6, Lemmas 4.1 and 4.4].

Lemma 1. *Every simple n-vertex graph with minimum degree at least 3 contains a diamond of size at most $6 \log_{3/2} n + 8$. Moreover, such a diamond can be found in polynomial time. The same holds for reduced graphs.*

Our algorithm for the diamond hitting set problem on unweighted graphs is described in Algorithm 1.

Algorithm 1. A $O(\log n)$-approximation algorithm for unweighted graphs.

```
– $X \leftarrow \varnothing$
– While $X$ is not a hitting set of $G$, repeat the following steps:
    • Compute a reduced graph $\widetilde{G}$ of $G - X$
    • Find a diamond $\widetilde{D}$ in $\widetilde{G}$ of size at most $6 \log_{3/2} |\widetilde{G}| + 8$        (using Lemma 1)
    • Include in $X$ all vertices of $\widetilde{D}$
```

The algorithm relies on the simple fact that every hitting set of a reduced graph $\widetilde{G}$ of a graph G is also a hitting set of G itself. The set of diamonds computed by the algorithm yields a collection $\mathcal{D}$ of pairwise vertex-disjoint diamonds in G. In particular, the size of a minimum hitting set is at least $|\mathcal{D}|$. For each diamond in $\mathcal{D}$, at most $6 \log_{3/2} n + 8$ vertices were added to the hitting set X. Hence, the approximation factor of the algorithm is $O(\log n)$.

5 A $O(\log n)$-Approximation Algorithm

The present section is devoted to a $O(\log n)$-approximation algorithm for the diamond hitting set problem in the weighted case, which is based on the primal-dual method. We start by defining, in Section 5.1, the actual LP relaxation of the problem used by the algorithm, together with its dual. Then, in Section 5.2, we sketch our approximation algorithm. Details and proofs can be found in the full version of this paper [6, Section 5].

5.1 The Working LP and Its Dual

Our approximation algorithm for the weighted case is based on the natural LP relaxation for the diamond hitting set problem, given on page 194. To simplify the presentation, we do not directly resort to that LP relaxation but to a possibly weaker relaxation that is constructed during the execution of the algorithm, that we call the *working LP*. At each iteration, an inequality is added to the working LP. These inequalities, that we name *blended diamond inequalities*, are all implied by diamond inequalities (3). The final working LP reads:

$$\begin{array}{llll} \text{(LP)} & \text{Min} & \displaystyle\sum_{v \in V(G)} c_v\, x_v & \\ & \text{s.t.} & \displaystyle\sum_{v \in V(G)} a_{i,v}\, x_v \geqslant \beta_i & \forall i \in \{1, \ldots, k\} \\ & & x_v \geqslant 0 & \forall v \in V(G), \end{array}$$

where k is the total number of iterations of the algorithm. The dual of (LP) is:

$$\begin{array}{llll} \text{(D)} & \text{Max} & \displaystyle\sum_{i=1}^{k} \beta_i\, y_i & \\ & \text{s.t.} & \displaystyle\sum_{i=1}^{k} a_{i,v}\, y_i \leqslant c_v & \forall v \in V(G) \\ & & y_i \geqslant 0 & \forall i \in \{1, \ldots, k\}. \end{array}$$

The algorithm is based on the primal-dual method. It maintains a boolean primal solution x and a feasible dual solution y. Initially, all variables are set to 0. Then the algorithm enters its main loop, that ends when x satisfies all diamond inequalities. At the ith iteration, a violated inequality $\sum_{v \in V} a_{i,v}\, x_v \geqslant \beta_i$ is added to the working LP and the corresponding dual variable y_i is increased. In order to preserve the feasibility of the dual solution, we stop increasing y_i whenever some dual inequality becomes tight. That is, we stop increasing when $\sum_{j=1}^{i} a_{j,v}\, y_j = c_v$ for some vertex v, that is said to be *tight*. (Actually, we should also stop increasing y_i in case a 'collision' occurs [6, Section 5.2.4].) All tight vertices v (if any) are then added to the primal solution. That is, the corresponding variables x_v are increased from 0 to 1. The current iteration then ends and we check whether x satisfies all diamond inequalities. If so, then we exit the loop, perform a reverse delete step, and output the current primal solution.

The precise way the violated blended diamond inequality is chosen depends among other things on the *residual cost* (or slack) of the vertices. The residual cost of vertex v at the ith iteration is the number $c_v - \sum_{j=1}^{i-1} a_{j,v}\, y_j$. Note that the residual cost of a vertex is always nonnegative, and zero if and only if the vertex is tight.

5.2 The Algorithm

The algorithm (rather, a simplified version of the algorithm) is described in Algorithm 2.

Algorithm 2. A $O(\log n)$-approximation algorithm for weighted graphs.

```
– $X \leftarrow \varnothing$;   $y \leftarrow 0$;   $i \leftarrow 0$
– While $X$ is not a hitting set of $G = (V, E)$, repeat the following steps:
    • $i \leftarrow i + 1$
    • Let $H$ be the graph obtained by shaving $G - X$
    • Find a reduced graph $\widetilde{H}$ of $H$
    • Find a diamond $\widetilde{D}$ in $\widetilde{H}$ of size at most $6 \log_{3/2} |\widetilde{H}| + 8$      (using Lemma 1)
    • Compute a violated blended diamond inequality $\sum_{v \in V} a_{i,v}\, x_v \geqslant \beta_i$
      based on $\widetilde{D}$, and the residual costs; add it to (LP)
    • Increase $y_i$ until some vertex becomes tight
    • Add all tight vertices $v$ to $X$, in a carefully chosen order [6, Section 5.2.6]
– $k \leftarrow i$
– Perform a reverse delete step on $X$
```

We remark that the set X naturally corresponds to a primal solution x, obtained by setting x_v to 1 if $v \in X$, to 0 otherwise, for every $v \in V(G)$. This vector x satisfies the diamond inequalities (3) exactly when we exit the *while* loop of the algorithm, that is, when X becomes a hitting set.

The reverse delete step consists in considering the vertices of X in the reverse order in which they were added to X and deleting those vertices v such that $X - \{v\}$ is still a hitting set. Observe that, because of this step, the hitting set X output by the algorithm is inclusion-wise minimal.

Theorem 1. *Algorithm 2 yields a $O(\log n)$-approximation for the diamond hitting set problem.*

We can prove that the integrality gap of the natural LP relaxation for the problem (see page 194) is $\Theta(\log n)$. This result is obtained using expander graphs with large girth [6, Section 5.4].

6 A 9-Approximation Algorithm

In this section we give a primal-dual 9-approximation algorithm for the diamond hitting set problem. We start with a sketch of the algorithm in Section 6.1. The algorithm makes use of the sparsity inequalities. In order to describe them, we first bound the number of edges in a forest of cacti in Section 6.2; using this bound, in Sections 6.3 and 6.4 we introduce the sparsity inequalities and prove their validity. Once again, missing proofs and details can be found in the full version of this paper [6, Section 6].

In the whole section, q is a global parameter of our algorithm which is set to 5. (We remark that the analysis below could be adapted to other values of q, but this would not give an approximation factor better than 9.)

6.1 The Algorithm

Our 9-approximation algorithm for the diamond hitting set problem is very similar to the $O(\log n)$-approximation algorithm. The main difference is that we use a larger set of inequalities to build the working LP relaxation. See Algorithm 3 for a description of (a simplified version of) the algorithm.

Algorithm 3. A 9-approximation algorithm.

```
– $X \leftarrow \varnothing$;   $y \leftarrow 0$;   $i \leftarrow 0$
– While $X$ is not a hitting set of $G = (V, E)$, repeat the following steps:
    • $i \leftarrow i + 1$
    • Let $H$ be the graph obtained by shaving $G - X$
    • Find a reduced graph $\widetilde{H}$ of $H$
    • If, in $\widetilde{H}$, no two cycles of size at most $q$ share an edge then let
      $\sum_{v \in V} a_{i,v}\, x_v \geqslant \beta_i$ be the extended sparsity inequality with support $V(H)$
    • Otherwise, $\widetilde{H}$ contains a diamond $\widetilde{D}$ with at most $2q - 1$ edges and let
      $\sum_{v \in V} a_{i,v}\, x_v \geqslant \beta_i$ be a blended diamond inequality deduced from $\widetilde{D}$ and
      the residual costs
    • Add the inequality $\sum_{v \in V} a_{i,v}\, x_v \geqslant \beta_i$ to the working LP
    • Increase $y_i$ until some vertex becomes tight
    • Add all tight vertices to $X$, in a carefully chosen order
– $k \leftarrow i$
– Perform a reverse delete step on $X$
```

Our main result is as follows.

Theorem 2. *Algorithm 3 yields a 9-approximation for the diamond hitting set problem.*

6.2 Bounding the Number of Edges in a Forest of Cacti

The following lemma provides a bound on the number of edges in a forest of cacti. For $i \in \{2, \ldots, q\}$, we denote by $\gamma_i(G)$ the number of cycles of length i of a graph G.

Lemma 2. *Let F be a forest of cacti with k components and let $q \geqslant 2$. Then*

$$||F|| \leqslant \frac{q+1}{q}(|F|-k) + \sum_{i=2}^{q} \frac{q-i+1}{q} \gamma_i(F).$$

Proof. Denote by $\gamma_{>q}(F)$ the number of cycles of F whose length exceeds q. We have

$$||F|| = |F| - k + \sum_{i=2}^{q} \gamma_i(F) + \gamma_{>q}(F). \tag{4}$$

In the right hand side, the first two terms represent the number of edges in a spanning forest of F, while the last terms give the number of edges that should be added to obtain the forest of cacti F.

Because every two cycles in F are edge disjoint, we have

$$||F|| \geqslant \sum_{i=2}^{q} i\, \gamma_i(F) + (q+1)\, \gamma_{>q}(F).$$

Combining this with (4), we get

$$\gamma_{>q}(F) \leqslant \frac{1}{q}\Big(|F| - k - \sum_{i=2}^{q} (i-1)\gamma_i(F)\Big). \tag{5}$$

From (4) and (5), we finally infer

$$\begin{aligned} ||F|| &\leqslant |F| - k + \sum_{i=2}^{q} \gamma_i(F) + \frac{1}{q}\Big(|F| - k - \sum_{i=2}^{q} (i-1)\, \gamma_i(F)\Big) \\ &\leqslant \frac{q+1}{q}(|F|-k) + \sum_{i=2}^{q} \frac{q-i+1}{q}\, \gamma_i(F). \end{aligned}$$

□

6.3 The Sparsity Inequalities

We define the *load* of a vertex v in a graph G as

$$\ell_G(v) := d_G(v) - \sum_{i=2}^{q} \lambda_i\, \gamma_i(G, v),$$

where, for $i \in \{2, \dots, q\}$, $\gamma_i(G, v)$ denotes the number of cycles of length i incident to v in G and

$$\lambda_i := \frac{q-i+1}{\lfloor i/2 \rfloor\, q}.$$

Lemma 3. *Let X be a hitting set of a graph G where no two cycles of length at most q share an edge. Then,*

$$\sum_{v \in X} \Big(\ell_G(v) - \frac{q+1}{q}\Big) \geqslant ||G|| - \frac{q+1}{q}|G| - \sum_{i=2}^{q} \frac{q-i+1}{q}\, \gamma_i(G) + \frac{q+1}{q}. \tag{6}$$

We call Inequality (6) a *sparsity inequality.*

Proof. For $i \in \{2, \ldots, q\}$ and $j \in \{0, \ldots, i\}$, we denote by ξ_i^j the number of cycles of G that have length i and exactly j vertices in X.

Letting $||X||$ and $|\delta(X)|$ respectively denote the number of edges of G with both ends in X and the number of edges of G having an end in X and the other in $V(G) - X$, we have

$$\begin{aligned}\sum_{v \in X} \ell_G(v) &= 2||X|| + |\delta(X)| - \sum_{i=2}^{q}\sum_{j=1}^{i} j\,\lambda_i\,\xi_i^j \\ &= ||X|| + ||G|| - ||G - X|| - \sum_{i=2}^{q}\sum_{j=1}^{i} j\,\lambda_i\,\xi_i^j \\ &\geqslant ||X|| + ||G|| - \frac{q+1}{q}(|G-X|-1) - \sum_{i=2}^{q}\frac{q-i+1}{q}\,\xi_i^0 - \sum_{i=2}^{q}\sum_{j=1}^{i} j\,\lambda_i\,\xi_i^j,\end{aligned}$$

where the last inequality follows from Lemma 2 applied to the forest of cacti $G - X$ (notice that $\gamma_i(G - X) = \xi_i^0$).

Because no two cycles of length at most q share an edge and, in a cycle of length i, each subset of size j induces a subgraph that contains at least $2j - i$ edges, we have

$$||X|| \geqslant \sum_{i=2}^{q}\sum_{j=1+\lfloor \frac{i}{2} \rfloor}^{i} (2j-i)\,\xi_i^j.$$

Thus, we obtain

$$\begin{aligned}\sum_{v \in X} \ell_G(v) \geqslant \sum_{i=2}^{q}\sum_{j=1+\lfloor \frac{i}{2} \rfloor}^{i} (2j-i)\,\xi_i^j + ||G|| - \frac{q+1}{q}(|G-X|-1) \\ - \sum_{i=2}^{q}\frac{q-i+1}{q}\,\xi_i^0 - \sum_{i=2}^{q}\sum_{j=1}^{i} j\,\lambda_i\,\xi_i^j.\end{aligned}$$

We leave it to the reader to check that, in the right hand side of the last inequality, the total coefficient of ξ_i^j is at least $-\frac{q-i+1}{q}$, for all $i \in \{2, \ldots, q\}$ and $j \in \{0, \ldots, i\}$. Hence,

$$\sum_{v \in X} \ell_G(v) \geqslant ||G|| - \frac{q+1}{q}|G| + \frac{q+1}{q}|X| + \frac{q+1}{q} - \sum_{i=2}^{q}\frac{q-i+1}{q}\,\gamma_i(G).$$

Inequality (6) follows. □

6.4 The Extended Sparsity Inequalities

Consider a shaved graph H and denote by $\widetilde{H}$ a reduced graph of H. Suppose that, in $\widetilde{H}$, no two cycles of length at most q share an edge. The sparsity inequality (6) for $\widetilde{H}$ yields a valid inequality also for H, where the coefficient of each variable corresponding to a vertex that was removed when H is zero. However, as it is, the inequality is useless. We have to raise the coefficients to those variables in such a way that the resulting inequality remains valid.

First, we associate to each edge of $\widetilde{H}$ a corresponding *primitive subgraph* in H, defined as follows. Consider an edge $e \in E(\widetilde{H})$. If e was already present in H, then its primitive subgraph is the edge itself and its two ends. Otherwise, the primitive subgraph of e is the bond whose reduction produced e. In particular, if e has a twin edge e', then the primitive subgraphs of e and e' coincide. The primitive subgraph J of a subgraph $\widetilde{J} \subseteq \widetilde{H}$ is defined simply as the union of the primitive subgraphs of every edge in $E(\widetilde{J})$.

Thus, the graph H can be uniquely decomposed into simple or double pieces corresponding respectively to edges or pairs of parallel edges in $\widetilde{H}$. Here, the pieces of H are defined as follows: let v and w be two adjacent vertices of $\widetilde{H}$, and let $\widetilde{J}$ denote the subgraph of $\widetilde{H}$ induced by $\{v, w\}$ (thus $\widetilde{J}$ is either an edge or a pair of parallel edges, together with the endpoints v and w). The primitive subgraph of $\widetilde{J}$ in H, say J, is a *piece* of H with ends v and w. Such a piece is *simple* if $\widetilde{J}$ has exactly one edge and *double* otherwise (see Fig. 3), that is, if $\widetilde{J}$ has two parallel edges. The vertices of H are of two types: the branch vertices are those that survive in $\widetilde{H}$, and the other vertices are internal to some piece of H.

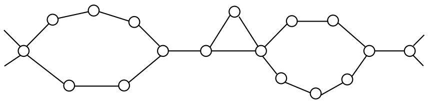


Fig. 3. A double piece

Consider a double piece Q of H (if any) and a cycle C contained in Q. Then C is a block of Q. A vertex v of C is said to be an *end* of the cycle C if v is an end of the piece Q or v belongs to a block of Q distinct from C. Observe that C has always two distinct ends. The cycle C has also two *handles*, defined as the two v–w paths in C, where v and w are the two ends of C.

The two handles of C are labelled *top* and *bottom* as described in [6, Section 6]. Denote by $t(C)$ (resp. $b(C)$) the minimum residual cost of a vertex in the top handle (resp. bottom handle) of C. Thus, $t(C) \leqslant b(C)$. Also, we choose a cycle of Q (if any) and declare it to be *special*. If possible, the special cycle is chosen among the cycles C contained in Q with $t(C) = b(C)$. (So *every* double piece of H has a special cycle.)

The *extended sparsity inequality* for H reads

$$\sum_{v \in V(H)} a_v x_v \geqslant \beta, \tag{7}$$

where

$$a_v := \begin{cases} \ell_{\widetilde{H}}(v) - \dfrac{q+1}{q} & \text{if } v \text{ is a branch vertex,} \\ 1 & \text{if } v \text{ is an internal vertex of a simple piece,} \\ 2 & \text{if } v \text{ is an internal vertex of a double piece} \\ & \text{and does not belong to any handle,} \\ 0 & \text{if } v \text{ belongs to the top handle} \\ & \text{of a cycle } C \text{ with } t(C) < b(C), \\ 2 & \text{if } v \text{ belongs to the bottom handle} \\ & \text{of a cycle } C \text{ with } t(C) < b(C), \\ 1 & \text{if } v \text{ belongs to a handle of a cycle } C \text{ with } t(C) = b(C) \\ & \text{or } C \text{ is special,} \end{cases}$$

and

$$\beta := ||\widetilde{H}|| - \frac{q+1}{q}|\widetilde{H}| - \sum_{i=2}^{q} \frac{q-i+1}{q}\gamma_i(\widetilde{H}) + \frac{q+1}{q}.$$

In the definition of a_v above, we always assume that the cycle C is contained in a double piece of H. The next lemma is proved in [6, Lemma 6.3].

Lemma 4. *Let H be a graph and let $\widetilde{H}$ be a reduced graph of H such that no two cycles of length at most q share an edge. Then Inequality (7) is valid, that is,*

$$\sum_{v \in X} a_v \geqslant \beta$$

whenever X is a hitting set of H.

The following lemma is the key of our analysis of Algorithm 3. Notice that the constant involved is smaller than the approximation guarantee of our algorithm. The number 9 comes from the fact that we use blended diamond inequalities for diamonds with up to 9 pieces.

Lemma 5. *Let H be a shaved graph and let $\widetilde{H}$ be a reduced graph of H. Suppose that, in $\widetilde{H}$, no two cycles of length at most q share an edge. Then,*

$$\sum_{v \in X} a_v \leqslant 8\,\beta$$

for every minimal hitting set X of H.

Acknowledgements

We thank Dirk Oliver Theis for his valuable input in the early stage of this research. We also thank Jean Cardinal and Marcin Kamiński for stimulating discussions.

References

1. Bafna, V., Berman, P., Fujito, T.: A 2-approximation algorithm for the undirected feedback vertex set problem. SIAM Journal on Discrete Mathematics 12(3), 289–297 (1999)
2. Bar-Yehuda, R., Geiger, D., Naor, J., Roth, R.M.: Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and Bayesian inference. SIAM Journal on Computing 27(4), 942–959 (1998)
3. Becker, A., Geiger, D.: Optimization of Pearl's method of conditioning and greedy-like approximation algorithms for the vertex feedback set problem. Artificial Intelligence 83, 167–188 (1996)
4. Chudak, F.A., Goemans, M.X., Hochbaum, D.S., Williamson, D.P.: A primaldual interpretation of two 2-approximation algorithms for the feedback vertex set problem in undirected graphs. Operations Research Letters 22, 111–118 (1998)
5. Even, G., Naor, J., Schieber, B., Zosin, L.: Approximating minimum subset feedback sets in undirected graphs with applications. SIAM Journal on Discrete Mathematics 13(2), 255–267 (2000)
6. Fiorini, S., Joret, G., Pietropaoli, U.: Hitting diamonds and growing cacti (2010), http://arxiv.org/abs/0911.4366v2
7. Goemans, M.X., Williamson, D.P.: The primal-dual method for approximation algorithms and its application to network design problems. In: Approximation Algorithms for NP-Hard Problems, ch. 4, pp. 144–191. PWS Publishing Company (1997)
8. Khot, S., Regev, O.: Vertex cover might be hard to approximate to within $2 - \varepsilon$. Journal of Computer and System Sciences 74(3), 334–349 (2008)
9. Papadimitriou, C.H., Steiglitz, K.: Combinatorial Optimization: Algorithms and Complexity. Prentice-Hall, Englewood Cliffs (1982)

Approximability of 3- and 4-Hop Bounded Disjoint Paths Problems

Andreas Bley[1,⋆] and Jose Neto[2]

[1] TU Berlin, Institute of Mathematics
Straße des 17. Juni 136, D-10623 Berlin, Germany
bley@math.tu-berlin.de

[2] Institut Telecom, Telecom & Management SudParis,
CNRS UMR 5157, 9 rue Charles Fourier, F-91011 Evry, France
Jose.Neto@it-sudparis.eu

Abstract. A path is said to be ℓ-bounded if it contains at most ℓ edges. We consider two types of ℓ-bounded disjoint paths problems. In the maximum edge- or node-disjoint path problems MEDP(ℓ) and MNDP(ℓ), the task is to find the maximum number of edge- or node-disjoint ℓ-bounded (s,t)-paths in a given graph G with source s and sink t, respectively. In the weighted edge- or node-disjoint path problems WEDP(ℓ) and WNDP(ℓ), we are also given an integer $k \in \mathbb{N}$ and non-negative edge weights $c_e \in \mathbb{N}$, $e \in E$, and seek for a minimum weight subgraph of G that contains k edge- or node-disjoint ℓ-bounded (s,t)-paths. Both problems are of great practical relevance in the planning of fault-tolerant communication networks, for example.

Even though length-bounded cut and flow problems have been studied intensively in the last decades, the $\mathcal{NP}$-hardness of some 3- and 4-bounded disjoint paths problems was still open. In this paper, we settle the complexity status of all open cases showing that WNDP(3) can be solved in polynomial time, that MEDP(4) is $\mathcal{APX}$-complete and approximable within a factor of 2, and that WNDP(4) and WEDP(4) are $\mathcal{APX}$-hard and $\mathcal{NPO}$-complete, respectively.

Keywords: Graph algorithms; length-bounded paths; complexity; approximation algorithms.

1 Introduction

Two major concerns in the design of modern communication networks are the protection against potential failures and the permanent provision of a guaranteed minimum level of service quality. A wide variety of models expressing such requirements may be found in the literature, e.g. [1,2,3,4]. Coping simultaneously with both requirements naturally leads to length-restricted disjoint paths problems: In order to ensure that a pair of nodes remains connected also after some nodes or edges of the network fail, one typically demands the existence

⋆ Supported by the DFG Research Center Matheon – Mathematics for key technologies.

of several node- or edge-disjoint transmission paths between them. Each node on a transmission path, however, may lead to additional packet delay, jitter, and potential transmission errors for the corresponding data stream. To provide a guaranteed level of transmission service quality, these paths thus must not contain more than a certain number of intermediate nodes or, equivalently, of edges.

Mathematically, the task of verifying if a given network satisfies the robustness and quality requirements of a given node pair can be formulated as an edge- or node-disjoint paths problem. Let $G = (V, E)$ be a simple graph with source $s \in V$ and sink $t \in V$ and let $k \in \mathbb{N}$. A path in G is said to be ℓ-bounded for a given number $\ell \in \mathbb{N}$ if it contains at most ℓ edges. In the edge-disjoint paths problem $EDP(\ell)$, the task is to decide if there are k edge-disjoint ℓ-bounded (s,t)-paths in G or not. In the corresponding maximum edge-disjoint paths problem MEDP(ℓ), we wish to find the maximum number of edge-disjoint ℓ-bounded (s,t)-paths. The analogous node-disjoint path problems are denoted as $NDP(\ell)$ and $MNDP(\ell)$. The task of designing a network that satisfies the requirements of a single node pair can be modeled as a weighted edge- or node-disjoint path problems WEDP(ℓ) and WNDP(ℓ). In these problems, we are given the graph G, source s and sink t, the number of paths k, and non-negative edge weights $c_e \in \mathbb{N}$, $e \in E$. The task is to find a minimum cost subset $E' \subseteq E$ such that the subgraph (V, E') contains at least k edge- or node-disjoint ℓ-bounded (s,t)-paths, respectively.

Due to their great practical relevance, problems asking for disjoint paths or unsplittable flows between some node pairs have received considerable attention in the literature. Structural results, complexity issues, and approximation algorithms for disjoint paths problems without length restrictions are discussed in [5,6,7], for example.

In a seminal article Menger [8] shows that the maximum number of edge- or node-disjoint (s,t)-paths in a graph is equal to the minimum size of an (s,t)-edge- or (s,t)-node-cut, respectively. Lovász *et al.* [9], Exoo [10], and Niepel *et al.* [11] showed that this strong duality between disjoint paths and (suitably defined) cuts still holds for 4-bounded node-disjoint paths and node-cuts and for 3-bounded edge-disjoint paths and edge-cuts, but that Menger's theorem does not hold for length bounds $\ell \geq 5$. The ratio between the number of paths and the cut size is studied in [12,13]. Generalizations of Menger's theorem and of Ford and Fulkerson's max flow min cut theorem to length-bounded flows are an area of active research [14,15].

Polynomial time algorithms for the minimum ℓ-bounded edge-cut problem with $\ell \leq 3$ have been presented by Mahjoub and McCormick [16]. Baier *et al.* [17] proved that the minimum ℓ-bounded edge-cut problem is $\mathcal{APX}$-hard for $\ell \geq 4$ and that the corresponding node-cut problem is $\mathcal{APX}$-hard for $\ell \geq 5$.

Itai *et al.* [18] and Bley [19] showed that the problems MEDP(ℓ) and MNDP(ℓ) of finding the maximum number of edge- and node-disjoint ℓ-bounded paths are polynomially solvable for $\ell \leq 3$ and $\ell \leq 4$, respectively, and that both problems are $\mathcal{APX}$-complete for $\ell \geq 5$. Heuristics to find large sets of disjoint length bounded

Table 1. Known and new (bold) complexity results for node- and edge-disjoint ℓ-bounded paths problems

ℓ	MNDP(ℓ)	WNDP(ℓ)	MEDP(ℓ)	WEDP(ℓ)
≤ 2	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{P}$
$= 3$	$\mathcal{P}$	**P**	$\mathcal{P}$	$\mathcal{P}$
$= 4$	$\mathcal{P}$	**APX-hard** (at least)	**APX-complete**	**NPO-complete**
≥ 5	$\mathcal{APX}$-complete	$\mathcal{NPO}$-complete	$\mathcal{APX}$-complete	$\mathcal{NPO}$-complete

paths can be found, e.g., in [18,20,21]. Polyhedral approaches to these problems are investigated in [22,23,24]. The weighted disjoint paths problems WEDP(ℓ) and WNDP(ℓ) are known to be $\mathcal{NPO}$-complete for $\ell \geq 5$ and to be polynomially solvable for $\ell \leq 2$ in the node-disjoint case and for $\ell \leq 3$ in the edge-disjoint case [19]. Further results and a finer analysis of the complexity of disjoint paths problems by means of different parameterizations (namely w.r.t. the number of paths, their length, or the graph treewidth) are presented in [25,26]. The complexity of MEDP(4), WEDP(4), WNDP(3), and WNDP(4), however, has been left open until now.

The contribution of this paper is to close all these open cases. In Section 2, we prove that the maximum edge-disjoint 4-bounded paths problem MEDP(4) is $\mathcal{APX}$-complete, presenting a 2-approximation algorithm and an approximation preserving reduction from Max-k-Sat(3) to MEDP(4). This implies that the corresponding weighted edge-disjoint paths problem WEDP(4) is $\mathcal{NPO}$-complete. In Section 3, we then show how to solve the weighted node-disjoint 3-bounded paths problem WNDP(3) via matching techniques in polynomial time and prove that the 4-bounded version of this problem is at least $\mathcal{APX}$-hard. Table 1 summarizes the known and new complexity results regarding these problems. All hardness results and algorithms presented in this paper generalize in a straightforward way to directed graphs and to non-simple graphs containing parallel edges.

2 Edge-Disjoint 4-Bounded Paths

In this section, we study the approximability of the two edge-disjoint 4-bounded problems. First, we consider the problem of maximizing the number of edge-disjoint paths. One easily observes that any inclusion-wise maximal set of edge-disjoint 4-bounded (s,t)-paths, which can be computed in polynomial time by greedily adding disjoint paths to the solution, is a 4-approximate solution for MEDP(4) [19].

A 2-approximation algorithm is obtained as shown in algorithm ExFlow on Page 208. In the first step of algorithm ExFlow, we construct the directed graph $G' = (V', E')$ with $V' = \bigcup_{i=0}^{4} V_i$ for $V_0 := \{s_0\}$, $V_4 := \{t_4\}$, and $V_i := \{v_i \mid v \in V \setminus \{s,t\}$ with $dist_G(v,s) \leq i$ and $dist_G(v,t) \leq 4-i\}$ for all $i \in \{1,2,3\}$, and $E' := \bigcup_{i=0}^{4} E_i$ with $E_0 := \{(s_0, t_4)\}$ if $st \in E$, $E_0 := \emptyset$ if $st \notin E$, and

$E_i := \{(v_{i-1}, w_i) \in V_{i-1} \times V_i \mid vw \in E \text{ or } v = w\}$ for $i \in \{1, \ldots, 4\}$, where $dist_G(u, v)$ denotes the distance from node u to node v in G. We assign cost 0 and capacity 1 to all edges $u_i u_{i+1} \in E'$ and capacity 1 and cost 1 to all other edges in E'. Figure 1 illustrates this construction.

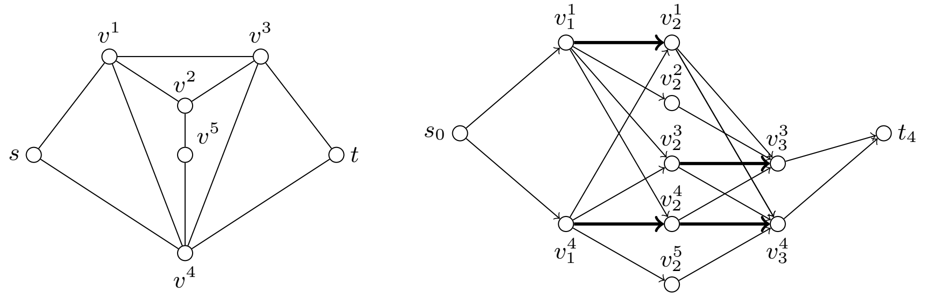


Fig. 1. Construction of the hop-extended digraph G' (right) from the given graph G (left) in Step 1 of algorithm ExFlow. Arcs with cost 0 in G' are thick.

In this layered digraph, we compute an (integer) minimum cost maximum (s_0, t_4)-flow and its decomposition into paths $P'_1, \ldots, P'_k$. Each such path $P'_i = (s_0, u_1, v_2, w_3, t_4)$ defines a 4-bounded walk (s, u, v, w, t) in G, which can be shortened to a simple 4-bounded path P_i. Let $\mathcal{F} = \{P_1, \ldots, P_k\}$ be the set of these paths. Note that these paths are 4-bounded, but not necessarily edge-disjoint.

In the second step, we create the associated "conflict graph" $H = (\mathcal{F}, \{P_i P_j \mid P_i \cap P_j \neq \emptyset\})$. By Lemma 1, H consists only of disjoint paths and isolated nodes. Choosing all isolated nodes and a maximum independent set in each of these paths, we thus can compute an independent set $\mathcal{S} \subseteq \mathcal{F}$ of size $|\mathcal{S}| \geq |\mathcal{F}|/2$ in H. This is done in the third step of our algorithm.

Finally, we return the path set corresponding to this independent set.

Steps 1, 2, and 4 of this algorithm clearly can be done in polynomial time. The possibility to perform also Step 3 in polynomial time follows from the following lemma.

Lemma 1. *The conflict graph $H = (\mathcal{F}, \{P_i P_j \mid P_i \cap P_j \neq \emptyset\})$ created in Step 3 of algorithm* ExFlow *consists of isolated nodes and disjoint paths only.*

```
ExFlow
1. Compute a min cost max $(s_0, t_4)$-flow $f$ in the hop-extended digraph $G'$.
   Let $\mathcal{F} := \{P_1, \ldots, P_k\}$ be the corresponding 4-bounded simple paths in $G$.
2. Create the conflict graph $H := (\mathcal{F}, \{P_i P_j \mid P_i \cap P_j \neq \emptyset\})$.
3. Compute an independent set $\mathcal{S} \subseteq \mathcal{F}$ in $H$ with $|\mathcal{S}| \geq \frac{1}{2}|\mathcal{F}|$.
4. Return $\mathcal{S}$.
```

Proof. Let f be the minimum cost maximum (s_0, t_4)-flow in G' computed in Step 1 of ExFlow and let $P'_1, \ldots, P'_k$ be its path decomposition. Note that the paths P'_i are edge-disjoint in G'.

By construction of G', each edge $e \in \delta(s) \cup \delta(t)$ corresponds to at most one arc (s_0, v_1), (v_3, t_4), or (s_0, t_4) in G'. Thus, any such edge is contained in at most one path in $\mathcal{F}$. Furthermore, for each edge $e = uv \in E \setminus \delta(s) \setminus \delta(t)$, the paths $P'_1, \ldots, P'_k$ in G' contain at most one of the arcs (u_1, v_2) and (v_1, u_2) and at most one of the arcs (u_2, v_3) and (v_2, u_3). Otherwise, these paths do not correspond to a *minimum cost* maximum flow: If there were two paths $P'_1 = (s_0, u_1, v_2, w_3, t_4)$ and $P'_2 = (s_0, v_1, u_2, q_3, t_4)$, for example, then replacing these paths by the paths $P''_1 = (s_0, u_1, u_2, q_3, t_4)$ and $P''_2 = (s_0, v_1, v_2, w_3, t_4)$ would reduce the cost of the corresponding flow. Consequently, any edge $e \in E \setminus \delta(s) \setminus \delta(t)$ can be contained in at most two of the paths in $\mathcal{F}$ and, further on, a path in $\mathcal{F}$ can intersect with at most two other paths in $\mathcal{F}$. This implies that the conflict graph H constructed in Step 2 of ExFlow consists only of isolated nodes and disjoint paths and cycles.

To see that H cannot contain cycles, let $C = \{P_1, \ldots, P_n\}$ be the shortest cycle in H. Then each edge in $M := \bigcup_{i \in C} P_i \setminus \delta(s) \setminus \delta(t)$ must appear in exactly two paths in C, once as the second and once as the third edge. If there were two paths P_1 and P_2 in C that traverse one of the edges $e = uv \in M$ in opposite directions, then the corresponding paths in G' would be of the form $P'_1 = (s_0, u_1, v_2, w_3, t_4)$ and $P'_2 = (s_0, q_1, v_2, u_3, t_4)$. In this case, replacing P'_1 and P'_2 by $P''_1 = (s_0, u_1, u_2, u_3, t_4)$ and $P''_2 = (s_0, q_1, v_2, w_3, t_4)$ would reduce the cost of the corresponding flow in G' (and the size of the remaining cycle in C), which is a contradiction to our assumption that the paths P'_i correspond to a *minimum cost* maximum flow in G'.

So, we may assume that the paths in C traverse each edge $e \in M$ in the same direction. Then, for each $e = uv \in M$, there is exactly one path of the form (s, u, v, w, t) and exactly one path of the form (s, q, u, v, t) in C. In this case, however, we can replace each path $P'_i = (s_0, u_1, v_2, w_3, t_4)$ that corresponds to a path in C by the less costly path $P''_i = (s_0, u_1, v_2, v_3, t_4)$ without exceeding the edge capacities in G'. This is again a contradiction to our assumption that the paths P'_i define a minimum cost maximal flow in G'. Consequently, there are no cycles in H. □

Theorem 2. ExFlow *is a 2-approximation algorithm for MEDP(4).*

Proof. By Lemma 1, all steps of the algorithm can be executed in polynomial time. The paths in $\mathcal{S}$ are derived from the 4-bounded (s_0, t_4)-flow paths in G', so they are clearly 4-bounded. As $\mathcal{S}$ is an independent set in the conflict graph H, the paths in $\mathcal{S}$ are also edge-disjoint.

Furthermore, any set of edge-disjoint 4-bounded (s, t)-paths in G defines a feasible (s_0, t_4)-flow in G'. Hence, $k = |\mathcal{F}|$ is an upper bound on the maximum number k^* of edge-disjoint 4-bounded (s, t)-paths in G, which immediately implies $|\mathcal{S}| \geq \frac{1}{2} k^*$. □

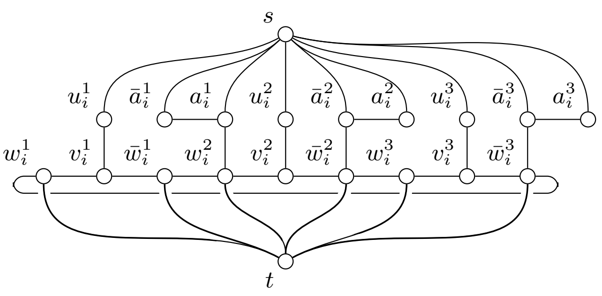


Fig. 2. Graph G_i for variable x_i occurring as literals x_i^1, $\bar{x}_i^2$, $\bar{x}_i^3$

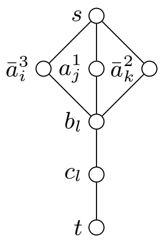


Fig. 3. Graph H_l for clause $C_l = (\bar{x}_i^3 \vee x_j^1 \vee \bar{x}_k^2)$

In order to show that MEDP(4) is $\mathcal{APX}$-hard, i.e., that there is some $c > 1$ such that approximating MEDP(4) within a factor less than c is $\mathcal{NP}$-hard, we construct an approximation preserving reduction from the MAX-k-SAT(3) problem to MEDP(4). Given a set X of boolean variables and a collection C of disjunctive clauses such that each clause contains at most k literals and each variable occurs at most 3 times as a literal, the MAX-k-SAT(3) problem is to find a truth assignment to the variables that maximizes the number of satisfied clauses. MAX-k-SAT(3) is known to be $\mathcal{APX}$-complete [27].

Theorem 3. *MEDP(4) is $\mathcal{APX}$-hard.*

Proof. We construct an approximation preserving reduction from MAX-k-SAT(3) to MEDP(4). Let x_i, $i \in I$, be the boolean variables and C_l, $l \in L$ be the clauses of the given MAX-k-SAT(3) instance. Without loss of generality we may assume that each variable x_i occurs exactly 3 times as a literal and denote these occurrences by x_i^j, $j \in J := \{1, \ldots, 3\}$.

We construct an undirected graph $G = (V, E)$ that consists of $|I| + |L|$ subgraphs, one for each variable and one for each clause, as follows. For each $i \in I$, we construct a variable graph $G_i = (V_i, E_i)$ as shown in Figure 2. G_i contains the nodes and edges

$$
\begin{aligned}
V_i :=& \{s,\, t\} \cup \{u_i^j,\, v_i^j,\, w_i^j,\, \bar{w}_i^j,\, a_i^j,\, \bar{a}_i^j \mid j \in J\} \quad \text{and} \\
E_i :=& \{su_i^j,\, u_i^j v_i^j,\, v_i^j w_i^j,\, v_i^j \bar{w}_i^j,\, w_i^j t,\, \bar{w}_i^j t,\, sa_i^j,\, s\bar{a}_i^j,\, \bar{w}_i^j w_i^{j+1} \mid j \in J\} \\
& \cup \{a_i^j w_i^{j+1} \mid j \in J : x_i^j \text{ occurs as unnegated literal } x_i^j\} \\
& \cup \{\bar{a}_i^j \bar{w}_i^j \mid j \in J : x_i^j \text{ occurs as negated literal } \bar{x}_i^j\} \ ,
\end{aligned}
$$

where $w_i^4 = w_i^1$ for notational simplicity. The nodes s and t are contained in all subgraphs and serve as source and destination for all paths. For each $l \in L$, we construct a clause graph $H_l = (W_l, F_l)$ as shown in Figure 3. In addition to the nodes and edges it shares with the variable graphs, H_l contains 2 nodes and

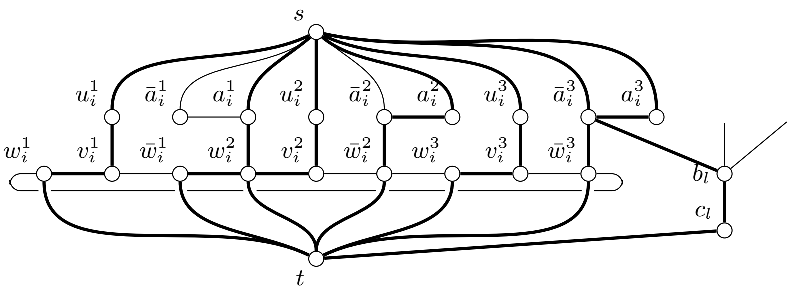


Fig. 4. Union of G_i and H_l for variable x_i and clause $C_l = (\bar{x}_i^3 \vee \ldots)$. Thick lines are paths in $\bar{\mathcal{P}}_i$ and path Q_{li}.

$k' + 2$ edges, where k' is the number of literals in clause C_l. Formally, W_l and F_l are defined as

$$\begin{aligned} W_l :=& \{s,\, t,\, b_l,\, c_l\} \cup \{\bar{a}_i^j \mid i \in I,\, j \in J : \text{negated literal } \bar{x}_i^j \text{ occurs in } C_l\} \\ &\cup \{a_i^j \mid i \in I,\, j \in J : \text{unnegated literal } x_i^j \text{ occurs in } C_l\} \quad \text{and} \\ F_l :=& \{b_l c_l, c_l t\} \cup \{s\bar{a}_i^j,\, \bar{a}_i^j b_l \mid i \in I,\, j \in J : \text{negated literal } \bar{x}_i^j \text{ occurs in } C_l\} \\ &\cup \{s a_i^j,\, a_i^j b_l \mid i \in I,\, j \in J : \text{unnegated literal } x_i^j \text{ occurs in } C_l\}\ . \end{aligned}$$

The goal in the constructed MEDP(4) instance is to find the maximum number of edge-disjoint 4-bounded (s,t)-paths in the simple undirected graph G obtained as the union of all variable and clause (sub)-graphs. It is clear that the constructions can be performed in polynomial time.

For notational convenience, we denote for each $i \in I$ and $j \in J$ the paths

$$P_{ij} = (s, u_i^j, v_i^j, \bar{w}_i^j, t)\ , \qquad P'_{ij} = \begin{cases} (s, \bar{a}_i^j, \bar{w}_i^j, w_i^{j+1}, t) & \text{if } \bar{x}_i^j \text{ occurs} \\ (s, \bar{a}_i^j, a_i^j, w_i^{j+1}, t) & \text{if } x_i^j \text{ occurs} \end{cases},$$

$$\bar{P}_{ij} = (s, u_i^j, v_i^j, w_i^j, t)\ , \qquad \bar{P}'_{ij} = \begin{cases} (s, a_i^j, \bar{a}_i^j, \bar{w}_i^j, t) & \text{if } \bar{x}_i^j \text{ occurs} \\ (s, a_i^j, w_i^{j+1}, \bar{w}_i^j, t) & \text{if } x_i^j \text{ occurs} \end{cases}.$$

For each $i \in I$ and $l \in L$ such that variable x_i occurs in clause C_l, we denote

$$Q_{li} = \begin{cases} (s, a_i^j, b_l, c_l, t) & \text{if literal } x_i^j \text{ occurs in } C_l \\ (s, \bar{a}_i^j, b_l, c_l, t) & \text{if literal } \bar{x}_i^j \text{ occurs in } C_l. \end{cases}$$

Furthermore, we define $\mathcal{P}_i := \{P_{ij},\, P'_{ij} \mid j \in J\}$ and $\bar{\mathcal{P}}_i := \{\bar{P}_{ij},\, \bar{P}'_{ij} \mid j \in J\}$ for all $i \in I$, and $\mathcal{Q}_l := \{Q_{li} \mid i \in I : x_i \text{ occurs in } C_l\}$ for all $l \in L$. Figure 4 illustrates the paths in $\bar{\mathcal{P}}_i$ and path Q_{li}.

In the first part of the proof we show that any truth assignment $\hat{x}$ that satisfies r clauses of the given Max-k-Sat(3) instance can be transformed into a set $\mathcal{S}(\hat{x})$ of $6|I| + r$ edge-disjoint 4-bounded (s,t)-paths in G. Let $\hat{x}$ be a truth assignment.

For each clause C_l that is satisfied by this truth assignment, let $i_l(\hat{x})$ be one of the variables whose literal evaluates to *true* in C_l. We define

$$\mathcal{S} = \mathcal{S}(\hat{x}) := \bigcup_{i \in I : \hat{x}_i = true} \mathcal{P}_i \cup \bigcup_{i \in I : \hat{x}_i = false} \bar{\mathcal{P}}_i \cup \{ Q_{l i_l(\hat{x})} \mid l \in L : C_l(\hat{x}) = true \} \ .$$

Clearly, all paths in $\mathcal{S}$ contain at most 4 edges, $|\mathcal{S}| = 6|I| + r$, and all paths in $\mathcal{S} \cap \bigcup_i (\mathcal{P}_i \cup \bar{\mathcal{P}}_i)$ are edge-disjoint. Note that if some path Q_{li} is contained in $\mathcal{S}$, then either the negated literal $\bar{x}_i^j$ occurring in clause C_l evaluates to *true*, which implies that $x_i = false$ and $P'_{ij} \notin \mathcal{S}$, or the unnegated literal x_i^j occurring in C_l evaluates to *true* and, hence, $\bar{P}'_{ij} \notin \mathcal{S}$. Furthermore, observe that these paths P'_{ij} and $\bar{P}'_{ij}$ are the only paths that may be contained in $\mathcal{S}$ and share an edge with Q_{li_l}. Consequently, each path $Q_{li} \in \mathcal{S}$ is edge-disjoint to any other path in $\mathcal{S}$ and, thus, all paths in $\mathcal{S}$ are edge-disjoint.

In the second part of the proof we show that any set $\mathcal{S}$ of $6|I| + r$ edge-disjoint 4-bounded (s,t)-paths in G can be transformed into a truth assignment $\hat{x}(\mathcal{S})$ that satisfies a least r clauses of the given MAX-k-SAT(3) instance. We may ignore path sets with $|\mathcal{S}| < 6|I|$, as the path set $\bigcup_{i \in I} \mathcal{P}_i$ is a feasible solution for the constructed MEDP(4) instance with $6|I|$ paths. Furthermore, we may restrict our attention to path sets $\mathcal{S}$ that satisfy the property that, for each $i \in I$, either $\mathcal{P}_i \subseteq \mathcal{S}$ or $\bar{\mathcal{P}}_i \subseteq \mathcal{S}$. Any path set $\mathcal{S}$ that does not satisfy this property can be turned into a path set $\mathcal{S}'$ with $|\mathcal{S}'| \geq \mathcal{S}$ that does as follows:

Suppose that, for some i, neither $\mathcal{P}_i \subseteq \mathcal{S}$ nor $\bar{\mathcal{P}}_i \subseteq \mathcal{S}$. Let $\mathcal{S}_i \subseteq \mathcal{S}$ be the set of paths in $\mathcal{S}$ that are fully contained in the variable subgraph G_i. As there are only 6 edges adjacent to t in G_i, we have $|\mathcal{S}_i| \leq 6$. Observe that each 4-bounded (s,t)-path in G is either of the form Q_{li} or it is fully contained in one of the variable subgraphs G_i. Furthermore, all (s,t)-paths of length exactly 4 in G_i are contained in $\mathcal{P}_i \cup \bar{\mathcal{P}}_i$. The only other 4-bounded paths in G_i are the three paths of length 3, which we denote $\bar{P}''_{ij} = (s, \bar{a}_i^j, \bar{w}_i^j, t)$ for the negated literals $\bar{x}_i^j$ and $P''_{ij} = (s, a_i^j, w_i^{j+1}, t)$ for the unnegated literals $\bar{x}_i^j$. In terms of edge-disjointness, however, the paths P''_{ij} and $\bar{P}''_{ij}$ conflict with the same 4-bounded (s,t)-paths as the paths P'_{ij} or $\bar{P}'_{ij}$, respectively. Replacing all paths P''_{ij} and $\bar{P}''_{ij}$ in $\mathcal{S}$ by the paths P'_{ij} and $\bar{P}'_{ij}$, respectively, thus yields a set of edge-disjoint 4-bounded path of the same size as $\mathcal{S}$. Hence, we can assume that $\mathcal{S}_i \subseteq \mathcal{P}_i \cup \bar{\mathcal{P}}_i$.

Now consider the paths Q_{il} corresponding to the clauses C_l in which variable x_i occurs. Recall that variable x_i occurs exactly 3 times in the clauses, so there are at most 3 paths Q_{il} in $\mathcal{S}$ that may share an edge with the paths in $\mathcal{P}_i \cup \bar{\mathcal{P}}_i$. If variable x_i occurs uniformly in all 3 clauses negated or unnegated, then these three paths Q_{il} are edge-disjoint from either all 6 paths in $\mathcal{P}_i$ or from all 6 paths in $\bar{\mathcal{P}}_i$. Replacing the paths in $\mathcal{S}_i$ by $\mathcal{P}_i$ or $\bar{\mathcal{P}}_i$, respectively, yields an edge-disjoint path set $\mathcal{S}'$ with $|\mathcal{S}'| \geq |\mathcal{S}|$. If variable x_i occurs non-uniformly, then either the paths in $\mathcal{P}_i$ or the paths in $\bar{\mathcal{P}}_i$ conflict with at most one of the three Q_{il} paths. In this case we have $\mathcal{S}_i \leq 5$, as the only edge-disjoint path sets of size 6 in $\mathcal{P}_i \cup \bar{\mathcal{P}}_i$ are $\mathcal{P}_i$ and $\bar{\mathcal{P}}_i$ themselves. Replacing the at most 5 paths in $\mathcal{S}_i$ and the 1 potentially conflicting path Q_{il} (if it is contained in $\mathcal{S}$ at all) by either $\mathcal{P}_i$ or $\bar{\mathcal{P}}_i$ thus yields a path set $\mathcal{S}'$ with $|\mathcal{S}'| \geq |\mathcal{S}|$ and either $\mathcal{P}_i \subseteq \mathcal{S}'$ or $\bar{\mathcal{P}}_i \subseteq \mathcal{S}'$. Repeating this procedure for all $i \in I$, we obtain a path set with the desired property.

So, suppose we are given a set $\mathcal{S}$ of 4-bounded edge-disjoint (s,t)-paths in G with $|\mathcal{S}| = 6|I| + r$ and $\mathcal{P}_i \subseteq \mathcal{S}$ or $\bar{\mathcal{P}}_i \subseteq \mathcal{S}$ for each $i \in I$. Then we define the truth assignment $\hat{x}(\mathcal{S})$ as

$$\hat{x}_i(\mathcal{S}) := \begin{cases} true & \text{if } \mathcal{P}_i \subset \mathcal{S}, \\ false & \text{otherwise} \end{cases} \quad \text{for all } i \in I.$$

To see that $\hat{x}(\mathcal{S})$ satisfies at least r clauses, consider the (s,t)-cut in G formed by the edges adjacent to node t. As $\mathcal{S}$ contains either $\mathcal{P}_i$ or $\bar{\mathcal{P}}_i$ for each $i \in I$, which amounts to a total of $6|I|$ paths, each of the remaining r paths in $\mathcal{S}$ must be of the form Q_{il} for some $i \in I$ and $l \in L$. Path Q_{il}, however, can be contained in $\mathcal{S}$ only if clause C_l evaluates to *true*. Otherwise it would intersect with the path P'_{ij} or $\bar{P}'_{ij}$ in $\mathcal{S}$ that corresponds to literal x_i^j occurring in clause C_l. Hence, at least r clauses of the given Max-k-Sat(3) instance are satisfied by the truth assignment $\hat{x}(\mathcal{S})$.

It now follows in a straightforward way that MEDP(4) is $\mathcal{APX}$-hard. Suppose there is an algorithm ALG to approximate MEDP(4) within a factor of $\alpha > 1$ and denote by $\mathcal{S}$ the solution computed by this algorithm. Let $r(\mathcal{S})$ be the number of clauses satisfied by the truth assignment $\hat{x}(\mathcal{S})$ and let $|\mathcal{S}^*|$ and r^* be optimal solution values of MEDP(4) and Max-k-Sat(3), respectively. The fact that at least half of the clauses in any Max-k-Sat(3) instance can be satisfied implies $r^* \geq \frac{1}{2}|L|$ and, further on, $r^* \geq \frac{3}{2k}|I|$. Applying the problem transformation and algorithm ALG to a given Max-k-Sat(3) instance, we get

$$r(\mathcal{S}) \geq |\mathcal{S}| - 6|I| \geq \frac{1}{\alpha}|\mathcal{S}^*| - 6|I| \geq \frac{1}{\alpha}(r^* + 6|I|) - 6|I| \geq \frac{1 + 4k - 4k\alpha}{\alpha} r^*$$

As there is a threshold $c > 1$ such that approximating Max-k-Sat(3) within a factor smaller than c is $\mathcal{NP}$-hard, it is also $\mathcal{NP}$-hard to approximate MEDP(4) within a factor less than $c' = \frac{4kc+c}{4kc+1} > 1$. □

Theorem 3 immediately implies the following corollary.

Corollary 4. *Given a graph $G = (V,E)$, $s,t \in V$, and $k \in \mathbb{Z}_+$, it is $\mathcal{NP}$-hard to decide if there are k edge-disjoint 4-bounded (s,t)-paths in G.*

Now consider the weighted problem WEDP(4). By Corollary 4, it is already $\mathcal{NP}$-hard to decide whether a given subgraph of the given graph contains k edge-disjoint (s,t)-path and, thus, comprises a feasible solution or not. Consequently, finding a minimum cost such subgraph is $\mathcal{NPO}$-complete.

Theorem 5. *WEDP(4) is $\mathcal{NPO}$-complete.*

As a consequence of Theorem 5, it is $\mathcal{NP}$-hard to approximate WEDP(4) within a factor $2^{f(n)}$ for any polynomial function f in the input size n of the problem.

3 Node-Disjoint 3- and 4-Bounded Paths

In this section we study the complexity of the node-disjoint paths problems. The maximum disjoint paths problem MNDP(ℓ) is known to be polynomially solvable for $\ell \leq 4$ and to be $\mathcal{APX}$-hard for $\ell \geq 5$ [19,18]. The weighted problem WNDP(ℓ) is solvable in polynomial time for $\ell \leq 2$, and $\mathcal{NPO}$-complete for $\ell \geq 5$. In the special case where $c_e \leq \sum_{f \in C-e} c_f$ holds for every cycle C in G and every edge $e \in C$, the weighted problem can be solved polynomially also for $\ell = 3$ and $\ell = 4$ [19]. For $\ell = 3$, the problem can still be solved efficiently if this condition is not satisfied.

Theorem 6. *WNDP(3) can be solved in polynomial time.*

Proof. Let S and T denote the set of neighbors of node s and t in the given graph G, respectively. We may assume w.l.o.g. that each node in G is contained in $\{s,t\} \cup S \cup T$, for otherwise it may not appear in any 3-bounded (s,t)-path.

We reduce WNDP(3) to the problem of finding a minimum weight matching with cardinality k in an auxiliary graph $G' = (V', E')$, which is constructed as follows: For each node $v \in S$ (resp. $w \in T$), there is an associated node $u_v \in V'$, (resp. $u'_w \in V'$). For each node $v \in S \cap T$, there is an associated edge $e_v = (u_v, u'_v) \in E'$ with weight $c_{sv} + c_{vt}$. Choosing this edge in the matching corresponds to choosing the path (s,v,t) in G. For each edge $(v,w) \in (S \times T) \setminus (S \cap T)^2$, there is an associated edge $(\overline{u}_v, \overline{u}_w) \in E'$, with $\overline{u}_z = u'_z$ if $z \in T$ and $\overline{u}_z = u_z$ otherwise for any $z \in V$. The weight of edge $(\overline{u}_v, \overline{u}_w)$ is set to $c_{sv} + c_{vw} + c_{wt}$. Choosing $(\overline{u}_v, \overline{u}_w)$ in the matching in G' corresponds to choosing path (s,v,w,t) in G. For each edge $(v,w) \in (S \cap T)^2$, there is an associated edge $(\overline{u}_v, \overline{u}_w) \in E'$, with weight $\min\{c_{sv} + c_{vw} + c_{wt},\ c_{sw} + c_{wv} + c_{vt}\}$, which represents the paths (s,v,w,t) and (s,w,v,t) in G. For each edge $(s,t) \in E$, there is an associated edge $(u_s, u_t) \in E'$ with weight c_{uv}.

Clearly, this construction can be performed in polynomial time. One easily verifies that any set of k vertex-disjoint 3-bounded (s,t)-paths in G corresponds to a matching of size k and the same cost in G', and vice versa. Since the cardinality constrained minimum weight matching problem can be solved in polynomial time [28,29], the claim follows. □

For $\ell = 4$, the problem becomes at least $\mathcal{APX}$-hard in the general case.

Theorem 7. *WNDP(4) is (at least) $\mathcal{APX}$-hard.*

Proof. We use a construction similar to the one presented in the previous section to reduce MAX-k-SAT(3) to WEDP(4). Again, we let x_i, $i \in I$, be the boolean variables and C_l, $l \in L$ be the clauses of the given MAX-k-SAT(3) instance and we denote the three occurrences of variable x_i by x_i^j, $j \in J := \{1, \ldots, 3\}$.

For each $l \in L$, we construct a clause graph $H_l = (W_l, F_l)$ exactly as in the proof of Theorem 3 and shown in Figure 3. For each $i \in I$, we construct a variable graph $G_i = (V_i, E_i)$ as

$$\begin{aligned} V_i :=& \{s,\, t\} \cup \{a_i^j,\, \bar{a}_i^j,\, u_i^j,\, \bar{u}_i^j,\, v_i^j,\, w_i^j,\, r_i^j \mid j \in J\} \quad \text{and} \\ E_i :=& \{sa_i^j,\, s\bar{a}_i^j,\, su_i^j,\, s\bar{u}_i^j,\, a_i^j u_i^j,\, \bar{a}_i^j \bar{u}_i^j,\, u_i^j v_i^j,\, \bar{u}_i^j v_i^j, \\ & v_i^j w_i^j,\, u_i^j r_i^j,\, \bar{u}_i^j r_i^{j+1},\, r_i^j t,\, w_i^j t \mid j \in J\} \ , \end{aligned}$$

where $r_i^4 = r_i^1$. Figure 5 illustrates these graphs. The graph G is obtained as the union of all G_i and H_l (sub-)graphs. Finally, we assign weight 1 to all edges su_i^j and $s\bar{u}_i^j$ and weight 0 to all other edges in G. The goal in the constructed WNDP(4) instance is to find a minimum cost subgraph of G that contains (at least) $6|I| + |L|$ node-disjoint 4-bounded (s,t)-paths.

For each $i \in I$ and $j \in J$, we denote the paths

$$\begin{aligned} P_{ij} &= (s, u_i^j, v_i^j, w_i^j, t) \ , & P'_{ij} &= (s, \bar{a}_i^j, \bar{u}_i^j, r_i^{j+1}, t) \ , & P''_{ij} &= (s, \bar{u}_i^j, r_i^{j+1}, t) \ , \\ \bar{P}_{ij} &= (s, \bar{u}_i^j, v_i^j, w_i^j, t) \ , & \bar{P}'_{ij} &= (s, a_i^j, u_i^j, r_i^j, t) \ , & \bar{P}''_{ij} &= (s, u_i^j, r_i^j, t) \ . \end{aligned}$$

For each variable x_i that occurs in clause C_l, we denote

$$Q_{li} = \begin{cases} (s, a_i^j, b_l, c_l, t) & \text{if literal } x_i^j \text{ occurs in } C_l, \\ (s, \bar{a}_i^j, b_l, c_l, t) & \text{if literal } \bar{x}_i^j \text{ occurs in } C_l. \end{cases}$$

Note that these are the only 4-bounded (s,t)-paths in G. Furthermore, we let $\mathcal{P}_i := \{P_{ij},\, P'_{ij} \mid j \in J\}$, $\bar{\mathcal{P}}_i := \{\bar{P}_{ij},\, \bar{P}'_{ij} \mid j \in J\}$, and $\mathcal{Q}_l := \{Q_{li} \mid i \in I : x_i \text{ occurs in } C_l\}$. Figure 5 illustrates the paths in $\bar{\mathcal{P}}_i$ and path Q_{li}.

As in the proof of Theorem 3, one finds that a truth assignment $\hat{x}$ that satisfies r clauses of the given Max-k-Sat(3) instance corresponds to a path set

$$\mathcal{S} = \mathcal{S}(\hat{x}) := \bigcup_{i \in I : \hat{x}_i = true} \mathcal{P}_i \cup \bigcup_{i \in I : \hat{x}_i = false} \bar{\mathcal{P}}_i \cup \{Q_{li_l(\hat{x})} \mid l \in L : C_l(\hat{x}) = true\}$$

with $|\mathcal{S}| = 6|I| + r$ and cost $c(\mathcal{S}) = 3|I|$. In order to obtain a set of $6|I| + |L|$ paths, we modify $\mathcal{S}$ as follows: For each $l \in L$ with $C_l(\hat{x}) = false$, we arbitrarily chose one i such that x_i^j or $\bar{x}_i^j$ occurs in C_l, add the path Q_{li} to $\mathcal{S}$, and replace the

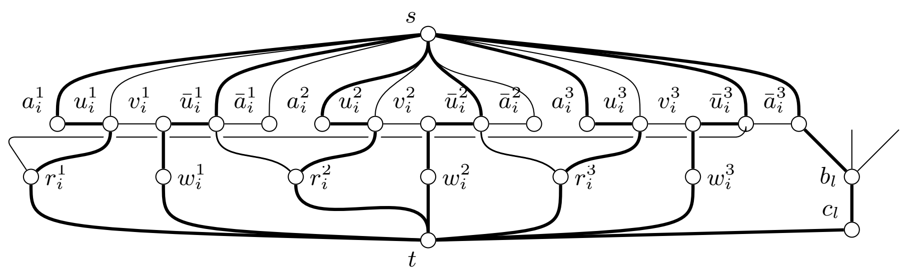


Fig. 5. Union of G_i and H_l for variable x_i and clause $C_k = (\bar{x}_i^3 \vee \dots)$. Thick lines are paths in $\bar{\mathcal{P}}_i$ and path Q_{li}.

path P'_{ij} or $\bar{P}'_{ij}$ in $\mathcal{S}$ with P''_{ij} or $\bar{P}''_{ij}$, respectively. This modification maintains the node-disjointness of the paths in $\mathcal{S}$ and increases both the size and the cost of $\mathcal{S}$ by $|L| - r$. The cost of the resulting path set $\mathcal{S}$ thus is

$$c(\mathcal{S}(\hat{x})) = 3|I| + |L| - r \ . \tag{1}$$

Conversely, one finds that any set $\mathcal{S}$ of $6|I| + |L|$ node-disjoint 4-bounded (s,t)-paths must contain one path from each set $\mathcal{Q}_l$ and 6 paths within each variable subgraph G_i. The only way to have 6 node-disjoint 4-bounded path within G_i, however, is to have either all 3 paths P_{ij} or all 3 paths $\bar{P}_{ij}$, complemented with 3 paths of the type P'_{ij} and P''_{ij} or with 3 paths of the type $\bar{P}'_{ij}$ and $\bar{P}''_{ij}$, respectively. The cost of such a path set is equal to the number of P_{ij} and $\bar{P}_{ij}$ paths it contains, which amounts to a total of $3|I|$, plus the number of P''_{ij} and $\bar{P}''_{ij}$ paths. Note that the paths P''_{ij} and $\bar{P}''_{ij}$ contain only a subset of the nodes in P'_{ij} and $\bar{P}'_{ij}$, respectively, and that the cost induced by P''_{ij} and $\bar{P}''_{ij}$ is 1, while the cost induced by P'_{ij} and $\bar{P}'_{ij}$ is 0. Thus, we may assume that $\mathcal{S}$ contains path P''_{ij} or $\bar{P}''_{ij}$ only if it contains path Q_{li} for the clause l in which literal x_i^j occurs. Let $\hat{x}(\mathcal{S})$ be the truth assignment defined as

$$\hat{x}_i(\mathcal{S}) := \begin{cases} true & \text{if } P_{i1} \in \mathcal{S}, \\ false & \text{otherwise,} \end{cases} \quad \text{for all } i \in I.$$

Consider a path $Q_{li} \in \mathcal{S}$ and suppose $C_l(\hat{x}(\mathcal{S})) = false$. Then also the literal x_i^j or $\bar{x}_i^j$ occurring in C_l evaluates to $false$. Since $\mathcal{S}$ contains either P_{ij} or $\bar{P}_{ij}$, it also must contain P''_{ij} or $\bar{P}''_{ij}$, respectively. As these paths induce a cost of one, the number of clauses satisfied by $\hat{x}(S)$ is

$$r(\hat{x}(\mathcal{S})) \geq |L| + 3|I| - c(\mathcal{S}) \ . \tag{2}$$

As in the proof of Theorem 3, it follows straightforward from (1) and (2) that approximation ratios are transformed linearly by the presented reduction and, hence, WNDP(4) is $\mathcal{APX}$-hard. □

Unfortunately, it remains open if WNDP(4) is approximable within a constant factor or not. The best known approximation ratio for WNDP(4) is $\mathcal{O}(k)$, which is achieved by a simple greedy algorithm.

Theorem 8. *WNDP(4) can be approximated within a factor of $4k$.*

Proof. Consider the algorithm, which adds the edges in order of non-decreasing cost until the constructed subgraph contains k node-disjoint 4-bounded (s,t)-paths and then returns the subgraph defined by these paths. As, in each iteration, we can check in polynomial time whether such paths exist or not [18], this algorithms runs in polynomial time. Furthermore, the optimal solution must contain at least one edge whose cost is at least as big as the cost of the last edge added by the greedy algorithm. Therefore, the total cost of the greedy solution is at most $4k$ times the optimal solution's cost. □

4 Conclusion

In this paper we show that the maximum edge-disjoint 4-bounded paths problem MEDP(4) is $\mathcal{APX}$-complete and that the corresponding weighted edge-disjoint paths problem WEDP(4) is $\mathcal{NPO}$-complete. The weighted node-disjoint ℓ-bounded paths problem was proven to be polynomially solvable for $\ell = 3$ and to be at least $\mathcal{APX}$-hard for $\ell = 4$. This closes all basic complexity issues that were left open in [18,19]. In addition, we presented a 2-approximation algorithm for WEDP(4) and a $4k$-approximation algorithm WNDP(4). It remains open whether WNDP(4) is approximable within a factor better than $\mathcal{O}(k)$ or if there is a stronger, non-constant approximation threshold.

The hardness results and algorithms presented in this paper also hold for directed graphs and for graphs containing parallel edges.

References

1. Alevras, D., Grötschel, M., Wessäly, R.: Capacity and survivability models for telecommunication networks. ZIB Technical Report SC-97-24, Konrad-Zuse-Zentrum für Informationstechnik Berlin (1997)
2. Gouveia, L., Patricio, P., Sousa, A.D.: Compact models for hop-constrained node survivable network design: An application to MPLS. In: Anandaligam, G., Raghavan, S. (eds.) Telecommunications Planning: Innovations in Pricing, Network Design and Management. Springer, Heidelberg (2005)
3. Gouveia, L., Patricio, P., Sousa, A.D.: Hop-constrained node survivable network design: An application to MPLS over WDM. Networks and Spatial Economics 8(1) (2008)
4. Grötschel, M., Monma, C., Stoer, M.: Design of Survivable Networks. In: Handbooks in Operations Research and Management Science, Volume Networks, pp. 617–672. Elsevier, Amsterdam (1993)
5. Chakraborty, T., Chuzhoy, J., Khanna, S.: Network design for vertex connectivity. In: Proceedings of the 40th Annual ACM Symposium on the Theory of Computing (STOC '08), pp. 167–176 (2008)
6. Chekuri, C., Khanna, S., Shepherd, F.: An $\mathcal{O}(\sqrt{n})$ approximation and integrality gap for disjoint paths and unsplittable flow. Theory of Computing 2, 137–146 (2006)
7. Lando, Y., Nutov, Z.: Inapproximability of survivable networks. Theoretical Computer Science 410, 2122–2125 (2009)
8. Menger, K.: Zur allgemeinen Kurventheorie. Fund. Mathematicae 10, 96–115 (1927)
9. Lovász, L., Neumann-Lara, V., Plummer, M.: Mengerian theorems for paths of bounded length. Periodica Mathematica Hungarica 9(4), 269–276 (1978)
10. Exoo, G.: On line disjoint paths of bounded length. Discrete Mathematics 44, 317–318 (1983)
11. Niepel, L., Safarikova, D.: On a generalization of Menger's theorem. Acta Mathematica Universitatis Comenianae 42, 275–284 (1983)
12. Ben-Ameur, W.: Constrained length connectivity and survivable networks. Networks 36, 17–33 (2000)

13. Pyber, L., Tuza, Z.: Menger-type theorems with restrictions on path lengths. Discrete Mathematics 120, 161–174 (1993)
14. Baier, G.: Flows with path restrictions. PhD thesis, Technische Universität Berlin (2003)
15. Martens, M., Skutella, M.: Length-bounded and dynamic k-splittable flows. In: Operations Research Proceedings 2005, pp. 297–302 (2006)
16. Mahjoub, A., McCormick, T.: Max flow and min cut with bounded-length paths: Complexity, algorithms and approximation. Mathematical Programming (to appear)
17. Baier, G., Erlebach, T., Hall, A., Köhler, E., Kolman, P., Panagrác, O., Schilling, H., Skutella, M.: Length-bounded cuts and flows. ACM Transactions on Algorithms (to appear)
18. Itai, A., Perl, Y., Shiloach, Y.: The complexity of finding maximum disjoint paths with length constraints. Networks 12, 277–286 (1982)
19. Bley, A.: On the complexity of vertex-disjoint length-restricted path problems. Computational Complexity 12, 131–149 (2003)
20. Perl, Y., Ronen, D.: Heuristics for finding a maximum number of disjoint bounded paths. Networks 14 (1984)
21. Wagner, D., Weihe, K.: A linear-time algorithm for edge-disjoint paths in planar graphs. Combinatorica 15(1), 135–150 (1995)
22. Botton, Q., Fortz, B., Gouveia, L.: The k-edge 3-hop constrained network design polyhedron. In: Proceedings of the 9th INFORMS Telecommunications Conference, Available as preprint at Université Catholique de Louvain: Le polyedre du probleme de conception de reseaux robustes K-arête connexe avec 3 sauts (2008)
23. Dahl, G., Huygens, D., Mahjoub, A., Pesneau, P.: On the k edge-disjoint 2-hop-constrained paths polytope. Operations Research Letters 34, 577–582 (2006)
24. Huygens, D., Mahjoub, A.: Integer programming formulations for the two 4-hop constrained paths problem. Networks 49(2), 135–144 (2007)
25. Golovach, P., Thilikos, D.: Paths of bounded length and their cuts: Parameterized complexity and algorithms. In: Chen, J., Fomin, F. (eds.) IWPEC 2009. LNCS, vol. 5917, pp. 210–221. Springer, Heidelberg (2009)
26. Guruswami, V., Khanna, S., Shepherd, B., Rajaraman, R., Yannakakis, M.: Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems. In: Proceedings of the 31st Annual ACM Symposium on the Theory of Computing (STOC '99), pp. 19–28 (1999)
27. Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A., Protasi, M.: Complexity and Approximation: Combinatorial Optimization Problems and their Approximability Properties. Springer, Heidelberg (1999)
28. Edmonds, J.: Maximum matching and a polyhedron with 0-1 vertices. Journal of Research of the National Bureau of Standards 69B, 125–130 (1965)
29. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency. Springer, Heidelberg (2003)

A Polynomial-Time Algorithm for Optimizing over N-Fold 4-Block Decomposable Integer Programs

Raymond Hemmecke[1], Matthias Köppe[2], and Robert Weismantel[3]

[1] Technische Universität Munich, Germany
[2] University of California, Davis, USA
[3] ETH, Zürich, Switzerland

Abstract. In this paper we generalize N-fold integer programs and two-stage integer programs with N scenarios to N-fold 4-block decomposable integer programs. We show that for fixed blocks but variable N, these integer programs are polynomial-time solvable for any linear objective. Moreover, we present a polynomial-time computable optimality certificate for the case of fixed blocks, variable N and any convex separable objective function. We conclude with two sample applications, stochastic integer programs with second-order dominance constraints and stochastic integer multi-commodity flows, which (for fixed blocks) can be solved in polynomial time in the number of scenarios and commodities and in the binary encoding length of the input data. In the proof of our main theorem we combine several non-trivial constructions from the theory of Graver bases. We are confident that our approach paves the way for further extensions.

Keywords: N-fold integer programs, Graver basis, augmentation algorithm, polynomial-time algorithm, stochastic multi-commodity flow, stochastic integer programming.

1 Introduction

Let $A \in \mathbb{Z}^{d\times n}$ be a matrix. We associate with A a finite set $\mathcal{G}(A)$ of vectors with remarkable properties. Consider the set $\ker(A)\cap\mathbb{Z}^n$. Then we put into $\mathcal{G}(A)$ all nonzero vectors $\mathbf{v} \in \ker(A)\cap\mathbb{Z}^n$ that cannot be written as a sum $\mathbf{v} = \mathbf{v}' + \mathbf{v}''$ of nonzero vectors $\mathbf{v}', \mathbf{v}'' \in \ker(A)\cap\mathbb{Z}^n$ that lie in the same orthant (or equivalently, have the same sign pattern in $\{\geq \mathbf{0}, \leq \mathbf{0}\}^n$) as $\mathbf{v}$. The set $\mathcal{G}(A)$ has been named the *Graver basis* of A, since Graver [6] introduced this set $\mathcal{G}(A)$ in 1975 and showed that it constitutes an optimality certificate for a whole family of integer linear programs that share the same problem matrix, A. By this we mean, that $\mathcal{G}(A)$ provides an augmenting vector/step to any non-optimal feasible solution and hence allows the design of a simple augmentation algorithm to solve the integer linear program.

In the last 10 years, a tremendous theoretical progress has been made in the theory of Graver bases. It has been shown that $\mathcal{G}(A)$ constitutes an optimality certificate for a much wider class of integer minimization problems, namely for those minimizing a concave or a separable convex objective function over $\{\mathbf{z} : A\mathbf{z} = \mathbf{b}, \mathbf{l} \leq \mathbf{z} \leq \mathbf{u}, \mathbf{z} \in \mathbb{Z}^n\}$ [2,13,15]. Moreover, it has been shown that only polynomially many Graver basis augmentation steps are needed to find a feasible solution and to turn it into an optimal feasible solution [7,8,18]. Finally, based on the fundamental finiteness results for certain highly structured matrices A (N-fold IPs and two- and multi-stage stochastic IPs) [1,9,10,17], it has been shown that concave and separable convex N-fold IPs and two- and multi-stage stochastic IPs can be solved in polynomial time [3,8] for fixed blocks.

In this paper, we will combine the two cases of N-fold IPs and of two-stage stochastic IPs by considering problems with a problem matrix that is N-fold 4-block decomposable as follows:

$$\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)} := \begin{pmatrix} C & D & D & \cdots & D \\ B & A & 0 & & 0 \\ B & 0 & A & & 0 \\ \vdots & & & \ddots & \\ B & 0 & 0 & & A \end{pmatrix}$$

for some given $N \in \mathbb{Z}_+$ and N copies of A. We call $\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)}$ an N-fold 4-block matrix. For $B = \mathbf{0}$ and $C = \mathbf{0}$ we recover the problem matrix of an N-fold IP and for $C = \mathbf{0}$ and $D = \mathbf{0}$ we recover the problem matrix of a two-stage stochastic IP.

Note that N-fold 4-block decomposable matrices also arise in the context of combinatorial optimization [19,20]. More precisely, for totally unimodular matrices C, A their 1-sum is totally unimodular ($B = \mathbf{0}, D = \mathbf{0}$). Similarly, total unimodularity is preserved under the 2-sum and 3-sum composition. Indeed, it can be verified that a repeated application of specialized 1-sum, 2-sum and 3-sum compositions leads to a particular family of N-fold 4-block decomposable matrices with structure regarding the matrices B and D.

Example. For matrices C and A, column vector $\mathbf{a}$ and row vector $\mathbf{b}^\intercal$ of appropriate dimensions, the 2-sum of $\left(\begin{smallmatrix} C & \mathbf{a} \end{smallmatrix}\right)$ and $\left(\begin{smallmatrix} \mathbf{b}^\intercal \\ A \end{smallmatrix}\right)$ gives $\left(\begin{smallmatrix} C & \mathbf{ab}^\intercal \\ \mathbf{0} & A \end{smallmatrix}\right)$. The 2-sum of $\left(\begin{smallmatrix} C & \mathbf{ab}^\intercal & a \\ \mathbf{0} & A & 0 \end{smallmatrix}\right)$ and $\left(\begin{smallmatrix} \mathbf{b}^\intercal \\ B \end{smallmatrix}\right)$ creates the matrix $\left(\begin{smallmatrix} C & \mathbf{ab}^\intercal & \mathbf{ab}^\intercal \\ \mathbf{0} & \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & A \end{smallmatrix}\right)$, which is the 2-fold 4-block decomposable matrix $\left(\begin{smallmatrix} C & \mathbf{ab}^\intercal \\ \mathbf{0} & A \end{smallmatrix}\right)^{(2)}$. □

Our main result is the following.

Theorem 1. *Let $A \in \mathbb{Z}^{d_A \times n_A}$, $B \in \mathbb{Z}^{d_A \times n_B}$, $C \in \mathbb{Z}^{d_C \times n_B}$, $D \in \mathbb{Z}^{d_C \times n_A}$ be fixed matrices. For given $N \in \mathbb{Z}_+$ let $\mathbf{l}, \mathbf{u} \in \mathbb{Z}^{n_B + N n_A}$, $\mathbf{b} \in \mathbb{Z}^{d_C + N d_A}$, and let $f : \mathbb{R}^{n_B + N n_A} \to \mathbb{R}$ be a separable convex function and denote by $\hat{f}$ the maximum of $|f|$ over the feasible region of the convex integer minimization problem*

$$(\mathrm{IP})_{N,\mathbf{b},\mathbf{l},\mathbf{u},f} : \qquad \min\left\{ f(\mathbf{z}) : \left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)} \mathbf{z} = \mathbf{b}, \mathbf{l} \leq \mathbf{z} \leq \mathbf{u}, \mathbf{z} \in \mathbb{Z}^{n_B + N n_A} \right\}.$$

We assume that f is given only by a comparison oracle that, when queried on $\mathbf{z}$ and $\mathbf{z}'$ decides whether $f(\mathbf{z}) < f(\mathbf{z}')$, $f(\mathbf{z}) = f(\mathbf{z}')$ or $f(\mathbf{z}) > f(\mathbf{z}')$. Then the following hold:

(a) There exists an algorithm that computes a feasible solution to $(\mathrm{IP})_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$ *or decides that no such solution exists and that runs in time polynomial in* N*, in the binary encoding lengths* $\langle \mathbf{l},\mathbf{u},\mathbf{b}\rangle$*.*

(b) Given a feasible solution $\mathbf{z}_0$ *to* $(\mathrm{IP})_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$*, there exists an algorithm that decides whether* $\mathbf{z}_0$ *is optimal or finds a better feasible solution* $\mathbf{z}_1$ *to* $(\mathrm{IP})_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$ *with* $f(\mathbf{z}_1) < f(\mathbf{z}_0)$ *and that runs in time polynomial in* N*, in the binary encoding lengths* $\langle \mathbf{l},\mathbf{u},\mathbf{b},\hat{f}\rangle$*, and in the number of calls to the evaluation oracle for* f*.*

(c) If f *is linear, there exists an algorithm that finds an optimal solution to* $(\mathrm{IP})_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$ *or decides that* $(\mathrm{IP})_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$ *is infeasible or unbounded and that runs in time polynomial in* N*, in the binary encoding lengths* $\langle \mathbf{l},\mathbf{u},\mathbf{b},\hat{f}\rangle$*, and in the number of calls to the evaluation oracle for* f*.*

This theorem generalizes a similar statement for N-fold integer programming and for two-stage stochastic integer programming. In these two special cases, one can even prove claim (c) of Theorem 1 for all separable convex functions and for a certain class of separable convex functions, respectively. It is a fundamental open question, whether one can construct not only some augmenting vector for a given separable convex objective function f in polynomially many steps but a best-improvement (or greedy) augmentation step $\alpha\mathbf{v}$ with $\alpha \in \mathbb{Z}_+$ and $\mathbf{v} \in \mathcal{G}\left(\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)}\right)$. If this can be done, part (c) of Theorem 1 can be extended from linear f to a class of separable convex functions f by applying the main result from [8].

In fact, Theorem 1 will be a consequence of the following structural result about $\mathcal{G}\left(\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)}\right)$.

Theorem 2. *If* $A \in \mathbb{Z}^{d_A \times n_A}$, $B \in \mathbb{Z}^{d_A \times n_B}$, $C \in \mathbb{Z}^{d_C \times n_B}$, $D \in \mathbb{Z}^{d_C \times n_A}$ *are fixed matrices, then* $\max\left\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}\left(\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)}\right)\right\}$ *is bounded by a polynomial in* N*.*

It should be noted that in the two special cases of N-fold IPs and of two-stage stochastic IPs each component of any Graver basis element is bounded by a constant (depending only on the fixed problem matrices and not on N). Hence, Theorem 2 specialized to these two cases is essentially trivial. In the general N-fold 4-block situation, however, each component of any Graver basis element is bounded only by a polynomial in N. This fact demonstrates that N-fold 4-block IPs are much richer and more difficult to solve than the two special cases of N-fold IPs and of two-stage stochastic IPs. Moreover, a proof of Theorem 2 in this general setting is not obvious.

In the next section, we present two sample applications of Theorem 1: stochastic integer programming with second-order dominance constraints [5,11] and stochastic integer multi-commodity flows [12,16]. For both cases we will develop tractability results based on our general theory. We do, however, not claim that these algorithms are particularly useful in practice. While the first application has an N-fold 4-block matrix as problem matrix, the second application can be modeled as an N-fold 4-block IP after a suitable transformation. To state the result, we introduce the following type of matrices. For given $N \in \mathbb{Z}_+$ let

$$\left[\begin{smallmatrix} A & B \\ D & C \end{smallmatrix}\right]^{(N)} := \begin{pmatrix} A & & & B & \cdots & B \\ & \ddots & & \vdots & & \vdots \\ & & A & B & \cdots & B \\ D & \cdots & D & C & & \\ \vdots & & \vdots & & \ddots & \\ D & \cdots & D & & & C \end{pmatrix},$$

where we have N copies of A and of C. Then the following holds.

Corollary 1. *Let $A \in \mathbb{Z}^{d_A \times n_A}$, $B \in \mathbb{Z}^{d_A \times n_B}$, $C \in \mathbb{Z}^{d_C \times n_B}$, $D \in \mathbb{Z}^{d_C \times n_A}$ be fixed matrices. For given $N \in \mathbb{Z}_+$ let $\mathbf{l}, \mathbf{u} \in \mathbb{Z}^{N(n_A+n_B)}$, $\mathbf{b} \in \mathbb{Z}^{N(d_A+d_C)}$, and let $f : \mathbb{R}^{N(n_A+n_B)} \to \mathbb{R}$ be a separable convex function and denote by $\hat{f}$ the maximum of $|f|$ over the feasible region of the convex integer minimization problem*

$$(\mathrm{IP})'_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}: \qquad \min\left\{ f(\mathbf{z}) : \left[\begin{smallmatrix} A & B \\ D & C \end{smallmatrix}\right]^{(N)} \mathbf{z} = \mathbf{b}, \mathbf{l} \le \mathbf{z} \le \mathbf{u}, \mathbf{z} \in \mathbb{Z}^{N(n_A+n_B)} \right\}.$$

We assume that f is given only by a comparison oracle that, when queried on $\mathbf{z}$ and $\mathbf{z}'$ decides whether $f(\mathbf{z}) < f(\mathbf{z}')$, $f(\mathbf{z}) = f(\mathbf{z}')$ or $f(\mathbf{z}) > f(\mathbf{z}')$. Then the following hold:

(a) There exists an algorithm that computes a feasible solution to $(\mathrm{IP})'_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$ or decides that no such solution exists and that runs in time polynomial in N, in the binary encoding lengths $\langle \mathbf{l}, \mathbf{u}, \mathbf{b} \rangle$.

(b) Given a feasible solution $\mathbf{z}_0$ to $(\mathrm{IP})'_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$, there exists an algorithm that decides whether $\mathbf{z}_0$ is optimal or finds a better feasible solution $\mathbf{z}_1$ to $(\mathrm{IP})'_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$ with $f(\mathbf{z}_1) < f(\mathbf{z}_0)$ and that runs in time polynomial in N, in the binary encoding lengths $\langle \mathbf{l}, \mathbf{u}, \mathbf{b}, \hat{f} \rangle$, and in the number of calls to the evaluation oracle for f.

(c) If f is linear, there exists an algorithm that finds an optimal solution to $(\mathrm{IP})'_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$ or decides that $(\mathrm{IP})'_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$ is infeasible or unbounded and that runs in time polynomial in N, in the binary encoding lengths $\langle \mathbf{l}, \mathbf{u}, \mathbf{b}, \hat{f} \rangle$, and in the number of calls to the evaluation oracle for f.

We do now present problems to which Theorem 1 and its Corollary 1 apply. Thereafter, we prove our claims. Our proof of Theorem 1 combines several non-trivial constructions from the theory of Graver bases. Although each of these constructions has been used before, we are confident that our combined approach paves the way for further extensions.

2 Sample Applications

In this section we present two N-fold 4-block decomposable integer programming problems that are polynomial-time solvable for given fixed blocks and variable N by Theorem 1 and its Corollary 1.

2.1 Stochastic Integer Multi-commodity Flow

Stochastic integer multi-commodity flows have been considered for example in [12,16]. Let us now introduce our setting. Let there be M integer (in contrast to continuous) commodities to be transported over a given network. While we assume that supply and demands are deterministic, we assume that the upper bounds for the capacities per edge are uncertain and given initially only via some probability distribution. The problem setup is as follows: first, we have to decide how to transport the M commodities over the given network without knowing the true capacities per edge. Then, after observing the true capacities per edge, penalties have to be paid if the capacity is exceeded. Assuming that we have knowledge about the probability distributions of the uncertain upper bounds, we wish to minimize the costs for the integer multi-commodity flow plus the expected penalties to be paid for exceeding capacities. To solve this problem, we discretize as usual the probability distribution for the uncertain upper bounds into N scenarios. Doing so, we obtain a (typically large-scale) (two-stage stochastic) integer programming problem with problem matrix

$$\begin{pmatrix} A & & & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ & \ddots & & \vdots & \vdots & & \vdots & \vdots \\ & & A & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ I & \cdots & I & I & -I & & & \\ \vdots & & \vdots & & & \ddots & & \\ I & \cdots & I & & & & I & -I \end{pmatrix}.$$

Herein, A is the node-edge incidence matrix of the given network, I is an identity matrix of appropriate size, and the columns containing $-I$ correspond to the penalty variables. If the network is kept fix, A, I, and $-I$ are fix, too. As the problem matrix is simply $\left[\begin{smallmatrix} A & \mathbf{0} \\ I & (I \ -I) \end{smallmatrix}\right]^{(N)}$, we can apply Corollary 1 and obtain the following.

Theorem 3. *For given fixed network the two-stage stochastic integer linear multi-commodity flow problem is solvable in polynomial time in the number M of commodities, in the number N of scenarios, and in the encoding lengths of the input data.*

Proof. The only issue that prevents us to apply Corollary 1 directly is the fact that M and N are different. But by introducing additional commodities or scenarios, we can easily obtain an equivalent (bigger) problem with $M = N$ for which we can apply Corollary 1. If $M < N$, we introduce additional commodities with zero flow and if $M > N$, we take one scenario, copy it additional $M-N$ times and choose for each of these $M-N+1$ identical scenarios $1/(M-N+1)$ times the original cost vector. So, in total, these $M-N+1$ scenarios are equivalent to the one we started from. $\square$

It should be noted that we can extend the problem and still get the same polynomiality result. For example, we may assume that we are allowed to change the routing of the M commodities in the second-stage decision. Penalties could be enforced for the amount of change of the first-stage decision or only for the amount of additional flow on edges compared to the first-stage decision. Writing down the constraints and introducing suitable additional variables with zero lower and upper bounds, one obtains again a problem matrix that allows the application of Corollary 1.

2.2 Stochastic Integer Programs with Second-Order Dominance Constraints

Stochastic integer programs with second-order dominance constraints were considered for example in [5,11]. We will consider here the special situation where all scenarios have the same probability. In Proposition 3.1 of [5], the following mixed-integer linear program was obtained as a deterministic equivalent to solve the stochastic problem at hand. We refer the reader to [5] for the details.

$$\text{(SIP)} : \min \left\{ \mathbf{g}^\intercal \mathbf{x} : \begin{array}{rl} \mathbf{c}^\intercal\mathbf{x} + \mathbf{q}^\intercal\mathbf{y}_{lk} - a_k \leq \mathbf{v}_{lk} & \forall l \forall k \\ T\mathbf{x} + W\mathbf{y}_{lk} = \mathbf{z}_l & \forall l \forall k \\ \sum_{l=1}^{L} \pi_l \mathbf{v}_{lk} \leq \bar{a}_k & \forall k \\ \mathbf{x} \in X, & \\ \mathbf{y}_{lk} \in \mathbb{Z}_+^{\bar{m}} \times \mathbb{R}_+^{m'}, \mathbf{v}_{lk} \geq \mathbf{0} & \forall l \forall k \end{array} \right\}$$

We assume now that all variables are integral and, for simplicity of exposition, we assume that the inequalities of the polyhedron X are incorporated into the constraints $T\mathbf{x} + W\mathbf{y}_{lk} = \mathbf{z}_l$. Moreover, we assume that all scenarios have the same probability, that is, $\pi_l = 1/L$, $l = 1, \ldots, L$.

Theorem 4. *For given fixed matrices T and W and for fixed number K, problem* (SIP) *is solvable in polynomial time in the number L of (data) scenarios, and in the encoding lengths of the input data.*

Proof. We transform the problem in such a way that Theorem 1 can be applied. First, we include the constraints $\mathbf{c}^\intercal\mathbf{x} + \mathbf{q}^\intercal\mathbf{y}_{lk} - a_k \leq \mathbf{v}_{lk}$ into the constraint $T\mathbf{x} + W\mathbf{y}_{lk} = \mathbf{z}_l$ (by adding slack variables to get an equation). Then, we set $\bar{T} = \begin{pmatrix} T \\ \vdots \\ T \end{pmatrix}$ and $\bar{W} = \begin{pmatrix} W & & \\ & \ddots & \\ & & W \end{pmatrix}$, in which we use K copies of T and W, respectively. As T, W, and K are assumed to be fixed, so are $\bar{T}$ and $\bar{W}$. With this, the problem matrix now becomes

$$\begin{pmatrix} \mathbf{0} & I & \cdots & I & I \\ \bar{T} & \bar{W} & & & \mathbf{0} \\ \vdots & & \ddots & & \vdots \\ \bar{T} & & & \bar{W} & \mathbf{0} \end{pmatrix}.$$

Introducing suitable additional variables with zero lower and upper bounds, we obtain a problem matrix of the form $\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(l)}$ with $A = \left(\begin{smallmatrix} \bar{W} & \mathbf{0} \end{smallmatrix}\right)$, $B = \bar{T}$, $C = \mathbf{0}$, and $D = \left(\begin{smallmatrix} I & I \end{smallmatrix}\right)$. Thus, we can apply Theorem 1 and the result follows. □

3 Proof of Main Results

For a concise introduction to Graver bases (and to the results on N-fold IPs), including short proofs to the main results, we refer the reader to the survey paper by Onn [14]. In this section, we state and prove results on Graver bases needed for the proof of our main theorem in the next section. Let us start by bounding the 1-norm of Graver basis elements of matrices with only one row. This lemma is a straight-forward consequence of Theorem 2 in [4].

Lemma 1. *Let $A \in \mathbb{Z}^{1\times n}$ be a matrix consisting of only one row and let M be an upper bound on the absolute values of the entries of A. Then we have* $\max\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}(A)\} \leq 2M - 1$.

Let us now prove some more general degree bounds on Graver bases that we will use in the proof of the main theorem below.

Lemma 2. *Let $A \in \mathbb{Z}^{d\times n}$ and let $B \in \mathbb{Z}^{m\times n}$. Moreover, put $C := \left(\begin{smallmatrix} A \\ B \end{smallmatrix}\right)$. Then we have*

$$\max\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}(C)\} \leq \max\{\|\boldsymbol{\lambda}\|_1 : \boldsymbol{\lambda} \in \mathcal{G}(B \cdot \mathcal{G}(A))\} \cdot \max\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}(A)\}.$$

Proof. Let $\mathbf{v} \in \mathcal{G}(C)$. Then $\mathbf{v} \in \ker(A)$ implies that $\mathbf{v}$ can be written as a non-negative integer linear sign-compatible sum $\mathbf{v} = \sum \lambda_i \mathbf{g}_i$ using Graver basis vectors $\mathbf{g}_i \in \mathcal{G}(A)$. Adding zero components if necessary, we can write $\mathbf{v} = \mathcal{G}(A)\boldsymbol{\lambda}$. We now claim that $\mathbf{v} \in \mathcal{G}(C)$ implies $\lambda \in \mathcal{G}(B \cdot \mathcal{G}(A))$ and the result follows.

First, observe that $\mathbf{v} \in \ker(B)$ implies $B\mathbf{v} = B \cdot (\mathcal{G}(A)\boldsymbol{\lambda}) = (B \cdot \mathcal{G}(A))\boldsymbol{\lambda} = \mathbf{0}$ and thus, $\boldsymbol{\lambda} \in \ker(B \cdot \mathcal{G}(A))$. If $\boldsymbol{\lambda} \notin \mathcal{G}(B \cdot \mathcal{G}(A))$, then it can be written as a sign-compatible sum $\boldsymbol{\lambda} = \boldsymbol{\mu} + \boldsymbol{\nu}$ with $\boldsymbol{\mu}, \boldsymbol{\nu} \in \ker(B \cdot \mathcal{G}(A))$. But then

$$\mathbf{v} = (\mathcal{G}(A)\boldsymbol{\mu}) + (\mathcal{G}(A)\boldsymbol{\nu})$$

gives a sign-compatible decomposition of $\mathbf{v}$ into vectors $\mathcal{G}(A)\boldsymbol{\mu}, \mathcal{G}(A)\boldsymbol{\nu} \in \ker(C)$, contradicting the minimality property of $\mathbf{v} \in \mathcal{G}(C)$. Hence, $\lambda \in \mathcal{G}(B \cdot \mathcal{G}(A))$ and the result follows. □

We will employ the following simple corollary.

Corollary 2. *Let $A \in \mathbb{Z}^{d\times n}$ and let $a^\intercal \in \mathbb{Z}^n$ be a row vector. Moreover, put $C := \left(\begin{smallmatrix} A \\ a \end{smallmatrix}\right)$. Then we have*

$$\max\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}(C)\} \leq (2 \cdot \max\{|a^\intercal \mathbf{v}| : \mathbf{v} \in \mathcal{G}(A)\} - 1) \cdot \max\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}(A)\}.$$

In particular, if $M := \max\{|a^{(i)}| : i = 1, \ldots, n\}$ then

$$\max\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}(C)\} \leq 2nM \left(\max\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}(A)\}\right)^2.$$

Proof. By Lemma 2, we already get

$$\max\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}(C)\} \leq \max\{\|\boldsymbol{\lambda}\|_1 : \boldsymbol{\lambda} \in \mathcal{G}(a^\intercal \cdot \mathcal{G}(A))\} \cdot \max\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}(A)\}.$$

Now, observe that $a^\intercal \cdot \mathcal{G}(A)$ is a $1 \times |\mathcal{G}(A)|$-matrix. Thus, the degree bound of primitive partition identities, Lemma 1, applies, which gives

$$\max\{\|\boldsymbol{\lambda}\|_1 : \boldsymbol{\lambda} \in \mathcal{G}(a \cdot \mathcal{G}(A))\} \leq 2 \cdot \max\left\{|a^\intercal \mathbf{v}| : \mathbf{v} \in \mathcal{G}(A)\right\} - 1,$$

and thus, the first claim is proved. The second claim is a trivial consequence of the first. □

Let us now extend this corollary to a form that we need to prove Theorem 1.

Corollary 3. *Let $A \in \mathbb{Z}^{d \times n}$ and let $B \in \mathbb{Z}^{m \times n}$. Let the entries of B be bounded by M in absolute value. Moreover, put $C := \left(\begin{smallmatrix} A \\ B \end{smallmatrix}\right)$. Then we have*

$$\max\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}(C)\} \leq (2nM)^{2^m - 1} \left(\max\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}(A)\}\right)^{2^m}.$$

Proof. This claim follows by simple induction, adding one row of B at a time, and by using the second inequality of Corollary 2 to bound the sizes of the intermediate Graver bases in comparison to the Graver basis of the matrix with one row of B less. □

In order to prove Theorem 1, let us consider the submatrix $\left(\begin{smallmatrix} \mathbf{0} & \mathbf{0} \\ B & A \end{smallmatrix}\right)^{(N)}$. A main result from [9] is the following.

Lemma 3. *Let $A \in \mathbb{Z}^{d_A \times n_A}$ and $B \in \mathbb{Z}^{d_A \times n_B}$. There exists a number $g \in \mathbb{Z}_+$ depending only on A and B but not on N such that for every $N \in \mathbb{Z}_+$ and for every $\mathbf{v} \in \mathcal{G}\left(\left(\begin{smallmatrix} \mathbf{0} & \mathbf{0} \\ B & A \end{smallmatrix}\right)^{(N)}\right)$, the components of $\mathbf{v}$ are bounded by g in absolute value. In particular, $\|\mathbf{v}\|_1 \leq (n_B + Nn_A)g$ for all $\mathbf{v} \in \mathcal{G}\left(\left(\begin{smallmatrix} \mathbf{0} & \mathbf{0} \\ B & A \end{smallmatrix}\right)^{(N)}\right)$.*

Combining this result with Corollary 3, we get a bound for the 1-norms of the Graver basis elements of $\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)}$. Note that the second claim of the following corollary is exactly Theorem 2.

Corollary 4. *Let $A \in \mathbb{Z}^{d_A \times n_A}$, $B \in \mathbb{Z}^{d_A \times n_B}$, $C \in \mathbb{Z}^{d_C \times n_B}$, $D \in \mathbb{Z}^{d_C \times n_A}$ be given matrices. Moreover, let M be a bound on the absolute values of the entries in C and D, and let $g \in \mathbb{Z}_+$ be the number from Lemma 3. Then for any $N \in \mathbb{Z}_+$ we have*

$$\begin{aligned} &\max\left\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}\left(\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)}\right)\right\} \\ &\leq (2(n_B + Nn_A)M)^{2^{d_C} - 1} \left(\max\left\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}\left(\left(\begin{smallmatrix} \mathbf{0} & \mathbf{0} \\ B & A \end{smallmatrix}\right)^{(N)}\right)\right\}\right)^{2^{d_C}} \\ &\leq (2(n_B + Nn_A)M)^{2^{d_C} - 1} \left((n_B + Nn_A)g\right)^{2^{d_C}}. \end{aligned}$$

If A, B, C, D are fixed matrices, then $\max\left\{\|\mathbf{v}\|_1 : \mathbf{v} \in \mathcal{G}\left(\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)}\right)\right\}$ is bounded by a polynomial in N.

Proof. While the first claim is a direct consequence of Lemma 3 and Corollary 3, the polynomial bound for fixed matrices A, B, C, D and varying N follows immediately by observing that n_A, n_B, d_C, M, g are constants as they depend only on the fixed matrices A, B, C, D. $\square$

Now we are ready to prove our main theorem.

Proof of Theorem 1. Let $N \in \mathbb{Z}_+$, $\mathbf{l}, \mathbf{u} \in \mathbb{Z}^{n_B+Nn_A}$, $\mathbf{b} \in \mathbb{Z}^{d_C+Nd_A}$, and a separable convex function $f : \mathbb{R}^{n_B+Nn_A} \to \mathbb{R}$ be given. To prove claim (a), observe that one can turn any integer solution to $\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)} \mathbf{z} = \mathbf{b}$ (which can be found in polynomial time using for example the Hermite normal form of $\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)}$) into a feasible solution (that in addition fulfills $\mathbf{l} \leq \mathbf{z} \leq \mathbf{u}$) by a sequence of linear integer programs (with the same problem matrix $\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)}$) that "move" the components of $\mathbf{z}$ into the direction of the given bounds, see [7]. This step is similar to phase I of the Simplex Method in linear programming. In order to solve these linear integer programs, it suffices (by the result of [18]) to find Graver basis augmentation vectors from $\mathcal{G}\left(\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)}\right)$ for a directed augmentation oracle. So, claim (b) will imply both claim (a) and claim (c).

Let us now assume that we are given a feasible solution $\mathbf{z}_0 = (\mathbf{x}, \mathbf{y}_1, \ldots, \mathbf{y}_N)$ and that we wish to decide whether there exists another feasible solution $\mathbf{z}_1$ with $f(\mathbf{z}_1) < f(\mathbf{z}_0)$. By the main result in [13], it suffices to decide whether there exists some vector $\mathbf{v} = (\hat{\mathbf{x}}, \hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_N)$ in the Graver basis of $\left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)}$ such that $\mathbf{z}_0 + \mathbf{v}$ is feasible and $f(\mathbf{z}_0 + \mathbf{v}) < f(\mathbf{z}_0)$. By Corollary 4 and by the fact that n_B is constant, there is only a polynomial number of candidates for the $\hat{\mathbf{x}}$-part of $\mathbf{v}$. For each such candidate $\hat{\mathbf{x}}$, we can find a best possible choice for $\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_N$ by solving the following separable convex N-fold IP:

$$\min \left\{ f\left(\begin{pmatrix} \mathbf{x}+\hat{\mathbf{x}} \\ \mathbf{y}_1+\hat{\mathbf{y}}_1 \\ \vdots \\ \mathbf{y}_N+\hat{\mathbf{y}}_N \end{pmatrix}\right) : \begin{array}{c} \left(\begin{smallmatrix} C & D \\ B & A \end{smallmatrix}\right)^{(N)} \begin{pmatrix} \mathbf{x}+\hat{\mathbf{x}} \\ \mathbf{y}_1+\hat{\mathbf{y}}_1 \\ \vdots \\ \mathbf{y}_N+\hat{\mathbf{y}}_N \end{pmatrix} = \mathbf{b}, \\ \mathbf{l} \leq \begin{pmatrix} \mathbf{x}+\hat{\mathbf{x}} \\ \mathbf{y}_1+\hat{\mathbf{y}}_1 \\ \vdots \\ \mathbf{y}_N+\hat{\mathbf{y}}_N \end{pmatrix} \leq \mathbf{u}, \\ \mathbf{y}_1, \ldots, \mathbf{y}_N \in \mathbb{Z}^{n_A} \end{array} \right\},$$

for given $\mathbf{z}_0 = (\mathbf{x}, \mathbf{y}_1, \ldots, \mathbf{y}_N)$ and $\hat{\mathbf{x}}$. Observe that the problem $(\mathrm{IP})_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$ does indeed simplify to a separable convex N-fold IP with problem matrix $\left(\begin{smallmatrix} 0 & D \\ 0 & A \end{smallmatrix}\right)^{(N)}$ because $\mathbf{z}_0 = (\mathbf{x}, \mathbf{y}_1, \ldots, \mathbf{y}_N)$ and $\hat{\mathbf{x}}$ are fixed. For fixed matrices A and D, however, each such N-fold IP is solvable in polynomial time [8]. If the N-fold IP is feasible and if for the resulting optimal vector $\mathbf{v} := (\hat{\mathbf{x}}, \hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_N)$ we have $f(\mathbf{z}_0 + \mathbf{v}) \geq f(\mathbf{z}_0)$, then no augmenting vector can be constructed using this particular choice of $\hat{\mathbf{x}}$. If on the other hand we have $f(\mathbf{z}_0 + \mathbf{v}) < f(\mathbf{z}_0)$, then $\mathbf{v}$ is a desired augmenting vector for $\mathbf{z}_0$ and we can stop. As we solve polynomially many polynomially solvable N-fold IPs, claim (b) and thus also claims (a) and (c) follow. □

Proof of Corollary 1. To prove Corollary 1, observe that after introducing additional variables, problem $(\mathrm{IP})'_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$ can be modeled as an N-fold 4-block IP and is thus polynomial-time solvable by Theorem 1. First, write the constraint $\left[\begin{smallmatrix} A & B \\ D & C \end{smallmatrix}\right]^{(N)} \mathbf{z} = \mathbf{b}$ in $(\mathrm{IP})'_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$ as follows:

$$\begin{pmatrix} A & & & B & \cdots & B \\ & \ddots & & \vdots & & \vdots \\ & & A & B & \cdots & B \\ D & \cdots & D & C & & \\ \vdots & & \vdots & & \ddots & \\ D & \cdots & D & & & C \end{pmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \\ \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_N \end{pmatrix} = \begin{pmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_N \\ \mathbf{b}_{N+1} \\ \vdots \\ \mathbf{b}_{2N} \end{pmatrix}.$$

Now introduce variables $\mathbf{w}_x = \sum_{i=1}^N \mathbf{x}_i$ and $\mathbf{w}_y = \sum_{i=1}^N \mathbf{y}_i$. Then we get the new constraints

$$\left(\begin{array}{cc|ccccccc} -I & & I & & I & & \cdots & I & \\ & -I & & I & & I & \cdots & & I \\ \hline D & & & C & & & & & \\ & B & A & & & & & & \\ D & & & & & C & & & \\ & B & & & A & & & & \\ \vdots & & & & & & \ddots & & \\ D & & & & & & & & C \\ & B & & & & & & A & \end{array}\right) \begin{pmatrix} \mathbf{w}_x \\ \mathbf{w}_y \\ \mathbf{x}_1 \\ \mathbf{y}_1 \\ \vdots \\ \mathbf{x}_N \\ \mathbf{y}_N \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_N \\ \mathbf{b}_{N+1} \\ \vdots \\ \mathbf{b}_{2N} \end{pmatrix}.$$

Hence, $(\mathrm{IP})'_{N,\mathbf{b},\mathbf{l},\mathbf{u},f}$ can be modeled as an N-fold 4-block decomposable IP and thus, Corollary 1 follows by applying Theorem 1 to this transformed integer program. □

Acknowledgments. We wish to thank Rüdiger Schultz for valuable comments on Section 2 and for pointing us to [5]. The second author was supported by grant DMS-0914873 of the National Science Foundation.

References

1. Aschenbrenner, M., Hemmecke, R.: Finiteness theorems in stochastic integer programming. Foundations of Computational Mathematics 7, 183–227 (2007)
2. De Loera, J.A., Hemmecke, R., Onn, S., Rothblum, U., Weismantel, R.: Convex integer maximization via Graver bases. Journal of Pure and Applied Algebra 213, 1569–1577 (2009)
3. De Loera, J.A., Hemmecke, R., Onn, S., Weismantel, R.: N-fold integer programming. Discrete Optimization 5, 231–241 (2008)

4. Diaconis, P., Graham, R., Sturmfels, B.: Primitive partition identities. In: Miklós, D., Sós, V.T., Szonyi, T. (eds.) Combinatorics, Paul Erdos is Eighty, pp. 173–192. Janos Bolyai Mathematical Society, Budapest (1996)
5. Gollmer, R., Gotzes, U., Schultz, R.: A note on second-order stochastic dominance constraints induced by mixed-integer linear recourse. Mathematical Programming (to appear, 2010), doi:10.1007/s10107-009-0270-0
6. Graver, J.E.: On the foundation of linear and integer programming I. Mathematical Programming 9, 207–226 (1975)
7. Hemmecke, R.: On the positive sum property and the computation of Graver test sets. Mathematical Programming 96, 247–269 (2003)
8. Hemmecke, R., Onn, S., Weismantel, R.: A polynomial oracle-time algorithm for convex integer minimization. Mathematical Programming, Series A (to appear, 2010), doi:10.1007/s10107-009-0276-7
9. Hemmecke, R., Schultz, R.: Decomposition of test sets in stochastic integer programming. Mathematical Programming 94, 323–341 (2003)
10. Hoşten, S., Sullivant, S.: Finiteness theorems for Markov bases of hierarchical models. Journal of Combinatorial Theory, Series A 114(2), 311–321 (2007)
11. New Formulations for Optimization Under Stochastic Dominance Constraints. SIAM J. Optim. 19, 1433–1450 (2008)
12. Mirchandani, P.B., Soroush, H.: The stochastic multicommodity flow problem. Networks 20, 121–155 (1990)
13. Murota, K., Saito, H., Weismantel, R.: Optimality criterion for a class of nonlinear integer programs. Operations Research Letters 32, 468–472 (2004)
14. Onn, S.: Theory and Applications of N-fold Integer Programming. In: IMA Volume on Mixed Integer Nonlinear Programming. Frontier Series. Springer, Heidelberg (in preparation 2010)
15. Onn, S., Rothblum, U.: Convex combinatorial optimization. Discrete Computational Geometry 32, 549–566 (2004)
16. Powell, W.B., Topaloglu, H.: Dynamic-Programming Approximations for Stochastic Time-Staged Integer Multicommodity-Flow Problems. INFORMS Journal on Computing 18, 31–42 (2006)
17. Santos, F., Sturmfels, B.: Higher Lawrence configurations. Journal of Combinatorial Theory, Series A 103, 151–164 (2003)
18. Schulz, A.S., Weismantel, R.: A polynomial time augmentation algorithm for integer programming. In: Proc. of the 10th ACM-SIAM Symposium on Discrete Algorithms, Baltimore (1999)
19. Schrijver, A.: Theory of linear and integer programming. Wiley, Chichester (1986)
20. Seymour, P.D.: Decomposition of regular matroids. Journal of Combinatorial Theory, Series B 28, 305–359 (1980)

Universal Sequencing on a Single Machine

Leah Epstein[1], Asaf Levin[2], Alberto Marchetti-Spaccamela[3,⋆], Nicole Megow[4], Julián Mestre[4], Martin Skutella[5,⋆⋆], and Leen Stougie[6,⋆⋆⋆]

[1] Dept. of Mathematics, University of Haifa, Israel
lea@math.haifa.ac.il
[2] Chaya fellow. Faculty of Industrial Engineering and Management, The Technion, Haifa, Israel
levinas@ie.technion.ac.il
[3] Dept. of Computer and System Sciences, Sapienza University of Rome, Italy
alberto.marchetti@dis.uniroma1.it
[4] Max-Planck-Institut für Informatik, Saarbrücken, Germany
{nmegow,jmestre}@mpi-inf.mpg.de
[5] Inst. für Mathematik, Technische Universität Berlin, Germany
martin.skutella@tu-berlin.de.
[6] Dept. of Econometrics and Operations Research, Vrije Universiteit Amsterdam & CWI, Amsterdam, The Netherlands
stougie@cwi.nl

Abstract. We consider scheduling on an unreliable machine that may experience unexpected changes in processing speed or even full breakdowns. We aim for a universal solution that performs well without adaptation for any possible machine behavior. For the objective of minimizing the total weighted completion time, we design a polynomial time deterministic algorithm that finds a universal scheduling sequence with a solution value within 4 times the value of an optimal clairvoyant algorithm that knows the disruptions in advance. A randomized version of this algorithm attains in expectation a ratio of e. We also show that both results are best possible among all universal solutions. As a direct consequence of our results, we answer affirmatively the question of whether a constant approximation algorithm exists for the offline version of the problem when machine unavailability periods are known in advance.

When jobs have individual release dates, the situation changes drastically. Even if all weights are equal, there are instances for which any universal solution is a factor of $\Omega(\log n / \log\log n)$ worse than an optimal sequence. Motivated by this hardness, we study the special case when the processing time of each job is proportional to its weight. We present a non-trivial algorithm with a small constant performance guarantee.

1 Introduction

Traditional scheduling problems normally assume that jobs run on an ideal machine that provides a constant performance throughout time. While in some settings this is a good

⋆ Supported by EU project 215270 FRONTS.
⋆⋆ Supported by DFG research center MATHEON in Berlin.
⋆⋆⋆ Supported by the Dutch BSIK-BRIKS project.

enough approximation of real life machine behavior, in other situations this assumption is decidedly unreasonable. Our machine, for example, can be a server shared by multiple users; if other users suddenly increase their workload, this can cause a general slowdown; or even worse, the machine may become unavailable for a given user due to priority issues. In other cases, our machine may be a production unit that can break down altogether and remain offline for some time until it is repaired. In these cases, it is crucial to have schedules that take such unreliable machine behavior into account.

Different machine behaviors will typically lead to widely different optimal schedules. This creates a burden on the scheduler who would have to periodically recompute the schedule from scratch. In some situations, recomputing the schedule may not even be feasible: when submitting a set of jobs to a server, a user can choose the order in which it presents these jobs, but cannot alter this ordering later on. Therefore, it is desirable in general to have a fixed master schedule that will perform well regardless of the actual machine behavior. In other words, we want a *universal schedule* that, for any given machine behavior, has cost close to that of an optimal clairvoyant algorithm.

In this paper we initiate the study of universal scheduling by considering the problem of sequencing jobs on a single machine to minimize average completion times. Our main result is an algorithm for computing a universal schedule that is always a constant factor away from an optimal clairvoyant algorithm. We complement this by showing that our upper bound is best possible among universal schedules. We also consider the case when jobs have release dates. Here we provide an almost logarithmic lower bound on the performance of universal schedules, thus showing a drastic difference with respect to the setting without release dates. Finally, we design an algorithm with constant performance for the interesting case of scheduling jobs with release dates and proportional weights. Our hope is that these results stimulate the study of universal solutions for other scheduling problems, and, more broadly, the study of more realistic scheduling models. In the rest of this section we introduce our model formally, discuss related work, and explain our contributions in detail.

The model. We are given a job set J with processing times $p_j \in \mathbb{Q}^+$ and weights $w_j \in \mathbb{Q}^+$ for each job $j \in J$. Using a standard scaling argument, we can assume w.l.o.g. that $w_j \geq 1$ for $j \in J$. The problem is to find a sequence π of jobs to be scheduled on a single machine that minimizes the total sum of weighted completion times. The jobs are processed in the prefixed order π no matter how the machine may change its processing speed or whether it becomes unavailable. In case of a machine breakdown the currently running job is preempted and will be resumed processing at any later moment when the machine becomes available again. We analyze the worst case performance by comparing the solution value provided by an algorithm with that of an optimal clairvoyant algorithm that knows the machine behavior in advance, and that is even allowed to preempt jobs at any time.

We also consider the more general problem in which each job $j \in J$ has its individual release date $r_j \geq 0$, which is the earliest point in time when it can start processing. In this model, it is necessary to allow job preemption, otherwise no constant performance guarantee is possible as simple examples show. We allow preemption in the actual scheduling procedure, however, as in the case without release dates, we aim for non-adaptive universal solutions. That is, a schedule will be specified by a total ordering of the jobs. At any point in time we work on the first job in this ordering that has not finished yet and that has already been released. This procedure is called *preemptive list scheduling* [9, 28]. Note that a newly released job will preempt the job that is currently running if it comes earlier than the current job in the ordering.

Related work. The concept of *universal* solutions, that perform well for every single input of a superset of possible inputs, has been used already decades ago in different contexts, as e.g. in hashing [4] and routing [31]. The latter is also known as *oblivious routing* and has been studied extensively; see [26] for a state-of-the-art overview. Jia et al. [12] considered universal approximations for TSP, Steiner Tree, and Set Cover Problems. All this research falls broadly into the field of robust optimization [3]. The term *robust* is not used consistently in the literature. In particular, the term *robust scheduling* refers mainly to robustness against uncertain processing times; see e.g. [17, chap. 7] and [23]. Here, quite strong restrictions on the input or weakened notions of robustness are necessary to guarantee meaningful worst case solutions. We emphasize, that our results in this paper are robust in the most conservative, classical notion of robustness originating by Soyster [30], also called *strict robustness* [22], and in this regard, we follow the terminology of universal solutions.

Scheduling with limited machine availability is a subfield of machine scheduling that has been studied for over twenty years; see, e.g., the surveys [27, 20, 7]. Different objective functions, stochastic breakdowns, as well as the offline problem with known availability periods have been investigated. Nevertheless, only few results are known on the problem of scheduling to minimize the total weighted completion time, and none of these deal with release dates. If all jobs have equal weights, a simple interchange argument shows that sequencing jobs in non-increasing order of processing times is optimal as it is in the setting with continuous machine availability [29]. Obviously, this result immediately transfers to the universal setting in which machine breakdowns or changes in processing speeds are not known beforehand. The special case of proportional jobs, in which the processing time of each job is proportional to its weight, has been studied in [32]. The authors showed that scheduling in non-increasing order of processing times (or weights) yields a 2-approximation for preemptive scheduling. However, for the general problem with arbitrary job weights, it remained an open question [32] if a polynomial time algorithm with constant approximation ratio exists, even without release dates. In this case, the problem is strongly NP-hard [32].

A major line of research within this area focused on the offline scheduling problem with a single unavailable period. This problem is weakly NP-hard in both, the preemptive [19] and the non-preemptive variant [1, 21]. Several approximation results have been derived, see [19, 21, 32, 13, 24]. Only very recently, and independently of us, Kellerer and Strusevich [16] derived FPTASes with running time $\mathcal{O}(n^4/\epsilon^2)$ for the non-preemptive problem and $\mathcal{O}(n^6/\epsilon^3)$ in the preemptive case. An even improved non-preemptive FPTAS with running time $\mathcal{O}(n^2/\epsilon^2)$ is claimed in [14]. However, the proof seems incomplete in bounding the deviation of an algorithm's solution from an optimal one; in particular, the claim after Ineq. (11) in the proof of Lem. 1 is not proved.

Our results. Our main results are algorithms that compute deterministic and randomized universal schedules for jobs without release dates. These algorithms run in polynomial time and output an ordering of the jobs such that scheduling the jobs in this order will always yield a solution that remains within multiplicative factor 4 and within multiplicative factor e in expectation from any given schedule. Furthermore, we show that our algorithms can be adapted to solve more general problem instances with certain types of precedence constraints without loosing performance quality. We also show that our upper bounds are best possible for universal scheduling. This is done by establishing an interesting connection between our problem and a certain online bidding problem [5].

It may seem rather surprising that universal schedules with constant performance guarantee should always exist. In fact, our results immediately answer affirmatively an open question in the area of offline scheduling with limited machine availability: whether there exists a constant factor approximation algorithm for scheduling jobs in a machine having multiple unavailable periods that are known in advance.

To derive our results, we study the objective of minimizing the total weight of uncompleted jobs at any point in time. First, we show that the performance guarantee is given directly by a bound on the ratio between the remaining weight of our algorithm and that of an optimal clairvoyant algorithm at every point in time on an ideal. Then, we devise an algorithm that computes the job sequence iteratively backwards: in each iteration we find a subset of jobs with largest total processing time subject to a bound on their total weight. The bound is doubled in each iteration. Our approach is related to, but not equivalent to, an algorithm of Hall et al. [9] for online scheduling on ideal machines—the doubling there happens in the time horizon. Indeed, this type of *doubling* strategy has been applied successfully in the design of algorithms for various problems; the interested reader is referred to the excellent survey of Chrobak and Kenyon-Mathieu [6] for a collection of such examples.

The problem of minimizing the total weight of uncompleted jobs at any time was previously considered [2] in the context of on-line scheduling to minimize flow time on a single machine; there, a constant approximation algorithm is presented with a worst case bound of 24. Our results imply an improved 4-approximation for this problem. Furthermore, we show that the same guarantee holds for the setting with release dates; unfortunately, unlike in the case without release dates, this does not translate into the same performance guarantee for universal schedules. In fact, when jobs have individual release dates, the problem changes drastically.

In Section 4 we show that in the presence of release dates, even if all weights are equal, there are instances for which the ratio between the value of any universal solution and that of an optimal schedule is $\Omega(\log n/\log\log n)$. Our proof relies on the classical theorem of Erdős and Szekeres [8] on the existence of long increasing/decreasing subsequences of a given sequence of numbers. Motivated by this hardness, we study the class of instances with proportional jobs. We present a non-trivial algorithm and prove a performance guarantee of 5. Additionally, we give a lower bound of 3 for all universal solutions in this special case.

Our last result, Section 5, is a fully polynomial time approximation scheme (FPTAS) for offline scheduling on a machine with a single unavailable period. Compared to the FPTAS presented recently in [16], our scheme, which was discovered independently from the former, is faster and seems to be simpler, even though the basic ideas are similar. Our FPTAS for the non-preemptive variant has running time $\mathcal{O}(n^3/\epsilon^2)$ and for the preemptive variant $\mathcal{O}(n^4/\epsilon^3 \log p_{\max})$.

2 Preliminaries and Key Observations

Given a single machine that runs continuously at unit speed (ideal machine), the completion time C_j^π of job j when applying preemptive list scheduling to sequence π is uniquely defined. For some point in time $t \geq 0$ let $W^\pi(t)$ denote the total weight of jobs that are not yet completed by time t according to sequence π, i.e., $W^\pi(t) := \sum_{j:C_j^\pi > t} w_j$. Then,

$$\sum_{j\in J} w_j C_j^\pi = \int_0^\infty W^\pi(t)dt. \tag{1}$$

Clearly, breaks or fluctuations in the speed of the machine delay the completion times. To describe a particular machine behavior, let $f : \mathbb{R}_+ \to \mathbb{R}_+$ be a non-decreasing continuous function, with $f(t)$ being the aggregated amount of processing time available on the machine up to time t. We refer to f as the *machine capacity function*. If the derivative of f at time t exists, it can be interpreted as the speed of the machine at that point in time.

For a given capacity function f, let $S(\pi, f)$ denote the single machine schedule when applying preemptive list scheduling to permutation π, and let $C_j^{S(\pi,f)}$ denote the completion time of job j in this particular schedule. For some point in time $t \geq 0$, let $W^{S(\pi,f)}(t)$ denote the total weight of jobs that are not yet completed by time t in schedule $S(\pi, f)$. Then,

$$\sum_{j\in J} w_j C_j^{S(\pi,f)} = \int_0^\infty W^{S(\pi,f)}(t)dt\,.$$

For $t \geq 0$ let $W^{S^*(f)}(t) := \min_\pi W^{S(\pi,f)}(t)$.

Observation 1. *For a given machine capacity function f,*

$$\int_0^\infty W^{S^*(f)}(t)dt \tag{2}$$

is a lower bound on the objective function of any schedule.

We construct a universal sequence of jobs π such that, no matter how the single machine behaves, the objective value of the corresponding schedule $S(\pi, f)$ is within a constant factor of the optimum.

Lemma 1. *Let π be a sequence of jobs, and let $c > 0$. Then, the value $\sum_{j\in J} w_j C_j^{S(\pi,f)}$ is at most c times the optimum for all machine capacity functions f if and only if*

$$W^{S(\pi,f)}(t) \leq cW^{S^*(f)}(t) \qquad \textit{for all } t \geq 0, \textit{ and for each } f.$$

Proof. The "if" part is clear, since by Observation 1

$$\sum_{j\in J} w_j C_j^{S(\pi,f)} = \int_0^\infty W^{S(\pi,f)}(t)dt \leq c\int_0^\infty W^{S^*(f)}(t)dt.$$

We prove the "only if" part by contradiction. Assume that $W^{S(\pi,f)}(t_0) > cW^{S^*(f)}(t_0)$ for some t_0 and f. For any $t_1 > t_0$ consider the following machine capacity function

$$f'(t) = \begin{cases} f(t) & \text{if } t \leq t_0, \\ f(t_0) & \text{if } t_0 < t \leq t_1, \\ f(t - t_1 + t_0) & \text{if } t > t_1 \end{cases}$$

which equals f up to time t_0 and then remains constant at value $f'(t) = f(t_0)$ for the time interval $[t_0, t_1]$. Hence,

$$\sum_{j \in J} w_j C_j^{S(\pi,f')} = \sum_{j \in J} w_j C_j^{S(\pi,f')} + (t_1 - t_0) W^{S(\pi,f')}(t_0). \tag{3}$$

On the other hand, let π^* be a sequence of jobs with $W^{S(\pi^*,f')}(t_0) = W^{S^*(f')}(t_0)$. Then,

$$\sum_{j \in J} w_j C_j^{S(\pi^*,f')} = \sum_{j \in J} w_j C_j^{S(\pi^*,f')} + (t_1 - t_0) W^{S^*(f')}(t_0). \tag{4}$$

As t_1 tends to infinity, the ratio of (3) and (4) tends to $W^{S(\pi,f')}(t_0)/W^{S^*(f')}(t_0) > c$, a contradiction. □

In case that all release dates are equal, approximating the sum of weighted completion times on a machine with unknown processing behavior is equivalent to approximating the total remaining weight at any point in time on an ideal machine: $f(t) = t,\ t \geq 0$. Scheduling according to sequence π on such a machine yields for each j, $C_j^\pi := \sum_{k:\pi(k) \leq \pi(j)} p_k$. The completion time under machine capacity function f is

$$C_j^{S(\pi,f)} = \min\{t \mid f(t) \geq C_j^\pi\}.$$

Observation 2. *For any machine capacity function f and any sequence π of jobs without release dates,*

$$W^{S(\pi,f)}(t) = W^\pi(f(t)) \qquad \textit{for all } t \geq 0.$$

For $f(t) = t$ let $W^*(t) := W^{S^*(f)}(t)$. With Observation 2 we can significantly strengthen the statement of Lemma 1.

Lemma 2. *Let π be a sequence of jobs with equal release dates, and let $c > 0$. Then, the objective value $\sum_{j \in J} w_j C_j^{S(\pi,f)}$ is at most c times the optimum for all machine capacity functions f if and only if*

$$W^\pi(t) \leq cW^*(t) \qquad \textit{for all } t \geq 0.$$

Simple counter examples show that this lemma is only true if all release dates are equal, otherwise, Observation 2 is simply not true.

3 Universal Scheduling without Release Dates

3.1 Upper Bounds

In the sequel we use for a subset of jobs $J' \subseteq J$ the notation $p(J') := \sum_{j \in J'} p_j$ and $w(J') := \sum_{j \in J'} w_j$. Based on key Lemma 2, we aim at approximating the minimum total weight of uncompleted jobs at any point in time on an ideal machine, i.e., we approximate the value of $W^*(t)$ for all values of $t \leq p(J)$ for a machine with capacity function $f(t) = t, t \geq 0$. In our algorithm we do so by solving the problem to find the set of jobs that has maximum total processing time and total weight within a given bound. By sequentially doubling the weight bound, a sequence of job sets is obtained. Jobs in job sets corresponding to smaller weight bounds are to come later in the schedule, breaking ties arbitrarily.

Algorithm DOUBLE:

1. For $i \in \{0, 1, \ldots, \lceil \log w(J) \rceil\}$, find a subset J_i^* of jobs of total weight $w(J_i^*) \leq 2^i$ and maximum total processing time $p(J_i^*)$. Notice that $J^*_{\lceil \log w(J) \rceil} = J$.
2. Construct a permutation π as follows. Start with an empty sequence of jobs. For $i = \lceil \log w(J) \rceil$ down to 0, append the jobs in $J_i^* \setminus \bigcup_{k=0}^{i-1} J_k^*$ in any order at the end of the sequence.

Theorem 1. *For every scheduling instance,* DOUBLE *produces a permutation π such that the objective value $\sum_{j \in J} w_j C_j^{S(\pi,f)}$ is less than 4 times the optimum for all machine capacity functions f.*

Proof. Using Lemma 2 it is sufficient to show that $W^\pi(t) < 4W^*(t)$ for all $t \geq 0$. Let $t \geq 0$ and let i be minimal such that $p(J_i^*) \geq p(J) - t$. By construction of π, only jobs j in $\bigcup_{k=0}^{i} J_k^*$ have a completion time $C_j^\pi > t$. Thus,

$$W^\pi(t) \leq \sum_{k=0}^{i} w(J_k^*) \leq \sum_{k=0}^{i} 2^k = 2^{i+1} - 1. \tag{5}$$

In case $i = 0$, the claim is trivially true since $w_j \geq 1$ for any $j \in J$, and thus, $W^*(t) = W^\pi(t)$. Suppose $i \geq 1$, then by our choice of i, it holds that $p(J_{i-1}^*) < p(J) - t$. Therefore, in any sequence π', the total weight of jobs completing after time t is larger than 2^{i-1}, because otherwise we get a contradiction to the maximality of $p(J_{i-1}^*)$. That is, $W^*(t) > 2^{i-1}$. Together with (5) this concludes the proof. □

Notice that the algorithm takes exponential time since finding the subsets of jobs J_i^* is a KNAPSACK problem and, thus, NP-hard [15]. However, we adapt the algorithm by, instead of J_i^*, computing a subset of jobs J_i of total weight $w(J_i) \leq (1 + \epsilon/4)2^i$ and processing time $p(J_i) \geq \max\{p(J') \mid J' \subseteq J \text{ and } w(J') \leq 2^i\}$. This can be done in time polynomial in the input size and $1/\epsilon$ adapting, e.g., the FPTAS in [11] for KNAPSACK. The subsets J_i obtained in this way are turned into a sequence π' as in DOUBLE.

Theorem 2. *Let $\epsilon > 0$. For every scheduling instance, we can construct a permutation π in time polynomial in the input size and $1/\epsilon$ such that the value $\sum_{j \in J} w_j C_j^{S(\pi,f)}$ is less than $4 + \epsilon$ times the optimum for all machine capacity functions f.*

Proof. Again, by Lemma 2 it is sufficient to prove that $W^{\pi}(t) < 4W^*(t)$ for all $t \geq 0$. Instead of inequality (5) we get the slightly weaker bound

$$W^{\pi'}(t) \leq \sum_{k=0}^{i} w(J_k) \leq \sum_{k=0}^{i} (1 + \epsilon/4) 2^k = (1 + \epsilon/4)(2^{i+1} - 1) < (4 + \epsilon)\, 2^{i-1}.$$

Moreover, the lower bound $W^*(t) > 2^{i-1}$ still holds. □

We improve Theorem 1 by adding randomization to DOUBLE in a quite standard fashion. Instead of the fixed bound of 2^i on the total weight of job set J_i^* in iteration $i \in \{0, 1, \ldots, \lceil \log w(J) \rceil\}$ we use the randomly chosen bound Xe^i where $X = e^Y$ and Y is picked uniformly at random from $[0, 1]$ before the first iteration. We omit the proof.

Theorem 3. *Let $\epsilon > 0$. For every scheduling instance, randomized* DOUBLE *constructs a permutation π in time that is polynomial in the input size and $1/\epsilon$ such that the objective value $\sum_{j \in J} w_j C_j^{S(\pi,f)}$ is in expectation less than $e + \epsilon$ times the optimum value for all machine capacity functions f.*

A natural generalization of the universal sequencing problem requires that jobs are sequenced in compliance with given precedence constraints. We extend the results in Theorems 1 and 3 to this model for certain classes of precedence constraints such as directed out-trees, two dimensional orders, and the complement of chordal bipartite orders.

3.2 Lower Bounds

In this section we show a connection between the performance guarantee for sequencing jobs on a single machine without release dates and an online bidding problem investigated by Chrobak et al. [5]. This allows us to prove tight lower bounds for our problem.

In online bidding, we are given a universe $\mathcal{U} = \{1, \ldots, n\}$. A bid set is just a subset of $\mathcal{U}$. A given bid set $\mathcal{B}$ is said to be α-competitive if

$$\sum_{b \in \mathcal{B} \,:\, b < T} b + \min_{b \in \mathcal{B} \,:\, b \geq T} b \leq \alpha T \qquad \forall T \in \mathcal{U}. \tag{6}$$

Chrobak et al. [5] gave lower bounds of $4-\epsilon$ and $e-\epsilon$, for any $\epsilon > 0$, for deterministic and randomized algorithms, respectively.

Theorem 4. *For any $\epsilon > 0$, there exists an instance of the universal scheduling problem without release dates on which the performance ratio of any deterministic schedule is at least $4 - \epsilon$ and the performance ratio of any randomized schedule is at least $e - \epsilon$.*

Proof. Take an instance of the online bidding problem and create the following instance of the scheduling problem: For each $j \in \mathcal{U}$ create job j with weight $w_j = j$ and processing time $p_j = j^j$. Consider any permutation π of the jobs $\mathcal{U}$. For any $j \in \mathcal{U}$, let $k(j)$ be the largest index such that $\pi_{k(j)} \geq j$. Since $p_j > \sum_{i=1}^{j-1} p_j$, at time $t = p(\mathcal{U}) - p_j$ we have $W^\pi(t) = \sum_{k=k(j)}^{n} w_{\pi_k}$, while $W^*(t) = w_j$. If sequence π yields a performance ratio of α then, Lemma 2 tell us that

$$\sum_{k=k(j)}^{n} \pi_k \leq \alpha\, j \qquad \forall\, j \in \mathcal{U}. \tag{7}$$

From sequence π we extract another sequence of jobs as follows: $\mathcal{W}_1 = \pi_n$, $\mathcal{W}_k = \operatorname{argmax}_{i \in \mathcal{U}} \left\{\pi^{-1}(i) \mid i > \mathcal{W}_{k-1}\right\}$. Then $\mathcal{W}_{i+1} > \mathcal{W}_i$, and all j with $\pi^{-1}(\mathcal{W}_{i+1}) < \pi^{-1}(j) < \pi^{-1}(\mathcal{W}_i)$ have weight less than $\mathcal{W}_i$. Therefore, we have $\{i \in \mathcal{W} \mid i < j\} \cup \min\{i \in \mathcal{W} \mid i \geq j\} \subset \{\pi_{k(j)}, \ldots, \pi_n\}$, for all $j \in \mathcal{U}$. Hence, if π achieves a performance ratio of α then

$$\sum_{i \in \mathcal{W}\,:\,i<j} i \;+ \min_{i \in \mathcal{W}\,:\,i \geq j} i \;\leq\; \sum_{k=k(j)}^{n} \pi_k \;\leq\; \alpha\, j \qquad \forall\, j \in \mathcal{U},$$

that is, the bid set $\mathcal{W}$ induced by the sequence π must be α-competitive. Since there is a lower bound of $4 - \epsilon$ for the competitiveness of deterministic strategies for online bidding, the same bound holds for the performance ratio of deterministic universal schedules.

The same approach yields the lower bound for randomized strategies. □

4 Universal Scheduling with Release Dates

In this section we study the problem of the previous section when jobs have release dates. Algorithm DOUBLE, which aims at minimizing the total remaining weight, can be adapted to the setting with release dates: Instead of a knapsack algorithm we use, within a binary search routine, Lawler's pseudo-polynomial time algorithm [18] or the FPTAS by Pruhs and Woeginger [25] for preemptively scheduling jobs with release dates and due dates on a single machine to minimize the total weight of late jobs.

However, this approach does not yield a bounded performance guarantee for the universal scheduling problem. In the presence of release dates approximation ratios on an ideal machine do not translate directly to a performance guarantee of the universal scheduling strategy, see Section 2. In fact, universal scheduling with release dates cannot be approximated within a constant ratio as we show below.

4.1 Lower Bound

Theorem 5. *There exists an instance with n jobs with equal weights and release dates, where any universal schedule has a performance guarantee of $\Omega(\log n / \log\log n)$.*

In our lower bound instance each job j has $w_j = 1, j = 0, 1, \ldots, n-1$. Their processing times form a geometric series $p_j = 2^j$, $j = 0, 1, \ldots, n-1$, and they are released in reversed order $r_j = \sum_{i>j}^{n} 2^i = \sum_{i>j} p_i$, $j = 0, 1, \ldots, n-1$.

To show the bound, we rely on a classic theorem of Erdős and Szekeres [8] or, more precisely, on Hammersley's proof [10] of this result.

Lemma 3 (Hammersley [10]). *Given a sequence of n distinct numbers $x_1, x_2, \ldots, x_n$, we can decompose this set into k increasing subsequences $\ell_1, \ell_2, \ldots, \ell_k$ such that:*

- *There is a decreasing subsequence of length k;*
- *If x_i belongs to ℓ_a then for all $j > i$ if $x_j < x_i$ then x_j belongs to ℓ_b and $b > a$.*

The idea is now to view a universal schedule as a permutation of $\{0, 1, \ldots, n-1\}$ and use Lemma 3 to decompose the sequence into k increasing subsequences. This decomposition is then used to design a breakdown pattern that will yield Theorem 5. The next two lemmas outline two kinds of breakdown patterns that apply to the two possibilities offered by Lemma 3.

Lemma 4. *The performance guarantee of a universal schedule that has ℓ as a decreasing subsequence is at least $|\ell|$.*

Proof. Let j be the first job in ℓ. The machine has breakdowns $[r_j, r_0]$ and $[r_0 + 2^j - 1, L]$ for large L. At time r_0 all jobs have been released. $2^j - 1$ time units later, at the start of the second breakdown, all jobs in ℓ belong to the set of jobs uncompleted by the universal schedule, whereas an optimal solution can complete all jobs except j. Choosing L large enough implies the lemma. □

Lemma 5. *Let $\ell_1, \ell_2, \ldots, \ell_k$ be the decomposition described in Lemma 3 when applied to a universal schedule. Then for all $i = 1, \ldots, k$ the performance guarantee is at least $\frac{|\ell_i|+|\ell_{i-1}|+\cdots+|\ell_1|}{1+|\ell_{i-1}|+\cdots+|\ell_1|}$*

Proof. For each job j in ℓ_i there is a breakdown $[r_j, r_j+\epsilon]$. For each job j in $\ell_{i+1}, \ldots, \ell_k$ there is a breakdown $[r_j, r_j + p_j] = [r_j, r_j + 2^j]$. As a consequence, at time $2^n - 1$ the universal schedule has all jobs in ℓ_i and all jobs in $\ell_{i+1}, \ldots, \ell_k$ uncompleted, whereas, a schedule exists that leaves the last job of ℓ_i and all jobs in $\ell_{j+1}, \ldots, \ell_k$ uncompleted. Therefore, a breakdown $[2^n - 1, L]$ for L large enough implies the lemma. □

Proof (Proof of Theorem 5). Consider an arbitrary universal scheduling solution and its decomposition into increasing subsequences $\ell_1, \ldots, \ell_k$ as in Lemma 3 and let α be its performance guarantee. Using Lemma 5, one can easily prove by induction that $|\ell_i| \leq \alpha^{k-i+1}$. Since $\ell_1, \ldots, \ell_k$ is a partition of the jobs, we have

$$n = \sum_{i=1}^{k} |\ell_i| \leq \sum_{i=1}^{k} \alpha^{k-i+1} \leq \alpha^{k+1}.$$

By Lemma 4, it follows that $k \leq \alpha$. Therefore $\log n = O(\alpha \log \alpha)$ and $\alpha = \Omega\left(\frac{\log n}{\log\log n}\right)$.

□

4.2 Jobs with Proportional Weights

Motivated by the negative result in the previous section, we turn our attention to the special case with proportional weights, that is, there exists a fixed $\gamma \in \mathbb{Q}$ such that $w_j = \gamma p_j$, for all $j \in J$. Using a standard scaling argument we can assume w.l.o.g. that $p_j = w_j$, for all j. We provide an algorithm with a performance guarantee 5, and prove a lower bound of 3 on the performance guarantee of any universal scheduling algorithm.

Algorithm SORTCLASS:

1. Partition the set of jobs into $z := \lceil \log \max_{j \in J} w_j \rceil$ classes, such that j belongs to class J_i, for $i \in 1, 2, \ldots, z$, if and only if $p_j \in (2^{i-1}, 2^i]$.
2. Construct a permutation π as follows. Start with an empty sequence of jobs. For $i = z$ down to 1, append the jobs of class J_i in non-decreasing order of release dates at the end of π.

Theorem 6. *The performance guarantee of* SORTCLASS *for universal scheduling of jobs with proportional weights and release dates is exactly* 5.

Proof. Let π be the job sequence computed by SORTCLASS. By Lemma 1, it is sufficient to prove

$$W^{S(\pi,f)}(t) \;\leq\; 5W^{S^*(f)}(t) \quad \forall t > 0. \tag{8}$$

Take any time t and any machine capacity function f. Let $j \in J_i$ be the job being processed at time t according to the schedule $S(\pi, f)$. We say that a job other than job j is *in the stack* at time t if it was processed for a positive amount of time before t. The algorithm needs to complete all jobs in the stack, job j, and jobs that did not start before t, which have a total weight of at most $p(J) - f(t)$, the amount of remaining processing time at time t to be done by the algorithm.

Since jobs within a class are ordered by release times, there is at most one job per class in the stack at any point in time. Since jobs in higher classes have higher priority and job $j \in J_i$ is processed at time t, there are no jobs in $J_{i+1}, \ldots, J_z$ in the stack at time t. Thus the weight of the jobs in the stack together with the weight of job j is at most $\sum_{k=1}^{i} 2^k = 2^{i+1} - 1$. Hence,

$$W^{S(\pi,f)}(t) \;< 2^{i+1} + p(J) - f(t)\,. \tag{9}$$

A first obvious lower bound on the remaining weight of any schedule at time t is

$$W^{S^*(f)}(t) \;\geq p(J) - f(t)\,. \tag{10}$$

For another lower bound, let t' be the last time before t in which the machine is available but it is either idle or a job of a class $J_{i'}$ with $i' < i$ is being processed. Note that t' is well-defined. By definition, all jobs processed during the time interval $[t', t]$ are in classes with index at least i, but also, they are released in the interval $[t', t]$ since at t' a job of a lower class was processed or the machine was idle. Since at time t at least one of these jobs is unfinished in $S(\pi, f)$, even though the machine continuously processed only those jobs, no algorithm can complete all these jobs. Thus, at time t, an optimal schedule also still needs to complete at least one job with weight at least 2^{i-1}:

$$W^{S^*(f)}(t) \geq 2^{i-1}\,. \tag{11}$$

Combining (9), (10), and (11) yields (8) and thus the upper bound of the theorem.
We omit the example that shows that the analysis is tight. $\square$

We complement this result by a lower bound of 3, but have to omit the proof.

Theorem 7. *There is no algorithm with performance guarantee strictly smaller than* 3 *for universal scheduling of jobs with release dates and $w_j = p_j$, for all $j \in J$.*

5 The Offline Problem

Clearly, the performance guarantees derived in Sections 3 and 4 also hold in the offline version of our problem in which machine breakdowns and changes in speed are known in advance. Additionally, we investigate in this section the special case in which the machine has a single, a priori-known non-availability interval $[s, t]$, for $1 \leq s < t$.

Theorem 8. *There exists an FPTAS with running time $\mathcal{O}(n^3/\epsilon^2)$ for non-preemptive scheduling to minimize $\sum w_j C_j$ on a single machine that is not available for processing during a given time interval $[s, t]$. The approximation scheme can be extended to the preemptive (resumable) setting with an increased running time of $\mathcal{O}(n^4/\epsilon^2 \log p_{\max})$.*

Due to space limitations we defer all details to the full version of the paper. The idea for our FPTAS is based on a natural non-preemptive dynamic programming algorithm, used also in [16]. Given a non-available time interval $[s, t]$, the job set must be partitioned into jobs that complete before s and jobs that complete after t. Clearly, the jobs in each individual set are scheduled in non-increasing order of ratios w_j/p_j. This order is known to be optimal on an ideal machine [29].

The main challenge in designing the FPTAS is to discretize the range of possible total processing times of jobs scheduled before s in an appropriate way. Notice that we cannot afford to round these values since they contain critical information on how much processing time remains before the break. Perturbing this information causes a considerable change in the set of feasible schedules that cannot be controlled easily. The intuition behind our algorithm is to reduce the number of states by removing those with the same (rounded) objective value and nearly the same total processing time before the break. Among them, we want to store those with smallest amount of processing before the break in order to make sure that enough space remains for further jobs that need to be scheduled there.

The algorithm can be extended easily to the preemptive (resumable) problem. We can assume, w.l.o.g., that in an optimal solution there is at most one job j interrupted by the break $[s, t]$ and it resumes processing as soon as the machine is available again. For a given job j and with start time S_j, we define a non-preemptive problem with non-available period $[S_j, S_j + p_j + t - s]$, which we solve by the FPTAS above. Thus, we can solve the preemptive problem by running the FPTAS above $\mathcal{O}(n \log p_{\max})$ times.

6 Further Remarks

In Section 4 we have shown that the performance of universal scheduling algorithms may deteriorate drastically when generalizing the universal scheduling problem slightly. Other generalizations do not admit any (exponential time) algorithm with bounded performance guarantee. If a non-adaptive algorithm cannot guarantee to finish within the minimum makespan, then an adversary creates an arbitrarily long breakdown at the moment that an optimal schedule has completed all jobs. Examples of such variations are the problem with two or more machines instead of a single machine, or the problem in which preempting or resuming a job requires (even the slightest amount of) extra work.

The offline version of our problem (without release dates) in which preemption is not allowed or causes extra work is not approximable in polynomial time; a reduction from 2-PARTITION shows that the problem with two or more non-available periods is not approximable, unless P=NP, even if all jobs have equal weight. A reduction in that spirit has been used in [33] for a scheduling problem with some jobs having a fixed position in the schedule. Similarly, we can rule out constant approximation factors for any preemptive problem variant in which the makespan cannot be computed exactly in polynomial time. This is shown by simple reductions from the corresponding decision version of the makespan minimization problem. Such variations of our problem are scheduling with release dates and scheduling with general precedence constraints.

References

1. Adiri, I., Bruno, J., Frostig, E., Rinnooy Kan, A.: Single machine flow-time scheduling with a single breakdown. Acta Informatica 26(7), 679–696 (1989)
2. Becchetti, L., Leonardi, S., Marchetti-Spaccamela, A., Pruhs, K.: Online weighted flow time and deadline scheduling. In: Goemans, M.X., Jansen, K., Rolim, J.D.P., Trevisan, L. (eds.) RANDOM 2001 and APPROX 2001. LNCS, vol. 2129, pp. 36–47. Springer, Heidelberg (2001)
3. Ben-Tal, A., Nemirovski, A.: Robust solutions of linear programming problems contaminated with uncertain data. Mathematical Programming 88, 411–424 (2000)
4. Carter, J., Wegman, M.: Universal classes of hash functions. Journal of Computer and System Sciences 18, 143–154 (1979)
5. Chrobak, M., Kenyon, C., Noga, J., Young, N.E.: Incremental medians via online bidding. Algorithmica 50(4), 455–478 (2008)
6. Chrobak, M., Kenyon-Mathieu, C.: Sigact news online algorithms column 10: Competitiveness via doubling. SIGACT News 37(4), 115–126 (2006)
7. Diedrich, F., Jansen, K., Schwarz, U.M., Trystram, D.: A survey on approximation algorithms for scheduling with machine unavailability. In: Lerner, J., Wagner, D., Zweig, K.A. (eds.) Algorithmics of Large and Complex Networks. LNCS, vol. 5515, pp. 50–64. Springer, Heidelberg (2009)
8. Erdős, P., Szekeres, G.: A combinatorial problem in geometry. Compositio Mathematica 2, 463–470 (1935)
9. Hall, L., Schulz, A.S., Shmoys, D., Wein, J.: Scheduling to minimize average completion time: off-line and on-line approximation algorithms. Mathematics of Operations Research 22, 513–544 (1997)
10. Hammersley, J.: A few seedlings of research. In: Proceedings Sixth Berkeley Symp. Math. Statist. and Probability, vol. 1, pp. 345–394. University of California Press, Berkeley (1972)
11. Ibarra, O.H., Kim, C.E.: Fast approximation algorithms for the knapsack and sum of subset problems. Journal of the ACM 22(4), 463–468 (1975)

12. Jia, L., Lin, G., Noubir, G., Rajaraman, R., Sundaram, R.: Universal approximations for TSP, Steiner tree, and set cover. In: Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC '05), pp. 386–395 (2005)
13. Kacem, I.: Approximation algorithm for the weighted flow-time minimization on a single machine with a fixed non-availability interval. Computers & Industrial Engineering 54(3), 401–410 (2008)
14. Kacem, I., Mahjoub, A.R.: Fully polynomial time approximation scheme for the weighted flow-time minimization on a single machine with a fixed non-availability interval. Computers & Industrial Engineering 56(4), 1708–1712 (2009)
15. Karp, R.M.: Reducibility among combinatorial problems. In: Complexity of computer computations (Proc. Sympos., IBM Thomas J. Watson Res. Center), pp. 85–103. Plenum, New York (1972)
16. Kellerer, H., Strusevich, V.A.: Fully polynomial approximation schemes for a symmetric quadratic knapsack problem and its scheduling applications (2009) (accepted for publication in Algorithmica)
17. Kouvelis, P., Yu, G.: Robust Discrete Optimization and Its Applications. Springer, Heidelberg (1997)
18. Lawler, E.L.: A dynamic programming algorithm for preemptive scheduling of a single machine to minimize the number of late jobs. Annals of Operations Research 26, 125–133 (1990)
19. Lee, C.Y.: Machine scheduling with an availability constraint. Journal of Global Optimimization 9, 395–416 (1996)
20. Lee, C.Y.: Machine scheduling with availability constraints. In: Leung, J.Y.T. (ed.) Handbook of scheduling. CRC Press, Boca Raton (2004)
21. Lee, C.Y., Liman, S.D.: Single machine flow-time scheduling with scheduled maintenance. Acta Informatica 29(4), 375–382 (1992)
22. Liebchen, C., Lübbecke, M., Möhring, R.H., Stiller, S.: Recoverable robustness. Technical report ARRIVAL-TR-0066, ARRIVAL Project (2007)
23. Mastrolilli, M., Mutsanas, N., Svensson, O.: Approximating single machine scheduling with scenarios. In: Goel, A., Jansen, K., Rolim, J.D.P., Rubinfeld, R. (eds.) APPROX and RANDOM 2008. LNCS, vol. 5171, pp. 153–164. Springer, Heidelberg (2008)
24. Megow, N., Verschae, J.: Note on scheduling on a single machine with one non-availability period (2008) (unpublished)
25. Pruhs, K., Woeginger, G.J.: Approximation schemes for a class of subset selection problems. Theoretical Computer Science 382(2), 151–156 (2007)
26. Räcke, H.: Survey on oblivious routing strategies. In: Ambos-Spies, K., Löwe, B., Merkle, W. (eds.) Mathematical Theory and Computational Practice, Proceedings of 5th Conference on Computability in Europe (CiE). LNCS, vol. 5635, pp. 419–429. Springer, Heidelberg (2009)
27. Schmidt, G.: Scheduling with limited machine availability. European Journal of Operational Research 121(1), 1–15 (2000)
28. Schulz, A.S., Skutella, M.: The power of α-points in preemptive single machine scheduling. Journal of Scheduling 5, 121–133 (2002)
29. Smith, W.E.: Various optimizers for single-stage production. Naval Research Logistics Quarterly 3, 59–66 (1956)
30. Soyster, A.: Convex programming with set-inclusive constraints and applications to inexact linear programming. Operations Research 21(4), 1154–1157 (1973)
31. Valiant, L.G., Brebner, G.J.: Universal schemes for parallel communication. In: Proc. of STOC, pp. 263–277 (1981)
32. Wang, G., Sun, H., Chu, C.: Preemptive scheduling with availability constraints to minimize total weighted completion times. Annals of Operations Research 133, 183–192 (2005)
33. Yuan, J., Lin, Y., Ng, C., Cheng, T.: Approximability of single machine scheduling with fixed jobs to minimize total completion time. European Journal of Operational Research 178(1), 46–56 (2007)

Fault-Tolerant Facility Location: A Randomized Dependent LP-Rounding Algorithm[*]

Jaroslaw Byrka[1,**], Aravind Srinivasan[2], and Chaitanya Swamy[3]

[1] Institute of Mathematics, Ecole Polytechnique Federale de Lausanne, CH-1015 Lausanne, Switzerland
jaroslaw.byrka@epfl.ch

[2] Dept. of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, USA
srin@cs.umd.edu

[3] Dept. of Combinatorics & Optimization, Faculty of Mathematics, University of Waterloo, Waterloo, ON N2L 3G1, Canada
cswamy@math.uwaterloo.ca

Abstract. We give a new randomized LP-rounding 1.725-approximation algorithm for the metric Fault-Tolerant Uncapacitated Facility Location problem. This improves on the previously best known 2.076-approximation algorithm of Swamy & Shmoys. To the best of our knowledge, our work provides the first application of a dependent-rounding technique in the domain of facility location. The analysis of our algorithm benefits from, and extends, methods developed for Uncapacitated Facility Location; it also helps uncover new properties of the dependent-rounding approach.

An important concept that we develop is a novel, hierarchical clustering scheme. Typically, LP-rounding approximation algorithms for facility location problems are based on partitioning facilities into disjoint clusters and opening at least one facility in each cluster. We extend this approach and construct a laminar family of clusters, which then guides the rounding procedure: this allows us to exploit properties of dependent rounding, and provides a quite tight analysis resulting in the improved approximation ratio.

1 Introduction

In Facility Location problems we are given a set of clients $\mathcal{C}$ that require a certain service. To provide such a service, we need to open a subset of a given set of

[*] This work was partially supported by: (i) the Future and Emerging Technologies Unit of EC (IST priority - 6th FP), under contract no. FP6-021235-2 (project ARRIVAL), (ii) MNISW grant number N N206 1723 33, 2007-2010, (iii) NSF ITR Award CNS-0426683 and NSF Award CNS-0626636, and (iv) NSERC grant 327620-09 and an Ontario Early Researcher Award.

[**] Work of this author was partially conducted at CWI Amsterdam, TU Eindhoven, and while visiting the University of Maryland.

facilities $\mathcal{F}$. Opening each facility $i \in \mathcal{F}$ costs f_i and serving a client j by facility i costs c_{ij}; the standard assumption is that the c_{ij} are symmetric and constitute a metric. (The non-metric case is much harder to approximate.) In this paper, we follow Swamy & Shmoys [11] and study the Fault-Tolerant Facility Location (FTFL) problem, where each client has a positive integer specified as its *coverage requirement* r_j. The task is to find a minimum-cost solution which opens some facilities from $\mathcal{F}$ and connects each client j to r_j different open facilities.

The FTFL problem was introduced by Jain & Vazirani [7]. Guha et al. [6] gave the first constant factor approximation algorithm with approximation ratio 2.408. This was later improved by Swamy & Shmoys [11] who gave a 2.076-approximation algorithm. FTFL generalizes the standard Uncapacitated Facility Location (UFL) problem wherein $r_j = 1$ for all j, for which Guha & Khuller [5] proved an approximation lower bound of ≈ 1.463. The current-best approximation ratio for UFL is achieved by the 1.5-approximation algorithm of Byrka [2].

In this paper we give a new LP-rounding 1.7245-approximation algorithm for the FTFL problem. It is the first application of the dependent rounding technique of [10] to a facility location problem.

Our algorithm uses a novel clustering method, which allows clusters not to be disjoint, but rather to form a laminar family of subsets of facilities. The hierarchical structure of the obtained clustering exploits properties of dependent rounding. By first rounding the "facility-opening" variables within smaller clusters, we are able to ensure that at least a certain number of facilities is open in each of the clusters. Intuitively, by allowing clusters to have different sizes we may, in a more efficient manner, guarantee the opening of sufficiently-many facilities around clients with different coverage requirements r_j. In addition, one of our main technical contributions is Theorem 2, which develops a new property of the dependent-rounding technique that appears likely to have further applications. Basically, suppose we apply dependent rounding to a sequence of reals and consider an arbitrary subset S of the rounded variables (each of which lies in $\{0, 1\}$) as well as an arbitrary integer $k > 0$. Then, a natural fault-tolerance-related objective is that if X denotes the number of variables rounded to 1 in S, then the random variable $Z = \min\{k, X\}$ be "large". (In other words, we want X to be "large", but X being more than k does not add any marginal utility.) We prove that if X_0 denotes the corresponding sum wherein the reals are rounded *independently* and if $Z_0 = \min\{k, X_0\}$, then $\mathbf{E}[Z] \geq \mathbf{E}[Z_0]$. Thus, for analysis purposes, we can work with Z_0, which is much more tractable due to the independence; at the same time, we derive all the benefits of dependent rounding (such as a given number of facilities becoming available in a cluster, with probability one). Given the growing number of applications of dependent-rounding methodologies, we view this as a useful addition to the toolkit.

2 Dependent Rounding

Given a fractional vector $y = (y_1, y_2, \ldots, y_N) \in [0, 1]^N$ we often seek to round it to an integral vector $\hat{y} \in \{0, 1\}^N$ that is in a problem-specific sense very "close to" y. The dependent-randomized-rounding technique of [10] is one such approach known for preserving the sum of the entries deterministically, along with concentration bounds for any linear combination of the entries; we will generalize a known property of this technique in order to apply it to the FTFL problem. The very useful *pipage rounding* technique of [1] was developed prior to [10], and can be viewed as a derandomization (deterministic analog) of [10] via the method of conditional probabilities. Indeed, the results of [1] were applied in the work of [11]; the probabilistic intuition, as well as our generalization of the analysis of [10], help obtain our results.

Define $[t] = \{1, 2, \ldots, t\}$. Given a fractional vector $y = (y_1, y_2, \ldots, y_N) \in [0, 1]^N$, the rounding technique of [10] (henceforth just referred to as "dependent rounding") is a polynomial-time randomized algorithm to produce a random vector $\hat{y} \in \{0, 1\}^N$ with the following three properties:

(P1): marginals. $\forall i,\ \Pr[\hat{y}_i = 1] = y_i$;
(P2): sum-preservation. With probability one, $\sum_{i=1}^N \hat{y}_i$ equals either $\lfloor \sum_{i=1}^N y_i \rfloor$ or $\lceil \sum_{i=1}^N y_i \rceil$; and
(P3): negative correlation. $\forall S \subseteq [N]$, $\Pr[\bigwedge_{i\in S}(\hat{y}_i = 0)] \leq \prod_{i\in S}(1 - y_i)$, and $\Pr[\bigwedge_{i\in S}(\hat{y}_i = 1)] \leq \prod_{i\in S} y_i$.

In this paper, we also exploit the order in which the entries of the given fractional vector y are rounded. We initially define a laminar family of subsets of indices $\mathcal{S} \subseteq 2^{[N]}$. When applying the dependent rounding procedure, we first round within the smaller sets, until at most one fractional entry in a set is left, then we proceed with bigger sets possibly containing the already rounded entries. It can easily be shown that it assures the following version of property (P2) for all subsets S from the laminar family $\mathcal{S}$:

(P2'): sum-preservation. With probability one, $\sum_{i\in S} \hat{y}_i = \sum_{i\in S} y_i$ and $|\{i \in S : \hat{y}_i = 1\}| = \lfloor \sum_{i\in S} y_i \rfloor$.

Now, let $S \subseteq [N]$ be any subset, not necessarily from $\mathcal{S}$. In order to present our results, we need two functions, Sum_S and $g_{\lambda,S}$. For any vector $x \in [0, 1]^n$, let $\mathrm{Sum}_S(x) = \sum_{i\in S} x_i$ be the sum of the elements of x indexed by elements of S; in particular, if x is a (possibly random) vector with all entries either 0 or 1, then $\mathrm{Sum}_S(x)$ counts the number of entries in S that are 1. Next, given $s = |S|$ and a real vector $\lambda = (\lambda_0, \lambda_1, \lambda_2, \ldots, \lambda_s)$, we define, for any $x \in \{0, 1\}^n$,

$$g_{\lambda,S}(x) = \sum_{i=0}^{s} \lambda_i \cdot \mathcal{I}(\mathrm{Sum}_S(x) = i),$$

where $\mathcal{I}(\cdot)$ denotes the indicator function. Thus, $g_{\lambda,S}(x) = \lambda_i$ if $\mathrm{Sum}_S(x) = i$.

Let $\mathcal{R}(y)$ be a random vector in $\{0, 1\}^N$ obtained by *independently* rounding each y_i to 1 with probability y_i, and to 0 with the complementary probability of $1 - y_i$. Suppose, as above, that $\hat{y}$ is a random vector in $\{0, 1\}^N$ obtained by applying the dependent rounding technique to y. We start with a general theorem and then specialize it to Theorem 2 that will be very useful for us:

Theorem 1. *Suppose we conduct dependent rounding on $y = (y_1, y_2, \ldots, y_N)$. Let $S \subseteq [N]$ be any subset with cardinality $s \geq 2$, and let $\lambda = (\lambda_0, \lambda_1, \lambda_2, \ldots, \lambda_s)$ be any vector, such that for all r with $0 \leq r \leq s-2$ we have $\lambda_r - 2\lambda_{r+1} + \lambda_{r+2} \leq 0$. Then, $\mathbf{E}[g_{\lambda,S}(\hat{y})] \geq \mathbf{E}[g_{\lambda,S}(\mathcal{R}(y))]$.*

Theorem 2. *For any $y \in [0,1]^N$, $S \subseteq [N]$, and $k = 1, 2, \ldots$, we have*

$$\mathbf{E}[\min\{k, Sum_S(\hat{y})\}] \geq \mathbf{E}[\min\{k, Sum_S(\mathcal{R}(y))\}].$$

Using the notation $\exp(t) = e^t$, our next key result is:

Theorem 3. *For any $y \in [0,1]^N$, $S \subseteq [N]$, and $k = 1, 2, \ldots$, we have*

$$\mathbf{E}[\min\{k, Sum_S(\mathcal{R}(y))\}] \geq k \cdot (1 - \exp(-Sum_S(y)/k)).$$

The above two theorems yield a key corollary that we will use:
Corollary 1

$$\mathbf{E}[\min\{k, Sum_S(\hat{y})\}] \geq k \cdot (1 - \exp(-Sum_S(y)/k)).$$

Proofs will appear in the full version of the paper (see also [3]).

3 Algorithm

3.1 LP-Relaxation

The FTFL problem is defined by the following Integer Program (IP).

$$\begin{array}{llll}
\text{minimize } \sum_{i\in\mathcal{F}} f_i y_i + \sum_{j\in\mathcal{C}} \sum_{i\in\mathcal{F}} c_{ij} x_{ij} & & & (1) \\
\text{subject to:} & \sum_i x_{ij} \geq r_j & \forall j \in \mathcal{C} & (2) \\
& x_{ij} \leq y_i & \forall j \in \mathcal{C}\ \forall i \in \mathcal{F} & (3) \\
& y_i \leq 1 & \forall i \in \mathcal{F} & (4) \\
& x_{ij}, y_i \in Z_{\geq 0} & \forall j \in \mathcal{C}\ \forall i \in \mathcal{F}, & (5)
\end{array}$$

where $\mathcal{C}$ is the set of clients, $\mathcal{F}$ is the set of possible locations of facilities, f_i is a cost of opening a facility at location i, c_{ij} is a cost of serving client j from a facility at location i, and r_j is the amount of facilities client j needs to be connected to.

If we relax constraint (5) to $x_{ij}, y_i \geq 0$ we obtain the standard LP-relaxation of the problem. Let (x^*, y^*) be an optimal solution to this LP relaxation. We will give an algorithm that rounds this solution to an integral solution $(\tilde{x}, \tilde{y})$ with cost at most $\gamma \approx 1.7245$ times the cost of (x^*, y^*).

3.2 Scaling

We may assume, without loss of generality, that for any client $j \in \mathcal{C}$ there exists at most one facility $i \in \mathcal{F}$ such that $0 < x_{ij} < y_i$. Moreover, this facility can be assumed to have the highest distance to client j among the facilities that fractionally serve j in (x^*, y^*).

We first set $\tilde{x}_{ij} = \tilde{y}_i = 0$ for all $i \in \mathcal{F}$, $j \in \mathcal{C}$. Then we scale up the fractional solution by the constant $\gamma \approx 1.7245$ to obtain a fractional solution $(\hat{x}, \hat{y})$. To be precise: we set $\hat{x}_{ij} = \min\{1, \gamma \cdot x^*_{ij}\}$, $\hat{y}_i = \min\{1, \gamma \cdot y^*_i\}$. We open each facility i with $\hat{y}_i = 1$ and connect each client-facility pair with $\hat{x}_{ij} = 1$. To be more precise, we modify $\hat{y}$, $\tilde{y}$, $\hat{x}$, $\tilde{x}$ and service requirements r as follows. For each facility i with $\hat{y}_i = 1$, set $\hat{y}_i = 0$ and $\tilde{y}_i = 1$. Then, for every pair (i, j) such that $\hat{x}_{ij} = 1$, set $\hat{x}_{ij} = 0$, $\tilde{x}_{ij} = 1$ and decrease r_j by one. When this process is finished we call the resulting r, $\hat{y}$ and $\hat{x}$ by $\overline{r}$, $\overline{y}$ and $\overline{x}$. Note that the connections that we made in this phase can be paid for by a difference in the connection cost between $\hat{x}$ and $\overline{x}$. We will show that the remaining connection cost of the solution of the algorithm is expected to be at most the cost of $\overline{x}$.

For the feasibility of the final solution, it is essential that if we connected client j to facility i in this initial phase, we will not connect it again to i in the rest of the algorithm. There will be two ways of connecting clients in the process of rounding $\overline{x}$. The first one connects client j to a subset of facilities serving j in $\overline{x}$. Recall that if j was connected to facility i in the initial phase, then $\overline{x}_{ij} = 0$, and no additional i-j connection will be created.

The connections of the second type will be created in a process of *clustering*. The clustering that we will use is a generalization of the one of Chudak & Shmoys for the UFL problem [4]. As a result of this clustering process, client j will be allowed to connect itself via a different client j' to a facility open around j'. j' will be called a *cluster center* for a subset of facilities, and it will make sure that at least some guaranteed number of these facilities will get opened.

To be certain that client j does not get again connected to facility i with a path via client j', facility i will never be a member of the set of facilities clustered by client j'. We call a facility i *special* for client j iff $\tilde{y}_i = 1$ and $0 < \overline{x}_{ij} < 1$. Note that, by our earlier assumption, there is at most one special facility for each client j, and that a special facility must be at maximal distance among facilities serving j in $\overline{x}$. When rounding the fractional solution in Section 3.5, we take care that special facilities are not members of the formed clusters.

3.3 Close and Distant Facilities

Before we describe how do we cluster facilities, we specify the facilities that are interesting for a particular client in the clustering process. The following can be fought of as a version of a *filtering* technique of Lin and Vitter [8], first applied to facility location by Shmoys et al. [9]. The analysis that we use here is a version of the argument of Byrka [2].

As a result of the scaling that was described in the previous section, the connection variables $\overline{x}$ amount for a total connectivity that exceeds the requirement $\overline{r}$. More precisely, we have $\sum_{i \in \mathcal{F}} \overline{x}_{ij} \geq \gamma \cdot \overline{r}_j$ for every client $j \in \mathcal{C}$. We will consider for each client j a subset of facilities that are just enough to provide it a fractional connection of $\overline{r}_j$. Such a subset is called a set of *close facilities* of client j and is defined as follows.

For every client j consider the following construction. Let $i_1, i_2, \ldots, i_{|\mathcal{F}|}$ be the ordering of facilities in $\mathcal{F}$ in a nondecreasing order of distances c_{ij} to client j. Let i_k be the facility in this ordering, such that $\sum_{l=1}^{k-1} \overline{x}_{i_l j} < \overline{r}_j$ and $\sum_{l=1}^{k} \overline{x}_{i_l j} \geq \overline{r}_j$. Define

$$\overline{x}^{(c)}_{i_l j} = \begin{cases} \overline{x}_{i_l j} & \text{for } l < k, \\ \overline{r}_j - \sum_{l=1}^{k-1} \overline{x}_{i_l j} & \text{for } l = k, \\ 0 & \text{for } l > k \end{cases}$$

Define $\overline{x}^{(d)}_{ij} = \overline{x}_{ij} - \overline{x}^{(c)}_{ij}$ for all $i \in \mathcal{F}, j \in \mathcal{C}$.

We will call the set of facilities $i \in \mathcal{F}$ such that $\overline{x}^{(c)}_{ij} > 0$ the set of *close facilities* of client j and we denote it by C_j. By analogy, we will call the set of facilities $i \in \mathcal{F}$ such that $\overline{x}^{(d)}_{ij} > 0$ the set of *distant facilities* of client j and denote it D_j. Observe that for a client j the intersection of C_j and D_j is either empty, or contains exactly one facility. In the latter case, we will say that this facility is both distant and close. Note that, unlike in the UFL problem, we cannot simply split this facility to the close and the distant part, because it is essential that we make at most one connection to this facility in the final integral solution. Let $d^{(max)}_j = c_{i_k j}$ be the distance from client j to the farthest of its close facilities.

3.4 Clustering

We will now construct a family of subsets of facilities $\mathcal{S} \in 2^{\mathcal{F}}$. These subsets $S \in \mathcal{S}$ will be called clusters and they will guide the rounding procedure described next. There will be a client related to each cluster, and each single client j will be related to at most one cluster, which we call S_j.

Not all the clients participate in the clustering process. Clients j with $\overline{r}_j = 1$ and a special facility $i' \in C_j$ (recall that a special facility is a facility that is fully open in $\hat{y}$ but only partially used by j in $\overline{x}$) will be called special and will not take part in the clustering process. Let $\mathcal{C}'$ denote the set of all other, non-special clients. Observe that, as a result of scaling, clients j with $\overline{r}_j \geq 2$ do not have any special facilities among their close facilities (since $\sum_i \overline{x}_{ij} \geq \gamma \overline{r}_j > \overline{r}_j + 1$). As a consequence, there are no special facilities among the close facilities of clients from $\mathcal{C}'$, the only clients actively involved in the clustering procedure.

For each client $j \in \mathcal{C}'$ we will keep two families A_j and B_j of disjoint subsets of facilities. Initially $A_j = \{\{i\} : i \in C_j\}$, i.e., A_j is initialized to contain a singleton set for each close facility of client j; B_j is initially empty. A_j will be used to store these initial singleton sets, but also clusters containing only close facilities of j; B_j will be used to store only clusters that contain at least one close facility of j. When adding a cluster to either A_j or B_j we will remove all the subsets it intersects from both A_j and B_j, therefore subsets in $A_j \cup B_j$ will always be pairwise disjoint.

The family of clusters that we will construct will be a laminar family of subsets of facilities, i.e., any two clusters are either disjoint or one entirely contains the other. One can imagine facilities being leaves and clusters being internal nodes of a forest that eventually becomes a tree, when all the clusters are added.

We will use $\overline{y}(S)$ as a shorthand for $\sum_{i\in S}\overline{y}_i$. Let us define $\underline{y}(S)=\lfloor\overline{y}(S)\rfloor$. As a consequence of using the family of clusters to guide the rounding process, by Property (P2') of the dependent rounding procedure when applied to a cluster, th quantity $\underline{y}(S)$ lower bounds the number of facilities that will certainly be opened in cluster S. Additionally, let us define the residual requirement of client j to be $rr_j=\overline{r}_j-\sum_{S\in(A_j\cup B_j)}\underline{y}(S)$, that is $\overline{r}_j$ minus a lower bound on the number of facilities that will be opened in clusters from A_j and B_j.

We use the following procedure to compute clusters. While there exists a client $j\in\mathcal{C}'$, such that $rr_j>0$, take such j with minimal $d_j^{(max)}$ and do the following:

1. Take X_j to be an inclusion-wise minimal subset of A_j, such that $\sum_{S\in X_j}(\overline{y}(S)-\underline{y}(S))\geq rr_j$. Form the new cluster $S_j=\bigcup_{S\in X_j}S$.
2. Make S_j a new cluster by setting $\mathcal{S}\leftarrow\mathcal{S}\cup\{S_j\}$.
3. Update $A_j\leftarrow(A_j\setminus X_j)\cup\{S_j\}$.
4. For each client j' with $rr_{j'}>0$ do
   - If $X_j\subseteq A_{j'}$, then set $A_{j'}\leftarrow(A_{j'}\setminus X_j)\cup\{S_j\}$.
   - If $X_j\cap A_{j'}\neq\emptyset$ and $X_j\setminus A_{j'}\neq\emptyset$,
     then set $A_{j'}\leftarrow A_{j'}\setminus X_j$ and $B_{j'}\leftarrow\{S\in B_{j'}:S\cap S_j=\emptyset\}\cup\{S_j\}$.

Eventually, add a cluster $S_r=\mathcal{F}$ containing all the facilities to the family $\mathcal{S}$.

We call a client j' active in a particular iteration, if before this iteration its residual requirement $rr_j=\overline{r}_j-\sum_{S\in(A_j\cup B_j)}\underline{y}(S)$ was positive. During the above procedure, all active clients j have in their sets A_j and B_j only maximal subsets of facilities, that means they are not subsets of any other clusters (i.e., they are roots of their trees in the current forest). Therefore, when a new cluster S_j is created, it contains all the other clusters with which it has nonempty intersections (i.e., the new cluster S_j becomes a root of a new tree).

We shall now argue that there is enough fractional opening in clusters in A_j to cover the residual requirement rr_j when cluster S_j is to be formed. Consider a fixed client $j\in\mathcal{C}'$. Recall that at the start of the clustering we have $A_j=\{\{i\}:i\in C_j\}$, and therefore $\sum_{S\in A_j}(\overline{y}(S)-\underline{y}(S))=\sum_{i\in C_j}\overline{y}_i\geq\overline{r}_j=rr_j$. It remains to show, that $\sum_{S\in A_j}(\overline{y}(S)-\underline{y}(S))-rr_j$ does not decrease over time until client j is considered. When a client j' with $d_{j'}^{(max)}\leq d_j^{(max)}$ is considered and cluster $S_{j'}$ is created, the following cases are possible:

1. $S_{j'}\cap(\bigcup_{S\in A_j}S)=\emptyset$: then A_j and rr_j do not change;
2. $S_{j'}\subseteq(\bigcup_{S\in A_j}S)$: then A_j changes its structure, but $\sum_{S\in A_j}\overline{y}(S)$ and $\sum_{S\in B_j}\underline{y}(S)$ do not change; hence $\sum_{S\in A_j}(\overline{y}(S)-\underline{y}(S))-rr_j$ also does not change;
3. $S_{j'}\cap(\bigcup_{S\in A_j}S)\neq\emptyset$ and $S_{j'}\setminus(\bigcup_{S\in A_j}S)\neq\emptyset$: then, by inclusion-wise minimality of set $X_{j'}$, we have $\underline{y}(S_{j'})-\sum_{S\in B_j,S\subseteq S_{j'}}\underline{y}(S)-\sum_{S\in A_j,S\subseteq S_{j'}}\overline{y}(S)\geq 0$; hence, $\sum_{S\in A_j}(\overline{y}(S)-\underline{y}(S))-rr_j$ cannot decrease.

Let $A'_j=A_j\cap\mathcal{S}$ be the set of clusters in A_j. Recall that all facilities in clusters in A'_j are close facilities of j. Note also that each cluster $S_{j'}\in B_j$ was created from close facilities of a client j' with $d_{j'}^{(max)}\leq d_j^{(max)}$. We also have for each

$S_{j'} \in B_j$ that $S_{j'} \cap C_j \neq \emptyset$, hence, by the triangle inequality, all facilities in $S_{j'}$ are at distance at most $3 \cdot d_j^{(max)}$ from j. We thus infer the following

Corollary 2. *The family of clusters $\mathcal{S}$ contains for each client $j \in \mathcal{C}'$ a collection of disjoint clusters $A'_j \cup B_j$ containing only facilities within distance $3 \cdot d_j^{(max)}$, and $\sum_{S \in A'_j \cup B_j} \lfloor \sum_{i \in S} \overline{y}_i \rfloor \geq \overline{r}_j$.*

Note that our clustering is related to, but more complex then the one of Chudak and Shmoys [4] for UFL and of Swamy and Shmoys [11] for FTFL, where clusters are pairwise disjoint and each contains facilities whose fractional opening sums up to or slightly exceeds the value of 1.

3.5 Opening of Facilities by Dependent Rounding

Given the family of subsets $\mathcal{S} \in 2^{\mathcal{F}}$ computed by the clustering procedure from Section 3.4, we can proceed with rounding the fractional opening vector $\overline{y}$ into an integral vector y^R. We do it by applying the rounding technique of Section 2, guided by the family $\mathcal{S}$, which is done as follows.

While there is more than one fractional entry, select a minimal subset of $S \in \mathcal{S}$ which contains more than one fractional entry and apply the rounding procedure to entries of $\overline{y}$ indexed by elements of S until at most one entry in S remains fractional. Eventually, if there remains a fractional entry, round it independently and let y^R be the resulting vector.

Observe that the above process is one of the possible implementations of dependent rounding applied to $\overline{y}$. As a result, the random integral vector y^R satisfies properties (P1),(P2), and (P3). Additionally, property (P2') holds for each cluster $S \in \mathcal{S}$. Hence, at least $\lfloor \sum_{i \in S} \overline{y}_i \rfloor$ entries in each $S \in \mathcal{S}$ are rounded to 1. Therefore, by Corollary 2, we get

Corollary 3. *For each client $j \in \mathcal{C}'$.*

$$|\{i \in \mathcal{F} | y_i^R = 1 \text{ and } c_{ij} \leq 3 \cdot d_j^{(max)}\}| \geq \overline{r}_j.$$

Next, we combine the facilities opened by rounding y^R with facilities opened already when scaling which are recorded in $\tilde{y}$, i.e., we update $\tilde{y} \leftarrow \tilde{y} + y^R$.

Eventually, we connect each client $j \in \mathcal{C}$ to r_j closest opened facilities and code it in $\tilde{x}$.

4 Analysis

We will now estimate the expected cost of the solution $(\tilde{x}, \tilde{y})$. The tricky part is to bound the connection cost, which we do as follows. We argue that a certain fraction of the demand of client j can be satisfied from its close facilities, then some part of the remaining demand can be satisfied from its distant facilities. Eventually, the remaining (not too large in expectation) part of the demand is satisfied via clusters.

4.1 Average Distances

Let us consider weighted average distances from a client j to sets of facilities fractionally serving it. Let d_j be the average connection cost in $\overline{x}_{ij}$ defined as

$$d_j = \frac{\sum_{i \in \mathcal{F}} c_{ij} \cdot \overline{x}_{ij}}{\sum_{i \in \mathcal{F}} \overline{x}_{ij}}.$$

Let $d_j^{(c)}$, $d_j^{(d)}$ be the average connection costs in $\overline{x}_{ij}^{(c)}$ and $\overline{x}_{ij}^{(d)}$ defined as

$$d_j^{(c)} = \frac{\sum_{i \in \mathcal{F}} c_{ij} \cdot \overline{x}_{ij}^{(c)}}{\sum_{i \in \mathcal{F}} \overline{x}_{ij}^{(c)}},$$

$$d_j^{(d)} = \frac{\sum_{i \in \mathcal{F}} c_{ij} \cdot \overline{x}_{ij}^{(d)}}{\sum_{i \in \mathcal{F}} \overline{x}_{ij}^{(d)}}.$$

Let R_j be a parameter defined as

$$R_j = \frac{d_j - d_j^{(c)}}{d_j}$$

if $d_j > 0$ and $R_j = 0$ otherwise. Observe that R_j takes value between 0 and 1. $R_j = 0$ implies $d_j^{(c)} = d_j = d_j^{(d)}$, and $R_j = 1$ occurs only when $d_j^{(c)} = 0$. The role played by R_j is that it measures a certain parameter of the instance, big values are good for one part of the analysis, small values are good for the other.

Lemma 1. $d_j^{(d)} \leq d_j(1 + \frac{R_j}{\gamma - 1})$.

Proof. Recall that $\sum_{i \in \mathcal{F}} \overline{x}_{ij}^{(c)} = \overline{r}_j$ and $\sum_{i \in \mathcal{F}} \overline{x}_{ij}^{(d)} \geq (\gamma - 1) \cdot \overline{r}_j$. Therefore, we have $(d_j^{(d)} - d_j) \cdot (\gamma - 1) \leq (d_j - d_j^{(c)}) \cdot 1 = R_j \cdot d_j$, which can be rearranged to get $d_j^{(d)} \leq d_j(1 + \frac{R_j}{\gamma - 1})$. □

Finally, observe that the average distance from j to the distant facilities of j gives an upper bound on the maximal distance to any of the close facilities of j. Namely, $d_j^{(max)} \leq d_j^{(d)}$.

4.2 Amount of Service from Close and Distant Facilities

We now argue that in the solution $(\tilde{x}, \tilde{y})$, a certain portion of the demand is expected to be served by the close and distant facilities of each client. Recall that for a client j it is possible, that there is a facility that is both its close and its distant facility. Once we have a solution that opens such a facility, we would like to say what fraction of the demand is served from the close facilities. To make our analysis simpler we will toss a properly biased coin to decide if using this facility counts as using a close facility. With this trick we, in a sense,

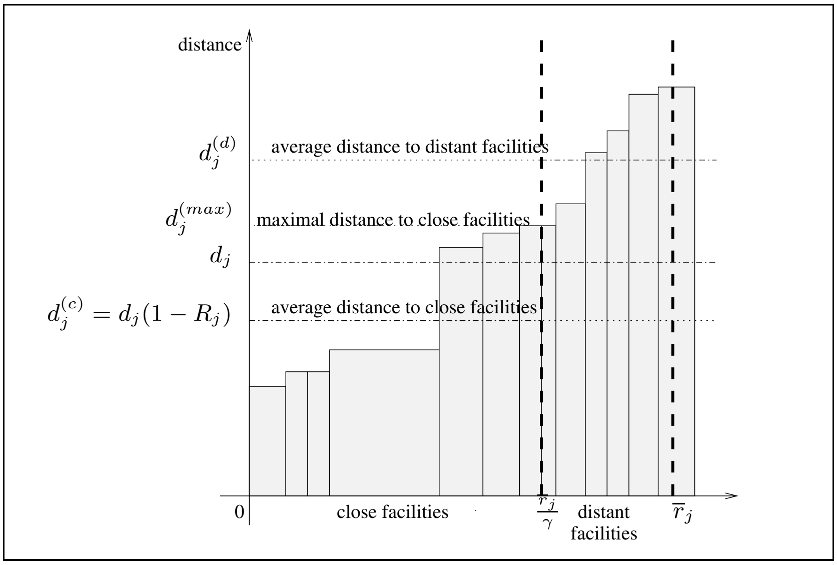


Fig. 1. Distances to facilities serving client j in $\overline{x}$. The width of a rectangle corresponding to facility i is equal to $\overline{x}_{ij}$. Figure helps to understand the meaning of R_j.

split such a facility into a close and a distant part. Note that we can only do it for this part of the analysis, but not for the actual rounding algorithm from Section 3.5. Applying the above-described split of the undecided facility, we get that the total fractional opening of close facilities of client j is exactly $\overline{r}_j$, and the total fractional opening of both close and distant facilities is at least $\gamma \cdot \overline{r}_j$. Therefore, Corollary 1 yields the following:

Corollary 4. *The amount of close facilities used by client j in a solution described in Section 3.5 is expected to be at least* $(1 - \frac{1}{e}) \cdot \overline{r}_j$.

Corollary 5. *The amount of close and distant facilities used by client j in a solution described in Section 3.5 is expected to be at least* $(1 - \frac{1}{e^\gamma}) \cdot \overline{r}_j$.

Motivated by the above bounds we design a selection method to choose a (large-enough in expectation) subset of facilities opened around client j:

Lemma 2. *For $j \in \mathcal{C}'$ we can select a subset F_j of open facilities from $C_j \cup D_j$ such that:*

$$|F_j| \leq \overline{r}_j \text{ (with probability 1)},$$

$$E[F_j] = (1 - \frac{1}{e^\gamma}) \cdot \overline{r}_j,$$

$$E[\sum_{i \in F_j} c_{ij}] \leq ((1 - 1/e) \cdot \overline{r}_j) \cdot d_j^{(c)} + (((1 - \frac{1}{e^\gamma}) - (1 - 1/e)) \cdot \overline{r}_j) \cdot d_j^{(d)}.$$

A technical but not difficult proof we sketch in the Appendix.

4.3 Calculation

We can now combine the pieces into the algorithm ALG:

1. solve the LP-relaxation of (1)-(5);
2. scale the fractional solution as described in Section 3.2;
3. create a family of clusters as described in Section 3.4;
4. round the fractional openings as described in Section 3.5;
5. connect each client j to r_j closest open facilities;
6. output the solution as $(\tilde{x}, \tilde{y})$.

Theorem 4. *ALG is an* 1.7245*-approximation algorithm for FTFL.*

Proof. First observe that the solution produced by ALG is trivially feasible to the original problem (1)-(5), as we simply choose different r_j facilities for client j in step 5. What is less trivial is that all the r_j facilities used by j are within a certain small distance. Let us now bound the expected connection cost of the obtained solution.

For each client $j \in \mathcal{C}$ we get $r_j - \overline{r}_j$ facilities opened in Step 2. As we already argued in Section 3.2, we can afford to connect j to these facilities and pay the connection cost from the difference between $\sum_i c_{ij}\hat{x}_{ij}$ and $\sum_i c_{ij}\overline{x}_{ij}$. We will now argue, that client j can connect to the remaining $\overline{r}_j$ with the expected connection cost bounded by $\sum_i c_{ij}\overline{x}_{ij}$.

For a special client $j \in (\mathcal{C} \setminus \mathcal{C}')$ we have $\overline{r}_j = 1$ and already in Step 2 one special facility at distance $d_j^{(max)}$ from j is opened. We cannot always just connect j to this facility, since $d_j^{(max)}$ may potentially be bigger then $\gamma \cdot d_j$. What we do instead is that we first look at close facilities of j that, as a result of the rounding in Step 4, with a certain probability, give one open facility at a small distance. By Corollary 4 this probability is at least $1 - 1/e$. It is easy to observe that the expected connection cost to this open facility is at most $d_j^{(c)}$. Only if no close facility is open, we use the special facility, which results in the expected connection cost of client j being at most

$$(1-1/e)d_j^{(c)} + (1/e)d_j^{(d)} \leq (1-1/e)d_j^{(c)} + (1/e)d_j(1+\frac{R_j}{\gamma-1}) \leq d_j(1+1/(e\cdot(\gamma-1)) \leq \gamma \cdot d_j,$$

where the first inequality is a consequence of Lemma 1, and the last one is a consequence of the choice of $\gamma \approx 1.7245$.

In the remaining, we only look at non-special clients $j \in \mathcal{C}'$. By Lemma 2, client j can select to connect itself to the subset of open facilities F_j, and pay for this connection at most $((1-1/e)\cdot\overline{r}_j)\cdot d_j^{(c)} + (((1-\frac{1}{e^\gamma}) - (1-1/e))\cdot\overline{r}_j)\cdot d_j^{(d)}$ in expectation. The expected number of facilities needed on top of those from F_j is $\overline{r}_j - E[|F_j|] = (\frac{1}{e^\gamma}\cdot\overline{r}_j)$. These remaining facilities client j gets deterministically within the distance of at most $3 \cdot d_j^{(max)}$, which is possible by the properties of the rounding procedure described in Section 3.5, see Corollary 3. Therefore, the expected connection cost to facilities not in F_j is at most $(\frac{1}{e^\gamma}\cdot\overline{r}_j)\cdot(3\cdot d_j^{(max)})$.

Concluding, the total expected connection cost of j may be bounded by

$$
\begin{aligned}
&((1-1/e)\cdot\overline{r}_j)\cdot d_j^{(c)} + (((1-\frac{1}{e^\gamma})-(1-1/e))\cdot\overline{r}_j)\cdot d_j^{(d)} + (\frac{1}{e^\gamma}\cdot\overline{r}_j)\cdot(3\cdot d_j^{(max)})\\
\leq\ & \overline{r}_j\cdot\left((1-1/e)\cdot d_j^{(c)} + ((1-\frac{1}{e^\gamma})-(1-1/e))\cdot d_j^{(d)} + \frac{1}{e^\gamma}\cdot(3d_j^{(d)})\right)\\
=\ & \overline{r}_j\cdot\left((1-1/e)\cdot d_j^{(c)} + ((1+\frac{2}{e^\gamma})-(1-1/e))\cdot d_j^{(d)}\right)\\
\leq\ & \overline{r}_j\cdot\left((1-1/e)\cdot(1-R_j)\cdot d_j + ((1+\frac{2}{e^\gamma})-(1-1/e))\cdot(1+\frac{R_j}{\gamma-1})\cdot d_j\right)\\
=\ & \overline{r}_j\cdot d_j\cdot\left((1-1/e)\cdot(1-R_j) + (\frac{2}{e^\gamma}+1/e)\cdot(1+\frac{R_j}{\gamma-1})\right)\\
=\ & \overline{r}_j\cdot d_j\cdot\left((1-1/e) + (\frac{2}{e^\gamma}+1/e) + R_j\cdot((\frac{2}{e^\gamma}+1/e)\cdot\frac{1}{\gamma-1}-(1-1/e))\right)\\
=\ & \overline{r}_j\cdot d_j\cdot\left(1+\frac{2}{e^\gamma}+R_j\cdot\left(\frac{(\frac{2}{e^\gamma}+1/e)}{\gamma-1}-(1-1/e)\right)\right),
\end{aligned}
$$

where the second inequality follows from Lemma 1 and the definition of R_j.

Observe that for $1<\gamma<2$, we have $\frac{(\frac{2}{e^\gamma}+1/e)}{\gamma-1}-(1-1/e)>0$. Recall that by definition, $R_j\leq 1$; so, $R_j=1$ is the worst case for our estimate, and therefore

$$\overline{r}_j\cdot d_j\cdot\left(1+\frac{2}{e^\gamma}+R_j\cdot\left(\frac{(\frac{2}{e^\gamma}+1/e)}{\gamma-1}-(1-1/e)\right)\right)\leq\overline{r}_j\cdot d_j\cdot(1/e+\frac{2}{e^\gamma})(1+\frac{1}{\gamma-1}).$$

Recall that $\overline{x}$ incurs, for each client j, a fractional connection cost $\sum_{i\in\mathcal{F}}c_{ij}\overline{x}_{ij}\geq\gamma\cdot\overline{r}_j\cdot d_j$. We fix $\gamma=\gamma_0$, such that $\gamma_0=(1/e+\frac{2}{e^{\gamma_0}})(1+\frac{1}{\gamma_0-1})\leq 1.7245$.

To conclude, the expected connection cost of j to facilities opened during the rounding procedure is at most the fractional connection cost of $\overline{x}$. The total connection cost is, therefore, at most the connection cost of $\hat{x}$, which is at most γ times the connection cost of x^*.

By property (P1) of dependent rounding, every single facility i is opened with the probability $\hat{y}_i$, which is at most γ times y_i^*. Therefore, the total expected cost of the solution produced by ALG is at most $\gamma\approx 1.7245$ times the cost of the fractional optimal solution (x^*,y^*). □

5 Concluding Remarks

We have presented improved approximation algorithms for the metric Fault-Tolerant Uncapacitated Facility Location problem. The main technical innovation is the usage and analysis of dependent rounding in this context. We believe that variants of dependent rounding will also be fruitful in other location problems. Finally, we conjecture that the approximation threshold for both UFL and FTFL is the value $1.46\cdots$ suggested by [5]; it would be very interesting to prove or refute this.

Acknowledgment. We thank the IPCO referees for their helpful comments.

References

1. Ageev, A., Sviridenko, M.: Pipage rounding: a new method of constructing algorithms with proven performance guarantee. Journal of Combinatorial Optimization 8(3), 307–328 (2004)
2. Byrka, J.: An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem. In: APPROX-RANDOM, pp. 29–43 (2007)
3. Byrka, J., Srinivasan, A., Swamy, C.: Fault-tolerant facility location: a randomized dependent LP-rounding algorithm. arXiv:1003.1295v1
4. Chudak, F.A., Shmoys, D.B.: Improved approximation algorithms for the uncapacitated facility location problem. SIAM J. Comput. 33(1), 1–25 (2003)
5. Guha, S., Khuller, S.: Greedy strikes back: Improved facility location algorithms. J. Algorithms 31(1), 228–248 (1999)
6. Guha, S., Meyerson, A., Munagala, K.: A constant factor approximation algorithm for the fault-tolerant facility location problem. J. Algorithms 48(2), 429–440 (2003)
7. Jain, K., Vazirani, V.V.: An approximation algorithm for the fault tolerant metric facility location problem. Algorithmica 38(3), 433–439 (2003)
8. Lin, J.-H., Vitter, J.S.: Epsilon-approximations with minimum packing constraint violation (extended abstract). In: STOC, pp. 771–782 (1992)
9. Shmoys, D.B., Tardos, É., Aardal, K.: Approximation algorithms for facility location problems (extended abstract). In: STOC, pp. 265–274 (1997)
10. Srinivasan, A.: Distributions on level-sets with applications to approximation algorithms. In: FOCS, pp. 588–597 (2001)
11. Swamy, C., Shmoys, D.B.: Fault-tolerant facility location. ACM Transactions on Algorithms 4(4) (2008)

Appendix

Proof. **(for Lemma 2, a sketch)** Given client j, fractional facility opening vector $\overline{y}$, distances c_{ij}, requirement $\overline{r}_j$, and facility subsets C_j and D_j, we will describe how to randomly choose a subset of at most $k = \overline{r}_j$ open facilities from $C_j \cup D_j$ with the desired properties.

For this argument we assume that all the numbers are rational. Recall that the opening of facilities is decided in a dependent rounding routine, that in a single step couples two fractional entries to leave at most one of them fractional.

Observe that, for the purpose of this argument, we can split a single facility into many identical copies with smaller fractional opening. One can think that the input facilities and their original openings were obtained along the process of dependent rounding applied to the multiple "small" copies that we prefer to consider here. Therefore, without loss of generality, we can assume that all the facilities have fractional opening equal ϵ, i.e., $\overline{y}_i = \epsilon$ for all $i \in C_j \cup D_j$. Moreover, we can assume that sets C_j and D_j are disjoint.

By renaming facilities we obtain that $C_j = \{1, 2, \dots, |C_j|\}$, $D_j = \{|C_j| + 1, \dots, |C_j| + |D_j|\}$, and $c_{ij} \leq c_{i'j}$ for all $1 \leq i < i' \leq |C_j| + |D_j|$.

Consider random set $S_0 \subseteq C_j \cup D_j$ created as follows. Let $\hat{y}$ be the outcome of rounding the fractional opening vector $\overline{y}$ with the dependent rounding procedure, and define $S_0 = \{i : \hat{y}_i = 1, (\sum_{j<i} \hat{y}) < k\}$. By Corollary 1, we have that

$\mathbf{E}[|S_0|] \geq k\cdot(1-\exp(-\mathrm{Sum}_{C_j\cup D_j}(\overline{y})/k))$. Define random set S_α for $\alpha \in (0, |C_j|+|D_j|]$ as follows. For $i = 1, 2, \dots \lfloor |C_j| + |D_j| - \alpha \rfloor$ we have $i \in S_\alpha$ if and only if $i \in S_0$. For $i = \lceil |C_j| + |D_j| - \alpha \rceil$, in case $i \in S_0$ we toss a (suitably biased) coin and include i in S_α with probability $\alpha - \lfloor \alpha \rfloor$. For $i > \lceil |C_j| + |D_j| - \alpha \rceil$ we deterministically have $i \notin S_\alpha$.

Observe that $\mathbf{E}[|S_\alpha|]$ is a continuous monotone non-increasing function of α, hence there is α_0 such that $\mathbf{E}[|S_{\alpha_0}|] = k\cdot(1-\exp(-\mathrm{Sum}_{C_j\cup D_j}(\overline{y})/k))$. We fix $F_j = S_{\alpha_0}$ and claim that it has the desired properties. By definition, we have $\mathbf{E}[|F_j|] = k\cdot(1-\exp(-\mathrm{Sum}_{C_j\cup D_j}(\overline{y})/k)) = (1-\frac{1}{e^\gamma})\cdot\overline{r}_j$. We next show that the expected total connection cost between j and facilities in F_j is not too large.

Let $p_i^\alpha = Pr[i \in S_\alpha]$ and $p'_i = p_i^{\alpha_0} = Pr[i \in F_j]$. Consider the cumulative probability defined as $cp_i^\alpha = \sum_{j\leq i} p_j^\alpha$. Observe that application of Corollary 1 to subsets of first i elements of $C_j \cup D_j$ yields $cp_i^0 \geq k\cdot(1-\exp(-\epsilon i/k))$ for $i = 1, \dots, |C_j| + |D_j|$. Since $(1-\exp(-\epsilon i/k))$ is a monotone increasing function of i one easily gets that also $cp_i^\alpha \geq k\cdot(1-\exp(-\epsilon i/k))$ for $\alpha \leq \alpha_0$ and $i = 1, \dots, |C_j| + |D_j|$. In particular, we get $cp_{|C_j|}^{\alpha_0} \geq k\cdot(1-\exp(-\epsilon|C_j|/k))$.

Since $(1-\exp(-\epsilon i/k))$ is a concave function of i, we also have $cp_i^{\alpha_0} \geq k\cdot(1-\exp(-\epsilon i/k)) \geq (i/|C_j|)\cdot k\cdot(1-\exp(-\epsilon|C_j|/k)) = (i/|C_j|)\cdot(1-\frac{1}{e})\cdot\overline{r}_j$ for all $1 \leq i \leq |C_j|$. Analogously, we get

$$\begin{aligned} cp_i^{\alpha_0} \geq{} & (k\cdot(1-\exp(-\epsilon|C_j|/k))) \\ & +((i-|C_j|)/|D_j|)\cdot k\cdot\left((1-\exp(\frac{-\epsilon(|C_j|+|D_j|)}{k})) - (1-\exp(-\epsilon|C_j|/k))\right) \\ ={} & \overline{r}_j\cdot(1-\frac{1}{e}) + \overline{r}_j\cdot\left(((i-|C_j|)/|D_j|)((1-\frac{1}{e^\gamma})-(1-\frac{1}{e}))\right) \end{aligned}$$

for all $|C_j| < i \leq |C_j| + |D_j|$.

Recall that we want to bound $\mathbf{E}[\sum_{i\in F_j} c_{ij}] = \sum_{i\in C_j\cup D_j} p'_i c_{ij}$. From the above bounds on the cumulative probability, we get that, by shifting the probability from earlier facilities to later ones, one can obtain a probability vector p'' with $p''_i = 1/|C_j|\cdot((1-\frac{1}{e})\cdot\overline{r}_j)$ for all $1 \leq i \leq |C_j|$, and $p''_i = 1/|D_j|\cdot((1-\frac{1}{e^\gamma})-(1-\frac{1}{e}))\cdot\overline{r}_j$ for all $|C_j| < i \leq |C_j|+|D_j|$. As connection costs c_{ij} are monotone non-decreasing in i, shifting the probability never decreases the weighted sum; hence,

$$\begin{aligned} \mathbf{E}[\sum_{i\in F_j} c_{ij}] &= \sum_{i\in F_j} p'_i c_{ij} \\ &\leq \sum_{i\in F_j} p''_i c_{ij} \\ &= \sum_{1\leq i\leq|C_j|} 1/|C_j|\cdot((1-\frac{1}{e})\cdot\overline{r}_j)c_{ij} \\ &\quad + \sum_{|C_j|<i\leq|C_j|+|D_j|} 1/|D_j|\cdot(((1-\frac{1}{e^\gamma})-(1-\frac{1}{e}))\cdot\overline{r}_j)c_{ij} \\ &= ((1-1/e)\cdot\overline{r}_j)\cdot d_j^{(c)} + (((1-\frac{1}{e^\gamma})-(1-1/e))\cdot\overline{r}_j)\cdot d_j^{(d)}. \qquad \square \end{aligned}$$

Integer Quadratic Quasi-polyhedra

Adam N. Letchford

Department of Management Science, Lancaster University,
Lancaster LA1 4YW, United Kingdom
A.N.Letchford@lancaster.ac.uk

Abstract. This paper introduces two fundamental families of 'quasi-polyhedra' — polyhedra with a countably infinite number of facets — that arise in the context of integer quadratic programming. It is shown that any integer quadratic program can be reduced to the minimisation of a linear function over a quasi-polyhedron in the first family. Some fundamental properties of the quasi-polyhedra are derived, along with connections to some other well-studied convex sets. Several classes of facet-inducing inequalities are also derived. Finally, extensions to the mixed-integer case are briefly examined.

Keywords: mixed-integer non-linear programming, polyhedral combinatorics, convex analysis.

1 Introduction

In recent years, there has been increasing interest in *Mixed-Integer Non-Linear Programming* (MINLP), due to the realisation that it has a wealth of applications. This paper is concerned with a special case of MINLP: *Integer Quadratic Programming* (IQP). It is assumed that instances of IQP are written in the following standard form:

$$\min \left\{ c^T x + x^T Q x : \ Ax = b, \ x \in \mathbb{Z}^n_+ \right\} \quad , \tag{1}$$

where $c \in \mathbb{Q}^n$, $Q \in \mathbb{Q}^{n \times n}$, $A \in \mathbb{Q}^{m \times n}$ and $b \in \mathbb{Q}^m$. (As in linear programming, inequalities can be converted into equations using slack variables, and free variables can be expressed as the difference between two non-negative variables.)

We assume (without loss of generality) that the matrix Q is symmetric, but we do not require it to be positive semidefinite. That is, we do not assume that the objective function is convex.

Polyhedral combinatorics — the study of polyhedra associated with combinatorial problems — has proven to be a very useful tool for deriving strong formulations of *Mixed-Integer Linear Programs* (e.g., [1,16]). The purpose of this paper is to apply it to IQP. It turns out, however, that one has to deal with 'quasi-polyhedra': convex sets that are the intersection of a countably infinite number of half-spaces. For this reason, polyhedral theory has to be combined with elements of convex analysis. (A similar strategy was used in [5] to study a continuous quadratic optimisation problem.)

The paper is structured as follows. In Sect. 2, two families of quasi-polyhedra are defined, and it is shown that any IQP instance can be reduced to the problem of optimising a linear function over a quasi-polyhedron in the first family. In Sect. 3, some fundamental properties of the quasi-polyhedra are derived, such as dimension, extreme points, affine symmetries, and relationships with some other well-studied convex sets. In Sect. 4, we derive several classes of valid inequalities, all of which are proven to induce facets under mild conditions. Finally, in Sect. 5, we suggest possible extensions to the mixed-integer case, and pose some questions for future research.

2 The Quasi-polyhedra

A standard trick when dealing with quadratic optimisation problems is to linearise the objective and/or constraints by introducing additional variables (e.g., [14,17,18]). More precisely, for $1 \leq i \leq j \leq n$, we define a new variable y_{ij}, which represents the product $x_i x_j$. The IQP (1) can then be reformulated as:

$$\min \left\{ c^T x + q^T y : \; Ax = b, \; x \in \mathbb{Z}_+^n, \; y_{ij} = x_i x_j \; (1 \leq i \leq j \leq n) \right\} \quad ,$$

where $q \in \mathbb{Q}^{\binom{n+1}{2}}$ is defined appropriately. Notice that the non-linearity (and non-convexity, if any) is now captured in the constraints $y_{ij} = x_i x_j$.

It is an interesting fact that the linear equations can be eliminated from the problem. Indeed, we can delete an arbitrary linear equation $a^T x = r$, provided that we add $M(a^T x - r)^2$ to the objective function, where M is a suitable large integer. For this reason, one can concentrate on the unconstrained case, in which the linear system $Ax = b$ is vacuous.

The set of feasible solutions to an unconstrained IQP, in the extended (x, y)-space, is:

$$F_n^+ := \left\{ (x, y) \in \mathbb{Z}_+^{n+\binom{n+1}{2}}, \; y_{ij} = x_i x_j \; (1 \leq i \leq j \leq n) \right\} \quad .$$

We wish to apply a polyhedral approach to IQP, and are therefore interested in the convex hull of this set. Unfortunately, there are two minor technical issues to address.

The first technical issue is that the convex hull of F_n^+ is not closed, as expressed in the following proposition:

Proposition 1. *The convex hull of F_n^+ is not closed for any n.*

Proof. For any $t \in \mathbb{Z}_+$, let (x^t, y^t) be the member of F_n^+ that arises when $x_1 = t$, $y_{11} = t^2$, and all other variables are equal to zero. Moreover, let

$$(\tilde{x}^t, \tilde{x}^t) = \frac{1}{t^2}(x^t, y^t) + \frac{t^2 - 1}{t^2}(x^0, y^0) \quad .$$

Note that $(\tilde{x}^t, \tilde{x}^t)$ is a convex combination of members of F_n^+ and therefore lies in the convex hull. Note also that $(\tilde{x}^t, \tilde{x}^t)$ is obtained by setting $x_1 = 1/t$, $y_{11} = 1$, and all other variables to zero. On the other hand, the point with $y_{11} = 1$ and all other variables at zero does not lie in the convex hull of F_n. Since the convex hull does not contain all of its limit points, it is not closed. □

We are therefore led to look at the closure of the convex hull, which we denote by IQ_n^+. Figure 1 represents IQ_1^+. It can be seen that it is described by the non-negativity inequality $x_1 \geq 0$, together with the inequalities $y_{11} \geq (2t + 1)x_1 - t(t+1)$ for all $t \in \mathbb{Z}_+$. (A similar observation was made by Michaels & Weismantel [11] for a closely-related family of polytopes.)

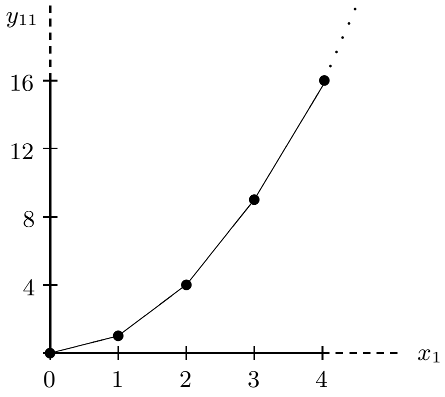


Fig. 1. The convex set IQ_1^+

The second technical issue is that IQ_n^+ is, in fact, not a polyhedron. A polyhedron is defined as the intersection of a *finite* number of half-spaces, but we have seen that IQ_1^+ is the intersection of a *countably infinite* number of half-spaces. (The results that we give in Sect. 4 show that the same holds when $n > 1$ as well.) The correct term for such sets is *quasi-polyhedra* (see, e.g., Anderson *et al.* [2]). Fortunately, this issue does not cause any difficulty in what follows.

For the purposes of what follows, we introduce a closely-related family of quasi-polyhedra, obtained by omitting the non-negativity requirement. Specifically, we define

$$F_n := \left\{ (x,y) \in \mathbb{Z}^{n+\binom{n+1}{2}},\ y_{ij} = x_i x_j \ (1 \leq i \leq j \leq n) \right\} \ ,$$

and then let IQ_n denote the closure of the convex hull of F_n. One can check that IQ_1 is described by the inequalities $y_{11} \geq (2t+1)x_1 - t(t+1)$ for all $t \in \mathbb{Z}$.

Next, we present two simple complexity results:

Proposition 2. *Minimising a linear function over* IQ_n^+ *is* $\mathcal{NP}$*-hard in the strong sense.*

Proof. It follows from the above discussion that this problem is equivalent to IQP. Now, IQP is clearly $\mathcal{NP}$-hard in the strong sense, since it contains Integer Linear Programming as a special case. □

Proposition 3. *Minimising a linear function over* IQ_n *is* $\mathcal{NP}$*-hard in the strong sense.*

Proof. The well-known *Closest Vector Problem*, proven to be strongly $\mathcal{NP}$-hard by van Emde Boas [8], takes the form:

$$\min \left\{ \|Bx - t\|_2 : \, x \in \mathbb{Z}^n \right\},$$

where $B \in \mathbb{Z}^{n \times n}$ is a basis matrix and $t \in \mathbb{Q}^n$ is a target point. Clearly, squaring the objective function leaves the optimal solution unchanged. The resulting problem is one of minimising a quadratic function over the integer lattice $\mathbb{Z}^n$. It follows from the definitions that this is equivalent to minimising a linear function over IQ_n. □

We therefore cannot expect to obtain complete linear descriptions of IQ_n or IQ_n^+ for general n.

On a more positive note, we have the following result:

Proposition 4. *Minimising a linear function over* IQ_n *is solvable in polynomial time when n is fixed.*

Proof. As already mentioned, minimising a linear function over IQ_n is equivalent to minimising a quadratic function over the integer lattice $\mathbb{Z}^n$. Now, if the function is not convex, the problem is easily shown to be unbounded. If, on the other hand, the function is convex, then the problem can be solved for fixed n by the algorithm of Khachiyan & Porkolab [9]. □

There is therefore some hope of obtaining a complete linear description of IQ_n for small values of n (just as we have already done for the case $n = 1$). We do not know the complexity of minimising a linear function over IQ_n^+ for fixed n.

3 Fundamental Properties of the Quasi-polyhedra

In this section, we establish some fundamental properties of the quasi-polyhedra IQ_n and IQ_n^+.

3.1 Dimension and Extreme Points

We begin with two elementary results:

Proposition 5. *For all n, both* IQ_n *and* IQ_n^+ *are full-dimensional, i.e., of dimension* $n + \binom{n+1}{2}$.

Proof. Consider the following extreme points of IQ_n^+:

– the origin (i.e., all variables set to zero);
– for $i = 1, \ldots, n$, the point having $x_i = y_{ii} = 1$ and all other variables zero;
– for $i = 1, \ldots, n$, the point having $x_i = 2$, $y_{ii} = 4$ and all other variables zero;
– for $1 \leq i < j \leq n$, the point having $x_i = x_j = 1$, $y_{ii} = y_{jj} = y_{ij} = 1$, and all other variables zero.

These $n + \binom{n+1}{2} + 1$ points are easily shown to be affinely independent, and therefore IQ_n^+ is full-dimensional. Since IQ_n^+ is contained in IQ_n, the same is true for IQ_n. □

Proposition 6. *Every point in F_n is an extreme point of IQ_n, and every point in F_n^+ is an extreme point of IQ_n^+.*

Proof. Let $\bar{x}$ be an arbitrary point in $\mathbb{Z}^n$, and let $(\bar{x}, \bar{y})$ be the corresponding member of F_n. The quadratic function $\sum_{i=1}^n (x_i - \bar{x}_i)^2$ has a unique minimum at $x = \bar{x}$. Since every point in F_n satisfies $y_{ij} = x_i x_j$ for all $1 \le i \le j \le n$, the *linear* function $\sum_{i=1}^n (y_{ii} - 2\bar{x}_i x_i + \bar{x}_i^2)$ has a unique minimum at $(\bar{x}, \bar{y})$. Therefore $(\bar{x}, \bar{y})$ is an extreme point of IQ_n. The proof for IQ_n^+ is similar. □

3.2 Affine Symmetries

Now we examine the affine symmetries of the quasi-polyhedra, i.e., affine transformations that map the quasi-polyhedra onto themselves.

Proposition 7. *Let π be an arbitrary permutation of the index set $\{1, \dots, n\}$. Consider the linear transformation that takes any $(x, y) \in \mathbb{R}^{n+\binom{n}{2}}$ and maps it to a point $(x', y') \in \mathbb{R}^{n+\binom{n}{2}}$, where*

- *$x'_i = x_{\pi(i)}$ for all $i \in \{1, \dots, n\}$,*
- *$y'_{ij} = y_{\pi(i),\pi(j)}$ for all $1 \le i \le j \le n$.*

This transformation maps IQ_n^+ onto itself.

Proof. Trivial. □

Theorem 1. *Let U be a unimodular integral square matrix of order n, and let $w \in \mathbb{Z}^n$ be an arbitrary integer vector. Consider the affine transformation that takes any $(x, y) \in \mathbb{R}^{n+\binom{n}{2}}$ and maps it to a point $(x', y') \in \mathbb{R}^{n+\binom{n}{2}}$, where*

- *$x' = Ux + w$;*
- *$y'_{ij} = x'_i x'_j$ for all $1 \le i \le j \le n$.*

This transformation maps IQ_n onto itself.

Proof. Let (x, y) be an extreme point of IQ_n, and let (x', y') be its image under the transformation. Since U and w are integral, x' is integral. Moreover, since $y'_{ij} = x'_i x'_j$, (x', y') is an extreme point of IQ_n. For the reverse direction, let (x', y') be an extreme point of IQ_n, and let (x, y) be its image under the inverse transformation. Note that $x = U^{-1}(x' - w)$, and is therefore integral. Moreover, $y_{ij} = x_i x_j$ for all $1 \le i \le j \le n$, which implies that (x, y) is an extreme point of IQ_n. □

Proposition 7 simply states that IQ_n^+ is invariant under a permutation of the index set $\{1, \dots, n\}$, which is unsurprising. Theorem 1, on the other hand, has a very useful corollary:

Corollary 1. *Let U be a unimodular integral square matrix of order n, and let $w \in \mathbb{Z}^n$ be an arbitrary integer vector. If the linear inequality $\alpha^T x + \beta^T y \geq \gamma$ is facet-inducing for IQ_n, then so is the linear inequality $\alpha^T x' + \beta^T y' \geq \gamma$, where (x', y') is defined as in Theorem 1.*

Intuitively speaking, this means that any inequality inducing a facet of IQ_n can be 'rotated' and 'translated' to yield a countably infinite family of facet-inducing inequalities.

It is also possible to convert any facet-inducing inequality for IQ_n into a facet-inducing inequality for IQ_n^+:

Theorem 2. *Suppose the inequality $\alpha^T x + \beta^T y \geq \gamma$ induces a facet of IQ_n. Then there exists a vector $v \in \mathbb{Z}_+^n$ such that the inequality $\alpha^T x' + \beta^T y' \geq \gamma$ induces a facet of IQ_n^+, where:*

– $x' = x - v$;
– $y'_{ij} = x'_i x'_j$ *for all* $1 \leq i \leq j \leq n$.

Proof. Let $d = n + \binom{n+1}{2}$. Since the original inequality $\alpha^T x + \beta^T y \geq \gamma$ induces a facet of IQ_n, there exist d affinely-independent members of F_n that satisfy it at equality. Let $x^1, \ldots, x^d$ denote the corresponding x vectors. Now, for $i = 1, \ldots, n$, set v_i to $\min_{1 \leq j \leq d} x_i^j$. The resulting transformed inequality $\alpha^T x' + \beta^T y' \geq \gamma$ induces a facet of IQ_n by Theorem 1. It is also valid for IQ_n^+, since IQ_n^+ is contained in IQ_n. Moreover, the points $x^1 - v, \ldots, x^d - v$ all lie in the non-negative orthant by construction. These points correspond to affinely-independent members of F_n^+ that satisfy the transformed inequality at equality. Therefore the transformed inequality induces a facet of IQ_n^+. □

Therefore, any inequality inducing a facet of IQ_n yields a countably infinite family of facet-inducing inequalities for IQ_n^+ as well.

3.3 Two Related Cones

Recall that a symmetric matrix $M \in \mathbb{R}^{n \times n}$ is called *positive semidefinite* (psd) if it can be factorised as AA^T for some real matrix A. The set of psd matrices of order n forms a convex cone in $\mathbb{R}^{n \times n}$. It is well known that this cone is completely described by the linear inequalities $v^T M v \geq 0$ for all vectors $v \in \mathbb{R}^n$.

We now use a standard construction [10,17] to establish a connection between IQ_n and the psd cone. Define the $n \times n$ symmetric matrix $Y = xx^T$, and note that, for any $1 \leq i \leq j \leq n$, $Y_{ij} = y_{ij}$. Define also the augmented matrix

$$\hat{Y} := \begin{pmatrix} 1 \\ x \end{pmatrix} \begin{pmatrix} 1 \\ x \end{pmatrix}^T = \begin{pmatrix} 1 & x^T \\ x & Y \end{pmatrix} .$$

Since $\hat{Y}$ is the product of a vector and its transpose, it must be psd. Equivalently, $v^T Y v + (2s) v^T x + s^2 \geq 0$ for all vectors $v \in \mathbb{R}^n$ and scalars $s \in \mathbb{R}$. This observation immediately yields the following result:

Proposition 8. *The following 'psd inequalities' are valid for* IQ_n *(and therefore also for* IQ_n^+*):*

$$(2s)v^T x + \sum_{i=1}^{n} v_i^2 y_{ii} + 2 \sum_{1 \le i < j \le n} v_i v_j y_{ij} + s^2 \ge 0 \qquad (\forall v \in \mathbb{R}^n, s \in \mathbb{R}) \ . \quad (2)$$

To the knowledge of the author, the validity of the psd inequalities for extended formulations of quadratic optimisation problems was first observed by Ramana [15]. The inequalities can be shown to induce proper faces of IQ_n and IQ_n^+ under mild conditions. We will see in the next section, however, that they never induce facets.

Now recall that a symmetric matrix $M \in \mathbb{R}^{n \times n}$ is called *completely positive* if it can be factorised as AA^T for some *non-negative* real matrix A. The set of completely positive matrices of order n also forms a convex cone in $\mathbb{R}^{n \times n}$. Using exactly the same argument as above, any valid inequality for the completely positive cone yields a valid inequality for IQ_n^+. Unfortunately, this additional information does not help us much, because a complete linear description of the completely positive cone is unknown, and unlikely to be found for general n [12].

3.4 A Connection to the Boolean Quadric Polytope

We close this section by pointing out a connection between IQ_n, IQ_n^+ and the so-called *boolean quadric polytope*. The boolean quadric polytope of order n is denoted by BQP_n and is defined as:

$$\mathrm{BQP}_n = \mathrm{conv}\left\{(x,y) \in \{0,1\}^{n+\binom{n}{2}} : y_{ij} = x_i x_j \ (1 \le i < j \le n)\right\} \ .$$

Note that the y_{ii} variables are not present in the case of BQP_n.

The boolean quadric polytope was defined by Padberg [14] in the context of quadratic 0-1 programming. It has many applications in other fields and has been studied in great depth [7].

We will need the following lemma:

Lemma 1. *For all* $1 \le i \le n$*, the inequality* $y_{ii} \ge x_i$ *is valid for* IQ_n*.*

Proof. This follows from the fact that all members of F_n satisfy $y_{ii} = x_i^2$, and the fact that $t^2 \ge t$ for any integer t. □

The following proposition states that BQP_n is essentially nothing but a face of IQ_n:

Proposition 9. *Let* H *be the face of* IQ_n *obtained by setting the inequality* $y_{ii} \ge x_i$ *to an equation for all* $1 \le i \le n$*. The boolean quadric polytope* BQP_n *is an affine image of* H*.*

Proof. Note that $t^2 = t$ if and only if $t \in \{0,1\}$. Therefore, the extreme points of H are precisely the members of F_n that satisfy $x \in \{0,1\}^n$. So there is a one-to-one correspondence between extreme points of H and extreme points of BQP_n. Moreover, every extreme point (x^*, y^*) of BQP_n can be mapped onto an extreme point of H simply by setting $y^*_{ii} = x^*_i$ for all $i = 1, \ldots, n$. This mapping is affine. □

An immediate consequence of Proposition 9 is that valid or facet-inducing inequalities for BQP_n can be *lifted* to yield valid or facet-inducing inequalities for IQ_n:

Corollary 2. *Suppose the inequality*

$$\sum_{i=1}^{n} a_i x_i + \sum_{1 \le i < j \le n} b_{ij} y_{ij} \le c$$

induces a facet of BQP_n*. Then there exists at least one facet-inducing inequality for* IQ_n *of the form*

$$\sum_{i=1}^{n} (a_i - \lambda_i) x_i + \sum_{i=1}^{n} \lambda_i y_{ii} + \sum_{1 \le i < j \le n} b_{ij} y_{ij} \le c \ ,$$

with $\lambda \in \mathbb{Q}^n$*.*

Similar results can be shown to hold for IQ_n^+.

4 Some Facet-Inducing Inequalities

We now move on to consider some specific classes of facet-inducing inequalities.

4.1 Non-negativity Inequalities

Since IQ_n^+ is contained in the completely positive cone, it is clear that all variables are constrained to be non-negative. The following theorem states conditions under which non-negativity inequalities induce facets of IQ_n^+:

Theorem 3. *The inequalities* $x_i \ge 0$ *for all* $1 \le i \le n$*, and the inequalities* $y_{ij} \ge 0$ *for all* $1 \le i < j \le n$*, induce facets of* IQ_n^+*. The inequalities of the form* $y_{ii} \ge 0$*, on the other hand, never induce facets of* IQ_n^+*.*

Proof. To see that the inequalities of the form $y_{ij} \ge 0$ induce facets, simply note that all but one of the affinely-independent points listed in the proof of Proposition 5 satisfy $y_{ij} = 0$. To see that the inequalities of the form $y_{ii} \ge 0$ do not induce facets, simply note that they are dominated by the inequalities $x_i \ge 0$ and $y_{ii} \ge x_i$ (refer to Fig. 1). The inequalities of the form $x_i \ge 0$ are a little more tricky: one can easily construct $n + \binom{n}{2}$ affinely-independent points with $x_i = 0$, but to complete the proof one needs an additional n extreme rays of IQ_n^+ having $x_i = 0$. The proof of Proposition 1 shows that there is a ray with $y_{ii} = 1$ and all other variables zero. Using a similar argument, one can show that, for all $j \ne i$, there is a ray with $x_j = y_{ij} = 1$ and all other variables zero. □

The non-negativity inequalities are of course not valid for IQ_n.

4.2 Split Inequalities

In this subsection, we introduce a more interesting class of inequalities, valid for both IQ_n^+ and IQ_n. Before presenting them, we recall the definition of *split disjunctions*, taken from [6]. A split disjunction is a disjunction of the form $(v^T x \leq s) \vee (v^T \geq s+1)$, where $v \in \mathbb{Z}^n$ and $s \in \mathbb{Z}$. Split disjunctions are obviously satisfied by all lattice points $x \in \mathbb{Z}^n$. An example of a split disjunction is illustrated in Fig. 2.

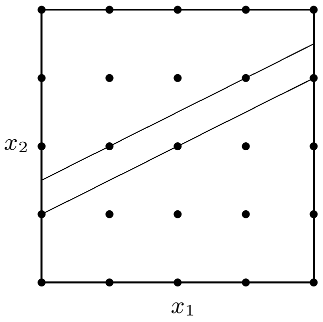


Fig. 2. The split disjunction $(x_1 - 2x_2 \leq -2) \vee (x_1 - 2x_2 \geq -1)$

The following proposition uses split disjunctions to derive an infinite family of valid inequalities:

Proposition 10. *For any vector $v \in \mathbb{Z}^n$ and scalar $s \in \mathbb{Z}$, the following 'split' inequality is valid for both IQ_n and IQ_n^+:*

$$(2s+1)v^T x + \sum_{i=1}^{n} v_i^2 y_{ii} + 2 \sum_{1 \leq i < j \leq n} v_i v_j y_{ij} + s(s+1) \geq 0 \ . \tag{3}$$

Proof. The split disjunction $(v^T x \leq -s-1) \vee (v^T x \geq -s)$ implies the quadratic inequality $(v^T x + s)(v^T x + s + 1) \geq 0$. Expanding this and substituting Y for xx^T yields $v^T Y v + (2s+1)v^T x + s(s+1) \geq 0$, which is equivalent to the inequality (3). $\square$

We remark that an important class of cutting planes for Mixed-Integer Linear Programs, called *split cuts*, can be derived using split disjunctions [3,6]. It is important to note however that the split inequalities (3) are *not* split cuts in the traditional sense. Indeed, split cuts arise from the interaction between a split disjunction and a set of linear constraints, whereas the split inequalities (3) are directly implied by the disjunctions themselves.

One can check that IQ_1 is completely described by the split inequalities, and that IQ_1^+ is completely described by the split inequalities together with the non-negativity inequality $x_1 \geq 0$. The following three theorems give further evidence that split inequalities are theoretically strong:

Theorem 4. *The split inequalities (3) dominate the psd inequalities (2).*

Proof. First, suppose that a psd inequality is derived using an integral vector v and an integral scalar s. Recall that the psd inequality can be written as $v^TYv + (2s)v^Tx + s^2 \geq 0$. This is dominated by the two inequalities $v^TYv + (2s+1)v^Tx + s(s+1) \geq 0$ and $v^TYv + (2s-1)v^Tx + s(s-1) \geq 0$, which are both split inequalities.

To complete the proof, we must show that the psd inequalities derived from integral v and s dominate all the others. Suppose a point (x^*, y^*) violates a psd inequality with non-integral v or s, and let ϵ be a small positive quantity. Let v' be a rational vector such that $|v'_i - v_i| < \epsilon$ for all i, and let s' be a rational number such that $|s' - s| < \epsilon$. Provided ϵ is small enough, the psd inequality obtained by using v' and s' in place of v and s will also be violated by (x^*, y^*). Now let M be a positive integer such that $Mv' \in \mathbb{Z}^n$ and $Ms' \in \mathbb{Z}$. The psd inequality with Mv' and Ms' in place of v' and s' will also be violated by (x^*, y^*). Therefore the original psd inequality is redundant. □

Theorem 5. *Split inequalities induce facets of* IQ_n *if the non-zero components of* v *are relatively prime.*

Proof. First, note that the trivial inequality $y_{11} \geq x_1$ is a split inequality, obtained by linearising the quadratic inequality $(x_1 - 1)x_1 \geq 0$. This trivial split inequality induces a facet of IQ_n, because all but one of the affinely-independent points listed in the proof of Proposition 5 satisfy $y_{11} = x_1$.

Now consider a non-trivial split inequality of the form (3), and assume that the non-zero components of v are relatively prime. A well-known result on integral matrices (see, e.g., p. 15 of Newman [13]) implies that there exists a unimodular matrix $U \in \mathbb{Z}^{n \times n}$ having v as its first row. Let U be such a matrix, and let $w \in \mathbb{Z}^n$ be an arbitrary vector satisfying $w_1 = s + 1$. Note that, if (x, y) is an extreme point of IQ_n and (x', y') is the transformed extreme point described in Theorem 1, then $x'_1 = v^Tx + s + 1$ and $y'_{11} = (x'_1)^2 = v^TYv + 2(s+1)v^Tx + (s+1)^2$. Thus, if we apply the transformation mentioned in Corollary 1 to the trivial split inequality $y_{11} \geq x_1$, we obtain the inequality $v^TYv + 2(s+1)v^Tx + (s+1)^2 \geq v^Tx + s + 1$. This is equivalent to the non-trivial split inequality. By Corollary 1, it induces a facet of IQ_n. □

Theorem 6. *Split inequalities induce facets of* IQ_n^+ *if the non-zero components of* v *are relatively prime and not all of the same sign.*

Proof. First, note that when v satisfies the stated condition, there exists a vector $w \in \mathbb{Z}^n$ such that $v^Tw = 0$ and such that $w_i > 0$ for all i. To see this, let k and k' be the number of components of v that are positive and negative, respectively, and let m be the product of the non-zero components of v. The desired vector w can be obtained by setting w_i to $k'|m|/v_i$ when $v_i > 0$, to $k|m|/|v_i|$ when $v_i < 0$, and to 1 otherwise.

Second, observe that an extreme point $(\bar{x}, \bar{y})$ of IQ_n satisfies the split inequality (3) at equality if and only if $v^T\bar{x} \in \{-s-1, -s\}$. Therefore, if $(\bar{x}, \bar{y})$ is such an extreme point, then so is the extreme point obtained by replacing $\bar{x}$ with $\bar{x}+w$, and adjusting $\bar{y}$ accordingly. Let us call this (affine) transformation 'shifting'.

Now, since the split inequality induces a facet of IQ_n under the stated conditions, there exist $n+\binom{n+1}{2}$ affinely-independent points in F_n that satisfy the split inequality at equality. By shifting this set of points, repeatedly if necessary, we obtain $n+\binom{n+1}{2}$ affinely-independent points in F_n^+ that satisfy the split inequality at equality. Therefore the split inequality induces a facet of IQ_n^+ as well. □

If the non-zero components of the vector v all have the same sign, then the split inequality need not induce even a proper face of IQ_n^+ (because there may not exist a lattice point $x \in \mathbb{Z}_+^n$ such that $v^T x \in \{-s-1, -s\}$). Theorem 2 implies however the following result:

Corollary 3. *Let $v \in \mathbb{Z}^n$ be such that all its components are relatively prime and of the same sign. Then there exists an integer s, of the opposite sign, such that the split inequality (3) induces a facet of* IQ_n^+.

To close this subsection, we remark that Propositions 9 and 10 imply the validity of the following inequalities for BQP_n:

$$\sum_{i=1}^{n} v_i(v_i+2s+1)x_i + 2\sum_{1\le i<j\le n} v_i v_j y_{ij} + s(s+1) \ge 0 \qquad (\forall v \in \mathbb{Z}^n, s \in \mathbb{Z}) \ . \quad (4)$$

These inequalities were discovered by Boros & Hammer [4].

4.3 Other Inequalities

We have seen that IQ_1^+ is completely described by the split inequalities and the non-negativity inequality $x_1 \ge 0$. A natural question is whether the split and non-negativity inequalities are enough to describe IQ_2^+. This is unfortunately not the case, as we now explain.

Consider the two lines in $\mathbb{R}^2$ defined by the equations $x_1 + x_2 = 3$ and $x_1 + 2x_2 = 4$. As illustrated in Fig. 3, these lines pass through several points in $\mathbb{Z}_+^2$. Moreover, all points in $\mathbb{Z}_+^2$ are either above both lines (satisfying $x_1 + x_2 \ge 3$ and $x_1 + 2x_2 \ge 4$), or below both lines (satisfying $x_1 + x_2 \le 3$ and $x_1 + 2x_2 \le 4$). This implies that all points in F_2^+ satisfy the non-linear inequality $(x_1 + x_2 - 3)(x_1 + 2x_2 - 4) \ge 0$. This implies that the linear inequality

$$-7x_1 - 9x_2 + y_{11} + 3y_{12} + 2y_{22} \ge 12$$

is valid for IQ_2^+. One can check (either by hand or with the aid of a computer) that this inequality induces a facet of IQ_2^+.

Using 'non-standard' split disjunctions of this kind, one can easily derive other inequalities that induce facets of IQ_n^+ for $n \ge 2$. Details will be given in the full version of the paper.

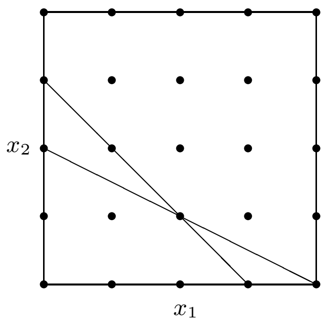


Fig. 3. A 'non-standard' split when $n = 2$

Turning attention to IQ_n, we have seen that IQ_1 is completely described by split inequalities. A natural question is whether the split inequalities are enough to describe IQ_2. We do not know the answer to this question. We are however able to show that the split inequalities do not completely describe IQ_6. Indeed, one can show using results in [7] that the boolean quadric polytope BQP_6 is not completely described by the Boros-Hammer inequalities ([4]). This implies, via Corollary 2, that there exist facet-inducing inequalities for IQ_6 that are not split inequalities. Specific inequalities of this kind will be presented in the full version of the paper.

5 Concluding Remarks

This paper marks a first step in applying polyhedral methods to Integer Quadratic Programs. There are many interesting open questions. We have already mentioned the question of whether one can optimise a linear function over IQ_n^+ in polynomial time for fixed n, and whether the split inequalities completely describe IQ_2. Another important question is whether the separation problem for the split inequalities can be solved in polynomial time.

Perhaps more importantly, it would be worthwhile extending the approach given in this paper to the mixed-integer case. Some preliminary observations on this case are the following. First, one has to deal with general convex sets rather than quasi-polyhedra, since the number of feasible solutions is no longer countable. Second, the split inequalities should be defined only when the components of the vector v are zero for all continuous variables, since otherwise they may not be valid. Third, it is no longer the case that the psd inequalities are dominated by the split inequalities. Indeed, if the vector v has a non-zero component for at least one continuous variable, it is even possible for a psd inequality to induce a maximal face of the convex set. Details will be given in the full version of the paper.

Acknowledgement. The author was supported by the Engineering and Physical Sciences Research Council under grant `EP/D072662/1`.

References

1. Aardal, K.I., Weismantel, R.: Polyhedral combinatorics. In: Dell'Amico, M., Mafioli, F., Martello, S. (eds.) Annotated Bibliographies in Combinatorial Optimization. Wiley, New York (1997)
2. Anderson, E.J., Goberna, M.A., López, M.A.: Simplex-like trajectories on quasi-polyhedral sets. Math. Oper. Res. 26, 147–162 (2001)
3. Balas, E.: Disjunctive programming. Ann. Discr. Math. 5, 3–51 (1979)
4. Boros, E., Hammer, P.L.: Cut-polytopes, Boolean quadric polytopes and nonnegative quadratic pseudo-Boolean functions. Math. Oper. Res. 18, 245–253 (1993)
5. Burer, S., Letchford, A.N.: On non-convex quadratic programming with box constraints. SIAM J. Opt. 20, 1073–1089 (2009)
6. Cook, W., Kannan, R., Schrijver, A.: Chvátal closures for mixed integer programming problems. Math. Program. 47, 155–174 (1990)
7. Deza, M.M., Laurent, M.: Geometry of Cuts and Metrics. Springer, Berlin (1997)
8. van Emde Boas, P.: Another NP-complete problem and the complexity of computing short vectors in a lattice. Technical Report 81-04, Mathematics Institute, University of Amsterdam (1981)
9. Khachiyan, L., Porkolab, L.: Integer optimization on convex semialgebraic sets. Discr. Comput. Geom. 23, 207–224 (2000)
10. Lovász, L., Schrijver, A.J.: Cones of matrices and set-functions and 0-1 optimization. SIAM J. Opt. 1, 166–190 (1991)
11. Michaels, D., Weismantel, R.: Polyhedra related to integer-convex polynomial systems. Math. Program. 105, 215–232 (2006)
12. Murty, K.G., Kabadi, S.N.: Some $\mathcal{NP}$-complete problems in quadratic and nonlinear programming. Math. Program. 39, 117–129 (1987)
13. Newman, M.: Integral Matrices. Academic Press, New York (1972)
14. Padberg, M.W.: The boolean quadric polytope: some characteristics, facets and relatives. Math. Program. 45, 139–172 (1989)
15. Ramana, M.: An Algorithmic Analysis of Multiquadratic and Semidefinite Programming Problems. PhD thesis, Johns Hopkins University, Baltimore, MD (1993)
16. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency. Springer, Berlin (2003)
17. Sherali, H.D., Adams, W.P.: A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems. Kluwer, Dordrecht (1998)
18. Yajima, Y., Fujie, T.: A polyhedral approach for nonconvex quadratic programming problems with box constraints. J. Global Opt. 13, 151–170 (1998)

An Integer Programming and Decomposition Approach to General Chance-Constrained Mathematical Programs

James Luedtke

Department of Industrial and Systems Engineering
University of Wisconsin-Madison, Madison, WI 53706, USA
jrluedt1@wisc.edu

Abstract. We present a new approach for exactly solving general chance constrained mathematical programs having discrete distributions. Such problems have been notoriously difficult to solve due to nonconvexity of the feasible region, and currently available methods are only able to find provably good solutions in certain very special cases. Our approach uses both decomposition, to enable processing subproblems corresponding to one possible outcome at a time, and integer programming techniques, to combine the results of these subproblems to yield strong valid inequalities. Computational results on a chance-constrained two-stage problem arising in call center staffing indicate the approach works significantly better than both an existing mixed-integer programming formulation *and* a simple decomposition approach that does not use strong valid inequalities. Thus, the strength of this approach results from the successful merger of stochastic programming decomposition techniques with integer programming techniques for finding strong valid inequalities.

Keywords: Stochastic programming, integer programming, chance constraints, probabilistic constraints, decomposition.

1 Introduction

We introduce a new approach for exactly solving general chance-constrained mathematical programs (CCMPs). A chance constraint states that the chosen decision vector should, with high probability, lie within a region that depends on a set of random variables. A generic CCMP can be stated as

$$\min\{f(x) \mid \mathbb{P}\{x \in P(\omega)\} \geq 1-\epsilon,\ x \in X\}\ , \tag{1}$$

where $x \in \mathbb{R}^n$ is the vector of decision variables to be chosen to minimize $f(x)$, ω is a random vector and $P(\omega) \subseteq \mathbb{R}^n$ is a region parameterized by ω. The interpretation is that the region $P(\omega)$ is defined such that the event $x \notin P(\omega)$ is an undesirable outcome. The parameter $\epsilon \in (0,1)$ is a risk tolerance, typically small, that limits the likelihood of such an outcome. A problem with uncertain *linear* constraints is the special case of this problem in which $P(\omega) = \{x \mid T(\omega)x \geq b(\omega)\}$

and a two-stage problem with the possibility to take recourse after observing the random outcome has $P(\omega) = \{x \mid \exists y \text{ with } T(\omega)x + W(\omega)y \geq b(\omega)\}$. (In §5.1 we describe an example application.)

Our approach works for CCMPs with *discrete* (and finite support) distribution. Specifically, we assume that[1] $\mathbb{P}\{\omega = \omega^k\} = 1/N$ for $k = 1, \ldots, N$. We refer to the possible outcomes as *scenarios*. While this is a restriction, recent results on using sample average approximations on problems with general distributions [1] demonstrate that such finite support approximations, when obtained from a Monte Carlo sample of the original distribution, can be used to find good feasible solutions to the original problem *and* statistical bounds on solution quality. We also assume that the sets $P_k := P(\omega^k)$ are polyhedra described by

$$P_k = \{x \in \mathbb{R}^n_+ \mid \exists y \in \mathbb{R}^d_+ \text{ with } T^k x + W^k y \geq b^k\} , \tag{2}$$

where $b^k \in \mathbb{R}^m$ and T^k and W^k are appropriately sized matrices. The special case with $d = 0$ yields a mathematical program with chance-constrained linear constraints having random coefficients: $\mathbb{P}\{T(\omega)x \geq b(\omega)\} \geq 1 - \epsilon$. We omit the details here, but our approach can be extended to the case in which $P(\omega)$ is convex, provided we have oracles for separation and optimization over $P(\omega)$.

CCMPs have a long history dating back to Charnes, Cooper and Symonds [2]. The general version considered here, which enforces a *system* of constraints to be enforced with high probability, was introduced by Prékopa [3]. However, solution of the general problem (1) has remained computationally challenging for two reasons: the feasible region is generally not convex, and evaluating solution feasibility requires multi-dimensional integration. As discussed above, the latter difficulty can be addressed by a sample-average approximation approach. However, this approach still requires a computationally efficient method for solving the resulting approximation problem which still has the form (1), except that the probability distribution is simplified to one with finite support.

Methods for obtaining provably good solutions for CCMPs have been successful in only a couple very special cases. If the chance constraint consists of a single row and all random coefficients are normally distributed [4,5], then a deterministic (nonlinear and convex) reformulation is possible. If the randomness appears only in the right-hand side of the chance constraints (i.e., $P(\omega) = \{x \mid Tx \geq b(\omega)\}$) and the random vectors $b(\omega)$ have continuous and log-concave distributions, the resulting feasible region is convex and so nonlinear programming techniques can be used [3]. If the randomness appears only in the right-hand side and the distribution of $b(\omega)$ is discrete, then approaches based on certain "efficient points" of the random vector [6,7] or on strong integer programming formulations [8,9] have been proposed.

Very few methods are available for finding provably good solutions for CCMPs with the general structure we consider here, e.g., for problems having linear constraints with random coefficients or two stage problems as in (2). In [10], an approach based on an integer programming formulation (which we give in §2),

[1] The extension to more general discrete distributions of the form $\mathbb{P}\{\omega = \omega^k\} = p_k$, where $p_k \geq 0$ and $\sum_k p_k = 1$, is straightforward and is omitted to simplify exposition.

strengthened with precedence constraints is presented. In more recent work, [11] presents a specialized branch-and-cut algorithm based on identification of irreducible infeasible sets of certain linear inequality systems. While these are important contributions, the size of instances that are demonstrated to be solvable with these approaches is very limited, in particular, because these approaches do not enable decomposition. In another recent important stream of research, a number of *conservative* approximations [12,13,14,15,16,17,18] have been studied that solve tractable (convex) approximations to yield feasible solutions to general CCMPs. However, these approaches do not say anything about the cost of the resulting solutions relative to the optimal, and tend to yield highly conservative solutions.

Our approach is important because it is an *exact* approach for solving problems with general chance constraints, and as we show in §5, has the potential to solve problems with high-dimensional random parameters and a large number of scenarios. The approach builds on the ideas of [19,8] that were very successful for solving chance-constrained problems with *random right-hand side only* by developing a method to apply the same types of valid inequalities used there to the much more general case considered here. The other important aspect of our approach is that it enables decomposition of the problem into single scenario subproblems. This is important for solving CCMPs with discrete distributions because the problem size grows as the size of the support increases. The ability of this approach to solve large instances of this problem, even for the particular structure of the test problem described in §5.1, is significant because, until now, a major impediment to using a chance-constrained model has been the difficulty in solving such problems in all but a few very special cases. The approach we present here, when combined with the sample average approximation results of [1], has the potential to remove this barrier.

Decomposition has long been used for solving traditional two-stage stochastic programming problems, where the objective is to minimize the sum of costs of the first stage decisions and the expected costs of second-stage recourse decisions (see, e.g., [20,21,22]). For CCMPs, the only existing paper we are aware of that considers a decomposition approach is [23] which applies a decomposition approach to a chance-constrained formulation of an application insuring vital arcs in a critical path network. The decomposition idea is similar to what we present here, but the mechanism for generating cuts is significantly different: they use a convex hull reformulation (based on Relaxation-Linearization techniques) which involves "big-M" constants, likely leading to weak inequalities. In contrast, we combine the valid inequalities we obtain from different subproblems in a way that avoids the need for "big-M" constants and hence yields strong valid inequalities for the overall problem. As we will see in the computational results in §5, the use of strong valid inequalities makes a very significant difference beyond the benefits obtained from decomposition.

The remainder of this extended abstract is organized as follows. We start with an overview of the approach in §2. In §3 we describe how we generate strong valid inequalities, and in §4 we describe the decomposition branch-and-cut algorithm. Finally, we present preliminary computational results of the approach in §5.

2 Overview of the Approach

To fix notation, and motivate the approach, we first describe a standard integer programming formulation of problem (1). We also make a couple assumptions that assure this formulation is well-defined, and that also simplify exposition of the main results in the rest of the paper. We assume without loss of generality that the sets P_k are non-empty for all $k \in \mathcal{N}$, since we could otherwise discard such a scenario and consider a problem with risk tolerance $\epsilon' = \epsilon - 1/N$. We also assume that the sets $P_k \cap X$ are compact for all $k \in \mathcal{N}$. Finally, for notational convenience we define the scenario index set $\mathcal{N} = \{1, \ldots, N\}$.

The standard mixed-integer programming formulation (e.g., [10]) uses a binary variable z_k for each scenario k, where $z_k = 0$ implies the constraints of scenario k should be satisfied:

$$\min\ f(x) \tag{3a}$$

$$\text{s.t. } T^k x + W^k y^k + z_k M_k \geq b^k, \quad k \in \mathcal{N}\ , \tag{3b}$$

$$\sum_{k=1}^{N} z_k \leq p\ , \tag{3c}$$

$$x \in X, \quad z \in \{0,1\}^N, y^k \in \mathbb{R}^d_+,\ k \in \mathcal{N}\ . \tag{3d}$$

Here $p := \lfloor (1-\epsilon)N \rfloor$ and $M_k \in \mathbb{R}^m_+$ are sufficiently large to ensure that when $z_k = 1$, constraints (3b) are not active. On the other hand, when $z_k = 0$, constraints (3b) enforce $x \in P_k$. Thus, (3c), which is a rewritten and strengthened version of the constraint

$$\frac{1}{N}\sum_{k=1}^{N}(1 - z_k) \geq 1 - \epsilon\ ,$$

successfully models the constraint $\mathbb{P}\{x \in P(\omega)\} \geq 1 - \epsilon$. Our approach is motivated by the desire to avoid the use of big-M constants as in (3b), which are likely to lead to weak lower bounds when solving a continuous relaxation of (3), and also to use decomposition to avoid explicit introduction of the constraints (3b) and recourse variables y^k which may make (3) very large-scale if N is large.

The goal of our approach, to avoid using big-M constraints and associated variables, is similar in spirit to the goal of combinatorial Benders' cuts introduced by Codato and Fischetti [24]. However, we are able to take advantage of the structural properties of the CCMP to obtain stronger valid inequalities. In particular, the valid inequalities we use include both the "logical" z_k variables and the variables x, in contrast to the combinatorial Benders' cuts that are based only on the logical variables. We refer the reader to [11] for an approach to CCMPs that has a closer connection to combinatorial Benders' cuts.

Our decomposition algorithm is based on a master problem that includes the original variables x, and the binary variables z. The constraints (3b) are enforced implicitly with cutting planes, similar in spirit to a Benders' decomposition approach. The key difference, however, is that given the mixed-integer nature of our master problem, we seek to add cutting planes that are strong. Specifically, we are interested in strong valid inequalities for the projection of the feasible region of (3) into the space of x and z variables. Specifically, we define this projection as

$$F = \big\{x \in X, z \in \{0,1\}^N \mid \exists y \in \mathbb{R}^{d\times N}_+ \;\text{ s.t. (3b)} - \text{(3c) hold}\big\} . \tag{4}$$

Note that $(x, z) \in F$ if and only if $x \in X$, $z \in \{0,1\}^N$ satisfies (3c), and $x \in P_k$ for any k with $z_k = 0$. Given this definition of F, we can then succinctly state a reformulation of the original chance-constrained problem (1) as:

$$\min\{f(x) \mid (x,z) \in F\} . \tag{5}$$

Our algorithm solves this reformulation.

In §3 we describe how we obtain strong valid inequalities for F, for a *given* set of coefficients on the x variables. Then, in §4 we describe the decomposition approach which naturally suggests a choice for the coefficients on the x variables that leads to a convergent branch-and-cut algorithm. In our current implementation, we use only this approach for choosing these coefficients, but we believe that, depending on the problem structure, alternative approaches may be useful for yielding additional strong valid inequalities.

3 Generating Strong Valid Inequalities

We now describe our procedure for generating valid inequalities of the form

$$\alpha x + \pi z \geq \beta \tag{6}$$

for the set F defined in (4), where $\alpha \in \mathbb{R}^n$, $\pi \in \mathbb{R}^N$, and $\beta \in \mathbb{R}$. We assume here that the coefficients α are given, so our task is find π and β that make (6) valid for F. In addition, given a possibly fractional solution $(\hat{x}, \hat{z})$ our *separation* task is to find, if possible, π and β such that $(\hat{x}, \hat{z})$ violate the resulting inequality.

The approach is very similar to that used in [19,8], which applies only to chance-constrained problems with random right-hand side. However, by exploiting the fact that we have assumed α to be fixed, we are able to reduce our significantly more general problem to the structure studied in [19,8] and ultimately apply the same types of valid inequalities.

The first step in our procedure is to solve the following auxiliary "single scenario" problems:

$$h_k(\alpha) := \min\big\{\alpha x \mid x \in P_k \cap \bar{X}\big\}, \quad k \in \mathcal{N} . \tag{7}$$

Here $\bar{X} \subseteq \mathbb{R}^n$ is a relaxation of the set X, i.e., $\bar{X} \supseteq X$, chosen such that $P_k \cap \bar{X}$ is non-empty and compact, guaranteeing that the above optimal values exist. The choice of $\bar{X}$ represents a trade-off in time to compute the values $h_k(\alpha)$ and strength of the resulting valid inequalities. Choosing $\bar{X} = \mathbb{R}^n$ leads to a problem for calculating $h_k(\alpha)$ that has the fewest number of constraints (and presumably the shortest computation time), but choosing $\bar{X} = X$ yields the strongest inequalities. In particular, if X is described as a polyhedron with additional integer restrictions on some of the variables, problem (7) would become a mixed-integer program and hence could be computationally demanding to solve, although doing so may yield significantly better valid inequalities.

Observe that calculation of the $h_k(\alpha)$ values decomposes by scenario and can be easily implemented in parallel. Having obtained the values $h_k(\alpha)$ for $k \in \mathcal{N}$, we then sort them to obtain a permutation σ of $\mathcal{N}$ such that:

$$h_{\sigma_1}(\alpha) \geq h_{\sigma_2}(\alpha) \geq \cdots \geq h_{\sigma_N}(\alpha) \, .$$

Although the permutation depends on α, we suppress this dependence to simplify notation. Our first lemma uses these values to establish a set of "base" inequalities that are valid for F, which we ultimately combine to obtain stronger valid inequalities.

Lemma 1. *The following inequalities are valid for F:*

$$\alpha x + (h_{\sigma_i}(\alpha) - h_{\sigma_{p+1}}(\alpha)) z_{\sigma_i} \geq h_{\sigma_i}(\alpha), \quad i = 1, \ldots, p \, . \tag{8}$$

The proof of this result is almost identical to an argument in [19] and follows from the observation that $z_k = 0$ implies $\alpha x \geq h_k(\alpha)$, whereas (3c) implies that $z_k = 0$ for at least one of the $p+1$ largest values of $h_k(\alpha)$.

Now, as was done in [19,8], we can apply the *star* inequalities of [25], or equivalently in this case, the mixing inequalities of [26] to "mix" the inequalities (8) to obtain additional strong valid inequalities.

Theorem 1 ([25,26]). *Let $T = \{t_1, t_2, \ldots, t_l\} \subseteq \{\sigma_1, \ldots, \sigma_p\}$ be such that $h_{t_i}(\alpha) \geq h_{t_{i+1}}(\alpha)$ for $i = 1, \ldots, l$, where $h_{t_{l+1}}(\alpha) := h_{\sigma_{p+1}}(\alpha)$. Then the inequality*

$$\alpha x + \sum_{i=1}^{l} (h_{t_i}(\alpha) - h_{t_{i+1}}(\alpha)) z_{t_i} \geq h_{t_1}(\alpha) \tag{9}$$

is valid for F.

These inequalities are strong in the sense that, if we consider the set Y defined by

$$Y = \left\{ (y, z) \in \mathbb{R} \times \{0,1\}^p \mid y + (h_{\sigma_i}(\alpha) - h_{\sigma_{p+1}}(\alpha)) z_{\sigma_i} \geq h_{\sigma_i}(\alpha), \quad i = 1, \ldots, p \right\} ,$$

then the inequalities (9), with αx replaced by y, define the convex hull of Y [25]. Furthermore, the inequalities of Theorem 1 are facet-defining for the convex hull of Y (again with $y = \alpha x$) if and only if $h_{t_1}(\alpha) = h_{\sigma_1}(\alpha)$, which suggests that when searching for a valid inequality of the form (9), one should always include $\sigma_1 \in T$. In particular, the valid inequalities

$$\alpha x + (h_{\sigma_1}(\alpha) - h_{\sigma_i}(\alpha)) z_{\sigma_1} + (h_{\sigma_i}(\alpha) - h_{\sigma_{p+1}}(\alpha)) z_{\sigma_i} \geq h_{\sigma_1}(\alpha), \quad i = 1, \ldots, p \, , \tag{10}$$

dominate the inequalities (8) which can be obtained by aggregating (10) with the valid inequalities $z_{\sigma_1} \leq 1$ with a weight of $(h_{\sigma_1}(\alpha) - h_{\sigma_i}(\alpha))$ on the latter.

Theorem 1 presents an exponential family of valid inequalities, but given a point $(\hat{x}, \hat{z})$ separation of these inequalities can be accomplished very efficiently. In [25] an algorithm based on finding a longest path in an acyclic graph is presented that has complexity $O(p^2)$, and [26] gives an $O(p \log p)$ algorithm. We use the algorithm of [26].

4 Decomposition Algorithm

We are now ready to describe the branch-and-cut decomposition algorithm. The algorithm works with a master relaxation defined as follows:

$$\mathrm{RP}^*(N_0, N_1, C) := \min f(x) \tag{11a}$$

$$\text{s.t.} \sum_{k=1}^{N} z_k \leq p, \quad (x, z) \in C, \quad x \in X, \quad z \in [0,1]^N , \tag{11b}$$

$$z_k = 0, k \in N_0, z_k = 1, \ k \in N_1 . \tag{11c}$$

Here N_0 is the set of binary variables currently fixed to 0, N_1 is the set of binary variables currently fixed to 1, and C is the relaxation defined by all the globally valid inequalities added so far when the relaxation is solved. At the root node in the search tree, we set $N_0 = N_1 = \emptyset$ and $C = \mathbb{R}^{n \times N}$.

Algorithm 1 presents a simple version of the proposed approach. The algorithm is a basic branch-and-bound algorithm, with branching being done on the binary variables z_k, with the only important difference being how the nodes are processed (Step 2 in the algorithm). In this step, the current node relaxation (11) is solved repeatedly until no cuts have been added to the description of C or the lower bound exceeds the incumbent objective value U. Whenever an integer feasible solution $\hat{z}$ is found, and optionally otherwise, the cut separation routine SepCuts is called. The SepCuts routine *must* be called when $\hat{z}$ is integer feasible to check whether the solution $(\hat{x}, \hat{z})$ is truly feasible to the set F. The routine is optionally called otherwise to possibly improve the lower bound.

The SepCuts routine, described in Algorithm 2, attempts to find strong violated inequalities using the approach described in §3. The key here is the method for selecting the coefficients α that are taken as given in §3. The idea is to consider all scenarios k such that $\hat{z}_k = 0$, so that the associated constraints $x \in P_k$ are supposed to be satisfied, and for such scenarios test whether indeed this holds. If $\hat{x} \in P_k$, then the condition that $\hat{z}_k = 0$ should imply $\hat{x} \in P_k$ is not violated. However, if $\hat{x} \notin P_k$, this contradicts the value of $\hat{z}_k$, and hence we seek to find an inequality that cuts off this infeasible solution. We therefore find an inequality, say $\alpha x \geq \beta$, that is facet-defining for P_k, and that separates $\hat{x}$ from P_k. We then use the coefficients α to generate one or more strong valid inequalities as derived in §3. While stated as two separate steps, the test of $\hat{x} \in P_k$ (line 3) and subsequent finding of a facet-defining inequality of P_k that cuts off $\hat{x}$ if not would typically be done together. For example, if we have an inequality description of P_k (possibly in a lifted space such as in (2)) then this can be accomplished by solving an appropriate linear program. If P_k has special structure

Algorithm 1. Simple version of branch-and-cut decomposition algorithm.

```
1  $t \leftarrow 0$, $N_0(0) \leftarrow \emptyset, N_1(0) \leftarrow \emptyset, C \leftarrow \mathbb{R}^{n\times N}$, OPEN $\leftarrow \{0\}$, $U \leftarrow +\infty$;
2  while OPEN $\neq \emptyset$ do
3      Step 1: Choose $l \in$ OPEN and let OPEN $\leftarrow$ OPEN $\setminus \{l\}$;
4      Step 2: Process node $l$;
5        repeat
6            Solve (11);
7            if (11) is infeasible then
8                CUTFOUND $\leftarrow$ FALSE;
9            else
10               Let $(\hat{x}, \hat{z})$ be an optimal solution to (11), and
                 $\mathsf{lb} \leftarrow \mathrm{RP}^*(N_0(l), N_1(l), C)$;
11               if $\hat{z} \in \{0,1\}^N$ then
12                   CUTFOUND $\leftarrow$ SepCuts$(\hat{x}, \hat{z}, C)$;
13                   if CUTFOUND = FALSE then $U \leftarrow \mathsf{lb}$;
14               else
15                   CUTFOUND $\leftarrow$ FALSE;
16                   Optional: CUTFOUND $\leftarrow$ SepCuts$(\hat{x}, \hat{z}, C)$;
17               end
18           end
19       until CUTFOUND $\neq$ TRUE or $\mathsf{lb} \geq U$ ;
20     Step 3: Branch if necessary;
21       if $\mathsf{lb} < U$ then
22           Choose $k \in \mathcal{N}$ such that $\hat{z}_k \in (0,1)$;
23           $N_0(t+1) \leftarrow N_0(l) \cup \{k\}$, $N_1(t+1) \leftarrow N_1(l)$;
24           $N_0(t+2) \leftarrow N_0(l)$, $N_1(t+2) \leftarrow N_1(l) \cup \{k\}$;
25           $t \leftarrow t+2$;
26       end
27 end
```

(such as the constraint set of a shortest path problem) it may be accomplished with a specialized (e.g., combinatorial) algorithm.

Observe that, in line 2 of Algorithm 2, we actually test whether $\hat{x} \in P_k$ for any k such that $\hat{z}_k < 1$. To obtain a convergent algorithm, it would be sufficient to check only those k such that $\hat{z}_k = 0$; we also optionally check k such that $\hat{z}_k \in (0,1)$ in order to possibly generate additional strong valid inequalities. We now establish that Algorithm 1 solves (5).

Theorem 2. *Algorithm 1 terminates finitely, and at termination if $U = +\infty$, problem (5) is infeasible, otherwise U is the optimal value of (5).*

Proof (Sketch). The details of the proof are left out of this extended abstract. However, the first main point is that the algorithm terminates finitely because it is based on branching on a finite number of binary variables, and the processing of each node terminates finitely because the valid inequalities are derived from a finite number of facet-defining inequalities (for the polyhedral sets P_k). The

Algorithm 2. Cut separation routine SepCuts($\hat{x}, \hat{z}, C$).

```
Data: $\hat{x}, \hat{z}, C$
Result: If valid inequalities for $F$ are found that are violated by $(\hat{x}, \hat{z})$, adds
        these to description of $C$ and returns TRUE, else returns FALSE.
1  CUTFOUND $\leftarrow$ FALSE;
2  for $k \in \mathcal{N}$ such that $\hat{z}_k < 1$ do
3      if $\hat{x} \notin P_k$ then
4          Separate $\hat{x}$ from $P_k$: Find an inequality $\alpha x \geq \beta$ that is facet-defining for
           $P_k$ such that $\alpha\hat{x} < \beta$;
5          Using the coefficients $\alpha$, find a violated inequality for $F$ of the form (9)
           that is violated by $(\hat{x}, \hat{z})$ and add this to the description of $C$;
6          CUTFOUND $\leftarrow$ TRUE;
7          Optionally break;
8      end
9  end
10 return CUTFOUND;
```

second point is that the algorithm never cuts off an optimal solution because the branching never excludes part of the feasible region and only valid inequalities for the set F are added. The final point is that no solutions that are *not* in the feasible region F are accepted for updating the incumbent objective value U (in line 13 of the algorithm) because the SepCuts routine is always called for integer feasible solutions $\hat{z}$ and it can be shown that it is guaranteed to find a violated inequality if $(\hat{x}, \hat{z}) \notin F$.

Aside from solving the master relaxation (11) the main work of Algorithm 1 happens within the SepCuts routine. An advantage of this approach is that most of this work is done for one scenario at a time and can be implemented to be done in parallel. In particular, checking whether $\hat{x} \in P_k$ (and finding a violated facet-defining if not) for any k such that $\hat{z}_k < 1$ can be done in parallel. The subsequent work of generating a strong valid inequality is dominated by calculation of the values $h_k(\alpha)$ as in (7), which can also be done in parallel.

We have stated our approach in relatively simple form in Algorithm 1. However, as this approach is essentially a variant of branch-and-cut for solving a (particularly structured) integer programming problem, we can also use all the computational enhancements commonly used in such algorithms. In particular, using heuristics to find good feasible solutions early and using some sort of pseudocost branching [27], strong branching [28], or reliability branching [29] approach for choosing which variable to branch on would be important. In our implementation (described in §5.2) we have embedded the key cut generation step of our algorithm within the CPLEX commercial integer programming solver which has such enhancements already implemented.

In our definition of the master relaxation (11), we have enforced the constraints $x \in X$. If, X is a polyhedron and $f(x)$ is linear, (11) is a linear program. However, if X is not a polyhedron, suitable modifications to the algorithm could be made to ensure that the relaxations solved remain linear programming problems. For example, if X is defined by a polyhedron Q with integrality constraints on some of the variables, then we could instead define the master relaxation to enforce $x \in Q$, and then perform branching both on the integer-constrained x variables and on the z_k variables. Such a modification is also easy to implement within existing integer programming solvers. Note also that, in this case, Q would be a natural choice for the relaxation $\bar{X}$ of X used in §3 when obtaining the $h_k(\alpha)$ values as in (7).

5 Preliminary Computational Results

5.1 Call Center Staffing Problem

We tested our approach on randomly generated instances of a call center staffing problem recently studied in [30]. In this problem the staffing levels of different types of available servers (x_i for $i = 1, \ldots, n$) must be set before knowing what the actual arrival rates of the customers will be. The routing of arriving customers to servers, however, can be done as the arrivals are observed. In [30], a static and fluid approximation of the second-stage dynamic routing problem is used in which servers are simply (fractionally) allocated to customers, leading to the following formulation:

$$\min\{cx \mid \mathbb{P}\{x \in P(\Lambda)\} \geq 1 - \epsilon,\ x \in \mathbb{R}^n_+\},$$

where $c \in \mathbb{R}^n_+$ represent the staffing costs, Λ is a m-dimensional random vector of arrival rates, and

$$P(\lambda) = \Big\{x \in \mathbb{R}^n_+ \mid \exists y \in \mathbb{R}^{nm}_+ \text{ s.t. } \sum_{j=1}^m y_{ij} \leq x_i,\ \forall i,\ \sum_{i=1}^n \mu_{ij} y_{ij} \geq \lambda_j,\ \forall j\Big\}. \quad (12)$$

Here μ_{ij} is the service rate of server type i when serving customer type j ($\mu_{ij} = 0$ if server type i cannot serve customer type j). This formulation aims to choose minimum cost staffing levels such that the probability of meeting quality of service targets is high.

When generating the test instances, we first generated the service rates, and the mean and covariance of the arrival rate vector. We then generated the cost vector in such a way that "more useful" server types were generally more expensive, in order to make the solutions nontrivial. Finally, to generate specific instances with finite support, we sampled N joint-normally distributed arrival rate vectors independently using the fixed mean and covariance matrix for various sample sizes N. In all our test instances we use $\epsilon = 0.1$ as the risk tolerance.

We see that this problem has the two-stage structure given in (2), and hence available methods for finding exact solutions (or even any solution with a bound on optimality error) are very limited. However, we point out that the form of $P(\lambda)$ still possesses some special structure in that the second-stage constraints have no random coefficients (i.e., in the form of (2) the matrices T^k and W^k do not vary with k). In addition, the constraints $x \in X$ are very simple for this problem; we simply have $X = \mathbb{R}^n_+$. Thus, while this test problem is certainly beyond the capabilities of existing approaches, it is not yet a test of the algorithm in the most general settings.

Technically, this problem does not satisfy our assumptions given in §2 because the sets $\mathbb{R}^n_+ \cap P(\lambda)$ are not bounded. However, our approach really only requires that the optimal solutions to (7) always exist for any coefficient vector α of a facet-defining inequality for $P(\lambda)$. As valid inequalities for $P(\lambda)$ necessarily have non-negative coefficients, this clearly holds.

5.2 Implementation Details

We implemented our approach within the commercial integer programming solver CPLEX 11.2. The main component of the approach, separation of valid inequalities of the form (9), was implemented within a cut callback that CPLEX calls whenever it has finished solving a node (whether the solution is integer feasible or not) and also after it has found a heuristic solution. In the feasibility checking phase of the SepCuts routine (line 2) we searched for k with $\hat{z}_k < 1$ and $x \notin P_k$ in increasing order of $\hat{z}_k$ (so, in particular we always first check the scenarios k with $\hat{z}_k = 0$). For the *first* such k we find (and only the first) we add *all* the violated valid inequalities of the form (10) as well as the single most violated inequality of the form (9). Our motivation for adding the inequalities (10) is that they are sparse and this is a simple way to add additional valid inequalities in one round; we found that doing this yielded somewhat faster convergence.

5.3 Results

We compared our algorithm against the Big-M formulation (3) (with the M values chosen as small as possible) and also against a basic decomposition algorithm that does not use the strong valid inequalities of §3. We compare against this simple decomposition approach to understand whether the success of our algorithm is due solely to the decomposition, or whether the strong inequalities are also important. The difference between the basic decomposition algorithm and the strengthened version is in the type of cuts that are added in the SepCuts routine. Specifically, in the case of an uncertainty set P_k of the form (12), if we find a scenario k with $\hat{z}_k = 0$, and a valid inequality $\alpha x \geq \beta$ for the set P_k that is violated by $\hat{x}$, the basic decomposition algorithm simply adds the inequality

$$\alpha x \geq \beta z_k \ .$$

It is not hard to see that when the sets P_k have the form (12), this inequality is valid for F because $x \geq 0$ and any valid inequality for P_k has $\alpha \geq 0$. Furthermore, this inequality successfully cuts off the infeasible solution $(\hat{x}, \hat{z})$.

Table 1 presents the results of these three approaches for varying problem size in terms of number of agent types (n), number of customer types (m) - which is also the dimension of the random vector Λ, and number of scenarios N. These tests were done with a time limit of one hour. Unless stated otherwise,

Table 1. Results on call center staffing instances; solution time (sec) or final optimality gap (%)

n	m	N	Big-M	Basic Decomp	Strong Decomp
10	20	500	23.0%	1752[a]	2.9
		1000	27.3%	5.5%	17.3
		2000	-	10.1%	143.4
20	30	1000	28.9%[b]	7.3%	11.8
		2000	-	16.7%	27.5
		3000	-	24.3%	73.9
40	50	1000	-	16.2%	65.3
		2000	-	24.1%	190.9
		3000	-	28.7%	256.3

[a] Average based on nine instances that solved in time limit.
[b] Gap for one instance, remaining nine instances failed.

each entry is an average over ten randomly generated samples from the same base underlying instance (i.e., the instance is fixed, but ten different samples of the N scenarios are taken). The big-M formulation (3) only successfully solves the LP relaxation and finds a feasible solution for the two smallest instance sizes. The entries '-' in the other cases mean that either no solution was found in the time limit, or that the LP relaxation did not solve in the time limit. For the largest instances, CPLEX failed with an out-of-memory error before the time limit was reached. Using the basic decomposition approach makes a significant improvement over the big-M formulation in that feasible solutions are now found for all instances. However, only the smallest of the instances (and only 9 of 10 of them) could be solved to optimality, and the larger instances had very large optimality gaps after the limit. Combining decomposition with strong valid inequalities ("Strong Decomp" in the table) we are able to solve all the instances to optimality in an average of less than five minutes.

To understand these results a little better, we present in Table 2 the root gaps (relative to the optimal values) after all cuts have been added for the

Table 2. Average root gaps and nodes for decomposition approaches

			Root gap (%)		Nodes	
n	m	N	Basic	Strong	Basic	Strong
10	20	500	20.3%	0.00%	22969	0
		1000	20.1%	0.01%	15034	0
		2000	19.5%	0.01%	4641	6.7
20	30	1000	20.2%	0.00%	6271	0
		2000	19.9%	0.01%	557	0
		3000	20.4%	0.00%	399	0.1
40	50	1000	20.1%	0.00%	878	0.3
		2000	20.7%	0.00%	101	0.1
		3000	20.7%	0.00%	13	1

two decomposition approaches. We also present the average number of nodes processed in each approach (to the time limit for the basic approach, and to optimality for our approach). It is clear that the strong valid inequalities lead to very strong relaxations for this particular problem, and hence very few nodes need to be explored. In comparison, for the smallest instance size, in which the basic decomposition approach can solve most of the instances, the average number of nodes required is over 20,000. (The smaller number of nodes for the larger instances merely reflects that fewer could be processed in the time limit.)

6 Discussion

We have presented a promising approach for solving general CCMPs, although additional computational tests are needed on problems having more general structures than the test problem we considered. The approach uses both decomposition, to enable processing subproblems corresponding to one scenario at a time, and integer programming techniques, to yield strong valid inequalities. From a stochastic programming perspective, it is not surprising that decomposition is necessary to yield an efficient algorithm, as this is well-known for traditional two-stage stochastic programs. From an integer programming perspective, it is not surprising that using strong valid inequalities has an enormous impact. The approach presented here represents a successful merger of these approaches to solve CCMPs.

Acknowledgments. The author thanks Shabbir Ahmed for the suggestion to compare the presented approach with a basic decomposition algorithm. This research has been supported in part by the National Science Foundation under grant CMMI-0952907.

References

1. Luedtke, J., Ahmed, S.: A sample approximation approach for optimization with probabilistic constraints. SIAM J. Optim. 19, 674–699 (2008)
2. Charnes, A., Cooper, W.W., Symonds, G.H.: Cost horizons and certainty equivalents: an approach to stochastic programming of heating oil. Manage. Sci. 4, 235–263 (1958)
3. Prékopa, A.: On probabilistic constrained programmming. In: Kuhn, H.W. (ed.) Proceedings of the Princeton Symposium on Mathematical Programming, Princeton, NJ, pp. 113–138. Princeton University Press, Princeton (1970)
4. Charnes, A., Cooper, W.W.: Deterministic equivalents for optimizing and satisficing under chance constraints. Oper. Res. 11, 18–39 (1963)
5. Calafiore, G., El Ghaoui, L.: On distributionally robust chance-constrained linear programs. J. Optim. Theory Appl. 130, 1–22 (2006)
6. Beraldi, P., Ruszczyński, A.: The probabilistic set-covering problem. Oper. Res. 50, 956–967 (2002)
7. Dentcheva, D., Prékopa, A., Ruszczyński, A.: Concavity and efficient points of discrete distributions in probabilistic programming. Math. Program. 89, 55–77 (2000)

8. Luedtke, J., Ahmed, S., Nemhauser, G.L.: An integer programming approach for linear programs with probabilistic constraints. Math. Program. 12, 247–272 (2010)
9. Saxena, A., Goyal, V., Lejeune, M.: MIP reformulations of the probabilistic set covering problem. Math. Program. 121, 1–31 (2009)
10. Ruszczyński, A.: Probabilistic programming with discrete distributions and precedence constrained knapsack polyhedra. Math. Program. 93, 195–215 (2002)
11. Tanner, M., Ntaimo, L.: IIS branch-and-cut for joint chance-constrained programs with random technology matrices (2008)
12. Ben-Tal, A., Nemirovski, A.: Robust solutions of linear programming problems contaminated with uncertain data. Math. Program. 88, 411–424 (2000)
13. Bertsimas, D., Sim, M.: The price of robustness. Oper. Res. 52, 35–53 (2004)
14. Calafiore, G., Campi, M.: Uncertain convex programs: randomized solutions and confidence levels. Math. Program. 102, 25–46 (2005)
15. Nemirovski, A., Shapiro, A.: Scenario approximation of chance constraints. In: Calafiore, G., Dabbene, F. (eds.) Probabilistic and Randomized Methods for Design Under Uncertainty, pp. 3–48. Springer, London (2005)
16. Nemirovski, A., Shapiro, A.: Convex approximations of chance constrained programs. SIAM J. Optim. 17, 969–996 (2006)
17. Erdoğan, E., Iyengar, G.: Ambiguous chance constrained problems and robust optimization. Math. Program. 107, 37–61 (2006)
18. Erdoğan, E., Iyengar, G.: On two-stage convex chance constrained problems. Math. Meth. Oper. Res. 65, 115–140 (2007)
19. Luedtke, J., Ahmed, S., Nemhauser, G.: An integer programming approach for linear programs with probabilistic constraints. In: Fischetti, M., Williamson, D.P. (eds.) IPCO 2007. LNCS, vol. 4513, pp. 410–423. Springer, Heidelberg (2007)
20. Birge, J., Louveaux, F.: Introduction to stochastic programming. Springer, New York (1997)
21. Van Slyke, R., Wets, R.J.: L-shaped linear programs with applications to optimal control and stochastic programming. SIAM J. Appl. Math. 17, 638–663 (1969)
22. Higle, J.L., Sen, S.: Stochastic decomposition: an algorithm for two-stage stochastic linear programs. Math. Oper. Res. 16, 650–669 (1991)
23. Shen, S., Smith, J., Ahmed, S.: Expectation and chance-constrained models and algorithms for insuring critical paths (2009) (submitted for publication)
24. Codato, G., Fischetti, M.: Combinatorial benders' cuts for mixed-integer linear programming. Oper. Res. 54, 756–766 (2006)
25. Atamtürk, A., Nemhauser, G.L., Savelsbergh, M.W.P.: The mixed vertex packing problem. Math. Program. 89, 35–53 (2000)
26. Günlük, O., Pochet, Y.: Mixing mixed-integer inequalities. Math. Program. 90, 429–457 (2001)
27. Linderoth, J., Savelsbergh, M.: A computational study of search strategies for mixed integer programming. INFORMS J. Comput. 11, 173–187 (1999)
28. Applegate, D., Bixby, R., Chvátal, V., Cook, W.: Finding cuts in the TSP. Technical Report 95-05, DIMACS (1995)
29. Achterberg, T., Koch, T., Martin, A.: Branching rules revisited. Oper. Res. Lett. 33, 42–54 (2004)
30. Gurvich, I., Luedtke, J., Tezcan, T.: Call center staffing with uncertain arrival rates: a chance-constrained optimization approach. Technical report (2009)

An Effective Branch-and-Bound Algorithm for Convex Quadratic Integer Programming

Christoph Buchheim[1,2], Alberto Caprara[2], and Andrea Lodi[2]

[1] Fakultät für Mathematik, Technische Universität Dortmund
Vogelpothsweg 87, D-44227 Dortmund, Germany
christoph.buchheim@tu-dortmund.de
[2] DEIS, Università di Bologna
Viale Risorgimento 2, I-40136 Bologna, Italy
{alberto.caprara,andrea.lodi}@unibo.it

Abstract. We present a branch-and-bound algorithm for minimizing a convex quadratic objective function over integer variables subject to convex constraints. In a given node of the enumeration tree, corresponding to the fixing of a subset of the variables, a lower bound is given by the continuous minimum of the restricted objective function. We improve this bound by exploiting the integrality of the variables using suitably-defined lattice-free ellipsoids. Experiments show that our approach is very fast on both unconstrained problems and problems with box constraints. The main reason is that all expensive calculations can be done in a preprocessing phase, while a single node in the enumeration tree can be processed in linear time in the problem dimension.

1 Introduction

Nonlinear integer optimization has attracted a lot of attention recently. Besides its practical importance, it is challenging from a theoretical and methodological point of view. While intensive research has led to tremendous progress in the practical solution of integer linear programs in the last decades [9], practical methods for the nonlinear case are still rare [5]. This is true even in special cases, such as *Convex Quadratic Integer Programming* (CQIP):

$$\begin{aligned} \min \;\; & f(x) = x^\top Qx + L^\top x + c \\ \text{s.t.} \;\; & x \in \mathbb{Z}^n \cap X \; , \end{aligned} \tag{1}$$

where Q is an $n \times n$ positive definite symmetric matrix, $L \in \mathbb{R}^n$, $c \in \mathbb{R}$, and $X \subseteq \mathbb{R}^n$ is a convex set for which membership can be tested in polynomial time. Positive definiteness of Q guarantees strict convexity of f.

1.1 The Two Applications Considered

Our original motivating application is in Electronics and arises in the development of pulse coders for actuation, signal synthesis and audio amplification. The aim is to synthesize periodic waveforms by either bipolar or tripolar pulse codes. The latter problem amounts to solving CQIP for $X = [-1, 1]^n$. In other words, each variable can take three different values $\{-1, 0, 1\}$, leading to a *ternary* CQIP. The real problem is called *Filtered Approximation* and corresponds to finding a *discrete-valued* $x(kT)$ so that the discrete-time *continuous* signal $w(kT)$ and $x(kT)$ are as similar as possible once filtered through a discrete-time, linear, causal filter $H(z)$ with three states. This must be done in the time interval $[n_1T, n_2T]$ where $n_2 - n_1 = n$ in the mapping of FA to CQIP. Moreover, the mapping shows that Q only depends on the filter $H(z)$ and it is then strictly positive definite because the quantity x^TQx can be interpreted as the energy over the period of $x(nT)$ filtered by $H(z)$ (see, e.g., [2]).

Besides the original motivation, if Q is positive definite and $X = \mathbb{R}^n$, CQIP is equivalent to the *Closest Vector Problem* (CVP), which, given a basis $b^1, \ldots, b^n$ of $\mathbb{R}^n$ and an additional vector v, calls for an integer linear combination of the vectors of the basis which is as close as possible (with respect to the Euclidean distance) to v. Equivalently, the problem calls for scalars $\lambda_1, \ldots, \lambda_n \in \mathbb{Z}$ such that $\| \sum_{i=1}^n \lambda_i b^i - v\|^2$ is minimized. It is elementary to check that this amounts to solving (1) for a positive definite matrix $Q = B^\top B$, where B is the $n \times n$ matrix whose columns are $b^1, \ldots, b^n$. Viceversa, given an instance of (1) in which Q is positive definite and symmetric, a corresponding CVP instance is obtained by computing a Cholesky decomposition of $Q = B^\top B$, which yields the basis $b^1, \ldots, b^n$, and by defining v accordingly. This problem has a wide relevance from both the theoretical and practical viewpoints. Specifically, it is very hard to approximate and it is used in cryptosystems (see, e.g., [10]).

1.2 Literature Review and State-of-the-Art

It is clear that the ternary CQIP is strongly related to at least two famous combinatorial optimization problems, namely *Unconstrained Binary Quadratic Programming* (UBQP) and *Maximum Cut* (MC). More precisely, UBQP is the special case of CQIP in which $X = [0, 1]^n$, while, given a graph $G = (V, E)$ and an edge function $c \in \mathbb{R}^E$, MC is to find a cut $\delta(W)$ of G having maximum weight $c(\delta(W))$. Although UBQP and MC have been treated in an almost separate way in the literature, a well-known result by [13] shows that they have in fact the same polyhedral description. Thus, one might be tempted to use the available algorithmic technology (as well as the software) to solve CQIP. We tried three different approaches, one based on the rooted-semimetric MC relaxation [4,12], one based on reduction to a binary CQIP by the obvious replacement $x_i = x_i^+ - x_i^-$ for $i = 1, \ldots, n$, and one based on convex MINLP approaches using Bonmin [1]. These three approaches performed quite poorly, and were widely outperformed by the direct application of the CPLEX MIQP solver [6] to the original problem, which was able to solve instances with n up to 50 for our real-world application in Electronics, too slowly however to be of practical use to engineers.

Regarding the second application, we are not aware of any computational work for the unconstrained CQIP, i.e., for CVP. Thus, for such a problem as well we used CPLEX MIQP as a reference, which was able to solve instances with n up to about 55.

1.3 Our Contribution

In this paper, we present a branch-and-bound algorithm for CQIP that is very fast at processing nodes but still computes reasonable lower bounds. The main observation behind the method is that, in the unconstrained case $X = \mathbb{R}^n$, strict convexity leads to a unique continuous minimum $\bar{x}$ of f over $\mathbb{R}^n$ which is easy to compute, yielding a lower bound that is of course valid also for any X. This bound can be improved by taking the integrality constraints into account, using lattice-free ellipsoids. Roughly speaking, the general idea is to center a given ellipsoid E in $\bar{x}$ and to compute the value λ such that the scaled ellipsoid λE contains at least one integer point on its border and no integer point in its interior. This can be done quickly if E is chosen appropriately. Then we find the minimum of the function f over the border of λE, which yields an improved lower bound on f.

Our enumeration strategy is depth-first, branching by fixing the value of one of the n variables. A crucial property of our algorithm is that we restrict the order in which variables are fixed. In other words, the set of variables fixed only depends on the depth of the node in the tree. We thus loose the flexibility of choosing the best branching variable, but this strategy allows us to process a single node in the tree much faster. Our main observation is that all expensive calculations to be performed in a node actually only depend on the depth d, i.e., on the set of variables fixed, but not on the particular values to which variables are fixed. This allows us to move these calculations to a preprocessing phase. We will show that, after this preprocessing, the running time per node is only $O(n - d)$, i.e., sublinear in the problem input size which is $\Theta(n^2)$.

Experimentally, we show that our approach leads to the solution of large CQIP instances for our real-world Filtered Approximation application in Electronics, with n up to about 120, allowing engineers to validate their practical approaches [2]. For the CVP we solve instances with n up to about 70. Even for the largest instances we can solve, our algorithm is able to process around $400,000$ nodes per second.

In Section 2 we discuss our methods for computing lower bounds, explaining how to incorporate constraints into our framework, whereas in Section 3 we present an outline of the overall branch-and-bound algorithm, illustrating how to compute the lower bounds in linear time. Finally, in Section 4 we present computational results for our algorithm. For space reasons, proofs are deferred to the full paper.

1.4 Basic Definitions and Notation

We will denote scalars by lower case letters, matrices by upper case letters, and vectors by both lower and upper case letters. Given a (column) vector $x \in \mathbb{R}^n$, we will let $x^\top$ denote its transposed (row) vector and x_i its ith component. In some cases it will be convenient to use apices to indicate vectors, such as x^i, and in this case we will denote the transposed vector by $(x^i)^\top$. As customary, given a matrix Q we will let $q_{i,j}$ denote its entry in the ith row and jth column. Given a scalar $a \in \mathbb{R}$, we will let $\lfloor a \rceil$ denote the integer value closest to a. Analogously, given a vector x we will let $\lfloor \bar{x} \rceil$ denote the componentwise rounding of x to the nearest integer.

A *box* $X \subseteq \mathbb{R}^n$ is a set of the form $X = \{x \in \mathbb{R}^n : l \le x \le u\}$, where $l, u \in \mathbb{R}^n$, $l \le u$. Let Q' be a positive semidefinite matrix. For $x' \in \mathbb{R}^n$ we consider the corresponding *ellipsoid*

$$E(Q', x') := \{x \in \mathbb{R}^n : (x - x')^\top Q'(x - x') \le 1\} \, ,$$

which is the translation of $E(Q') := E(Q', 0)$ by the vector x'. Moreover, for $\alpha \in \mathbb{R}^+$, we let $\alpha E(Q', x')$ denote $E(Q', x')$ scaled by α with respect to the center x', i.e.,

$$\alpha E(Q', x') := x' + \alpha E(Q') = \{x' + \alpha x : x \in E(Q')\} \, .$$

Given a closed convex set X, we let int X denote the interior of X and bd X the border of X.

2 Lower Bounds

As anticipated, the main inspiring observation for our method is that the computation of the minimum of the objective function (neglecting all constraints including integrality) simply requires solving a system of linear equations.

Remark 1. The unique minimum of $f(x) = x^\top Q x + L^\top x + c$ in case Q is positive definite is attained at $\bar{x} = -\frac{1}{2} Q^{-1} L$ and has value $c - \frac{1}{4} L^\top Q^{-1} L$. Moreover, for every $x \in \mathbb{R}^n$, $f(x) = f(\bar{x}) + (x - \bar{x})^T Q (x - \bar{x})$.

Our aim in this section is to get stronger bounds by exploiting the integrality of the variables and possibly the structure of X.

2.1 The Unconstrained Case

For a given $x' \in \mathbb{R}^n$, we let

$$\mu(Q', x') := \max\{\alpha : (\text{int}\, \alpha E(Q', x')) \cap \mathbb{Z}^n = \emptyset\} = \min\{\alpha : \alpha E(Q', x') \cap \mathbb{Z}^n \neq \emptyset\} \, ,$$

be the scaling factor α such that the ellipsoid $\alpha E(Q', x')$ contains some integer point on its border but no integer point in its interior.

Observation 1. *$\mu(Q', x') = \max\{\alpha : (x - x')^\top Q'(x - x') \ge \alpha^2$ for each $x \in \mathbb{Z}^n\}$.*

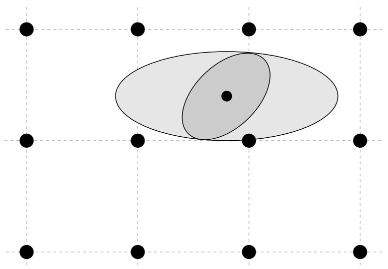


Fig. 1. Improving the lower bounds: the light gray ellipsoid is $E(Q', \bar{x})$ scaled by $\mu(Q', \bar{x})$; the dark gray ellipsoid is $E(Q, \bar{x})$ scaled by $\lambda(Q, Q')\mu(Q', \bar{x})$

Note that, given our objective function $f(x) = x^\top Qx + L^\top x + c$ and the associated continuous minimum $\bar{x}$, the level sets of $f(x)$ are precisely the borders of ellipsoids of the form $\alpha E(Q, \bar{x})$. Given this, it is easy to visualize the fact that finding the integer point that minimizes f is equivalent to scaling $E(Q, \bar{x})$ by α starting from $\alpha = 0$ and stopping as soon as the border of $\alpha E(Q, \bar{x})$ contains an integer point. This is the same as computing $\mu(Q, \bar{x})$. Since this is as hard as solving (1) when $X = \mathbb{R}^n$, we rather do the same scaling for some other ellipsoid $E(Q', \bar{x})$, and then scale $E(Q, \bar{x})$ in turn until it touches the border of the first ellipsoid, see Figure 1. This requires one to be able to compute $\mu(Q', \bar{x})$ as well as the maximum $\alpha \in \mathbb{R}_+$ such that $\alpha E(Q)$ is contained in $E(Q')$:

$$\lambda(Q, Q') := \max\{\alpha : \alpha E(Q) \subseteq E(Q')\} ,$$

noting that this latter value has nothing to do with $\bar{x}$.

Observation 2. $\lambda(Q, Q') = \max\{\alpha : Q - \alpha^2 Q' \succeq 0\} = \min\{1/\sqrt{x^\top Q' x} : x \in E(Q)\}$.

Proposition 1. *Given $f(x) = x^\top Qx + L^\top x + c$ with Q positive definite and continuous minimum $\bar{x}$ and a positive semidefinite matrix Q' of the same size as Q,*

$$\min\{f(x) : x \in \mathbb{Z}^n\} \geq f(\bar{x}) + \lambda^2(Q, Q')\mu^2(Q', \bar{x}) .$$

Note that, in order to find hopefully strong lower bounds, one would like to have matrices Q' such that on the one hand $E(Q')$ is as close as possible to $E(Q)$ and on the other $\mu(Q', \bar{x})$ is fast to compute. It is particularly fast to compute $\mu(Q', \bar{x})$ if Q' is a *split*, i.e., if $Q' = dd^\top$ for some vector $d \in \mathbb{Z}^n \setminus \{0\}$ with $d_1, \ldots, d_n$ coprime. In this case, we have

Observation 3. $\mu(dd^\top, x') = |d^\top x' - \lfloor d^\top x' \rceil|$.

In order to derive strong lower bounds, we aim at splits Q' that yield large factors $\lambda(Q, Q')$. To this end, we consider *flat directions* of the ellipsoid $E(Q)$, i.e., vectors $d \in \mathbb{Z}^n \setminus \{0\}$ minimizing the *width* of $E(Q)$ along d, defined as

$$\max\{d^\top x : x \in E(Q)\} - \min\{d^\top x : x \in E(Q)\} = 2\max\{d^\top x : x \in E(Q)\} .$$

Observation 4. *$d \in \mathbb{Z}^n \setminus \{0\}$ maximizes $\lambda(Q, dd^\top)$ if and only if it is a flat direction of $E(Q)$.*

The following remark is stated explicitly, e.g., in [3].

Remark 2. If $Q = B^\top B$ then the width of $E(Q)$ along d is given by $2\|d^\top B^{-1}\|$.

In other words, finding a flat direction of $E(Q)$ is equivalent to finding the coefficients $d_1, \dots, d_n$ yielding a shortest non-zero vector in the lattice generated by the columns of $(B^{-1})^\top$, which is well known to be NP-hard. A natural heuristic to compute short vectors is obtained by taking as candidates the vectors in a reduced basis of the lattice. Accordingly, we compute such a reduced basis by the LLL algorithm. If $t^1, \dots, t^n \in \mathbb{Z}^n \setminus \{0\}$ are the columns of the corresponding transformation matrix T, from the original basis to the reduced basis, we use the splits $Q'_i := t^i(t^i)^\top$ and compute $\mu(Q'_i, \bar{x}) = |(t^i)^\top \bar{x} - \lfloor (t^i)^\top \bar{x} \rceil|$ as in Observation 3.

Moreover, we consider the matrix $Q'_0 := \sum_{i=1}^n \lambda^2(Q, Q'_i) Q'_i$, so that the ellipsoid $E(Q'_0)$ is axis-parallel with respect to $t^1, \dots, t^n$.

Observation 5. $\mu(Q'_0, x') = \sqrt{\sum_{i=1}^n \lambda^2(Q, Q'_i)\mu^2(Q'_i, x')}$.

Note that $\lambda(Q, Q'_0)$ can be strictly smaller than one, so that the lower bound derived from Q'_0,

$$f(\bar{x}) + \lambda^2(Q, Q'_0)\mu^2(Q'_0, \bar{x}) = f(\bar{x}) + \lambda^2(Q, Q'_0) \sum_{i=1}^n \lambda^2(Q, Q'_i)\mu^2(Q'_i, \bar{x}) ,$$

can be weaker than the bound $f(\bar{x}) + \lambda^2(Q, Q'_i)\mu^2(Q'_i, \bar{x})$ derived from Q'_i for some $i \geq 1$. In general, which Q'_i gives the strongest lower bound depends on the position of $\bar{x}$.

Example 1. We illustrate the ideas in Section 2 by an example in the plane. Let

$$Q = \begin{pmatrix} 1 & -2 \\ -2 & 8 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -2 & -2 \end{pmatrix} \begin{pmatrix} 1 & -2 \\ 0 & -2 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & -2 \\ 0 & -2 \end{pmatrix}, \quad B^{-1} = \begin{pmatrix} 1 & -1 \\ 0 & -1/2 \end{pmatrix} .$$

The ellipse $E(Q)$ is shown in Figure 2. Short vectors of the lattice generated by the rows of B^{-1}, the vectors $(1, -1)^\top$ and $(0, -1/2)^\top$, are $(0, -1/2)^\top$ and $(1, 0)^\top$. These correspond to the transformation matrix

$$T = \begin{pmatrix} 0 & 1 \\ 1 & -2 \end{pmatrix}, \quad T^{-1} = \begin{pmatrix} -2 & 1 \\ 1 & 0 \end{pmatrix} ,$$

and hence to the (hopefully) flat directions $(0, 1)^\top$ and $(1, -2)^\top$. Thus

$$Q'_1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}, \quad Q'_2 = \begin{pmatrix} 1 & -2 \\ -2 & 4 \end{pmatrix}$$

and $\lambda(Q, Q'_1) = 2$, $\lambda(Q, Q'_2) = 1$. The ellipses $E(Q'_1)$ and $E(Q'_2)$ are illustrated in Figure 2. Finally, in this case we are lucky to obtain $Q'_0 = 4Q'_1 + Q'_2 = Q$, so that the improved lower bound given by Q'_0 agrees with the optimal solution of the problem, independently of L and c.

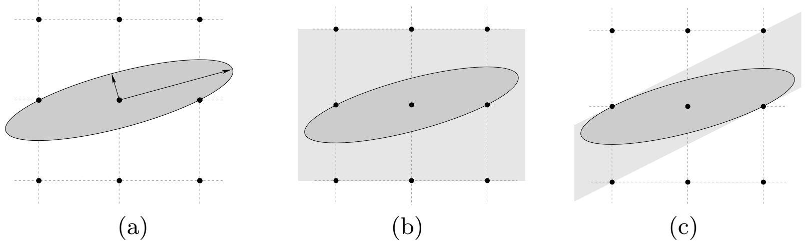


Fig. 2. The ellipse $E(Q)$ in (a); the split $E(Q'_1)$ given by $(1, 0)$ in (b); the split $E(Q'_2)$ given by $(1, -2)$ in (c)

2.2 Dealing with Constraints

Even if X is a box, determination of the continuous minimum of f over X is much more complex than in the unconstrained case [11] and would not fit well within our method, which is aimed at exploring branch-and-bound nodes quickly. On the other hand, even if we stick to the computation of the unconstrained continuous minimum, we can take into account the structure of X in the lower bound improvement of Section 2.1.

Assume we are given an arbitrary X and a linear optimization oracle for $X \cap \mathbb{Z}^n$, which in many applications is a reasonable assumption (and trivially true for a box). We can replace the definition of $\mu(Q', x')$ by

$$\mu_X(Q', x') := \max\{\mu : (\text{int } \mu E(Q', x')) \cap X \cap \mathbb{Z}^n = \emptyset\} .$$

In other words, we scale $E(Q', x')$ until it touches a feasible point, instead of simply any integer point. The counterpart of Proposition 1 reads:

Proposition 2. *Given $f(x) = x^\top Qx + L^\top x + c$ with Q positive definite and continuous minimum $\bar{x}$ and a positive semidefinite matrix Q' of the same size as Q,*

$$\min\{f(x) : x \in X \cap \mathbb{Z}^n\} \geq f(\bar{x}) + \lambda^2(Q, Q')\mu_X^2(Q', \bar{x}) .$$

Unfortunately, in this case we cannot compute the factors exactly, even for splits, in polynomial time. However, we can give lower bounds for $\mu_X(Q', x')$ that improve strongly over $\mu(Q', x')$ in general.

Observation 6. *Let $d_{\min}$ and $d_{\max}$ denote the minimum and maximum of $d^\top x$ over $X \cap \mathbb{Z}^n$, respectively. Then,*

$$\mu_X(dd^\top, x') \geq \max \begin{cases} |d^\top x' - \lfloor d^\top x' \rceil| \\ d^\top x' - d_{\max} \\ d_{\min} - d^\top x' \end{cases} .$$

Letting Q'_i be defined as in the previous section for $i = 0, \ldots, n$, this observation applies to Q'_i with $i = 1, \ldots, n$. We also have

Observation 7. $\mu_X(Q'_0, x') \geq \sqrt{\sum_{i=1}^n \lambda^2(Q, Q'_i)\mu_X^2(Q'_i, x')}$.

Note that this approach is still correct if the oracle optimizes over X instead of $X \cap \mathbb{Z}^n$, or any set containing $X \cap \mathbb{Z}^n$, but this may lead to weaker bounds. Finally, note that the calculation of $d_{\min}$ and $d_{\max}$ is trivial if X is a box.

3 Outline of the Algorithm

Our algorithm is a branch-and-bound algorithm with a depth-first enumeration strategy. Branching consists of fixing a single variable to an integer value, and is illustrated in detail in the following. Assume that the next variable to be fixed is x_i. We consider the value $\bar{x}_i$ of x_i in the continuous minimum computed in the current node. We fix x_i to integer values by increasing distance from $\bar{x}_i$. More precisely, if $\lfloor \bar{x}_i \rceil = \lfloor \bar{x}_i \rfloor$, the variable x_i is fixed to

$$\lfloor \bar{x}_i \rfloor, \lceil \bar{x}_i \rceil, \lfloor \bar{x}_i \rfloor - 1, \lceil \bar{x}_i \rceil + 1, \ldots \tag{2}$$

while otherwise it is fixed to

$$\lceil \bar{x}_i \rceil, \lfloor \bar{x}_i \rfloor, \lceil \bar{x}_i \rceil + 1, \lfloor \bar{x}_i \rfloor - 1, \ldots \tag{3}$$

By the convexity of f and its symmetry with respect to $\bar{x}$, the continuous minima with respect to these fixings are non-decreasing, so that we can stop as soon as one of these minima exceeds the current upper bound. In particular, we get a finite algorithm even without bounds on the variables, since we assume that f is strictly convex.

In order to enumerate subproblems as quickly as possible, our aim is to perform the most time-consuming computations in a preprocessing phase. In particular, having fixed d variables, we get the reduced objective function $\bar{f} : \mathbb{R}^{n-d} \to \mathbb{R}$ of the form

$$\bar{f}(x) = x^\top \bar{Q}_d x + \bar{L}^\top x + \bar{c}\ .$$

If x_i is fixed to r_i for $i = 1, \ldots, d$, we have $\bar{c} = c + \sum_{i=1}^d L_i r_i + \sum_{i=1}^d \sum_{j=1}^d q_{ij} r_i r_j$ and $\bar{L}_{j-d} = L_j + 2\sum_{i=1}^d q_{ij} r_i$ for $j = d+1, \ldots, n$. On the other hand, the matrix $\bar{Q}_d$ is obtained from Q by deleting the first d rows and columns, and therefore is positive definite and does not depend on the values at which the first d variables are fixed.

3.1 Achieving Quadratic Time Per Node

For $\bar{Q}_d$, we need the inverse matrix and all factors $\lambda(\bar{Q}_d, Q')$. For this reason, we do not change the order of fixing variables, i.e., we always fix the first unfixed variable according to an order that is determined before starting the enumeration. This implies that, in total, we only have to consider n different matrices $\bar{Q}_d$, which we know in advance as soon as the fixing order is determined. (If the variables to be fixed were chosen freely, the number of such matrices could be exponential.)

Algorithm 1: Outline of the basic algorithm

```
Input: a strictly convex function $f\colon \mathbb{R}^n \to \mathbb{R},\ x \mapsto x^\top Qx + L^\top x + c$
Output: a vector $x \in \mathbb{Z}^n$ minimizing $f(x)$
determine a variable order $x_1, \ldots, x_n$;
let $\bar{Q}_d$ be the submatrix of $Q$ for rows and columns $d+1, \ldots, n$;
compute the inverse matrices $\bar{Q}_d^{-1}$ for $d = 1, \ldots, n$;
set $d := 0$, ub $:= \infty$;
while $d \geq 0$ do
    define $\bar{f} : \mathbb{R}^{n-d} \to \mathbb{R}$ by $\bar{f}(x) := f((r_1, \ldots, r_d, x_1, \ldots, x_{n-d}))$;
    compute $\bar{L}$ and $\bar{c}$ such that $\bar{f}(x) = x^\top \bar{Q}_d x + \bar{L}^\top x + \bar{c}$;
    // compute lower bound
    compute the continuous minimum $\bar{x} := -\frac{1}{2}(\bar{Q}_d^{-1}\bar{L}) \in \mathbb{R}^{n-d}$ of $\bar{f}$;
    set lb $:= \bar{f}(\bar{x})$;
    // compute upper bound
    set $r_j := \lfloor \bar{x}_{j-d} \rceil$ for $j = d+1, \ldots, n$ to form $r \in \mathbb{Z}^n$;
    apply primal heuristics to improve $r$;
    // update solution
    if $\bar{f}((r_{d+1}, \ldots, r_n)) <$ ub then
        set $r^* := r$;
        set ub $:= \bar{f}((r_{d+1}, \ldots, r_n))$;
    end
    // prepare next node
    if lb $<$ ub then
        // branch on variable $x_{d+1}$
        set $d := d + 1$;
        set $r_d := \lfloor \bar{x}_1 \rceil$;
    else
        // always holds if $d = n$
        // prune current node
        set $d := d - 1$;
        if $d > 0$ then
            // go to next node
            increment $r_d$ according to (2) or (3);
        end
    end
end
```

See Algorithm 1 for an outline of our method, for the case in which $X = \mathbb{R}^n$ and we simply use the continuous lower bound. Clearly, the running time of this algorithm is exponential in general. However, every node in the enumeration tree can be processed in $O(n^2)$ time (if the primal heuristics obey the same runtime bound), the bottleneck being the computation of the continuous minimum given the pre-computed inverse matrix $\bar{Q}_d^{-1}$. Note that Algorithm 1 can easily be adapted to the constrained case where $X \neq \mathbb{R}^n$. In this case, we just prune all nodes with invalid variable fixings.

For the computation of stronger lower bounds as explained in Section 2, at each node we consider the matrices $\bar{Q}'_0, \ldots, \bar{Q}'_n$ derived from $\bar{Q}_d$. It is crucial that the values $\lambda(\bar{Q}_d, \bar{Q}'_i)$ can be computed in the preprocessing phase for each depth d and for $i = 0, \ldots, n$. In the unconstrained case, the running time per node is then affected by the time needed to compute $\mu(\bar{Q}'_i, \bar{x})$. This requires $O(n)$ time for $i = 1, \ldots, n$ and an additional $O(n)$ time for $i = 0$, i.e., $O(n^2)$ time in total.

The same applies to the constrained case, where in each node we compute the stronger bounds given by Observations 6 and 7. For this, we determine all $t^i_{\min}$ and $t^i_{\max}$ in the preprocessing phase by calling the linear optimization oracle $2n^2$ times in total. After that, we only need two additional comparisons for each t^i in order to compute the improved bounds.

The above discussion is summarized in the following proposition.

Proposition 3. *The running time per node of the branch-and-bound algorithm in Figure 1 in which the lower bounds are improved as illustrated in Section 2 is $O(n^2)$.*

3.2 Improvement to Linear Time Per Node

With some adjustments, the running time to process a node of depth d in our algorithm can be decreased from $O(n^2)$ to $O(n-d)$. For this, we have to improve the running time of two different components: the computation of the continuous minima of $\bar{f}$, and the lower bound improvement by ellipsoids described in Section 2.

For the computation of the continuous minima, we can replace the $O((n-d)^2)$ method by an incremental technique needing $O(n-d)$ time only, which determines the new continuous minimum from the old one in linear time whenever a new variable is fixed. For this, we exploit the basic observation that in a given node, the continuous minima according to all possible fixings of the next variable lie on a line. Moreover, the direction of this line only depends on which variables have been fixed so far, but not on the values to which they were fixed. This implies that, in our algorithm, the direction of this line is fully determined by the depth of the current node. Additional care has to be taken for computing the objective function values of the continuous minima, since a direct evaluation of the objective function would take quadratic time.

Formally, recall that by $\bar{f}(x) = x^\top \bar{Q}_d x + \bar{L}^\top x + \bar{c}$ we denote the function obtained from f by fixing variable x_i to r_i for $i = 1, \ldots, d$, and let $\bar{x} \in \mathbb{R}^{n-d}$ be the continuous minimum of $\bar{f}$, noting that $\bar{x}_1$ corresponds to the value of the original variable x_{d+1}. Finally, let

$$z^d := (1, -(q_{d+1,d+2}, q_{d+1,d+3}, \ldots, q_{d+1,n})\, \bar{Q}^{-1}_{d+1})^\top \in \mathbb{R}^{n-d} .$$

Observation 8. *If we fix variable x_{d+1} to r_{d+1} and re-optimize $\bar{f}$, the resulting continuous minimum is given by $\bar{x} + (r_{d+1} - \bar{x}_1)z^d$.*

The above discussion implies that, in order to find the continuous optimum for the nodes generated by branching from a given node and the associated value, we simply have to compute $\bar{x}+\alpha z^d$ and $\bar{f}(\bar{x}+\alpha z^d)$ for a given $\alpha \in \mathbb{R}$. As to the latter, if we define

$$v^d := 2\bar{Q}_d z^d \in \mathbb{R}^{n-d}\ , \qquad s_d := (z^d)^\top \bar{Q}_d z^d \in \mathbb{R}\ ,$$

then we get

$$\bar{f}(\bar{x}+\alpha z^d) = \bar{f}(\bar{x}) + \alpha(\bar{x}^\top v^d + \bar{L}^\top z^d) + \alpha^2 s_d\ .$$

Since $\bar{L}$ can be computed incrementally in $O(n-d)$ time, we get:

Proposition 4. *If, in the preprocessing phase of the algorithm in Figure 1, we compute z^d, v^d, s_d as defined above for $d = 0,\ldots,n-1$, then the computation of the continuous minimum and the associated lower bound can be carried out in $O(n-d)$ time per node.*

When improving lower bounds by ellipsoids as illustrated in Section 2, the following natural restriction leads to linear time per node: if the splits in the root node are defined by the columns of the transformation matrix T, then the splits on level d are defined by the columns of the matrix $\bar{T}_d$ arising from T by deleting the first d rows and columns. Indeed, in this case, for the continuous minimum $\bar{\bar{x}} = \bar{x} + (r_{d+1} - \bar{x}_1)z^d$ obtained after having fixed variable x_{d+1} to r_{d+1}, we have to compute $\bar{T}_{d+1}^\top(\bar{\bar{x}}_2,\ldots,\bar{\bar{x}}_{n-d})^\top$ (recall the notation in Section 1.4) in order to determine the scaling factors $\mu(\bar{Q}_i', (\bar{\bar{x}}_2,\ldots,\bar{\bar{x}}_{n-d})^\top)$. If we define

$$w^{d+1} := \bar{T}_{d+1}^\top (z_2^d,\ldots,z_{n-d}^d)^\top \in \mathbb{R}^{n-d-1}\ ,$$

we have

$$\begin{aligned}\bar{T}_{d+1}^\top(\bar{\bar{x}}_2,\ldots,\bar{\bar{x}}_{n-d})^\top &= \bar{T}_{d+1}^\top(\bar{x}_2,\ldots,\bar{x}_{n-d})^\top + \bar{T}_{d+1}^\top(r_{d+1}-\bar{x}_1)(z_2^d,\ldots,z_{n-d}^d)^\top \\ &= ((\bar{T}_d^\top\bar{x})_2,\ldots,(\bar{T}_d^\top\bar{x})_{n-d}) - \bar{x}_1(t_{d+1,d+2},\ldots,t_{d+1,n})^\top + (r_{d+1}-\bar{x}_1)w^{d+1}\ .\end{aligned}$$

Now $\bar{T}_d^\top\bar{x}$ has already been determined, hence each of the $n-d$ factors $\mu(\bar{Q}_i', (\bar{\bar{x}}_2,\ldots,\bar{\bar{x}}_{n-d})^\top)$ can be computed in constant extra time. After that, the last factor $\mu(\bar{Q}_0', (\bar{\bar{x}}_2,\ldots,\bar{\bar{x}}_{n-d})^\top)$ can be computed in $O(n-d)$ time by Observation 5. In the constrained case, we can determine improved bounds as explained in Section 3.1. In summary, we thus have

Proposition 5. *If, in the preprocessing phase of the algorithm in Figure 1, we compute w^{d+1} and all $t^i_{\min}$ and $t^i_{\max}$ for $d = 0,\ldots,n-1$, then the lower bound improvement as illustrated in Section 2 can be carried out in $O(n-d)$ time per node.*

4 Computational Results

In this section, we present experimental results for the two special cases of CQIP mentioned in the introduction, namely the cases $X = [-1,1]^n$ (Section 4.1) and $X = \mathbb{R}^n$ (Section 4.2). In all cases, we compare our algorithm with the CPLEX MIQP solver [6], which, as mentioned in the introduction, turned out to be by far the best method among those available when we approached the problem. For our algorithm, we also state results obtained without the lower bound improvements discussed in Section 2.

We implemented the versions of our algorithm running in quadratic and linear time per node in C. All experiments were run on Intel Xeon processors running at 2.33 GHz. For basis reduction, we used the LLL-algorithm [7] implemented in LiDIA [8]. The runtime limit for all instances and solution methods was 8 hours. Besides total running times, we investigated the time needed for preprocessing and the time per node in the enumeration tree. Moreover, we state the total number of nodes processed.

4.1 Convex Quadratic Ternary Optimization

In Table 1, we present the experimental results for instances corresponding to second order Butterworth filters, for a sinusoidal target signal. In the first column, we state the parameters of the problem instances: the amplitude of the signal (γ), the cut-off frequency (σ), and the number of pulses per period, which agrees with the number of variables n. We compare the $O(n-d)$ time/node and the $O(n^2)$ time/node versions of our algorithm with CPLEX MIQP. The

Table 1. Experimental results for second order Butterworth filter instances

instance			Algorithm 1, $O(n-d)$			Algorithm 1, $O(n^2)$			CPLEX MIQP		
γ	σ	n	tt/s	pt/s	nodes	tt/s	pt/s	nodes	tt/s	nodes	optimum
0.2	2.25	30	0.04	0.04	1.85e+03	0.05	0.04	1.84e+03	1.13	1.20e+04	6.657892e-04
0.2	2.25	40	0.11	0.09	1.69e+04	0.31	0.15	1.68e+04	61.71	5.14e+05	2.703067e-04
0.2	2.25	50	0.24	0.21	3.00e+04	0.88	0.41	3.00e+04	20995.52	1.20e+08	1.088225e-04
0.2	2.25	60	0.77	0.43	2.62e+05	6.24	0.90	2.62e+05	—	—	5.987001e-05
0.2	2.25	70	2.49	0.82	1.09e+06	32.96	1.91	1.09e+06	—	—	3.811803e-05
0.2	2.25	80	2.38	1.46	5.05e+05	25.91	3.64	5.05e+05	—	—	1.863941e-05
0.2	2.25	90	22.72	2.45	1.09e+07	523.11	6.41	1.09e+07	—	—	1.398550e-05
0.2	2.25	100	104.58	3.93	5.12e+07	2958.38	10.68	5.12e+07	—	—	9.677417e-06
0.2	2.25	110	1039.53	6.05	4.99e+08	—	—	—	—	—	7.226546e-06
0.2	2.25	120	7815.37	9.04	3.58e+09	—	—	—	—	—	5.332916e-06
0.3	2.25	30	0.03	0.03	2.50e+03	0.06	0.04	2.49e+03	1.83	1.86e+04	7.691583e-04
0.3	2.25	40	0.12	0.08	3.89e+04	0.52	0.15	3.89e+04	211.93	1.79e+06	3.624281e-04
0.3	2.25	50	0.32	0.21	9.36e+04	1.78	0.40	9.36e+04	—	—	1.453055e-04
0.3	2.25	60	1.53	0.43	8.75e+05	16.94	0.90	8.75e+05	—	—	7.434848e-05
0.3	2.25	70	2.83	0.82	1.34e+06	39.34	1.91	1.34e+06	—	—	4.060845e-05
0.3	2.25	80	11.10	1.45	5.87e+06	213.88	3.66	5.87e+06	—	—	2.452317e-05
0.3	2.25	90	28.07	2.45	1.41e+07	688.08	6.41	1.41e+07	—	—	1.513070e-05
0.3	2.25	100	78.37	3.92	3.84e+07	2135.50	10.69	3.84e+07	—	—	9.448114e-06
0.3	2.25	110	322.24	6.04	1.46e+08	10425.21	16.85	1.46e+08	—	—	6.957344e-06
0.3	2.25	120	10814.91	9.03	4.99e+09	—	—	—	—	—	5.685910e-06
0.4	2.25	30	0.03	0.03	2.05e+03	0.06	0.04	2.04e+03	1.22	1.29e+04	8.177199e-04
0.4	2.25	40	0.09	0.08	1.06e+04	0.26	0.15	1.06e+04	82.13	6.80e+05	3.136527e-04
0.4	2.25	50	0.29	0.20	7.70e+04	1.53	0.40	7.70e+04	—	—	1.518196e-04
0.4	2.25	60	0.73	0.43	2.15e+05	5.49	0.90	2.15e+05	—	—	7.385543e-05
0.4	2.25	70	3.57	0.82	1.88e+06	51.64	1.92	1.88e+06	—	—	4.473205e-05
0.4	2.25	80	1.67	1.46	1.04e+05	8.60	3.65	1.04e+05	—	—	1.776351e-05
0.4	2.25	90	20.82	2.45	9.98e+06	520.05	6.85	9.98e+06	—	—	1.587669e-05
0.4	2.25	100	244.66	3.94	1.24e+08	6615.51	11.62	1.24e+08	—	—	1.167412e-05
0.4	2.25	110	804.77	6.06	3.86e+08	24945.14	16.85	3.86e+08	—	—	7.950806e-06
0.4	2.25	120	8702.36	9.11	4.08e+09	—	—	—	—	—	5.842935e-06

Table 2. Experimental results for CVP instances

	Algorithm 1, $O(n-d)$					CPLEX MIQP			
n	#	tt/s	pt/s	nt/μs	nodes	#	tt/s	nt/μs	nodes
20	10	0.01	0.01	0.00	3.26e+02	10	0.08	87.10	8.66e+02
25	10	0.02	0.02	0.00	1.96e+03	10	0.40	93.80	4.40e+03
30	10	0.04	0.03	1.16	1.07e+04	10	2.54	101.75	2.51e+04
35	10	0.14	0.05	1.22	7.39e+04	10	17.72	117.49	1.50e+05
40	10	0.75	0.09	1.31	4.97e+05	10	176.80	138.89	1.26e+06
45	10	6.46	0.14	1.51	4.21e+06	10	1759.09	167.00	1.03e+07
50	10	40.45	0.21	1.59	2.51e+07	10	10875.65	202.87	5.18e+07
55	10	144.87	0.32	1.74	8.39e+07	3	15379.42	235.16	6.52e+07
60	10	2936.50	0.45	2.00	1.45e+09	0	—	—	—
65	10	6935.54	0.63	2.18	3.17e+09	0	—	—	—
70	5	15065.94	0.86	2.33	6.45e+09	0	—	—	—

columns marked "tt/s" contain the total time in seconds needed to solve the instance to optimality, while "pt/s" states the time in seconds needed for the preprocessing. Finally, "nodes" contains the total number of nodes processed in the enumeration tree. The last column contains the value of the optimal solution.

As is obvious from Table 1, our algorithm outperforms CPLEX by several orders of magnitude. CPLEX could not solve instances with more than 50 variables within the time limit of 8 hours, while the linear-time version of our algorithm solves all instances up to 120 variables. As expected, quadratic running time per node leads to much slower total times. However, even the quadratic version clearly outperforms CPLEX.

4.2 Closest Vector Problem

As anticipated in Section 1.2, we are not aware of computational contributions to the (optimal) solution of the CVP. Thus, even for the practical applications arising in cryptography, we could not find any public library of test problems. Therefore, in this section, we present results for random instances of the problem whose generation is illustrated in the full paper. Given that all methods widely benefit from this, we apply basis reduction to these instances before solving them.

The results are presented in Table 2, where 10 random instances have been considered for each n. The first column for every solution method shows the number of instances solved to optimality, the remaining figures are averages over all solved instances. Again it turns out that (the linear-time version of) our algorithm is much faster than CPLEX, even if the difference is smaller than in the ternary case.

Acknowledgments. This work was partially supported by the University of Bologna within the OpIMA project. The first author is supported by the DFG under contract BU 2313/1-1.

References

1. Bonami, P., Biegler, L.T., Conn, A.R., Cornuéjols, G., Grossmann, I.E., Laird, C.D., Lee, J., Lodi, A., Margot, F., Sawaya, N., Wächter, A.: An algorithmic framework for convex mixed integer nonlinear programs. Discrete Optimization 5, 186–204 (2008)
2. Callegari, S., Bizzarri, F., Rovatti, R., Setti, G.: Discrete quadratic programming problems by $\Delta\Sigma$ modulation: the case of circulant quadratic forms. Technical report, Arces, University of Bologna (2009)
3. Eisenbrand, F.: Integer programming and algorithmic geometry of numbers. In: Jünger, M., Liebling, T., Naddef, D., Nemhauser, G., Pulleyblank, W., Reinelt, G., Rinaldi, G., Wolsey, L.A. (eds.) 50 Years of Integer Programming 1958-2008. The Early Years and State-of-the-Art Surveys. Springer, Heidelberg (2009)
4. Frangioni, A., Lodi, A., Rinaldi, G.: Optimizing over semimetric polytopes. In: Bienstock, D., Nemhauser, G.L. (eds.) IPCO 2004. LNCS, vol. 3064, pp. 431–443. Springer, Heidelberg (2004)
5. Hemmecke, R., Köppe, M., Lee, J., Weismantel, R.: Nonlinear integer programming. In: Jünger, M., Liebling, T., Naddef, D., Nemhauser, G., Pulleyblank, W., Reinelt, G., Rinaldi, G., Wolsey, L.A. (eds.) 50 Years of Integer Programming 1958-2008. The Early Years and State-of-the-Art Surveys. Springer, Heidelberg (2009)
6. ILOG, Inc. ILOG CPLEX 12.1 (2009), http://www.ilog.com/products/cplex
7. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261, 515–534 (1982)
8. LiDIA. LiDIA: A C++ Library For Computational Number Theory (2006), http://www.cdc.informatik.tu-darmstadt.de/TI/LiDIA/
9. Lodi, A.: MIP computation and beyond. In: Jünger, M., Liebling, T., Naddef, D., Nemhauser, G., Pulleyblank, W., Reinelt, G., Rinaldi, G., Wolsey, L.A. (eds.) 50 Years of Integer Programming 1958-2008. The Early Years and State-of-the-Art Surveys. Springer, Heidelberg (2009)
10. Micciancio, D., Goldwasser, S.: Complexity of Lattice Problems: A Cryptographic Perspective. Springer, Heidelberg (2002)
11. Moré, J.J., Toraldo, G.: On the solution of large quadratic programming problems with bound constraints. SIAM Journal on Optimization 1, 93–113 (1991)
12. Rajan, D., Dash, S., Lodi, A.: $\{-1,0,1\}$ unconstrained quadratic programs using max-flow based relaxations. Technical Report OR/05/13, DEIS, University of Bologna (2005)
13. De Simone, C.: The cut polytope and the boolean quadric polytope. Discrete Mathematics 79, 71–75 (1989)

Extending SDP Integrality Gaps to Sherali-Adams with Applications to Quadratic Programming and MaxCutGain

Siavosh Benabbas and Avner Magen

Department of Computer Science
University of Toronto
{siavosh,avner}@cs.toronto.edu

Abstract. We show how under certain conditions one can extend constructions of integrality gaps for semidefinite relaxations into ones that hold for stronger systems: those SDP to which the so-called k-level constraints of the Sherali-Adams hierarchy are added. The value of k above depends on properties of the problem. We present two applications, to the Quadratic Programming problem and to the MaxCutGain problem.

Our technique is inspired by a paper of Raghavendra and Steurer [Raghavendra and Steurer, FOCS 09] and our result gives a doubly exponential improvement for Quadratic Programming on another result by the same authors [Raghavendra and Steurer, FOCS 09]. They provide tight integrality-gap for the system above which is valid up to $k = (\log\log n)^{\Omega(1)}$ whereas we give such a gap for up to $k = n^{\Omega(1)}$.

1 Introduction

A powerful tool in obtaining approximation algorithms for NP-hard problems is semidefinite programming. Here one formulates a problem as a quadratic program and then "relaxes" it to a semidefinite program. The interesting quantity one tries to minimize is then the *integrality gap* defined as the ratio of the objective value in the relaxation compared to the objective value of the original problem. When this integrality gap is too big (the relaxation is not tight enough) one often considers various strengthenings and hopes that they provide a better integrality gap.

Lift-and-Project systems are systematic ways to produce stronger and stronger relaxations for a problem from a canonical LP or SDP relaxation. These systems are equipped with a parameter called "level" which tunes the strength of the relaxation. As the level increases the relaxation becomes stronger while its size (and time required to solve it) increases. At one extreme the zeroth level is the original relaxation and at the other extreme the nth level is an exact formulation of the problem but has exponential size. In between one gets a spectrum of stronger and stronger relaxations the lth of which can be solved in time $n^{\Theta(l)}$. See [8, 9, 15] for a more in-depth discussion about this methodology. Recently, there has been a growing interest in understanding the exact interplay between the quality of such relaxations and the level parameter. See [3, 5, 10, 13, 14, 16, 17] for a few examples.

Our main problem of interest will be QUADRATIC PROGRAMMING as defined by [4]. Here the input is a matrix $A_{n\times n}$ and the objective is to find $x \in \{-1,1\}^n$ that maximizes the quadratic form $\sum_{i\neq j} a_{ij}x_i x_j$. The natural semidefinite programming relaxation of this problem replaces the integer (± 1) valued x_i's with unit vectors, and products with inner products. This problem has applications in correlation clustering and its relaxation is related to the well-known Grothendieck inequality[6] (see [1].) Charikar and Wirth [4] show that the integrality gap of this relaxation is at most $O(\log n)$ and a matching lower bound was later established by [1, 2, 7]. To lowerbound the integrality gap Khot and O'Donnell [7] construct a solution for the SDP relaxation that has objective value $\Theta(\log n)$ times bigger than any integral (± 1) solution.

It is then natural to ask if strengthening the SDP with a lift and project method will reduce this $\Theta(\log n)$ gap resulting in a better approximation algorithm. We investigate the performance of one of the stronger Lift-and-Project systems, the so called Sherali-Adams system for this problem. We show that for Quadratic Programing the integrality gap does not asymptotically decrease even after level n^δ of the Sherali-Adams hierarchy for some constant δ. The technique used is somewhat general and can be applied to other problems, and as an example we apply it to the related problem of MAXCUTGAIN. We can prove the following two theorems.

Theorem 1. *The standard SDP formulation of* QUADRATIC PROGRAMMING *admits a $\Theta(\log N)$ integrality gap even when strengthened with $N^{\Omega(1)}$ levels of the Sherali-Adams hierarchy.*

Theorem 2. *The standard SDP formulation of* MAXCUTGAIN *has integrality gap $\omega(1)$ even when strengthened with $\omega(1)$ levels of the Sherali-Adams hierarchy.*

It should be mentioned that it is known [2, 7] that assuming the Unique Games Conjecture no polynomial time algorithm can improve upon the $\Theta(\log n)$ factor for QUADRATIC PROGRAMMING or $\omega(1)$ for MAXCUTGAIN. However, our results do not rely on any such assumptions. Also, given that the Sherali-Adams strengthening can only be solved in polynomial time for constant levels, our results preclude an algorithm based on the standard relaxation with Sherali-Adams strengthening that runs in time 2^{n^δ} for some small enough constant δ.

Here is an outline of our approach. We start with a known integrality gap instance for the SDP relaxation and its SDP solution A. Now consider an integral solution B to the same instance (obviously B is a solution to an SDP strengthened with an arbitrary level of the Sherali-Adams hierarchy.) We combine A and B into a candidate solution C that has some of the good qualities of both, namely it has high objective value (like A) and has almost the same behavior with regards to the Sherali-Adams hierarchy as B. In particular, the only conditions of the Sherali-Adams hierarchy C violates are some positivity conditions, and those violations can be bounded by some quantity ξ. To handle these violations we take a convex combination of C with a solution in which these violated conditions are satisfied with substantial enough slack. In fact a uniform distribution over all integral points (considered as a solution) has this property. The weights we need to give to C and the uniform solution in the convex combination will depend on ξ and the amount of slack in the uniform solution which are in turn a function of the level parameter k. In turn these two weights determine the objective value of the resulting solution.

This idea of "smoothening" a solution like the above to satisfy Sherali-Adams constraints is due to Raghavendra and Steurer[12]. They use this idea together with dimensionality reduction to give a rounding algorithm that is optimal for a large class of constraint satisfaction problems. In another relevant new result [13] the same authors show how to get integrality gaps for the Sherali-Adams SDPs similar to our work. However, in [13] the main vehicle in getting the results are reductions from the Unique Games Problem and "smoothening" is used as a small step.

It is interesting to compare our results with [13]. While [13] gives a more general result in that it applies to a broader spectrum of problems, it only holds to a relatively low level of the Sherali-Adams hierarchy. In particular, for Quadratic Programming their result is valid for Sherali-Adams SDPs of level up to $(\log\log n)^{\Omega(1)}$ whereas ours is valid up to level $n^{\Omega(1)}$. We note that the two results exhibit different level/integrality gap tradeoffs as well, in that [13] provides the same integrality gap asymptotically until the "critical level" in which point it breaks down completely. Our results supplies a more smooth tradeoff with the integrality gap dropping "continuously" as the level increases. An additional difference is that our result (which does not use Unique Games reductions) is elementary.

The rest of the paper is organized as follows. In Section 2 we are going to introduce our notation and some basic definitions and identities from Fourier analysis on the cube. In the same section we will review the relevant definition of the Sherali-Adams hierarchy. In section 3 we are going to state and prove our main technical lemma. Section 4 presents an application of the lemma to Quadratic Programming. There we prove an integrality gap for the SDP relaxation of this problem strengthened to the n^δth level of the Sherali-Adams hierarchy (for some $\delta > 0$.) In Section 5 we present our application to the MaxCutGain problem. In particular we present super-constant integrality gaps for some super-constant level of the Sherali-Adams SDP hierarchy. We conclude in section 6 by pointing out some of the limitations of our approach and some open problems.

2 Preliminaries

Notation and Fourier Analysis on the cube

We denote by $[n]$ the set $\{1,\ldots,n\}$ and by $\binom{[n]}{k}$ and $\binom{[n]}{\leq k}$ the set of subsets of $[n]$ of size exactly k and at most k respectively. For a distribution μ on some finite space D we denote by $\mu(x)$ the probability of choosing $x \in D$ according to μ. Note that this allows us to think of distributions as real functions with domain D.

Consider the set of real functions with domain $\{-1,1\}^n$ as a linear space of dimension 2^n. It is well known that under the inner product $\langle f,g\rangle \overset{\text{def}}{=} \mathbb{E}_x\left[f(x)g(x)\right]$ the functions $\{\chi_S\}_{S\subseteq[n]}$ defined as $\chi_S(x) \overset{\text{def}}{=} \prod_{i\in S} x_i$ form an orthonormal basis called the *Fourier basis* for this space. In particular, any function f has a unique *Fourier expansion*:

$$f = \sum_S \widehat{f}(S)\chi_S, \qquad\qquad \widehat{f}(S) = \langle f, \chi_S\rangle.$$

We call $\widehat{f}(S)$'s the *Fourier coefficients* of f. An immediate corollary is Parseval's identity,

$$\mathop{\mathbb{E}}_x\left[f(x)g(x)\right] = \langle f,g\rangle = \sum_S \widehat{f}(S)\widehat{g}(S).$$

For a distribution μ on $\{-1,1\}^n$ and for $S\subseteq[n]$ we let $\operatorname{Mar}_S \mu$ be the marginal distribution of μ on the set S. In particular, for $y\in\{-1,1\}^S$ we have $(\operatorname{Mar}_S\mu)(y) = \Pr_{z\sim\mu}\left[z_S = y_S\right]$. Alternatively,

$$(\operatorname{Mar}_S\mu)(y) = \sum_{z\in\{-1,1\}^{[n]\setminus S}} \mu(y\circ z),$$

where $\circ$ denotes the concatenation of two vectors on disjoint set of coordinates in the natural way. In fact, for any *function* $f:\{-1,1\}^n\to\mathbb{R}$ we define $(\operatorname{Mar}_S f) : \{-1,1\}^S \to \mathbb{R}$ like above. One can easily express the Fourier coefficients of $\operatorname{Mar}_S\mu$ in terms of those of μ,

$$\begin{aligned}
\widehat{\operatorname{Mar}_S\mu}(U) &= \mathop{\mathbb{E}}_{x\in\{-1,1\}^S}\left[(\operatorname{Mar}_S\mu)(x)\chi_U(x)\right]\\
&= \mathop{\mathbb{E}}_{x\in\{-1,1\}^S}\left[\sum_{z\in\{-1,1\}^{[n]\setminus S}} \mu(x\circ z)\chi_U(x)\right]\\
&= 2^{-|S|}\sum_{x\in\{-1,1\}^S}\ \sum_{z\in\{-1,1\}^{[n]\setminus S}} \mu(x\circ z)\chi_U(x)\\
&= 2^{-|S|}\sum_{y\in\{-1,1\}^{[n]}} \mu(y)\chi_U(y) = 2^{n-|S|}\widehat{\mu}(U). \qquad (1)
\end{aligned}$$

In particular, for all $x\in\{-1,1\}^S$ we have that

$$(\operatorname{Mar}_S\mu)(x) = \sum_{U\subseteq S}\widehat{\operatorname{Mar}_S\mu}(U)\chi_U(x) = 2^{n-|S|}\sum_{U\subseteq S}\widehat{\mu}(U)\chi_U(x). \qquad (2)$$

We will use $\mathbb{S}^{d-1}$ for the $(d-1)$-dimensional unit sphere, i.e. , the set of unit vectors in $\mathbb{R}^d$. Throughout the paper we will use bold letters for real vectors, and $\langle\mathbf{v},\mathbf{u}\rangle$ for the inner product of two vectors $\mathbf{v},\mathbf{u}\in\mathbb{R}^d$.

The Sherali-Adams Hierarchy

The Sherali-Adams hierarchy[15] is a Lift and Project method to strengthen a canonical LP or SDP formulation of a problem. In this method one starts with an exact formulation of a problem as an integer program and relaxes it to a Linear Program or Semidefinite Program, which we call the canonical formulation. One then strengthens the canonical formulation by adding extra variables and constraints valid for the integer solutions in a specific way. The method has a parameter, level, to choose the strength of the resulting relaxation, and one can think of the resulting relaxations (for different level parameters) as making a hierarchy, each stronger than the previous. The weakest relaxation is then the canonical formulation for level $k = 0$, and the strongest is a perfect formulation of the problem (albeit of exponential size) for $k = n$, where n is the number of variables in the canonical formulation. It is known that one can solve the strengthened relaxation of level k in running time $n^{O(k)}$. We use the term Sherali-Adams SDP of level k to identify the Sherali-Adams strengthening of level k of the canonical SDP formulation of a problem.

In this work we concentrate on SDPs for combinatorial problems that have no constraints. In particular, consider an optimization problem of the form

$$\begin{aligned} \max \quad & c(x_1, \ldots, x_n) & \\ \text{Subject to} \quad & x_i \in \{-1, 1\} & \text{for all } i. \end{aligned}$$

where c is a quadratic polynomial. For such a problem the definition of the hierarchy is simplified using the following definition.

Definition [Compatibility]. Let $k \leq n$ be integers. We say that vectors $\mathbf{u}_0, \ldots, \mathbf{u}_n$ and a family of distributions $\{\mu_S\}_{S \subset [n], |S| \leq k}$ where each μ_S is a distribution on $\{-1, 1\}^S$ are compatible if the distributions are consistent on marginals and the inner products of the vectors agree with appropriate biases and correlations as follows

$$\forall i \in [n] \qquad \langle \mathbf{u}_0, \mathbf{u}_i \rangle = \mathop{\mathbb{E}}_{x \sim \mu_{\{i\}}} [x_i], \tag{P1}$$

$$\forall i \neq j \in [n] \qquad \langle \mathbf{u}_i, \mathbf{u}_j \rangle = \mathop{\mathbb{E}}_{x \sim \mu_{\{i,j\}}} [x_i x_j], \tag{P2}$$

$$\forall T \subseteq S \subset [n], |S| \leq k \qquad \mu_T = \mathrm{Mar}_T\, \mu_S. \tag{P3}$$

Fact 1. *A set of vectors is a solution to a Sherali-Adams SDP of level k of such a problem if and only if they are compatible with some family of distribution (on subsets of size up to k) as above.*

In particular, for Quadratic Programming, MaxCutGain and constraint satisfaction problems where we do not have any hard constraints on the variables we can just focus on the definition of compatibly as above. We do not supply the equivalence proof for Fact 1 in this extended abstract, but instead refer the reader to a similar concept laid forth by [3, 17].)

The goal of the present work is to show that (for particular problems), even for large values of k, the Sherali-Adams SDP of level k is not much stronger than the canonical relaxation. In particular, its has asymptotically the same integrality gap.

3 Extending Vector Solutions to Sherali-Adams Hierarchy

In this section we provide the general framework that suggests how vector (or SDP) solutions can be extended to Sherali-Adams solutions of comparable objective value. This framework is captured by the following lemma.

Lemma 2. *Let $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_n \in \mathbb{S}^{n-1}$ be unit vectors, ν be a distribution on $\{-1,1\}^n$, k be a positive integer and $\epsilon \in (0, 1/2]$ be small enough to satisfy,*

$$\text{for all } i \in [n] \qquad 2\epsilon k^2 |\langle \mathbf{v}_0, \mathbf{v}_i \rangle - \mathop{\mathbb{E}}_{x\sim\nu}[x_i]| \le 1,$$
$$\text{for all } i \neq j \in [n] \qquad 2\epsilon k^2 |\langle \mathbf{v}_i, \mathbf{v}_j \rangle - \mathop{\mathbb{E}}_{x\sim\nu}[x_i x_j]| \le 1.$$

Then there exist unit vectors $\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_n \in \mathbb{S}^{n-1}$ and a family of distributions $\{\mu_S\}_{S\subset[n],|S|\le k}$ such that $\mathbf{u}_i$'s are compatible with $\{\mu_S\}$ as defined in Section 2 i.e., they are a valid solution for Sherali-Adams SDP of level k. Furthermore, the inner products of $\mathbf{u}_i$'s are related to those of $\mathbf{v}_i$'s as follows

$$\text{for all } i \in [n] \qquad \langle \mathbf{u}_0, \mathbf{u}_i \rangle = \epsilon \langle \mathbf{v}_0, \mathbf{v}_i \rangle, \tag{3}$$
$$\text{for all } i \neq j \in [n] \qquad \langle \mathbf{u}_i, \mathbf{u}_j \rangle = \epsilon \langle \mathbf{v}_i, \mathbf{v}_j \rangle. \tag{4}$$

Before we prove the lemma we will briefly describe its use. For simplicity we will only describe the case where ν is the uniform distribution on $\{-1,1\}^n$. To use the lemma one starts with an integrality gap instance for the canonical SDP relaxation of the problem of interest (say Quadratic Programming) and its vector solution $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_n$. Given that the instance is an integrality gap instance one knows that the objective value attained by $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_n$ is much bigger than what is attainable by any integral solution. The simplest thing one can hope for is to show that this vector solution is also a solution to the Sherali-Adams SDP of level k, or in the language of Fact 1 the vectors are compatible with a set of distributions $\{\mu_S\}_{S\subset[n],|S|\le k}$. However, this is not generally possible. Instead we use the lemma (in the simplest case with ν being the uniform distribution on $\{-1,1\}^n$) to get another set of vectors $\mathbf{u}_0, \ldots, \mathbf{u}_n$ which are in fact compatible with some $\{\mu_S\}_{S\subset[n],|S|\le k}$. Given that in the problems we consider the objective function is a quadratic polynomial, and given the promise of the lemma that inner products of $\mathbf{u}_i$'s is ϵ times that of $\mathbf{v}_i$'s, it follows that the objective value attained by $\mathbf{u}_i$'s is ϵ times that attained by $\mathbf{v}_i$'s. As $\mathbf{v}_i$'s are the vector solution for an integrality gap instance, it follows that the integrality gap will decrease by a multiplicative factor of at most ϵ when one goes from the canonical SDP relaxation to the Sherali-Adams SDP of level k.

How big one can take ϵ (satisfying the requirements of the lemma) will then determine the quality of the resulting integrality gap. In the simplest case one can take ν to be the uniform distribution on $\{-1,1\}^n$ and argue that in this case the requirements of the lemma are satisfied as long as $2\epsilon k^2 \leq 1$ and in particular for $\epsilon = 1/2k^2$. In fact our application to the MaxCutGain problem detailed in section 5 follows this simple outline. For Quadratic Programming we can get a better result by taking a close look at the particular structure of $\mathbf{v}_i$'s for the integrality gap instance of [7] and using a more appropriate distribution for ν.

We will now prove Lemma 2.

Proof (of Lemma 2). Our proof proceeds in two steps. First, we construct a family of functions $\{f_S : \{-1,1\}^S \to \mathbb{R}\}_{S\subset[n],|S|\leq k}$ that satisfy all the required conditions except being the probability mass function of distributions. In particular, while for any S the sum of the values of f_S is 1, this function can take negative values for some inputs. The construction of f_S uses the distribution ν and guarantees that while f_S may take negative values, these values are not too small. In the second step we take a convex combination of the f_S's with the uniform distribution on $\{-1,1\}^S$ to get the desired family of distributions $\{\mu_S\}_{|S|\leq k}$.

For any subset $S \in \binom{[n]}{\leq k}$ we define f_S as a "hybrid" object that is using both the vectors $\{\mathbf{v}_i\}$'s and the distribution ν. Given that a function is uniquely determined by its Fourier expansion we will define f_S in terms of its Fourier expansion,

$$\begin{aligned}
& \widehat{f_S}(\emptyset) = 2^{-|S|} = 2^{n-|S|}\widehat{\nu}(\emptyset), \\
\forall i \in S \qquad & \widehat{f_S}(\{i\}) = 2^{-|S|}\langle \mathbf{v}_0, \mathbf{v}_i\rangle, \\
\forall i \neq j \in S \qquad & \widehat{f_S}(\{i,j\}) = 2^{-|S|}\langle \mathbf{v}_i, \mathbf{v}_j\rangle, \\
\forall T \subseteq S, |T| > 2 \qquad & \widehat{f_S}(T) = 2^{n-|S|}\widehat{\nu}(T).
\end{aligned}$$

Comparing the above definition with (1), f_S is exactly like the marginal distribution of ν on the set S except it has different degree one and degree two Fourier coefficients. First, observe that for any S, the sum of the values of f_S is 1.

$$\sum_{x\in\{-1,1\}^S} f_S(x) = 2^{|S|}\mathop{\mathbb{E}}_x[f_S(x)] = 2^{|S|}\widehat{f_S}(\emptyset) = 1.$$

Then observe that by (1), for all $U \subseteq T \subset S$,

$$\widehat{\mathrm{Mar}_T\, f_S}(U) = 2^{|S|-|T|}\widehat{f_S}(U) = \widehat{f_T}(U).$$

So, f_S satisfies (P3). Now observe that,

$$\sum_{x\in\{-1,1\}^S} f_S(x)x_i = 2^{|S|}\mathop{\mathbb{E}}_x\left[f_S(x)\chi_{\{i\}}(x)\right] = 2^{|S|}\widehat{f_S}(\{i\}) = \langle \mathbf{v}_0, \mathbf{v}_i\rangle,$$

$$\sum_{x\in\{-1,1\}^S} f_S(x)x_ix_j = 2^{|S|}\mathop{\mathbb{E}}_x\left[f_S(x)\chi_{\{i,j\}}(x)\right] = 2^{|S|}\widehat{f_S}(\{i,j\}) = \langle \mathbf{v}_i, \mathbf{v}_j\rangle.$$

So, f_S's satisfy (P1) and (P2) and are compatible with $\mathbf{v}_i$'s (except, they are not distributions.[1]) Next we show that $f_S(y)$ cannot be too negative.

$$\begin{aligned}
f_S(y) &= \sum_{T\subseteq S} \widehat{f_S}(T)\chi_T(y)\\
&= 2^{n-|S|}\sum_{T\subseteq S}\widehat{\nu}(T)\chi_T(y) + \sum_{T\subseteq S}(\widehat{f_S}(T) - 2^{n-|S|}\widehat{\nu}(T))\chi_T(y)\\
&= (\mathrm{Mar}_S\,\nu)(y) + \sum_{T\subseteq S}(\widehat{f_S}(T) - 2^{n-|S|}\widehat{\nu}(T))\chi_T(y) && \text{by (2)}\\
&\geq -\sum_{T\subseteq S}\left|\widehat{f_S}(T) - 2^{n-|S|}\widehat{\nu}(T)\right|\\
&= -\sum_{i\in S}\left|\widehat{f_S}(\{i\}) - 2^{n-|S|}\widehat{\nu}(\{i\})\right|\\
&\quad -\sum_{i\neq j\in S}\left|\widehat{f_S}(\{i,j\}) - 2^{n-|S|}\widehat{\nu}(\{i,j\})\right|\\
&= -2^{-|S|}\left(\sum_{i\in S}\left|\langle\mathbf{v}_0,\mathbf{v}_i\rangle - \mathop{\mathbb{E}}_{x\sim\nu}[x_i]\right| + \sum_{i\neq j\in S}\left|\langle\mathbf{v}_i,\mathbf{v}_j\rangle - \mathop{\mathbb{E}}_{x\sim\nu}[x_ix_j]\right|\right)\\
&\geq -2^{-|S|}/2\epsilon,
\end{aligned}$$

where the last step follows from the condition on ϵ and because $|S|\leq k$. This completes the first step of the proof.

Next, define π to be the uniform distribution on $\{-1,1\}^n$ and μ_S as a convex combination of f_S and $\mathrm{Mar}_S\,\pi$, i.e.,

$$\forall y\in\{-1,1\}^S \qquad \mu_S(y) = \epsilon f_S(y) + (1-\epsilon)(\mathrm{Mar}_S\,\pi)(y).$$
$$\mathbf{u}_i = \sqrt{\epsilon}\cdot\mathbf{v}_i + \sqrt{1-\epsilon}\cdot\mathbf{w}_i.$$

Here, $\mathbf{w}_i$'s are defined such that they are perpendicular to all $\mathbf{v}_i$'s and each other.

It is easy to check that $\mathbf{u}_i$'s are compatible with μ_S's and satisfy all the required properties of the lemma (except that μ_S's can potentially be negative.) In fact (P1-P3) are linear and given that they hold for $\{f_S\}$ and $\{\mathbf{v}_i\}$ and for $\{\mathrm{Mar}_S\,\pi\}$ and $\{\mathbf{w}_i\}$ they must hold for $\{\mu_S\}$ and $\{\mathbf{u}_i\}$. Finally, for any $S\in\binom{[n]}{\leq k}$ and $y\in\{-1,1\}^S$ we have

$$\begin{aligned}
\mu_S(y) = \epsilon f_S(y) + (1-\epsilon)(\mathrm{Mar}_S\,\pi)(y) &\geq -\epsilon 2^{-|S|}/2\epsilon + (1-\epsilon)2^{-|S|}\\
&= 2^{-|S|-1} - \epsilon 2^{-|S|} \geq 0.
\end{aligned}$$

Remark: Some of the conditions in Lemma 2 can readily be relaxed. First, notice that the choice of π as the uniform distribution is not essential. The only property that was needed for the argument to work was that the marginal

[1] Note that for a distribution μ_S we have $\mathbb{E}_{x\sim\mu_S}[x_i] = \sum_{x\in\{-1,1\}^S}\mu_S(x)x_i$. Hence, the above sums are relevant to (P1) and (P2).

distribution of π over S assigns positive probability to every member of $\{-1,1\}^S$ (the existence of compatible $\mathbf{w}_i$'s is also required but is true for any distribution on $\{-1,1\}^n$.) More precisely there is a positive function $\delta(k)$ so that,

$$\Pr_{x \sim \mathrm{Mar}_S\, \pi} [x = y] \geq \delta(k).$$

One would need a stronger condition on ϵ in this case that depends on $\delta(k)$. The inner product of the resulting vectors would of course depend on the distribution π, namely,

$$\langle \mathbf{u}_0, \mathbf{u}_i \rangle = \epsilon \langle \mathbf{v}_0, \mathbf{v}_i \rangle + (1-\epsilon) \mathop{\mathbb{E}}_{x \sim \pi} [x_i]\,,$$
$$\langle \mathbf{u}_i, \mathbf{u}_j \rangle = \epsilon \langle \mathbf{v}_i, \mathbf{v}_j \rangle + (1-\epsilon) \mathop{\mathbb{E}}_{x \sim \pi} [x_i x_j]\,.$$

Another observation is that ν and π do not have to be true (global) distributions on $\{-1,1\}^n$. Instead, we can start with two families of distributions on sets of size at most k and a set of vectors $\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_n$ such that π_S's are compatible with $\mathbf{w}_i$'s and ν_S's are consistent with each other in the sense of equality (P3).

4 Application to Quadratic Programming

In the Quadratic Programming problem one is given a matrix $A_{n\times n}$ and is asked to find $x \in \{-1,1\}^n$ maximizing the quadratic form $\sum_{i\neq j} a_{ij} x_i x_j$. The canonical semidefinite programming relaxation of this problem replaces the integer (± 1) valued x_i's with unit vectors $\mathbf{v}_i$'s, and products $x_i x_j$ with the inner products $\langle \mathbf{v}_i, \mathbf{v}_j \rangle$. In particular, showing an integrality gap of $\Theta(\log N)$ for the canonical SDP relaxation is equivalent to showing that for any large enough N there exists a matrix $A_{N\times N}$ and value ξ, such that the following hold; (a) for all $x \in \{-1,1\}^N$, $\sum_{i\neq j} a_{ij} x_i x_j \leq O(\xi/\log(N))$, (b) there exist unit vectors $\{\mathbf{v}_i\}$ such that $\sum_{i\neq j} a_{ij} \langle \mathbf{v}_i, \mathbf{v}_j \rangle \geq \Omega(\xi)$. In order to show an integrality gap for the Sherali-Adams SDP of level k one in addition needs to show that $\{\mathbf{v}_i\}$'s are compatible with a set of distributions $\{\mu_S\}_{S \subset [n], |S| \leq k}$.

As discussed earlier we will start with the integrality gap instance of Khot and O'Donnell [7] and apply lemma 2. The following is a fairly immediate corollary that follows from Theorem 4.4 and Proposition 2.2 in [7]. It's worth noting that Khot and O'Donnell are mainly interested in the *existence* of integrality gap instances, and the matrix A is implicit in their work.

Corollary 1 (follows from [7]). *Let $\xi > 0$ be sufficiently small and let $d = 1/\xi^3$ and $n = \Theta(d^7)$. Further, let $m = \frac{1}{\xi}\log(\frac{1}{\xi}) = \Theta(n^{1/21}\log n)$ and $N = nm$. Choose n vectors $\mathbf{u}_1, \ldots, \mathbf{u}_n \in \mathbb{R}^d$ according to the d-dimensional Gaussian distribution. Then one can define $A_{N\times N}$ as a function of $\mathbf{u}_i$'s such that almost surely the following two conditions hold:*

1. *$\sum_{i\neq j} a_{ij} x_i x_j \leq O(\xi/\log(1/\xi))$ for all $x \in \{-1,1\}^N$.*
2. *There exist unit vectors $\mathbf{v}_i$ such that $\sum_{i\neq j} a_{ij} \langle \mathbf{v}_i, \mathbf{v}_j \rangle \geq \Omega(\xi)$.*

Furthermore, the $\mathbf{v}_i$*'s are produced in this simple manner based on* $\mathbf{u}_i$*'s. Divide the* N *variables into* n *classes of size* m *each, and assign the vector* $\mathbf{u}_j/\|\mathbf{u}_j\|$ *to the variables in the* j*th class. Formally,* $\mathbf{v}_i = \mathbf{u}_{\lceil i/m\rceil}/\|\mathbf{u}_{\lceil i/m\rceil}\|$.

We will need the the following property of the $\mathbf{v}_i$'s which easily follows from well-known properties of random unit vectors. We will prove it in appendix A for completeness.

Fact 3. *In the SDP solution of [7], with probability at least* $1-4/n^4$*, for all pairs of indices* $1 \le i,j \le N$ *the inner product* $\langle \mathbf{v}_i, \mathbf{v}_j\rangle$ *has the following property,*

$$\begin{aligned} \langle \mathbf{v}_i, \mathbf{v}_j\rangle &= 1 && \text{if } i \text{ and } j \text{ are in the same class, i.e. } \lceil i/m\rceil = \lceil j/m\rceil, \\ |\langle \mathbf{v}_i, \mathbf{v}_j\rangle| &\le \sqrt{(12\log n)/d} && \text{if } i \text{ and } j \text{ are in different classes, i.e. } \lceil i/m\rceil \ne \lceil j/m\rceil. \end{aligned}$$

Recall that in Lemma 2 a choice of a distribution ν on $\{-1,1\}^n$ is required. In particular, if for every pair of variables i,j, $\mathbb{E}_{x\sim\nu}[x_i x_j]$ is close to $\langle \mathbf{v}_i, \mathbf{v}_j\rangle$ one can choose a big value for ϵ in the lemma, which in turn means that the resulting $\mathbf{u}_i$'s will have inner products close to those of $\mathbf{v}_i$'s.

Indeed, the key to Theorem 1 is using ν which is “agreeable” with fact 3: two variables will have a large or a small covariance depending on whether they are from the same or different classes, respectively. Luckily, this is easily achievable by identifying variables in the same class and assigning values independently across classes. In other words the distribution ν will choose a random value from $\{-1,1\}$ for $x_1 = x_2 = \cdots = x_m$, an independently chosen value for $x_{m+1} = \cdots = x_{2m}$, and similarly an independently chosen value for $x_{nm-m+1} = \cdots = x_{nm}$. Such ν clearly satisfies,

$$\mathop{\mathbb{E}}_{x\sim\nu}[x_i x_j] = \begin{cases} 1 \text{ if } i \text{ and } j \text{ are in the same class,} \\ 0 \text{ otherwise.} \end{cases}$$

Consider the vector solution of [7], $\mathbf{v}_1, \ldots, \mathbf{v}_{nm}$ and define $\mathbf{v}_0$ as a vector perpendicular to all other $\mathbf{v}_i$'s. Consider the distribution ν defined above and apply Lemma 2 for $\mathbf{v}_0, \ldots, \mathbf{v}_i$, ν, $k = d^{0.2}$, and $\epsilon = 1/2$. By Fact 3 the inner products of the $\mathbf{v}_i$ vectors are close to the corresponding correlations of the distribution ν. It is easy to check that the conditions of Lemma 2 are satisfied,

$$\begin{aligned} 2\epsilon k^2 |\langle \mathbf{v}_0, \mathbf{v}_i\rangle - \mathop{\mathbb{E}}_{x\sim\nu}[x_i]\,| &= 2\epsilon k^2 |0-0| = 0, \\ 2\epsilon k^2 |\langle \mathbf{v}_i, \mathbf{v}_j\rangle - \mathop{\mathbb{E}}_{x\sim\nu}[x_i x_j]\,| &\le d^{0.4}\sqrt{(12\log n)/d} = \sqrt{12\log n}/d^{0.1} \ll 1, \end{aligned}$$

and the lemma applies for large enough n thus the resulting vectors, $\mathbf{u}_i$'s, are a solution to the Sherali-Adams SDP of level k. It is now easy to see that (4) implies a big objective value for this solution,

$$\sum_{i\ne j} a_{ij}\langle \mathbf{u}_i, \mathbf{u}_j\rangle = \epsilon \sum_{i\ne j} a_{ij}\langle \mathbf{v}_i, \mathbf{v}_j\rangle \ge \Omega(\xi).$$

It remains to estimate the value of k in terms of N:

$$k = d^{0.2} = \Theta(n^{1/35}) = \Theta(N^{21/770}/\log^{1/35} N) = \Omega(N^{1/37}),$$

and we conclude,

Theorem 1 (restated). *For $\delta = 1/37$, the Sherali-Adams SDP of level N^δ for* QUADRATIC PROGRAMMING *has integrality gap $\Omega(\log N)$.*

5 Application to MAXCUTGAIN

The MAXCUTGAIN problem is an important special case of the QUADRATIC PROGRAMMING problem where all entries of the matrix A are nonpositive. In other words the input is a *nonpositive* n by n matrix A and the objective is to find $x \in \{-1,1\}^n$ that maximizes the quadratic form $\sum_{i \neq j} a_{ij} x_i x_j$. This problem gets its name and main motivation from studying algorithms for the MAXCUT problem that perform well for graphs with maximum cut value close to half the edges. See [7] for a discussion.

Naturally, constructing integrality gap instances for MAXCUTGAIN is harder than Quadratic Programing. The best integrality gap instances are due to Khot and O'Donnell [7] who, for any constant Δ, construct an instance of integrality gap at least Δ. The following is a restatement of their Theorem 4.1 tailored to our application.

Theorem 3 ([7]). *The standard SDP relaxation for* MAXCUTGAIN *has super constant integrality gap. Specifically, for any constant $\xi > 0$, there is a big enough n and a matrix $A_{n\times n}$ such that,*

1. $\sum_{i \neq j} a_{ij} x_i x_j \leq \xi / \log \frac{1}{\xi}$ *for all $x \in \{-1,1\}^n$.*
2. *There are unit vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ such that $\sum_{i \neq j} a_{ij} \langle \mathbf{v}_i, \mathbf{v}_j \rangle \geq \Omega(\xi)$.*

It should be mentioned that the proof of [7] is continuous in nature and it is not entirely clear how n grows as a function of $1/\xi$ (once some form of discretization is applied to their construction.) However, an integrality gap of $f(n)$ for some function $f(n) = \omega(1)$ is implicit in the above theorem.

Consider the instance from Theorem 3 and the function $f(n)$ as above and let $g(n) = \sqrt[3]{f(n)}$. We know that for every n, there are unit vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ such that $\sum_{i \neq j} a_{ij} \langle \mathbf{v}_i, \mathbf{v}_j \rangle \geq \Omega(\xi)$. Let $\mathbf{v}_0$ be a unit vector perpendicular to all $\mathbf{v}_i$'s and set $k = g(n)$, $\epsilon = g(n)^{-2}/2$ and let ν be the uniform distribution on $\{-1,1\}^n$. Note that for all $i < j$,

$$2\epsilon k^2 |\langle \mathbf{v}_0, \mathbf{v}_i \rangle - \mathop{\mathbb{E}}_{x\sim\nu}[x_i]| = |\langle \mathbf{v}_0, \mathbf{v}_i \rangle| = 0,$$
$$2\epsilon k^2 |\langle \mathbf{v}_i, \mathbf{v}_j \rangle - \mathop{\mathbb{E}}_{x\sim\nu}[x_i x_j]| = |\langle \mathbf{v}_i, \mathbf{v}_j \rangle| \leq 1,$$

hence the conditions of Lemma 2 hold. Consequently, there are vectors $\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_n$ compatible with a family of distributions $\{\mu_S\}$ on subsets of size up to $k = g(n)$ which satisfy (4). Now,

$$\sum_{i \neq j} a_{ij} \langle \mathbf{u}_i, \mathbf{u}_j \rangle = \epsilon \sum_{i \neq j} a_{ij} \langle \mathbf{v}_i, \mathbf{v}_j \rangle \geq \Omega(\xi / g(n)^2),$$

$$\forall x \in \{-1,1\}^n \qquad \sum_{i \neq j} a_{ij} x_i x_j \leq O(\xi / f(n)) = O(\xi / g(n)^3)$$

and we obtain,

Theorem 2 (restated). *There exists a function $g(n) = \omega(1)$, such that the Sherali-Adams SDP of level $g(n)$ for* MaxCutGain *has integrality gap $\Omega(g(n))$.*

6 Discussion

We saw that the Sherali-Adams SDP of level $n^{\Omega(1)}$ for Quadratic Programming has the same asymptotic integrality gap as the canonical SDP, namely $\Omega(\log n)$. It is interesting to see other problems for which this kind of construction can prove meaningful integrality gap results. It is easy to see that as long as a problem does not have "hard" constraints, and a super constant integrality gap for the canonical SDP relaxation is known, one can get super constant integrality gaps for super constant levels of the Sherali-Adams SDP just by plugging in the uniform distribution for ν in Lemma 2.

It is possible to show that the same techniques apply when the objective function is a polynomial of degree greater than 2 (but still constant.) This is particularly relevant to Max-CSP(P) problems. When formulated as a maximization problem of a polynomial q over ± 1 valued variables, q will have degree r, the arity of P. In fact, the canonical SDP formulation for the case $r > 2$ will be very similar to Sherali-Adams SDP of level r in our case. In order to adapt Lemma 2 to this setting, the Fourier expansion of f_S's should be adjusted appropriately. Specifically, their Fourier expansion would match that of the starting SDP solution up to level r and that of ν beyond level r. It is also possible to define "gain" versions of Max-CSP(P) problems in this setting and extend existing superconstant integrality gaps to the Sherali-Adams SDP of superconstant level (details omitted in this extended abstract.)

The first obvious open problem is to extend these constructions to be applicable to problems with "hard" constraints for which SDP integrality gaps have been (or may potentially be) shown. A possibly reasonable candidate along these lines would be the Sparsest Cut problem in which we have one normalizing constraint in the SDP relaxation, and for which superconstant integrality gaps are known. In contrast, it seems quite unlikely that these techniques can be extended for Vertex Cover where the integrality gap is constant. Another direction is to extend these techniques to the Lasserre hierarchy for which very few integrality gaps are known. We believe that this is possible but more sophisticated "merge operators" of distributions and vectors á la Lemma 2 will be necessary.

References

1. Alon, N., Makarychev, K., Makarychev, Y., Naor, A.: Quadratic forms on graphs. Invent. Math. 163(3), 499–522 (2006)
2. Arora, S., Berger, E., Kindler, G., Safra, M., Hazan, E.: On non-approximability for quadratic programs. In: FOCS '05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science, pp. 206–215. IEEE Computer Society, Washington (2005)

3. Charikar, M., Makarychev, K., Makarychev, Y.: Integrality gaps for sherali-adams relaxations. In: STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing, pp. 283–292. ACM, New York (2009)
4. Charikar, M., Wirth, A.: Maximizing quadratic programs: Extending grothendieck's inequality. In: FOCS '04: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science, pp. 54–60. IEEE Computer Society, Washington (2004)
5. Georgiou, K., Magen, A., Tulsiani, M.: Optimal sherali-adams gaps from pairwise independence. In: APPROX-RANDOM, pp. 125–139 (2009)
6. Grothendieck, A.: Résumé de la théorie métrique des produits tensoriels topologiques. Bol. Soc. Mat. São Paulo 8, 1–79 (1953)
7. Khot, S., O'Donnell, R.: Sdp gaps and ugc-hardness for maxcutgain. In: FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, pp. 217–226. IEEE Computer Society, Washington (2006)
8. 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)
9. Lovász, L., Schrijver, A.: Cones of matrices and set-functions and 0–1 optimization. SIAM Journal on Optimization 1(2), 166–190 (1991)
10. Mathieu, C., Sinclair, A.: Sherali-adams relaxations of the matching polytope. In: STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing, pp. 293–302. ACM, New York (2009)
11. Matousek, J.: Lectures on Discrete Geometry. Springer, New York (2002)
12. Raghavendra, P., Steurer, D.: How to round any csp. In: FOCS '09: Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (to appear 2009)
13. Raghavendra, P., Steurer, D.: Integrality gaps for strong sdp relaxations of unique games. In: FOCS '09: Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (to appear 2009)
14. Schoenebeck, G.: Linear level lasserre lower bounds for certain k-csps. In: FOCS '08: Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science, pp. 593–602. IEEE Computer Society, Washington (2008)
15. Sherali, H.D., Adams, W.P.: A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. SIAM Journal on Discrete Mathematics 3(3), 411–430 (1990)
16. Tulsiani, M.: Csp gaps and reductions in the lasserre hierarchy. In: STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing, pp. 303–312. ACM, New York (2009)
17. de la Vega, W.F., Kenyon-Mathieu, C.: Linear programming relaxations of maxcut. In: SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, pp. 53–61. Society for Industrial and Applied Mathematics, Philadelphia (2007)

A Proof of Fact 3

We will need the following simple and well known lemma that most of the area of $\mathbb{S}^{d-1}$ is concentrated around the equator.

Lemma 4. *For any unit vector* $\mathbf{v} \in \mathbb{S}^{d-1}$*, if a unit vector* $\mathbf{x} \in \mathbb{S}^{d-1}$ *is chosen uniformly at random,* $|\langle \mathbf{v}, \mathbf{x} \rangle|$ *is sharply concentrated:*

$$\Pr\left[|\langle \mathbf{v}, \mathbf{x} \rangle| \geq t/\sqrt{d}\right] \leq 4e^{-t^2/2}.$$

Proof. Define,

$$f(x) \overset{\text{def}}{=} \langle \mathbf{v}, \mathbf{x} \rangle,$$

and apply Lévy's lemma (see Theorem 14.3.2 of [11]) observing that $f(x)$ is 1-Lipschitz. We will have,

$$\begin{aligned}\Pr\left[|\langle \mathbf{v}, \mathbf{x} \rangle| \geq t/\sqrt{d}\right] &= \Pr\left[|f(x) - \text{median}(f)| \geq t/\sqrt{d}\right] \leq 4e^{-(t/\sqrt{d})^2 d/2} \\ &= 4e^{-t^2/2}.\end{aligned}$$

Now, proving the fact is a matter of looking at the actual definition of the solution vectors and applying lemma 4.

Fact 3 (restated). *In the SDP solution of [7], with probability at least* $1-4/n^4$*, for all pairs of indices* $1 \leq i, j \leq N$ *the inner product* $\langle \mathbf{v}_i, \mathbf{v}_j \rangle$ *has the following property,*

$$\begin{aligned} \langle \mathbf{v}_i, \mathbf{v}_j \rangle &= 1 && \textit{if } i \textit{ and } j \textit{ are in the same class,} \\ |\langle \mathbf{v}_i, \mathbf{v}_j \rangle| &\leq \sqrt{(12 \log n)/d} && \textit{if } i \textit{ and } j \textit{ are in different classes.} \end{aligned}$$

Proof. The first case follows from the definition of $\mathbf{v}_i$'s. For the second case $\mathbf{v}_i$ and $\mathbf{v}_j$ are independent d-dimensional vectors distributed uniformly on $\mathbb{S}^{d-1}$. Consider a particular choice of $\mathbf{v}_i$, according to lemma 4,

$$\Pr_{\mathbf{v}_j}\left[|\langle \mathbf{v}_i, \mathbf{v}_j \rangle| \geq \sqrt{(12 \log n)/d}\right] \leq 4e^{-6 \log n} = 4n^{-6}.$$

Applying union bound on all n^2 pairs of classes shows that the condition of the lemma holds for all pairs with probability at least,

$$1 - n^2 4n^{-6} = 1 - 4/n^4.$$

The Price of Collusion in Series-Parallel Networks[⋆]

Umang Bhaskar[1], Lisa Fleischer[1], and Chien-Chung Huang[2,⋆⋆]

[1] Department of Computer Science, Dartmouth College, Hanover, NH 03755, U.S.A.
{umang,lkf}@cs.dartmouth.edu
[2] Max-Planck-Institut für Informatik, Saarbrücken, 66123, Germany
villars@mpi-inf.mpg.de

Abstract. We study the quality of equilibrium in atomic splittable routing games. We show that in single-source single-sink games on series-parallel graphs, the *price of collusion* — the ratio of the total delay of atomic Nash equilibrium to the Wardrop equilibrium — is at most 1. This proves that the existing bounds on the price of anarchy for Wardrop equilibria carry over to atomic splittable routing games in this setting.

1 Introduction

In a *routing game*, players have a fixed amount of flow which they route in a network [16,18,24]. The flow on any edge in the network faces a delay, and the delay on an edge is a function of the total flow on that edge. We look at routing games in which each player routes flow to minimize his own delay, where a player's delay is the sum over edges of the product of his flow on the edge and the delay of the edge. This objective measures the average delay of his flow and is commonly used in traffic planning [11] and network routing [16].

Routing games are used to model traffic congestion on roads, overlay routing on the Internet, transportation of freight, and scheduling tasks on machines. Players in these games can be of two types, depending on the amount of flow they control. Nonatomic players control only a negligible amount of flow, while atomic players control a larger, non-negligible amount of flow. Further, atomic players may or may not be able to split their flow along different paths. Depending on the players, three types of routing games are: games with (i) nonatomic players, (ii) atomic players who pick a single path to route their flow, and (iii) atomic players who can split their flow along several paths. These are *nonatomic* [21,22,24], *atomic unsplittable* [3,10] and *atomic splittable* [8,16,19] routing games respectively. We study atomic splittable routing games in this work. These games are less well-understood than either nonatomic or atomic unsplittable routing games. One significant challenge here is that, unlike most other routing games, each player has an infinite strategy space. Further, unlike nonatomic routing games, the players are asymmetric since each player has different flow value.

[⋆] This work was supported in part by NSF grant CCF-0728869.
[⋆⋆] Research supported by an Alexander von Humboldt fellowship.

An equilibrium flow in a routing game is a flow where no single player can change his flow pattern and reduce his delay. Equilibria are of interest since they are a stable outcome of games. In both atomic splittable and nonatomic routing games, equilibria exist under mild assumptions on the delay functions [4,17]. We refer to equilibria in atomic splittable games as *Nash* equilibria and in nonatomic games as *Wardrop* equilibria [24]. While the Wardrop equilibrium is known to be essentially unique [24], atomic splittable games can have multiple equilibria [5].

One measure of the quality of a flow is the *total delay* of the flow: the sum over all edges of the product of the flow on the edge and the induced delay on the edge. For routing games, one conern is the degradation in the quality of equilibrium flow caused by lack of central coordination. This is measured by the *price of anarchy* of a routing game, defined as the ratio of the total delay of worst-case equilibrium in a routing game to the total delay of the flow that minimizes the total delay. Tight bounds on the price of anarchy are known for nonatomic routing games [20], and are extensively studied in various settings [8,9,20,21,19,22,23]. In [13], Hayrapetyan et al. consider the total delay of nonatomic routing games when nonatomic players form cost-sharing coalitions. These coalitions behave as atomic splittable players. Hayrapetyan et al. introduce the notion of *price of collusion* as a measure of the price of forming coalitions. For an atomic splittable routing game the price of collusion is defined as the ratio of the total delay of the worst Nash equilibrium to the Wardrop equilibrium. Together, a bound α on the price of anarchy for nonatomic routing games and a bound β on the price of collusion for an atomic splittable routing game, imply the price of anarchy for the atomic splittable routing game is bounded by $\alpha\beta$.

For atomic splittable routing games, bounds on the price of anarchy are obtained in [8,12]. These bounds do not match the best known lower bounds. Bounds on the price of collusion in general also remains an open problem. Previously, the price of collusion has been shown to be 1 only in the following special cases: in the graph consisting of parallel links [13]; when the players are symmetric, i.e. each player has the same flow value and the same source and sink [8]; and when all delay functions are monomials of a fixed degree [2]. Conversely, if there are multiple sources and sinks, the total delay of Nash equilibrium can be worse than the Wardrop equilibrium of equal flow value, i.e., the price of collusion can exceed 1, even with linear delays [7,8].

Our Contribution. Let $\mathcal{C}$ denote the class of differentiable nondecreasing convex functions. We prove the following theorem for atomic splittable routing games.

Theorem 1. *In single source-destination routing games on series-parallel graphs with delay functions drawn from the class $\mathcal{C}$, the price of collusion is 1.*

We first consider the case when all delays are affine. We show that in the case of affine delays in the setting described above, the total delay at equilibrium is largest when the players are symmetric, i.e. all players have the same flow value (Section 3). To do this, we first show that the equilibrium flow for a player i remains unchanged if we modify the game by changing slightly the value of flow of any player with larger flow value than player i. Then starting from a game with symmetric players, we show that if one moves flow from a player i evenly to all players with higher flow value the cost of the corresponding equilibrium flow never increases. Since it is known that the price of collusion is 1 if the players are symmetric [8], this shows that the bound extends to arbitrary players with affine delays.

In Section 4, we extend the result for general convex delays, by showing that the worst case price of collusion is obtained when the delays are affine.

In contrast to Theorem 1 which presents a bound on the price of collusion, we also present a new bound on the price of anarchy of atomic splittable routing games in series-parallel graphs.

Theorem 2. *In single source-destination routing games on series-parallel graphs, the price of anarchy is bounded by k, the number of players.*

This bound was proven earlier for parallel links in [12]. For nonatomic routing games bounds on the price of anarchy depend on the delay functions in the graph and, in the case of polynomial delays, the price of anarchy is bounded by $O(d/\log d)$. These bounds are known to be tight even on simple graphs consisting of 2 parallel links [20]. Theorem 2 improves on the bounds obtained by Theorem 1 when $k \leq d/\log d$. All missing proofs are contained in the full version [6].

2 Preliminaries

Let $G = (V, E)$ be a directed graph, with two special vertices s and t called the *source* and *sink*. The vector f, indexed by edges $e \in E$, is defined as a *flow* of *value* v if the following conditions are satisfied.

$$\sum_w f_{uw} - \sum_w f_{wu} = 0, \quad \forall u \in V - \{s, t\} \tag{1}$$

$$\sum_w f_{sw} - \sum_w f_{ws} = v \tag{2}$$

$$f_e \geq 0, \quad \forall e \in E \ .$$

Here f_{uw} represents the flow on arc (u, w). If there are several flows $f^1, f^2, \cdots, f^k$, we define $f := (f^1, f^2, \cdots, f^k)$ and f^{-i} is the vector of the flows except f^i. In this case the flow on an edge $f_e = \sum_{i=1}^k f_e^i$.

Let $\mathcal{C}$ be the class of differentiable nondecreasing convex functions. Each edge e is associated with a delay function $l_e : \mathcal{R}^+ \to \mathcal{R}$ drawn from $\mathcal{C}$. Note that we allow delay functions to be negative. For a given flow f, the *induced delay* on edge e is $l_e(f_e)$. We define the *total delay* on an edge e as the product of the flow on the edge and the induced delay $C_e(f_e) := f_e l_e(f_e)$. The *marginal delay* on an edge e is the rate of change of the total delay: $L_e(f_e) := f_e l'_e(f_e) + l_e(f_e)$. The total delay of a flow f is $C(f) = \sum_{e \in E} f_e l_e(f_e)$.

An *atomic splittable routing game* is a tuple $(G, \boldsymbol{v}, \boldsymbol{l}, s, t)$ where $\boldsymbol{l}$ is a vector of delay functions for edges in G and $\boldsymbol{v} = (v^1, v^2, \cdots, v^k)$ is a tuple indicating the flow value of the players from 1 to k. We always assume that the players are indexed by the order of decreasing flow value, hence $v^1 \geq v^2 \cdots \geq v^k$. All players have source s and destination t. Player i has a strategy space consisting of all possible s-t flows of volume v^i. Let $(f^1, f^2, \cdots, f^k)$ be a strategy vector. Player i incurs a delay $C^i_e(f^i_e, f_e) := f^i_e l_e(f_e)$ on each edge e, and his objective is to minimize his delay $C^i(f) := \sum_{e \in E} C^i_e(f^i_e, f_e)$. A set of players are *symmetric* if each player has the same flow value.

A flow is a *Nash equilibrium* if no player can unilaterally alter his flow and reduce his delay. Formally,

Definition 3 (Nash Equilibrium). *In an atomic splittable routing game, flow f is a* Nash equilibrium *if and only if for every player i and every s-t flow g of volume v^i, $C^i(f^i, f^{-i}) \leq C^i(g, f^{-i})$.*

For player i, the marginal delay on edge e is defined as the rate of change of his delay on the edge $L^i_e(f^i_e, f_e) := l_e(f_e) + f^i_e l'_e(f_e)$. For any s-t path p, the marginal delay on path p is defined as the rate of change of total delay of player i when he adds flow along the edges of the path: $L^i_p(f) := \sum_{e \in p} L^i_e(f^i_e, f_e)$. The following lemma follows from Karush-Kuhn-Tucker optimality conditions for convex programs [15] applied to player i's minimization problem.

Lemma 4. *Flow f is a Nash equilibrium flow if and only if for any player i and any two directed paths p and q between the same pair of vertices such that on all edges $e \in p$, $f^i_e > 0$, then $L^i_p(f) \leq L^i_q(f)$.*

By Lemma 4, at equilibrium the marginal delay of a player is the same on any s-t path on every edge of which he has positive flow. For a player i, the *marginal delay* is $L^i(f) := L^i_p(f)$, where p is any s-t path on which player i has positive flow on every edge.

For a given flow f and for every player i, we let $E^i(f) = \{e | f^i_e > 0\}$. $\mathcal{P}^i$ is the set of all directed s-t paths p on which for every $e \in p$, $f^i_e > 0$. We will use $e \in \mathcal{P}^i$ to mean that the edge e is in some path $p \in \mathcal{P}^i$; then $e \in \mathcal{P}^i \Leftrightarrow e \in E^i$. Let p be a directed simple s-t path. A *path flow* on path p is a directed flow on p of value f_p. A *cycle flow* along cycle C is a directed flow along C of value f_C. Any flow f can be decomposed into a set of directed path flows and directed cycle flows $\{f_p\}_{p \in \mathcal{P}} \cup \{f_c\}_{c \in C}$, [1]. This is a *flow decomposition* of f. Directed cycle flows cannot exist in atomic splittable or nonatomic games (this follows easily from Lemma 4). Thus, f^i in these games can be expressed as a set of path flows $\{f^i_p\}_{p \in \mathcal{P}^i}$ such that $f^i_e = \sum_{p \in \mathcal{P}^i : e \in p} f^i_p$. This is a *path flow decomposition* of the given flow. A *generalized path flow decomposition* is a flow decomposition along paths where we allow the path flows to be negative.

Series-Parallel Graphs. Given graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ and vertices $v_1 \in V_1$, $v_2 \in V_2$, the operation merge(v_1, v_2) creates a new graph $G' = (V' = V_1 \cup V_2, E' = E_1 \cup E_2)$, replaces v_1 and v_2 in V' with a single vertex v and replaces each edge $e = (u, w) \in E'$ incident to v_1 or v_2 by an edge incident to v, directed in the same way as the original edge.

Definition 5. *A tuple (G, s, t) is* series-parallel *if G is a single edge $e = (s, t)$, or is obtained by a series or parallel composition of two series-parallel graphs (G_1, s_1, t_1) and (G_2, s_2, t_2). Nodes s and t are* terminals *of G.*

(i) Parallel Composition*: $s = merge(s_1, s_2)$, $t = merge(t_1, t_2)$,*
(ii) Series Composition*: $s := s_1$, $t := t_2$, $v = merge(s_2, t_1)$.*

In directed series-parallel graphs, all edges are directed from the source to the destination and the graph is acyclic in the directed edges. This is without loss of generality, since any edge not on an s-t path is not used in an equilibrium flow, and no flow is sent along a directed cycle. The following lemma describes a basic property of flows in a directed series-parallel graph.

Lemma 6. *Let $G = (V, E)$ be a directed series-parallel graph with terminals s and t. Let h be an s-t flow of value $|h|$, and c is a function defined on the edges of the graph G. (i) If $\sum_{e \in p} c(e) \geq \kappa$ on every s-t path p, then $\sum_{e \in E} c(e) h_e \geq \kappa |h|$. (ii) If $\sum_{e \in p} c(e) = \kappa$ on every s-t paths p then $\sum_{e \in E} c(e) h_e = \kappa |h|$.*

Vectors and matrices in the paper, except for flow vectors, will be referred to using boldface. $\mathbf{1}$ and $\mathbf{0}$ refer to the column vectors consisting of all ones and all zeros respectively. When the size of the vector or matrix is not clear from context, we use a subscript to denote it, e.g. $\mathbf{1}_n$.

Uniqueness of Equilibrium Flow. The equilibria in atomic splittable and nonatomic routing games are known to be unique for affine delays, up to induced delays on the edges (this is true for a larger class of delays [4], [17], but here we only need affine delays). Although there may be multiple equilibrium flows, in each of these flows the delay on an edge remains unchanged. If the delay functions are strictly increasing, then the flow on each edge is uniquely determined. However with constant delays, for two parallel links between s and t with the same constant delay on each edge, any valid flow is an equilibrium flow. In this paper, we assume only that the delay functions are differentiable, nondecreasing and convex, hence we allow edges to have constant delays. We instead assume that in the graph, between any pair of vertices, there is at most one path on which all edges have constant delay. This does not affect the generality of our results. In graphs without this restriction there are Nash and Wardrop equilibrium flows in which for every pair of vertices, there is at most one constant delay path which has flow in either equilibrium. To see this, consider any equilibrium flow in a general graph. For every pair of vertices with more than one constant delay path between them, only the minimum delay path will be used at equilibrium. If there are multiple minimum constant delay paths, we can shift all the flow onto a single path; this does not affect the marginal delay of any player on any path, and hence the flow is still an equilibrium flow.

Lemma 7. *For atomic splittable and nonatomic routing games on series-parallel networks with affine delays and at most one path between any pair of vertices with constant delays on all edges, the equilibrium flow is unique.*

For technical reasons, for proving Theorem 1 we also require that every s-t path in the graph have at least one edge with strictly increasing delay. We modify the graph in the following way: we add a single edge e in series with graph G, with delay function $l_e(x) = x$. It is easy to see that for any flow, this increases the total delay by exactly v^2 where v is the value of the flow, and does not change the value of flow on any edge at equilibrium. In addition, if the price of collusion in the modified graph is less than one, then the price of collusion in the original graph is also less than one. The proof of Theorem 2 does not use this assumption.

3 Equilibria with Affine Delays

In this section we prove Theorem 1 where all delays are affine functions of the form $l_e(x) = a_e x + b_e$. Our main result in this section is:

Theorem 8. *In a series-parallel graph with affine delay functions, the total delay of a Nash equilibrium is bounded by that of a Wardrop equilibrium of the same total flow value.*

We first present the high-level ideas of our proof. Given a series-parallel graph G, terminals s and t, and edge delay functions $\boldsymbol{l}$, let $f(\cdot) : \mathbb{R}_+^k \rightarrow \mathbb{R}_+^{m \times k}$ denote the function mapping a vector of flow values to the equilibrium flow in the atomic splittable routing game. By Lemma 7, the equilibrium flow is unique and hence the function $f(\cdot)$ is well-defined. Let $(G, \boldsymbol{u}, \boldsymbol{l}, s, t)$ be an atomic splittable routing game. Our proof consists of the following three steps:

Step 1. Start with $v^i = \sum_{j=1}^{k} u^j/k$ for each player i, i.e. the players are symmetric.

Step 2. Gradually adjust the flow values $\boldsymbol{v}$ of the k players so that the total delay of the equilibrium flow $f(\boldsymbol{v})$ is monotonically nonincreasing.

Step 3. Stop the flow redistribution process when for each i, $v^i = u^i$.

In step 1, we make use of a result of Cominetti et al. [8].

Lemma 9. [8] *Let $(G, \boldsymbol{v}, \boldsymbol{l}, s, t)$ denote an atomic splittable routing game with k symmetric players. Let g be a Wardrop equilibrium of the same flow value $\sum_{i=1}^{k} v^i$. Then $C(f(\boldsymbol{v})) \leq C(g)$.*

Step 2 is the heart of our proof. The flow redistribution works as follows. Let v^i denote the current flow value of player i. Initially, each player i has $v^i = \sum_{j=1}^{k} u^j/k$. Consider each player in turn from k to 1. We decrease the flow of the kth player and give it *evenly* to the first $k-1$ players until $v^k = u^k$. Similarly, when we consider the rth player, for any $r < k$, we decrease v^r and give the flow evenly to the first $r-1$ players until $v^r = u^r$. Throughout the following discussion and proofs, player r refers specifically to the player whose flow value is currently being decreased in our flow redistribution process.

Our flow redistribution strategy traces out a curve $\mathcal{S}$ in $\mathbb{R}_+^k$, where points in the curve correspond to flow value vectors $\boldsymbol{v}$.

Lemma 10. *For all $e \in E$, $i \in [k]$, the function $f(\boldsymbol{v})$ is continuous and piecewise linear along the curve $\mathcal{S}$, with breakpoints occurring where the set of edges used by any player changes.*

In what follows, we consider expressions of the form $\frac{\partial J(f(\boldsymbol{v}))}{\partial v^i}$, where J is some differentiable function defined on a flow (e.g., the total delay, or the marginal delay along a path). The expression $\frac{\partial J(f(\boldsymbol{v}))}{\partial v^i}$ considers the change in the function $J(\cdot)$ evaluated at the equilibrium flow, as the flow value of player i changes by an infinitesimal amount, keeping the flow values of the other players constant. Though $f(\boldsymbol{v})$ is not differentiable at all points in $\mathcal{S}$, $\mathcal{S}$ is continous. Therefore, it suffices to look at the intervals between these breakpoints of $\mathcal{S}$. In the rest of the paper, we confine our attention to these intervals.

We show that when the flow values are adjusted as described, the total delay is monotonically nonincreasing.

Lemma 11. *In a series-parallel graph, suppose that $v^1 = v^2 = \cdots = v^{r-1} \geq v^r \geq \cdots \geq v^k$. If $i < r$, then $\frac{\partial C(f(\boldsymbol{v}))}{\partial v^i} \leq \frac{\partial C(f(\boldsymbol{v}))}{\partial v^r}$.*

Proof of Theorem 8. By Lemma 9, the equilibrium flow in Step 1 has total delay at most the delay of the Wardrop equilibrium. We show below that during step 2, $C(f(\boldsymbol{v}))$ does not increase. Since the total volume of flow remains fixed, the Wardrop equilibrium is unchanged throughout. Thus, the price of collusion does not increase above 1, and hence the final equilibrium flow when $\boldsymbol{v} = \boldsymbol{u}$ also has this property.

Let $\boldsymbol{v}$ be the current flow values of the players. Since $C(f(\boldsymbol{v}))$ is a continuous function of $\boldsymbol{v}$ (Lemma 10), it is sufficient to show that the $C(f(\boldsymbol{v}))$ does not increase between breakpoints. Define $\boldsymbol{x}$ as follows: $\boldsymbol{x}^r = -1$; $\boldsymbol{x}^i = 0$, if $i > r$; and $\boldsymbol{x}^i = \frac{1}{r-1}$, if $1 \leq i < r$. The vector $\boldsymbol{x}$ is the rate of change of $\boldsymbol{v}$ when we decrease the flow of player r in Step 2. Thus, using Lemma 11, the change in total delay between two breakpoints in $\mathcal{S}$ satisfies

$$\lim_{\delta \to 0} \frac{C(f(\boldsymbol{v} + \delta \boldsymbol{x})) - C(f(\boldsymbol{v}))}{\delta} = -\frac{\partial C(f(\boldsymbol{v}))}{\partial v^r} + \sum_{i=1}^{r-1} \frac{\partial C(f(\boldsymbol{v}))}{\partial v^i} \frac{1}{r-1} \leq 0 .$$

□

The proof of Lemma 11 is described in Section 3.2. Here we highlight the main ideas. To simplify notation, when the vector of flow values is clear from the context, we use f instead of $f(\boldsymbol{v})$ to denote the equilibrium flow.

By chain rule, we have that $\frac{C(f)}{\partial v^i} = \sum_{e \in E} \frac{\partial L_e(f_e)}{\partial f_e} \frac{\partial f_e}{\partial v^i}$. The exact expressions of $\frac{\partial C(f)}{\partial v^i}$, for $1 \leq i \leq r$, are given in Lemmas 18 and 19 in Section 3.2. Our derivations use the fact that it is possible to simplify the expression $\frac{\partial f_e}{\partial v^i}$ using the following "nesting property" of a series-parallel graph.

Definition 12. *A graph G with delay functions $\boldsymbol{l}$, source s, and destination t satisfies the* nesting property *if all atomic splittable routing games on G satisfy* *the following condition: for any players i and j with flow values v^i and v^j, $v^i > v^j$ if and only if on every edge $e \in E$, for the equilibrium flow f, either $f_e^i = f_e^j = 0$ or $f_e^i > f_e^j$.*

Lemma 13 ([5]). *A series-parallel graph satisfies the nesting property for any choice of non-decreasing, convex delay functions.*

If a graph satisfies the nesting property, symmetric players have identical flows at equilibrium. When the flow value of player r is decreased in Step 2, the first $r-1$ players are symmetric. Thus, by Lemma 13, these players have identical flows at equilibrium. Hence, for any player $i < r$, $f_e^i = f_e^1$ and $L_e^i(f_e^i, f_e) = L_e^1(f_e^1, f_e)$ for any edge e. With affine delays, the nesting property has the following implication.

Lemma 14 (Frozen Lemma). *Let f be an equilibrium flow in an atomic splittable routing game $(G,\boldsymbol{v},\boldsymbol{l},s,t)$ with affine delays on the edges, and assume that the nesting property holds for $(G,\boldsymbol{l},s,t)$. Then for all players j, $j \neq i$ with $E^j(f) \subseteq E^i(f)$ and all edges e, $\dfrac{\partial f_e^j}{\partial v^i} = 0$.*

The frozen lemma has two important implications for our proof. Firstly, in Step 2, players $r+1, \cdots, k$ will not change their flow at equilibrium. Secondly, this implies a simple expression for $\frac{\partial f_e}{\partial v^i}$, $1 \leq i \leq r$,

$$\frac{\partial f_e}{\partial v^r} = \sum_{i=1}^{k} \frac{\partial f_e^i}{\partial v^r} = (r-1)\frac{\partial f_e^1}{\partial v^r} + \frac{\partial f_e^r}{\partial v^r} \ . \tag{3}$$

$$\frac{\partial f_e}{\partial v^i} = \sum_{i=1}^{k} \frac{\partial f_e^i}{\partial v^i} = \frac{\partial f_e^i}{\partial v^i}, \quad \forall i < r \ . \tag{4}$$

3.1 Proof of Lemma 14 (Frozen Lemma)

By Lemma 10, we can assume that f is between the breakpoints of $\mathcal{S}$ and is thus differentiable.

Lemma 15. *If player h has positive flow on every edge of two directed paths p and q between the same pair of vertices, then $\frac{\partial L_p^h(f)}{\partial v^i} = \frac{\partial L_q^h(f)}{\partial v^i}$.*

Proof. Since f is an equilibrium, Lemma 4 implies that $L_p^h(f) = L_q^h(f)$. Differentiation of the two quantities are the same since f is maintained as an equilibrium. □

Lemma 16. *Let G be a directed acyclic graph. For an atomic splittable routing game $(G, \boldsymbol{v}, \boldsymbol{l}, s, t)$ with equilibrium flow f, let c and κ be defined as in Lemma 6. Then $\sum_{e \in E} c(e) \frac{\partial f_e^i(\boldsymbol{v})}{\partial v^j} = \kappa$ if $i = j$, and is zero otherwise.*

Proof. Define $\boldsymbol{x}$ as follows: $\boldsymbol{x}^j = 1$ and $\boldsymbol{x}^i = 0$ for $j \neq i$. Then

$$\sum_{e\in E} c(e)\frac{\partial f_e^i(\boldsymbol{v})}{\partial v^j} = \sum_{e\in E} c(e)\left(\lim_{\delta\to 0}\frac{f_e^i(\boldsymbol{v}+\delta\boldsymbol{x}) - f_e^i(\boldsymbol{v})}{\delta}\right)$$
$$= \lim_{\delta\to 0}\frac{\sum_{e\in E} c(e)(f_e^i(\boldsymbol{v}+\delta\boldsymbol{x}) - f_e^i(\boldsymbol{v}))}{\delta} ,$$

where the second equality is due to the fact that $f_e^i(\cdot)$ is differentiable.

For any two s-t flows f^i, g^i, it follows from Lemma 6 that $\sum_{e\in E} c(e)(f_e^i - g_e^i) = \kappa(|f^i| - |g^i|)$. If $i \neq j$ then $|f^i(\boldsymbol{v}+\delta\boldsymbol{x})| = |f^i(\boldsymbol{v})|$, hence $\sum_{e\in E} c(e)(f_e^i(\boldsymbol{v} + \delta\boldsymbol{x}) - f_e^i(\boldsymbol{v})) = 0$. If $i = j$, then $|f^i(\boldsymbol{v}+\delta\boldsymbol{x})| - |f^i(\boldsymbol{v})| = \delta$, implying that $\sum_{e\in E} c(e)(f_e^i(\boldsymbol{v}+\delta\boldsymbol{x}) - f_e^i(\boldsymbol{v})) = \kappa\delta$. The proof follows. $\square$

Proof of Lemma 14. We prove by induction on the decreasing order of the index of j. We make use of the folllowing claim.

Claim 17. *Let $S^j = \{h : E^h(f) \supseteq E^j(f)\}$. For player j and an s-t path p on which j has positive flow,*

$$|S^j|\frac{\partial L_p^j(f)}{\partial v^i} - \sum_{h\in S^j\setminus\{j\}}\frac{\partial L_p^h(f)}{\partial v^i} = (|S^j|+1)\sum_{e\in p} a_e\frac{\partial f_e^j}{\partial v^i} + \sum_{e\in p} a_e\frac{\partial\sum_{h:E^h(f)\subset E^j(f)} f_e^h}{\partial v^i} .$$

Proof. Given players i and h,

$$\frac{\partial L_p^h(f)}{\partial v^i} = \sum_{e\in p} a_e\frac{\partial(f_e + f_e^h)}{\partial v^i} . \tag{5}$$

Summing (5) over all players h in $S^j\setminus\{j\}$ and subtract it from $|S^j|$ times (5) for player j gives the proof. $\square$

Let $G^j = (V, E^j)$. By definition, all players $h \in S^j$ have flow on every s-t path in this graph. Lemma 15 implies that for any s-t paths p, q in G^j and any player $h \in S^j$, $\frac{\partial L_p^h(f)}{\partial v^i} = \frac{\partial L_q^h(f)}{\partial v^i}$. The expression on the left hand side of Claim 17 is thus equal for any path $p \in \mathcal{P}^j$, and therefore so is the expression on the right.

For the base case $j = k$, the set $\{h : E^h(f) \subset E^j(f)\}$ is empty. Hence, the second term on the right of Claim 17 is zero, and by the previous discussion, the quantity $\sum_{e\in p} a_e\frac{\partial f_e^k}{\partial v^i}$ is equal for any path $p \in \mathcal{P}^k$. Define $c(e) = a_e\frac{\partial f_e^k}{\partial v^i}$ for each $e \in E^k$ and $\kappa = \sum_{e\in p} a_e\frac{\partial f_e^k}{\partial v^i}$ for any s-t path p in G^k. By Lemma 16, $\sum_{e\in E^j(f)} c(e)\frac{\partial f_e^k}{\partial v^i} = \sum_{e\in E^j(f)} a_e\left(\frac{\partial f_e^k}{\partial v^i}\right)^2 = 0$. Hence, $\frac{\partial f_e^k}{\partial v^i} = 0$, $\forall e \in E$.

For the induction step $j < k$, due to the inductive hypothesis, $\frac{\partial f_e^h}{\partial v^i} = 0$ for $h > j$. Since by the nesting property if $E^h(f) \subset E^j(f)$ then $h > j$, the second term on the right of Claim 17 is again zero. By the same argument as in the base case, $\frac{\partial f_e^j}{\partial v^i} = 0$, for each $e \in E$, proving the lemma. $\square$

3.2 Proof of Lemma 11

An unstated assumption for all lemmas in this section is that the nesting property holds. For the proof of Lemma 11, our first step is to express the rate of change of total delay in terms of the rate of change of marginal delay of the players, as the flow value of player r is being decreased. The next lemma gives this expression for the first $r-1$ players.

Lemma 18. *For $f = f(\boldsymbol{v})$, and for each $i < r$, $\frac{\partial C(f)}{\partial v^i} = L^i(f) + \frac{\partial L^i(f)}{\partial v^i} \frac{\sum_{j=2}^k v^j}{2}$.*

Proof. For any player j, the set of edges used by player j is a subset of the edges used by player $i < r$, since player i has the largest flow value and we assume that the nesting property holds. Hence, the total delay at equilibrium $C(f) = \sum_{e \in E^i(f)} C_e(f_e)$.

$$\begin{aligned}\frac{\partial C(f)}{\partial v^i} &= \sum_{e \in E^i(f)} \frac{\partial C_e(f_e)}{\partial f_e} \frac{\partial f_e}{\partial v^i} = \sum_{e \in E^i(f)} (2a_e f_e + b_e) \frac{\partial f_e}{\partial v^i} \\ &= \sum_{e \in E^i(f)} \frac{\partial f_e}{\partial v^i} \left(L_e^i(f_e^i, f_e) + a_e \sum_{j \neq i} f_e^j \right) . \qquad (6)\end{aligned}$$

By Lemma 16 with $c(e) = L_e^i(f_e^i, f_e)$ and $\kappa = L^i(f)$, $\sum_{e \in E^i} L_e^i(f_e^i, f_e) \frac{\partial f_e}{\partial v^i} = L^i(f)$. Thus, $\dfrac{\partial C(f)}{\partial v^i} = L^i(f) + \displaystyle\sum_{j \neq i} \sum_{e \in E^i} a_e f_e^j \frac{\partial f_e}{\partial v^i}$.

By (4), we have that $a_e \frac{\partial f_e}{\partial v^i} = \frac{1}{2} a_e \frac{\partial (f_e + f_e^i)}{\partial v^i} = \frac{1}{2} \frac{\partial L_e^i(f_e^i, f_e)}{\partial v^i}$. It follows that

$$\begin{aligned}\frac{\partial C(f)}{\partial v^i} &= L^i(f) + \frac{1}{2} \sum_{j \neq i} \sum_{e \in E^i} f_e^j \frac{\partial L_e^i(f_e^i, f_e)}{\partial v^i} \\ &= L^i(f) + \frac{1}{2} \sum_{j \neq i} \sum_{e \in E^i} \sum_{q \in \mathcal{P}^i : e \in q} f_q^j \frac{\partial L_e^i(f_e^i, f_e)}{\partial v^i} ,\end{aligned}$$

where the last equality is because for any player j, $f_e^j = \sum_{q \in \mathcal{P}^j : e \in q} f_q^j = \sum_{q \in \mathcal{P}^i : e \in q} f_q^j$, and the nesting property. Reversing the order of summation and observing that $\sum_{e \in p : p \in \mathcal{P}^i} \frac{\partial L_e^i(f_e^i, f_e)}{\partial v^i} = \frac{\partial L^i(f)}{\partial v^i}$ and $v^i = v^1$, we have the required expression. □

We obtain a similar expression for $\frac{\partial C(f)}{\partial v^r}$.

Lemma 19. *Let $f = f(\boldsymbol{v})$. For player r whose flow value decreases in Step 2,*

$$\begin{aligned}\frac{\partial C(f)}{\partial v^r} &= L^1(f) + \frac{r-1}{r+1} \left(\frac{\partial L^1(f)}{\partial v^r} \sum_{i=r}^k v^i \right) + \frac{1}{r+1} \left(\frac{\partial L^r(f)}{\partial v^r} \sum_{i=r}^k v^i \right) \\ &\quad + (r-2) \left(\sum_{e \in E^1} a_e f_e^1 \frac{\partial f_e}{\partial v^r} \right) . \qquad (7)\end{aligned}$$

Let $\mathcal{P}$ denote the set of all s-t paths in G, and for equilibrium flow f, let $\{f_p^i\}_{p\in\mathcal{P},i\in[k]}$ denote a path flow decomposition of f. For players $i,j \in [r]$ with player r defined as in the flow redistribution, we will be interested in the rate of change of marginal delay of player i along an s-t path p as the value of flow controlled by player j changes. Given a decomposition $\{f_p^i\}_{p\in\mathcal{P},i\in[k]}$ along paths of the equilibrium flow, this rate of change can be expressed as

$$\begin{aligned}\frac{\partial L_p^i(f)}{\partial v^j} &= \sum_{e\in p} a_e \frac{\partial(f_e + f_e^i)}{\partial v^j} = \sum_{e\in p} a_e \sum_{q\in\mathcal{P}:e\in q} \frac{\partial(f_q + f_q^i)}{\partial v^j} \\ &= \sum_{q\in\mathcal{P}} \frac{\partial(f_q + f_q^i)}{\partial v^j} \sum_{e\in q\cap p} a_e \ . \end{aligned} \tag{8}$$

Let $u_{pq} = \sum_{e\in p\cap q} a_e$ for any paths $p,q \in \mathcal{P}$ and the matrix $\mathbf{U}$ is defined as the matrix of size $|\mathcal{P}| \times |\mathcal{P}|$ with entries $[u_{pq}]_{p,q\in\mathcal{P}}$.

Lemma 20. *For an equilibrium flow f, there exists a generalized path flow decomposition $\{f_p^i\}_{p\in\overline{\mathcal{P}}^i,i\in[k]}$ so that $\overline{\mathcal{P}}^i \subseteq \mathcal{P}^i$ for all $i\in[k]$ and $\overline{\mathcal{P}}^1 \supseteq \overline{\mathcal{P}}^2 \supseteq \cdots \overline{\mathcal{P}}^k$. Moreover, each of the submatrices $\mathbf{U}^i = [u_{pq}]_{p,q\in\overline{\mathcal{P}}^i}$ of $\mathbf{U}$ is invertible, $\forall i \in [k]$.*

Since $\overline{\mathcal{P}}^i \subseteq \overline{\mathcal{P}}^{i-1}$, we can arrange the rows and columns of $\mathbf{U}$ so that $\mathbf{U}^i$ is a leading principal submatrix of $\mathbf{U}$ for every player i.

Since matrix $\mathbf{U}^i$ is invertible, we define $\mathbf{W}^i = \mathbf{U}^{-1}$. For a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, we use $\mathbf{A}_p$ to refer to the pth row vector and a_{pq} to refer to the entry in the pth row and qth column. We define $\|\mathbf{A}\| = \sum\limits_{i\in[m],j\in[n]} a_{ij}$.

Lemma 21. *For equilibrium flow f and sets $\overline{\mathcal{P}}^i \subseteq \mathcal{P}$ as described in Lemma 20, for all players $i\in[k]$, $\|\mathbf{W}^i\| \geq \|\mathbf{W}^{i+1}\|$ and $\|\mathbf{W}^k\| > 0$.*

The next lemma gives the rate of change of marginal delay at equilibrium.

Lemma 22. *For player r defined as in the flow redistribution process and any player $i < r$, for $f = f(\boldsymbol{v})$,*

(i) $\dfrac{\partial L^i(f(\boldsymbol{v}))}{\partial v^i} = \dfrac{2}{\|\mathbf{W}^i\|}$,

(ii) $\dfrac{\partial L^i(f)}{\partial v^r} = \dfrac{1}{\|\mathbf{W}^i\|}$,

(iii) $\dfrac{\partial L^r(f)}{\partial v^r} = \dfrac{r+1}{r}\dfrac{1}{\|\mathbf{W}^r\|} + \dfrac{r-1}{r}\dfrac{1}{\|\mathbf{W}^1\|}$.

If we have just two players, it follows by substituting $i = 1$ and $r = 2$ and the expressions from Lemma 22 into Lemma 18 and Lemma 19 that $\frac{\partial C(f)}{\partial v^2} - \frac{\partial C(f)}{\partial v^1} = \frac{1}{2}v^2\left(\frac{1}{\|\mathbf{W}^2\|} - \frac{1}{\|\mathbf{W}^1\|}\right)$. By Lemma 21, $\|\mathbf{W}^1\| \geq \|\mathbf{W}^2\|$, and hence $\frac{\partial C(f)}{\partial v^2} - \frac{\partial C(f)}{\partial v^1} \geq 0$, proving Lemma 11 for the case of two players. However, if we have more than two players, when $r \neq 2$ the fourth term on the right hand side of (7) has nonzero contribution. Calculating this term is complicated. However, we show the following inequality for this expression.

Lemma 23. *For $f = f(v)$ and the player r as defined in the flow redistribution process,* $\sum_{e \in E^1} a_e f_e^1 \frac{\partial f_e}{\partial v^r} \geq \frac{v^1}{\|\mathbf{W}^1\|} - \frac{v^r}{r}\left(\frac{1}{\|\mathbf{W}^1\|} - \frac{1}{\|\mathbf{W}^r\|}\right)$.

Proof of Lemma 11. For any player $i < r$, substituting the expression for $\frac{\partial L^i(f)}{\partial v^i}$ from Lemma 22 into Lemma 18, and observing that $L^i(f) = L^1(f)$ and $\|\mathbf{W}^i\| = \|\mathbf{W}^1\|$ since the flow of the first $r-1$ players is identical,

$$\frac{\partial C(f)}{\partial v^i} = L^1(f) + \frac{\sum_{j=2}^k v^j}{\|\mathbf{W}^1\|} \ . \tag{9}$$

Similarly, substituting from Lemmas 22 and 23 into Lemma 19 and simplifying,

$$\frac{\partial C(f)}{\partial v^r} \geq L^1(f) + \frac{\sum_{i=2}^k v^i}{\|\mathbf{W}^1\|} + \frac{1}{r}\left(\frac{1}{\|\mathbf{W}^r\|} - \frac{1}{\|\mathbf{W}^1\|}\right)\left(\sum_{i=2}^k v^i - (r-2)(v^1 - v^r)\right). \tag{10}$$

We subtract (9) from (10) to obtain, for any player $i < r$,

$$\frac{\partial C(f)}{\partial v^r} - \frac{\partial C(f)}{\partial v^i} \geq \frac{1}{r}\left(\frac{1}{\|\mathbf{W}^r\|} - \frac{1}{\|\mathbf{W}^1\|}\right)\left(\sum_{i=2}^k v^i - (r-2)(v^1 - v^r)\right) \ . \tag{11}$$

From Lemma 21 we know that $\|\mathbf{W}^1\| \geq \|\mathbf{W}^r\|$. Also, $\sum_{i=2}^k v^i = (r-2)v^1 + \sum_{i=r}^k v^i \geq (r-2)(v^1 - v^r)$. Hence, the expression on the right of (11) is nonnegative, completing the proof. □

4 Convex Delays on Series-Parallel Graphs

Let $\mathcal{C}$ denote the class of continuous, differentiable, nondecreasing and convex functions. In this section we prove the following result.

Theorem 24. *The price of collusion on a series-parallel graph with delay functions taken from the set $\mathcal{C}$ is at most the price of collusion with linear delay functions.*

This theorem combined with Theorem 8, suffices to prove Theorem 1. The following lemma is proved by Milchtaich.[1]

[1] Milchtaich in fact shows the same result for *undirected* series-parallel graphs. In our context, every simple s-t path in the underlying undirected graph is also an s-t path in the directed graph G.

Lemma 25 ([14]). *Let $(G,v,\boldsymbol{l},s,t)$ and $(G,\tilde{v},\tilde{\boldsymbol{l}},s,t)$ be nonatomic routing games on a directed series-parallel graph with terminals s and t, where $v \geq \tilde{v}$, and $\forall x \in \mathbb{R}^+$ and $e \in E$, $l_e(x) \geq \tilde{l}_e(x)$. Let f and $\tilde{f}$ be equilibrium flows for the games with delays l and $\tilde{l}$ respectively. Then $C(f) \geq \tilde{C}(\tilde{f})$.*

We now use Lemma 25 to prove Theorem 24.

Proof of Theorem 24. Given a series-parallel graph G with delay functions $\boldsymbol{l}$ taken from $\mathcal{C}$, let g denote the atomic equilibrium flow and f denote the nonatomic equilibrium. We define a set of linear delay functions $\tilde{\boldsymbol{l}}$ as follows. For an edge, $\tilde{l}_e(x) = a_e x + b_e$, where $a_e = \frac{\partial l_e(f_e)}{\partial f_e}\Big|_{f_e = g_e}$ and $b_e = l_e(g_e) - a_e g_e$. Hence, the delay function $\tilde{l}_e$ is the tangent to the original delay function at the atomic equilibrium flow. Note that a convex continuous differentiable function lies above all of its tangents.

Let $\tilde{g}$ and $\tilde{f}$ denote the atomic and nonatomic equilibrium flows respectively with delay functions $\tilde{\boldsymbol{l}}$. Then by the definition of $\tilde{\boldsymbol{l}}$, $\tilde{g} = g$ and $\tilde{\boldsymbol{l}}(\tilde{g}) = \boldsymbol{l}(g)$. Hence, $\tilde{C}(\tilde{g}) = C(g)$. Further, by Lemma 25, $C(f) \geq \tilde{C}(\tilde{f})$. Since $\frac{C(g)}{C(f)} \leq \frac{\tilde{C}(\tilde{g})}{\tilde{C}(\tilde{f})}$, the proof follows.

5 Total Delay without the Nesting Property

If the nesting property does not hold, the total delay can increase as we decrease the flow of a smaller player and increase the flow of a larger player, thus causing our flow redistribution strategy presented in Section 3.2 to break down. See the full version for an example.

References

1. Ahuja, R.K., Magnanti, T.L., Orlin, J.B.: Network Flows. Prentice-Hall, Englewood Cliffs (1993)
2. Altman, E., Basar, T., Jimenez, T., Shimkin, N.: Competitive routing in networks with polynomial costs. IEEE Transactions on Automatic Control 47(1), 92–96 (2002)
3. Awerbuch, B., Azar, Y., Epstein, A.: The price of routing unsplittable flow. In: STOC, pp. 57–66. ACM, New York (2005)
4. Beckmann, M., McGuire, C.B., Winsten, C.B.: Studies in the Economics of Transportation. Yale University Press, New Haven (1956)
5. Bhaskar, U., Fleischer, L., Hoy, D., Huang, C.-C.: Equilibria of atomic flow games are not unique. In: SODA, pp. 748–757 (2009)
6. Bhaskar, U., Fleischer, L., Huang, C.-C.: The price of collusion in series-parallel networks (unpublished manuscript 2010)
7. Catoni, S., Pallottino, S.: Traffic equilibrium paradoxes. Transportation Science 25(3), 240–244 (1991)
8. Cominetti, R., Correa, J.R., Stier-Moses, N.E.: The impact of oligopolistic competition in networks. Operations Research 57(6), 1421–1437 (2009)

9. Correa, J.R., Schulz, A.S., Stier-Moses, N.E.: On the inefficiency of equilibria in congestion games. In: Jünger, M., Kaibel, V. (eds.) IPCO 2005. LNCS, vol. 3509, pp. 167–181. Springer, Heidelberg (2005)
10. Fotakis, D., Spirakis, P.: Selfish unsplittable flows. In: TCS, pp. 593–605. Springer, Heidelberg (2004)
11. Harker, P.: Multiple equilibrium behaviors on networks. Transportation Science 22(1), 39–46 (1988)
12. Harks, T.: Stackelberg strategies and collusion in network games with splittable flow. In: Bampis, E., Skutella, M. (eds.) WAOA 2008. LNCS, vol. 5426, pp. 133–146. Springer, Heidelberg (2009)
13. Hayrapetyan, A., Tardos, É., Wexler, T.: The effect of collusion in congestion games. In: STOC, pp. 89–98. ACM Press, New York (2006)
14. Milchtaich, I.: Network topology and the efficiency of equilibrium. Games and Economic Behavior 57(2), 321–346 (2006)
15. Nocedal, J., Wright, S.T.: Numerical Optimization. Springer, Heidelberg (2006)
16. Orda, A., Rom, R., Shimkin, N.: Competitive routing in multiuser communication networks. IEEE/ACM Transactions on Networking 1(5), 510–521 (1993)
17. Rosen, J.B.: Existence and uniqueness of equilibrium points for concave n-person games. Econometrica 33(3), 520–534 (1965)
18. Rosenthal, R.W.: A class of games possessing pure-strategy nash equilibria. Intl. J. of Game Theory 2, 65–67 (1973)
19. Roughgarden, T.: Selfish routing with atomic players. In: SODA, pp. 1184–1185 (2005)
20. Roughgarden, T.: The price of anarchy is independent of the network topology. J. Comput. Syst. Sci. 67(2), 341–364 (2003)
21. Roughgarden, T.: Selfish Routing and the Price of Anarchy. The MIT Press, Cambridge (2005)
22. Roughgarden, T., Tardos, E.: How bad is selfish routing? Journal of the ACM 49(2), 236–259 (2002)
23. Roughgarden, T., Tardos, E.: Bounding the inefficiency of equilibria in nonatomic congestion games. Games and Economic Behavior 47, 389–403 (2004)
24. Wardrop, J.G.: Some theoretical aspects of road traffic research. In: Proc. Institute of Civil Engineers, Pt. II, vol. 1, pp. 325–378 (1952)

The Chvátal-Gomory Closure of an Ellipsoid Is a Polyhedron

Santanu S. Dey[1] and Juan Pablo Vielma[2,3]

[1] H. Milton Stewart School of Industrial and Systems Engineering,
Georgia Institute of Technology, USA
santanu.dey@isye.gatech.edu
[2] Business Analytics and Mathematical Sciences Department
IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
[3] Department of Industrial Engineering
University of Pittsburgh, Pittsburgh, PA, USA
jvielma@pitt.edu

Abstract. It is well-know that the Chvátal-Gomory (CG) closure of a rational polyhedron is a rational polyhedron. In this paper, we show that the CG closure of a bounded full-dimensional ellipsoid, described by rational data, is a rational polytope. To the best of our knowledge, this is the first extension of the polyhedrality of the CG closure to a non-polyhedral set. A key feature of the proof is to verify that all non-integral points on the boundary of ellipsoids can be separated by some CG cut. Given a point u on the boundary of an ellipsoid that cannot be trivially separated using the CG cut parallel to its supporting hyperplane, the proof constructs a sequence of CG cuts that eventually separates u. The polyhedrality of the CG closure is established using this separation result and a compactness argument. The proof also establishes some sufficient conditions for the polyhedrality result for general compact convex sets.

1 Introduction

Nonlinear Integer Programming has received significant attention from the Integer Programming (IP) community in recent time. Although, some special classes are efficiently solvable [32], even simple nonlinear IP problems can be NP-Hard or undecidable [33]. However, there has been considerable progress in the development of practical algorithms that can be effective for many important applications (e.g. [1,8,9,10,32,36,37]). Building on work for linear IP, practical algorithms for nonlinear IP have benefited from the development of several classes of cutting planes or valid inequalities (e.g. [3,4,5,6,13,14,25,29,30,31,35,28,39,40,43]). Many of these inequalities are based on the generalization of ideas used in linear IP. For example, [4,5,39,14] exploit the interaction between superadditive functions and nonlinear constraints to develop techniques that can yield several strong valid inequalities.

Following the success of such approaches we study some theoretical properties of this interaction when the superadditive function is the integer round down operation $\lfloor \cdot \rfloor$ and the nonlinear constraints are convex. Specifically we study the polyhedrality of the (first) Chvátal-Gomory (CG) closure [15,26,27,41] of a non-polyhedral convex set. The study of properties of the CG closure of a rational polyhedron has yielded many well known results for linear IP. In this case, the closure is a rational polyhedron [41] for which the associated optimization, separation and membership problems are NP-hard even for restricted cases [11,12,21,34]. However, optimization over the CG closure of a polyhedron has been successfully used to show its strength computationally [22,23]. Similar results have also been obtained for closures associated to other valid inequalities such as split cuts [2,7,12,17,19,20,44].

CG cuts for non-polyhedral sets are considered implicitly in [15,41] and explicitly in [14], but only [41] deals with the polyhedrality of the CG closure. Although [41] shows that for rational polyhedra the closure is a rational polyhedron, the result does not automatically extend to non-polyhedral sets. Furthermore, neither of the known proofs of the result for rational polyhedra [16,41,42] can be easily adapted to consider other convex sets. In fact, as noted in [41] even the polyhedrality of the CG closure of non-rational polytopes remains unknown. Because of this, we study the polyhedrality of the CG closure of an ellipsoid as the first natural step towards understanding the closure of other non-polyhedral convex sets.

Let a *rational ellipsoid* be the image of an Euclidean ball under a rational affine transformation. Our main result is to show that the CG closure of a full-dimensional bounded rational ellipsoid is a rational polytope. To the best of our knowledge, this is the first extension to a non-polyhedral set of the well known result for rational polyhedra. Additionally, the proof of our main result reveals some general sufficient conditions for the polyhedrality of the CG closure and other interesting properties. For example, we show that every non-integral point on the boundary of an ellipsoid can be separated by a CG cut. We recently verified [18] that this geometrically natural property holds for some other classes of convex sets.

The rest of the paper is organized as follows. In Section 2, we give some background on CG cuts, formally state the main result of the paper and present an outline of its proof. In Section 3, we present notation and review some standard results from convex analysis. In Section 4, we consider two separation results that are needed for the proof of the main theorem, which we present in Section 5. We end with some remarks in Section 6.

2 Background, Main Result and Proof Outline

For a polyhedron $P \subset \mathbb{R}^n$, the CG cutting plane procedure [15,26,27] can be described as follows. For an integer vector $a \in \mathbb{Z}^n$, let $d \in \mathbb{R}$ be such that $\{x \in \mathbb{R}^n : \langle a, x \rangle \leq d\} \supset P$ where $\langle u, v \rangle$ is the inner product between u and v. We then have that $P_I := P \cap \mathbb{Z}^n \subset \{x \in \mathbb{R}^n : \langle a, x \rangle \leq \lfloor d \rfloor\}$ and hence the CG cut $\langle a, x \rangle \leq \lfloor d \rfloor$ is a valid inequality for $conv(P_I)$. The first CG closure P^1 of P is defined as the convex set obtained by adding all possible CG cuts to P. If P is a rational polyhedron, then P^1 is also a polyhedron [41] and hence we can recursively define the k-th CG closure P^k of P as the first CG closure of P^{k-1}. Furthermore, for any rational polyhedron P we have that there exists $k \in \mathbb{Z}_+$ such that $P^k = conv(P_I)$ [15,41]. Non-rational polytopes are also considered in [15] and the CG procedure is extended to the feasible region of Conic Programming (CP) problems in [14]. In fact, the CG procedure can be extended to, at least, any compact convex set as follows.

Let $C \subset \mathbb{R}^n$ be a compact convex set and let $\sigma_C(a) := \sup_{x \in C} \langle a, x \rangle$ be its support function so that $C = \bigcap_{a \in \mathbb{R}^n} \{x \in \mathbb{R}^n : \langle a, x \rangle \leq \sigma_C(a)\}$. Because $\mathbb{Q}^n$ is dense in $\mathbb{R}^n$ and $\sigma_C(a)$ is positively homogeneous and continuous, it can be verified that $C = \bigcap_{a \in \mathbb{Z}^n} \{x \in \mathbb{R}^n : \langle a, x \rangle \leq \sigma_C(a)\}$.

Definition 1. *For any $S \subset \mathbb{Z}^n$, let $CC(S, C) := \bigcap_{a \in S} \{x \in \mathbb{R}^n : \langle a, x \rangle \leq \lfloor \sigma_C(a) \rfloor\}$. We recursively define the k-th CG closure C^k of C as $C^1 := CC(\mathbb{Z}^n, C)$ and $C^{k+1} := CC(\mathbb{Z}^n, C^k)$ for all $k > 1$.*

The definition is consistent because C^1 is a closed convex set contained in C and when C is a polyhedron it coincides with the traditional definition. Furthermore, $C_I := C \cap \mathbb{Z} \subset C^k$ for all k and, as noted in [41], the following theorem follows from [15,41].

Theorem 1 ([15,41]). *There exist k such that $C^k = conv(C_I)$.*

Theorem 1 is also shown in [14] for CP problems with bounded feasible regions. However, the result neither implies nor requires the polyhedrality of C^1. In fact, the original proof of Theorem 1 in [15] does not use the polyhedrality of either P or P^1. Although surprising, it could be entirely possible for Theorem 1 to hold and for C^k to be the only polyhedron in the hierarchy $\{C^l\}_{l=1}^k$. Our main result is the first proof of the polyhedrality of C^1 for a non-polyhedral set C.

Theorem 2 (Main Theorem). *Let T be a full-dimensional bounded rational ellipsoid. Then $CC(\mathbb{Z}^n, T)$ is a rational polytope.*

Before presenting an outline of our proof of Theorem 2, we discuss why some of the well-known polyhedrality proofs and results do not easily extend to ellipsoids. We begin by noting that it is not clear how to extend the polyhedrality proofs in [16,41,42] beyond rational polyhedra because they rely on properties that are characteristic of these sets such as TDI systems and finite integral generating sets. However, we could try to prove Theorem 2 by using the polyhedrality of the first CG closure of polyhedral approximations of T. One natural scheme could be to attempt constructing a sequence of rational polytope pairs $\{P_i, Q_i\}_{i \in \mathbb{N}}$ such that (i) $P_i \cap \mathbb{Z}^n = Q_i \cap \mathbb{Z}^n = T \cap \mathbb{Z}^n$, (ii) $P_i \subset T \subset Q_i$ and (iii) $Vol(Q_i \setminus P_i) \leq 1/i$. We then would have that

$$P_i^k \subset T^k \subset Q_i^k, \tag{1}$$

for all $i, k \geq 1$. As noted in [41], using this approach Theorem 1 in general follows directly from Theorem 1 for rational polytopes. Unfortunately, it is not clear how to show that there exists i such that (1) holds as equality for $k = 1$ without knowing a priori that T^1 is a polyhedron. Finally, we note that cut domination arguments commonly used in polyhedrality proofs of closures do not seem to adapt well to the proof of Theorem 2.

Due of the reasons stated above, to prove Theorem 2 we resort to a different approach that relies on being able to separate with a CG cut every non-integral point on the boundary of T. Specifically, we show that $CC(\mathbb{Z}^n, T)$ can be generated with the procedure described in Figure 1.

Step 1 Construct a polytope Q defined by a finite number of CG cuts such that:
- $Q \subset T$.
- $Q \cap bd(T) \subset \mathbb{Z}^n$.

Step 2 Update Q with a CG cut that separates a point of $Q \setminus CC(\mathbb{Z}^n, T)$ until no such cut exists.

Fig. 1. A procedure to generate the first CG closure for ellipsoid

To show that Step 1 can be accomplished, we first show that every non-integral point on the boundary of T can be separated by a CG cut. If there are no integral points on the boundary of T, then this separation result allows us to cover the boundary of T with a possibly infinite number of open sets that are associated to the CG cuts. We then use compactness of the boundary of T to obtain a finite sub-cover that yields a finite number of CG cuts that separate every point on the boundary of T. If there are integer points on the boundary, then we use a stronger separation result and a similar argument to show that there is a finite set of CG cuts that separate every non-integral point on the boundary of T.

To show that Step 2 terminates finitely, we simply show that the set of CG cuts that separate at least one point in $Q \setminus CC(\mathbb{Z}^n, T)$ is finite.

We note that the separation of non-integral points using CG cuts on the boundary of T, required in Step 1 of Figure 1, is not straightforward. A natural first approach to separate a non-integral point u on the boundary of T is to take an inequality $\langle a, x\rangle \leq \sigma_T(a)$ that is supporting at u, scale it so that $a \in \mathbb{Z}^n$, and then generate the CG cut $\langle a, x\rangle \leq \lfloor \sigma_T(a) \rfloor$. If $\sigma_T(a) \notin \mathbb{Z}$, then the CG cut will separate u because a was selected such that $\langle a, u\rangle = \sigma_T(a)$. Unfortunately, as the following examples show, this approach can fail either because a cannot be scaled to be integral or because $\sigma_T(a) \in \mathbb{Z}$ for any scaling that yields $a \in \mathbb{Z}^n$.

Example 1. Let $T := \{x \in \mathbb{R}^2 \mid \sqrt{x_1^2 + x_2^2} \leq 1\}$ and $u = (1/2, \sqrt{3}/2)^T \in bd(T)$. We have that the supporting inequality for u is $a_1x_1 + a_2x_2 \leq \sigma_T(a)$ where $a = u$. Since u is irrational in only one component, observe that a cannot be scaled to be integral.

For Example 1, it is easy to see that selecting an alternative integer left-hand-side vector a' resolves the issue. We can use $a' = (1, 1)$ which has $\sigma_T(a') = \sqrt{2}$ to obtain the CG cut $x_1 + x_2 \leq 1$. In Example 1 this CG cut separates every non-negative non-integral point on the boundary of T. In Section 4, we will show that given any non-integral point u on the boundary of T such that the left-hand-side of its supporting hyperplane cannot be scaled to be integral, there exists an alternative left-hand-side integer vector a' such that the CG cut $\langle a', x\rangle \leq \lfloor \sigma_T(a')\rfloor$ separates u. This vector a' will be systematically obtained using simultaneous diophantine approximation of the left-hand-side of an inequality describing the supporting hyperplane at u.

Example 2. Let $T := \{x \in \mathbb{R}^2 \,|\, \sqrt{x_1^2 + x_2^2} \leq 5\}$ and $u = (25/13, 60/13)^T \in bd(T)$. We have that the supporting inequality for u can be scaled to $a_1x_1 + a_2x_2 \leq \sigma_T(a)$ for $a = (5, 12)^T$ which has $\sigma_T(a) = 65$. Because 5 and 12 are coprime and $\sigma_T(\cdot)$ is positively homogeneous, a cannot be scaled so that $a \in \mathbb{Z}^2$ and $\sigma_T(a) \notin \mathbb{Z}$.

Observe that Example 2 is not an isolated case. In fact, these cases are closely related to primitive Pythagorean triples. For $T := \{x \in \mathbb{R}^2 \,|\, \sqrt{x_1^2 + x_2^2} \leq r\}$, select any primitive Pythagorean triple $v_1^2 + v_2^2 = v_3^2$, and consider the point $r(\frac{v_1}{v_3}, \frac{v_2}{v_3})$ (such that $r(\frac{v_1}{v_3}, \frac{v_2}{v_3}) \notin \mathbb{Z}^2$). Then since v_1 and v_2 are coprimes, the behavior in Example 2 will be observed. Also note that these examples are not restricted only to Euclidean balls in $\mathbb{R}^2$, since it is easy to construct integers $a_1, ..., a_n, a_{n+1}$ such that $\sum_{i=1}^n a_i^2 = a_{n+1}^2$ (e.g. $3^2 + 4^2 + 12^2 = 13^2$). For the class of points $u \in bd(T)$ where the left-hand-side of an inequality describing the supporting hyperplane is scalable to an integer vector a, we will show in Section 4 that there exists a systematic method to obtain $a' \in \mathbb{Z}^n$ such that $\langle a', x\rangle \leq \lfloor \sigma_T(a')\rfloor$ separates u.

3 Notation and Standard Results from Convex Analysis

In this paper we consider an ellipsoid given by a non-singular and surjective rational linear transformation of an Euclidean ball followed by a rational translation. Without loss of generality, we may assume that that this ellipsoid is described as $T := \{x \in \mathbb{R}^n : \gamma_B(x - c) \leq 1\}$ where $c \in \mathbb{Q}^n$, and $\gamma_B(x) := \|Ax\|$ is the gauge of $B := \{x \in \mathbb{R}^n : \|Ax\| \leq 1\}$ such that $A \in \mathbb{Q}^{n\times n}$ is a symmetric positive definite matrix. Then T is the translation by c of B. The set B is a full dimensional compact convex set with the zero vector in its interior and hence has the following properties.

- The support function of B is $\sigma_B(a) = \|A^{-1}a\|$.
- The polar of B given by $B^\circ := \{a \in \mathbb{R}^n \,|\, \langle a, x\rangle \leq 1 \quad \forall x \in B\} = \{a \in \mathbb{R}^n \,|\, \sigma_B(a) \leq 1\}$ is a full-dimensional and compact convex set.
- For any $u \in bd(B)$ we have that $s_B(u) := A^T A(u)$ is such that $\langle s_B(u), u\rangle = \sigma_B(s_B(u)) = 1$ and hence $\langle s_B(u), x\rangle \leq 1 = \sigma_B(s_B(u))$ is a valid inequality for B that is supporting at u.
- $\langle a, x\rangle \leq \sigma_B(a)\gamma_B(x)$.
- The boundary of B is $bd(B) := \{x \in \mathbb{R}^n : \gamma_B(x) = 1\}$.

Because $T = B + c$ we also have the following properties of T.

- The support function of T is $\sigma_T(a) = \sigma_{B+c}(a) = \sigma_B(a) + \langle a, c\rangle = \|A^{-1}a\| + \langle a, c\rangle$.
- For any $u \in bd(T)$ we have that $s_T(u) := s_B(u - c) = A^T A(u - c)$ is such that $\langle s_T(u), u - c\rangle = \langle s_B(u - c), u - c\rangle = \sigma_B(s_B(u - c)) = \sigma_B(s(u)) = 1$ and hence $\langle s(u), x\rangle \leq 1 + \langle s(u), c\rangle = \sigma_T(s(u))$ is a valid inequality for T that is supporting at u.
- The boundary of T is $bd(T) := \{x \in \mathbb{R}^n : \gamma_B(x - c) = 1\}$.

To simplify the notation, we regularly drop the T from $\sigma_T(\cdot)$, $s_T(\cdot)$ and $CC(\cdot, T)$ so that $\sigma(\cdot) := \sigma_T(\cdot)$, $s(\cdot) := s_T(\cdot)$ and $CC(\cdot) := CC(\cdot, T)$. In addition, for $u \in \mathbb{R}$ we denote its fractional part by $F(u) := u - \lfloor u\rfloor$.

4 Separation

To prove Theorem 2 we need two separation results. The first one simply states that every non-integral point on the boundary of T can be separated by a CG cut.

Proposition 1. *If $u \in bd(T) \setminus \mathbb{Z}^n$, then there exists a CG cut that separates point u.*

An integer point $u \in bd(T) \cap \mathbb{Z}^n$ cannot be separated by a CG cut, but Proposition 1 states that every point in $bd(T)$ that is close enough to u will be separated by a CG cut. However, for the compactness argument to work we need a stronger separation result for points on the boundary that are close to integral boundary points. This second result states that all points in $bd(T)$ that are sufficiently close to an integral boundary point can be separated by a finite number of CG cuts.

Proposition 2. *Let $u \in bd(T) \cap \mathbb{Z}^n$. Then there exists $\varepsilon_u > 0$ and a finite set $W_u \subset \mathbb{Z}^n$ such that*

$$\langle w, u\rangle = \lfloor \sigma(w)\rfloor \quad \forall w \in W_u, \tag{2}$$

$$\forall v \in bd(T) \cap \{x \in \mathbb{R}^n : \|x - u\| < \varepsilon_u\} \setminus \{u\} \quad \exists w \in W_u \ s.t \ \langle w, v\rangle > \lfloor \sigma(w)\rfloor, \tag{3}$$

and

$$\forall v \in int(T) \quad \exists w \in W_u \ s.t. \ \langle w, v\rangle < \lfloor \sigma(w)\rfloor. \tag{4}$$

The main ideas used in the proof of Proposition 2 are as follows. First, it is verified that for any nonzero integer vector q, there exists a finite $i \in \mathbb{Z}_+$ such that the CG cut of the form $\langle q + i\lambda s(u), x\rangle \leq \lfloor \sigma(q + i\lambda s(u))\rfloor$ satisfies (2) (here $\lambda s(u) \in \mathbb{Z}^n$ for some scalar $\lambda \neq 0$). Second, it is verified that by carefully selecting a finite number of integer vectors and applying the above construction, all points in a sufficiently small neighborhood of u can be separated. Finally, (4) is established by adding the supporting hyperplane at u which is trivially a CG cut.

Although this proof of Proposition 2 is similar to the proof of Proposition 1, it is significantly more technical. We therefore refer the readers to [18] where a more general version of Proposition 2 is proven and confine our discussion to an outline of the proof of Proposition 1 here.

4.1 Outline of Proof of Proposition 1

To prove Proposition 1 we construct a separating CG cut for $u \in bd(T) \setminus \mathbb{Z}^n$ by modifying the supporting inequality for T at u. In the simplest case, we scale $\langle s(u), x \rangle \leq \sigma(s(u))$ by $\lambda > 0$ so that $\lambda s(u) \in \mathbb{Z}^n$, to obtain a CG cut $\langle \lambda s(u), x \rangle \leq \lfloor \sigma(\lambda s(u)) \rfloor$ that separates u. If this is not successful, then we approximate the direction $s(u)$ by a sequence $\{s^i\}_{i\in\mathbb{N}} \subset \mathbb{Z}^n$ such that $s^i/\|s^i\| \to s(u)/\|s(u)\|$ and for which $\langle s^i, x \rangle \leq \lfloor \sigma(s^i) \rfloor$ separates u for sufficiently large i. For this approach to work we will need a sequence that complies with the following two properties.

C1 $\lim_{i\to+\infty} \langle s^i, u \rangle - \sigma(s^i) = 0$
C2 $\lim_{i\to+\infty} F(\sigma(s^i)) = \delta > 0$. (A weaker condition like $\limsup_{i\to+\infty} F(\sigma(s^i)) > 0$ is sufficient, but we will verify the stronger condition).

Neither condition holds for every sequence such that $s^i/\|s^i\| \to s(u)/\|s(u)\|$. For instance, for $s(u) = (0,1)^T$ the sequence $s^i = (k, k^2)$ does not comply with condition C1. For these reason we need the following proposition.

Proposition 3. *Let $u \in bd(T) \setminus \mathbb{Z}^n$ and let e^l be the l-th unit vector for some $l \in \{1, \ldots, n\}$ such that $u_l \notin \mathbb{Z}$.*

(a) If there exists $\lambda > 0$ such that $p := \lambda s(u) \in \mathbb{Z}^n$ and $\sigma(\lambda s(u)) \in \mathbb{Z}$, then $s^i := e^l + ip$ complies with conditions C1 and C2.
(b) If $\lambda s(u) \notin \mathbb{Z}^n$ for all $\lambda > 0$, then let $\{(p^i, q_i)\}_{i\in\mathbb{N}} \subset \mathbb{Z}^n \times (\mathbb{Z}_+ \setminus \{0\})$ be the coefficients obtained using Dirichlet's Theorem to approximate $s(u)$. That is $\{(p^i, q_i)\}_{i\in\mathbb{N}}$ is such that

$$|q_i s(u)_j - p^i_j| < \frac{1}{i} \ \forall j \in \{1, ..., n\}.$$

For $M \in \mathbb{Z}_+$ such that $Mc \in \mathbb{Z}^n$ we have that $s^i := e^l + Mp^i$ complies with conditions C1 and C2.

With this proposition we can proceed to the proof of Proposition 1

Proof (Proof of Proposition 1). Let $u \in bd(T) \setminus \mathbb{Z}^n$. There are three possible cases:

1. There exists $\lambda > 0$ such that $\lambda s(u) \in \mathbb{Z}^n$ and $\sigma(\lambda s(u)) \notin \mathbb{Z}$.
2. There exists $\lambda > 0$ such that $\lambda s(u) \in \mathbb{Z}^n$ and $\sigma(\lambda s(u)) \in \mathbb{Z}$.
3. $\lambda s(u) \notin \mathbb{Z}^n$ for all $\lambda > 0$.

Case 1: $\langle \lambda s(u), x \rangle \leq \lfloor \sigma(\lambda s(u)) \rfloor$ is a CG cut that separates u.
Cases 2 and 3: From Proposition 3, we have that in both cases there exists a sequence $\{s^i\}_{i\in\mathbb{N}} \subset \mathbb{Z}^n$ satisfying conditions C1 and C2. Together with

$$\langle s^i, u \rangle - \lfloor \sigma(s^i) \rfloor = \langle s^i, u \rangle - \sigma(s^i) + F(\sigma(s^i)), \tag{5}$$

conditions C1 and C2 yields that for sufficiently large i we have $\langle s^i, u \rangle - \lfloor \sigma(s^i) \rfloor > 0$ and hence $\langle s^i, x \rangle \leq \lfloor \sigma(s^i) \rfloor$ separates u.

We next discuss the proof of Proposition 3 in the next two subsections.

Condition C1 in Proposition 3. Condition C1 is not difficult to satisfy. In fact, it is satisfied by any sequence for which the angle between s^i and s converges fast enough (e.g. if $\|s^i\| \to +\infty$, then C1 is satisfied if we have that $\left\|(s^i/\|s^i\|) - (s(u)/\|s(u)\|)\right\| \in o(1/\|s^i\|)$). For the specific sequences in Proposition 3 (a) and 3 (b) condition C1 can be verified using properties from Section 3 and the following lemma which we do not prove here.

Lemma 1. *Let $w \in \mathbb{R}^n$ and $\{v^i\}_{i\in\mathbb{N}} \subset \mathbb{R}^n$ be any sequence such that there exists $\mathcal{N} > 0$ for which*

$$|v^i_j w_k - v^i_k w_j| < \mathcal{N} \quad \forall i \in \mathbb{N},\, j,k \in \{1,\dots,n\},\, j \neq k \tag{6}$$

and $\lim_{i\to+\infty}\langle v^i, w\rangle = +\infty$. Then

$$\lim_{i\to+\infty} \langle v^i, w\rangle - \|v^i\|\|w\| = 0.$$

Condition C2 in Proposition 3. Condition C2 is much more interesting and showing that it holds for our specific sequences is the crux of the proof of Proposition 3. The intuition behind the proof is the following: For the sequence in Proposition 3 (a) we have $s^i = e^l + ip$. For large enough i, $\sigma(s^i) \approx \langle e^l + ip, u\rangle = u_l + i\langle \lambda s(u), u\rangle = u_l + i\sigma(\lambda s(u))$. Now since $\sigma(\lambda s(u))$ is integral, the fractional part of $\sigma(s^i)$ is therefore approximately equal to u_l. The formal proof is presented next. We first present a simple lemma.

Lemma 2. *Let $\alpha \in \mathbb{R}$, $t \in \mathbb{R}_+$ and $\{\beta_i\}_{i\in\mathbb{N}} \subset \mathbb{R}$ be such that $\lim_{i\to\infty}\beta_i = \infty$. Then, for every $\varepsilon > 0$ there exists N_ε such that*

$$\alpha + \beta_i \le \sqrt{(\alpha+\beta_i)^2 + t} \le \alpha + \beta_i + \varepsilon \quad \forall i \ge N_\varepsilon$$

Lemma 3. *The sequence in Proposition 3 (a) satisfies Condition C2.*

Proof. Let $\alpha = \langle A^{-1}e^l, A^{-1}p\rangle/\|A^{-1}p\|$, $\beta_i = i\|A^{-1}p\|$ and $t = \|A^{-1}e^l\|^2 - (\langle A^{-1}r^l, A^{-1}p\rangle/\|A^{-1}p\|)^2$. We have that $\lim_{i\to\infty}\beta_i = \infty$ because $\|A^{-1}p\| > 0$ and $t \ge 0$ by Cauchy-Schwarz inequality. Observe that,

$$\begin{aligned}\|A^{-1}s^i\| &= \sqrt{i^2\|A^{-1}p\|^2 + 2i\langle A^{-1}e^l, A^{-1}p\rangle + \|A^{-1}e^l\|^2} \\ &= \sqrt{\left(\frac{\langle A^{-1}e^l, A^{-1}p\rangle}{\|A^{-1}p\|} + i\|A^{-1}p\|\right)^2 + \|A^{-1}e^l\|^2 - \left(\frac{\langle A^{-1}e^l, A^{-1}p\rangle}{\|A^{-1}p\|}\right)^2} \\ &= \sqrt{(\alpha+\beta_i)^2 + t}.\end{aligned}$$

Then, by Lemma 2, we have that

$$\begin{aligned}\sigma(s^i) &= \sqrt{(\alpha+\beta_i)^2 + t} + \langle c, s^i\rangle \\ &\ge \frac{\langle A^{-1}e^l, A^{-1}p\rangle}{\|A^{-1}p\|} + i\|A^{-1}p\| + \langle c, e^l + ip\rangle \\ &= i\sigma(p) + \left\lfloor \frac{\langle A^{-1}e^l, A^{-1}p\rangle}{\|A^{-1}p\|} + \langle c, e^l\rangle \right\rfloor + F\left(\frac{\langle A^{-1}e^l, A^{-1}p\rangle}{\|A^{-1}p\|} + \langle c, e^l\rangle\right)\end{aligned}$$

and, similarly, for $i \geq N_\varepsilon$ we have

$$\sigma(s^i) \leq i\sigma(p) + \left\lfloor \frac{\langle A^{-1}e^l, A^{-1}p\rangle}{\|A^{-1}p\|} + \langle c, e^l\rangle \right\rfloor + F\left(\frac{\langle A^{-1}e^l, A^{-1}p\rangle}{\|A^{-1}p\|} + \langle c, e^l\rangle\right) + \varepsilon.$$

Hence, by setting $k := i\sigma(p) + \lfloor \langle A^{-1}e^l, A^{-1}p\rangle / \|A^{-1}p\| + \langle c, e^l\rangle \rfloor$ and $\delta := F(\langle A^{-1}e^l, A^{-1}p\rangle / \|A^{-1}p\| + \langle c, e^l\rangle)$ we have that

$$k + \delta \leq \sigma(s^i) \leq k + \delta + \varepsilon \quad \forall i \geq N_\varepsilon. \tag{7}$$

Now, $k \in \mathbb{Z}$ and

$$\begin{aligned}\delta &= F(\langle A^{-1}e^l, A^{-1}p\rangle / \|A^{-1}p\| + \langle c, e^l\rangle)\\ &= F(\langle A^{-1}e^l, \lambda A(u-c)\rangle / \lambda + \langle c, e^l\rangle)\\ &= F(\langle e^l, A^{-1}\lambda A(u-c)\rangle / \lambda + \langle c, e^l\rangle)\\ &= F(\langle e^l, u\rangle) = F(u_l) \in (0,1),\end{aligned}$$

because $u_l \notin \mathbb{Z}$ and because $p = \lambda s(u)$ implies $\|A^{-1}p\| = \lambda \|A^{-1}s(u)\| = \lambda$. Thus $\lim_{i\to+\infty} F(\sigma(s^i)) = \delta > 0$.

Lemma 4. *The sequence in Proposition 3 (b) satisfies Condition C2.*

Proof. Let $\bar{\epsilon}^i := p^i - q_i s(u)$, so that $\|\bar{\epsilon}^i\| \leq \frac{\sqrt{n}}{i}$. We then have that

$$\lim_{i\to+\infty} \|A^{-1}\bar{\epsilon}^i\| = \lim_{i\to+\infty} \|A^{-1}(-\bar{\epsilon}^i)\| = 0. \tag{8}$$

Now observe that

$$\begin{aligned}\|A^{-1}s^i\| &= \|A^{-1}(Mp^i + e^l)\|\\ &= \|Mq_iA^{-1}s(u) + A^{-1}e^l + MA^{-1}\bar{\epsilon}^i\|\\ &\leq \|Mq_iA^{-1}s(u) + A^{-1}e^l\| + M\|A^{-1}\bar{\epsilon}^i\|\\ &= \sqrt{\left(\frac{\langle A^{-1}s(u), A^{-1}e^l\rangle}{\|A^{-1}s(u)\|} + Mq_i\right)^2 + t} + M\|A^{-1}\bar{\epsilon}^i\|.\end{aligned}$$

where $t := \|A^{-1}e^l\|^2 - \left(\frac{\langle A^{-1}s(u), A^{-1}e^l\rangle}{\|A^{-1}s(u)\|}\right)^2$. Since $\|A^{-1}s(u)\| = \|A(u-c)\| = 1$, $t = \|A^{-1}e^l\|^2 - \langle A(u-c), A^{-1}e^l\rangle^2$ which is non-negative by the Chauchy-Schwartz inequality. Therefore by setting $\alpha := \frac{\langle A^{-1}s(u), A^{-1}e^l\rangle}{\|A^{-1}s(u)\|}$, $\beta_i := Mq_i$ we can use Lemma 2 and the fact that $\|A^{-1}s(u)\| = \|A(u-c)\| = 1$ to obtain that for every $\varepsilon > 0$ there exists N_ε such that

$$\|A^{-1}s^i\| \leq Mq_i + \langle A(u-c), A^{-1}e^l\rangle + \varepsilon + M\|A^{-1}\bar{\epsilon}^i\| \quad \forall i \geq N_\varepsilon. \tag{9}$$

Similarly, we also have that

$$\begin{aligned}
\|A^{-1}s^i\| &= \|A^{-1}(Mp^i + e^l)\| \\
&\geq \|MA^{-1}p^i + A^{-1}e^l - MA^{-1}\bar{\epsilon}^i\| - M\|A^{-1}(-\bar{\epsilon}^i)\| \\
&= \|Mq_iA^{-1}s(u) + A^{-1}e^l\| - M\|A^{-1}(-\bar{\epsilon}^i)\| \\
&= \sqrt{\left(\frac{\langle A^{-1}s(u), A^{-1}e^l\rangle}{\|A^{-1}s(u)\|} + Mq_i\right)^2 + t} - M\|A^{-1}(-\bar{\epsilon}^i)\| \\
&\geq Mq_i + \langle A(u-c), A^{-1}e^l\rangle - M\|A^{-1}(-\bar{\epsilon}^i)\| \\
&= Mq_i + \langle u-c, e^l\rangle - \|A^{-1}(-\bar{\epsilon}^i)\|. \qquad (10)
\end{aligned}$$

Combining (9) and (10) and using (8) and the definition of $\sigma(\cdot)$ we obtain that for every $\tilde{\varepsilon} > 0$ there exists $N_{\tilde{\varepsilon}}$ such that

$$Mq_i + \langle p^i, Mc\rangle + \langle u, e^l\rangle - \tilde{\varepsilon} \leq \sigma(s^i) \leq Mq_i + \langle p^i, Mc\rangle + \langle u, e^l\rangle + \tilde{\varepsilon} \qquad (11)$$

holds for all $i \geq N_{\tilde{\varepsilon}}$. Noting that $Mq_i + \langle p^i, Mc\rangle \in \mathbb{Z}$ for all i we obtain that $\lim_{i\to+\infty} F(\sigma(s^i)) = \langle u, e^l\rangle > 0$.

5 Proof of Main Theorem

To prove Theorem 2, we first verify that Step 2 in Figure 1 can be accomplished using a finite number of CG cuts.

Proposition 4. *If there exists a finite set $S \subset \mathbb{Z}^n$ such that*

$$CC(S,T) \subset T \qquad (12a)$$

$$CC(S,T) \cap bd(T) \subset \mathbb{Z}^n, \qquad (12b)$$

then $CC(\mathbb{Z}^n, T)$ is a rational polytope.

Proof. Let V be the set of vertices of $CC(S)$. By (12) we have that $bd(T) \cap V \subset \mathbb{Z}^n \cap T \subset CC(\mathbb{Z}^n)$. Hence any CG cut that separates $u \in CC(S) \setminus CC(\mathbb{Z}^n)$ must also separate a point in $V \setminus bd(T)$. It is then sufficient to show that the set of CG cuts that separates some point in $V \setminus bd(T)$ is finite. To achieve this we will use the fact that, because $V \setminus bd(T) \subset T \setminus bd(T)$ and $|V| < \infty$, there exists $1 > \varepsilon > 0$ such that

$$\gamma_B(v - c) \leq 1 - \varepsilon \quad \forall v \in V \setminus bd(T). \qquad (13)$$

Now, if a CG cut $\langle a, x\rangle \leq \lfloor\sigma(a)\rfloor$ for $a \in \mathbb{Z}^n$ separates $v \in V \setminus bd(T)$, then

$$\begin{aligned}
& \langle a, v\rangle > \lfloor\sigma(a)\rfloor & (14)\\
\Rightarrow\quad & \langle a, v\rangle > \sigma(a) - 1 & (15)\\
\Rightarrow\quad & \langle a, v\rangle > \sigma_B(a) + \langle a, c\rangle - 1 & (16)\\
\Rightarrow\quad & \sigma_B(a)\gamma_B(v-c) \geq \langle a, v-c\rangle > \sigma_B(a) - 1 & (17)\\
\Rightarrow\quad & \sigma_B(a) < \frac{1}{1 - \gamma_B(v-c)} \leq 1/\varepsilon & (18)\\
\Rightarrow\quad & a \in (1/\varepsilon)B^\circ. & (19)
\end{aligned}$$

The result follows from the fact that $(1/\varepsilon)B^\circ$ is a bounded set.

The separation results from Section 4 allows the construction of the set required in Proposition 4, which proves our main result.

Proof (Proof of Theorem 2). Let $\mathcal{I} := bd(T) \cap \mathbb{Z}^n$ be the finite (and possibly empty) set of integer points on the boundary of T. We divide the proof into the following cases

1. $CC(\mathbb{Z}^n) = \emptyset$.
2. $CC(\mathbb{Z}^n) \neq \emptyset$ and $CC(\mathbb{Z}^n) \cap int(T) = \emptyset$.
3. $CC(\mathbb{Z}^n) \cap int(T) \neq \emptyset$.

For the first case, the result follows directly. For the second case, by Proposition 1 and the strict convexity of T, we have that $|\mathcal{I}| = 1$ and $CC(\mathbb{Z}^n) = \mathcal{I}$ so the result again follows directly. For the third case we show the existence of a set S complying with conditions (12) presented in Proposition 4.

For each $u \in \mathcal{I}$, let $\varepsilon_u > 0$ be the value from Proposition 2. Let $\mathcal{D} := bd(T) \setminus \bigcup_{u\in\mathcal{I}} \{x \in \mathbb{R}^n : \|x - u\| < \varepsilon_u\}$. Observe that $\mathcal{D} \cap \mathbb{Z}^n = \emptyset$ by construction and that $\mathcal{D}$ is compact because it is obtained from compact set $bd(T)$ by removing a finite number of open sets. Now, for any $a \in \mathbb{Z}^n$ let $O(a) := \{x \in bd(T) \mid \langle a, x\rangle > \lfloor\sigma(a)\rfloor\}$ be the set of points of $bd(T)$ that are separated by the CG cut $\langle a, x\rangle \leq \lfloor\sigma(a)\rfloor$. This set is open with respect to $\mathcal{D}$. Furthermore, by Proposition 1 and the construction of $\mathcal{D}$, we have that $\mathcal{D} \subset \bigcup_{a\in\mathcal{A}} O(a)$ for a possibly infinite set $\mathcal{A} \subset \mathbb{Z}^n$. However, since $\mathcal{D}$ is a compact set we have that there exists a finite subset $\mathcal{A}_0 \subset \mathcal{A}$ such that

$$\mathcal{D} \subset \bigcup_{a\in\mathcal{A}_0} O(a). \tag{20}$$

Let $S := \mathcal{A}_0 \cup \bigcup_{u\in\mathcal{I}} W_u$ where, for each $u \in \mathcal{I}$, W_u is the set from Proposition 2. Then by (20) and Proposition 2 we have that S is a finite set that complies with condition (12b).

To show that S complies with condition (12a) we will show that if $p \notin T$, then $p \notin CC(S, T)$. To achieve this, we use the fact that $CC(\mathbb{Z}^n) \cap int(T) \neq \emptyset$. Let $\tilde{c} \in CC(\mathbb{Z}^n) \cap int(T)$, $\tilde{B} = B + c - \tilde{c}$ and $\gamma_{\tilde{B}}(x) = \inf\{\lambda > 0 : x \in \lambda\tilde{B}\}$ be the gauge of $\tilde{B}$. Then $\tilde{B}$ is a convex body with $0 \in int(\tilde{B})$, $T = \{x \in \mathbb{R}^n : \gamma_{\tilde{B}}(x - \tilde{c}) \leq 1\}$ and $bd(T) = \{x \in \mathbb{R}^n : \gamma_{\tilde{B}}(x - \tilde{c}) = 1\}$. Now, for $p \notin T$, let $\bar{p} := \tilde{c} + (p - \tilde{c})/\gamma_{\tilde{B}}(p - \tilde{c})$ so that $\bar{p} \in \{\mu\tilde{c} + (1 - \mu)p : \mu \in (0, 1)\}$ and $\bar{p} \in bd(T)$. If $\bar{p} \notin \mathbb{Z}^n$, then by the definitions of S and $\tilde{c}$ we have that there exists $a \in S$ such that $\langle a, \tilde{c}\rangle \leq \lfloor\sigma(a)\rfloor$ and $\langle a, \bar{p}\rangle > \lfloor\sigma(a)\rfloor$. Then $\langle a, p\rangle > \lfloor\sigma(a)\rfloor$ and hence $p \notin CC(S, T)$. If $\bar{p} \in \mathbb{Z}^n$ let $w \in W_{\bar{p}}$ be such that $\langle w, \tilde{c}\rangle < \lfloor\sigma(w)\rfloor$ and $\langle w, \bar{p}\rangle = \lfloor w\rfloor$. Then $\langle w, p\rangle > \lfloor\sigma(w)\rfloor$ and hence $p \notin CC(S, T)$.

6 Remarks

We note that the proof of Proposition 4 only uses the fact that T is a convex body and Theorem 2 uses the fact that T is additionally an ellipsoid only through Proposition 1 and Proposition 2. Therefore, we have the following general sufficient conditions for the polyhedrality of the first CG closure of a compact convex set.

Corollary 1. *Let T be any compact convex set. $CC(\mathbb{Z}^n, T)$ is a rational polyhedron if any of the following conditions hold*

Property 1 *There exists a finite $S \subset \mathbb{Z}^n$ such that* (12) *holds.*
Property 2 *For any $u \in bd(T) \setminus \mathbb{Z}^n$ there exists a CG cut that separates u and for any $u \in bd(T) \cap \mathbb{Z}^n$ there exist $\varepsilon_u > 0$ and a finite set $W_u \subset \mathbb{Z}^n$ such that* (2)–(4) *hold.*

A condition similar to (12) was considered in [41] for polytopes that are not necessarily rational. Specifically the author stated that if P is a polytope in real space such that $CC(\mathbb{Z}^n, P) \cap bd(P) = \emptyset$, then $CC(\mathbb{Z}^n, P)$ is a rational polytope. We imagine that the proof he had in mind could have been something along the lines of Proposition 4.

We also note that Step 2 of the procedure described in Section 2 can be directly turned into a finitely terminating algorithm by simple enumeration. However, it is not clear how to obtain a finitely terminating algorithmic version of Step 1 because it requires obtaining a finite subcover of the boundary of T from a quite complicated infinite cover.

Acknowledgements. We would like to thank Shabbir Ahmed, George Nemhauser and Arkadi Nemirovski for various discussions on this problem.

References

1. Abhishek, K., Leyffer, S., Linderoth, J.T.: FilMINT: An outer-approximation-based solver for nonlinear mixed integer programs. In: Preprint ANL/MCS-P1374-0906, Argonne National Laboratory, Mathematics and Computer Science Division, Argonne, IL (September 2006)
2. Andersen, K., Cornuéjols, G., Li, Y.: Split closure and intersection cuts. Mathematical Programming 102, 457–493 (2005)
3. Atamtürk, A., Narayanan, V.: Cuts for conic mixed-integer programming. In: Fischetti and Williamson [24], pp. 16–29
4. Atamtürk, A., Narayanan, V.: Lifting for conic mixed-integer programming. Research Report BCOL.07.04, IEOR, University of California-Berkeley, October 2007, Forthcoming in Mathematical Programming (2007)
5. Atamtürk, A., Narayanan, V.: The submodular 0-1 knapsack polytope. Discrete Optimization 6, 333–344 (2009)
6. Atamtürk, A., Narayanan, V.: Conic mixed-integer rounding cuts. Mathematical Programming 122, 1–20 (2010)
7. Balas, E., Saxena, A.: Optimizing over the split closure. Mathematical Programming 113, 219–240 (2008)
8. Belotti, P., Lee, J., Liberti, L., Margot, F., Waechter, A.: Branching and bound tightening techniques for non-convex MINLP. Optimization Methods and Software 24, 597–634 (2009)
9. 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)

10. Bonami, P., Kilinç, M., Linderoth, J.: Algorithms and software for convex mixed integer nonlinear programs, Technical Report 1664, Computer Sciences Department, University of Wisconsin-Madison (October 2009)
11. Caprara, A., Fischetti, M.: $\{0, \frac{1}{2}\}$-Chvátal-Gomory cuts. Mathematical Programming 74, 221–235 (1996)
12. Caprara, A., Letchford, A.N.: On the separation of split cuts and related inequalities. Mathematical Programming 94, 279–294 (2003)
13. Ceria, S., Soares, J.: Perspective cuts for a class of convex 0-1 mixed integer programs. Mathematical Programming 86, 595–614 (1999)
14. Çezik, M.T., Iyengar, G.: Cuts for mixed 0-1 conic programming. Mathematical Programming 104, 179–202 (2005)
15. Chvatal, V.: Edmonds polytopes and a hierarchy of combinatorial problems. Discrete Mathematics 4, 305–337 (1973)
16. Cook, W.J., Cunningham, W.H., Pulleyblank, W.R., Schrijver, A.: Combinatorial optimization. John Wiley and Sons, Inc., Chichester (1998)
17. Cook, W.J., Kannan, R., Schrijver, A.: Chvátal closures for mixed integer programming problems. Mathematical Programming 58, 155–174 (1990)
18. Dadush, D., Dey, S.S., Vielma, J.P.: The Chvátal-Gomory closure of strictly convex sets. Working paper, Geogia Institute of Technology (2010)
19. Dash, S., Günlük, O., Lodi, A.: On the MIR closure of polyhedra. In: Fischetti and Williamson [24], pp. 337–351
20. Dash, S., Günlük, O., Lodi, A.: MIR closures of polyhedral sets. Mathematical Programming 121, 33–60 (2010)
21. Eisenbrand, F.: On the membership problem for the elementary closure of a polyhedron. Combinatorica 19, 297–300 (1999)
22. Fischetti, M., Lodi, A.: Optimizing over the first Chvàtal closure. In: Jünger, M., Kaibel, V. (eds.) IPCO 2005. LNCS, vol. 3509, pp. 12–22. Springer, Heidelberg (2005)
23. Fischetti, M., Lodi, A.: Optimizing over the first Chvátal closure. Mathematical Programming, Series B 110, 3–20 (2007)
24. Fischetti, M., Williamson, D.P. (eds.): IPCO 2007. LNCS, vol. 4513. Springer, Heidelberg (2007)
25. Frangioni, A., Gentile, C.: Perspective cuts for a class of convex 0-1 mixed integer programs. Mathematical Programming 106, 225–236 (2006)
26. Gomory, R.E.: Outline of an algorithm for integer solutions to linear programs. Bulletin of the American Mathematical Society 64, 275–278 (1958)
27. Gomory, R.E.: An algorithm for integer solutions to linear programs. In: Recent advances in mathematical programming, pp. 269–302. McGraw-Hill, New York (1963)
28. Grossmann, I., Lee, S.: Generalized convex disjunctive programming: Nonlinear convex hull relaxation. Computational Optimization and Applications 26, 83–100 (2003)
29. 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)
30. Günlük, O., Linderoth, J.: Perspective relaxation of mixed integer nonlinear programs with indicator variables. In: Lodi, et al. (eds.) [38], pp. 1–16
31. Günlük, O., Linderoth, J.: Perspective relaxation of mixed integer nonlinear programs with indicator variables. Mathematical Programming, Series B (to appear 2009)

32. Hemmecke, R., Köppe, M., Lee, J., Weismantel, R.: Nonlinear integer programming. IBM Research Report RC24820, IBM, Yorktown Heights, NY (December 2008); Juenger, M., Liebling, T., Naddef, D., Nemhauser, G., Pulleyblank, W., Reinelt, G., Rinaldi, G., Wolsey, L.: 50 Years of Integer Programming 1958–2008: The Early Years and State-of-the-Art Surveys. Springer, Heidelberg (to appear 2010), ISBN 3540682740.
33. Jeroslow, R.: There cannot be any algorithm for integer programming with quadratic constraints. Operations Research 21, 221–224 (1973)
34. Letchford, A.N., Pokutta, S., Schulz, A.S.: On the membership problem for the $\{0, 1/2\}$-closure. Working paper, Lancaster University (2009)
35. Letchford, A.N., Sørensen, M.M.: Binary positive semidefinite matrices and associated integer polytopes. In: Lodi, et al. (eds.) [38], pp. 125–139
36. Leyffer, S., Linderoth, J.T., Luedtke, J., Miller, A., Munson, T.: Applications and algorithms for mixed integer nonlinear programming. Journal of Physics: Conference Series 180 (2009)
37. Leyffer, S., Sartenaer, A., Wanufelle, E.: Branch-and-refine for mixed-integer nonconvex global optimization. In: Preprint ANL/MCS-P1547-0908, Argonne National Laboratory, Mathematics and Computer Science Division, Argonne, IL (September 2008)
38. Lodi, A., Panconesi, A., Rinaldi, G. (eds.): IPCO 2008. LNCS, vol. 5035. Springer, Heidelberg (2008)
39. Richard, J.-P.P., Tawarmalani, M.: Lifting inequalities: a framework for generating strong cuts for nonlinear programs. Mathematical Programming 121, 61–104 (2010)
40. Saxena, A., Bonami, P., Lee, J.: Disjunctive cuts for non-convex mixed integer quadratically constrained programs. In: Lodi, et al. (eds.) [38], pp. 17–33
41. Schrijver, A.: On cutting planes. Annals of Discrete Mathematics 9, 291–296 (1980); Combinatorics 79 (Proc. Colloq., Univ. Montréal, Montreal, Que., 1979), Part II (1979)
42. Schrijver, A.: Theory of linear and integer programming. John Wiley & Sons, Inc., New York (1986)
43. Stubbs, R.A., Mehrotra, S.: A branch-and-cut method for 0-1 mixed convex programming. Mathematical Programming 86, 515–532 (1999)
44. Vielma, J.P.: A constructive characterization of the split closure of a mixed integer linear program. Operations Research Letters 35, 29–35 (2007)

A Pumping Algorithm for Ergodic Stochastic Mean Payoff Games with Perfect Information[⋆]

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

[1] RUTCOR, Rutgers University, 640 Bartholomew Road, Piscataway NJ 08854-8003
{boros,gurvich}@rutcor.rutgers.edu
[2] Max-Planck-Institut für Informatik, 66123 Saarbrücken, Germany
elbassio@mpi-sb.mpg.de
[3] Graduate School of Information Science and Technology, University of Tokyo, Tokyo, 113-8656, Japan
makino@mist.i.u-tokyo.ac.jp

Abstract. In this paper, we consider two-person zero-sum stochastic mean payoff games with perfect information, or BWR-games, given by a digraph $G = (V = V_B \cup V_W \cup V_R, E)$, with local rewards $r : E \to \mathbb{R}$, and three types of vertices: black V_B, white V_W, and random V_R. The game is played by two players, White and Black: When the play is at a white (black) vertex v, White (Black) selects an outgoing arc (v, u). When the play is at a random vertex v, a vertex u is picked with the given probability $p(v, u)$. In all cases, Black pays White the value $r(v, u)$. The play continues forever, and White aims to maximize (Black aims to minimize) the limiting mean (that is, average) payoff. It was recently shown in [7] that BWR-games are polynomially equivalent with the classical Gillette games, which include many well-known subclasses, such as cyclic games, simple stochastic games (SSG′s), stochastic parity games, and Markov decision processes. In this paper, we give a new algorithm for solving BWR-games in the *ergodic case*, that is when the optimal values do not depend on the initial position. Our algorithm solves a BWR-game by reducing it, using a potential transformation, to a canonical form in which the optimal strategies of both players and the value for every initial position are obvious, since a locally optimal move in it is optimal in the whole game. We show that this algorithm is pseudo-polynomial when the number of random nodes is constant. We also provide an almost matching lower bound on its running time, and show that this bound holds for a wider class of algorithms. Let us add that the general (non-ergodic) case is at least as hard as SSG′s, for which no pseudo-polynomial algorithm is known.

Keywords: mean payoff games, local reward, Gillette model, perfect information, potential, stochastic games.

[⋆] This research was partially supported by DIMACS, Center for Discrete Mathematics and Theoretical Computer Science, Rutgers University, and by the Scientific Grant-in-Aid from Ministry of Education, Science, Sports and Culture of Japan.

1 Introduction

1.1 BWR-Games

We consider two-person zero-sum stochastic games with perfect information and mean payoff: Let $G = (V, E)$ be a digraph whose vertex-set V is partitioned into three subsets $V = V_B \cup V_W \cup V_R$ that correspond to black, white, and random positions, controlled respectively, by two players, *Black* - the *minimizer* and *White* - the *maximizer*, and by nature. We also fix a *local reward* function $r : E \to \mathbb{R}$, and probabilities $p(v, u)$ for all arcs (v, u) going out of $v \in V_R$. Vertices $v \in V$ and arcs $e \in E$ are called *positions* and *moves*, respectively. In a personal position $v \in V_W$ or $v \in V_B$ the corresponding player White or Black selects an arc (v, u), while in a random position $v \in V_R$ a move (v, u) is chosen with the given probability $p(v, u)$. In all cases Black pays White the reward $r(v, u)$.

From a given initial position $v_0 \in V$ the game produces an infinite walk (called a *play*). White's objective is to maximize the *limiting mean payoff*

$$c = \liminf_{n\to\infty} \frac{\sum_{i=0}^{n} b_i}{n+1}, \tag{1}$$

where b_i is the reward incurred at step i of the play, while the objective of Black is the opposite, that is, to minimize $\limsup_{n\to\infty} \frac{\sum_{i=0}^{n} b_i}{n+1}$.

For this model it was shown in [7] that a *saddle point* exists in *pure positional uniformly optimal* strategies. Here "pure" means that the choice of a move (v, u) in a personal position $v \in V_B \cup V_R$ is deterministic, not random; "positional" means that this choice depends solely on v, not on previous positions or moves; finally, "uniformly optimal" means that it does not depend on the initial position v_0, either. The results and methods in [7] are similar to those of Gillette [17]; see also Liggett and Lippman [28]: First, we analyze the so-called *discounted* version, in which the payoff is discounted by a factor β^i at step i, giving the effective payoff: $a_\beta = (1 - \beta) \sum_{i=0}^{\infty} \beta^i b_i$, and then we proceed to the limit as the *discount factor* $\beta \in [0, 1)$ tends to 1.

This class of the *BWR-games* was introduced in [19]; see also [10]. It was recently shown in [7] that the BWR-games and classical Gillette games [17] are polynomially equivalent. The special case when there are no random positions, $V_R = \emptyset$, is known as *cyclic*, or *mean payoff*, or *BW-games*. They were introduced for the complete bipartite digraphs in [32,31], for all (not necessarily complete) bipartite digraphs in [15], and for arbitrary digraphs in [19]. A more special case was considered extensively in the literature under the name of *parity games* [2,3,11,20,22,24], and later generalized also to include random nodes in [10]. A BWR-game is reduced to a *minimum mean cycle problem* in case $V_W = V_R = \emptyset$, see, for example [25]. If one of the sets V_B or V_W is empty, we obtain a *Markov decision process*; see, for example, [30]. Finally, if both are empty $V_B = V_W = \emptyset$, we get a *weighted Markov chain*.

It was noted in [9] that "parlor games", like Backgammon (and even Chess) can be solved in pure positional uniformly optimal strategies, based on their BWR-model.

In the special case of a BWR-game, when all rewards are zero except at a single node t called the terminal, at which there is a self-loop with reward 1, we obtain the so-called *simple stochastic games* (SSG), introduced by Condon [12,13] and considered in several papers [18,20]. In these games, the objective of White is to maximize the probability of reaching the terminal, while Black wants to minimize this probability. Recently, it was shown that Gillette games (and hence BWR-games by [7]) are equivalent to SSG's under polynomial-time reductions [1]. Thus, by recent results of Björklund, Vorobyov [5], and Halman [20], all these games can be solved in randomized strongly subexponential time $2^{O(\sqrt{n_d \log n_d})}$, where $n_d = |V_B| + |V_W|$ is the number of *deterministic* vertices. Let us note that several pseudo-polynomial and subexponential algorithms exists for BW-games [19,26,33,6,4,20,36]; see also [14] for a so called policy iteration method, and [24] for parity games.

Besides their many applications (see e.g. [29,23]), all these games are of interest to Complexity Theory: Karzanov and Lebedev [26] (see also [36]) proved that the decision problem "whether the value of a BW-game is positive" is in the intersection of NP and co-NP. Yet, no polynomial algorithm is known for these games, see e.g., the recent survey by Vorobyov [35]. A similar complexity claim can be shown to hold for SSG′s and BWR-games, see [1,7].

While there are numerous pseudo-polynomial algorithms known for the BW-case, it is a challenging open question whether a pseudo-polynomial algorithm exists for SSG′s or BWR-games.

1.2 Potential Transformations and Canonical Forms

Given a BWR-game, we consider *potential transformations* $x : V \to \mathbb{R}$, assigning a real-value $x(v)$ to each vertex $v \in V$, and transforming the local reward on each arc (v, u) to $r_x(v, u) = r(v, u) + x(v) - x(u)$. It is known that for BW-games there exists a potential transformation such that, in the obtained game the locally optimal strategies are globally optimal, and hence, the value and optimal strategies become obvious [19]. This result was extended for the more general class of BWR-games in [7]: in the transformed game, the equilibrium value $\mu(v) = \mu_x(v)$ is given simply by the maximum local reward for $v \in V_W$, the minimum local reward for $v \in V_B$, and the average local reward for $v \in V_R$. In this case we say that the transformed game is in *canonical* form.

It is not clear how the algorithm given in [19] for the BW-case can be generalized to BWR-case. The proof in [7] follows by considering the discounted case and then taking the discount factor β to the limit. While such an approach is sufficient to prove the existence of a canonical form, it does not provide an algorithm to compute the potentials, since the corresponding limits appear to be infinite. In this paper, we give such an algorithm that does *not* go through the discounted case. Our method computes an optimal potential transformation in case the game is *ergodic*, that is, when the optimal values do not depend on the initial position. If the game is not ergodic then our algorithm terminates with a proof of non-ergodicity, by exhibiting at least two vertices with provably distinct values. Unfortunately, our approach cannot be applied recursively in this case.

This is not a complete surprise, since this case is at least as hard as SSG′s, for which no pseudo-polynomial algorithm is known.

Theorem 1. *Consider a BWR-game with k random nodes, a total of n vertices, and integer rewards in the range $[-R, R]$, and assume that all probabilities are rational whose common denominator is bounded by W. Then there is an algorithm that runs in time $n^{O(k)}W^{O(k^2)}R\log(nRW)$ and either brings the game by a potential transformation to canonical form, or proves that it is non-ergodic.*

Let us remark that the ergodic case is frequent in applications. For instance, it is the case when $G = (V_W \cup V_B \cup V_R, E)$ is a complete tripartite digraph (where $p(v, u) > 0$ for all $v \in V_R$ and $(v, u) \in E$); see Section 3 for more general sufficient conditions.

Theorem 1 states that our algorithm is pseudo-polynomial if the number of random nodes is fixed. As far as we know, this is the first algorithm with such a guarantee (in comparison, for example, to strategy improvement methods [4,21,34], for which exponential lower bounds are known [16]; it is worth mentioning that the algorithm of [21] also works only for the ergodic case). In fact, we are not aware of any previous results bounding the running time of an algorithm for a class of BWR-games in terms of the number of random nodes, except for [18] which shows that simple stochastic games on k random nodes can be solved in time $O(k!(|V||E| + L))$, where L is the maximum bit length of a transition probability. It is worth remarking here that even though BWR-games are polynomially reducible to simple stochastic games, under this reduction the number of random nodes k becomes a polynomial in n, even if the original BWR-game has constantly many random nodes. In particular, the result in [18] does not imply a bound similar to that of Theorem 1 for general BWR-games.

One should also contrast the bound in Theorem 1 with the subexponential bounds in [20]: roughly, the algorithm of Theorem 1 will be more efficient if $|V_R|$ is $o((|V_W| + |V_B|)^{\frac{1}{4}})$ (assuming that W and R are polynomials in n). However, our algorithm could be practically much faster since it can stop much earlier than its estimated worst-case running time (unlike the subexponential algorithms [20], or those based on dynamic programming [36]). In fact, as our preliminary experiments indicate, to approximate the value of a random game on up to $15,000$ nodes within an additive error $\varepsilon = 0.001$, the algorithm takes no more than a few hundred iterations, even if the maximum reward is very large. One more desirable property of this algorithm is that it is of the *certifying* type (see e.g. [27]), in the sense that, given an optimal pair of strategies, the vector of potentials provided by the algorithm can be used to verify optimality in *linear* time (otherwise verifying optimality requires solving two linear programs).

1.3 Overview of the Techniques

Our algorithm for proving Theorem 1 is quite simple. Starting form zero potentials, and depending on the current locally optimal rewards (maximum for White, minimum for Black, and average for Random), the algorithm keeps selecting a subset of nodes and reducing their potentials by some value, until either the locally optimal rewards at different nodes become sufficiently close to each other, or a proof of non-ergodicity is obtained in the form of a certain partition of the nodes. The upper bound on the running time consists of three technical parts. The first one is to show that if the number of iterations becomes too large, then there is a large enough potential gap to ensure an ergodic partition. In the second part, we show that the range of potentials can be kept sufficiently small throughout the algorithm, namely $\|x^*\|_\infty \leq nRk(2W)^k$, and hence the range of the transformed rewards does not explode. The third part concerns the required accuracy. It can be shown that it is enough in our algorithm to get the value of the game within an accuracy of

$$\varepsilon = \frac{1}{n^{2(k+1)}k^{2k}(2W)^{4k+2k^2+2}}, \tag{2}$$

in order to guarantee that it is equal to the exact value. As far as we know, such a bound in terms of k is new, and it could be of independent interest. We also show the lower bound $W^{\Omega(k)}$ on the running time of the algorithm of Theorem 1 by providing an instance of the problem, with only random nodes.

The paper is organized as follows. In the next section, we formally define BWR-games, canonical forms, and state some useful propositions. In Section 3, we give a sufficient condition for the ergodicity of a BWR-game, which will be used as one possible stopping criterion in our algorithm. We give the algorithm in Section 4.1, and prove it converges in Section 4.2. In Section 5, we show that this convergence proof can, in fact, be turned into a quantitative statement giving the precise bounds stated in Theorem 1. The last section gives a lower bound example for the algorithm. Due to lack of space, most of the proofs are omitted (see [8] for details).

2 Preliminaries

2.1 BWR-Games

A BWR-game is defined by the quadruple $\mathcal{G} = (G, P, v_0, r)$, where $G = (V = V_W \cup V_B \cup V_R, E)$ is a digraph that may have loops and multiple arcs, but no terminal vertices[1], i.e., vertices of out-degree 0; P is the set of probability distributions for all $v \in V_R$ specifying the probability $p(v,u)$ of a move form v to u; $v_0 \in V$ is an initial position from which the play starts; and $r : E \to \mathbb{R}$ is a local reward function. We assume that $\sum_{u \mid (v,u)\in E} p(v,u) = 1 \quad \forall v \in V_R$. For convenience we will assume that $p(v,u) > 0$ whenever $(v,u) \in E$ and $v \in V_R$, and set $p(v,u) = 0$ for $(v,u) \notin E$.

Standardly, we define a strategy $s_W \in S_W$ (respectively, $s_B \in S_B$) as a mapping that assigns a move $(v,u) \in E$ to each position $v \in V_W$ (respectively, $v \in V_B$). A pair of strategies $s = (s_W, s_B)$ is called a *situation*. Given a

[1] This assumption is without loss of generality since otherwise one can add a loop to each terminal vertex.

BWR-game $\mathcal{G} = (G, P, v_0, r)$ and situation $s = (s_B, s_W)$, we obtain a (weighted) Markov chain $\mathcal{G}_s = (G, P_s, v_0, r)$ with transition matrix P_s in the obvious way:

$$p_s(v,u) = \begin{cases} 1 & \text{if } (v \in V_W \text{ and } u = s_W(v)) \text{ or } (v \in V_B \text{ and } u = s_B(v)); \\ 0 & \text{if } (v \in V_W \text{ and } u \neq s_W(v)) \text{ or } (v \in V_B \text{ and } u \neq s_B(v)); \\ p(v,u) & \text{if } v \in V_R. \end{cases}$$

In the obtained Markov chain $\mathcal{G}_s = (G, P_s, v_0, r)$, we define the limiting (mean) effective payoff $c_s(v_0)$ as

$$c_s(v_0) = \sum_{v \in V} p^*(v) \sum_u p_s(v,u) r(v,u), \tag{3}$$

where $p^* : V \to [0,1]$ is the limiting distribution for $\mathcal{G}_s$ starting from v_0. Doing this for all possible strategies of Black and White, we obtain a matrix game $C_{v_0} : S_W \times S_B \to \mathbb{R}$, with entries $C_{v_0}(s_W, s_B)$ defined by (3).

2.2 Solvability and Ergodicity

It is known that every such game has a saddle point in pure strategies [17,28].

Moreover, there are optimal strategies (s_W^*, s_B^*) that do not depend on the starting position v_0, so-called uniformly optimal strategies. In contrast, the value of the game $\mu(v_0) = C_{v_0}(s_W^*, s_B^*)$ may depend on v_0.

The triplet $\mathcal{G} = (G, P, r)$ is called a *un-initialized BWR-game*. Furthermore, $\mathcal{G}$ is called *ergodic* if the value $\mu(v_0)$ of each corresponding BWR-game (G, P, v_0, r) is the same for all initial positions $v_0 \in V$.

2.3 Potential Transforms

Given a BWR-game $\mathcal{G} = (G, P, v_0, r)$, let us introduce a mapping $x : V \to \mathbb{R}$, whose values $x(v)$ will be called *potentials*, and define the transformed reward function $r_x : E \to \mathbb{R}$ as:

$$r_x(v,u) = r(v,u) + x(v) - x(u), \quad \text{where } (v,u) \in E. \tag{4}$$

It is not difficult to verify that the two normal form matrices C_x and C, of the obtained game $\mathcal{G}_x$ and the original game $\mathcal{G}$, are equal (see [7]). In particular, their optimal (pure positional) strategies coincide, and the values also coincide: $\mu_x(v_0) = \mu(v_0)$. Given a BWR-game $\mathcal{G} = (G, P, r)$, let us define a mapping $m : V \to \mathbb{R}$ as follows:

$$m(v) = \begin{cases} \max(r(v,u) \mid u : (v,u) \in E) & \text{for } v \in V_W, \\ \min(r(v,u) \mid u : (v,u) \in E) & \text{for } v \in V_W, \\ \text{mean}(r(v,u) \mid u : (v,u) \in E) = \sum_{u \mid (v,u) \in E} r(v,u)\, p(v,u) & \text{for } v \in V_R. \end{cases} \tag{5}$$

A move $(v,u) \in E$ in a position $v \in V_W$ (respectively, $v \in V_B$) is called *locally optimal* if it realizes the maximum (respectively, minimum) in (5). A strategy s_W of White (respectively, s_B of Black) is called *locally optimal* if it chooses a locally optimal move $(v,u) \in E$ in every position $v \in V_W$ (respectively, $v \in V_B$).

Proposition 1. *If in a BWR-game the function $m(v) = M$ for all $v \in V$, then (i) every locally optimal strategy is optimal and (ii) the game is ergodic: M is its value for every initial position $v_0 \in V$.*

3 Sufficient Conditions for Ergodicity of BWR-Games

A digraph $G = (V = V_W \cup V_B \cup V_R, E)$ is called *ergodic* if any un-initialized BWR-game $\mathcal{G} = (G, P, r)$ on G is ergodic, that is, the values of the games $\mathcal{G} = (G, P, v_0, r)$ do not depend on v_0. We will give a simple characterization of ergodic digraphs, which, obviously, provides a sufficient condition for ergodicity of the BWR-games.

In addition to partition $\Pi_p : V = V_W \cup V_B \cup V_R$, let us consider one more partition $\Pi_r : V = V^W \cup V^B \cup V^R$ with the following properties:

(i) Sets V^W and V^B are not empty (while V^R might be empty).
(ii) There is no arc $(v, u) \in E$ such that $(v \in (V_W \cup V_R) \cap V^B$ and $u \notin V^B)$ or, vice versa, $(v \in (V_B \cup V_R) \cap V^W$ and $u \notin V^W)$. In other words, White cannot leave V^B, Black cannot leave V^W, and there are no random moves from $V^W \cup V^B$.
(iii) For each $v \in V_W \cap V^W$ (respectively, $v \in V_B \cap V^B$) there is a move $(v, u) \in E$ such that $u \in V^W$ (respectively, $u \in V^B$). In other words, White (Black) cannot be forced to leave V^W (respectively, V^B).

In particular, the properties above imply that the induced subgraphs $G[V^W]$ and $G[V^B]$ have no terminal vertex.

Partition $\Pi_r : V = V^W \cup V^B \cup V^R$ satisfying (i), (ii), and (iii) will be called a *contra-ergodic* partition for digraph $G = (V_W \cup V_B \cup V_R, E)$.

Theorem 2. *A digraph G is ergodic iff it has no contra-ergodic partition.*

The "only if part" can be strengthened as follows:

Proposition 2. *Given a BWR-game $\mathcal{G}$ whose graph has a contra-ergodic partition, if $m(v) > m(u)$ for every $v \in V^W, u \in V^B$ then $\mu(v) > \mu(u)$ for every $v \in V^W, u \in V^B$.*

Definition 1. *A* contra-ergodic decomposition *of $\mathcal{G}$ is a contra-ergodic partition $\Pi_r : V = V^W \cup V^B \cup V^R$ such that $m(v) > m(u)$ for every $v \in V^W$ and $u \in V^B$.*

By Proposition 2, $\mathcal{G}$ is not ergodic whenever it has such a decomposition.

4 Pumping Algorithm for the Ergodic BWR-Games

4.1 Description of the Algorithm

Given a BWR-game $\mathcal{G} = (G, P, r)$, let us compute $m(v)$ for all $v \in V$ using (5). Throughout, we will denote by $[m] \overset{\text{def}}{=} [m^-, m^+]$ and $[r] \overset{\text{def}}{=} [r^-, r^+]$ the range of functions m and r, respectively, and let $M = m^+ - m^-$ and $R = r^+ - r^-$. Given potentials $x : V \to \mathbb{R}$, we denote by m_x the function m in (5) in which r is replaced by the transformed reward r_x. Given a subset $I \subseteq [m]$, let $V(I) = \{v \in V \mid m(v) \in I\} \subseteq V$. In the following algorithm, set I will always be a closed or semi-closed interval within $[m]$.

Let $m^- = t_0 < t_1 < t_2 < t_3 < t_4 = m^+$ be given thresholds. We will successively apply potential transforms $x : V \to \mathbb{R}$ such that no vertex ever leaves the interval $[t_0, t_4]$ or $[t_1, t_3]$; in other words, $V_x[t_0, t_4] = V[t_0, t_4]$ and $V_x[t_1, t_3] \supseteq V[t_1, t_3]$ for all considered transforms x, where $V_x(I) = \{v \in V \mid m_x(v) \in I\}$. Let us initialize potentials $x(v) = 0$ for all $v \in V$. We will fix

$$t_0 := m_x^-,\ t_1 := m_x^- + \frac{1}{4}M_x,\ t_2 := m_x^- + \frac{1}{2}M_x,\ t_3 := m_x^- + \frac{3}{4}M_x,\ t_4 := m_x^+, \quad (6)$$

where $M_x = m_x^+ - m_x^-$. Then, let us reduce all potentials of $V_x[t_2, t_4]$ by a maximum constant δ such that no vertex leaves the closed interval $[t_1, t_3]$; in other words, we stop the iteration whenever a vertex from this interval reaches its border. After this we compute potentials $x(v)$, new values $m_x(v)$, for $v \in V$, and start the next iteration.

It is clear that δ can be computed in linear time: it is the maximum value δ such that $m_x^\delta(v) \geq t_1$ for all $v \in V_x[t_2, t_4]$ and $m_x^\delta(v) \leq t_3$ for all $v \in V_x[t_0, t_2)$, where $m_x^\delta(v)$ is the new value of $m_x(v)$ after all potentials in $V_x[t_2, t_4]$ have been reduce by δ.

It is also clear from our update method (and important) that $\delta \geq M_x/4$. Indeed, vertices from $[t_2, t_4]$ can only go down, while vertices from $[t_0, t_2)$ can only go up. Each of them must traverse a distance of at least $M_x/4$ before it can reach the border of the interval $[t_1, t_3]$. Moreover, if after some iteration one of the sets $V_x[t_0, t_1)$ or $V_x(t_3, t_4]$ becomes empty then the range of m_x is reduced at least by 25%.

Procedure PUMP$(\mathcal{G}, \varepsilon)$ below tries to reduce any BWR-game $\mathcal{G}$ by a potential transformation x into one in which $M_x \leq \varepsilon$. Two subroutines are used in the procedure. REDUCE-POTENTIALS$(\mathcal{G}, x)$ replaces the current potential x with another potential with a sufficiently small norm (cf. Lemma 3 below). This reduction is needed since without it the potentials and, hence, the transformed local rewards too, can grow exponentially. The second routine FIND-PARTITION$(\mathcal{G}, x)$ uses the current potential vector x to construct a contra-ergodic decomposition of $\mathcal{G}$ (cf. line 19 of the algorithm below). We will prove in Lemma 2 that if the number of pumping iterations performed is large enough:

$$N = \frac{8n^2 R_x}{M_x \theta^{k-1}} + 1, \quad (7)$$

where $R_x = r_x^+ - r_x^-$, $\theta = \min\{p(v, u)\ :\ (v, u) \in E\}$, and k is the number of random nodes, and yet the range of m_x is not reduced, then we will be able to find a contra-ergodic decomposition.

In Section 4.2, we will first argue that the algorithm terminates in finite time if $\varepsilon = 0$ and the considered BWR-game is ergodic. In the following section, this will be turned into a quantitative argument with the precise bound on the running time. Yet, in Section 6, we will show that this time can be exponential already for R-games.

4.2 Proof of Finiteness for the Ergodic Case

Let us assume without loss of generality that range of m is $[0, 1]$, and the initial potential $x^0 = 0$. Suppose that during N iterations no new vertex enters the interval $[1/4, 3/4]$. Then, $-x(v) \geq N/4$ for each $v \in V(3/4, 1]$, since these vertices "were pumped" N times, and $x(v) \equiv 0$ for each $v \in V[0, 1/4)$, since these vertices "were not pumped" at all. We will show that if N is sufficiently large then the considered game is not ergodic.

Consider infinitely many iterations $i = 0, 1, \ldots$, and denote by $V^B \subseteq V$ (respectively, by $V^W \subseteq V$) the set of vertices that were pumped just finitely many times (respectively, always but finitely many times); in other words, $m_{x^0}(v) \in [1/2, 1]$ if $v \in V^W$ (respectively, $m_{x^0}(v) \in [0, 1/2)$ if $v \in V^B$) for all but finitely many i's. It is not difficult to verify that the partition $\Pi_r : V = V^W \cup V^B \cup V^R$, where $V^R = V \setminus (V^W \cup V^B)$, is contra-ergodic. It is also clear that after sufficiently many iterations $m_{x^0}(v) > 1/2$ for all $v \in V^W$, while $m_{x^0}(v) \leq 1/2$ for all $v \in V^B$. Thus, by Proposition 2, the considered game $\mathcal{G}$ is not ergodic, or in other words, our algorithm is finite for the ergodic BWR-games. We shall give an upper bound below for the number of times a vertex can oscillate around $1/2$ before it finally settles itself down in $[1/4, 1/2)$ or in $(1/2, 3/4]$.

5 Running Time Analysis

Consider the execution of the algorithm on a given BWR-game. We define a *phase* to be a set of iterations during which the range of m_x, defined with respect to the current potential x, is not reduced by a constant factor of what it was at the beginning of the phase, i.e., none of the sets $V_x[t_0, t_1)$ or $V_x(t_3, t_4]$ becomes empty (cf. line 12 of the procedure). Note that the number of iterations in each phase is at most N defined by (7). Lemma 2 states that if N iterations are performed in a phase then, the game is not ergodic. Lemma 4 bounds the total number of phases and estimates the overall running time.

5.1 Finding a Contra-Ergodic Decomposition

We assume throughout this section that we are inside phase h of the algorithm, which started with a potential x^h, and that $M_x > \frac{3}{4} M_{x^h}$ in all N iterations of the phase, and hence we proceed to step 19. For convenience, we will write $(\cdot)_{x^h}$ as $(\cdot)_h$, where $(\cdot)$ could be m, r, r^+, etc, (e.g., $m_h^- = m_{x^h}^-$, $m_h^+ = m_{x^h}^+$). For simplicity, we assume that the phase starts with local reward function $r = r_h$ and hence [2] $x^h = 0$. Given a potential vector x, we use the following notation:

$$\mathrm{EXT}_x = \{(v, u) \in E \ : \ v \in V_B \cup V_W \text{ and } r_x(v, u) = m_x(v)\}, \quad \Delta_x = \min\{x(v) : v \in V\}.$$

[2] In particular, note that $r_x(v, u)$ and $m_x(v)$ are used, for simplicity of notation, to actually mean $r_{x+x^h}(v, u)$ and $m_{x+x^h}(v)$, respectively.

Algorithm 1. PUMP($\mathcal{G}, \varepsilon$)

```
Input: A BWR-game $\mathcal{G} = (G = (V, E), P, r)$ and a desired accuracy $\varepsilon$
Output: a potential $x : V \to \mathbb{R}$ s.t. $|m_x(v) - m_x(u)| \le \varepsilon$ for all $u, v \in V$ if the game
    is ergodic, and a contra-ergodic decomposition otherwise
 1: let $x^0(v) := x(v) := 0$ for all $v \in V$; $i := 1$
 2: let $t_0, t_1, \ldots, t_4$, and $N$ be as defined by (6) and (7)
 3: while $i \le N$ do
 4:    if $M_x \le \varepsilon$ then
 5:       return $x$
 6:    end if
 7:    $\delta := \max\{\delta' \mid m_x^{\delta'}(v) \ge t_1$ for all $v \in V_{x^0}[t_2, t_4]$ and $m_x^{\delta'}(v) \le t_3$ for all $v \in$
       $V_{x^0}[t_0, t_2)\}$
 8:    if $\delta = \infty$ then
 9:       return the ergodic partition $V_{x^0}[t_0, t_2) \cup V_{x^0}[t_2, t_4]$
10:    end if
11:    $x(v) := x(v) - \delta$ for all $v \in V_{x^0}[t_2, t_4]$
12:    if $V_{x^0}[t_0, t_1) = \emptyset$ or $V_{x^0}(t_3, t_4] = \emptyset$ then
13:       $x := x^0$:=REDUCE-POTENTIALS($\mathcal{G}, x$); $i := 1$
14:       recompute the thresholds $t_0, t_1, \ldots, t_4$ and $N$ using (6) and (7)
15:    else
16:       $i := i + 1$;
17:    end if
18: end while
19: $V^W \cup V^B \cup V^R$:=FIND-PARTITION($\mathcal{G}, x$)
20: return contra-ergodic partition $V^W \cup V^B \cup V^R$
```

Let $t_l < 0$ be the largest value satisfying the following conditions:

(i) there are no arcs $(v, u) \in E$ with $v \in V_W \cup V_R$, $x(v) \ge t_l$ and $x(u) < t_l$;
(i) there are no arcs $(v, u) \in \text{EXT}_x$ with $v \in V_B$, $x(v) \ge t_l$ and $x(u) < t_l$.

Let $X = \{v \in V \ : \ x(v) \ge t_l\}$. In words, X is the set of nodes with potential as close to 0 as possible, such that no white or random node in X has an arc crossing to $V \setminus X$, and no black node has an extremal arc crossing to $V \setminus X$. Similarly, define $t_u > \Delta_x$ to be the smallest value satisfying the following conditions:

(i) there are no arcs $(v, u) \in E$ with $v \in V_B \cup V_R$, $x(v) \le t_u$ and $x(u) > t_u$;
(i) there are no arcs $(v, u) \in \text{EXT}_x$ with $v \in V_W$, $x(v) \le t_u$ and $x(u) > t_u$,

and let $Y = \{v \in V \ : \ x(v) \le t_u\}$. Note that the sets X and Y can be computed in $O(|V| \log |V| + |E|)$ time.

Lemma 1. *It holds that* $\max\{-t_l, t_u - \Delta_x\} \le nR_h \left(\frac{1}{\theta}\right)^{k-1}$.

The correctness of the algorithm follows from the following lemma.

Lemma 2. *Suppose that pumping is performed for $N_h \ge 2nT_h + 1$ iterations, where $T_h = \frac{4nR_h}{M_h \theta^{k-1}}$, and neither the set $V_h[t_0, t_1)$ nor $V_h(t_3, t_4]$ becomes empty. Let $V^B = X$ and $V^W = Y$ be the sets constructed as above, and $V^R = V \setminus (X \cup Y)$. Then $V^W \cup V^B \cup V^R$ is a contra-ergodic decomposition.*

5.2 Potential Reduction

One problem that arises during the pumping procedure is that the potentials can increase exponentially in the number of phases, making our bounds on the number of iterations per phase also exponential in n. For the BW-case Pisaruk [33] solved this problem by giving a procedure that reduces the range of the potentials after each round, while keeping all its desired properties needed for the running time analysis. Pisaruk's potential reduction procedure can be thought of as a combinatorial procedure for finding an *extreme point* of a polyhedron, given a point in it. Indeed, given a BWR-game and a potential x, let us assume without loss of generality, by shifting the potentials if necessary, that $x \geq 0$, and let $E' = \{(v,u) \in E \ : \ r_x(v,u) \in [m_x^-, m_x^+], \ v \in V_B \cup V_W\}$, where r is the *original* local reward function. Then the following polyhedron is non-empty:

$$\Gamma_x = \left\{ x' \in \mathbb{R}^V \;\middle|\; \begin{array}{ll} m_x^- \leq r(v,u) + x'(v) - x'(u) \leq m_x^+, \forall (v,u) \in E' & \\ r(v,u) + x'(v) - x'(u) \leq m_x^+, & \forall v \in V_W, \ (v,u) \in E \setminus E' \\ m_x^- \leq r(v,u) + x'(v) - x'(u), & \forall v \in V_B, \ (v,u) \in E \setminus E' \\ m_x^- \leq \sum_{u \in V} p(v,u)(r(v,u) + x'(v) - x'(u)) \leq m_x^+, & \forall v \in V_R \\ x(v) \geq 0 & \forall v \in V \end{array} \right\}.$$

Moreover, Γ_x is pointed, and hence, it must have an extreme point.

Lemma 3. *Consider a BWR-game in which all rewards are integral with range $R = r^+ - r^-$, and probabilities $p(v,u)$ are rational with common denominator at most W, and let $k = |V_R|$. Then any extreme point x^* of Γ_x satisfies $\|x^*\|_\infty \leq nRk(2W)^k$.*

Note that any point $x' \in \Gamma_x$ satisfies $[m_{x'}] \subseteq [m_x]$, and hence, replacing x by x^* does not increase the range of m_x.

5.3 Proof of Theorem 1

Consider a BWR-game $\mathcal{G} = (G = (V,E), P, r)$ with $|V| = n$ vertices and k random nodes. Assume r to be integral in the range $[-R, R]$ and all transition probabilities are rational with common denominator W. From Lemmas 2 and 3, we can conclude the following bound.

Lemma 4. *Procedure PUMP$(\mathcal{G}, \varepsilon)$ terminates in $O(nk(2W)^k(\frac{1}{\varepsilon} + n^2|E|)R\log(\frac{R}{\varepsilon}))$ time.*

Theorem 1 follows by setting ε sufficiently small:

Corollary 1. *When procedure PUMP$(\mathcal{G}, \varepsilon)$ is run with ε as in (2), it either outputs a potential vector x such $m_x(v)$ is constant for all $v \in V$, or finds a contra-ergodic partition. The total running time is $n^{O(k)}W^{O(k^2)}R\log(nRW)$.*

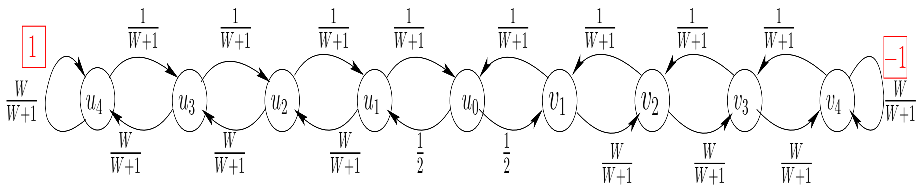


Fig. 1. An exponential Example

6 Lower Bound Example

We show now that the execution time of the algorithm, in the worst case, can be exponential in the number of random nodes k, already for weighted Markov chains, that is, for R-games. Consider the following example. Let $G = (V, E)$ be a digraph on $k = 2l + 1$ vertices $u_l, \dots, u_1, u_0 = v_0, v_1, \dots, v_l$, and with the following set of arcs:

$$E = \{(u_l, u_l), (v_l, v_l)\} \cup \{(u_{i-1}, u_i), (u_i, u_{i-1}), (v_{i-1}, v_i), (v_i, v_{i-1}) : \ i = 1, \dots, l\}.$$

Let $W \geq 1$ be an integer. All nodes are random with the following transition probabilities: $p(u_l, u_l) = p(v_l, v_l) = 1 - \frac{1}{W+1}$, $p(u_0, u_1) = p(u_0, v_1) = \frac{1}{2}$, $p(u_{i-1}, u_i) = p(v_{i-1}, v_i) = 1 - \frac{1}{W+1}$, for $i = 2, \dots, l$, and $p(u_i, u_{i-1}) = p(v_i, v_{i-1}) = \frac{1}{W+1}$, for $i = 1, \dots, l$. The local rewards are zero on every arc, except for $r(u_l, u_l) = -r(v_l, v_l) = 1$. Clearly this Markov chain consists of a single recurrent class, and it is easy to verify that the limiting distribution p^* is as follows:

$$p^*(u_0) = \frac{W-1}{(W+1)W^l - 2}, \ p^*(u_i) = p^*(v_i) = \frac{W^{i-1}(W^2-1)}{2((W+1)W^l - 2)} \text{ for } i = 1, \dots, l.$$

The optimal expected reward at each vertex is

$$\mu(u_i) = \mu(v_i) = -1 \cdot (1 - \frac{1}{W+1})p^*(u_l) + 1 \cdot (1 - \frac{1}{W+1})p^*(u_l) = 0,$$

for $i = 0, \dots, l$. Up to a shift, there is a unique set of potentials that transform the Markov chain into canonical form, and they satisfy a linear system of equations in $\Delta_i = x(u_i) - x(u_{i-1})$ and $\Delta'_i = x(v_i) - x(v_{i-1})$; solving this system we get $\Delta_i = -\Delta'_i = W^{k-i+1}$, for $i = 1, \dots, l$. Any pumping algorithm that starts with 0 potentials and modifies the potentials in each iteration by at most γ will not have a number of iterations less than $\frac{W^{l-1}}{2\gamma}$ on the above example. In particular, the algorithm in Section 4 has $\gamma \leq 1/\min\{p(v,u) : \ (v,u) \in E, \ p(v,u) \neq 0\}$, which is $\Omega(W)$ in our example. We conclude that the running time of the algorithm is $\Omega(W^{l-2}) = W^{\Omega(k)}$ on this example.

References

1. Andersson, D., Miltersen, P.B.: The complexity of solving stochastic games on graphs. In: Dong, Y., Du, D.-Z., Ibarra, O. (eds.) ISAAC 2009. LNCS, vol. 5878, pp. 112–121. Springer, Heidelberg (2009)
2. Beffara, E., Vorobyov, S.: Adapting Gurvich-Karzanov-Khachiyan's algorithm for parity games: Implementation and experimentation. Technical Report 2001-020, Department of Information Technology, Uppsala University (2001), https://www.it.uu.se/research/reports/#2001
3. Beffara, E., Vorobyov, S.: Is randomized Gurvich-Karzanov-Khachiyan's algorithm for parity games polynomial? Technical Report 2001-025, Department of Information Technology, Uppsala University (2001), https://www.it.uu.se/research/reports/#2001
4. Björklund, H., Sandberg, S., Vorobyov, S.: A combinatorial strongly subexponential strategy improvement algorithm for mean payoff games. DIMACS Technical Report 2004-05, DIMACS, Rutgers University (2004)
5. Björklund, H., Vorobyov, S.: Combinatorial structure and randomized subexponential algorithms for infinite games. Theoretical Computer Science 349(3), 347–360 (2005)
6. Björklund, H., Vorobyov, S.: A combinatorial strongly sub-exponential strategy improvement algorithm for mean payoff games. Discrete Applied Mathematics 155(2), 210–229 (2007)
7. Boros, E., Elbassioni, K., Gurvich, V., Makino, K.: Every stochastic game with perfect information admits a canonical form. RRR-09-2009, RUTCOR. Rutgers University (2009)
8. Boros, E., Elbassioni, K., Gurvich, V., Makino, K.: A pumping algorithm for ergodic stochastic mean payoff games with perfect information. RRR-19-2009, RUTCOR. Rutgers University (2009)
9. Boros, E., Gurvich, V.: Why chess and back gammon can be solved in pure positional uniformly optimal strategies? RRR-21-2009, RUTCOR. Rutgers University (2009)
10. Chatterjee, K., Henzinger, T.A.: Reduction of stochastic parity to stochastic mean-payoff games. Inf. Process. Lett. 106(1), 1–7 (2008)
11. Chatterjee, K., Jurdziński, M., Henzinger, T.A.: Quantitative stochastic parity games. In: SODA '04: Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms, pp. 121–130. Society for Industrial and Applied Mathematics, Philadelphia (2004)
12. Condon, A.: The complexity of stochastic games. Information and Computation 96, 203–224 (1992)
13. Condon, A.: An algorithm for simple stochastic games. In: Advances in computational complexity theory. DIMACS series in discrete mathematics and theoretical computer science, vol. 13 (1993)
14. Dhingra, V., Gaubert, S.: How to solve large scale deterministic games with mean payoff by policy iteration. In: Valuetools '06: Proceedings of the 1st international conference on Performance evaluation methodolgies and tools, vol. 12. ACM, New York (2006)
15. Eherenfeucht, A., Mycielski, J.: Positional strategies for mean payoff games. International Journal of Game Theory 8, 109–113 (1979)
16. Friedmann, O.: An exponential lower bound for the parity game strategy improvement algorithm as we know it. In: Symposium on Logic in Computer Science, pp. 145–156 (2009)

17. Gillette, D.: Stochastic games with zero stop probabilities. In: Dresher, M., Tucker, A.W., Wolfe, P. (eds.) Contribution to the Theory of Games III. Annals of Mathematics Studies, vol. 39, pp. 179–187. Princeton University Press, Princeton (1957)
18. Gimbert, H., Horn, F.: Simple stochastic games with few random vertices are easy to solve. In: Amadio, R.M. (ed.) FOSSACS 2008. LNCS, vol. 4962, pp. 5–19. Springer, Heidelberg (2008)
19. Gurvich, V., Karzanov, A., Khachiyan, L.: Cyclic games and an algorithm to find minimax cycle means in directed graphs. USSR Computational Mathematics and Mathematical Physics 28, 85–91 (1988)
20. Halman, N.: Simple stochastic games, parity games, mean payoff games and discounted payoff games are all LP-type problems. Algorithmica 49(1), 37–50 (2007)
21. Hoffman, A.J., Karp, R.M.: On nonterminating stochastic games. Management Science, Series A 12(5), 359–370 (1966)
22. Jurdziński, M.: Deciding the winner in parity games is in UP $\cap$ co-UP. Inf. Process. Lett. 68(3), 119–124 (1998)
23. Jurdziński, M.: Games for Verification: Algorithmic Issues. PhD thesis, Faculty of Science, University of Aarhus, USA (2000)
24. Jurdziński, M., Paterson, M., Zwick, U.: A deterministic subexponential algorithm for solving parity games. In: SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm, pp. 117–123. ACM, New York (2006)
25. Karp, R.M.: A characterization of the minimum cycle mean in a digraph. Discrete Math. 23, 309–311 (1978)
26. Karzanov, A.V., Lebedev, V.N.: Cyclical games with prohibition. Mathematical Programming 60, 277–293 (1993)
27. Kratsch, D., McConnell, R.M., Mehlhorn, K., Spinrad, J.P.: Certifying algorithms for recognizing interval graphs and permutation graphs. In: SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms, pp. 158–167. Society for Industrial and Applied Mathematics, Philadelphia (2003)
28. Liggett, T.M., Lippman, S.A.: Stochastic games with perfect information and time-average payoff. SIAM Review 4, 604–607 (1969)
29. Littman, M.L.: Algorithm for sequential decision making, CS-96-09. PhD thesis, Dept. of Computer Science, Brown Univ., USA (1996)
30. Mine, H., Osaki, S.: Markovian decision process. American Elsevier Publishing Co., New York (1970)
31. Moulin, H.: Extension of two person zero sum games. Journal of Mathematical Analysis and Application 5(2), 490–507 (1976)
32. Moulin, H.: Prolongement des jeux à deux joueurs de somme nulle. Bull. Soc. Math. France, Memoire 45 (1976)
33. Pisaruk, N.N.: Mean cost cyclical games. Mathematics of Operations Research 24(4), 817–828 (1999)
34. Vöge, J., Jurdzinski, M.: A discrete strategy improvement algorithm for solving parity games. In: Emerson, E.A., Sistla, A.P. (eds.) CAV 2000. LNCS, vol. 1855, pp. 202–215. Springer, Heidelberg (2000)
35. Vorobyov, S.: Cyclic games and linear programming. Discrete Applied Mathematics 156(11), 2195–2231 (2008)
36. Zwick, U., Paterson, M.: The complexity of mean payoff games on graphs. Theoretical Computer Science 158(1-2), 343–359 (1996)

On Column-Restricted and Priority Covering Integer Programs[⋆]

Deeparnab Chakrabarty, Elyot Grant, and Jochen Könemann

Department of Combinatorics and Optimization
University of Waterloo, Waterloo, ON, Canada N2L 3G1
deeparnab@gmail.com, elyot@uwaterloo.ca, jochen@uwaterloo.ca

Abstract. In a column-restricted covering integer program (CCIP), all the non-zero entries of any column of the constraint matrix are equal. Such programs capture capacitated versions of covering problems. In this paper, we study the approximability of CCIPs, in particular, their relation to the integrality gaps of the underlying 0,1-CIP.

If the underlying 0,1-CIP has an integrality gap $O(\gamma)$, and assuming that the integrality gap of the *priority version* of the 0,1-CIP is $O(\omega)$, we give a factor $O(\gamma + \omega)$ approximation algorithm for the CCIP. Priority versions of 0,1-CIPs (PCIPs) naturally capture *quality of service* type constraints in a covering problem.

We investigate priority versions of the line (PLC) and the (rooted) tree cover (PTC) problems. Apart from being natural objects to study, these problems fall in a class of fundamental geometric covering problems. We bound the integrality of certain classes of this PCIP by a constant. Algorithmically, we give a polytime exact algorithm for PLC, show that the PTC problem is APX-hard, and give a factor 2-approximation algorithm for it.

1 Introduction

In a *0,1-covering integer program* (0,1-CIP, in short), we are given a constraint matrix $A \in \{0,1\}^{m \times n}$, demands $b \in \mathbb{Z}_+^m$, non-negative costs $c \in \mathbb{Z}_+^n$, and upper bounds $d \in \mathbb{Z}_+^n$, and the goal is to solve the following integer linear program (which we denote by $\texttt{Cov}(A, b, c, d)$).

$$\min\{c^T x \,:\, Ax \geq b, 0 \leq x \leq d, x \text{ integer}\}.$$

Problems that can be expressed as 0,1-CIPs are essentially equivalent to set multi-cover problems, where sets correspond to columns and elements correspond to rows. This directly implies that 0,1-CIPs are rather well understood in terms of approximability: the class admits efficient $O(\log n)$ approximation algorithms and this is best possible unless NP = P. Nevertheless, in many cases one can get better approximations by exploiting the structure of matrix A. For example, it is well known that whenever A is *totally unimodular* (TU)(e.g., see [18]), the

⋆ Supported by NSERC grant no. 288340 and by an Early Research Award.

canonical LP relaxation of a 0,1-CIP is integral; hence, the existence of efficient algorithms for solving linear programs immediately yields fast exact algorithms for such 0,1-CIPs as well.

While a number of general techniques have been developed for obtaining improved approximation algorithms for structured $0,1$-CIPs, not much is known for structured non-$0,1$ CIP instances. In this paper, we attempt to mitigate this problem, by studying the class of *column-restricted covering integer programs* (CCIPs), where all the non-zero entries of any column of the constraint matrix are equal. Such CIPs arise naturally out of $0,1$-CIPs, and the main focus of this paper is to understand how the structure of the underlying 0,1-CIP can be used to derive improved approximation algorithms for CCIPs.

Column-Restricted Covering IPs (CCIPs): Given a 0,1-covering problem $\texttt{Cov}(A,b,c,d)$ and a supply vector $s \in \mathbb{Z}_+^n$, the corresponding CCIP is obtained as follows. Let $A[s]$ be the matrix obtained by replacing all the 1's in the jth column by s_j; that is, $A[s]_{ij} = A_{ij}s_j$ for all $1 \le i \le m, 1 \le j \le n$. The column-restricted covering problem is given by the following integer program.

$$\min\{c^T x : A[s]x \ge b, 0 \le x \le d, x \text{ integer}\}. \qquad (\texttt{Cov}(A[s],b,c,d))$$

CCIPs naturally capture *capacitated* versions of 0,1-covering problems. To illustrate this we use the following 0,1-covering problem called the tree covering problem. The input is a tree $T = (V,E)$ rooted at a vertex $r \in V$, a set of *segments* $\mathcal{S} \subseteq \{(u,v) : u \text{ is a child of } v\}$, non-negative costs c_j for all $j \in \mathcal{S}$, and demands $b_e \in \mathbb{Z}_+$ for all $e \in E$. An edge e is contained in a segment $j = (u,v)$ if e lies on the unique u,v-path in T. The goal is to find a minimum-cost subset C of segments such that each edge $e \in E$ is contained in at least b_e segments of C. When T is just a line, we call the above problem, the *line cover* (LC) problem. In this example, the constraint matrix A has a row for each edge of the tree and a column for each segment in $\mathcal{S}$. It is not too hard to show that this matrix is TU and thus these can be solved exactly in polynomial time.

In the above tree cover problem, suppose each segment $j \in \mathcal{S}$ also has a capacity supply s_j associated with it, and call an edge e covered by a collection of segments C iff the total supply of the segments containing e exceeds the demand of e. The problem of finding the minimum cost subset of segments covering every edge is precisely the column-restricted tree cover problem. The column-restricted line cover problem encodes the minimum knapsack problem and is thus NP-hard.

For general CIPs, the best known approximation algorithm, due to Kolliopoulos and Young [15], has a performance guarantee of $O(1+\log\alpha)$, where α, called the *dilation* of the instance, denotes the maximum number of non-zero entries in any column of the constraint matrix. Nothing better is known for the special case of CCIPs unless one aims for *bicriteria* results where solutions violate the upper bound constraints $x \le d$ (see Section 1.1 for more details).

In this paper, our main aim is to understand how the approximability of a given CCIP instance is determined by the structure of the underlying $0,1$-CIP. In particular, if a $0,1$-CIP has a constant integrality gap, under what circumstances can one get a constant factor approximation for the corresponding CCIP? We make some steps toward finding an answer to this question.

In our main result, we show that there is a constant factor approximation algorithm for CCIP if *two* induced $0,1$-CIPs have constant integrality gap. The first is the underlying original 0,1-CIP. The second is a *priority* version of the 0,1-CIP (PCIP, in short), whose constraint matrix is derived from that of the 0,1-CIP as follows.

Priority versions of Covering IPs (PCIPs): Given a 0,1-covering problem $\texttt{Cov}(A, b, c, d)$, a priority supply vector $s \in \mathbb{Z}^n_+$, and a priority demand vector $\pi \in \mathbb{Z}^m_+$, the corresponding PCIP is as follows. Define $A[s, \pi]$ to be the following 0,1 matrix

$$A[s,\pi]_{ij} = \begin{cases} 1 & : \quad A_{ij} = 1 \text{ and } s_j \geq \pi_i \\ 0 & : \quad \text{otherwise,} \end{cases} \tag{1}$$

Thus, a column j covers row i, only if its priority supply is higher than the priority demand of row i. The priority covering problem is now as follows.

$$\min\{c^T x : A[s,\pi]x \geq \mathbb{1}, 0 \leq x \leq d, x \text{ integer}\}. \qquad (\texttt{Cov}(A[s,\pi], \mathbb{1}, c))$$

We believe that priority covering problems are interesting in their own right, and they arise quite naturally in covering applications where one wants to model *quality of service* (QoS) or priority restrictions. For instance, in the tree cover problem defined above, suppose each segment j has a *quality of service* (QoS) or priority supply s_j associated with it and suppose each edge e has a QoS or priority demand π_e associated with it. We say that a segment j covers e iff j contains e *and* the priority supply of j exceeds the priority demand of e. The goal is to find a minimum cost subset of segments that covers every edge. This is the priority tree cover problem.

Besides being a natural covering problem to study, we show that the priority tree cover problem is a special case of a classical geometric covering problem: that of finding a minimum cost cover of points by axis-parallel rectangles in 3 dimensions. Finding a constant factor approximation algorithm for this problem, even when the rectangles have uniform cost, is a long standing open problem.

We show that although the tree cover is polynomial time solvable, the priority tree cover problem is APX-hard. We complement this with a factor 2 approximation for the problem. Furthermore, we present constant upper bounds for the integrality gap of this PCIP in a number of special cases, implying constant upper bounds on the corresponding CCIPs in these special cases. We refer the reader to Section 1.2 for a formal statement of our results, which we give after summarizing works related to our paper.

1.1 Related Work

There is a rich and long line of work ([9,11,17,19,20]) on approximation algorithms for CIPs, of which we state the most relevant to our work. Assuming no upper bounds on the variables, Srinivasan [19] gave a $O(1 + \log \alpha)$-approximation to the problem (where α is the dilation as before). Later on, Kolliopoulos and Young [15] obtained the same approximation factor, respecting the upper bounds. However, these algorithms didn't give any better results when special structure of the constraint matrix was known. On the hardness side, Trevisan [21] showed that it is NP-hard to obtain a $(\log \alpha - O(\log \log \alpha))$-approximation algorithm even for 0,1-CIPs.

The most relevant work to this paper is that of Kolliopoulos [12]. The author studies CCIPs which satisfy a rather strong assumption, called the *no bottleneck assumption*, that the supply of any column is smaller than the demand of any row. Kolliopoulos [12] shows that if one is allowed to violate the upper bounds by a multiplicative constant, then the integrality gap of the CCIP is within a constant factor of that of the original 0,1-CIP[1]. As the author notes such a violation is necessary; otherwise the CCIP has unbounded integrality gap. If one is not allowed to violated upper bounds, nothing better than the result of [15] is known for the special case of CCIPs.

Our work on CCIPs parallels a large body of work on column-restricted *packing* integer programs (CPIPs). Assuming the *no-bottleneck assumption*, Kolliopoulos and Stein [14] show that CPIPs can be approximated asymptotically as well as the corresponding 0,1-PIPs. Chekuri et al. [7] subsequently improve the constants in the result from [14]. These results imply constant factor approximations for the column-restricted tree *packing* problem under the no-bottleneck assumption. Without the no-bottleneck assumption, however, only polylogarithmic approximation is known for the problem [6].

The only work on priority versions of covering problems that we are aware of is due to Charikar, Naor and Schieber [5] who studied the priority Steiner tree and forest problems in the context of QoS management in a network multicasting application. Charikar et al. present a $O(\log n)$-approximation algorithm for the problem, and Chuzhoy et al. [8] later show that no efficient $o(\log \log n)$ approximation algorithm can exist unless NP $\subseteq$ DTIME$(n^{\log \log \log n})$ (n is the number of vertices).

To the best of our knowledge, the column-restricted or priority versions of the line and tree cover problem have not been studied. The best known approximation algorithm known for both is the $O(\log n)$ factor implied by the results of [15] stated above. However, upon completion of our work, Nitish Korula [16] pointed out to us that a 4-approximation for column-restricted line cover is implicit in a result of Bar-Noy et al. [2]. We remark that their algorithm is not LP-based, although our general result on CCIPs is.

1.2 Technical Contributions and Formal Statement of Results

Given a 0,1-CIP $\texttt{Cov}(A, b, c, d)$, we obtain its *canonical LP relaxation* by removing the integrality constraint. The *integrality gap* of the CIP is defined as the

[1] Such a result is implicit in the paper; the author only states a $O(\log \alpha)$ integrality gap.

supremum of the ratio of optimal IP value to optimal LP value, taken over all non-negative integral vectors b, c, and d. The integrality gap of an IP captures how much the integrality constraint affects the optimum, and is an indicator of the *strength* of a linear programming formulation.

CCIPs: Suppose the CCIP is $\texttt{Cov}(A[s], b, c, d)$. We make the following two assumptions about the integrality gaps of the 0,1 covering programs, both the original 0,1-CIP and the priority version of the 0,1-CIP.

Assumption 1. *The integrality gap of the original 0,1-CIP is $\gamma \geq 1$. Specifically, for any non-negative integral vectors b, c, and d, if the canonical LP relaxation to the CIP has a fractional solution x, then one can find in polynomial time an integral feasible solution to the CIP of cost at most $\gamma \cdot c^T x$. We stress here that the entries of b, c, d could be 0 as well as ∞.*

Assumption 2. *The integrality gap of the PCIP is $\omega \geq 1$. Specifically, for any non-negative integral vectors s, π, c, if the canonical LP relaxation to the PCIP has a fractional solution x, then one can find in polynomial time, an integral feasible solution to the PCIP of cost at most $\omega \cdot c^T x$.*

We give an LP-based approximation algorithm for solving CCIPs. Since the canonical LP relaxation of a CCIP can have unbounded integrality gap, we strengthen it by adding a set of valid constraints called the *knapsack cover constraints*. We show that the integrality gap of this strengthened LP is $O(\gamma+\omega)$, and can be used to give a polynomial time approximation algorithm.

Theorem 1. *Under Assumptions 1 and 2, there is a $(24\gamma + 8\omega)$-approximation algorithm for column-restricted CIPs.*

Knapsack cover constraints to strengthen LP relaxations were introduced in [1,10,22]; Carr et al. [3] were the first to employ them in the design approximation algorithms. The paper of Kolliopoulos and Young [15] also use these to get their result on general CIPs.

The main technique in the design of algorithms for column-restricted problems is *grouping-and-scaling* developed by Kolliopoulos and Stein [13,14] for packing problems, and later used by Kolliopoulos [12] in the covering context. In this technique, the *columns* of the matrix are divided into groups of 'close-by' supply values; in a single group, the supply values are then scaled to be the same; for a single group, the integrality gap of the original 0,1-CIP is invoked to get an integral solution for that group; the final solution is a 'union' of the solutions over all groups.

There are two issues in applying the technique to the new strengthened LP relaxation of our problem. Firstly, although the original constraint matrix is column-restricted, the new constraint matrix with the knapsack cover constraints is not. Secondly, unless additional assumptions are made, the current grouping-and-scaling analysis doesn't give a handle on the degree of violation of the upper bound constraints. This is the reason why Kolliopoulos [12] needs the strong no-bottleneck assumption.

We get around the first difficulty by grouping the *rows* as well, into those that get most of their coverage from columns not affected by the knapsack constraints, and the remainder. On the first group of rows, we apply a subtle modification to the vanilla grouping-and-scaling analysis and obtain a $O(\gamma)$ approximate feasible solution satisfying these rows; we then show that one can treat the remainder of the rows as a PCIP and get a $O(\omega)$ approximate feasible solution satisfying them, using Assumption 2. Combining the two gives the $O(\gamma+\omega)$ factor. The full details are given in Section 2.

We stress here that apart from the integrality gap assumptions on the 0,1-CIPs, we do not make any other assumption (like the no-bottleneck assumption). In fact, we can use the modified analysis of the grouping-and-scaling technique to get a similar result as [12] for approximating CCIPs violating the upper-bound constraints, under a *weaker* assumption than the no-bottleneck assumption. The no-bottleneck assumption states that the supply of *any* column is less than the demand of *any* row. In particular, even though a column has entry 0 on a certain row, its supply needs to be less than the demand of that row. We show that if we weaken the no-bottleneck assumption to assuming that the supply of a column j is less than the demand of any row i only if $A[s]_{ij}$ is positive, a similar result can be obtained via our modified analysis.

Theorem 2. *Under assumption 1 and assuming $A_{ij}s_j \le b_i$, for all i,j, given a fractional solution x to the canonical LP relaxation of $\texttt{Cov}(A[s],b,c,d)$, one can find an integral solution $x^{\texttt{int}}$ whose cost $c \cdot x^{\texttt{int}} \le 10\gamma(c\cdot x)$ and $x^{\texttt{int}} \le 10d$.*

Priority Covering Problems. In the following, we use PLC and PTC to refer to the priority versions of the line cover and tree cover problems, respectively. Recall that the constraint matrices for line and tree cover problems are totally unimodular, and the integrality of the corresponding 0,1-covering problems is therefore 1 in both case. It is interesting to note that the 0,1-coefficient matrices for PLC and PTC are not totally unimodular in general. The following integrality gap bound is obtained via a primal-dual algorithm.

Theorem 3. *The canonical LP for priority line cover has an integrality gap of at least $3/2$ and at most 2.*

In the case of tree cover, we obtain constant upper bounds on the integrality gap for the case $c=\mathbb{1}$, that is, for the minimum cardinality version of the problem. We believe that the PCIP for the tree cover problem with general costs also has a constant integrality gap. On the negative side, we can show an integrality gap of at least $\frac{e}{e-1}$.

Theorem 4. *The canonical LP for* unweighted *PTC has an integrality gap of at most 6.*

We obtain the upper bound by taking a given PTC instance and a fractional solution to its canonical LP, and decomposing it into a collection of PLC instances with corresponding fractional solutions, with the following two properties. First, the total cost of the fractional solutions of the PLC instances is within a constant of the cost of the fractional solution of the PTC instance. Second, union of integral solutions to the PLC instances gives an integral solution to the PTC instance. The upper bound follows from Theorem 3. Using Theorem 1, we get the following as an immediate corollary.

Corollary 1. *There are $O(1)$-approximation algorithms for column-restricted line cover and the cardinality version of the column-restricted tree cover.*

We also obtain the following combinatorial results.

Theorem 5. *There is a polynomial-time exact algorithm for PLC.*

Theorem 6. *PTC is APX-hard, even when all the costs are unit.*

Theorem 7. *There is an efficient 2-approximation algorithm for PTC.*

The algorithm for PLC is a non-trivial dynamic programming approach that makes use of various structural observations about the optimal solution. The approximation algorithm for PTC is obtained via a similar decomposition used to prove Theorem 4.

We end by noting some interesting connections between the priority tree covering problem and set covering problems in computational geometry. The *rectangle cover* problem in 3-dimensions is the following: given a collection of points P in $\mathbb{R}^3$, and a collection C of axis-parallel rectangles with costs, find a minimum cost collection of rectangles that covers every point. We believe studying the PTC problem could give new insights into the rectangle cover problem.

Theorem 8. *The priority tree covering problem is a special case of the rectangle cover problem in 3-dimensions.*

Due to space restrictions, we omit many proofs. A full version of the paper is available [4].

2 General Framework for Column Restricted CIPs

In this section we prove Theorem 1. Our goal is to round a solution to a LP relaxation of $\texttt{Cov}(A[s], b, c, d)$ into an approximate integral solution. We strengthen the following canonical LP relaxation of the CCIP

$$\min\{c^T x \ : \ A[s]x \geq b, 0 \leq x \leq d, x \geq 0\}$$

by adding valid *knapsack cover* constraints. In the following we use $\mathcal{C}$ for the set of columns and $\mathcal{R}$ for the set of rows of A.

2.1 Strengthening the Canonical LP Relaxation

Let $F \subset \mathcal{C}$ be a subset of the columns in the column restricted CIP $\texttt{Cov}(A[s], b, c, d)$. For all rows $i \in \mathcal{R}$, define $b_i^F = \max\{0, b_i - \sum_{j\in F} A[s]_{ij} d_j\}$ to be the residual demand of row i w.r.t. F. Define matrix $A^F[s]$ by letting

$$A^F[s]_{ij} = \begin{cases} \min\{A[s]_{ij}, b_i^F\} & : \quad j \in \mathcal{C} \setminus F \\ 0 & : \quad j \in F, \end{cases} \tag{2}$$

for all $i \in \mathcal{C}$ and for all $j \in \mathcal{R}$. The following *Knapsack-Cover* (KC) inequality

$$\sum_{j\in\mathcal{C}} A^F[s]_{ij} x_j \geq b_i^F$$

is valid for the set of all integer solutions x for $\texttt{Cov}(A[s], b, c, d)$. Adding the set of all KC inequalities yields the following stronger LP formulation CIP. We note that the LP is not column-restricted, in that, different values appear on the same column of the new constraint matrix.

$$\begin{aligned} \texttt{opt}_P := \min \quad & \sum_{j\in\mathcal{C}} c_j x_j && && \text{(P)} \\ \text{s.t.} \quad & \sum_{j\in\mathcal{C}} A^F[s]_{ij} x_j \geq b_i^F && \forall F \subseteq \mathcal{C}, \forall i \in \mathcal{R} && (3) \\ & 0 \leq x_j \leq d_j && \forall j \in \mathcal{C} && \end{aligned}$$

It is not known whether (P) can be solved in polynomial time. For $\alpha \in (0, 1)$, call a vector x^* α-relaxed if its cost is at most $\texttt{opt}_P$, and if it satisfies (3) for $F = \{j \in \mathcal{C} : x_j^* \geq \alpha d_j\}$. An α-relaxed solution to (P) can be computed efficiently for any α. To see this note that one can check whether a candidate solution satisfies (3) for a set F; we are done if it does, and otherwise we have found an inequality of (P) that is violated, and we can make progress via the ellipsoid method. Details can be found in [3] and [15].

We fix an $\alpha \in (0, 1)$, specifying its precise value later. Compute an α-relaxed solution, x^*, for (P), and let $F = \{j \in \mathcal{C} : x_j^* \geq \alpha d_j\}$. Define $\bar{x}$ as, $\bar{x}_j = x_j^*$ if $j \in \mathcal{C} \setminus F$, and $\bar{x}_j = 0$, otherwise. Since x^* is an α-relaxed solution, we get that $\bar{x}$ is a feasible fractional solution to the *residual* CIP, $\texttt{Cov}(A^F[s], b^F, c, \alpha d)$. In the next subsection, our goal will be to obtain an *integral* feasible solution to the covering problem $\texttt{Cov}(A^F[s], b^F, c, d)$ using $\bar{x}$. The next lemma shows how this implies an approximation to our original CIP.

Lemma 1. *If there exists an integral feasible solution, $x^{\texttt{int}}$, to $\texttt{Cov}(A^F[s], b^F, c, d)$ with $c^T x^{\texttt{int}} \leq \beta \cdot c^T \bar{x}$, then there exists a $\max\{1/\alpha, \beta\}$-factor approximation to $\texttt{Cov}(A[s], b, c, d)$.*

2.2 Solving the Residual Problem

In this section we use a feasible fractional solution $\bar{x}$ of $\texttt{Cov}(A^F[s], b^F, c, \alpha d)$, to obtain an *integral* feasible solution $x^{\texttt{int}}$ to the covering problem $\texttt{Cov}(A^F[s], b^F, c, d)$, with $c^T x^{\texttt{int}} \leq \beta c^T \bar{x}$ for $\beta = 24\gamma + 8\omega$. Fix $\alpha = 1/24$.

***Converting to Powers of* 2.** For ease of exposition, we first modify the input to the residual problem $\texttt{Cov}(A^F[s], b^F, c, d)$ so that all entries of are powers of 2. For every $i \in \mathcal{R}$, let $\bar{b}_i$ denote the smallest power of 2 larger than b_i^F. For every column $j \in \mathcal{C}$, let $\bar{s}_j$ denote the largest power of 2 smaller than s_j.

Lemma 2. *$y = 4\bar{x}$ is feasible for $\texttt{Cov}(A^F[\bar{s}], \bar{b}, c, 4\alpha d)$.*

Partitioning the rows. We call $\bar{b}_i$ the residual demand of row i. For a row i, a column $j \in \mathcal{C}$ is *i-large* if the supply of j is at least the residual demand of row i; it is *i-small* otherwise. Formally,

$$\begin{aligned} \mathcal{L}_i &= \{j \in \mathcal{C} \,:\, A_{ij} = 1, \bar{s}_j \geq \bar{b}_i\} \quad \text{is the set of } i\text{-large columns} \\ \mathcal{S}_i &= \{j \in \mathcal{C} \,:\, A_{ij} = 1, \bar{s}_j < \bar{b}_i\} \quad \text{is the set of } i\text{-small columns} \end{aligned}$$

Recall the definition from (2), $A^F[\bar{s}]_{ij} = \min(A[\bar{s}]_{ij}, b_i^F)$. Therefore, $A^F[\bar{s}]_{ij} = A_{ij}b_i^F$ for all $j \in \mathcal{L}_i$ since $\bar{s}_j \geq \bar{b}_i \geq b_i^F$; and $A^F[\bar{s}]_{ij} = A_{ij}\bar{s}_j$ for all $j \in \mathcal{S}_i$, since being powers of 2, $\bar{s}_j < \bar{b}_i$ implies, $\bar{s}_j \leq \bar{b}_i/2 \leq b_i^F$.

We now partition the rows into large and small depending on which columns most of their coverage comes from. Formally, call a row $i \in \mathcal{R}$ *large* if

$$\sum_{j \in \mathcal{S}_i} A^F[\bar{s}]_{ij} y_j \leq \sum_{j \in \mathcal{L}_i} A^F[\bar{s}]_{ij} y_j,$$

and small otherwise. Note that Lemma 2 together with the fact that each column in row i's support is either small or large implies,

$$\begin{aligned} \sum_{j \in \mathcal{L}_i} A^F[\bar{s}]_{ij} y_j &\geq \bar{b}_i/2, \text{ for all large rows } i, \text{ and} \\ \sum_{j \in \mathcal{S}_i} A^F[\bar{s}]_{ij} y_j &\geq \bar{b}_i/2, \text{ for all small rows } i. \end{aligned}$$

Let $\mathcal{R}_L$ and $\mathcal{R}_S$ be the set of large and small rows.

In the following, we address small and large rows separately. We compute a pair of integral solutions $x^{\texttt{int},\mathcal{S}}$ and $x^{\texttt{int},\mathcal{L}}$ that are feasible for the small and large rows, respectively. We then obtain $x^{\texttt{int}}$ by letting

$$x_j^{\texttt{int}} = \max\{x_j^{\texttt{int},\mathcal{S}}, x_j^{\texttt{int},\mathcal{L}}\}, \tag{4}$$

for all $j \in \mathcal{C}$.

Small rows. For these rows we use the grouping-and-scaling technique a la [7,12,13,14]. However, as mentioned in the introduction, we use a modified analysis that bypasses the no-bottleneck assumptions made by earlier works.

Lemma 3. *We can find an integral solution $x^{\texttt{int},\mathcal{S}}$ such that*
a) $x_j^{\texttt{int},\mathcal{S}} \leq d_j$ for all j,
b) $\sum_{j \in \mathcal{C}} c_j x_j^{\texttt{int},\mathcal{S}} \leq 24\gamma \sum_{j \in \mathcal{C}} c_j \bar{x}_j$, and
c) for every small row $i \in \mathcal{R}_S$, $\sum_{j \in \mathcal{C}} A^F[s]_{ij} x_j^{\texttt{int},\mathcal{S}} \geq b_i^F$.

Proof. (Sketch) Since the rows are small, for any row i, we can zero out the entries that are larger than $\bar{b}_i$, and still $2y$ will be a feasible solution. Note that, now in each row, the entries are $< \bar{b}_i$, and thus are at most $\bar{b}_i/2$ (everything being powers of 2). We stress that it could be that $\bar{b}_i$ of some row is less than the entry in some other row, that is, we don't have the no-bottleneck assumption. However, when a particular row i is fixed, $\bar{b}_i$ is at least any entry of the matrix in the ith row. Our modified analysis of grouping and scaling then makes the proof go through.

We *group* the columns into classes that have s_j as the same power of 2, and for each row i we let $\bar{b}_i^{(t)}$ be the contribution of the class t columns towards the demand of row i. The columns of class t, the small rows, and the demands $\bar{b}_i^{(t)}$ form a CIP where all non-zero entries of the matrix are the same power of 2. We scale both the constraint matrix and $\bar{b}_i^{(t)}$ down by that power of 2 to get a 0,1-CIP, and using assumption 1, we get an integral solution to this 0,1-CIP. Our final integral solution is obtained by concatenating all these integral solutions over all classes.

Till now the algorithm is the standard grouping-and-scaling algorithm. The difference lies in our analysis in proving that this integral solution is feasible for the original CCIP. Originally the no-bottleneck assumption was used to prove this. However, we show that, since the column values in different classes are geometrically decreasing, the weaker assumption of $\bar{b}_i$ being at least any entry in the ith row is enough to make the analysis go through.

This completes the sketch of the proof.

Large rows. The large rows can be showed to be a PCIP problem and thus Assumption 2 can be invoked to get an analogous lemma to Lemma 3.

Lemma 4. *We can find an integral solution* $x^{\mathtt{int},\mathcal{L}}$ *such that*
a) $x_j^{\mathtt{int},\mathcal{L}} \leq 1$ *for all* j,
b) $\sum_{j\in\mathcal{C}} c_j x_j^{\mathtt{int},\mathcal{S}} \leq 8\omega \sum_{j\in\mathcal{C}} c_j \bar{x}_j$, *and*
c) for every large row $i \in \mathcal{R}_L$, $\sum_{j\in\mathcal{C}} A^F[s]_{ij} x_j^{\mathtt{int},\mathcal{S}} \geq b_i^F$.

Define $x^{\mathtt{int}}$ as $x_j^{\mathtt{int}} = \max\{x_j^{\mathtt{int},\mathcal{S}}, x_j^{\mathtt{int},\mathcal{L}}\}$ for all j; using the previous two lemmas and Lemma 1, this integral solution proves Theorem 1.

3 Priority Line Cover

In this extended abstract, we show that the integrality gap of the canonical linear programming relaxation of PLC is at most 2. Subsequently, we sketch an exact combinatorial algorithm for the problem.

3.1 Canonical LP Relaxation: Integrality Gap

We start with the canonical LP relaxation for PLC and its dual in Figure 1.

We use the terminology an edge e is larger than f, if $\pi_e \geq \pi_f$. The algorithm maintains a set of segments Q initially empty. Call an edge e *unsatisfied* if no

$$\min\Big\{\sum_{j\in\mathcal{S}} c_j x_j : \quad x \in R_+^{\mathcal{S}} \qquad \text{(Primal)} \qquad \sum_{j\in\mathcal{S}: j \text{ covers } e} x_j \geq 1, \quad \forall e \in E\Big\}$$

$$\max\Big\{\sum_{e\in E} y_e : \quad y \in R_+^{E} \quad \text{(Dual)} \qquad \sum_{e\in E: j \text{ covers } e} y_e \leq c_j, \quad \forall j \in \mathcal{S}\Big\}$$

Fig. 1. The PLC canonical LP relaxation and its dual

segment in Q covers e and let U be the set of unsatisfied edges. The algorithm picks the largest edge in U and raises the dual value y_e till some segments becomes tight. The segments with the farthest left-end point and the farthest right-end point are picked in Q, and all edges contained in any of them are removed from U. Note that since we choose the largest in U, all such edges are covered. The algorithm repeats this process till U becomes $\emptyset$, that is, all edges are covered. The final set of segments is obtained by a reverse delete step, where a segment is deleted if its deletion doesn't make any edge uncovered.

The algorithm is a factor 2 approximation algorithm. To show this it suffices by a standard argument for analysing primal-dual algorithms, that any edge with a positive dual y_e is contained in at most two segments in Q. These two segments correspond to the left-most and the right-most segments that cover e; it is not too hard to show if something else covers e, then either e has zero dual, or the third segment is removed in the reverse delete step.

3.2 An Exact Algorithm for PLC

We sketch the exact algorithm for PLC. A segment j covers only a subset of edges it contains. We call a contiguous interval of edges covered by j, a *valley* of j. The uncovered edges form *mountains*. Thus a segment can be thought of as forming a series of valleys and mountains.

Given a solution $S \subseteq \mathcal{S}$ to the PLC (or even a PTC) instance, we say that segment $j \in S$ is *needed* for edge e if j is the unique segment in S that covers e. We let $E_{S,j}$ be the set of edges that need segment j. We say a solution is *valley-minimal* if it satisfies the following two properties: (a) If a segment j is needed for edge e that lies in the valley v of j, then no higher supply segment of S intersects this valley v, and (b) every segment j is needed for its last and first edges. We show that an optimum solution can be assumed to be valley-minimal, and thus it suffices to find the minimum cost valley-minimal solution.

The crucial observation follows from properties (a) and (b) above. The valley-minimality of solution S implies that there is a unique segment $j \in S$ that covers the first edge of the line. At a very high level, we may now use j to decompose the given instance into a set of *smaller* instances. For this we first observe that each of the remaining segments in $S \setminus \{j\}$ is either fully contained in the strict interior of segment j, or it is disjoint from j, and lies to the right of it. The set of all segments that are disjoint from j form a feasible solution for the smaller PLC instance induced by the portion of the original line instance to the right of j. On the other hand, we show how to reduce the problem of finding an optimal solution for the part of the line contained in j to a single shortest-path computation in an auxiliary digraph. Each of the arcs in this digraph once again corresponds to a smaller sub-instance of the original PLC instance, and its cost is that of its optimal solution. The algorithm follows by dynamic programming.

4 Priority Tree Cover

In this extended abstract, we sketch a factor 2 approximation for the PTC problem, and show how the PTC problem is a special case of the 3 dimensional rectangle cover problem. For the APX hardness and the integrality gap of the unweighted PTC LP, we refer the reader to the full version.

4.1 An Approximation Algorithm for PTC

We use the exact algorithm for PLC to get the factor 2 algorithm for PTC. The crucial idea is the following. Given an optimum solution $S^* \subseteq \mathcal{S}$, we can partition the edge-set E of T into disjoint sets $E_1, \ldots, E_p$, and partition two copies of S^* into $S_1, \ldots, S_p$, such that E_i is a path in T for each i, and S_i is a priority line cover for the path E_i. Using this, we describe the 2-approximation algorithm which proves Theorem 7.

Proof of Theorem 7: For any two vertices t (top) and b (bottom) of the tree T, such that t is an ancestor of b, let P_{tb} be the unique path from b to t. Note that P_{tb}, together with the restrictions of the segments in $\mathcal{S}$to P_{tb}, defines an instance of PLC. Therefore, for each pair t and b, we can compute the optimal solution to the corresponding PLC instance using the exact algorithm; let the cost of this solution be c'_{tb}. Create an instance of the 0,1-tree cover problem with T and segments $\mathcal{S}' := \{(t, b) : t \text{ is an ancestor of } b\}$ with costs c'_{tb}. Solve the 0,1-tree cover instance exactly (recall we are in the rooted version) and for the segments (t, b) in $\mathcal{S}'$ returned, return the solution of the corresponding PLC instance of cost c'_{tb}.

One now uses the decomposition above to obtain a solution to the 0,1-tree cover problem $(T, \mathcal{S}')$ of cost at most 2 times the cost of S^*. This proves the theorem. The segments in $\mathcal{S}'$ picked are precisely the segments corresponding to paths E_i, $i = 1, \ldots, p$ and each S_i is a solution to the PLC instance. Since we find the optimum PLC, there is a solution to $(T, \mathcal{S}')$ with costs c' of cost less than total cost of segments in $S_1 \cup \cdots \cup S_p$. But that cost is at most twice the cost of S^* since each segment of S^* is in at most two S_i's.

4.2 Priority Tree Cover and Geometric Covering Problems

We sketch how the PTC problem can be encoded as a rectangle cover problem. To do so, an auxiliary problem is defined, which we call 2-PLC.

2-Priority Line Cover (2-PLC). The input is similar to PLC, except each segment and edge has now an ordered pair of priorities, and a segment covers an edge it contains iff each of the priorities of the segment exceeds the corresponding priority of the edge. The goal, as in PLC, is to find a minimum cost cover.

It is not too hard to show 2-PLC is a special case of rectangle cover. The edges correspond to points in 3 dimension and segments correspond to rectangles in 3-dimension; dimensions encoded by the linear coordinates on the line, and the two priority values. In general, p-PLC can be shown to be a special case of $(p + 1)$-dimensional rectangle cover.

What is more involved is to show PTC is a special case of 2-PLC. To do so, we run two DFS orderings on the tree, where the order in which children of a node are visited is completely opposite in the two DFS orderings. The first ordering gives the order in which these edges must be placed on a line. The second gives one of the priorities for the edges. The second priority of the edges comes from the original priority in PTC. It can be shown that the segments priorities can be so set that the feasible solutions are precisely the same in both the instances proving Theorem 8.

5 Concluding Remarks

In this paper we studied column restricted covering integer programs. In particular, we studied the relationship between CCIPs and the underlying 0,1-CIPs. We conjecture that the approximability of a CCIP should be asymptotically within a constant factor of the integrality gap of the original 0,1-CIP. We couldn't show this; however, if the integrality gap of a PCIP is shown to be within a constant of the integrality gap of the 0,1-CIP, then we will be done. At this point, we don't even know how to prove that PCIPs of special 0,1-CIPS, those whose constraint matrices are totally unimodular, have constant integrality gap. Resolving the case of PTC is an important step in this direction, and hopefully in resolving our conjecture regarding CCIPs.

References

1. Balas, E.: Facets of the knapsack polytope. Math. Programming 8, 146–164 (1975)
2. Bar-Noy, A., Bar-Yehuda, R., Freund, A., Naor, J., Schieber, B.: A unified approach to approximating resource allocation and scheduling. J. ACM 48(5), 1069–1090 (2001)
3. Carr, R.D., Fleischer, L.K., Leung, V.J., Phillips, C.A.: Strengthening integrality gaps for capacitated network design and covering problems. In: Proceedings of ACM-SIAM Symposium on Discrete Algorithms, pp. 106–115 (2000)
4. Chakrabarty, D., Grant, E., Könemann, J.: On column-restricted and priority covering integer programs. arXiv eprint (2010)
5. Charikar, M., Naor, J., Schieber, B.: Resource optimization in qos multicast routing of real-time multimedia. IEEE/ACM Trans. Netw. 12(2), 340–348 (2004)

6. Chekuri, C., Ene, A., Korula, N.: Unsplittable flow in paths and trees and column-restricted packing integer programs. In: Proceedings of International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (2009) (to appear)
7. Chekuri, C., Mydlarz, M., Shepherd, F.B.: Multicommodity demand flow in a tree and packing integer programs. ACM Trans. Alg. 3(3) (2007)
8. Chuzhoy, J., Gupta, A., Naor, J., Sinha, A.: On the approximability of some network design problems. ACM Trans. Alg. 4(2) (2008)
9. Dobson, G.: Worst-case analysis of greedy heuristics for integer programming with non-negative data. Math. Oper. Res. 7(4), 515–531 (1982)
10. Hammer, P., Johnson, E., Peled, U.: Facets of regular 0-1 polytopes. Math. Programming 8, 179–206 (1975)
11. Hochbaum, D.S.: Approximation algorithms for the set covering and vertex cover problems. SIAM Journal on Computing 11(3), 555–556 (1982)
12. Kolliopoulos, S.G.: Approximating covering integer programs with multiplicity constraints. Discrete Appl. Math. 129(2-3), 461–473 (2003)
13. Kolliopoulos, S.G., Stein, C.: Approximation algorithms for single-source unsplittable flow. SIAM Journal on Computing 31(3), 919–946 (2001)
14. Kolliopoulos, S.G., Stein, C.: Approximating disjoint-path problems using packing integer programs. Math. Programming 99(1), 63–87 (2004)
15. Kolliopoulos, S.G., Young, N.E.: Approximation algorithms for covering/packing integer programs. J. Comput. System Sci. 71(4), 495–505 (2005)
16. Korula, N.: Private Communication (2009)
17. Rajagopalan, S., Vazirani, V.V.: Primal-dual RNC approximation algorithms for (multi)set (multi)cover and covering integer programs. In: Proceedings of IEEE Symposium on Foundations of Computer Science (1993)
18. Schrijver, A.: Combinatorial optimization. Springer, New York (2003)
19. Srinivasan, A.: Improved approximation guarantees for packing and covering integer programs. SIAM Journal on Computing 29(2), 648–670 (1999)
20. Srinivasan, A.: An extension of the lovász local lemma, and its applications to integer programming. SIAM Journal on Computing 36(3), 609–634 (2006)
21. Trevisan, L.: Non-approximability results for optimization problems on bounded degree instances. In: Proceedings of ACM Symposium on Theory of Computing, pp. 453–461 (2001)
22. Wolsey, L.: Facets for a linear inequality in 0-1 variables. Math. Programming 8, 168–175 (1975)

On k-Column Sparse Packing Programs

Nikhil Bansal[1], Nitish Korula[2],
Viswanath Nagarajan[1], and Aravind Srinivasan[3]

[1] IBM T.J. Watson Research Center, Yorktown Heights, NY 10598
{nikhil,viswanath}@us.ibm.com

[2] Dept. of Computer Science, University of Illinois, Urbana IL 61801,
Partially supported by NSF grant CCF 07-28782 and a University of Illinois Dissertation Completion Fellowship
nkorula2@illinois.edu

[3] Dept. of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, Supported in part by NSF ITR Award CNS-0426683 and NSF Award CNS-0626636
srin@cs.umd.edu

Abstract. We consider the class of packing integer programs (PIPs) that are *column sparse*, where there is a specified upper bound k on the number of constraints that each variable appears in. We give an improved $(ek+o(k))$-approximation algorithm for k-column sparse PIPs. Our algorithm is based on a linear programming relaxation, and involves randomized rounding combined with alteration. We also show that the integrality gap of our LP relaxation is at least $2k-1$; it is known that even special cases of k-column sparse PIPs are $\Omega(\frac{k}{\log k})$-hard to approximate.

We generalize our result to the case of maximizing monotone submodular functions over k-column sparse packing constraints, and obtain an $\left(\frac{e^2 k}{e-1} + o(k)\right)$-approximation algorithm. In obtaining this result, we prove a new property of submodular functions that generalizes the fractionally subadditive property, which might be of independent interest.

1 Introduction

Packing integer programs (PIPs) are those of the form:

$$\max \left\{ w^T\ x \mid Sx \le c,\ x \in \{0,1\}^n \right\}, \qquad \text{where } w \in \mathbb{R}_+^n,\ c \in \mathbb{R}_+^m \text{ and } S \in \mathbb{R}_+^{m\times n}.$$

Above, n is the number of variables/columns, m is the number of rows/constraints, S is the matrix of *sizes*, c is the *capacity* vector, and w is the *weight* vector. In general, PIPs are very hard to approximate: a special case is the classic independent set problem, which is NP-Hard to approximate within a factor of $n^{1-\epsilon}$ [30], whereas an n-approximation is trivial. Thus, various special cases of PIPs are often studied. Here, we consider *k-column sparse PIPs* (denoted k-CS-PIP), which are PIPs where the number of non-zero entries in each column of matrix S is at most k. This is a fairly general class and models several basic problems such as k-set packing [19] and independent set in graphs with degree at most k.

Recently, in a somewhat surprising result, Pritchard [25] gave an algorithm for k-CS-PIP where the approximation ratio only depends on k; this is useful when k is small. This result is surprising because in contrast, no such guarantee is possible for k-row sparse PIPs. In particular, the independent set problem on general graphs is a 2-row sparse PIP, but is $n^{1-o(1)}$-hard to approximate. Pritchard's algorithm [25] had an approximation ratio of $2^k \cdot k^2$. Subsequently, an improved $O(k^2)$ approximation algorithm was obtained independently by Chekuri *et al.* [14] and Chakrabarty-Pritchard [11].

Our Results: In this paper, we first consider the k-CS-PIP problem and obtain an $(ek + o(k))$-approximation algorithm for it. Our algorithm is based on solving a strengthened version of the natural LP relaxation of k-CS-PIP, and then performing randomized rounding followed by suitable alterations. In the *randomized rounding* step, we pick each variable independently (according to its LP value) and obtain a set of variables with good expected weight; however some constraints may be violated. Then in the *alteration* step, we drop some variables so as to satisfy all constraints, while still having good expected weight. A similar approach can be used with the natural relaxation for k-CS-PIP obtained by simply dropping the integrality constraints on the variables; this gives a slightly weaker $8k$-approximation bound. However, the analysis of this weaker result is much simpler and we thus present it first. To obtain the $ek + o(k)$ bound, we construct a stronger LP relaxation by adding additional valid constraints to the natural relaxation for k-CS-PIP. The analysis of our rounding procedure is based on exploiting these additional constraints and using the positive correlation between various probabilistic events via the FKG inequality. Due to space constraints, we omit some details; these and other omitted proofs can be found in the full version of this paper [5].

Our result is almost the best possible that one can hope for using the LP based approach. We show that the integrality gap of the strengthened LP is at least $2k - 1$, so our analysis is tight up to a small constant factor $e/2 \approx 1.36$ for large values of k. Even without restricting to LP based approaches, an $O(k)$ approximation is nearly best possible since it is NP-Hard to obtain an $O(k/\log k)$-approximation for the special case of k-set packing [18]. We also obtain improved results for k-CS-PIP when capacities are large relative to the sizes. In particular, we obtain a $\Theta(k^{1/\lfloor B \rfloor})$-approximation algorithm for k-CS-PIP, where $B := \min_{i\in[n],j\in[m]} c_j/s_{ij}$ measures the relative slack between the capacities c and sizes S. We also show that this result is tight up to constant factors relative to its LP relaxation.

Our second main result is for the more general problem of maximizing a monotone submodular function over packing constraints that are k-column sparse. This problem is a common generalization of maximizing a submodular function over (a) a k-dimensional knapsack [22], and (b) the intersection of k partition matroids [24]. Here, we obtain an $\left(\frac{e^2k}{e-1} + o(k)\right)$-approximation algorithm for this problem. Our algorithm uses the continuous greedy algorithm of Vondrák [29] in conjunction with our randomized rounding plus alteration based approach. However, it turns out that the analysis of the approximation guarantee is much more intricate: In particular, we need a generalization of a result of Feige [16] that shows that submodular functions are also *fractionally subadditive.* See Section 3 for a statement of the new result, Theorem 5, and related context. This generalization is based on an interesting connection between submodular functions and the FKG inequality. We believe that this result and technique might be of further use in the study of submodular optimization.

Related Previous Work: Various special cases of k-CS-PIP have been extensively studied. An important special case is the *k-set packing* problem, where given a collection of sets of cardinality at most k, the goal is to find the maximum weight sub-collection of mutually disjoint sets. This is equivalent to k-CS-PIP where the constraint matrix S is 0-1 and the capacity c is all ones. Note that for $k = 2$ this is *maximum weight matching* which can be solved in polynomial time, and for $k = 3$ the problem becomes APX-hard [18]. After a long line of work [19,2,12,9], the best-known approximation ratio for this problem is $\frac{k+1}{2} + \epsilon$ obtained using local search techniques [9]. An improved bound of $\frac{k}{2} + \epsilon$ is also known [19] for the unweighted case, i.e., the weight vector $w = \mathbf{1}$. It is also known that the natural LP relaxation for this problem has integrality gap at least $k - 1 + 1/k$, and in particular this holds for the projective plane instance of order $k - 1$. Hazan *et al.* [18] showed that k-set packing is $\Omega(\frac{k}{\log k})$-hard to approximate.

Another special case of k-CS-PIP is the independent set problem in graphs with maximum degree at most k. This is equivalent to k-CS-PIP where the constraint matrix S is 0-1, capacity c is all ones, and each row is 2-sparse. This problem has an $O(k \log\log k/\log k)$-approximation [17], and is $\Omega(k/\log^2 k)$-hard to approximate [3], assuming the Unique Games Conjecture [20].

Shepherd and Vetta [26] studied the *demand matching* problem on graphs, which is k-CS-PIP with $k = 2$, with the further restriction that in each column the non-zero entries are equal, and that no two columns have non-zero entries in the same two rows. They gave an LP-based 3.264-approximation algorithm [26], and showed that the natural LP relaxation for this problem has integrality gap at least 3. They also showed the demand matching problem to be APX-hard even on bipartite graphs. For larger values of k, problems similar to *demand matching* have been studied under the name of column-restricted PIPs [21], which arise in the context of routing flow unsplittably (see also [6,7]). In particular, an $11.54k$-approximation algorithm was known [15] where (i) in each column all non-zero entries are equal, and (ii) the maximum entry in S is at most the minimum entry in c (this is also known as the no bottle-neck assumption); later, it was observed in [13] that even without the second of these conditions, one can obtain an $8k$ approximation. The literature on unsplittable flow is quite extensive; we refer the reader to [4,13] and references therein.

For the general k-CS-PIP, Pritchard [25] gave a $2^k k^2$-approximation algorithm, which was the first result with approximation ratio depending only on k. Pritchard's algorithm was based on solving an iterated LP relaxation, and then applying a randomized selection procedure. Independently, [14] and [11] showed that this final step could be derandomized, yielding an improved bound of $O(k^2)$. All these previous results crucially use the structural properties of basic feasible solutions of the LP relaxation. However, as stated above, our result is based on randomized rounding with alterations and does not use properties of basic solutions. This is crucial for the submodular maximization version of the problem, as a solution to the fractional relaxation there does not have these properties.

We remark that randomized rounding with alteration has also been used earlier by Srinivasan [28] in the context of PIPs. However, the focus of this paper is different from ours; in previous work [27], Srinivasan had bounded the integrality gap for PIPs by showing a randomized algorithm that obtained a "good" solution (one that satisfies all constraints) with positive — but perhaps exponentially small — probability. In [28], he proved that rounding followed by alteration leads to an efficient and parallelizable algorithm; the rounding gives a "solution" of good value in which most constraints are satisfied, and one can alter this solution to ensure that all constraints are satisfied. (We note that [27,28] also gave derandomized versions of these algorithms.)

Related issues have been considered in discrepancy theory, where the goal is to round a fractional solution to a k-column sparse linear program so that the capacity violation for any constraint is minimized. A celebrated result of Beck-Fiala [8] shows that the capacity violation is at most $O(k)$. A major open question in discrepancy theory is whether the above bound can be improved to $O(\sqrt{k})$, or even $O(k^{1-\epsilon})$ for some $\epsilon > 0$. While the result of [25] uses techniques similar to that of [8], a crucial difference in our problem is that no constraint can be violated at all.

There is a large body of work on constrained maximization of submodular functions; we only cite the relevant papers here. Calinescu *et al.* [10] introduced a continuous relaxation (called the *multi-linear extension* or extension-by-expectation) of submodular functions and subsequently Vondrák [29] gave an elegant $\frac{e}{e-1}$-approximation algorithm for solving this continuous relaxation over any "downward monotone" polytope $\mathcal{P}$, as long as there is a polynomial-time algorithm for optimizing linear functions over $\mathcal{P}$. We use this continuous relaxation in our algorithm for submodular maximization over k-sparse packing constraints. As noted earlier, k-sparse packing constraints generalize both k-partition matroids and k-dimensional knapsacks. Nemhauser *et al.* [24] gave a $(k+1)$-approximation for submodular maximization over the intersection of k partition matroids; when k is constant, Lee *et al.* [23] improved this to $k+\epsilon$. Kulik *et al.* [22] gave an $\left(\frac{e}{e-1}+\epsilon\right)$-approximation for submodular maximization over k-dimensional knapsacks when k is constant; if k is part of the input, the best known approximation bound is $O(k)$.

Problem Definition and Notation: Before we begin, we formally describe the k-CS-PIP problem and fix some notation. Let the items (i.e., columns) be indexed by $i \in [n]$ and the constraints (i.e., rows) be indexed by $j \in [m]$. We consider the following packing integer program.

$$\max\left\{\sum_{i=1}^{n} w_i x_i \;\middle|\; \sum_{i=1}^{n} s_{ij}\cdot x_i \le c_j,\ \forall j\in[m];\ \ x_i\in\{0,1\},\ \forall i\in[n]\right\}$$

We say that item i *participates* in constraint j if $s_{ij} > 0$. For each $i \in [n]$, let $N(i) := \{j \in [m] \mid s_{ij} > 0\}$ be the set of constraints that i participates in. In a k-column sparse PIP, we have $|N(i)| \leq k$ for each $i \in [n]$. The goal is to find the maximum weight subset of items such that all the constraints are satisfied.

We define the *slack* as $B := \min_{i\in[n],j\in[m]} c_j/s_{ij}$. By scaling the constraint matrix, we may assume that $c_j = 1$ for all $j \in [m]$. We also assume that $s_{ij} \leq 1$ for each i, j; otherwise, we can just fix $x_i = 0$. Finally, for each constraint j, we let $P(j)$ denote the set of items participating in this constraint. Note that $|P(j)|$ can be arbitrarily large.

Organization: In Section 2 we begin with the natural LP relaxation, and describe a simple $8k$-approximation algorithm. We then present a stronger relaxation, and sketch a proof of an $(e + o(1))k$-approximation. We also present the integrality gap of $2k - 1$ for this strengthened LP, implying that our result is almost tight. In Section 3, we describe the $O(k)$-approximation for k-column sparse packing problems over a submodular objective. Finally, in Section 4, we state the significantly better ratios that can be obtained for both linear and submodular objectives if the capacities of all constraints are large relative to the sizes; there are matching integrality gaps up to a constant factor.

2 The Algorithm for k-CS-PIP

Before presenting our algorithm, we describe a (seemingly correct) algorithm that does not quite work. Understanding why this easier algorithm fails gives useful insight into the design for the correct algorithm.

A Strawman Algorithm: Consider the following algorithm. Let x be some optimum solution to the natural LP relaxation of k-CS-PIP (i.e., dropping integrality). For each element $i \in [n]$, select it independently at random with probability $x_i/(2k)$. Let $\mathcal{S}$ be the chosen set of items. For any constraint $j \in [m]$, if it is violated, then discard all items in $\mathcal{S} \cap P(j)$, i.e., items $i \in \mathcal{S}$ for which $s_{ij} > 0$.

As the probabilities are scaled down by $2k$, by Markov's inequality any constraint j is violated with probability at most $1/(2k)$, and hence discards its items with at most this probability. By the k-sparse property, each element can be discarded by at most k constraints, and so by the union bound it is discarded with probability at most $k \cdot 1/(2k) = 1/2$. Since an element is chosen in $\mathcal{S}$ with probability $x_i/(2k)$, this implies that it lies in the overall solution with probability at least $x_i/(4k)$, implying that the proposed algorithm is a $4k$-approximation.

However, the above argument is not correct. Consider the following example. Suppose there is a single constraint (and so $k = 1$),

$$Mx_1 + x_2 + x_3 + x_4 + \ldots + x_M \leq M$$

where $M \gg 1$ is a large integer. Clearly, setting $x_i = 1/2$ for $i = 1, \ldots, M$ is a feasible solution. Now consider the execution of the strawman algorithm. Note that whenever item 1 is chosen in $\mathcal{S}$, it is very likely that some item other than

1 will also be chosen (since $M \gg 1$ and we pick each item independently with probability $x_i/(2k) = 1/4$); in this case, item 1 will be discarded. Thus the final solution will almost always *not contain* item 1, violating the claim that it lies in the final solution with probability at least $x_1/(4k) = 1/8$.

The key point is that we must consider the probability of an item being discarded by some constraint, *conditional* on it being chosen in the set $\mathcal{S}$ (for item 1 in the above example, this probability is close to one, not at most half). This is not a problem if either all item sizes are small (say $s_{ij} \le c_j/2$), or all item sizes are large (say $s_{ij} \approx c_j$). The algorithm we analyze shows that the difficult case is indeed when some constraints contain both large and small items, as in the example above.

2.1 A Simple Algorithm for k-CS-PIP

We use the obvious LP relaxation for k-CS-PIP (i.e., dropping the integrality condition) to obtain an $8k$-approximation algorithm. An item $i \in [n]$ is called *big* for constraint $j \in [m]$ iff $s_{ij} > \frac{1}{2}$, and *small* for constraint j iff $0 < s_{ij} \le \frac{1}{2}$. We first solve the LP relaxation to obtain an optimal fractional solution x, and then round to an integral solution as follows. With foresight, set $\alpha = 4$.

1. Sample each item $i \in [n]$ independently with probability $x_i/(\alpha k)$.
   Let $\mathcal{S}$ denote the set of chosen items. We call an item in $\mathcal{S}$ an $\mathcal{S}$-item.
2. For each item i, mark i (for deletion) if, for any constraint $j \in N(i)$, either:
   - $\mathcal{S}$ contains some *other* item $i' \in [n] \setminus \{i\}$ which is big for constraint j or
   - The sum of sizes of $\mathcal{S}$-items that are small for j exceeds 1. (i.e., the capacity).
3. Delete all marked items, and return $\mathcal{S}'$, the set of remaining items.

Analysis: We will show that this algorithm gives an $8k$-approximation.

Lemma 1. *Solution $\mathcal{S}'$ is feasible with probability one.*

Proof Sketch. Consider any fixed constraint $j \in [m]$. If there is some $i' \in \mathcal{S}'$ that is big for j, it will be the only item in $\mathcal{S}'$ that participates in constraint j. If all $\mathcal{S}'$-items participating in j are small, their total size is at most 1. □

We now prove the main technical result of this section.

Theorem 1. *For any item $i \in [n]$, the probability* $\Pr[i \in \mathcal{S}' \mid i \in \mathcal{S}] \ge 1 - \frac{2}{\alpha}$. *Equivalently, the probability that item i is deleted from $\mathcal{S}$ conditional on it being chosen in $\mathcal{S}$ is at most $2/\alpha$.*

Proof. For any item i and constraint $j \in N(i)$, let B_{ij} denote the event that i is marked for deletion from $\mathcal{S}$ because there is some other $\mathcal{S}$-item that is big for constraint j. Let G_j denote the event that the total size of $\mathcal{S}$-items that are small for constraint j exceeds 1. For any item $i \in [n]$ and constraint $j \in N(i)$, we will show that:

$$\Pr[B_{ij} \mid i \in \mathcal{S}] + \Pr[G_j \mid i \in \mathcal{S}] \le \frac{2}{\alpha k} \tag{1}$$

To see that (1) implies the theorem, for any item i, simply take the union bound over all $j \in N(i)$. Thus, the probability that i is deleted from $\mathcal{S}$ conditional on it being chosen in $\mathcal{S}$ is at most $2/\alpha$. Equivalently, $\Pr[i \in \mathcal{S}' \mid i \in \mathcal{S}] \geq 1 - 2/\alpha$.

We now prove (1) using the following intuition: The total extent to which the LP selects items that are *big* for any constraint cannot be more than 2 (each big item has size at least 1/2); therefore, B_{ij} is unlikely to occur since we scaled down probabilities by factor αk. Ignoring for a moment the conditioning on $i \in \mathcal{S}$, event G_j is also unlikely, by Markov's Inequality. But items are selected for $\mathcal{S}$ independently, so if i is big for constraint j, then its presence in $\mathcal{S}$ does not affect the event G_j at all. If i is small for constraint j, then *even if* $i \in \mathcal{S}$, the total size of $\mathcal{S}$-items is unlikely to exceed 1.

To prove (1) formally, let $B(j)$ denote the set of items that are big for constraint j, and $Y_j := \sum_{\ell \in B(j)} x_\ell$. By the LP constraint for j, it follows that $Y_j \leq 2$ (since each $\ell \in B(j)$ has size $s_{\ell j} > \frac{1}{2}$). Now by a union bound,

$$\Pr[B_{ij} \mid i \in \mathcal{S}] \leq \frac{1}{\alpha k} \sum_{\ell \in B(j) \setminus \{i\}} x_\ell \leq \frac{Y_j}{\alpha k} \leq \frac{2}{\alpha k}. \tag{2}$$

Now, let $G_{-i}(j)$ denote the set of items that are small for constraint j, *not counting* item i, even if it is small. Using the LP constraint j, we have:

$$\sum_{\ell \in G_{-i}(j)} s_{\ell j} \cdot x_l \leq 1 - \sum_{\ell \in B(j)} s_{\ell j} \cdot x_\ell \leq 1 - \frac{Y_j}{2}. \tag{3}$$

Since each item i' is chosen into $\mathcal{S}$ with probability $x_{i'}/(\alpha k)$, inequality (3) implies that the expected total size of $\mathcal{S}$-items in $G_{-i}(j)$ is at most $\frac{1}{\alpha k}(1 - Y_j/2)$. By Markov's inequality, the probability that the total size of these $\mathcal{S}$-items exceeds 1/2 is at most $\frac{2}{\alpha k}(1 - Y_j/2)$. Since items are chosen independently and $i \notin G_{-i}(j)$, we obtain this probability even conditioned on $i \in \mathcal{S}$.

Whether i is big or small for j, event G_j can occur only if the total size of $\mathcal{S}$-items in $G_{-i}(j)$ exceeds 1/2. Thus,

$$\Pr[G_j \mid i \in \mathcal{S}] \leq \frac{2}{\alpha k}\left(1 - \frac{Y_j}{2}\right) = \frac{2}{\alpha k} - \frac{Y_j}{\alpha k}$$

which, combined with inequality (2), yields (1).

Using the theorem above, we obtain the desired approximation:

Theorem 2. *There is a randomized $8k$-approximation algorithm for k-*CS-PIP.

Proof. From Lemma 1, our algorithm always outputs a feasible solution. To bound the objective value, recall that $\Pr[i \in \mathcal{S}] = \frac{x_i}{\alpha k}$ for all $i \in [n]$. Hence Theorem 1 implies that for all $i \in [n]$

$$\Pr[i \in \mathcal{S}'] \geq \Pr[i \in \mathcal{S}] \cdot \Pr[i \in \mathcal{S}' | i \in \mathcal{S}] \geq \frac{x_i}{\alpha k} \cdot \left(1 - \frac{2}{\alpha}\right).$$

Finally, using linearity of expectation and $\alpha = 4$, we obtain the theorem.

Remarks: We note that the analysis above only uses Markov's inequality conditioned on a single item being chosen in set $\mathcal{S}$. Thus a pairwise independent distribution suffices to choose the set $\mathcal{S}$, and hence the algorithm can be easily derandomized. More generally, one could consider k-CS-PIP with *arbitrary upper-bounds* on the variables: the above $8k$-approximation algorithm extends easily to this setting (details in the full version).

2.2 A Stronger LP, and Improved Approximation

We now present our strengthened LP and the $(ek + o(k))$-approximation algorithm for k-CS-PIP.

Stronger LP relaxation. Recall that entries are scaled so that all capacities are one. An item i is called *big* for constraint j iff $s_{ij} > 1/2$. For each constraint $j \in [m]$, let $B(j) = \{i \in [n] \mid s_{ij} > \frac{1}{2}\}$ denote the set of big items. Since no two items that are big for some constraint can be chosen in an integral solution, the inequality $\sum_{i \in B(j)} x_i \leq 1$ is valid for each $j \in [m]$. The strengthened LP relaxation that we consider is as follows.

$$\max \quad \sum_{i=1}^{n} w_i x_i \tag{4}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} s_{ij} \cdot x_i \leq c_j, \qquad \forall j \in [m] \tag{5}$$

$$\sum_{i \in B(j)} x_i \leq 1, \qquad \forall j \in [m]. \tag{6}$$

$$0 \leq x_i \leq 1, \qquad \forall i \in [n]. \tag{7}$$

The Algorithm: The algorithm obtains an optimal solution x to the LP relaxation (4-7), and rounds it to an integral solution $\mathcal{S}'$ as follows (parameter α will be set to 1 later).

1. Pick each item $i \in [n]$ independently with probability $x_i/(\alpha k)$, with $\alpha \geq 1$. Let $\mathcal{S}$ denote the set of chosen items.
2. For any item i and constraint $j \in N(i)$, let E_{ij} denote the event that the items $\{i' \in \mathcal{S} \mid s_{i'j} \geq s_{ij}\}$ have total size (in constraint j) exceeding one. Mark i for deletion if E_{ij} occurs for any $j \in N(i)$.
3. Return set $\mathcal{S}' \subseteq \mathcal{S}$ consisting of all items $i \in \mathcal{S}$ not marked for deletion.

Note the rule for deleting an item from $\mathcal{S}$. In particular, whether item i is deleted due to constraint j only depends on items that are at least as large as i in j.

Analysis: It is clear that $\mathcal{S}'$ is feasible with probability one. The improved approximation ratio comes from *four* different steps: First, we use the stronger LP relaxation. Second, the more careful alteration step does not discard items unnecessarily; the previous algorithm sometimes deleted items from $\mathcal{S}$ even when constraints were not violated. Third, in analyzing the probability that constraint

j causes item i to be deleted from $\mathcal{S}$, we further exploit discreteness of item sizes. And fourth, for each item i, we use the FKG inequality to bound the probability it is deleted instead of the weaker union bound over all constraints in $N(i)$.

The main lemma is the following, where we show that each item appears in $\mathcal{S}'$ with good probability.

Lemma 2. *For every item $i \in [n]$ and constraint $j \in N(i)$, we have* $\Pr[E_{ij} \mid i \in \mathcal{S}] \le \frac{1}{\alpha k}\left(1 + (\frac{2}{\alpha k})^{1/3}\right)$.

Proof Sketch. Let $\ell := (4\alpha k)^{1/3}$. We classify items in relation to constraints as:

– Item $i \in [n]$ is *big* for constraint $j \in [m]$ if $s_{ij} > \frac{1}{2}$.
– Item $i \in [n]$ is *medium* for constraint $j \in [m]$ if $\frac{1}{\ell} \le s_{ij} \le \frac{1}{2}$.
– Item $i \in [n]$ is *tiny* for constraint $j \in [m]$ if $s_{ij} < \frac{1}{\ell}$.

We separately bound $\Pr[E_{ij} | i \in \mathcal{S}]$ when item i is big, medium, and tiny.

Claim. For any $i \in [n]$ and $j \in [m]$:

1. If item i is big for constraint j, $\Pr[E_{ij} \mid i \in \mathcal{S}] \le \frac{1}{\alpha k}$.
2. If item i is medium for constraint j, $\Pr[E_{ij} \mid i \in \mathcal{S}] \le \frac{1}{\alpha k}\left(1 + \frac{\ell^2}{2\alpha k}\right)$.
3. If item i is tiny for constraint j, $\Pr[E_{ij} \mid i \in \mathcal{S}] \le \frac{1}{\alpha k}\left(1 + \frac{2}{\ell}\right)$.

In case 1, E_{ij} occurs only if some other big item for constraint j is chosen in $\mathcal{S}$; the new constraints (6) of the strengthened LP bound this probability. In case 2, E_{ij} can occur only if some big item or at least two medium items other than i are selected for $\mathcal{S}$; we argue that the latter probability is much smaller than $1/\alpha k$. In case 3, E_{ij} can occur only if the total size (in constraint j) of items in $\mathcal{S} \setminus \{i\}$ is greater than $1 - \frac{1}{\ell}$; Markov's inequality gives the desired result.

Thus, for any item i and constraint $j \in N(i)$, $\Pr[E_{ij} \mid i \in \mathcal{S}] \le \frac{1}{\alpha k}\max\{(1 + \frac{2}{\ell}), (1 + \frac{\ell^2}{2\alpha k})\}$. From the choice of $\ell = (4\alpha k)^{1/3}$, which makes the probability in parts 2 and 3 of the claim equal, we obtain the lemma. □

We now prove the main result of this section.

Theorem 3. *For each $i \in [n]$,* $\Pr[i \in \mathcal{S}' \mid i \in \mathcal{S}] \ge \left(1 - \frac{1}{\alpha k}\left(1 + (\frac{2}{\alpha k})^{1/3}\right)\right)^k$.

Proof. For any item i and constraint $j \in N(i)$, the conditional event $(\neg E_{ij} \mid i \in \mathcal{S})$ is a *decreasing* function over the choice of items in set $[n] \setminus \{i\}$. Thus, by the FKG inequality [1], for any fixed item $i \in [n]$, the probability that no event $(E_{ij} \mid i \in \mathcal{S})$ occurs is:

$$\Pr\left[\bigwedge_{j \in N(i)} \neg E_{ij} \;\middle|\; i \in \mathcal{S}\right] \ge \prod_{j \in N(i)} \Pr[\neg E_{ij} \mid i \in \mathcal{S}]$$

From Lemma 2, $\Pr[\neg E_{ij} \mid i \in \mathcal{S}] \ge 1 - \frac{1}{\alpha k}\left(1 + (\frac{2}{\alpha k})^{1/3}\right)$. As each item is in at most k constraints, we obtain the theorem.

Now, by setting $\alpha = 1$,[1] we have $\Pr[i \in \mathcal{S}] = 1/k$, and $\Pr[i \in \mathcal{S}' \mid i \in \mathcal{S}] \geq \frac{1}{e+o(1)}$, which immediately implies:

Theorem 4. *There is a randomized $(ek + o(k))$-approximation algorithm for k-*CS-PIP*.*

Remark: We note that this algorithm can be derandomized using conditional expectation and pessimistic estimators, since we can exactly compute estimates of the relevant probabilities. Also, using ideas from [28], the algorithm can be implemented in RNC. We defer details to the full version.

Integrality Gap of LP (4-7). Consider the instance on $n = m = 2k - 1$ items and constraints defined as follows. We view the indices $[n] = \{0, 1, \cdots, n-1\}$ as integers modulo n. The weights $w_i = 1$ for all $i \in [n]$. The sizes are:

$$s_{ij} := \begin{cases} 1 & \text{if } i = j \\ \epsilon & \text{if } j \in \{i+1, \cdots, i+k-1 \pmod n\} \\ 0 & \text{otherwise} \end{cases}, \qquad \forall i, j \in [n].$$

where $\epsilon > 0$ is arbitrarily small, in particular $\epsilon \ll \frac{1}{nk}$.

Observe that setting $x_i = 1 - k\epsilon$ for all $i \in [n]$ is a feasible fractional solution to the strengthened LP (4-7); each constraint has only one big item and so the new constraint (6) is satisfied. Thus the optimal LP value is at least $(1 - k\epsilon) \cdot n \approx n = 2k - 1$. On the other hand, it is easy to see that the optimal integral solution can only choose one item and hence has value 1. Thus the integrality gap of the LP we consider is at least $2k - 1$, for every $k \geq 1$.

3 Submodular Objective Functions

We now consider the more general case when the objective we seek to maximize is an arbitrary non-negative *monotone submodular function* $f : 2^{[n]} \to \mathbb{R}_+$. The problem we consider is:

$$\max \left\{ f(T) \mid \sum_{i \in T} s_{ij} \leq c_j, \ \forall j \in [m]; \ T \subseteq [n] \right\} \tag{8}$$

As is standard when dealing with submodular functions, we only assume *value-oracle* access to the function: i.e., the algorithm can query any subset $T \subseteq [n]$, and it obtains the function value $f(T)$ in constant time. Again, we let k denote the column-sparseness of the underlying constraint matrix. In this section we obtain an $O(k)$-approximation algorithm for Problem (8). The algorithm is similar to that for k-CS-PIP (where the objective was linear):

1. We first solve (approximately) a suitable continuous relaxation of (8). This step follows directly from the algorithm of Vondrák [29].

[1] Note that this is optimal only asymptotically; in the case of $k = 2$, for instance, it is better to choose $\alpha \approx 2.8$.

2. Then, using the fractional solution, we perform the randomized rounding with alteration described in Section 2. Although the algorithm is the same as for linear functions, the analysis requires considerably more work. In the process, we also establish a new property of submodular functions that generalizes *fractional subadditivity* [16].

Solving the Continuous Relaxation. The *extension-by-expectation* (also called the *multi-linear* extension) of a submodular function f is a continuous function $F : [0,1]^n \to \mathbb{R}_+$ defined as follows:

$$F(x) := \sum_{T \subseteq [n]} \Pi_{i \in T}\ x_i \cdot \Pi_{j \notin T}\ (1 - x_j) \cdot f(T)$$

Note that $F(x) = f(x)$ for $x \in \{0,1\}^n$ and hence F is an extension of f. Even though F is a non-linear function, using the continuous greedy algorithm of Vondrák [29], we can obtain an $\left(1 - \frac{1}{e}\right)$-approximation algorithm to the following *fractional relaxation* of (8):

$$\max \left\{ F(x) \;\middle|\; \sum_{i=1}^{n} s_{ij} \cdot x_i \leq c_j,\ \forall j \in [m];\ 0 \leq x_i \leq 1,\ \forall i \in [n] \right\} \tag{9}$$

In order to apply the algorithm from [29], one needs to solve in polynomial time the problem of maximizing a *linear* objective over the constraints $\{\sum_{i=1}^{n} s_{ij} \cdot x_i \leq c_j,\ \forall j \in [m];\ 0 \leq x_i \leq 1,\ \forall i \in [n]\}$. This is indeed possible since it is a linear program on n variables and m constraints.

The Rounding Algorithm and Analysis. The rounding algorithm is identical to that for k-CS-PIP. Let x denote any feasible solution to Problem (9). We apply the rounding algorithm from the previous section, to first obtain (possibly infeasible) solution $\mathcal{S} \subseteq [n]$ and then *feasible integral solution* $\mathcal{S}' \subseteq [n]$.

However, the analysis approach in Theorem 3 does not work. The problem is that even though S (which is chosen by random sampling) has good expected profit, i.e., $E[f(\mathcal{S})] = \Omega(\frac{1}{k})F(x)$, it may happen that the alteration step used to obtain $\mathcal{S}'$ from $\mathcal{S}$ may end up throwing away essentially all the profit. This was not an issue for linear objective functions since our alteration procedure guarantees that $\Pr[i \in \mathcal{S}' | i \in \mathcal{S}] = \Omega(1)$ for each $i \in [n]$; if f is linear, this implies $E[f(\mathcal{S})] = \Omega(1)\, E[f(\mathcal{S}')]$. However, this property is not enough for general monotone submodular functions. Consider the following:

Example: Let set $\mathcal{S} \subseteq [n]$ be drawn from the following distribution:

– With probability $1/2n$, $\mathcal{S} = [n]$.
– For each $i \in [n]$, $\mathcal{S} = \{i\}$ with probability $1/2n$.
– With probability $1/2 - 1/2n$, $\mathcal{S} = \emptyset$.

Define $\mathcal{S}' = \mathcal{S}$ if $\mathcal{S} = [n]$, and $\mathcal{S}' = \emptyset$ otherwise. For each $i \in [n]$, we have $\Pr[i \in \mathcal{S}' \mid i \in \mathcal{S}] = 1/2 = \Omega(1)$. However, consider the profit with respect to the "coverage" submodular function f, where $f(T) = 1$ if $T \neq \emptyset$ and $= 0$ otherwise. We have $E[f(\mathcal{S})] = 1/2 + 1/2n$, but $E[f(\mathcal{S}')]$ is only $1/2n \ll E[f(\mathcal{S})]$.

Remark: Note that if $\mathcal{S}'$ itself was chosen randomly from $\mathcal{S}$ such that $\Pr[i \in \mathcal{S}' | \mathcal{S} = T] = \Omega(1)$ for *every* $T \subseteq [n]$ *and* $i \in T$, then we would be done by Feige's Subadditivity Lemma [16]. Unfortunately, this is too much to hope for. In our rounding procedure, for any particular choice of $\mathcal{S}$, set $\mathcal{S}'$ is a fixed subset of $\mathcal{S}$; and there could be (bad) sets $\mathcal{S}$, where after the alteration step we end up with sets $\mathcal{S}'$ such that $|\mathcal{S}'| \ll |\mathcal{S}|$.

However, it turns out that we can use the following two *additional* properties of our algorithm to argue that $\mathcal{S}'$ has reasonable profit. First, the sets $\mathcal{S}$ we construct are drawn from a product distribution on the items. Second, our alteration procedure has the following 'monotonicity' property: Suppose $i \in T_1 \subseteq T_2 \subseteq [n]$, and $i \in \mathcal{S}'$ when $\mathcal{S} = T_2$. Then we are guaranteed that $i \in \mathcal{S}'$ when $\mathcal{S} = T_1$. (That is, if $\mathcal{S}$ contains additional items, it is more likely that i will be discarded by some constraint it participates in.) The example above does not satisfy either of these properties. Corollary 1 shows that these properties suffice. Roughly speaking, the intuition is that since f is submodular, the marginal contribution of item i to $\mathcal{S}$ is largest when $\mathcal{S}$ is "small"; this is also the case when i is most likely to be retained for $\mathcal{S}'$. That is, for every $i \in [n]$, both $\Pr[i \in \mathcal{S}' \mid i \in \mathcal{S}]$ and the marginal contribution of i to $f(\mathcal{S})$ are *decreasing* functions of $\mathcal{S}$. We prove (see [5]) the following generalization of Feige's Subadditivity Lemma.

Theorem 5. *Let $[n]$ denote a groundset, $x \in [0,1]^n$, and for each $B \subseteq [n]$ define $p(B) = \Pi_{i \in B} x_i \cdot \Pi_{j \notin B}(1 - x_j)$. Associated with each $B \subseteq [n]$, there is an arbitrary distribution over subsets of B, where each set $A \subseteq B$ has probability $q_B(A)$; so $\sum_{A \subseteq B} q_B(A) = 1$ for all $B \subseteq [n]$. That is, we choose B from a product distribution, and then retain a subset A of B by applying a randomized alteration. Suppose that the system satisfies the following conditions.*

Marginal Property:

$$\forall i \in [n], \quad \sum_{B \subseteq [n]} p(B) \sum_{A \subseteq B : i \in A} q_B(A) \;\geq\; \beta \cdot \sum_{B \subseteq [n] : i \in B} p(B). \tag{10}$$

Monotonicity: *For any two subsets $B \subseteq B' \subseteq [n]$ we have,*

$$\forall i \in B, \quad \sum_{A \subseteq B : i \in A} q_B(A) \;\geq\; \sum_{A' \subseteq B' : i \in A'} q_{B'}(A') \tag{11}$$

Then, for any monotone submodular function f,

$$\sum_{B \subseteq [n]} p(B) \sum_{A \subseteq B} q_B(A) \cdot f(A) \;\geq\; \beta \cdot \sum_{B \subseteq [n]} p(B) \cdot f(B). \tag{12}$$

Corollary 1. *Let $\mathcal{S}$ be a random set drawn from a product distribution on $[n]$. Let $\mathcal{S}'$ be another random set where for each choice of $\mathcal{S}$, set $\mathcal{S}'$ is an arbitrary subset of $\mathcal{S}$. Suppose that for each $i \in [n]$ the following hold.*

- $\Pr_{\mathcal{S}}[i \in \mathcal{S}' \mid i \in \mathcal{S}] \geq \beta$, *and*
- *For all $T_1 \subseteq T_2$ with $T_1 \ni i$, if $i \in \mathcal{S}'$ when $\mathcal{S} = T_2$ then $i \in \mathcal{S}'$ when $\mathcal{S} = T_1$.*

Then $E[f(\mathcal{S}')] \geq \beta E[f(\mathcal{S})]$.

We are now ready to prove the performance guarantee of our algorithm. Observe that our rounding algorithm satisfies the hypothesis of Corollary 1 with $\beta = \frac{1}{e+o(1)}$, when parameter $\alpha = 1$. Moreover, one can show that $E[f(\mathcal{S})] \geq F(x)/(\alpha k)$. Thus, $E[f(\mathcal{S}')] \geq \frac{1}{e+o(1)} E[f(\mathcal{S})] \geq \frac{1}{ek+o(k)} \cdot F(x)$. Combined with the fact that x is an $\frac{e}{e-1}$-approximate solution to the continuous relaxation (9), we have proved our main result:

Theorem 6. *There is a randomized algorithm for maximizing any monotone submodular function over k-column sparse packing constraints achieving approximation ratio $\frac{e^2}{e-1}k + o(k)$.*

4 k-CS-PIP Algorithm for Large B

We can obtain substantially better approximation guarantees for k-CS-PIP when the capacities are large relative to the sizes. Recall the definition of the slack parameter B. We consider the k-CS-PIP problem as a function of both k and B, and obtain improved approximation ratios given in the following.

Theorem 7. *There is a $\left(4e \cdot \left((e + o(1)) \lfloor B \rfloor k\right)^{1/\lfloor B \rfloor}\right)$-approximation algorithm for k-CS-PIP, and a $\left(\frac{4e^2}{e-1} \cdot \left((e + o(1)) \lfloor B \rfloor k\right)^{1/\lfloor B \rfloor}\right)$-approximation for maximizing monotone submodular functions over k-column sparse packing constraints.*

The algorithms that obtain these approximation ratios are similar to those of the preceding sections, but additional care is required in the analysis; as B is large, one can now use a smaller scaling factor in the randomized rounding step while bounding the probability that an element is deleted in the alteration step. We also show that the natural LP relaxation for k-CS-PIP has an $\Omega(k^{1/\lfloor B \rfloor})$ integrality gap for every $B \geq 1$.

Acknowledgements. NK thanks Chandra Chekuri and Alina Ene for detailed discussions on k-CS-PIP. We also thank Deeparnab Chakarabarty and David Pritchard for discussions and sharing a copy of [11]. We thank Jan Vondrák and Chandra Chekuri for pointing out an error in the original proof of Theorem 6, which prompted us to prove Theorem 5. Our thanks also to the IPCO referees for their helpful suggestions.

References

1. Alon, N., Spencer, J.: The Probabilistic Method, 3rd edn. Wiley-Interscience, New York (2008)
2. Arkin, E.M., Hassin, R.: On Local Search for Weighted k-Set Packing. In: European Symposium on Algorithms, pp. 13–22 (1997)
3. Austrin, P., Khot, S., Safra, S.: Inapproximability of Vertex Cover and Independent Set in Bounded Degree Graphs. In: Comp. Complexity Conference (2009)
4. Bansal, N., Friggstad, Z., Khandekar, R., Salavatipour, M.R.: A logarithmic approximation for unsplittable flow on line graphs. In: SODA (2009)
5. Bansal, N., Korula, N., Nagarajan, V., Srinivasan, A.: On k-Column Sparse Packing Programs (full version), arXiv (2010)

6. Baveja, A., Srinivasan, A.: Approximating Low-Congestion Routing and Column-Restricted Packing Problems. Information Proc. Letters (74), 19–25 (2000)
7. Baveja, A., Srinivasan, A.: Approximation Algorithms for Disjoint Paths and Related Routing and Packing Problems. Math. of Oper. Res. (25), 255–280 (2000)
8. Beck, J., Fiala, T.: "Integer making" theorems. Discrete Appl. Math. 3, 1–8 (1981)
9. Berman, P.: A $d/2$ approximation for maximum weight independent set in d-claw free graphs. Nordic Journal of Computing 7(3), 178–184 (2000)
10. Calinescu, G., Chekuri, C., Pál, M., Vondrák, J.: Maximizing a monotone submodular function under a matroid constraint. In: Fischetti, M., Williamson, D.P. (eds.) IPCO 2007. LNCS, vol. 4513, pp. 182–196. Springer, Heidelberg (2007)
11. Chakrabarty, D., Pritchard, D.: Personal Communication (2009)
12. Chandra, B., Halldórsson, M.: Greedy Local Improvement and Weighted Packing Approximation. In: SODA (1999)
13. 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 and RANDOM 2009. LNCS, vol. 5687, pp. 42–55. Springer, Heidelberg (2009)
14. Chekuri, C., Ene, A., Korula, N.: Personal Communication (2009)
15. Chekuri, C., Mydlarz, M., Shepherd, B.: Multicommodity Demand Flow in a Tree and Packing Integer Programs. ACM Trans. on Algorithms 3(3) (2007)
16. Feige, U.: On maximizing welfare when utility functions are subadditive. In: STOC, pp. 41–50 (2006)
17. Halperin, E.: Improved Approximation Algorithms for the Vertex Cover Problem in Graphs and Hypergraphs. SIAM J. Comput. 31(5), 1608–1623 (2002)
18. Hazan, E., Safra, S., Schwartz, O.: On the complexity of approximating k-set packing. Computational Complexity 15(1), 20–39 (2003)
19. Hurkens, A.J., Schrijver, A.: On the Size of Systems of Sets Every t of Which Have an SDR, with an Application to the Worst-Case Ratio of Heuristics for Packing Problems. SIAM J. Discrete Math. 2(1), 68–72 (1989)
20. Khot, S.: On the power of unique 2-prover 1-round games. In: STOC, pp. 767–775 (2002)
21. Kolliopoulos, S., Stein, C.: Approximating Disjoint-Path Problems using Packing Integer Programs. Mathematical Programming A (99), 63–87 (2004)
22. Kulik, A., Shachnai, H., Tamir, T.: Maximizing submodular functions subject to multiple linear constraints. In: SODA (2009)
23. Lee, J., Mirrokni, V., Nagarajan, V., Sviridenko, M.: Non-monotone submodular maximization under matroid and knapsack constraints. In: STOC, pp. 323–332 (2009)
24. Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functions II. Math. Prog. Study 8, 73–87 (1978)
25. Pritchard, D.: Approximability of Sparse Integer Programs. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 83–94. Springer, Heidelberg (2009)
26. Shepherd, B., Vetta, A.: The demand matching problem. Mathematics of Operations Research 32, 563–578 (2007)
27. Srinivasan, A.: Improved Approximation Guarantees for Packing and Covering Integer Programs. SIAM J. Comput. 29(2), 648–670 (1999)
28. Srinivasan, A.: New approaches to covering and packing problems. In: SODA, pp. 567–576 (2001)
29. Vondrák, J.: Optimal approximation for the submodular welfare problem in the value oracle model. In: STOC, pp. 67–74 (2008)
30. Zuckerman, D.: Linear Degree Extractors and the Inapproximability of Max Clique and Chromatic Number. Theory of Computing 3(1), 103–128 (2007)

Hypergraphic LP Relaxations for Steiner Trees

Deeparnab Chakrabarty, Jochen Könemann, and David Pritchard

University of Waterloo, Waterloo, Ontario N2L 3G1, Canada*

Abstract. We investigate hypergraphic LP relaxations for the Steiner tree problem, primarily the partition LP relaxation introduced by Könemann et al. [Math. Programming, 2009]. Specifically, we are interested in proving upper bounds on the integrality gap of this LP, and studying its relation to other linear relaxations. Our results are the following.

Structural results: We extend the technique of uncrossing, usually applied to families of sets, to families of partitions. As a consequence we show that any basic feasible solution to the partition LP formulation has sparse support. Although the number of variables could be exponential, the number of positive variables is at most the number of terminals.

Relations with other relaxations: We show the equivalence of the partition LP relaxation with other known hypergraphic relaxations. We also show that these hypergraphic relaxations are equivalent to the well studied bidirected cut relaxation, if the instance is quasibipartite.

Integrality gap upper bounds: We show an upper bound of $\sqrt{3} \doteq 1.729$ on the integrality gap of these hypergraph relaxations in general graphs. In the special case of uniformly quasibipartite instances, we show an improved upper bound of $73/60 \doteq 1.216$. By our equivalence theorem, the latter result implies an improved upper bound for the bidirected cut relaxation as well.

1 Introduction

In the *Steiner tree* problem, we are given an undirected graph $G = (V, E)$, non-negative costs c_e for all edges $e \in E$, and a set of *terminal* vertices $R \subseteq V$. The goal is to find a minimum-cost tree T spanning R, and possibly some *Steiner vertices* from $V \setminus R$. We can assume that the graph is complete and that the costs induce a metric. The problem takes a central place in the theory of combinatorial optimization and has numerous practical applications. Since the Steiner tree problem is NP-hard[1] we are interested in approximation algorithms for it. The best published approximation algorithm for the Steiner tree problem is due to Robins and Zelikovsky [20], which for any fixed $\epsilon > 0$, achieves a performance ratio of $1 + \frac{\ln 3}{2} + \epsilon \doteq 1.55$ in polynomial time; an improvement is currently in press [2], see also Remark 1.

* Supported by NSERC grant no. 288340 and by an Early Research Award.

[1] Chlebík and Chlebíková show that no $(96/95-\epsilon)$-approximation algorithm can exist for any positive ϵ unless P=NP [5].

In this paper, we study linear programming (LP) relaxations for the Steiner tree problem, and their properties. Numerous such formulations are known (e.g., see [7,11,16,17,24,25]), and their study has led to impressive running time improvements for integer programming based methods. Despite the significant body of work in this area, none of the known relaxations is known to exhibit an *integrality gap* provably smaller than 2. The integrality gap of a relaxation is the maximum ratio of the cost of integral and fractional optima, over all instances. It is commonly regarded as a measure of strength of a formulation. One of the contributions of this paper are improved bounds on the integrality gap for a number of Steiner tree LP relaxations.

A Steiner tree relaxation of particular interest is the *bidirected cut relaxation* [7,25] (precise definitions will follow in Section 1.2). This relaxation has a flow formulation using $O(|E||R|)$ variables and constraints, which is much more compact than the other relaxations we study. Also, it is also widely believed to have an integrality gap significantly smaller than 2 (e.g., see [3,19,23]). The largest lower bound on the integrality gap known is 8/7 (by Martin Skutella, reported in [15]), and Chakrabarty et al. [3] prove an upper bound of 4/3 in so called *quasi-bipartite* instances (where Steiner vertices form an independent set).

Another class of formulations are the so called *hypergraphic* LP relaxations for the Steiner tree problem. These relaxations are inspired by the observation that the minimum Steiner tree problem can be encoded as a minimum cost hyper-spanning tree (see Section 1.2) of a certain hypergraph on the terminals. They are known to be stronger than the bidirected cut relaxation [18], and it is therefore natural to try to use them to get better approximation algorithms, by drawing on the large corpus of known LP techniques. In this paper, we focus on one hypergraphic LP in particular: the *partition* LP of Könemann et al. [15].

1.1 Our Results and Techniques

There are three classes of results in this paper: structural results, equivalence results, and integrality gap upper bounds.

Structural results, Section 2: We extend the powerful technique of *uncrossing*, traditionally applied to families of sets, to families of *partitions*. Set uncrossing has been very successful in obtaining exact and approximate algorithms for a variety of problems (for instance, [9,14,21]). Using partition uncrossing, we show that any basic feasible solution to the partition LP has at most $(|R|-1)$ positive variables (even though it can have an exponentially large number of variables and constraints).

Equivalence results, Section 3: In addition to the partition LP, two other hypergraphic LPs have been studied before: one based on *subtour elimination* due to Warme [24], and a *directed hypergraph relaxation* of Polzin and Vahdati Daneshmand [18]; these two are known to be equivalent [18]. We prove that in fact *all three hypergraphic relaxations are equivalent* (that is, they have the same objective value for any Steiner tree instance).

We also show that, on *quasibipartite instances*, the hypergraphic and the bidirected cut LP relaxations are equivalent. This result is surprising since we are aware of no qualitative similarity to suggest why the two relaxations should be equivalent. We believe a better understanding of the bidirected cut relaxation is important because it is central in theory *and* practical for implementation.

Improved integrality gap upper bounds, Section 4: For *uniformly quasibipartite instances* (quasibipartite instances where for each Steiner vertex, all incident edges have the same cost), we show that the integrality gap of the hypergraphic LP relaxations is upper bounded by $73/60 \doteq 1.216$. Our proof uses the approximation algorithm of Gröpl et al. [13] which achieves the same ratio with respect to the (integral) optimum. We show, via a simple dual fitting argument, that this ratio is also valid with respect to the LP value. To the best of our knowledge this is the only nontrivial class of instances where the best currently known approximation ratio and integrality gap upper bound are the same.

For general graphs, we give simple upper bounds of $2\sqrt{2} - 1 \doteq 1.83$ and $\sqrt{3} \doteq 1.729$ on the integrality gap of the hypergraph relaxation. Call a graph *gainless* if the minimum *spanning* tree of the terminals is the optimal Steiner tree. To obtain these integrality gap upper bounds, we use the following key property of the hypergraphic relaxation which was implicit in [15]: on gainless instances (instances where the optimum terminal spanning tree is the optimal Steiner tree), the LP value equals the minimum spanning tree and the integrality gap is 1. Such a theorem was known for quasibipartite instances and the bidirected cut relaxation (implicitly in [19], explicitly in [3]); we extend techniques of [3] to obtain improved integrality gaps on all instances.

Remark 1. The recent independent work of Byrka et al. [2], which gives an improved approximation for Steiner trees in general graphs, also shows an integrality gap bound of 1.55 on the hypergraphic directed cut LP. This is stronger than our integrality gap bounds and was obtained prior to the completion of our paper; yet we include our bounds because they are obtained using fairly different methods which might be of independent interest in certain settings.

The proof in [2] can be easily modified to show an integrality gap upper bound of 1.28 in quasibipartite instances. Then using our equivalence result, we get an integrality gap upper bound of 1.28 for the bidirected cut relaxation on quasibipartite instances, improving the previous best of 4/3.

1.2 Bidirected Cut and Hypergraphic Relaxations

The Bidirected Cut Relaxation. The first bidirected LP was given by Edmonds [7] as an exact formulation for the spanning tree problem. Wong [25] later extended this to obtain the bidirected cut relaxation for the Steiner tree problem, and gave a dual ascent heuristic based on the relaxation. For this relaxation, introduce two arcs (u, v) and (v, u) for each edge $uv \in E$, and let both of their costs be c_{uv}. Fix an arbitrary terminal $r \in R$ as the root. Call a subset $U \subseteq V$ *valid* if it contains a terminal but not the root, and let valid(V) be the family of all valid sets. Clearly, the in-tree rooted at r (the directed tree with all vertices but the root having out-degree exactly 1) of a Steiner tree T must have at least one arc with tail in U and head outside U, for all valid U. This leads to the bidirected cut relaxation ($\mathcal{B}$) (shown in Figure 1 with dual) which has a variable for each arc $a \in A$, and a constraint for every valid set U. Here and later, $\delta^{\text{out}}(U)$ denotes the set of arcs in A whose tail is in U and whose head lies in $V \setminus U$. When there are no Steiner vertices, Edmonds' work [7] implies this relaxation is exact.

$$\min \sum_{a \in A} c_a x_a : \quad x \in \mathbf{R}^A_{\geq 0} \qquad (\mathcal{B})$$
$$\sum_{a \in \delta^{\text{out}}(U)} x_a \geq 1, \quad \forall U \in \text{valid}(V)$$

$$\max \sum_{U} z_U : \quad z \in \mathbf{R}^{\text{valid}(V)}_{\geq 0} \qquad (\mathcal{B}_D)$$
$$\sum_{U : a \in \delta^{\text{out}}(U)} z_U \leq c_a, \quad \forall a \in A$$

Fig. 1. The bidirected cut relaxation ($\mathcal{B}$) and its dual ($\mathcal{B}_D$)

Goemans & Myung [11] made significant progress in understanding the LP, by showing that the bidirected cut LP has the same value independent of which terminal is chosen as the root, and by showing that a whole "catalogue" of very different-looking LPs also has the same value; later Goemans [10] showed that if the graph is series-parallel, the relaxation is exact. Rajagopalan and Vazirani [19] were the first to show a non-trivial integrality gap upper bound of 3/2 on quasibipartite graphs; this was subsequently improved to 4/3 by Chakrabarty et al. [3], who gave another alternate formulation for ($\mathcal{B}$).

Hypergraphic Relaxations. Given a Steiner tree T, a *full component* of T is a maximal subtree of T all of whose leaves are terminals and all of whose internal nodes are Steiner nodes. The edge set of any Steiner tree can be partitioned in a *unique* way into full components by splitting at internal terminals; see Figure 2 for an example.

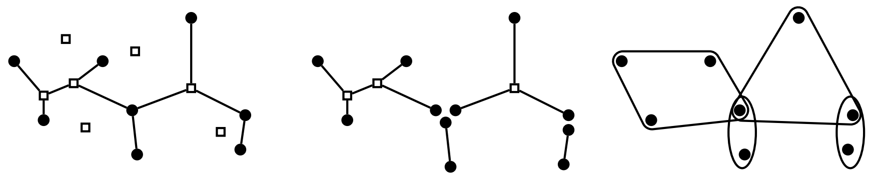


Fig. 2. Black nodes are terminals and white nodes are Steiner nodes. Left: a Steiner tree for this instance. Middle: the Steiner tree's edges are partitioned into full components; there are four full components. Right: the hyperedges corresponding to these full components.

Let $\mathcal{K}$ be the set of all nonempty subsets of terminals (*hyperedges*). We associate with each $K \in \mathcal{K}$ a fixed full component spanning the terminals in K, and let C_K be its cost[2]. The problem of finding a minimum-cost Steiner tree

[2] We choose the minimum cost full component if there are many. If there is no full component spanning K, we let C_K be infinity. Such a minimum cost component can be found in polynomial time, if $|K|$ is a constant.

spanning R now reduces to that of finding a minimum-cost hyper-spanning tree in the hypergraph $(R, \mathcal{K})$.

Spanning trees in (normal) graphs are well understood and there are many different exact LP relaxations for this problem. These exact LP relaxations for spanning trees in graphs inspire the *hypergraphic relaxations* for the Steiner tree problem. Such relaxations have a variable x_K for every[3] $K \in \mathcal{K}$, and the different relaxations are based on the constraints used to capture a hyper-spanning tree, just as constraints on edges are used to capture a spanning tree in a graph.

The oldest hypergraphic LP relaxation is the subtour LP introduced by Warme [24] which is inspired by Edmonds' subtour elimination LP relaxation [8] for the spanning tree polytope. This LP relaxation uses the fact that there are no hyper-cycles in a hyper-spanning tree, and that it is spanning. More formally, let $\rho(X) := \max(0, |X| - 1)$ be the *rank* of a set X of vertices. Then a sub-hypergraph $(R, \mathcal{K}')$ is a hyper-spanning tree iff $\sum_{K \in \mathcal{K}'} \rho(K) = \rho(R)$ and $\sum_{K \in \mathcal{K}'} \rho(K \cap S) \le \rho(S)$ for every subset S of R. The corresponding LP relaxation, denoted below as ($\mathcal{S}$), is called the *subtour elimination* LP relaxation.

$$\min \Big\{ \sum_{K \in \mathcal{K}} C_K x_K : \ x \in \mathbf{R}^{\mathcal{K}}_{\ge 0}, \ \sum_{K \in \mathcal{K}} x_K \rho(K) = \rho(R), \qquad (\mathcal{S})$$
$$\sum_{K \in \mathcal{K}} x_K \rho(K \cap S) \le \rho(S), \ \forall S \subset R \Big\}$$

Warme showed that if the maximum hyperedge size r is bounded by a constant, the LP can be solved in polynomial time.

The next hypergraphic LP introduced for Steiner tree was a directed hypergraph formulation ($\mathcal{D}$), introduced by Polzin and Vahdati Daneshmand [18], and inspired by the bidirected cut relaxation. Given a full component K and a terminal $i \in K$, let K^i denote the arborescence obtained by directing all the edges of K towards i. Think of this as directing the hyperedge K towards i to get the directed hyperedge K^i. Vertex i is called the *head* of K^i while the terminals in $K \setminus i$ are the *tails* of K. The cost of each directed hyperedge K^i is the cost of the corresponding undirected hyperedge K. In the directed hypergraph formulation, there is a variable x_{K^i} for every directed hyperedge K^i. As in the bidirected cut relaxation, there is a vertex $r \in R$ which is a root, and as described above, a subset $U \subseteq R$ of terminals is valid if it does not contain the root but contains at least one vertex in R. We let $\Delta^{\text{out}}(U)$ be the set of directed full components coming out of U, that is all K^i such that $U \cap K \neq \varnothing$ but $i \notin U$. Let $\overrightarrow{\mathcal{K}}$ be the set of all directed hyperedges. We show the directed hypergraph relaxation and its dual in Figure 3.

Polzin & Vahdati Daneshmand [18] showed that OPT ($\mathcal{D}$) = OPT ($\mathcal{S}$). Moreover they observed that this directed hypergraphic relaxation strengthens the bidirected cut relaxation.

[3] Observe that there could be exponentially many hyperedges. This computational issue is circumvented by considering hyperedges of size at most r, for some constant r. By a result of Borchers and Du [1], this leads to only a $(1 + \Theta(1/\log r))$ factor increase in the optimal Steiner tree cost.

$$\min\Big\{\sum_{K\in\mathcal{K},i\in K} C_K x_{K^i} : x\in\mathbf{R}_{\geq 0}^{\overrightarrow{\mathcal{K}}} \quad (\mathcal{D}) \qquad \sum_{K^i\in\Delta^{\text{out}}(U)} x_{K^i}\geq 1, \quad \forall \text{ valid } U\subseteq R\Big\}$$

$$\max\Big\{\sum_U z_U : \quad z\in\mathbf{R}_{\geq 0}^{\text{valid}(R)} \quad (\mathcal{D}_D) \qquad \sum_{U:K\cap U\neq\varnothing, i\notin U} z_U\leq C_K, \quad \forall K\in\mathcal{K}, i\in K\Big\}$$

Fig. 3. The directed hypergraph relaxation ($\mathcal{D}$) and its dual ($\mathcal{D}_D$)

Lemma 1 ([18]). *For any instance,* OPT ($\mathcal{D}$) $\geq$ OPT ($\mathcal{B}$)*. There are instances for which this inequality is strict.*

Könemann et al. [15], inspired by the work of Chopra [6], described a partition-based relaxation which captures that given any partition of the terminals, any hyper-spanning tree must have sufficiently many "cross hyperedges". More formally, a partition, π, is a collection of pairwise disjoint nonempty terminal sets $(\pi_1,\ldots,\pi_q)$ whose union equals R. The number of *parts* q of π is referred to as the partition's *rank* and denoted as $r(\pi)$. Let Π_R be the set of all partitions of R. Given a partition $\pi=\{\pi_1,\ldots,\pi_q\}$, define the *rank contribution* $\mathtt{rc}_K^\pi$ of hyperedge $K\in\mathcal{K}$ for π as the rank reduction of π obtained by merging the parts of π that are touched by K; i.e., $\mathtt{rc}_K^\pi := |\{i : K\cap\pi_i\neq\varnothing\}|-1$. Then a hyper-spanning tree $(R,\mathcal{K}')$ must satisfy $\sum_{K\in\mathcal{K}'}\mathtt{rc}_K^\pi\geq r(\pi)-1$. The partition based LP of [15] and its dual are given in Figure 4.

$$\min\Big\{\sum_{K\in\mathcal{K}} C_K x_K : \quad x\in\mathbf{R}_{\geq 0}^{\mathcal{K}} \quad (\mathcal{P}) \qquad \sum_{K\in\mathcal{K}} x_K\mathtt{rc}_K^\pi\geq r(\pi)-1, \quad \forall\pi\in\Pi_R\Big\}$$

$$\max\Big\{\sum_\pi (r(\pi)-1)y_\pi : \quad y\in\mathbf{R}_{\geq 0}^{\Pi_R} \quad (\mathcal{P}_D) \qquad \sum_{\pi\in\Pi_R} y_\pi\mathtt{rc}_K^\pi\leq C_K, \quad \forall K\in\mathcal{K}\Big\}$$

Fig. 4. The unbounded partition relaxation ($\mathcal{P}$) and its dual ($\mathcal{P}_D$)

The feasible region of ($\mathcal{P}$) is *unbounded*, since if x is a feasible solution for ($\mathcal{P}$) then so is any $x'\geq x$. We obtain a *bounded* partition LP relaxation, denoted by ($\mathcal{P}'$) and shown below, by adding a valid equality constraint to the LP.

$$\min\Big\{\sum_{K\in\mathcal{K}} C_K x_K : x\in(\mathcal{P}), \sum_{K\in\mathcal{K}} x_K(|K|-1)=|R|-1\Big\} \qquad (\mathcal{P}')$$

2 Uncrossing Partitions

In this section we are interested in *uncrossing* a minimal set of *tight partitions* that uniquely define a basic feasible solution to ($\mathcal{P}$). We start with a few preliminaries necessary to state our result formally.

2.1 Preliminaries

We introduce some needed well-known properties of partitions that arise in combinatorial lattice theory [22].

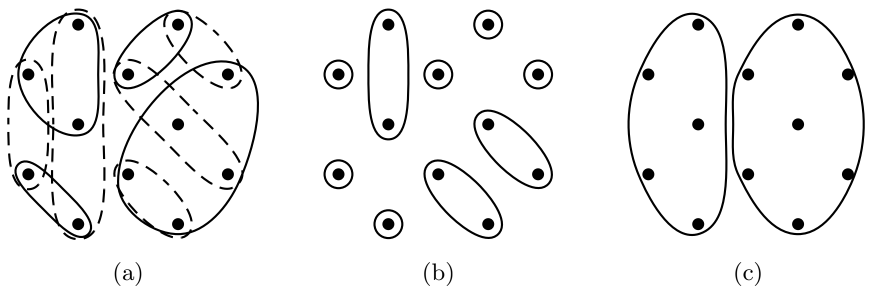


Fig. 5. Illustrations of some partitions. The black dots are the terminal set R. (a): two partitions; neither refines the other. (b): the meet of the partitions from (a). (c): the join of the partitions from (a).

Definition 1. *We say that a partition π'* refines *another partition π if each part of π' is contained in some part of π. We also say π* coarsens *π'. Two partitions* cross *if neither refines the other. A family of partitions forms a* chain *if no pair of them cross. Equivalently, a chain is any family $\pi^1, \pi^2, \dots, \pi^t$ such that π^i refines π^{i-1} for each $1 < i \le t$.*

The family Π_R of all partitions of R forms a *lattice* with a *meet operator* $\wedge : \Pi_R^2 \to \Pi_R$ and a *join operator* $\vee : \Pi_R^2 \to \Pi_R$. The meet $\pi \wedge \pi'$ is the coarsest partition that refines both π and π', and the join $\pi \vee \pi'$ is the most refined partition that coarsens both π and π'. See Figure 5 for an illustration.

Definition 2 (Meet of partitions). *Let the parts of π be $\pi_1, \dots, \pi_t$ and let the parts of π' be $\pi'_1, \dots, \pi'_u$. Then the parts of the meet $\pi \wedge \pi'$ are the nonempty intersections of parts of π with parts of π',*

$$\pi \wedge \pi' = \{\pi_i \cap \pi'_j \mid 1 \le i \le t, 1 \le j \le u \text{ and } \pi_i \cap \pi'_j \ne \varnothing\}.$$

Given a graph G and a partition π of $V(G)$, we say that G *induces* π if the parts of π are the vertex sets of the connected components of G.

Definition 3 (Join of partitions). *Let (R, E) be a graph that induces π, and let (R, E') be a graph that induces π'. Then the graph $(R, E \cup E')$ induces $\pi \vee \pi'$.*

Given a feasible solution x to ($\mathcal{P}$), a partition π is *tight* if $\sum_{K \in \mathcal{K}} x_K \mathtt{rc}_K^{\pi} = r(\pi) - 1$. Let $\mathtt{tight}(x)$ be the set of all tight partitions. We are interested in *uncrossing* this set of partitions. More precisely, we wish to find a cross-free set of partitions (chain) which uniquely defines x. One way would be to prove the following.

Property 1. *If two crossing partitions π and π' are in $\mathtt{tight}(x)$, then so are $\pi \wedge \pi'$ and $\pi \vee \pi'$.*

This type of property is already well-used [9,14,21] for sets (with meets and joins replaced by unions and intersections respectively), and the standard approach is the following. The typical proof considers the constraints in ($\mathcal{P}$) corresponding to π and π' and uses the "supermodularity" of the RHS and the "submodularity" of the coefficients in the LHS. In particular, if the following is true,

$$\forall \pi, \pi' : \; r(\pi \vee \pi') + r(\pi \wedge \pi') \quad \geq \quad r(\pi) + r(\pi') \tag{1}$$

$$\forall K, \pi, \pi' : \; \mathtt{rc}_K^{\pi} + \mathtt{rc}_K^{\pi'} \quad \geq \quad \mathtt{rc}_K^{\pi \vee \pi'} + \mathtt{rc}_K^{\pi \wedge \pi'} \tag{2}$$

then Property 1 can be proved easily by writing a string of inequalities.[4]

Inequality (1) is indeed true (see, for example, [22]), but unfortunately inequality (2) is not true in general, as the following example shows.

Example 1. Let $R = \{1, 2, 3, 4\}$, $\pi = \{\{1, 2\}, \{3, 4\}\}$ and $\pi' = \{\{1, 3\}, \{2, 4\}\}$. Let K denote the full component $\{1, 2, 3, 4\}$. Then $\mathtt{rc}_K^{\pi} + \mathtt{rc}_K^{\pi'} = 1 + 1 < 0 + 3 = \mathtt{rc}_K^{\pi \vee \pi'} + \mathtt{rc}_K^{\pi \wedge \pi'}$.

Nevertheless, Property 1 is true; its correct proof is given in the full version of this paper [4] and depends on a simple though subtle extension of the usual approach. The crux of the insight needed to fix the approach is not to consider *pairs* of constraints in ($\mathcal{P}$), but rather multi-sets which may contain more than two inequalities. Using this uncrossing result, we can prove the following theorem (details are given in [4]). Here, we let $\underline{\pi}$ denote $\{R\}$, the unique partition with (minimal) rank 1; later we use $\overline{\pi}$ to denote $\{\{r\} \mid r \in R\}$, the unique partition with (maximal) rank $|R|$.

Theorem 1. *Let x^* be a basic feasible solution of ($\mathcal{P}$), and let $\mathcal{C}$ be an inclusion-wise maximal chain in $\mathtt{tight}(x^*) \backslash \underline{\pi}$. Then x^* is uniquely defined by*

$$\sum_{K \in \mathcal{K}} \mathtt{rc}_K^{\pi} x_K^* = r(\pi) - 1 \quad \forall \pi \in \mathcal{C}. \tag{3}$$

Any chain of distinct partitions of R that does not contain $\underline{\pi}$ has size at most $|R| - 1$, and this is an upper bound on the rank of the system in (3). Elementary linear programming theory immediately yields the following corollary.

Corollary 1. *Any basic solution x^* of ($\mathcal{P}$) has at most $|R| - 1$ non-zero coordinates.*

3 Equivalence of Formulations

In this section we describe our equivalence results. A summary of the known and new results is given in Figure 6.

For lack of space, we present only sketches for our main equivalence results in this extended abstract, and refer the reader to [4] for details.

[4] In this hypothetical scenario we get $r(\pi) + r(\pi') - 2 = \sum_K x_K(\mathtt{rc}_K^{\pi} + \mathtt{rc}_K^{\pi'}) \geq \sum_K x_K(\mathtt{rc}_K^{\pi \wedge \pi'} + \mathtt{rc}_K^{\pi \vee \pi'}) \geq r(\pi \wedge \pi') + r(\pi \vee \pi') - 2 \geq r(\pi) + r(\pi') - 2$; thus the inequalities hold with equality, and the middle one shows $\pi \wedge \pi'$ and $\pi \vee \pi'$ are tight.

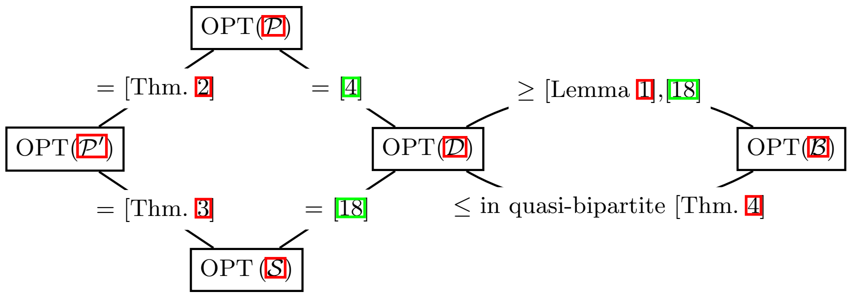


Fig. 6. Summary of relations among various LP relaxations

Theorem 2. *The LPs* ($\mathcal{P}'$) *and* ($\mathcal{P}$) *have the same optimal value.*

Proof sketch. To show this, it suffices to find an optimum solution of ($\mathcal{P}$) which satisfies the equality in ($\mathcal{P}'$); i.e., we want to find a solution for which the maximal-rank partition $\overline{\pi}$ is tight. We pick the optimum solution to ($\mathcal{P}$) which minimizes the sum $\sum_{K \in \mathcal{K}} x_K|K|$. Using Property 1, we show that either $\overline{\pi}$ is tight or there is a *shrinking* operation which decreases $\sum_{K \in \mathcal{K}} x_K|K|$ without increasing the cost. Since the latter is impossible, the theorem is proved.

Theorem 3. *The feasible regions of* ($\mathcal{P}'$) *and* ($\mathcal{S}$) *are the same.*

Proof sketch. We show that the inequalities defining ($\mathcal{P}'$) are valid for ($\mathcal{S}$), and vice-versa. Note that both have the same equality and non-negativity constraints. To show that the partition inequality of ($\mathcal{P}'$) for π holds for any $x \in$ ($\mathcal{S}$), we use the subtour inequalities in ($\mathcal{S}$) for every part of π. For the other direction, given any subset $S \subseteq R$, we invoke the inequality in ($\mathcal{P}'$) for the partition $\pi := \{\{S\}$ as one part and the remaining terminals as singletons$\}$.

Theorem 4. *On quasibipartite Steiner tree instances,* OPT ($\mathcal{B}$) $\geq$ OPT ($\mathcal{D}$).

Proof sketch. We look at the duals of the two LPs and we show OPT ($\mathcal{B}_D$) $\geq$ OPT ($\mathcal{D}_D$) in quasibipartite instances. Recall that the support of a solution to ($\mathcal{D}_D$) is the family of sets with positive z_U. A family of sets is called *laminar* if for any two of its sets A, B we have $A \subseteq B, B \subseteq A$, or $A \cap B = \varnothing$. The following fact follows along the standard line of "set uncrossing" argumentation.

Lemma 2. *There is an optimal solution to* ($\mathcal{D}_D$) *with laminar support.*

Given the above result, we may now assume that we have a solution z to ($\mathcal{D}_D$) whose support is laminar. The heart of the proof of Theorem 4 is to show that z can be converted into a feasible solution to ($\mathcal{B}_D$) of the same value.

Comparing ($\mathcal{D}_D$) and ($\mathcal{B}_D$) one first notes that the former has a variable for every valid subset of the terminals, while the latter assigns values to all valid subsets of the entire vertex set. We say that an edge uv is *satisfied* for a candidate solution z, if both a) $\sum_{U: u \in U, v \notin U} z_U \leq c_{uv}$ and b) $\sum_{U: v \in U, u \notin U} z_U \leq c_{uv}$ hold; z is then feasible for ($\mathcal{B}_D$) if *all* edges are satisfied.

Let z be a feasible solution to ($\mathcal{D}_D$). One easily verifies that all terminal-terminal edges are satisfied. On the other hand, terminal-Steiner edges may initially not be satisfied; e.g., consider the Steiner vertex v and its neighbours depicted in Figure 7 below. Initially, none of the sets in z's support contains v, and the load on the edges incident to v is quite *skewed*: the left-hand side of condition a) above may be large, while the left-hand side of condition b) is initially 0.

To construct a valid solution for ($\mathcal{B}_D$), we therefore *lift* the initial value z_S of each terminal subset S to supersets of S, by adding Steiner vertices. The lifting procedure processes each Steiner vertex v one at a time; when processing v, we change z by moving dual from some sets U to $U \cup \{v\}$. Such a dual transfer decreases the left-hand side of condition a) for edge uv, and increases the (initially 0) left-hand sides of condition b) for edges connecting v to neighbours other than v.

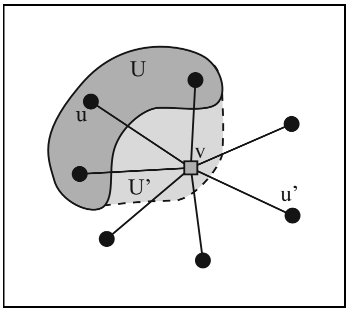


Fig. 7. Lifting variable z_U

We are able to show that there is a way of carefully lifting duals around v that ensures that all edges incident to v become satisfied. The definition of our procedure will ensure that these edges remain satisfied for the rest of the lifting procedure. Since there are no Steiner-Steiner edges, all edges will be satisfied once all Steiner vertices are processed.

Throughout the lifting procedure, we will maintain that z remains unchanged, when projected to the terminals. The main consequence of this is that the objective value $\sum_{U \subseteq V} z_U$ remains constant throughout, and the objective value of z in ($\mathcal{B}_D$) is not affected by the lifting. This yields Theorem 4.

4 Improved Integrality Gap Upper Bounds

In this extended abstract, we show the improved bound of 73/60 for uniformly quasibipartite graphs, and due to space restrictions, we only show the weaker $(2\sqrt{2} - 1) \doteq 1.828$ upper bound on general graphs.

4.1 Uniformly Quasibipartite Instances

Uniformly quasibipartite instances of the Steiner tree problem are quasibipartite graphs where the cost of edges incident on a Steiner vertex are the same. They were first studied by Gröpl et al. [13], who gave a 73/60 factor approximation algorithm. We start by describing the algorithm of Gröpl et al. [13] in terms of full components. A collection $\mathcal{K}'$ of full components is acyclic if there is no list of $t > 1$ distinct terminals and hyperedges in $\mathcal{K}'$ of the form $r_1 \in K_1 \ni r_2 \in K_2 \cdots \ni r_t \in K_t \ni r_1$ — i.e. there are no *hypercycles*.

```
Procedure RatioGreedy
1: Initialize the set of acyclic components $\mathcal{L}$ to $\varnothing$.
2: Let $L^*$ be a minimizer of $\frac{C_L}{|L|-1}$ over all full components $L$ such that $|L| \geq 2$
   and $L \cup \mathcal{L}$ is acyclic.
3: Add $L^*$ to $\mathcal{L}$.
4: Continue until $(R, \mathcal{L})$ is a hyper-spanning tree and return $\mathcal{L}$.
```

Theorem 5. *On a uniformly quasibipartite instance* RatioGreedy *returns a Steiner tree of cost at most* $\frac{73}{60}$ OPT ($\mathcal{P}$).

Proof sketch. Let t denote the number of iterations and $\mathcal{L} := \{L_1, \ldots, L_t\}$ be the ordered sequence of full components obtained. We now define a dual solution y to ($\mathcal{P}_D$). Let $\pi(i)$ denote the partition induced by the connected components of $\{L_1, \ldots, L_i\}$. Let $\theta(i)$ denote $C_{L_i}/(|L_i| - 1)$ and note that θ is nondecreasing. Define $\theta(0) = 0$ for convenience. We define a dual solution y with

$$y_{\pi(i)} = \theta(i+1) - \theta(i)$$

for $0 \leq i < t$, and all other coordinates of y set to zero. It is straightforward to verify that the objective value $\sum_i y_{\pi(i)}(r(\pi(i)) - 1)$ of y in ($\mathcal{P}_D$) equals $C(\mathcal{L})$.

The key is to show that for all $K \in \mathcal{K}$,

$$\sum_i y_{\pi(i)} \mathtt{rc}_K^{\pi(i)} \leq (|K| - 1 + H(|K| - 1))/|K| \cdot C_K, \tag{4}$$

where H denotes the harmonic series; this is obtained by using the greedy nature of the algorithm and the fact that, in uniformly quasi-bipartite graphs, $C_{K'} \leq C_K \frac{|K'|}{|K|}$ whenever $K' \subset K$. Now, $(|K| - 1 + H(|K| - 1))/|K|$ is always at most $\frac{73}{60}$. Thus (4) implies that $\frac{60}{73} \cdot y$ is a feasible dual solution, which completes the proof.

4.2 General Graphs

For conciseness we let a "graph" be a triple $G = (V, E, R)$ where $R \subset V$ are G's terminals. In the following, we let $\mathtt{mtst}(G; c)$ denote the minimum *terminal spanning tree*, i.e. the minimum spanning tree of the terminal-induced subgraph $G[R]$ under edge-costs $c : E \to \mathbf{R}$. We will abuse notation and let $\mathtt{mtst}(G; c)$ mean both the tree and its cost under c.

When contracting an edge uv in a graph, the new merged node resulting from contraction is defined to be a terminal iff at least one of u or v was a terminal; this is natural since a Steiner tree in the new graph is a minimal set of edges which, together with uv, connects all terminals in the old graph. Our algorithm performs contraction, which may introduce parallel edges, but one may delete all but the cheapest edge from each parallel class without affecting the analysis.

Our algorithm proceeds in stages. In each stage we apply the operation $G \mapsto G/K$ which denotes contracting all edges in some full component K. To describe and analyze the algorithm we introduce some notation. For a minimum terminal spanning tree $T = \mathtt{mtst}(G; c)$ define $\mathtt{drop}_T(K; c) := c(T) - \mathtt{mtst}(G/K; c)$. We also define $\mathtt{gain}_T(K; c) := \mathtt{drop}_T(K) - c(K)$, where $c(K)$ is the cost of full component K. A tree T is called *gainless* if for every full component K we have $\mathtt{gain}_T(K; c) \leq 0$. The following useful fact is implicit in [15] (see also [4]).

Theorem 6 (Implicit in [15]). *If* $\mathtt{mtst}(G; c)$ *is gainless, then* OPT ($\mathcal{P}$) *equals the cost of* $\mathtt{mtst}(G; c)$.

We now give the algorithm and its analysis, which uses a reduced cost trick introduced by Chakrabarty et al.[3].

```
Procedure Reduced One-Pass Heuristic
1: Define costs $c'_e$ by $c'_e := c_e/\sqrt{2}$ for all terminal-terminal edges $e$, and $c'_e = c_e$
   for all other edges. Let $G_1 := G$, $T_i := \mathtt{mtst}(G_i; c')$, and $i := 1$.
2: The algorithm considers the full components in any order. When we exam-
   ine a full component $K$, if $\mathtt{gain}_{T_i}(K; c') > 0$, let $K_i := K$, $G_{i+1} := G_i/K_i$,
   $T_{i+1} := \mathtt{mtst}(G_{i+1}; c')$, and $i := i + 1$.
3: Let $f$ be the final value of $i$. Return the tree $T_{alg} := T_f \cup \bigcup_{i=1}^{f-1} K_i$.
```

Note that the full components are scanned in *any* order and they are not examined a priori. Hence the algorithm works just as well if the full components arrive "online," which might be useful for some applications.

Theorem 7. $c(T_{alg}) \leq (2\sqrt{2} - 1)\,\mathrm{OPT}\,(\mathcal{P})$.

Proof. First we claim that $\mathtt{gain}_{T_f}(K; c') \leq 0$ for all K. To see this there are two cases. If $K = K_i$ for some i, then we immediately see that $\mathtt{drop}_{T_j}(K) = 0$ for all $j > i$ so $\mathtt{gain}_{T_f}(K) = -c(K) \leq 0$. Otherwise (if for all i, $K \neq K_i$) K had nonpositive gain when examined by the algorithm; and the well-known *contraction lemma* (e.g., see [12, §1.5]) immediately implies that $\mathtt{gain}_{T_i}(K)$ is nonincreasing in i, so $\mathtt{gain}_{T_f}(K) \leq 0$.

By Theorem 6, $c'(T_f)$ equals the value of ($\mathcal{P}$) on the graph G_f with costs c'. Since $c' \leq c$, and since at each step we only contract terminals, the value of this optimum must be at most OPT ($\mathcal{P}$). Using the fact that $c(T_f) = \sqrt{2}c'(T_f)$, we get

$$c(T_f) = \sqrt{2}c'(T_f) \leq \sqrt{2}\,\mathrm{OPT}\,(\mathcal{P}) \tag{5}$$

Furthermore, for every i we have $\mathtt{gain}_{T_i}(K_i; c') > 0$, that is, $\mathtt{drop}_{T_i}(K_i; c') > c'(K) = c(K)$. The equality follows since K contains no terminal-terminal edges. However, $\mathtt{drop}_{T_i}(K_i; c') = \frac{1}{\sqrt{2}}\mathtt{drop}_{T_i}(K_i; c)$ because all edges of T_i are terminal-terminal. Thus, we get for every $i = 1$ to f, $\mathtt{drop}_{T_i}(K_i; c) > \sqrt{2} \cdot c(K_i)$.

Since $\mathtt{drop}_{T_i}(K_i; c) := \mathtt{mtst}(G_i; c) - \mathtt{mtst}(G_{i+1}; c)$, we have

$$\sum_{i=1}^{f-1} \mathtt{drop}_{T_i}(K_i; c) = \mathtt{mtst}(G; c) - c(T_f).$$

Thus, we have

$$\sum_{i=1}^{f-1} c(K_i) \le \frac{1}{\sqrt{2}} \sum_{i=1}^{f} \texttt{drop}_{T_i}(K_i; c) = \frac{1}{\sqrt{2}}(\texttt{mtst}(G; c) - c(T_f))$$
$$\le \frac{1}{\sqrt{2}}(2\,\mathrm{OPT}\,(\mathcal{P}) - c(T_f))$$

where we use the fact that $\texttt{mtst}(G, c)$ is at most twice OPT ($\mathcal{P}$)[5]. Therefore

$$c(T_{alg}) = c(T_f) + \sum_{i=1}^{f-1} c(K_i) \le \Big(1 - \frac{1}{\sqrt{2}}\Big) c(T_f) + \sqrt{2}\,\mathrm{OPT}\,(\mathcal{P}).$$

Finally, using $c(T_f) \le \sqrt{2}\,\mathrm{OPT}\,(\mathcal{P})$ from (5), the proof of Theorem 7 is complete.

References

1. Borchers, A., Du, D.: The k-Steiner ratio in graphs. SIAM J. Comput. 26(3), 857–869 (1997)
2. Byrka, J., Grandoni, F., Rothvoß, T., Sanità, L.: An improved LP-based approximation for Steiner tree. In: Proc. 42nd STOC (to appear 2010)
3. Chakrabarty, D., Devanur, N.R., Vazirani, V.V.: New geometry-inspired relaxations and algorithms for the metric Steiner tree problem. In: Lodi, A., Panconesi, A., Rinaldi, G. (eds.) IPCO 2008. LNCS, vol. 5035, pp. 344–358. Springer, Heidelberg (2008)
4. Chakrabarty, D., Könemann, J., Pritchard, D.: Hypergraphic LP relaxations for Steiner trees. Technical Report 0910.0281, arXiv (2009)
5. Chlebík, M., Chlebíková, J.: Approximation hardness of the Steiner tree problem on graphs. In: Penttonen, M., Schmidt, E.M. (eds.) SWAT 2002. LNCS, vol. 2368, pp. 170–179. Springer, Heidelberg (2002)
6. Chopra, S.: On the spanning tree polyhedron. Operations Research Letters 8, 25–29 (1989)
7. Edmonds, J.: Optimum branchings. Journal of Research of the National Bureau of Standards B 71B, 233–240 (1967)
8. Edmonds, J.: Matroids and the greedy algorithm. Math. Programming 1, 127–136 (1971)
9. Edmonds, J., Giles, R.: A min-max relation for submodular functions on graphs. Annals of Discrete Mathematics 1, 185–204 (1977)
10. Goemans, M.X.: The Steiner tree polytope and related polyhedra. Math. Program. 63(2), 157–182 (1994)
11. Goemans, M.X., Myung, Y.: A catalog of Steiner tree formulations. Networks 23, 19–28 (1993)
12. Gröpl, C., Hougardy, S., Nierhoff, T., Prömel, H.J.: Approximation algorithms for the Steiner tree problem in graphs. In: Cheng, X., Du, D. (eds.) Steiner trees in industries, pp. 235–279. Kluwer Academic Publishers, Norvell (2001)

[5] This follows using standard arguments, and can be seen, for instance, by applying Theorem 6 to the cost-function with all terminal-terminal costs divided by 2, and using short-cutting.

13. Gröpl, C., Hougardy, S., Nierhoff, T., Prömel, H.J.: Steiner trees in uniformly quasi-bipartite graphs. Inform. Process. Lett. 83(4), 195–200 (2002); Preliminary version appeared as a Technical Report at TU Berlin (2001)
14. Jain, K.: A factor 2 approximation algorithm for the generalized Steiner network problem. Combinatorica 21(1), 39–60 (2001); Preliminary version appeared in Proc. 39th FOCS, pp. 448–457 (1998)
15. Könemann, J., Pritchard, D., Tan, K.: A partition-based relaxation for Steiner trees. Math. Programming (2009) (in press)
16. Polzin, T.: Algorithms for the Steiner Problem in Networks. PhD thesis, Universität des Saarlandes (February 2003)
17. Polzin, T., Vahdati Daneshmand, S.: A comparison of Steiner tree relaxations. Discrete Applied Mathematics 112(1-3), 241–261 (2001); Preliminary version appeared at COS 1998 (1998)
18. Polzin, T., Vahdati Daneshmand, S.: On Steiner trees and minimum spanning trees in hypergraphs. Oper. Res. Lett. 31(1), 12–20 (2003)
19. Rajagopalan, S., Vazirani, V.V.: On the bidirected cut relaxation for the metric Steiner tree problem. In: Proceedings of ACM-SIAM Symposium on Discrete Algorithms, pp. 742–751 (1999)
20. Robins, G., Zelikovsky, A.: Tighter bounds for graph Steiner tree approximation. SIAM J. Discrete Math. 19(1), 122–134 (2005); Preliminary version appeared as Improved Steiner tree approximation in graphs at SODA 2000 (2000)
21. Singh, M., Lau, L.C.: Approximating minimum bounded degree spanning trees to within one of optimal. In: Proc. 39th STOC, pp. 661–670 (2007)
22. Stanley, R.P.: Enumerative Combinatorics, vol. 1. Wadsworth & Brooks/Cole (1986)
23. Vazirani, V.: Recent results on approximating the Steiner tree problem and its generalizations. Theoret. Comput. Sci. 235(1), 205–216 (2000)
24. Warme, D.: Spanning Trees in Hypergraphs with Applications to Steiner Trees. PhD thesis, University of Virginia (1998)
25. Wong, R.T.: A dual ascent approach for Steiner tree problems on a directed graph. Math. Programming 28, 271–287 (1984)

Efficient Deterministic Algorithms for Finding a Minimum Cycle Basis in Undirected Graphs

Edoardo Amaldi[1], Claudio Iuliano[1], and Romeo Rizzi[2]

[1] Dipartimento di Elettronica e Informazione, Politecnico di Milano, Italy
{amaldi,iuliano}@elet.polimi.it

[2] Dipartimento di Matematica e Informatica, Università degli Studi di Udine, Italy
rrizzi@dimi.uniud.it

Abstract. We consider the problem of, given an undirected graph G with a nonnegative weight on each edge, finding a basis of the cycle space of G of minimum total weight, where the total weight of a basis is the sum of the weights of its cycles. Minimum cycle bases are of interest in a variety of fields. In [13] Horton proposed a first polynomial-time algorithm where a minimum cycle basis is extracted from a polynomial-size subset of candidate cycles in $O(m^3n)$ by using Gaussian elimination. In a different approach, due to de Pina [7] and refined in [15], the cycles of a minimum cycle basis are determined sequentially in $O(m^2n+mn^2\log n)$. A more sophisticated hybrid algorithm proposed in [18] has the best worst-case complexity of $O(m^2n/\log n + mn^2)$.

In this work we revisit Horton's and de Pina's approaches and we propose a simple hybrid algorithm which improves the worst-case complexity to $O(m^2n/\log n)$. We also present a very efficient related algorithm that relies on an adaptive independence test *à la* de Pina. Computational results on a wide set of instances show that the latter algorithm outperforms the previous algorithms by one or two order of magnitude on medium-size instances and allows to solve instances with up to 3000 vertices in a reasonable time.

1 Introduction

Let $G=(V,E)$ be an undirected graph with $n=|V|$ vertices and $m=|E|$ edges. Assume w.l.o.g. that G is simple, that is, without loops and multiple edges. An *elementary cycle* is a connected subset of edges such that all incident vertices have degree 2. A *cycle* C is a (possibly empty) subset of edges such that every vertex of V is incident to an even number of edges in C. Cycles can be viewed as the (possibly empty) union of edge-disjoint elementary cycles. Each cycle C can be represented by an edge incidence vector in $\{0,1\}^m$ where a component is equal to 1 precisely when $e \in C$. The composition of two cycles C_1 and C_2, denoted by $C_1 \oplus C_2$, is defined as the symmetric difference between the subsets of edges, i.e., $(C_1 \cup C_2) \setminus (C_1 \cap C_2)$, or equivalently as the sum *modulo 2* of their edge incidence vectors. For any undirected graph G, the edge incidence vectors of all the cycles, including the null cycle, form a vector space over GF(2), called the *cycle space*. If G is connected, the dimension of this space is $\nu = m - n + 1$. Since a cycle basis of G is the union of the cycle bases of the connected components of G, we assume w.l.o.g. that G is connected. A maximal set of linearly independent cycles is called a *cycle basis*.

We consider the following combinatorial optimization problem known as the *minimum cycle basis* problem.

> Min CB: Given an undirected connected graph G with a nonnegative weight w_e assigned to each edge $e \in E$, find a cycle basis $\mathcal{C}$ of minimum total weight, i.e., which minimizes $w(\mathcal{C}) = \sum_{i=1}^{\nu} w(C_i)$, where $w(C_i) = \sum_{e \in C_i} w_e$.

Cycle bases with small total weight are of interest in a variety of fields including electrical networks [4], periodic event scheduling [7], chemistry and biochemistry [10]. For instance, in the test of electrical circuits the problem arises when one wishes to check as fast as possible that Kirchhoff's law is satisfied along all its loops. Variants of Min CB where the cycle bases are restricted to have a certain structure (e.g. to be weakly fundamental [20] and strictly fundamental [8], integral or totally unimodular [16]) also arise in practice. The reader is referred to the recent survey [14].

Early references to Min CB go back to the Sixties (see e.g. [21,22] in Russian) and the first polynomial-time algorithm was proposed by Horton only in 1987 [13]. A different approach was presented in de Pina's Ph.D. thesis [7] and improved to $O(m^2 n + mn^2 \log n)$ in [15]. A hybrid algorithm in $O(m^2 n^2)$ was proposed in [17]. Currently, the best deterministic algorithm is the one described in [18] by Mehlhorn and Michail, with an $O(m^2 n / \log n + mn^2)$ worst-case complexity. In joint work with Jurkiewicz and Mehlhorn [1] we have recently presented the best randomized algorithms for undirected and directed graphs, and improved the deterministic algorithm for planar graphs in [12].

In this paper we are concerned with deterministic algorithms for Min CB. Most of the previous work aims at reducing the worst-case complexity and little attention has been devoted so far to evaluate the practical performance of these algorithms. The only detailed computational work we are aware of is [17].

We revisit Horton's and de Pina's approaches and propose a hybrid algorithm which improves the best worst-case complexity to $O(m^2 n / \log n)$. This is achieved by restricting attention to the so-called isometric cycles, which were mentioned in [13] but whose power was first exploited by us in [1], and by using a bit packing technique appropriately combined with the divide-and-conquer framework described in [15]. We also present a very efficient algorithm that focuses on isometric cycles and is based on an adaptive independence test *à la* de Pina, which outperforms in practice all previous algorithms on a set of benchmark instances. In particular, it is faster by one or two order of magnitude on medium-size instances and it allows to solve instances with up to 3000 vertices in a reasonable time.

2 Previous Work

Before presenting our improved algorithms, we need to summarize the main algorithms in the literature.

Horton's polynomial-time algorithm [13] proceeds as follows. First, all-pairs shortest (minimum weight) paths p_{uv}, for all $u, v \in V$, are computed in $O(nm + n^2 \log n)$, assuring uniqueness by a lexicographic ordering on the vertices [12,1,14]. For any vertex x, let T_x denote the shortest path tree rooted at x. The key observation is that a $O(nm)$ size set of candidate cycles $\mathcal{H}$, that we refer to as Horton set, is guaranteed to contain a minimum cycle basis. For each possible pair of vertex x and edge $[u, v]$ not in T_x, the set $\mathcal{H}$ contains the candidate cycle formed by the two shortest paths p_{xu} and p_{xv} plus the edge $[u, v]$ itself. Degenerate cases where p_{xu} and p_{xv} share some edges are not considered. The number of such pairs being at most $n\nu$, all candidate cycles in $\mathcal{H}$ can be generated in $O(nm)$. Since all cycles in a graph form a matroid [4], a greedy procedure can be applied: cycles are sorted by nondecreasing weight in $O(nm \log n)$ and the ν lightest independent cycles are extracted by an independence test. Since the latter step, which is performed via Gaussian elimination in $O(m^3 n)$, is the bottleneck, this is also the overall complexity of the algorithm. In [11], the complexity is reduced to $O(m^\omega)$, where ω is the exponent of the fast matrix multiplication and $\omega < 2.376$ [6].

A different approach is proposed by de Pina in [7]. The cycles of a minimum cycle basis are determined sequentially and linear independence is guaranteed by maintaining at each step a basis of the linear space orthogonal to the subspace spanned by the cycles selected so far. The vectors of this basis are referred to as *witnesses*. Given an arbitrary spanning tree T of G and the corresponding set $E'_T = \{e_1, \ldots, e_\nu\}$ of co-tree edges, the initial basis $\{S_1, \ldots, S_\nu\}$ is the standard basis of $\mathrm{GF}(2)^{E'_T}$, where each S_j, for $1 \leq j \leq \nu$, has only the e_j-th component equal to 1. Any cycle of G can be viewed as a restricted incidence vector in $\mathrm{GF}(2)^{E'_T}$ and it is easy to verify that linear independence of the restricted vectors is equivalent to linear independence of the full incidence vectors. At the beginning of the i-th iteration, the basis consists of linearly independent vectors $S_i, \ldots, S_\nu$ such that $\langle C_l, S_j \rangle = 0$, for $i \leq j \leq \nu$ and $1 \leq l \leq i-1$, where $C_1, \ldots, C_{i-i}$ are the cycles already selected. Given the current set of witnesses $\{S_i, \ldots, S_\nu\}$, the cycle C_i is determined as the lightest cycle C in the graph G such that $\langle C, S_i \rangle = 1$. This can be done in $O(nm + n^2 \log n)$ by n shortest path computations in an appropriate graph. Since C_i is not orthogonal to S_i, C_i is linearly independent from $C_1, \ldots, C_{i-1}$. The optimality of the resulting cycle basis is guaranteed by the fact that such a lightest cycle C is selected at each step. The set of witnesses $\{S_{i+1}, \ldots, S_\nu\}$ can then be updated in $O(m^2)$ to become orthogonal to C_i (while preserving the orthogonality w.r.t. $\{C_1, \ldots, C_{i-1}\}$) by setting $S_j = S_j \oplus S_i$ if $\langle C_i, S_j \rangle = 1$, for each S_j with $i+1 \leq j \leq \nu$. By operating recursively on bulks of witnesses and exploiting fast matrix multiplication, all witness updates can be carried out in $O(m^\omega)$ and the complexity of the overall algorithm can be reduced from $O(m^3 + mn^2 \log n)$ to $O(m^2 n + mn^2 \log n)$ [15].

A hybrid algorithm that combines the two previous approaches is presented in [17]. First, Horton candidate cycles are generated and sorted by nondecreasing weight. Then a variant of de Pina's scheme is applied where, at the i-th iteration, the search for the lightest cycle C_i is performed in $\mathcal{H}$. The Horton cycles are tested according to nondecreasing weight until one non-orthogonal to S_i is found. Since there are at most $n\nu$ cycles, each one with at most n edges, each iteration is $O(mn^2)$ for a total of $O(m^2n^2)$. This dominates the $O(m^\omega)$ improved witness update and is also the overall complexity of the algorithm.

A hybrid variant with two major differences is proposed by Mehlhorn and Michail in [18]. Horton candidate cycles are generated only for pairs of vertex x and edge $[u, v]$ with x in a close-to-minimum *Feedback Vertex Set*, that is, a set of vertices that intersects every cycle. Such a FVS-based set of candidate cycles is still guaranteed to contain a minimum cycle basis. Although finding a minimum FVS is NP-hard, an approximate solution within a factor 2 can be obtained in polynomial time [3]. A simple way to extract a minimum cycle basis from the resulting set of candidate cycles then leads to an overall $O(m^2n)$ complexity, which can be further reduced to $O(m^2n/\log n + mn^2)$ by using a bit-packing trick.

3 Improved Deterministic Algorithms

As it was the case in [1], an important step towards improved deterministic algorithms is to restrict attention to a subset of Horton candidate cycles, referred to as isometric cycles. A cycle C is *isometric* if for any two vertices u and v on C, p_{uv} is in C, i.e., if C cannot be obtained as the composition of two cycles of smaller weight. The set of isometric cycles, denoted by $\mathcal{I}$, is a subset of $\mathcal{H}$ and still contains a minimum cycle basis. A cycle in $\mathcal{H}$ can be generated from different pairs of vertex x and edge $[u, v]$. Each pair gives rise to a different *representation* $(x, [u, v])$ of the same cycle that we denote by $C(x, [u, v])$. Clearly, the number of the representations of any cycle C cannot exceed the number of vertices, and hence of edges, in C. As observed in [1], the isometric cycles are exactly those which have a number of representations in $\mathcal{H}$ equal to the number of edges.

Although the resulting reduced set $\mathcal{I}$ of candidate cycles is still of cardinality $O(nm)$, it has the following simple but fundamental property.

> **Sparseness property**: The total number of edges (nonzero components in the corresponding incidence vectors) over all isometric cycles is at most $n\nu$.

In [1] we also describe an efficient $O(nm)$ procedure which allows to detect a single representation of each isometric cycle without explicitly constructing the non-isometric cycles.

It is interesting to point out that the concept of isometric cycle encompasses other previous attempts to reduce the size of the set of candidate cycles $\mathcal{H}$. In [13] Horton suggests to remove for each cycle C the redundant representations (which yield duplicate candidate cycles) from $\mathcal{H}$ by keeping only that generated

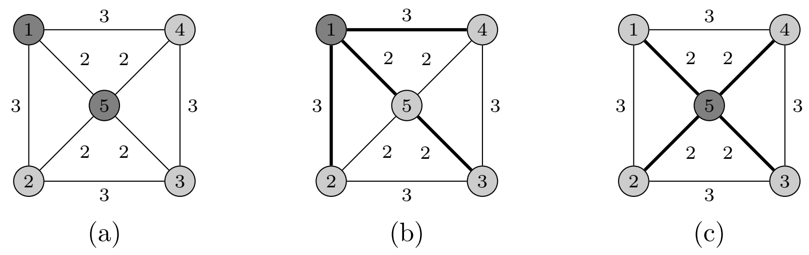


Fig. 1. (a) A small weighted graph in (a) with $n = 5$, $m = 8$, and $\nu = 4$. (b) T_1 in bold, with $C(1, [2, 5])$ and $C(1, [4, 5])$ of weight 7, $C(1, [2, 3])$ and $C(1, [3, 4])$ of weight 10; T_2, T_3, and T_4 are identical to T_1 up to rotations and lead to similar cycles. (c) T_5 in bold, with the 4 inner triangles of weight 7 which are the only isometric cycles. $\mathcal{H}$ without duplicates contains 7 cycles: the 4 isometric cycles plus $C(1, [2, 5])$, $C(1, [4, 5])$, $C(2, [3, 4])$, while $C(2, [1, 4])$ is discarded. Any minimum FVS leads to a FVS-based set of candidate cycles containing 6 cycles. For the FVS marked in (a), the FVS-based set contains $C(1, [2, 5])$ and $C(1, [4, 5])$ in addition to the 4 isometric cycles.

from the pair involving the vertex in C with smallest index, assuming a given numbering of the vertices from 1 to n. All redundant representations $(x, [u, v])$ can be discarded without additional time by preliminarily identifying if the path p_{xu} or the path p_{xv} contains a vertex smaller than x. Non-isometric cycles are discarded if they do not admit a representation for their vertex of smallest index. Since an isometric cycle has a representation for all its vertices and at least one of them belongs to any FVS, the FVS-based set of candidate cycles considered in [18] contains the set $\mathcal{I}$ of isometric cycles, while the reverse is not true. An example is illustrated in Fig. 1. As we shall see in Section 4, restricting attention to isometric cycles leads to a substantially smaller set of candidate cycles w.r.t. just removing all the duplicate ones or focusing on the FVS-based ones.

3.1 An $O(m^2n/\log n)$ Deterministic Algorithm

We follow a hybrid strategy. First, we generate all the isometric cycles ($\mathcal{I}$) with the above-mentioned $O(nm)$ procedure and we sort them by nondecreasing weight. This can be done in $O(nm \log n)$. Then we adopt de Pina's scheme with the key difference that we restrict the search for the cycles to the set $\mathcal{I}$.

We now describe how we achieve the $O(m^2n/\log n)$ worst-case complexity. We assume a RAM model of computation with words of logarithmic length (of at least $\lceil \log n \rceil$ bits) that allows bitwise operations in constant time. We refer to the divide-and-conquer framework described in [15] which exploits fast matrix multiplication to update many witnesses simultaneously. In [15], when searching for the new cycles to be included in the minimum basis, the witnesses are considered one by one but they are updated in bulks by using a recursive strategy. To achieve the improved worst-case complexity, we consider blocks of $\lceil \log n \rceil$ witnesses as single elements in the above-mentioned update procedure. In addition, within each block we exploit the sparsity property deriving from the restriction to isometric cycles and apply an *ad hoc* bit-packing technique.

We first present the underlying idea of the witness update procedure in [15] in an iterative, rather than recursive, manner. For the sake of exposition, we assume w.l.o.g. that the initial number of witnesses is a power of 2 with null vectors beyond the ν-th position. There are always $O(m)$ such vectors. At the i-th iteration, for $1 \leq i \leq \nu$, the i-th witness S_i is considered and the lightest cycle C_i with $\langle C_i, S_i \rangle = 1$ is found. The index i can be expressed in a unique way as $i = 2^q r$, where q is a nonnegative integer and r is a positive odd integer. Obviously, $q = 0$ if and only if i is odd. In the update phase, the 2^q witnesses $\{S_{i+1}, \ldots, S_{i+2^q}\}$ are updated so that they become orthogonal to $\{C_{i+1-2^q}, \ldots, C_i\}$, namely to the last 2^q cycles added to the basis. In [15] it is shown that the overall cost of all update phases is $O(m^\omega)$. In this work, we proceed by blocks of $b := \lceil \log n \rceil$ witnesses and consider at the i-th iteration the i-th block instead of a single witness. For exposition purpose, we assume w.l.o.g. that b divides the initial number of witnesses and that the number of blocks is a power of 2. For the i-th block, with $i = 2^q r$, after finding the corresponding b cycles of a minimum basis in the way we shall describe below, the next 2^q blocks of witnesses, namely the $2^q b$ witnesses contained in the blocks, are updated so that they become orthogonal to the last $2^q b$ cycles added to the basis. Since we consider blocks of witnesses, the total amount of work needed for all the witness updates is lower than that in [15] and still $O(m^\omega)$.

In order to find the b cycles of a minimum basis corresponding to the i-th block, we proceed as follows. At the beginning of the i-th iteration, we already have selected $s = (i-1)b$ cycles. We must specify how to find the following b cycles $C_{s+1}, \ldots, C_{s+b}$ using the corresponding b witnesses $S_{s+1}, \ldots, S_{s+b}$. We code the witnesses S_{s+j}, with $1 \leq j \leq b$, in the following way: for each edge $e \in E$ we have a word $W(e)$ in which the j-th bit is set to 1 if the e-th component of S_{s+j} is 1. Then, we scan the isometric cycles searching for the lightest cycle in the subspace spanned by $S_{s+1}, \ldots, S_{s+b}$, i.e., non-orthogonal to at least one witness S_{s+j}, for $1 \leq j \leq b$. A cycle $C \in \mathcal{I}$ is tested by making the x-or among logarithmic words $W(e)$, for each edge $e \in C$. Let $W(C)$ denote the resulting word. If $W(C)$ is null, then C is orthogonal to all considered witnesses, otherwise C is the selected cycle. The witnesses S_{s+j} non-orthogonal to C are those whose j-th bit in $W(C)$ is 1. Anyone of these witnesses, for example S_{s+t}, can be discarded while the remaining ones are updated. This can be implicitly done in $O(m)$ as follows: for each edge $e \in E$, if the e-th component in S_{s+t} is 1 then $W(e) := W(e)$ x-or $W(C)$, otherwise $W(e)$ remains unchanged. This is equivalent to make the update according to the standard rule, but has the advantage of preserving the witness coding in the desired form. Note that this internal update is necessary only after finding a cycle of a minimum basis, so that its cost in all phases is $O(m^2)$. After selecting C, we search for other $b-1$ cycles in the same way. To select the b cycles we examine each isometric cycle at most once. Due to the sparseness property, this takes $O(nm)$. Since the above step must be applied $O(m/\log n)$ times, a minimum cycle basis is obtained in $O(m^2 n/\log n)$. This dominates the $O(m^\omega)$ cost of the witness update and thus is the overall complexity of the algorithm.

Note that our algorithm is simpler than the $O(m^2n/\log n + mn^2)$ one proposed in [18] and has a better complexity for sparse graphs with $m < n \log n$.

3.2 An Efficient Adaptive Isometric Cycles Extraction Algorithm

Since computational results (see Section 4) show that the independence test is the bottleneck also in practice, we propose an efficient algorithm based not only on the isometric cycles but also on a very effective adaptive independence test *à la* de Pina. We refer to the overall algorithm as Adaptive Isometric Cycles Extraction (AICE).

The main drawback of de Pina's scheme is arbitrariness: the method works for any choice of the spanning tree T in G and for any ordering $e_1, \ldots, e_\nu$ of the corresponding co-tree edges that induces the initial set of witnesses. But different choices can lead to very different computational loads and times. Ideally, we would like to pick a spanning tree which requires, along the whole procedure, as few witness updates as possible and which tends to keep the witnesses as sparse as possible. We have devised a procedure that tries to reduce the overall number of elementary operations by iteratively and adaptively building the current spanning tree T. Roughly speaking, starting from an empty T we keep on adding the lightest available isometric cycle and greedily construct T including as many edges of this newly added cycle as possible. In the independence test we try to minimize the number of edges that must be considered and in the witness update we aim at maintaining vector sparsity.

A detailed description of AICE is given in Algorithm 1. To fully exploit the sparseness property, the isometric cycles and the witnesses are considered as ordered list of edges corresponding to nonzero components of their incidence vectors. Clearly, two incidence vectors are not orthogonal if and only if the intersection of the two lists have an odd cardinality.

In our algorithm, we restrict attention to the set $\mathcal{I}$ of isometric cycles and reverse the traditional de Pina's scheme, where for each witness S_i one looks for the lightest non-orthogonal cycle C_i. Indeed, we scan the isometric cycles according to nondecreasing weights and for each isometric cycle C_i we search for a witness S_i such that C_i and S_i are not orthogonal. If such a witness does not exist, C_i can be discarded. Another key difference is that we build the spanning tree T in an adaptive way. We start with an empty spanning tree T, whose set of edges is denoted by E_{T} (Step 2). Since the co-tree edges are unknown, we cannot build the witnesses of the initial standard basis but we can consider them implicitly until the co-tree edges are identified. The set $\mathcal{S}$ (initially empty) only contains the witnesses that have been made explicit by an update. In E_{coT} we also keep track of the co-tree edges relative to T that do no longer appear in the witnesses. Since the witness update is based on the symmetric difference of the edge sets, the edges in E_{coT} correspond to the components that are no longer relevant for the independence test. At each iteration, given the current (lightest) isometric cycle C (Step 4), we neglect the irrelevant edges (in E_{coT}) together with the edges in T (Step 5) and we check if some edges of C can be added to T without creating a cycle (Step 6). The remaining edges are then partitioned into

Algorithm 1. AICE algorithm for finding a minimum cycle basis

```
Input: an undirected graph $G = (V, E)$
Output: a minimum cycle basis $\mathcal{C}$
 1: Generate all the isometric cycles ($\mathcal{I}$) and sort them by nondecreasing weight
 2: $\mathcal{C} := \emptyset$, $\mathcal{S} := \emptyset$, $E_{\mathtt{T}} := \emptyset$, $E_{\mathtt{coT}} := \emptyset$
 3: while $E \neq E_{\mathtt{T}} \cup E_{\mathtt{coT}}$ do
 4:     Extract the lightest cycle $C$ from $\mathcal{I}$
 5:     $C_{\mathtt{R}} := C \backslash (E_{\mathtt{T}} \cup E_{\mathtt{coT}})$
 6:     $\forall e \in C_{\mathtt{R}} : E_{\mathtt{T}} \cup e$ is acyclic do $E_{\mathtt{T}} := E_{\mathtt{T}} \cup e$, $C_{\mathtt{R}} := C_{\mathtt{R}} \backslash e$
 7:     $E_{\mathtt{S}} := \{e \in C_{\mathtt{R}} : e \in S_j$ for some $S_j \in \mathcal{S}\}$, $E_{\mathtt{N}} := C_{\mathtt{R}} \backslash E_{\mathtt{S}}$
 8:     if $E_{\mathtt{N}} = \{e_1, \ldots, e_k\} \neq \emptyset$ then
 9:         if $k > 1$ then $\mathcal{S} := \mathcal{S} \cup S_j$ with $S_j := \{e_j, e_{j+1}\}$, for $1 \leq j \leq k-1$ end if
10:         if $E_{\mathtt{S}} \neq \emptyset$ then $\forall S_j \in \mathcal{S} : |E_{\mathtt{S}} \cap S_j|$ is odd do $S_j := S_j \cup e_1$ end if
11:         if $k = 1$ and no $S_j \in \mathcal{S}$ is updated in (10) then $E_{\mathtt{coT}} := E_{\mathtt{coT}} \cup e_1$ end if
12:     else if $E_{\mathtt{S}} \neq \emptyset$ and $\exists S_i \in \mathcal{S} : |E_{\mathtt{S}} \cap S_i|$ is odd then
13:         $\mathcal{S} := \mathcal{S} \backslash S_i$
14:         $\forall S_j \in \mathcal{S} : |E_{\mathtt{S}} \cap S_j|$ is odd do $S_j := S_j \oplus S_i$
15:         $E_{\mathtt{coT}} := E_{\mathtt{coT}} \cup \{e \in E_{\mathtt{S}} : e \notin S_j$ for every $S_j \in \mathcal{S}\}$
16:     else
17:         Discard $C$
18:     end if
19:     if $C$ is not discarded then $\mathcal{C} := \mathcal{C} \cup C$ end if
20: end while
```

$E_{\mathtt{S}}$ and $E_{\mathtt{N}}$ depending on whether they belong or do not belong to some explicit witnesses (Step 7), where $E_{\mathtt{N}}$ contains the newly identified co-tree edges. Different cases may occur. In the most favorable one (Step 11), C contains only one new co-tree edge and no witnesses need to be updated. Note that this is equivalent to select an implicit witness S_i with a single nonzero component that, from that point in time, can be neglected and hence added to $E_{\mathtt{coT}}$. If C contains k new co-tree edges, with $k > 2$, we should select one of the corresponding implicit witnesses and update the remaining $k - 1$ ones. This is achieved in Step 9 by defining each one of the $k - 1$ witnesses by two consecutive co-tree edges so that each one of these new co-tree edges is in at most two witnesses. It is easy to verify that these witnesses are orthogonal to C. Explicit witnesses in $\mathcal{S}$ must only be updated when $E_{\mathtt{S}}$ is not empty. Note that the inner product with the witnesses is restricted to this set of edges. The explicit witnesses that are not orthogonal to the current cycle C are then appropriately updated. Whenever possible the update is carried out by selecting as S_i an implicit witness with a single nonzero component (Step 10) or anyone of the explicit witnesses (Steps 12 to 15).

It is worth pointing out that in spite of the heuristic choices aiming at keeping the witnesses as sparse as possible, the overall algorithm is guaranteed to yield a minimum cycle basis. Although this approach does not seem to have a better worst-case complexity than the original de Pina's one, we shall see that it is computationally very efficient.

4 Computational Results

To evaluate the practical performance of our AICE algorithm, we carried out an extensive computational campaign and compared the results with those provided by the main alternative algorithms, namely, the improved de Pina's one described in [15], the hybrid one in [17], the hybrid one based on FVS [18] and an efficient implementation of Horton's method. In the previous computational work [17], implementations are based on LEDA [19] and the executables are available. For fairness of comparison, we have implemented all algorithms in C with the same data structures. Since, due to vector sparsity, we do not expect actual speed-up from fast matrix multiplication and bit packing tricks, we have not included these features in the implementation of the previous algorithms and we have not tested the algorithm described in Section 3.1.

As benchmark we use random graphs and hypercubes like those in [17] but of larger size, and Euclidean graphs like in [2]. Random graphs are generated using the $G(n;p)$ model [9], for $p = 0.3, 0.5, 0.9$ and for $p = 4/n$ (sparse graphs, $m \approx n$), and hypercubes are in dimension $d = 7, \ldots, 10$, with $n = 2^d$. Both unweighted and weighted cases are considered, with integer weights randomly chosen from the uniform distribution in the range $[0 \ldots 2^{16}]$. Euclidean graphs have density $p = 0.3, 0.5, 0.9$, and the weight of an edge is the Euclidean distance between its endpoints, supposing that vertices are randomly distributed on a 10x10 square. The algorithms are run on a Xeon 2.0 Ghz 64 bit machine with 3 GB of memory, in Linux. We use the GNU gcc 4.1 compiler (g++ 4.1 for LEDA implementations) with the -O3 optimization flag. For each type of graph, results are reported as average (for ν and the size of the witnesses the averages are rounded up) on 20 randomly generated instances with the same characteristics.

Table 1 reports the ratio between the size of the sets of candidate cycles and the cyclomatic number ν. Both sets $\mathcal{H}$ and $\mathcal{H}_{\texttt{FVS}}$ (the one of FVS-based candidate cycles) are without duplicates. The set $\mathcal{H}_{\texttt{FVS}}$ does not contain the representations $(x, [u, v])$ whose corresponding cycle has a smaller vertex than x also belonging to the given FVS. A close-to-minimum FVS is obtained by the heuristic in [5]. For weighted graphs, the number of isometric cycles is very close to ν, so that very few of them must be tested for independence. Also the standard deviation is in general very small, less than 2%. The size of $\mathcal{H}$ and $\mathcal{H}_{\texttt{FVS}}$ is up to 20 times larger than ν, with a higher standard deviation, $10 - 20\%$. Thus even a close-to-minimum FVS does not help. If duplicate cycles are not deleted, the size of both $\mathcal{H}$ and $\mathcal{H}_{\texttt{FVS}}$ turns out to be up to 10 times larger (for lack of space not shown in the table).

For unweighted graphs, isometric cycles are less effective for larger density because in this case a random graph tends to become a complete graph with $n\nu/3$ isometric triangles. The particular weighting of Euclidean graphs also reduces the impact of the restriction to isometric cycles.

In order to try to further trim the set of isometric cycles we have removed isometric cycles that admit a *wheel decomposition*, i.e., which can be expressed as the composition of more than two cycles of smaller weight with a vertex r not incident to any edge in C such that each co-tree edge in C relative to T_r closes a lighter fundamental cycle in T_r. For an example see Fig. 2(a). The

Table 1. Comparison of the number of candidate cycles divided by ν for the Horton cycles ($\mathcal{H}$), the FVS-based cycles ($\mathcal{H}_{\texttt{FVS}}$), and the isometric cycles ($\mathcal{I}$). Both $\mathcal{H}$ and $\mathcal{H}_{\texttt{FVS}}$ do not include duplicates. For unweighted graphs, the isometric cycles which do not admit a wheel decomposition ($\mathcal{I}_{\texttt{noWh}}$) are also considered, with the generation time (in seconds). The label "x" indicates memory limit exceeded (3 GB).

			Weighted			Unweighted			
	n	ν	$\mathcal{H}$	$\mathcal{H}_{\texttt{FVS}}$	$\mathcal{I}$	$\mathcal{H}$	$\mathcal{H}_{\texttt{FVS}}$	$\mathcal{I}$	$\mathcal{I}_{\texttt{noWh}}$ (Time)
Random 0.3	100	1375	6.67	6.35	1.07	9.28	10.86	4.57	1.97 (0.03)
	200	5775	11.65	11.51	1.06	13.02	15.94	7.78	2.44 (0.42)
	300	13173	16.39	16.19	1.05	15.89	18.94	10.74	2.67 (1.70)
	400	23527	19.40	19.23	1.04	19.13	23.82	13.76	2.83 (4.46)
	500	36903	23.42	23.29	1.04	22.12	27.56	16.78	2.91 (9.46)
Random 0.5	100	2379	6.29	6.21	1.04	10.81	12.55	9.05	1.76 (0.06)
	200	9749	11.65	11.59	1.03	19.06	20.18	17.35	1.83 (0.65)
	300	22119	15.45	15.38	1.03	27.47	29.29	25.67	1.89 (2.56)
	400	39556	20.01	19.88	1.03	35.98	40.43	34.19	1.93 (5.93)
	500	61871	24.14	24.17	1.02	44.31	46.25	42.41	1.97 (11.66)
Random 0.9	100	4360	6.62	6.66	1.02	27.34	27.36	27.12	1.06 (0.10)
	200	17706	11.15	11.15	1.02	54.26	54.58	54.04	1.05 (1.03)
	300	40052	15.15	15.09	1.01	81.22	82.04	81.02	1.01 (3.81)
	400	71425	20.30	20.27	1.01	108.27	108.32	108.09	1.00 (9.19)
	500	111770	23.53	23.49	1.01	135.27	135.70	135.06	x

Weighted random sparse				
n	ν	$\mathcal{H}$	$\mathcal{H}_{\texttt{FVS}}$	$\mathcal{I}$
500	510	18.21	13.72	2.15
750	761	25.59	19.45	2.51
1000	1010	31.86	23.10	2.73

Weighted hypercubes				
d	ν	$\mathcal{H}$	$\mathcal{H}_{\texttt{FVS}}$	$\mathcal{I}$
7	321	7.67	7.59	1.56
8	769	11.99	12.22	1.69
9	1793	20.12	20.70	1.79
10	4097	35.31	34.44	1.92

Euclidean $n = 200$					
p	ν	$\mathcal{H}$	$\mathcal{H}_{\texttt{FVS}}$	$\mathcal{I}$	$\mathcal{I}_{\texttt{noWh}}$ (Time)
0.1	1781	27.13	25.26	3.32	1.08 (0.08)
0.3	5770	41.69	40.96	7.92	1.03 (0.30)
0.5	9733	49.54	49.21	17.65	1.02 (0.68)
0.7	13728	56.89	56.78	33.04	1.01 (1.52)
0.9	17697	63.36	63.28	53.83	1.01 (2.96)

Table 2. Time in seconds and size of the witnesses for the AICE independence test and the original de Pina's test restricted to the isometric cycles. For AICE, the percentage of the selected cycles which do not require witness update (% c.n.u.) is also reported.

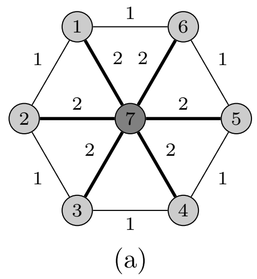
			Weighted							Unweighted						
			De Pina's test on $\mathcal{I}$			AICE test				De Pina's test on $\mathcal{I}$			AICE test			
	n	ν	Time	max $\|S_i\|$	$\lceil$avg$\|S_i\|\rceil$	Time	max $\|S_i\|$	$\lceil$avg$\|S_i\|\rceil$	% c.n.u.	Time	max $\|S_i\|$	$\lceil$avg$\|S_i\|\rceil$	Time	max $\|S_i\|$	$\lceil$avg$\|S_i\|\rceil$	% c.n.u.
Random 0.3	100	1375	0.10	628	9	0.00	12	2	97.20	0.08	665	11	0.00	39	2	74.73
	200	5775	2.02	2656	13	0.00	16	2	98.08	1.57	2767	16	0.01	56	2	87.18
	300	13173	15.88	6185	15	0.01	26	2	98.75	13.03	6399	18	0.02	39	2	92.51
Random 0.5	100	2379	0.29	1105	10	0.00	12	2	98.15	0.21	1117	9	0.00	9	2	96.41
	200	9749	5.67	4560	13	0.01	17	2	98.93	4.82	4567	11	0.02	8	2	98.33
	300	22119	74.68	10411	15	0.02	24	2	99.20	40.70	10461	12	0.07	8	2	98.83
Random 0.9	100	4360	0.92	2007	9	0.00	8	2	99.16	0.54	1973	7	0.01	2	2	99.80
	200	17706	30.25	8273	13	0.02	13	2	99.44	11.96	8074	7	0.07	2	2	99.91

Table 3. Comparison of the running times in seconds for the main algorithms, for our C implementations as well as the LEDA available ones. The labels "-" and "x" indicate that the time limit of 900 seconds and the memory limit of 3 GB are respectively exceeded.

	LEDA implementation			Our implementation					LEDA implementation			Our implementation				
	De Pina	Hybrid	Hyb FVS	De Pina	Hybrid	Hyb FVS	Horton	AICE	De Pina	Hybrid	Hyb FVS	De Pina	Hybrid	Hyb FVS	Horton	AICE
n	Weighted random 0.3								Unweighted random 0.3							
100	1.09	16.68	7.09	0.31	0.35	0.89	0.05	0.01	3.55	0.64	1.80	0.43	0.07	0.61	0.04	0.01
200	18.49	-	413.95	4.83	46.10	27.37	0.70	0.08	139.47	17.73	47.60	10.30	1.18	5.57	0.54	0.08
300	111.01	-	-	28.11	439.40	275.68	3.83	0.26	-	x	263.39	60.34	10.57	23.68	2.68	0.29
400	574.30	-	-	161.52	-	-	11.39	0.78	-	x	-	418.28	47.93	79.46	8.00	0.76
500	-	-	-	474.31	-	-	30.98	1.90	-	x	-	-	143.36	211.75	20.13	1.67
n	Weighted random 0.5								Unweighted random 0.5							
100	2.58	69.13	21.93	0.74	1.56	1.53	0.09	0.02	9.13	1.63	3.07	0.87	0.17	0.67	0.18	0.02
200	46.64	-	-	13.60	161.95	91.03	1.32	0.12	268.18	55.40	81.61	13.64	3.27	6.57	3.40	0.19
300	399.43	-	-	101.65	-	676.70	6.02	0.39	-	x	446.03	123.18	28.26	33.15	17.26	0.68
400	-	-	-	475.95	-	-	24.67	1.18	-	x	-	528.03	107.59	120.76	61.49	1.72
500	-	-	-	-	-	-	50.97	2.54	-	x	-	-	283.58	309.33	171.99	3.54
n	Weighted random 0.9								Unweighted random 0.9							
100	7.34	309.14	76.75	1.98	10.41	3.53	0.18	0.03	18.86	3.73	5.42	1.24	0.33	0.59	1.29	0.06
200	180.76	-	-	47.28	477.55	264.74	2.17	0.20	-	x	152.99	24.14	5.27	6.95	23.76	0.55
300	-	-	-	472.30	-	-	11.31	0.71	-	x	827.04	195.04	31.75	32.62	154.88	2.04
400	-	-	-	-	-	-	42.62	1.86	-	x	-	-	108.90	109.36	601.42	5.30
500	-	-	-	-	-	-	86.69	4.27	-	x	-	-	296.74	289.61	-	11.12

	LEDA implementation			Our implementation				
	De Pina	Hybrid	Hyb FVS	De Pina	Hybrid	Hyb FVS	Horton	AICE
n	Weighted random sparse							
500	0.83	2.52	31.71	0.44	0.27	59.54	0.27	0.21
750	2.33	7.65	108.38	1.25	0.81	260.37	0.81	0.63
1000	4.74	16.47	249.72	2.63	1.70	769.09	1.65	1.32
d	Weighted hypercubes							
7	0.14	0.21	0.88	0.06	0.02	0.79	0.01	0.00
8	0.96	2.49	14.25	0.44	0.11	7.44	0.07	0.03
9	6.41	25.21	134.52	3.04	0.95	75.57	0.58	0.32
10	42.85	265.08	-	19.28	10.37	-	3.92	1.96

	Our implementation				
	De Pina	Hybrid	Hyb FVS	Horton	AICE
p	Euclidean $n = 200$				
0.1	1.31	1.72	6.19	0.35	0.03
0.3	6.30	45.51	53.16	2.07	0.10
0.5	15.46	152.94	190.86	4.34	0.29
0.7	27.74	312.83	430.56	7.59	0.75
0.9	47.88	521.49	790.85	11.10	1.56



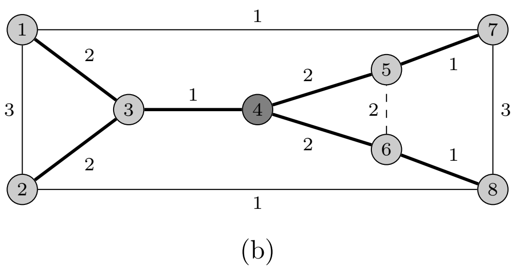


Fig. 2. (a) A small example of wheel decomposition centered in vertex 7 (T_7 in bold): the isometric hexagonal cycle of weight 6 can be obtained as the composition of the 6 inner triangles of weight 5. (b) A wheel decomposition centered in vertex 4 (T_4 in bold) of the outer cycle of weight 8. Since the non-isometric cycle $C(4,[7,8])$ of weight 9 can be separated by $[5,6]$ (dashed edge) in two cycles of strictly smaller weight (6 and 7), when decomposing the outer cycle the weight of the heavier of the two cycles, 7, is considered instead of 9.

isometric cycles are checked for a wheel decomposition after they are sorted by non-decreasing weight. If a fundamental cycle of a possible wheel decomposition is non-isometric, we consider the weight of the heavier of the two cycles in which it can be separated, see Fig. 2(b). Since the total number of edges is bounded by $n\nu$ (sparseness property) and there are n vertices, it is easy to see that all candidate cycles that admit a wheel decomposition can be discarded in $O(mn^2)$.

In Table 1 the set of isometric cycles that are left after checking for wheel decomposition is denoted by $\mathcal{I}_{\texttt{noWh}}$. Although its size is very close to ν, in particular for Euclidean graphs, the time needed to obtain $\mathcal{I}_{\texttt{noWh}}$ is very high compared to the total time of AICE (reported in Table 3). This suggests that, due to the efficiency of the independence test, it does not pay to further reduce the number of cycles with a similar technique.

In Table 2, we assess the impact of the adaptive independence test of AICE w.r.t. the original de Pina's test. For a fair comparison, they are both applied to the set $\mathcal{I}$ of isometric cycles. AICE results are averages on 20 instances, whereas de Pina's test results are averages on n instances ($n = 100, 200, 300$) corresponding to n different randomly generated spanning trees that induce the initial (standard) basis $\{S_1, \ldots, S_\nu\}$. As in [17], we report statistics on the size (number of nonzero components) of the witnesses S_i, for $1 \le i \le \nu$, used in the independence test, namely the maximum and the rounded up average cardinality. Note that in the AICE tests the rounded up average of the size of the witnesses S_i, with $1 \le i \le \nu$, is always equal to 2. The maximum has a large standard deviation, since it depends on the specific instance, but it is always much smaller than that of de Pina's test, whose standard deviation is less than 10%. Not only the size of the witnesses is very small but for almost all cycles identified by AICE no witness update is needed (Step 11 in Algorithm 1). Since many unnecessary operations are avoided, the overall computing time is greatly reduced.

In Table 3, we compare the running times of the main algorithms. First, we consider the algorithms whose implementation based on LEDA [19] is available,

Table 4. Comparison of the running times in seconds for large instances. In most cases, the previous algorithms exceed the time limit of 1800 seconds ("-").

	De Pina	Hybrid	Horton	AICE
n	Weighted random 0.3			
1000	-	-	361.04	24.10
1500	-	-	-	87.46
2000	-	-	-	195.82
2500	-	-	-	396.12
3000	-	-	-	685.16
n	Weighted random 0.5			
1000	-	-	629.57	32.76
1500	-	-	-	119.13
2000	-	-	-	278.72
2500	-	-	-	575.96
3000	-	-	-	1061.33

	De Pina	Hybrid	Horton	AICE
n	Weighted random 0.9			
1000	-	-	1261.80	59.66
1500	-	-	-	204.82
2000	-	-	-	490.56
2500	-	-	-	900.61
3000	-	-	-	1424.03
p	Euclidean $n = 500$			
0.1	42.53	416.87	16.41	0.85
0.3	366.34	-	85.69	3.01
0.5	1117.00	-	189.53	8.38
0.7	-	-	330.68	21.14
0.9	-	-	510.17	43.67

namely de Pina's [15], the Hybrid [17] and the FVS-based Hybrid [18] algorithms. For a fair comparison, these algorithms have also been implemented in C within the same environment used for AICE. De Pina's algorithm is implemented with the heuristics suggested in [15] and the Hybrid method is that in [17] but duplicates are removed from $\mathcal{H}$. In the FVS-based Hybrid algorithm a close-to-minimum FVS is obtained by the heuristic in [5], but the heuristic computing time is neglected, and duplicate cycles are also removed. We also devised an efficient version of Horton's algorithm using $\mathcal{H}$ without duplicates and an *ad hoc* Gaussian elimination exploiting operations on GF(2). For Euclidean graphs LEDA algorithms cannot be tested since they require integer weights. The time limit is set to 900 seconds. For all results the standard deviation is in general less than 10%. Our C implementation of the previous algorithms turns out to be more effective than the ones based on LEDA. It is worth pointing out that, except for dense unweighted random graphs, the *ad hoc* implementation of Horton's algorithm is substantially faster than sophisticated algorithms based on de Pina's idea. However, AICE outperforms all previous algorithms, in most cases by one or two order of magnitude.

Finally, in Table 4 we report the results of our C implementations for larger weighted instances. AICE finds an optimal solution for graphs with up to 3000 vertices within the time limit of 1800 seconds, while the other algorithms cannot solve most of the instances.

An interesting open question is whether it is possible to do without the independence test, even though in practice it is unlikely to lead to an efficient algorithm.

Acknowledgements. We thank Kurt Mehlhorn and Dimitrios Michail for making available the executables of their algorithms.

References

1. Amaldi, E., Iuliano, C., Jurkiewicz, T., Mehlhorn, K., Rizzi, R.: Breaking the $O(m^2n)$ barrier for minimum cycle bases. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 301–312. Springer, Heidelberg (2009)

2. Amaldi, E., Liberti, L., Maculan, N., Maffioli, F.: Edge-swapping algorithms for the minimum fundamental cycle basis problem. Mathematical Methods of Operations Research 69(12), 205–233 (2009)
3. Bafna, V., Berman, P., Fujito, T.: A 2-approximation algorithm for the undirected feedback vertex set problem. SIAM J. Discrete Math. 12(3), 289–297 (1999)
4. Bollobas, B.: Graduate Texts in Mathematics, vol. 184. Springer, Heidelberg (2nd printing)
5. Brunetta, L., Maffioli, F., Trubian, M.: Solving the feedback vertex set problem on undirected graphs. Discrete Applied Mathematics 101(1-3), 37–51 (2000)
6. Coppersmith, D., Winograd, S.: Matrix multiplication via arithmetic progressions. J. Symb. Comput. 9(3), 251–280 (1990)
7. De Pina, J.C.: Applications of shortest path methods. Ph.D. thesis, University of Amsterdam, The Netherlands (1995)
8. Deo, N., Prabhu, G., Krishnamoorthy, M.S.: Algorithms for generating fundamental cycles in a graph. ACM Trans. on Mathematical Software 8(1), 26–42 (1982)
9. Erdös, P., Rényi, A.: On random graphs, I. Publicationes Mathematicae (Debrecen) 6, 290–297 (1959)
10. Gleiss, P.M.: Short cycles: minimum cycle bases of graphs from chemistry and biochemistry. Ph.D. thesis, Universität Wien, Austria (2001)
11. Golynski, A., Horton, J.D.: A polynomial time algorithm to find the minimum cycle basis of a regular matroid. In: Penttonen, M., Schmidt, E.M. (eds.) SWAT 2002. LNCS, vol. 2368, pp. 200–209. Springer, Heidelberg (2002)
12. Hartvigsen, D., Mardon, R.: The all-pairs min cut problem and the minimum cycle basis problem on planar graphs. SIAM J. Discrete Math. 7(3), 403–418 (1994)
13. Horton, J.D.: A polynomial-time algorithm to find the shortest cycle basis of a graph. SIAM J. Computing 16(2), 358–366 (1987)
14. Kavitha, T., Liebchen, C., Mehlhorn, K., Michail, D., Rizzi, R., Ueckerdt, T., Zweig, K.A.: Cycle bases in graphs characterization, algorithms, complexity, and applications. Computer Science Review 3(4), 199–243 (2009)
15. Kavitha, T., Mehlhorn, K., Michail, D., Paluch, K.E.: An $\tilde{O}(m^2 n)$ algorithm for minimum cycle basis of graphs. Algorithmica 52(3), 333–349 (2008)
16. Liebchen, C., Rizzi, R.: Classes of cycle bases. Discrete Applied Mathematics 155(3), 337–355 (2007)
17. Mehlhorn, K., Michail, D.: Implementing minimum cycle basis algorithms. ACM Journal of Experimental Algorithmics 11 (2006)
18. Mehlhorn, K., Michail, D.: Minimum cycle bases: Faster and simpler. Accepted for publication in ACM Trans. on Algorithms (2007)
19. Mehlhorn, K., Näher, S.: LEDA: A Platform for Combinatorial and Geometric Computing. Cambridge University Press, Cambridge (1999)
20. Rizzi, R.: Minimum weakly fundamental cycle bases are hard to find. Algorithmica 53(3), 402–424 (2009)
21. Stepanec, G.F.: Basis systems of vector cycles with extremal properties in graphs. Uspekhi Mat. Nauk II 19, 171–175 (1964) (in Russian)
22. Zykov, A.A.: Theory of Finite Graphs. Nauka, Novosibirsk (1969) (in Russian)

Efficient Algorithms for Average Completion Time Scheduling

René Sitters[⋆]

Department of Econometrics and Operations Research, Free University, Amsterdam
rsitters@feweb.vu.nl

Abstract. We analyze the competitive ratio of algorithms for minimizing (weighted) average completion time on identical parallel machines and prove that the well-known shortest remaining processing time algorithm (SRPT) is 5/4-competitive w.r.t. the average completion time objective. For weighted completion times we give a deterministic algorithm with competitive ratio $1.791 + o(m)$. This ratio holds for preemptive and non-preemptive scheduling.

1 Introduction

There is a vast amount of papers on minimizing average completion in machine scheduling. Most appeared in the combinatorial optimization community in the last fifteen years. The papers by Schulz and Skutella [22] and Correa and Wagner [6] give a good overview.

The shortest remaining processing time (SRPT) algorithm is a well-known and simple online procedure for preemptive scheduling of jobs. It produces an optimal schedule on a single machine with respect to the average completion time objective [20]. The example in Figure 1 shows that this is not true when SRPT is applied to parallel machines. The best known upper bound on its competitive ratio was 2 [18] until recently (SODA2010), Chung et al. [5] showed that the ratio is at most 1.86. Moreover, they show that the ratio is not better than $21/19 > 1.105$. In this paper, we show that the competitive ratio of SRPT is at most 1.25.

The SRPT algorithm has a natural generalization to the case where jobs have given weights. Unfortunately, our proof does not carry over to this case. No algorithm is known to have a competitive ratio less than 2. Remarkably, even for the offline problem, the only ratio less than 2 results from the approximation scheme given by Afrati et al. [1]. Schulz and Skutella [22] give a randomized 2-approximate algorithm which can be derandomized and applied online (although not at the same time). A deterministic online algorithm for the preemptive case is given by Megow and Schulz [16] and for the non-preemptive case by Correa and Wagner [6]. The ratios are, respectively, 2 and 2.62. The first bound on the algorithm is tight, the latter is probably not. On the single machine, no

[⋆] Supported by a research grant from the Netherlands Organization for Scientific Research (NWO-veni grant).

non-preemptive online algorithm can be better than 2 competitive [27] but it was unknown if the same is true for parallel machines. We give a simple online algorithm that runs in $O(n \log n)$ time and has competitive ratio $1.791 + o(m)$, i.e., it drops down to 1.791 for $m \to \infty$. This gives new insight in online and offline algorithms for average completion time minimization on parallel machines.

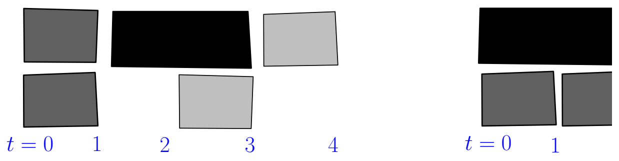


Fig. 1. There are two machines. At time 0, two jobs of length 1 and one job of length 2 are released and at time 2, two jobs of length 1 are released. The picture shows the suboptimal SRPT schedule and the optimal schedule.

The first approximation guarantee for weighted non-preemptive scheduling was given by Hall et al. [10]. This ratio of $4+\epsilon$ was reduced to 3.28 by Megow and Schulz [16] and then reduced to 2.62 by Correa and Wagner [6]. Table 1 gives a summary of known best ratios for a selection of problems. Remarkable is the large gap between lower and upper bounds for parallel machines. Not mentioned in the table are recent papers by Jaillet and Wagner [13] and by Chou et al. [4] which analyze the asymptotic ratio for several of these problems. Asymptotic, in this case, means that jobs have comparable weights and the number of jobs goes to infinity.

1.1 Problem Definition

An instance is given by a number of machines m, a job set $J \subset \mathbb{N}$ and for each $j \in J$ integer parameters $p_j \geq 1, r_j \geq 0, w_j \geq 0$ indicating the required processing time, the release time, and the weight of the job. A schedule is an assignment of jobs to machines over time such that no job is processed by more than one machine at the time and no machine processes more than one job at the time. In the non-preemptive setting, each job j is assigned to one machine and is processed without interruption. In the preemptive setting, we may repeatedly interrupt the processing of a job and continue it at any time on any machine.

Table 1. Known lower and upper bounds on the competitive ratio for randomized and deterministic online algorithm

Problem (Online)	L.B. Rand.		U.B. Rand.		L.B. Det.		U.B. Det.	
$1\|r_j, pmtn\|\sum_j C_j$	1		1	[20]	1		1	[20]
$1\|r_j, pmtn\|\sum_j w_j C_j$	1.038	[7]	4/3	[21]	1.073	[7]	1.57	[24]
$1\|r_j\|\sum_j C_j$	$e/(e-1)$	[26]	$e/(e-1) \approx 1.58$	[3]	2	[27]	2	[11] [15] [18]
$1\|r_j\|\sum_j w_j C_j$	$e/(e-1)$	[26]	1.69	[9]	2	[27]	2	[2] [19]
$P\|r_j, pmtn\|\sum_j C_j$	1	–	$1.86 \to 5/4$	[5]	1.047	[27]	$1.86 \to 5/4$	[5]
$P\|r_j, pmtn\|\sum_j w_j C_j$	1	–	$2 \to 1.791$	[22] [16]	1.047	[27]	$2 \to 1.791$	[16]
$P\|r_j\|\sum_j C_j$	1.157	[23]	$2 \to 1.791$	[22]	1.309	[27]	$2 \to 1.791$	[14]
$P\|r_j\|\sum_j w_j C_j$	1.157	[23]	$2 \to 1.791$	[22]	1.309	[27]	$2.62 \to 1.791$	[6]

The algorithm has to construct the schedule online, i.e., the number of machines is known a priori but jobs are only revealed at their release times . Even the number of jobs $n = |J|$ is unknown until the last job has been scheduled. Given a schedule, we denote the completion time of job j by C_j. The value of a schedule is the weighted average completion time $\frac{1}{n}\sum_{j\in J} w_j C_j$ and the objective is to find a schedule with small value. We say that an algorithm is c-competitive if it finds for any instance a schedule with value at most c times the optimal value.

2 The Competitive Ratio of SRPT

Phillips et al. [18] showed that *SRPT* is at most 2-competitive and showed that their analysis is tight. Hence, a new idea is needed to prove a smaller ratio. Indeed, the proof by Chung et al [5] is completely different and uses a sophisticated randomized analysis of the optimal solution. On the contrary, our proof builds on the original proof of Phillips et al. and continues where that proof stops. Their main lemma is one of the four lemmas in our proof (Lemma 2).

In the proof, we may restrict ourselves to schedules that preempt jobs only at integer time points since all processing times and release times are integer. For any integer $t \geq 1$ we define *slot* t as the time interval $[t-1, t]$. By this notation, the first slot that is available for j is slot $r_j + 1$. Given a (partial) schedule σ, we say that job j is *unfinished* at time t (or, equivalently, *unfinished* before slot $t+1$) if less than p_j units are processed before t in σ. A job j is *available* at time t (or, equivalently, *available* for slot $t+1$) if $r_j \leq t$ and j is unfinished at time t. Let $\sigma(t)$ be the set of jobs processed in slot t and denote by $\mu_i(\sigma)$ the i-th smallest completion time in σ.

The SRPT algorithm:

```
Let $t = 1$. Repeat:
If there are more than $m$ jobs available for slot $t$, then process $m$ jobs in slot $t$ that have the shortest remaining processing times among all available jobs. Otherwise, process all available jobs. Let $t = t + 1$.
```

The SRPT algorithm as defined here is not deterministic since it may need to choose between jobs with the same remaining processing time. We say that a schedule σ is an *SRPT schedule* for instance I if it is a possible output of the SRPT algorithm applied to I. Note that the values $\mu_i(\sigma)$ do not depend on the non-deterministic choices of the algorithm, i.e., if σ and σ' are SRPT schedules for the same instance on n jobs, then $\mu_i(\sigma) = \mu_i(\sigma')$ for all $i \in \{1, 2, \ldots, n\}$.

All four lemmas are quite intuitive. For the first lemma, imagine that for a given instance we reduce the release time of some job by δ and increase its processing time by at least the same amount. Then, the optimum value cannot improve since there is no advantage in starting a job earlier if this is undone by an increase in its processing time. The first lemma shows that SRPT has an even stronger property in this case.

Lemma 1. *Let I and I' satisfy $J = J'$ and for each $j \in J$ satisfy $r'_j = r_j - \delta_j \geq 0$ and $p'_j \geq p_j + \delta_j$, for some integers $\delta_j \geq 0$. Let σ and σ' be SRPT schedules for, respectively, I and I'. Then, for every $i \in \{1, 2, \dots, n\}$,*

$$\mu_i(\sigma) \leq \mu_i(\sigma').$$

Proof. We proof it by induction on the makespan of σ. Let $q_j(t)$ and $q'_j(t)$ be the remaining processing time of job j in, respectively, σ and σ' at time t. Define the multiset $Q(t) = \{q_j(t) \mid r_j \leq t\}$, i.e., it contains the remaining processing times of all jobs released at t or earlier. Let $Q'(t)$ contain the remaining processing times of the same set in σ', i.e., $Q'(t) = \{q'_j(t) \mid r_j \leq t\}$. Note that we take r_j and not r'_j in Q'. Let $Q_i(t)$ and $Q'_i(t)$ be the i-th smallest element in, respectively, $Q(t)$ and $Q'(t)$. We claim that for any time point t,

$$Q_i(t) \leq Q'_i(t), \text{ for all } i \in \{1, 2, \dots, |Q_i(t)|\}. \tag{1}$$

If we can show (1) then the proof follows directly since $\mu_i(\sigma)$ ($\mu_i(\sigma')$) is the smallest t such that $Q_i(t)$ ($Q'_i(t)$) has at least i zero elements.

The proof is by induction on t. It is true for $t = 0$ since $Q(0) = Q'(0)$. Now consider an arbitrary time t_0 and assume the claim is true for and $t \leq t_0$.

First we analyze the changes when no job is released at time $t_0 + 1$. If σ processes less than m jobs in slot t then all non-zero elements in $Q(t_0)$ are reduced by one, implying, $Q_i(t_0 + 1) \leq Q'_i(t_0 + 1)$ for all $i \leq |Q_i(t_0 + 1)|$. Now assume σ processes less than m jobs in slot t_0. Then it processes jobs with remaining processing times $Q_{k+1}(t_0), Q_{k+2}(t_0), \dots, Q_{k+m}(t_0)$ for some $k \geq 0$ while $Q_j(t_0) = 0$ for any $j \leq k$. Since $Q'_{k+1}(t_0), Q'_{k+2}(t_0), \dots, Q'_{k+m}(t_0)$ are also non-zero, only values $Q'_s(t_0)$ with $s \leq k + m$ are reduced for σ'. Again, $Q_i(t_0 + 1) \leq Q'_i(t_0 + 1)$ for all $i \leq |Q_i(t_0 + 1)|$.

Now assume some jobs are released at time $t_0 + 1$. We may use the analysis above and only consider the affect of the newly added jobs. For any new job j we have $p_j = q_j(t_0 + 1) \leq q'_j(t_0 + 1)$. Clearly, (1) remains valid after the addition of these jobs. □

The next lemma follow directly from Lemma 1 and was given before by Phillips et al. (Lemma 4.3 in [18])

Lemma 2. *Let instance I' be obtained from I by removing some of the jobs from I. Let σ and σ' be SRPT schedules for, respectively, I and I' and let n, n' be the number of jobs in I and I'. Then, for every $i \leq n'$,*

$$\mu_i(\sigma) \leq \mu_i(\sigma').$$

Proof. For each job j that is included in I but not in I' we add a job j to I' with $r'_j = r_j$ and $p'_j = \infty$ (or some large enough number). In the SRPT schedule for the extended instance, the added jobs will complete last and the other jobs are scheduled as in σ'. Now the lemma follows directly from Lemma 1 with $\delta_j = 0$ for all j. (N.B. Phillips et al. [18] use the same argument. However, we do need the stronger version of Lemma 1 with arbitrary $\delta_j \geq 0$ to prove Lemma 4.) □

An advantage of unweighted completion times over weighted completion times is that we can use a volume argument. For example, in any feasible schedule, the sum of the last m completion times is bounded from below by the sum of all processing times. To approximate the sum of the last m completion times we may compare the total volume that SRPT has done until a moment t with the volume that could have been done by any other schedule. This backlog argument enables us to bound the sum of the last m completion times as we do in Lemma 4.

Given schedule σ, let $V_t(\sigma)$ be the volume processed until time t. Call a schedule *busy* if at any moment t either all machines are busy or all jobs available at time t are being processed. Clearly, any SRPT schedule is busy. The next lemma gives an upper bound on the volume that a busy schedule may do less than any other schedule. Figure 2 shows that the lemma is tight for $m = 2$. (Basically the same lemma is given in [12] and was also found by the authors of [5]).

Lemma 3. *Let σ be a busy schedule and σ^* be any feasible schedule, both for the same instance I on m machines. Then, at any moment t*

$$V_t(\sigma^*) - V_t(\sigma) \leq mt/4.$$

Proof. A complete proof is given in [25]. Here we give a short proof that $V_t(\sigma^*) - V_t(\sigma) \leq mt/2$. Using this bound it follows that *SRPT* is at most 3/2-competitive. Fix an arbitrary time t and, for any j, let q_j and q_j^* be the number of units processed of j until time t in, respectively, σ and σ^*. Then $V_t(\sigma^*) - V_t(\sigma) = \sum_j (q_j^* - q_j)$. If $q_j^* - q_j \geq 0$ there are at least $q_j^* - q_j$ slots where j is processed in σ^* but not in σ. For each such slot mark the position (i.e. slot plus machine) in σ^*. Note that σ must process some other job at this position. Doing this for all jobs we see that the volume that σ processes before t is at least $\sum_j (q_j^* - q_j)$. Hence, $V_t(\sigma^*) - V_t(\sigma) \leq V_t(\sigma)$, which implies

$$2(V_t(\sigma^*) - V_t(\sigma)) \leq (V_t(\sigma^*) - V_t(\sigma)) + V_t(\sigma) = V_t(\sigma^*) \leq mt.$$

□

Lemma 4. *Given an instance I with $n \geq m$ jobs, let τ be its SRPT schedule and ρ be an arbitrary feasible schedule for I. Then,*

$$\sum_{i=n-m+1}^{n} \mu_i(\tau) \leq \frac{5}{4} \sum_{i=n-m+1}^{n} \mu_i(\rho).$$

Proof. Let $t = \mu_{n-m}(\rho)$. We change the instance I into I' as follows such that no job is released after time t in the new instance. Every job j with $r_j \geq t+1$ gets release time $r_j' = t$ and processing time $p_j + r_j - t$. Let τ' be an SRPT schedule for I'. Then, by Lemma 1 we have

$$\mu_i(\tau) \leq \mu_i(\tau'), \text{ for any } i \in \{1, 2, \ldots, n\}. \quad (2)$$

On the other hand, we can change ρ into a feasible schedule ρ' for I' without changing any of the completion times since at most m jobs are processed after time t in ρ. Hence, we may assume

$$\mu_i(\rho) = \mu_i(\rho'), \text{ for any } i \in \{1, 2, \ldots, n\}. \quad (3)$$

Let $W_t(\tau')$ and $W_t(\rho')$ be the total remaining processing time at time t in, respectively, τ' and ρ'. Since the last m jobs complete at time t or later in ρ' we have

$$\sum_{i=n-m+1}^{n} \mu_i(\rho') \geq mt + W_t(\rho'). \tag{4}$$

Since no jobs are released after t, the SRPT schedule satisfies

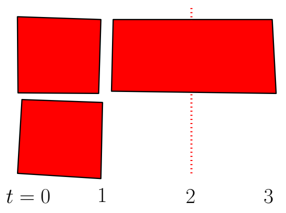


Fig. 2. A tight example for Lemma 3. Take $m = 2$ and two jobs of length 1 and one job of length 2. All are released at time 0. It is possible to complete the jobs by time 2. The remaining volume at time $t = 2$ in the SRPT schedule is $1 = mt/4$.

$$\sum_{i=n-m+1}^{n} \mu_i(\tau') \leq mt + W_t(\tau'). \tag{5}$$

(Equality holds if τ' completes at least m jobs at time t or later than t.) By Lemma 3, $W_t(\tau') - W_t(\rho') = V_t(\rho') - V_t(\tau') \leq mt/4$. This combined with (4) and (5) gives

$$\begin{aligned}\sum_{i=n-m+1}^{n} \mu_i(\tau') &\leq mt + W_t(\tau') \\ &\leq \frac{5}{4}mt + W_t(\rho') \\ &\leq \frac{5}{4}(mt + W_t(\rho')) \;\leq\; \frac{5}{4}\sum_{i=n-m+1}^{n} \mu_i(\rho').\end{aligned}$$

Equations (2) and (3) complete the proof. □

Theorem 1. *SRPT is* $5/4$*-competitive for minimizing total completion time on identical machines.*

Proof. Let φ be an optimal schedule. Take any $n' \leq n$ and let J' be the set of the first n' jobs completed in φ. Consider an SRPT schedule σ' for J'. By Lemma 2 we know that

$$\mu_i(\sigma) \leq \mu_i(\sigma') \text{ for all } i \leq |J'|. \tag{6}$$

We distinguish between the cases $n' \leq m$ and $n' \geq m$. In the first case we have $\mu_i(\sigma') \leq \mu_i(\varphi)$ since σ' starts each job at its release time and processes it without preemption. Combining this with (6) we get that

$$\mu_i(\sigma) \leq \mu_i(\varphi) \text{ for all } i \leq n'. \tag{7}$$

Now assume $n' \geq m$ and let φ' be the schedule φ restricted to jobs of J'. By definition,

$$\mu_i(\varphi') = \mu_i(\varphi) \text{ for all } i \leq |J'|. \tag{8}$$

We apply Lemma 4 with $\tau = \sigma'$ and $\rho = \varphi'$.

$$\sum_{i=n'-m+1}^{n'} \mu_i(\sigma') \leq \frac{5}{4} \sum_{i=n'-m+1}^{n'} \mu_i(\varphi'). \tag{9}$$

Using (6) and (8) we conclude that

$$\sum_{i=n'-m+1}^{n'} \mu_i(\sigma) \leq \frac{5}{4} \sum_{i=n'-m+1}^{n'} \mu_i(\varphi). \tag{10}$$

Hence, we see from (7) and (10) that the theorem follows by partitioning the completion times in groups of size m. The first group may be smaller. □

2.1 More Properties of SRPT

Given Lemmas 1 and 2 one might believe that a similar statement holds with respect to release times. However, it is not true that completion times do not decrease if release times are increased. In the example of Figure 1, SRPT will produce an optimal schedule if we change the release time of one small job from 0 to 1. The same example shows that SRPT may not be optimal even if no job is preempted. Finally, it is also not true that SRPT is optimal if it contains no idle time. This can be seen if we add two long jobs to example of Figure 1. This will not change the schedule of the other jobs and the sum of the completion times of the two long jobs is the same for SRPT and the optimal schedule. We conjecture that an SRPT schedule is optimal if it is non-preemptive and has no idle time.

3 Weighted Jobs

The SRPT algorithm has a natural generalization to the case where jobs have given weights. Unfortunately, our proof does not carry over to this case. A common approach in the analysis of the weighted average completion time is to use the *mean busy time* of a job which is defined as the average point in time that a job is processed. Given a schedule σ let $Z(\sigma)$ be the sum of weighted completion times and $Z^R(\sigma)$ be the sum of weighted mean busy times. On a single machine, the average (or total) weighted mean busy time is minimized by scheduling jobs preemptively in order of highest ratio of w_j/p_j [8]. This is called the preemptive *weighted shortest processing time* (WSPT) schedule. The WSPT-schedule is not unique but its total mean busy time is. Now consider a *fast single machine* that runs each job m times faster, i.e., job j has release time r_j and processing time p_j/m. For a given instance I, let $\sigma_m(I)$ be its preemptive WSPT-schedule on the fast single machine. The following inequality is a well-known lower bound on the optimal value of a preemptive and non-preemptive schedule [4,22].

$$Z^R(\sigma_m(I)) + \frac{1}{2}\sum_j w_j p_j \leq \textsc{Opt}(I). \tag{11}$$

Our algorithm uses the same two steps as the algorithms by Schulz and Skutella [22] and Correa and Wagner [6]: First, the jobs are scheduled on the fast single machine and then, as soon as an α-fraction of a job is processed, a job is placed as early as possible on one of the parallel machines. The algorithm in [22] uses random values of α and a random assignment to machines. The deterministic algorithm of [6] optimizes over α and simply takes the first available machine for each job. Our algorithm differs at three points: First, we take a fast single machine schedule of a modified instance I' instead of I. Second, we do not apply preemptive WSPT but use non-preemptive WSPT instead. Third, we simply take $\alpha = 0$ for each job. The behavior of our algorithm depends on the input I and a real number $\epsilon > 0$.

Theorem 2. *With $\epsilon = 1/\sqrt{m}$, algorithm* $\textsc{Online}(\epsilon)$ *is δ_m-competitive for minimizing total weighted completion time, where $\delta_m = (1+1/\sqrt{m})^2(3e-2)/(2e-2)$. The ratio holds for preemptive and non-preemptive scheduling on m identical parallel machines.*

We denote the start and completion time of job j in the fast machine ρ_m by, respectively, s_j and c_j and in the parallel machine schedule ρ by S_j and C_j. First, we prove that the optimal value does not change much by the modification made in step (i).

Lemma 5. $\textsc{Opt}(I') \leq (1+\epsilon)\textsc{Opt}(I)$.

Proof. Let σ^* be an optimal schedule for I and for any job j let C_j^* be the completion time of j in σ^*. We stretch the schedule by a factor $1+\epsilon$ such that each job j completes at time $(1+\epsilon)C_j^*$ and starts at time

$$(1+\epsilon)C_j^* - p_j \geq (1+\epsilon)(r_j + p_j) - p_j = (1+\epsilon)r_j + \epsilon p_j \geq r_j'.$$

We see that the schedule is feasible for I' and its value is exactly $1+\epsilon$ times the optimal value of I. □

Since we apply *non-preemptive* WSPT, the schedule ρ_m derived in step (ii) will in general not be the same as the fast single machine schedule $\sigma(I')$, which is derived by *preemptive* WSPT. Hence, we cannot use inequality (11) directly. We define a new instance I'' such that ρ_m is the fast machine schedule of I''. We shall prove this in Lemma 7 but first we introduce I'' and bound its optimal value like we did in the previous lemma. Let $I'' = \{(p_j'', w_j'', r_j'') \mid j = 1 \ldots n\}$ with $p_j'' = p_j, w_j'' = w_j$ and $r_j'' = \min\{\gamma_\epsilon r_j', s_j\}$, where $\gamma_\epsilon = 1 + 1/(\epsilon m)$.

Lemma 6. $\textsc{Opt}(I'') \leq (1 + 1/(\epsilon m))\textsc{Opt}(I')$.

Proof. The proof is similar to that of Lemma 5. Let σ' be an optimal schedule for I' and C'_j the completion time of j in σ'. We stretch the schedule by a factor γ_ϵ such that each job j completes at time $\gamma_\epsilon C'_j$ and starts at time

$$\gamma_\epsilon C'_j - p_j \geq \gamma_\epsilon (r'_j + p_j) - p_j = \gamma_\epsilon r'_j + (\gamma_\epsilon - 1)p_j \geq \gamma_\epsilon r'_j \geq r''_j.$$

We see that the schedule is feasible for I'' and its value is exactly γ_ϵ times the optimal value of I'. □

Algorithm Online(ϵ):

Input: Instance $I = \{(p_j, w_j, r_j) \mid j = 1 \ldots n\}$.

(i) Let $I' = \{(p'_j, w'_j, r'_j) \mid j = 1 \ldots n\}$ with $p'_j = p_j, w'_j = w_j$ and $r'_j = r_j + \epsilon p_j$.
(ii) Apply *non-preemptive* WSPT to I' on the fast single machine. Let ρ_m be this schedule and let s_j be the start time of job j in ρ_m.
(iii) Each job j is placed at time s_j on one of the parallel machines as early as possible (but not before s_j). Let ρ be the final schedule.

Clearly, $\text{Opt}(I) \leq \text{Opt}(I'')$ since we only shift release times forward. Combining Lemmas 5 and 6 we see that $\text{Opt}(I'') \leq (1 + 1/(\epsilon m))(1 + \epsilon)\text{Opt}(I)$. Choosing $\epsilon = 1/\sqrt{m}$ we obtain the following corollary.

Corollary 1

$$\text{Opt}(I) \leq \text{Opt}(I'') \leq \left(1 + \frac{1}{\sqrt{m}}\right)^2 \text{Opt}(I). \tag{12}$$

If we want to prove a bound on the competitive ratio of our algorithm only for large values of m, then we may just as well compare our schedule with the optimal schedule of I'' instead of I since $\text{Opt}(I'')/\text{Opt}(I) \to 1$ for $m \to \infty$. The next lemma states that the total mean busy time of ρ_m equals the total mean busy time of the preemptive WSPT-schedule of I'' on the single machine.

Lemma 7. $Z^R(\rho_m) = Z^R(\sigma(I''))$.

Proof. We show that schedule ρ_m is a preemptive WSPT schedule for I''. First, ρ_m is a feasible schedule for the fast single machine relaxation of I'' since, by definition, $r''_j \leq s_j$. Next we use $s_j \geq r'_j \geq \epsilon p_j$.

$$\begin{aligned} c_j/s_j &= (s_j + p_j/m)/s_j \qquad (13) \\ &= 1 + p_j/(m s_j) \\ &\leq 1 + p_j/(m\epsilon p_j) \\ &= 1 + 1/\sqrt{m}. \end{aligned}$$

Assume that at moment t, job j is being processed in ρ_m and job k is available in I'', i.e., $r''_k \leq t$. Denote $\gamma = 1 + 1/\sqrt{m}$, then by definition $r''_k = \min\{\gamma r'_k, s_k\}$. Since also $r''_k \leq t < s_k$ we must have $r''_k = \gamma r'_k$. Using (13) we get

$$r'_k = r''_k/\gamma \leq t/\gamma < c_j/\gamma \leq (1 + 1/\sqrt{m})s_j/\gamma = s_j.$$

We see that job k was available at the time we started job j in step (ii). Hence, we must have $w_k/p_k \leq w_j/p_j$. □

We apply the lower bound of (11) to instance I''.

$$Z^R(\sigma_m(I'')) + \frac{1}{2}\sum_j w_j p_j \leq \text{OPT}(I''). \tag{14}$$

Combining this with Corollary 1 and Lemma 7, we finally get a useful lower bound on the optimal solution.

Corollary 2

$$Z^R(\rho_m) + \frac{1}{2}\sum_j w_j p_j \leq \left(1 + \frac{1}{\sqrt{m}}\right)^2 \text{OPT}(I).$$

The lower bound of Corollary 2 together with the obvious lower bound $\text{OPT}(I) \geq \sum_j w_j p_j$ results in the following lemma.

Lemma 8. *Let $1 \leq \alpha \leq 2$. If $S_j \leq \alpha s_j$ for every job j, then*

$$\sum_j w_j C_j \leq \left(1 + \frac{\alpha}{2}\right)\left(1 + \frac{1}{\sqrt{m}}\right)^2 \text{OPT}(I).$$

Proof. Let b_j be the mean busy time of j in ρ_m, then $s_j = b_j - p_j/(2m) < b_j$.

$$\begin{aligned} C_j &= S_j + p_j \\ &\leq \alpha s_j + p_j \\ &< \alpha b_j + p_j \\ &= \alpha(b_j + p_j/2) + (1 - \alpha/2)p_j \end{aligned}$$

Next, we add weights and take the sum over all jobs.

$$\textstyle\sum_j w_j C_j \leq \alpha\left(Z^R(\rho_m) + \frac{1}{2}\sum_j w_j p_j\right) + (1 - \alpha/2)\sum_j w_j p_j$$

Now we use Corollary 2 and use that $\text{OPT}(I'') \geq \text{OPT}(I) \geq \sum_j w_j p_j$. For any $\alpha \leq 2$ we have

$$\begin{aligned} \textstyle\sum_j w_j C_j &\leq \alpha(1 + 1/\sqrt{m})^2 \text{OPT}(I) + (1 - \alpha/2)\text{OPT}(I) \\ &\leq (1 + \alpha/2)(1 + 1/\sqrt{m})^2 \text{OPT}(I). \end{aligned}$$

□

First we give a short proof that $\alpha \leq 2$. This shows that the competitive ratio is at most $2 + o(m)$.

Lemma 9. *$S_j \leq 2s_j$ for any job j.*

Proof. Consider an arbitrary job j. At time s_j, the total processing time of jobs k with $s_k < s_j$ is at most ms_j. Since these are the only jobs processed on the parallel machines between time s_j and S_j we have $ms_j \geq m(S_j - s_j)$. Hence, $S_j \leq 2s_j$. $\square$

The bound of the next lemma is stronger. The proof is given in the technical report [25]. Lemma 8 tells us that the competitive ratio is at most $1 + \frac{e}{2(e-1)} \approx 1.791$ in the limit.

Lemma 10. *$S_j \leq \frac{e}{e-1} s_j$ and this bound is tight.*

3.1 Removing the $o(m)$

We can easily get rid of the $o(m)$ term at the cost of a higher ratio. Correa and Wagner [6] give a randomized α_m-competitive algorithm for the preemptive problem and a β_m-competitive algorithm for the non-preemptive version, where $2-1/m = \alpha_m < \beta_m < 2$ for $m \geq 3$. Let δ_m be our ratio as defined in Theorem 2. Then $2-1/m > \delta_m$ for $m \geq 320$. Hence, we get a randomized $2-1/320 < 1.997$-competitive for the preemptive version when we apply our algorithm for $m \geq 320$ and the α_m-competitive for $m < 320$. The ratio for the non-preemptive version is even closer to 2 (but strictly less than 2).

4 Conclusion

We have shown that approximation ratios less than 2 can be obtained for parallel machines by simple and efficient online algorithms. The lower bounds indicate that competitive ratios close to 1 are possible for randomized algorithms, especially when preemption is allowed. Our analysis for SRPT is tight and it seems that a substantially different proof is needed to get below 1.25. Already, the gap with the lower bound, 1.105, is quite small. Muthukrishnan et al.[17] show that SRPT is at most 14 competitive w.r.t. the average stretch of jobs. Possibly, our result can reduce this ratio substantially. The analysis for algorithm Online is not tight and a slight modification of the algorithm and analysis may give a ratio $e/(e-1) + o(m) \approx 1.58 + o(m)$. Moreover, the analysis is not parameterized by m. A refined analysis will reduce the $o(m)$ for small values of m.

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: FOCS '99, pp. 32–44 (1999)
2. Anderson, E.J., Potts, C.N.: Online scheduling of a single machine to minimize total weighted completion time. Math. Oper. Res. 29, 686–697 (2004)

3. Chekuri, C., Motwani, R., Natarajan, B., Stein, C.: Approximation techniques for average completion time scheduling. SIAM Journal on Computing 31, 146–166 (2001)
4. Chou, M.C., Queyranne, M., Simchi-Levi, D.: The asymptotic performance ratio of an on-line algorithm for uniform parallel machine scheduling with release dates. Mathematical Programming 106, 137–157 (2006)
5. Chung, C., Nonner, T., Souza, A.: SRPT is 1.86-competitive for completion time scheduling. In: Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (Austin, Texas), pp. 1373–1388 (2010)
6. Correa, J.R., Wagner, M.R.: LP-based online scheduling: from single to parallel machines. Mathematical Programming 119, 109–136 (2009)
7. Epstein, L., van Stee, R.: Lower bounds for on-line single-machine scheduling. Theoretical Computer Science 299, 439–450 (2003)
8. Goemans, M.X.: Improved approximation algorithms for scheduling with release dates. In: Proc. 8th Symp. on Discrete Algorithms, New Orleans, Louisiana, United States, pp. 591–598 (1997)
9. Goemans, M.X., Queyranne, M., Schulz, A.S., Skutella, M., Wang, Y.: Single machine scheduling with release dates. SIAM Journal on Discrete Mathematics 15, 165–192 (2002)
10. Hall, L.A., Schulz, A.S., Shmoys, D.B., Wein, J.: Scheduling to minimize average completion time: Off-line and on-line approximation algorithms. Mathematics of Operations Research 22, 513–544 (1997)
11. Hoogeveen, J.A., Vestjens, A.P.A.: Optimal on-line algorithms for single-machine scheduling. In: Cunningham, W.H., Queyranne, M., McCormick, S.T. (eds.) IPCO 1996. LNCS, vol. 1084, pp. 404–414. Springer, Heidelberg (1996)
12. Hussein, M.E., Schwiegelshohn, U.: Utilization of nonclairvoyant online schedules. Theoretical Computer Science 362, 238–247 (2006)
13. Jalliet, P., Wagner, R.M.: Almost sure asymptotic optimality for online routing and machine scheduling problems. Networks 55, 2–12 (2009)
14. Liu, P., Lu, X.: On-line scheduling of parallel machines to minimize total completion times. Computers and Operations Research 36, 2647–2652 (2009)
15. Lu, X., Sitters, R.A., Stougie, L.: A class of on-line scheduling algorithms to minimize total completion time. Operations Research Letters 31, 232–236 (2002)
16. Megow, N., Schulz, A.S.: On-line scheduling to minimize average completion time revisited. Operations Research Letters 32, 485–490 (2004)
17. Muthukrishnan, S., Rajaraman, R., Shaheen, A., Gehrke, J.E.: Online scheduling to minimize average stretch. SIAM J. Comput. 34, 433–452 (2005)
18. Phillips, C., Stein, C., Wein, J.: Minimizing average completion time in the presence of release dates, networks and matroids; sequencing and scheduling. Mathematical Programming 82, 199–223 (1998)
19. Queyranne, M.: On the Anderson-Potts single machine on-line scheduling algorithm (2001) (unpublished manuscript)
20. Schrage, L.: A proof of the optimality of the shortest remaining processing time discipline. Operations Research 16(3), 687–690 (1968)
21. Schulz, A.S., Skutella, M.: The power of α-points in single machine scheduling. Journal of Scheduling 5, 121–133 (2002)
22. Schulz, A.S., Skutella, M.: Scheduling unrelated machines by randomized rounding. SIAM Journal on Discrete Mathematics 15, 450–469 (2002)
23. Seiden, S.: A guessing game and randomized online algorithms. In: Proceedings of the 32nd ACM Symposium on Theory of Computing, pp. 592–601 (2000)

24. Sitters, R.A.: Complexity and approximation in routing and scheduling, Ph.D. thesis, Eindhoven Universtity of Technology, the Netherlands (2004)
25. Sitters, R.A.: Efficient algorithms for average completion time scheduling, Tech. Report 2009-58, FEWEB research memorandum, Free University Amsterdam (2009)
26. Stougie, L., Vestjens, A.P.A.: Randomized on-line scheduling: How low can't you go? Operations Research Letters 30, 89–96 (2002)
27. Vestjens, A.P.A.: On-line machine scheduling, Ph.D. thesis, Department of Mathematics and Computing Science, Technische Universiteit Eindhoven, Eindhoven, the Netherlands (1997)

Experiments with Two Row Tableau Cuts

Santanu S. Dey[1], Andrea Lodi[2],
Andrea Tramontani[2], and Laurence A. Wolsey[3]

[1] H. Milton Stewart School of Industrial and Systems Engineering,
Georgia Institute of Technology, USA
santanu.dey@isye.gatech.edu
[2] DEIS, Università di Bologna, Italy
{andrea.lodi,andrea.tramontani}@unibo.it
[3] CORE, Catholic University of Louvain, Belgium
laurence.wolsey@uclouvain.be

Abstract. Following the flurry of recent theoretical work on cutting planes from two row mixed integer group relaxations of an LP tableau, we report on some computational tests to evaluate the effectiveness of two row cuts based on lattice-free (type 2) triangles having more than one integer point on one side. A heuristic procedure to generate such triangles is presented, and then the coefficients of the integer variables are tightened by lifting. As a first step in testing the effectiveness of the triangle cuts, we make comparisons between the gap closed using Gomory mixed integer cuts for one round and the gap closed in one round using all the triangles generated by our heuristic. Our tests are carried out on different classes of randomly generated instances designed to represent different models in the literature by varying the number of integer non-basic variables, bounds and non-negativity constraints.

1 Introduction

Addition of cutting planes to strengthen linear programming relaxations has proven to be an indispensable tool in solving mixed integer programs (MIPs). For general MIPs, Gomory mixed integer cuts [19] (GMIC) and mixed integer rounding inequalities [25] have been found to be important cutting planes. Construction of most cutting planes can be viewed as deriving valid inequalities for some relaxation of MIPs. Seen from this perspective, the above mentioned cutting planes can be obtained by deriving valid inequalities for single constraint relaxations of MIPs. Therefore a natural extension is to consider inequalities that are valid for multiple constraint relaxations of MIPs. This approach has been pursued recently with the aim of obtaining new cutting planes that could possibly improve on the gap closed by GMICs.

A two constraint relaxation of a simplex tableau introduced in [5] is

$$\{(z, y) \in \mathbb{Z}^2 \times \mathbb{R}^n_+ \mid z = f + \sum_{j=1}^{n} r^j y_j\} \ , \tag{1}$$

where $f \in \mathbb{Q}^2 \setminus \{(0,0)\}, r^j \in \mathbb{Q}^2 \; \forall j \in \{1,...,n\}$. Various variants of (1) have been studied; see for example [3,4,9,10,11,12,13,15,16,26] and [14] for a recent survey on the topic. The relaxation (1) can be obtained in three steps. First select two rows of a simplex tableau corresponding to integer basic variables currently at fractional value(s). Then relax the non-negativity of the integer basic variables. Finally relax the non-basic integer variables to be continuous variables. Two appealing features of this relaxation are the complete characterization of all facet-defining inequalities using the so-called lattice-free convex sets and the advantage of obtaining the strongest possible coefficients for continuous variables in the resulting cutting planes for MIPs. Note that the importance of obtaining cutting planes where the continuous variables have strong coefficients has been emphasized by previous theoretical and computational studies; see for example [2,18].

Since the continuous variables receive strong coefficients, the focus next is to improve the coefficients of non-basic integer variables. One way to strengthen the coefficients of non-basic integer variables is to consider lifting them to obtain valid inequalities for

$$\{(z,y,x) \in \mathbb{Z}^2 \times \mathbb{R}^n_+ \times \mathbb{Z}^k_+ \,|\, z = f + \sum_{j=1}^{n} r^j y_j + \sum_{j=n+1}^{n+k} r^j x_j\} \;, \tag{2}$$

where all data is rational. One possible approach for lifting is by the use of the so-called ‘fill-in functions’ [21,23] or ‘monoidal strengthening’ [8]. Observe that such lifted inequalities do not provide the complete list of facet-defining inequalities of the convex hull of (2). In fact, (2) is a face of the mixed integer master group relaxation and a complete description of the convex hull of the master group relaxation is unknown. (See discussion in [20,22]).

The goal of this paper is to evaluate the quality of the two-row lifted cutting planes described above computationally. In particular, we would like to understand how to select a small subset of these cutting planes which are useful in practice, to discover what are the potential weaknesses of these cutting planes, and to evaluate how far these weaknesses can be attributed to the different relaxations or the lack of knowledge of the complete description of the convex hull of (2). We work with a specific subclass of facet-defining inequality of the convex hull of (1) and consider its lifted version. We attempt to answer the following specific question. How good are these cutting planes in the the presence of integer non-basic variables? Is there a cut off on the ratio of number of integer and continuous non-basic variables for these cutting planes to be useful? What is the strength of these two row cuts in the presence of multiple rows? Can the sparsity structure be used to select important two row relaxations? How important is the effect of non-negativity of the basic variables? Through experiments designed to answer each of these questions individually by keeping all other parameters constant, we hope to gain insight into the strength of relaxations of m-row sets consisting of the intersection of the convex hulls of two-row relaxations, and possibly obtain guidelines for selecting the most useful two-row lifted cutting planes.

The outline of this paper is the following. In Section 2, we summarize the relevant results regarding valid inequalities of (1) and the lifting of these inequalities.

In Section 3, we briefly discuss the algorithmic aspects of generating the lifted two row inequalities. In Section 4, we describe various experiments conducted and the results. We conclude in Section 5.

2 Basics

We begin with a definition of maximal lattice-free convex sets and describe these sets in $\mathbb{R}^2$.

Definition 1 ([24]). *A set $M \subseteq \mathbb{R}^m$ is called lattice-free if $\text{int}(M) \cap \mathbb{Z}^m = \emptyset$. A lattice-free convex set M is maximal if there exists no lattice-free convex set $M' \neq M$ such that $M \subsetneq M'$.*

Proposition 1. *Let M be a full-dimensional maximal lattice-free convex set in $\mathbb{R}^2$. Then M is one of the following:*

1. *A split set $\{(x_1, x_2) \,|\, b \leq a_1x_1 + a_2x_2 \leq b + 1\}$ where a_1 and a_2 are coprime integers and b is an integer,*
2. *A triangle which is one of the following:*
   - *(a) A type 1 triangle: triangle with integral vertices and exactly one integral point in the relative interior of each edge,*
   - *(b) A type 2 triangle: triangle with at least one fractional vertex v, exactly one integral point in the relative interior of the two edges incident to v and at least two integral points on the third edge,*
   - *(c) A type 3 triangle: triangle with exactly three integral points on the boundary, one in the relative interior of each edge.*
3. *A quadrilateral containing exactly one integral point in the relative interior of each of its edges.*

A maximal lattice-free convex set M containing f in its interior can be used to derive the intersection cut $\sum_{j=1}^{n} \pi(r^j)y_j \geq 1$ ([7]) for (1) where the coefficients are obtained as:

$$\pi(r^j) = \begin{cases} \lambda & \text{if } \exists \lambda > 0 \text{ s.t. } f + \frac{1}{\lambda}r^j \in \text{boundary}(M) \\ 0 & \text{if } r^j \text{ belongs to the recession cone of } M. \end{cases} \tag{3}$$

All non-trivial valid inequalities for (1) are intersection cuts and can be derived from maximal lattice-free convex sets using (3). We refer the readers to [5,13] for a complete characterization of facet-defining inequalities of the convex hull of (1).

Next consider the strengthening of the coefficients for integer variables. It is possible to obtain the valid inequality $\sum_{j=1}^{n} \pi(r^j)y_j + \sum_{j=n+1}^{n+k} \phi(r^j)x_j \geq 1$ for (2) where $\phi(r^j) = \inf_{u \in \mathbb{Z}^2}\{\pi(r^j + u)\}$ ([21,23],[8]). For an infinite version of the relaxation (2), it has been shown [15] that this strengthening yields extreme inequalities if π was obtained using (3) and M is a type 1 or type 2 triangle. Moreover in the case in which M is a type 1 and type 2 triangle [15], the function ϕ can be evaluated as

$$\phi(r^j) = \min_{u \in \mathbb{Z}^2}\{\pi(r^j + u) \,|\, r^j + u + f \in M\} \ .$$

3 Generating Type 2 Triangles

In this paper, we focus on the inequalities that are generated using maximal lattice-free triangles of type 2.

If the exact rational representation of the data in (1) is known, then it is possible to efficiently enumerate all the vertices of (1) by the use of the Euclidean algorithm. Once all the vertices are known, by setting up the polar, all facet-defining inequalities can be obtained. Since the exact rational representation may not typically be known, we outline a different strategy to construct maximal lattice-free triangles of type 2.

Maximal lattice-free triangles of type 2 that are facet-defining inequalities have the following interpretation.

1. Construct a facet-defining inequality of the form $\alpha_{j_1}y_{j_1} + \alpha_{j_2}y_{j_2} \geq 1$ for the set $Y^2 := \{(z, y) \in \mathbb{Z}^2 \times \mathbb{R}^2_+ \,|\, z = r^{j_1}y_{j_1} + r^{j_2}y_{j_2}\}$ where $\alpha_{j_1}, \alpha_{j_2} > 0$. The set $\text{conv}\{f, f + \frac{r^{j_1}}{\alpha_{j_1}}, f + \frac{r^{j_2}}{\alpha_{j_2}}\} \subseteq \mathbb{R}^2$ is lattice-free and the line segment between $f + \frac{r^{j_1}}{\alpha_{j_1}}$ and $f + \frac{r^{j_2}}{\alpha_{j_2}}$ contains at least two integer points.
2. Lift a third continuous variable y_{j_3} to obtain the inequality $\alpha_{j_1}y_{j_1} + \alpha_{j_2}y_{j_2} + \alpha_{j_3}y_{j_3} \geq 1$. The set $\text{conv}\{f + \frac{r^{j_1}}{\alpha_{j_1}}, f + \frac{r^{j_2}}{\alpha_{j_2}}, f + \frac{r^{j_3}}{\alpha_{j_3}}\} \subseteq \mathbb{R}^2$ is lattice-free and at least one of the sides of the triangle (other than the side between $f + \frac{r^{j_1}}{\alpha_{j_1}}$ and $f + \frac{r^{j_2}}{\alpha_{j_2}}$) contains one integer point.

Based on the above interpretations, we now give details of our implementation for generating type 2 triangles.

1. Given three variables y_{j_1}, y_{j_2} and y_{j_3} such that $cone(r^{j_1}, r^{j_2}, r^{j_3}) = \mathbb{R}^2$, we first attempt to generate one facet $\alpha_{j_1}y_{j_1} + \alpha_{j_2}y_{j_2} \geq 1$ of $\text{conv}(Y^2)$ that is 'most different' from a GMIC. Since the line segment between $f + \frac{r^{j_1}}{\alpha_{j_1}}$ and $f + \frac{r^{j_2}}{\alpha_{j_2}}$ contains at least two integer points, we divide this step into two sub steps corresponding to finding these two integer points. At the end of these two steps, either the integer points are discovered or it is verified that no such integer points exist. For simplicity let $1 := j_1, 2 := j_2$ and $3 := j_3$.
   (a) Finding the first integer point: Let $c := (c_1, c_2)$ be a strict convex combination of r^1 and r^2. Solve $\min\{c^Tz \,|\, \lambda_1 r^1 + \lambda_2 r^2 = -f + z,\ \lambda_1, \lambda_2 \geq 0, z \in \mathbb{Z}^2\}$. Let the optimal objective value be w^0 and the optimal solution be $(\bar{\lambda}^0_1, \bar{\lambda}^0_2, \bar{z}^0)$. The point $\bar{z}^0$ is tentatively one of the integer points in the interior of the side of the triangle inequality containing multiple integer points.
   (b) Finding the second integer point: Let $v = \bar{\lambda}^0_1 r^1 + \bar{\lambda}^0_2 r^2$ (i.e., $v = \bar{z}^0 - f$). First an integer point belonging to the set $\{u \,|\, u = f + \mu_1 r^1 + \mu_2 r^2, \mu_1, \mu_2 \geq 0\}$ different from $\bar{z}^0$ is found. Let $r^{\text{new}} = v + \theta r^1$ for a suitable $\theta > 0$. For some c such that $c^T r^1 \geq 0$, $c^T r^{\text{new}} \geq 0$, solve $\min\{c^Tz \,|\, \lambda_1 r^1 + \lambda_2 r^{\text{new}} = -f + z, \lambda_1, \lambda_2 \geq 0, z \in \mathbb{Z}^2\}$. Let the optimal solution be $(\bar{\lambda}^1_1, \bar{\lambda}^1_2, \bar{z}^1)$.

Now we verify if the integer point z^1 is suitable to be the second integer point on the side of the triangle. If not, we iteratively update this point. Let $e^1 \in \mathbb{R}^2$ be such that $(e^1)^T \bar{z}^0 = {e^1}^T \bar{z}^1 = 1$ and ${e^1}^T f < 1$. This is equivalent to verifying if ${e^1}^T x \geq 1$ is a valid inequality for Y^2. Repeat the following step: Solve

$$\min\{{e^i}^T z \,|\, \lambda_1 r^1 + \lambda_2 r^2 = -f + z, \lambda_1, \lambda_2 \geq 0, z \in \mathbb{Z}^2\} \ .$$

Let the optimal objective value be w^{i+1} and the optimal solution be $(\bar{\lambda}_1^{i+1}, \bar{\lambda}_2^{i+1}, \bar{z}^{i+1})$. Let $e^{i+1} \in \mathbb{R}^2$ be such that ${e^{i+1}}^T \bar{z}^0 = {e^{i+1}}^T \bar{z}^{i+1} = 1$ and ${e^{i+1}}^T f < 1$. If $w^{i+1} = 1$, then denote the point $\bar{z}^{i+1}$ as $\bar{z}^{i_0}$ and stop. If $w^{i+1} < 1$, then set $i \leftarrow i + 1$ and repeat this step.

One final check is needed to verify that $\bar{z}^{i_0}$ is indeed a vertex of the conv(Y^2). This check becomes relevant when conv(Y^2) has only one vertex. Verify that $(\bar{z}^{i_0} - \bar{z}^0)$ and $(-\bar{z}^{i_0} + \bar{z}^0)$ do not belong to the cone formed by r^1 and r^2.

2. Lifting the third continuous variable: From the previous step we obtain two integer points $\bar{z}^0$ and $\bar{z}^{i_0}$ that are tight for the inequality we will obtain. Let z^{v_1} and z^{v_2} be the two points obtained by extending the line segment passing through $\bar{z}^0$ and $\bar{z}^{i_0}$ and intersecting the two half lines $f + \mu_j r^j$, $\mu_j \geq 0$ $j \in \{1, 2\}$. The next step is to identify two other integer points which lie in the relative interior of the other two sides of the triangle. Let $(a, b) = \bar{z}^0 - \bar{z}^{i_0}$. Update a and b by dividing them by their greatest common divisor and let $p, q \in \mathbb{Z}$ such that $pa + qb = \pm 1$ and $(q, -p)^T r^3 > 0$. Then the two other integer points are of the form $\bar{z}^0 + (q, -p) + k(a, b)$ and $\bar{z}^0 + (q, -p) + (k + 1)(a, b)$ for some integer k. The integer k can be calculated as follows. There are two cases:
   (a) f lies in the set $\{(w_1, w_2) \,|\, \bar{z}_1^0(-b) + \bar{z}_2^0 a \leq w_1(-b) + w_2(a) \leq \bar{z}_1^0(-b) + \bar{z}_2^0 a + 1\}$. In this case solve $\bar{z}^0 + (q, -p) + \lambda(a, b) = f + r^3\mu, \mu \geq 0$ for λ and μ and set $k = \lfloor \lambda \rfloor$.
   (b) f lies in the set $\{(w_1, w_2) \,|\, w_1(-b) + w_2(a) \geq \bar{z}_1^0(-b) + \bar{z}_2^0 a + 1\}$. Then solve $(q, -p) + \lambda(a, b) = \mu(f - \bar{z}^0)$, $\mu \geq 0$ for λ and μ and set $k = \lfloor \lambda \rfloor$.

Denote $\bar{z}^0 + (q, -p) + k(a, b)$ as z^L and $\bar{z}^0 + (q, -p) + k(a, b)$ as z^R. Construct the triangle by joining the two vertices z^{v_1} and z^{v_2} obtained in the previous step to z^L and z^R, and extending these line segments until they intersect at the third vertex.

We observe here that the method described above is a heuristic and not an exact method for selecting best possible type 2 triangles. This is for at least two reasons. First, due to the method employed to lift the third continuous variable in step 2, the resulting cut may not always be facet-defining. More importantly for every choice of two continuous variable that correspond to the set Y^2 in step 1, we select only one inequality, whereas there are typically possible many candidates.

4 Computational Experiments

The aim of this computational section is to analyze the effectiveness of cuts associated with type 2 triangles. The most natural setting which is generally used in these cases is to run the standard benchmarks on the MIPLIB 2003 [1] with a cutting plane approach with and without the new family of cuts for a fixed number of iterations/rounds and look at the usual performance indicators: separation time, overall computing time, percentage gap closed, density of cuts, etc. This is the framework adopted in [17] in which multi-row cuts were tested for the first time.

However, in this particular context there are a number of issues that prevent such an approach being entirely satisfactory. The first of these issues is precisely the fact that the relaxation is no longer single-row as in all cutting planes from the literature so far. Indeed, independently of the answer yes/no about the effectiveness of the new cuts, one has to deal with a series of specific questions. Below we list some that appear important:

1. Is the 2-row relaxation stronger than the classical 1-row one?
   (a) Should one consider cuts from 2 rows *instead* of cuts from 1 row?
   (b) Are cuts from 2 rows “complementary” to cuts from 1 row, i.e., are the cuts sufficiently different to be used together?
   (c) Would several rounds of 1-row cuts give as much improvement as 2-row cuts (if any)?
2. The theory developed so far considers separating facets of the 2-row relaxation (1) and then lifting the integer variables. In other words, 2-row cuts are concerned mostly with the continuous component of the problem because all integer non-basic variables are considered afterwards. Is that an effective model?
3. Again on the theoretical side, model (1) assumes no bounds on the two integer basic variables and only non-negativity on the non-basic variables. What is the role played by bounds on the variables?
4. Although it is rather straightforward separating and adding GMICs from the tableau, one for each basic fractional variable defined to be integer (at least for a limited number of iterations), it is clearly not possible to add all cuts from type 2 triangles without effecting the performance of the cutting plane algorithm in various ways. Specifically, the number of rows explodes, the tableau becomes too dense, and overall the numerical stability of the linear programming engine deteriorates. Thus, we have to face a serious cut selection problem which is far from trivial at this stage.

To begin to answer some of the questions raised above we have decided to use randomly generated instances of which we can control the size and (partially) the structure. In particular, we used multidimensional knapsack instances obtained by the random generator of Atamturk [6], kindly provided by the author. In addition, we considered a single round of cuts and we did not apply any cut/triangle selection rule.

It is clear that an overall assessment of the effectiveness of multi-row cuts will be their use within MIP solvers on real-world problems. However, as it took around 40 years before GMICs were proven to be effective within a computational setting, we believe that investing time in understanding multi-row cuts step-by-step is worthwhile.

4.1 Computational Setting

By using the generator [6], we first obtained multidimensional knapsack instances having m rows, and ℓ integer non-negative variables. These instances are of the form

$$\begin{aligned} \max \sum_{j=1}^{\ell} p_j x_j & \\ \sum_{j=1}^{\ell} w_{ij} x_j \leq b_i,\ & i = 1, \ldots, m, \\ x_j \geq 0, \text{integer},\ & j = 1, \ldots, \ell, \end{aligned}$$

where all the data are strictly positive. After solving the continuous relaxation of these instances we slightly modified them so as to obtain three sets having characteristics suitable for our tests. Namely:

A. We relaxed the m basic variables to be free variables, we changed 10% of the other variables to be non-negative continuous variables and we removed the other variables.

   In this way, we obtain instances with only continuous variables (besides the basic ones) but without too many different directions/rays (not too many variables) so that the number of type 2 triangles is small (discussed later).

B. As for set A but the remaining 90% are kept as non-negative integer variables.

   This is now a real mixed integer program in which the 10% of continuous variables need to interact with the non-basic integer variables.

C. As for set B but the objective coefficients of the 10% of continuous variables are divided by 100.

   In this way, the importance of the integer variables is significantly increased.

For the sets above, we considered: (i) $m \in \{2, 5\}$, (ii) $\ell \in \{50, 100, 150\}$, (iii) only one round of cuts, either GMICs, or type 2 triangle cuts, or both, and (iv) we consider 30 instances for each pair (m, ℓ).

Concerning type 2 triangle cuts, we separate cuts from any pair or rows (only one pair for $m = 2$) in the following way. Given a set, say R, of suitable rays: for each pair of rays $r^1, r^2 \in R$ we select a third ray $r^3 \in R$ such that $cone(r^1, r^2, r^3) = \mathbb{R}^2$ and try to construct a type 2 triangle as described in Section 3. We then distinguish two cases for R:

(y): R contains the set of rays associated with continuous variables.

(y+x): We consider rays associated with both continuous and integer variables. However, note that the direction read from the tableau for an integer variable is not well-defined because such a variable will be lifted afterwards, see Section 1. Thus, for each integer variable we consider the direction obtained by applying the GMIC lifting to the variable. Precisely, let $r^j = (r_1^j, r_2^j)$ the ray associated with the integer variable x_j in the tableau. We construct the ray $\hat{r}^j$ where

$$\hat{r}_i^j = \begin{cases} r_i^j - \lfloor r_i^j \rfloor & \text{if } f_i + r_i^j - \lfloor r_i^j \rfloor \le 1 \\ r_i^j - \lfloor r_i^j \rfloor - 1 & \text{otherwise} \end{cases} \quad i = 1, 2 \ . \tag{4}$$

Of course, $\hat{r}^j$ is only one of the possible liftings of variable x_j, not necessarily the best one.

The experiments with $m = 2$ are aimed at analyzing the case in which the two constraint relaxation (1) is supposed to capture much of the structure of the original problem (see Section 4.2 below). Of course, real problems have generally (many) more constraints but in this context we limited the setting to $m = 5$ because we do not apply any cut selection mechanism, and otherwise the number of separated type 2 triangle cuts would have been too big.

The code has been implemented in C++, using IBM ILOG Cplex 10.0 as LP solver. In order to limit possible numerical issues in the cut generation, we adopted several tolerances and safeguards. In particular, concerning the GMIC generation, we did not generate a GMIC from a tableau row if the corresponding basic variable has a fractional part smaller (resp. greater) than 0.001 (resp. 0.999). Similarly, we generated cuts from a triangle only if the fractional point f is safely in its interior. More precisely, a triangle is discarded if the Euclidean distance of f from its boundary is smaller than 0.001. In addition, both for GMICs and triangle cuts, we discarded all the generated cuts with "dynamism" greater than 10^9, where the dynamism is computed as the ratio between the greatest and smallest nonzero absolute value of the cut coefficients.

Before going into the details of the numbers, it is important to note that the way used to derive the instances accomplishes a very important goal. The cuts generated are exactly the same for the three sets A, B and C (except for the fact that no integer rays are present in case A). Indeed, what changes is the impact of the cuts on the solution process whereas (i) the number of continuous variables is constant from A to C, (ii) the number of integer variables is the same from B to C, and (iii) the rays of all variables stay the same from A to C. The changes are instead due to the fact that (1) no integer variables are present in A and (2) the objective function coefficients of the continuous variables have been decreased from B to C.

4.2 2-Row Results

In this section we report the computational results for instances with $m = 2$ and all the three sets A, B and C described above. In particular, note that on problems with only two rows and no integer non-basic variables (set A), model (1) is not a relaxation but coincides with the problem itself. Thus, type 2 triangle cuts are supposed to be really strong and the experiments are aimed at asserting their real effect in practice. On the other hand, sets B and C allow one to evaluate the impact of non-basic integer variables on the strength of the two constraint relaxation (1).

Moreover, in order to better understand the impact of bounds on the generated instances, we also considered introducing bounds on instances of sets B and C in the following way. For the generic set Z (Z $\in \{B, C\}$):

Z.1: means basic binary, $y \geq 0$, $x \geq 0$;
Z.2: means basic free, $y \geq 0$, x binary;
Z.3: means basic free, $0 \leq y \leq 1$, $x \geq 0$.

Table 1 compares one round of Gomory Mixed Integer Cuts (GMIC in the table) with type 2 triangle cuts either in the setting (y) above – only rays associated with continuous variables, Tr.(y) in the table – or in the setting (x+y) – rays from both integer and continuous variables, Tr.(x+y) in the table. In particular, all entries are average values of the results over 90 instances and for each set, the first 5 columns report the characteristics of the instances in the set: namely, number of y variables (ny), number of x variables (nx), number of non-basic slacks that are nonzero in the MIP optimal solution (nz.s), number of non-basic y variables that are nonzero in the MIP optimal solution (nz.y), number of non-basic x variables that are nonzero in the MIP optimal solution (nz.x). Then, for each of the cut family, Table 1 reports three columns: percentage gap closed (%gap), number of generated cuts (ncuts) and number of cuts tight at the second optimal tableau, i.e., after reoptimization (ntight).

Table 1 does not report computing times for the separation. It is easy to see that separating type 2 triangle cuts is more time consuming than separating GMICs, mainly because of the procedure for finding the edge of the triangle containing more than one integral point (see Section 3). However, the separation time is quite reasonable: the average 2740.8 type 2 triangle cuts for instances of type B and C require a separation time of 8 CPU seconds on a workstation

Table 1. 2-row Instances

	Characteristics					GMIC			Tr.(y)			Tr.(y+x)		
Set	ny	nx	nz.s	nz.y	nz.x	%gap	ncuts	ntight	%gap	ncuts	ntight	%gap	ncuts	ntight
A	9.8	—	0.0	2.0	—	75.10	2.0	1.6	97.99	68.1	2.1	—	—	—
B	9.8	90.2	0.0	2.0	0.5	65.33	2.0	1.5	82.01	68.1	2.1	91.13	2,740.8	2.5
C	9.8	90.2	0.3	0.1	2.8	16.27	2.0	1.3	25.80	68.1	1.9	37.17	2,740.8	3.0
B	9.8	90.2	0.0	2.0	0.5	65.33	2.0	1.5	82.01	68.1	2.1	—	—	—
B.1			0.0	2.0	0.5	69.60	2.0	1.6	87.54	68.1	2.2	—	—	—
B.2			0.0	2.0	0.4	66.10	2.0	1.6	86.69	68.1	2.2	—	—	—
B.3			0.0	2.0	0.5	63.55	2.0	1.5	79.54	68.1	2.1	—	—	—
C	9.8	90.2	0.3	0.1	2.8	16.27	2.0	1.3	25.80	68.1	1.9	—	—	—
C.1			0.5	0.2	3.3	17.36	2.0	1.4	25.77	68.1	2.3	—	—	—
C.2			0.4	0.3	3.2	14.68	2.0	1.5	22.82	68.1	2.4	—	—	—
C.3			0.3	0.1	2.9	16.20	2.0	1.3	25.67	68.1	1.9	—	—	—

with an Intel(R) 2.40 GHz processor running the SUSE Linux 10.1 Operating System. Again, developing effective selection strategies is likely to heavily reduce this computational effort.

Finally, ee did not report detailed results concerning the use of GMICs and type 2 triangle cuts together because the improvement over the latter is negligible (even if consistent).

Tentative Observations. An additional step in the interpretation of the results of Table 1 can be made by analyzing some statistics on the separated type 2 triangle cuts which are collected in Table 2. More precisely, the table contains for each triangle type and set of instances, the percentage of cuts whose smallest angle belongs to the ranges]0,15],]15,30],]30,45] and]45,60]. These numbers are reported separately for separated and tight cuts.

Below we note possible interpretations of the results reported in the Tables:

– Type 2 triangle cuts vs other lattice-free cuts: Type 2 triangle cuts appear to be quite important: A very few of them together with the GMICs close 98% of the gap in the instances of set A.
– Type 2 triangle cuts vs GMICs: The fact that the triangles close more of the gap than GMICs is very interesting. In fact, the number of triangles needed is not much more than the number of GMICs. This suggests that effort spent in finding the "right triangles" is probably very useful especially in the case in which the continuous variables are more important.
– Need for study of new relaxations: The instances when the integer variables are more important show that the performance of both the GMICs and the triangle inequalities deteriorate. This suggests that analysis of other relaxations based on integer non-basic variables should be pursued. The importance of generating inequalities with strong coefficients on integer variables is again illustrated by the improvement obtained by the triangle cutting planes

Table 2. Type 2 Triangle Statistics on 2-row Instances

		%Min angle			
Cut type	Separated/Tight	0–15	15–30	30–45	45–60
Tr.(y)	Separated in A,B,C	64.66	25.23	8.42	1.70
	Tight in A	62.23	29.79	6.38	1.60
	Tight in B	72.11	24.21	3.16	0.53
	Tight in C	78.16	17.82	3.45	0.57
Tr.(y)	Separated in B.n	64.66	25.23	8.42	1.70
	Tight in B	72.11	24.21	3.16	0.53
	Tight in B.1	66.50	27.50	4.50	1.50
	Tight in B.2	62.81	28.64	6.53	2.01
	Tight in B.3	71.66	24.60	3.21	0.53
Tr.(y)	Separated in C.n	64.66	25.23	8.42	1.70
	Tight in C	78.16	17.82	3.45	0.57
	Tight in C.1	74.76	17.62	6.67	0.95
	Tight in C.2	74.77	19.16	3.74	2.34
	Tight in C.3	78.16	17.82	3.45	0.57
Tr.(x+y)	Separated in B,C	61.86	24.94	10.52	2.67
	Tight in B	77.63	15.35	5.26	1.75
	Tight in C	91.21	5.86	2.56	0.37

based on all nonbasic variables over the triangle cutting planes obtained using only the continuous non-basic variables.
– Effect of bounds on variables: Although bounds on different types of variables deteriorate the quality of the triangle cutting planes, this reduction in the quality of the cutting planes is not significant. Note that the gap for set B (continuous variable more important) deteriorates when we have bounds on non-basic continuous variables. Similarly, the gap for set C (integer variable more important) deteriorates when we have bounds on non-basic integer variables. This illustrates that if a certain type of variables is important, then adding bound on these variables deteriorates the effect of the triangle inequalities and GMICs.
– Shape of Triangle and their importance: Almost consistently, the percentage of triangle cutting planes which have a lower min angle and are tight after resolving is greater than the percentage of total triangle cutting planes with a lower min angle. This observation may relate to the fact that the thin angle gives strong coefficient to integer non-basic variables. However, further study is needed to understand which triangles are more important.

4.3 5-Row Results

The natural question about the strength of the 2-row relaxation for m-row instances ($m > 2$) is addressed in this section in which $m = 5$. The results are reported in Table 3 in which we only consider the basic versions A, B, and C. In addition, we considered a fourth set of instances (C.s in the Table) obtained by modifying the original multidimensional knapsack instances with the aim of enforcing some structured sparsity in the constraints. More precisely, we first generated multidimensional knapsack instances having $m = 5$ rows and $\ell \in \{50, 100, 150\}$ integer non-negative variables as for the sets A, B and C. Then, we partitioned the ℓ variables in two sets $L_1 = \{1, \ldots, \ell/2\}$ and $L_2 = \{\ell/2+1, \ldots, \ell\}$, we fixed to 0 the coefficients of variables in L_1 in the first and the second knapsack constraint, we fixed to 0 the coefficients of variables in L_2 in the third and the fourth knapsack constraint, and we left unchanged the fifth knapsack constraint. Afterwards, as for the instances of set C, we solved the continuous relaxation, we relaxed the m basic variables to be free, we changed 10% of the variables in L_1 and L_2 to be non-negative continuous and we divided the profit of the continuous variables by 100 in the objective function.

The results reported are the same as in Table 3. Again we did not report the results of GMICs and type 2 triangle cuts together. As for the 2-row case, the

Table 3. 5-row Instances

	Characteristics					GMIC			Tr.(y)			Tr.(y+x)		
Set	ny	nx	nz.s	nz.y	nz.x	%gap	ncuts	ntight	%gap	ncuts	ntight	%gap	ncuts	ntight
A	9.5	—	0.3	4.7	—	37.76	5.0	2.3	62.83	1,260.0	4.1	—	—	—
B	9.5	90.5	0.1	5.0	0.9	32.18	5.0	2.1	51.14	1,260.0	4.5	56.79	30,430.8	5.1
C	9.5	90.5	2.9	0.3	4.5	8.97	5.0	1.9	14.16	1,260.0	3.2	18.80	30,430.8	4.4
C.s	9.5	90.5	2.5	1.5	2.3	33.53	5.0	2.7	43.04	182.4	4.5	51.54	10,261.3	5.8

improvement is negligible for sets A, B and C. However, the situation changes substantially for the set C.s where the percentage gap closed by Tr.(y) improves from 43.04 (see Table 3) to 45.08 with the 5 additional GMICs, a 2% improvement. A smaller, but still non-negligible improvement, is obtained by using GMICs together with Tr.(x+y): the percentage gap closed goes from 51.54 (see Table 3) to 52.33.

Tentative Observations. As for the 2-row instances, we collected statistics on the separated type 2 triangle cuts in Table 4. In addition to the information reported for 2-row instances, we collected the percentage of cuts separated by four distinct row-pairing types. Namely, the pairings are distinguished among: two sparse rows with many overlapping variables (s-s-s in the table), two sparse rows with few overlapping variables (s-s-d), one sparse and one dense row (s-d), and two dense rows (d-d). Note that the rows in the sets A, B and C are all dense, thus only case "d-d" above occurs.

Below we note possible interpretations of the results reported in the Tables:

– Gap closed by triangle cutting planes for five row problem: From 2 rows to 5 rows, we observe that the amount of gap closed deteriorates. This is not surprising. However, the triangle cutting planes still seem to do significantly well and could possibly be important with more rows.
– Effect of Sparsity: Sparsity significantly improves the quality of the gap closed both by the triangle cutting planes and GMICs. Interestingly, for the sparse instances, the GMICs and the triangle inequalities seem to be different from each other, and their effect is additive.
– Use of Sparsity for row selection: From the tables it is clear that that the triangle cutting planes that are generated by the use of two rows of the same sparsity pattern are more useful. This suggests that searching for tableau rows and selecting pairs that have similar sparsity pattern to generate two row cutting planes may be a useful heuristic.

Table 4. Type 2 Triangle Statistics on 5-row Instances

		%Min angle				%Row pairing			
Cut type	Sperated/Tight	0–15	15–30	30–45	45–60	s-s-s	s-s-d	s-d	d-d
T.(y)	Separated in A,B,C	65.33	23.30	8.87	2.49	0.00	0.00	0.00	100.00
	Tight in A	68.75	20.63	8.75	1.88	0.00	0.00	0.00	100.00
	Tight in B	79.43	12.00	8.00	0.57	0.00	0.00	0.00	100.00
	Tight in C	92.74	4.84	1.61	0.81	0.00	0.00	0.00	100.00
	Generated in C.s	67.68	23.42	7.59	1.31	8.69	0.00	55.39	35.92
	Tight in C.s	69.29	23.60	6.37	0.75	32.21	0.00	28.84	38.95
Tr.(x+y)	Separated in B,C	61.80	25.36	10.15	2.69	0.00	0.00	0.00	100.00
	Tight in B	81.41	14.57	3.52	0.50	0.00	0.00	0.00	100.00
	Tight in C	94.12	5.29	0.59	0.00	0.00	0.00	0.00	100.00
	Separated in C.s	61.57	25.36	10.45	2.62	7.29	0.00	43.90	48.81
	Tight in C.s	76.86	16.29	6.57	0.29	35.43	0.00	24.86	39.71

5 Conclusions

In this paper, we carried out experiments with type 2 triangle inequalities on different classes of randomly generated instances designed to study the effect of integer non-basic variables, multiple rows and their sparsity patterns. By comparing the gap closed by these inequalities and the Gomory mixed integer cuts for one round, we were able to obtain some understanding of these cutting planes. It is clear that the comparison might be unfair because only few GMICs can be separated with respect to a huge number of type 2 triangle cuts but this is inherent to the fact that we compared a one constraint relaxation with a two constraint relaxation. However, although there are numerous triangles of type 2, very few of these cuts are eventually useful. Concerning quality, we have observed that both GMICs and triangle inequalities significantly deteriorate in performance in the presence of important non-basic integer variables. Finally, we believe that the computational methodology introduced in this paper may be useful in the future to analyze multi-row cuts whose inherent structure is rather different from that of the classical 1-row cuts.

While these initial observations are interesting, the important question of cut selection still remains. Moreover, additional experiments need to be conducted on more application-oriented problems and possibly more rounds of cuts need to be considered to truly understand the impact of these cutting planes.

References

1. Achterberg, T., Koch, T., Martin, A.: MIPLIB 2003. Operations Research Letters 34, 361–372 (2006), http://miplib.zib.de
2. Andersen, K., Cornuéjols, G., Li, Y.: Reduce-and-split cuts: Improving the performance of mixed integer Gomory cuts. Management Science 51, 1720–1732 (2005)
3. Andersen, K., Louveaux, Q., Weismantel, R.: An analysis of mixed integer linear sets based on lattice point free convex sets. Mathematics of Operations Research 35, 233–256 (2010)
4. Andersen, K., Louveaux, Q., Weismantel, R.: Mixed-integer sets from two rows of two adjacent simplex bases (2009), http://hdl.handle.net/2268/35089
5. Andersen, K., Louveaux, Q., Weismantel, R., Wolsey, L.A.: Cutting planes from two rows of a simplex tableau. In: Fischetti, M., Williamson, D.P. (eds.) Proceedings 12th Conference on Integer and Combinatorial Optimization, pp. 1–15. Springer, Heidelberg (2007)
6. Atamturk, A.: http://ieor.berkeley.edu/~atamturk/data/
7. Balas, E.: Intersection cuts - a new type of cutting planes for integer programming. Operations Research 19, 19–39 (1971)
8. Balas, E., Jeroslow, R.: Strenghtening cuts for mixed integer programs. European Journal of Operations Research 4, 224–234 (1980)
9. Basu, A., Conforti, M., Cornuéjols, G., Zambelli, G.: Maximal lattice-free convex sets in linear subspaces (2009), http://www.math.unipd.it/~giacomo/
10. Basu, A., Conforti, M., Cornuéjols, G., Zambelli, G.: Minimal inequalities for an infinite relaxation of integer programs (2009), http://www.math.unipd.it/~giacomo/

11. Borozan, V., Cornuéjols, G.: Minimal valid inequalities for integer constraints. Mathematics of Operations Research 34, 538–546 (2009)
12. Conforti, M., Cornuéjols, G., Zambelli, G.: A geometric perspective on lifting (2009), http://www.math.unipd.it/~giacomo/
13. Cornuéjols, G., Margot, F.: On the facets of mixed integer programs with two integer variables and two constraints. Mathematical Programming 120, 429–456 (2009)
14. Dey, S.S., Tramontani, A.: Recent developments in multi-row cuts. Optima 80, 2–8 (2009)
15. Dey, S.S., Wolsey, L.A.: Lifting integer variables in minimal inequalities corresponding to lattice-free triangles. In: Lodi, A., Panconesi, A., Rinaldi, G. (eds.) Proceedings 13th Conference on Integer and Combinatorial Optimization, pp. 463–475. Springer, Heidelberg (2008)
16. Dey, S.S., Wolsey, L.A.: Constrained infinite group relaxations of MIPs, Tech. Report CORE DP 33, Université catholique de Louvain, Louvain-la-Neuve, Belgium (2009)
17. Espinoza, D.: Computing with multiple-row Gomory cuts. In: Lodi, A., Panconesi, A., Rinaldi, G. (eds.) IPCO 2008. LNCS, vol. 5035, pp. 214–224. Springer, Heidelberg (2008)
18. Fischetti, M., Saturni, C.: Mixed integer cuts from cyclic groups. Mathematical Programming 109, 27–53 (2007)
19. Gomory, R.E.: An algorithm for integer solutions to linear programs. In: Graves, R.L., Wolfe, P. (eds.) Recent Advances in Mathematical Programming, pp. 269–308. Mcgraw-Hill Book Company Inc., New York (1963)
20. Gomory, R.E., Johnson, E.L.: Some continuous functions related to corner polyhedra, part I. Mathematical Programming 3, 23–85 (1972)
21. Gomory, R.E., Johnson, E.L.: Some continuous functions related to corner polyhedra, part II. Mathematical Programming 3, 359–389 (1972)
22. Gomory, R.E., Johnson, E.L.: T-space and cutting planes. Mathematical Programming 96, 341–375 (2003)
23. Johnson, E.L.: On the group problem for mixed integer programming. Mathematical Programming Study 2, 137–179 (1974)
24. Lovász, L.: Geometry of numbers and integer programming. Mathematical Programming: Recent Developments and Applications, 177–210 (1989)
25. Nemhauser, G.L., Wolsey, L.A.: A recursive procedure to generate all cuts for 0-1 mixed integer programs. Mathematical Programming 46, 379–390 (1990)
26. Zambelli, G.: On degenerate multi-row Gomory cuts. Operations Research Letters 37, 21–22 (2009)

An $OPT + 1$ Algorithm for the Cutting Stock Problem with Constant Number of Object Lengths

Klaus Jansen[1,⋆] and Roberto Solis-Oba[2,⋆⋆]

[1] Institut für Informatik, Universität zu Kiel, Kiel, Germany
kj@informatik.uni-kiel.de

[2] Department of Computer Science, The University of Western Ontario, London, Canada
solis@csd.uwo.ca

Abstract. In the *cutting stock problem* we are given a set T of object types, where objects of type $T_i \in T$ have integer length $p_i > 0$. Given a set O of n objects containing n_i objects of type T_i, for each $i = 1, \ldots, d$, the problem is to pack O into the minimum number of bins of capacity β. In this paper we consider the version of the problem in which the number d of different object types is constant and we present an algorithm that computes a solution using at most $OPT + 1$ bins, where OPT is the value of an optimum solution.

1 Introduction

In the *cutting stock problem* we are given a set $T = \{T_1, T_2, \ldots, T_d\}$ of object types, where objects of type T_i have positive integer length p_i. Given an infinite set of bins, each of integer capacity β, the problem is to pack a set $\mathcal{O}$ of n objects into the minimum possible number of bins in such a way that the capacity of the bins is not exceeded; in set $\mathcal{O}$ there are n_i objects of type T_i, for all $i = 1, \ldots, d$. In this paper we consider the version of the problem in which the number d of different object types is constant.

In the related bin packing problem the goal is to pack a set of n objects with positive integer lengths into the minimum possible number of unit capacity bins. The cutting stock problem can be considered as the *high multiplicity* version of bin packing, as defined by Hochbaum and Shamir [8]. In a high multiplicity problem, the input objects are partitioned into types and all objects of the same type are identical. The number of objects of a given type is called the type's multiplicity. Note that a high multiplicity problem allows a compact representation of the input, as the attributes of each type need to be listed only once along with the multiplicity of the type. Hence, any instance of the cutting stock problem

⋆ Research partially supported by the European Project AEOLUS, contract 015964.
⋆⋆ Research partially supported by the Natural Sciences and Engineering Research Council of Canada, grant 227829-2009.

with a constant number of object types can be represented with a number of bits that is logarithmic in the number of objects.

There is extensive research literature on the bin packing and cutting stock problems, attesting to their importance, both, from the theoretical and practical points of view (see e.g. the survey by Coffman et al. [1]). The cutting stock problem was introduced by Eisemann [2] in 1957 under the name of the "Trim problem". The cutting stock and bin packing problems are known to be strongly NP-hard and no approximation algorithm for them can have approximation ratio smaller than 3/2 unless P = NP. In 1985 Marcotte [17] showed that the cutting stock problem with two different object types has the so called integer round-up property and so the algorithm by Orlin in [19] can solve this particular version of the problem in polynomial time. Later, McCormick et al. [18] presented a more efficient $O(\log^2 \beta \log n)$ time algorithm for this same version of the problem.

Filippi and Agnetis [4] proposed an algorithm for the cutting stock problem that uses at most $OPT + d - 2$ bins, where OPT is the value of an optimum solution; hence, this algorithm also finds an optimum solution for the case of $d = 2$. Recently, Filippi [5] improved on the above algorithm for the case when $d \geq 4$ by providing an algorithm that uses at most $OPT + 1$ bins for $2 < d \leq 6$ and at most $OPT + 1 + \lfloor (d-1)/3 \rfloor$ bins for $d > 6$. From the asymptotic point of view, the best known algorithm for the problem is by Karmarkar and Karp [12] and it produces solutions of value at most $OPT + \log^2 d$. This algorithm has running time that is polynomial in $\log n$, and interestingly the exponent of $\log n$ in the running time is independent of d.

It is not known whether the cutting stock problem can be solved in polynomial time for every fixed value d. Similarly, it is not known whether there is any polynomial time algorithm for bin packing that produces a solution of value at most $OPT + k$ for some constant value k. In this paper we make further progress towards answering these questions by providing an algorithm for the cutting stock problem that uses at most $OPT + 1$ bins, for any fixed value d.

Theorem 1. *There is an $O(d^{21d} 2^{2^{3d+3}} (\log^2 n + \log \beta)^4)$ time algorithm for the cutting stock problem with a constant number d of different object types, that solves the problem using at most $OPT + 1$ bins, where OPT is the value of an optimum solution.*

When computing time complexities we use the *log-cost* RAM model, where each arithmetic operation requires time proportional to the logarithm of the size of its operands. Our algorithm uses a variation of the integer programming formulation (IP) for the cutting stock problem of Gilmore and Gomory [6]; furthermore, we take advantage of a result by Eisenbrand and Shmonin [3] stating that IP has an optimum solution with only a constant number of positive variables. By partitioning the set of objects into two groups of small and big objects, we can re-write IP so that only a constant number of constraints is needed to restrict the placement of big objects in the bins. Then, by relaxing the integrality constraints on the variables controlling the packing for the small objects, we obtain a mixed integer program with a constant number of integer variables. We show that this mixed integer program can be solved in polynomial time using Lenstra's algorithm [15], and a simple rounding procedure can be then used to transform this solution into a feasible solution for the cutting stock problem that uses at most $OPT + 1$ bins.

2 Mixed Integer Program Formulation

Let $\varepsilon = \frac{1}{d(2^d+d+1)}$. We partition the set O of objects in two sets: The *big objects*, with lengths at least $\varepsilon\beta$, and the *small objects*, with lengths smaller than $\varepsilon\beta$. Without loss of generality, let $p_1, \ldots, p_\alpha$ be the different lengths of the big objects and let $p_{\alpha+1}, \ldots, p_d$ be the lengths of the small objects. Note that a bin can have at most $1/\varepsilon$ big objects in it.

A *configuration* C_i is a set of objects of total length at most β, so all objects in a configuration can be packed in a bin. Given a configuration C_i, the subset C_i^B of big objects in C_i is called a *big configuration* and the subset C_i^S of small objects in C_i is called a *small configuration.* Observe that C_i^B could be empty. A configuration C_i can be specified with a d-dimensional vector $C_i = \langle a(C_i,1), a(C_i,2) \ldots, a(C_i,d)\rangle$ in which the j-th entry, $a(C_i,j)$, specifies the number of objects of length p_j in C_i. As the number of different object lengths is d, the number of different configurations is at most n^d; similarly, the number of different big configurations is at most $1/\varepsilon^d$.

Let $\mathcal{C}$ be the set of all configurations. The cutting stock problem can be formulated as the following integer program, first proposed by Gilmore and Gomory [6].

$$\begin{aligned} \text{IP} : \min \sum_{C_i \in \mathcal{C}} x_{C_i} & \\ \text{s.}t. \sum_{C_i \in \mathcal{C}} a(C_i,j) x_{C_i} \geq n_j, & \quad \text{for } j = 1, \ldots, d \qquad (1) \\ x_{C_i} \in \mathbb{Z}_{\geq 0}, & \quad \text{for all } C_i \in \mathcal{C} \end{aligned}$$

In this integer program, n_j is the total number of objects of length p_j, and for each configuration C_i, variable x_{C_i} indicates the number of bins storing objects according to C_i. Constraint (1) ensures that all objects are placed in the bins.

Eisenbrand and Shmonin [3] show that this integer program has an optimum solution x^* in which at most 2^d of the variables $x^*_{C_i}$ are non-zero. We will use this result to re-write IP so that the number of big configurations used is at most 2^d. To do this, let us first split each configuration $C_i \in \mathcal{C}$ into a big, C_i^B, and a small,

C_i^S, configuration. Let $\mathcal{C}^B$ be the set of all big configurations. Note that $\mathcal{C}^B$ includes the configuration with no big objects in it.

IP, then, can be re-written as the following feasibility problem.

$$\text{IP1}: \sum_{\substack{C_i \in \mathcal{C} \\ C_i^B = B_\ell}} x_{C_i} \leq y_\ell, \quad \text{for all } B_\ell \in \mathcal{C}^B \tag{2}$$

$$\sum_{B_\ell \in \mathcal{C}^B} y_\ell = m^* \tag{3}$$

$$\sum_{B_\ell \in \mathcal{C}^B} a(B_\ell, j) y_\ell \geq n_j, \quad \text{for all } j = 1, \ldots, \alpha \tag{4}$$

$$\sum_{C_i \in \mathcal{C}} a(C_i, j) x_{C_i} \geq n_j, \quad \text{for all } j = \alpha + 1, \ldots, d \tag{5}$$

$$x_{C_i} \in \mathbb{Z}_{\geq 0}, \quad \text{for all } C_i \in \mathcal{C}$$

$$y_\ell \in \mathbb{Z}_{\geq 0}, \quad \text{for all } B_\ell \in \mathcal{C}^B$$

In this integer program $a(B_\ell, j)$ is the number of objects of length p_j in configuration B_ℓ, y_ℓ is a variable indicating the number of bins in which the big objects are packed according to big configuration B_ℓ, and m^* is the minimum number of bins needed to pack all the objects. Constraint (3) ensures that the number of bins used is the optimum one, while constraints (4) and (5) guarantee that all big and small objects are packed in the bins.

Lemma 1. *$x = \langle x_{C_i}, x_{C_2}, \ldots, x_{C_{|\mathcal{C}|}} \rangle$ is an optimum solution for IP if and only if (x, y) is a feasible solution for IP1, where $y = \langle y_1, y_2, \ldots, y_{|\mathcal{C}^B|} \rangle$, and $y_\ell = \sum_{\substack{C_i \in \mathcal{C} \\ C_i^B = B_\ell}} x_{C_i}$* for every index $\ell = 1, 2, \ldots, |\mathcal{C}^B|$.

From Corollary 6 in [3] and Lemma 1, IP1 has an optimum solution (x^*, y^*) with at most 2^d non-zero variables y^*. Let S^{B^*} be the set of at most 2^d big configurations corresponding to the non-zero variables in y^*. Thus, we can reduce the number of constraints of type (2) by not considering all big configurations $\mathcal{C}^B$, but only those in S^{B^*}. Since we do not know the optimum solution (x^*, y^*) we do not know either which big configurations to select. However, as the number of big configurations is only $1/\varepsilon^d$, there is a constant number $\binom{\varepsilon^{-d}}{2^d}$ of subsets S^B of 2^d big configurations, so we can try them all knowing that one of them will lead to an optimum solution. Furthermore, in IP1 the value of m^* is unknown. However, since $m^* \leq n$, we can use binary search to find in $O(\log n)$ iterations the smallest value for m^* for which IP1 has a feasible solution. Finally, we consider a mixed integer linear programming relaxation of IP1 by relaxing the integrality constraints on the variables x_{C_i}:

$$\text{MILP}(m, S^B): \quad \sum_{\substack{C_i \in \mathcal{C} \\ C_i^B = B_\ell}} x_{C_i} \leq y_\ell, \quad \text{for all } B_\ell \in S^B \tag{6}$$

$$\sum_{B_\ell \in S^B} y_\ell = m \tag{7}$$

$$\sum_{B_\ell \in S^B} a(B_\ell, j) y_\ell \geq n_j, \quad \text{for all } j = 1, \ldots, \alpha \tag{8}$$

$$\sum_{\substack{C_i \in \mathcal{C} \\ C_i^B \in S^B}} a(C_i, j) x_{C_i} \geq n_j, \quad \text{for all } j = \alpha + 1, \ldots, d \tag{9}$$

$$x_{C_i} \geq 0, \quad \text{for all } C_i \in \mathcal{C} \text{ such that } C_i^B \in S^B$$
$$y_\ell \in \mathbb{Z}_{\geq 0}, \quad \text{for all } B_\ell \in S^B$$
$$x_{C_i} = 0, \quad \text{for all } C_i \in \mathcal{C} \text{ such that } C_i^B \notin S^B$$
$$y_\ell = 0, \quad \text{for all } B_\ell \notin S^B$$

where m is the number of bins and S^B is a subset of big configurations.

3 Rounding

In Section 4 we show how to solve MILP(m, S^B) using Lenstra's algorithm [15]. Let (x^+, y^+) be the solution produced by Lenstra's algorithm for MILP(m^*, S^{B^*}); as we show below (see Theorem 2), in this solution at most $2^d + d + 1$ of the variables x^+ have non-zero value. We show now how to obtain an integer solution from (x^+, y^+) that uses at most $m^* + 1$ bins.

For each big configuration $B_\ell \in S^{B^*}$ such that $y_\ell^+ > 0$, the solution (x^+, y^+) uses y_ℓ^+ bins so that the big objects stored in them conform to B_ℓ. Let $\Delta_\ell^+ = y_\ell^+ - \sum_{C_i \in \mathcal{C}; C_i^B = B_\ell} x_{C_i}^+$. If for some big configuration B_ℓ, $\Delta_\ell^+ > 0$, then we select any $C_h \in \mathcal{C}$ such that $C_h^B = B_\ell$ and increase the value of $x_{C_h}^+$ by Δ_ℓ^+. Note that this change does not affect the number of bins that it uses, but it ensures that constraint (6) is satisfied with equality for every $B_\ell \in \mathcal{C}^B$; we need this property for our rounding procedure. For simplicity, let us denote the new solution as (x^+, y^+).

For each $C_i \in \mathcal{C}$ such that $x_{C_i}^+ > 0$, let $x'_{C_i} = x_{C_i}^+ - \lfloor x_{C_i}^+ \rfloor$, so $x_{C_i}^+$ can be split into an integer part $\lfloor x_{C_i}^+ \rfloor$ and a fractional one x'_{C_i}. We round each x'_{C_i} to an integer value as follows.

1. Let $\mathcal{C}' = \{C_i \mid C_i \in \mathcal{C} \text{ and } x'_{C_i} > 0\}$.
2. Consider a $B_\ell \in S^{B^*}$ such that $\sum_{C_i \in \mathcal{C}', C_i^B = B_\ell} x'_{C_i} > 0$ and select a set $Q \subseteq \mathcal{C}'$ such that $C_i^B = B_\ell$ for all $C_i \in Q$, $\sum_{C_i \in Q} x'_{C_i} \geq 1$, and $\sum_{C_i \in Q'} x'_{C_i} < 1$ for all $Q' \subset Q$.

Note that since condition (6) holds with equality for every $B_\ell \in S^{B^*}$ and y_ℓ is integer, then set Q always exists, unless $\mathcal{C}' = \emptyset$.

Take any configuration $C_p \in Q$ and split x'_{C_p} into two parts x''_{C_p}, x'''_{C_p} so that $x'_{C_p} = x''_{C_p} + x'''_{C_p}$ and

$$x''_{C_p} + \sum_{C_i \in Q \setminus \{C_p\}} x'_{C_i} = 1. \tag{10}$$

Observe that x'''_{C_p} could be equal to zero. For simplicity, set $x'_{C_p} = x''_{C_p}$.

3. All configurations in Q have the same big configuration B_ℓ, so we will combine all configurations of Q into a single pseudo-configuration $C_Q = \langle C_{Q_1}, C_{Q_2}, \ldots, C_{Q_d}\rangle$, where $C_{Q_j} = \sum_{C_i \in Q}(a(C_i, j)x'_{C_i})$. Note that

$$\begin{aligned}\sum_{j=1}^{d}(C_{Q_j} \times p_{\pi(j)}) &= \sum_{j=1}^{d}\left(\left(\sum_{C_i \in Q}(a(C_i,j)x'_{C_i})\right)p_{\pi(j)}\right)\\ &= \sum_{C_i \in Q}\left(x'_{C_i}\left(\sum_{j=1}^{d}(a(C_i,j)p_{\pi(j)})\right)\right)\\ &\le \sum_{C_i \in Q} x'_{C_i}\beta, \text{ as each } C_i \text{ is a configuration,}\\ &\qquad\qquad \text{so } \sum_{j=1}^{d}(a(C_i,j)p_{\pi(j)}) \le \beta\\ &= \beta, \text{ by (10).}\end{aligned}$$

so, the total size of the objects in C_Q is β. But, C_Q might not be a feasible configuration, as some of its components C_{Q_i} might not be integer. Let $\overline{C}_Q = \langle \lfloor C_{Q_1}\rfloor, \lfloor C_{Q_2}\rfloor, \ldots, \lfloor C_{Q_d}\rfloor\rangle$, and $C'_Q = \langle C_{Q_1} - \lfloor C_{Q_1}\rfloor, C_{Q_2} - \lfloor C_{Q_2}\rfloor, \ldots, C_{Q_d} - \lfloor C_{Q_d}\rfloor\rangle$. Clearly $\overline{C}_Q$ is a valid configuration and each component $C_{Q_i} - \lfloor C_{Q_i}\rfloor$ of C'_Q has value smaller than 1.

Note that the first α components of C'_Q are zero because for all configurations $C_j \in Q$, $C^B_j = B_\ell$, hence, for $j \le \alpha$, $C_{Q_j} = \sum_{C_i \in Q}\left(a(C_i,j)x'_{C_i}\right) = \sum_{C_i \in Q}(a(B_\ell,j)x'_{C_i}) = a(B_\ell,j)\sum_{C_i \in Q} x'_{C_i} = a(B_\ell,j)$, by (10); observe that $a(B_\ell,j)$ is integer. Thus, each C'_Q is of the form

$$C'_Q = \langle 0,\ldots,0,C'_{Q,\alpha+1},\ldots,C'_{Q,d}\rangle \text{ where } 0 \le C'_{Q,i} < 1 \text{ for all } i = 1,\ldots,d. \tag{11}$$

We can think of C'_Q as containing only a fraction (maybe equal to zero) of an object of each different small length. The fractional items in C'_Q are set aside for the time being.

4. Remove Q from $\mathcal{C}'$ and if $x'''_{C_p} > 0$, add C_p back to $\mathcal{C}'$ and set $x'_{C_p} = x'''_{C_p}$.
5. If $\mathcal{C}' \ne \emptyset$ go back to Step 2.

The above procedure yields a solution that uses m^* bins, but the objects from the configurations C'_Q identified in Step 3 are not yet packed. Let $\mathbb{C}'_Q$ be the set of all these configurations.

Lemma 2. *All objects in configurations $\mathbb{C}'_Q$ can be packed in one bin.*

Lemma 3. *The above rounding procedure packs all the objects in at most m^*+1 bins.*

A description of our algorithm for bin packing is given below.

Algorithm BinPacking(P, U, β)
Input: Sets $P = \{p_1, \ldots, p_d\}$, $U = \{n_1, \ldots, n_d\}$ of object lengths and their multiplicities; capacity β of each bin.
Output: A packing for the objects into at most $OPT + 1$ bins.

```
1. Set $\varepsilon = \frac{1}{d(2^d+d+1)}$ and then partition the set of objects into big (of length at
   least $\varepsilon\beta$) and small (of length smaller than $\varepsilon\beta$). Set $m^* = n$.
2. For each set $S^B$ of $2^d$ big objects do :
       Use Lenstra's algorithm and binary search over the set $V = \{1, 2, \ldots, m^*\}$ to find
       the smallest value $j \in V$, if any, for which MILP$(j, S^B)$ has a solution.
       If a value $j < m^*$ was found for which MILP$(j, S^B)$ has a solution, then set $m^* = j$
       and let $(x^+, y^+)$ be the solution computed by Lenstra's algorithm for MILP$(j, S^B)$.
3. Round $(x^+, y^+)$ as described above and output the corresponding packing
   of the objects into $m^* + 1$ bins.
```

Lemma 4. *Algorithm* BinPacking *computes a solution for the cutting stock problem that uses at most $OPT + 1$ bins.*

4 Solving the Mixed Integer Linear Program

Lenstra's algorithm [15] can be used to solve mixed integer linear programs in which the number of integer variables is constant; the time complexity of the algorithm is $O(P(N'))$, where P is a polynomial and $N' = O(M\eta \log k)$ is the maximum number of bits needed to specify the input, where M is the number of constraints in the mixed integer linear program, η is the number of variables, and k is the maximum of the absolute values of the coefficients of the constraints. Since MILP(m, S^B) has $O(n^d)$ variables, it might seem that the time complexity of Lenstra's algorithm is too high for our purposes as an instance of the high multiplicity bin packing problem is specified with only $N = \sum_{i=1}^{d}(\log p_i + \log n_i) + \log \beta = O(\log \beta + \log n)$ bits, and so $P(N')$ is not a polynomial function of N. In this section we show that Lenstra's algorithm can, in fact, be implemented to run in time polynomial in N.

The set of constraints of MILP(m, S^B) can be written in the form $A(y, x) \leq b$. Let K' denote the closed convex set

$$K' = \{(y, x) \in \mathbb{R}^{2^d + n^d} \mid A(y, x) \leq b\}$$

and let

$$K = \{y \in \mathbb{R}^{2^d} \mid \text{there exists } x \in \mathbb{R}^{n^d} \text{ such that } (y, x) \in K'\};$$

then deciding whether MILP(m, S^B) has a feasible solution is equivalent to deciding whether $K \cap \mathbb{Z}^{2^d} \neq \emptyset$. For completeness we give below a brief description of Lenstra's algorithm.

Algorithm Lenstra(K)
Input: Closed convex set K of dimension D.
Output: A point in $K \cap \mathbb{Z}^D$, if one exists, or null if $K \cap \mathbb{Z}^D = \emptyset$.

```
1. Reduce the dimension of $K$ until we ensure that $K$ has positive volume.
2. Compute a linear transformation $\tau$ that maps $K$ into a ball-like set $\tau K$ such
   that there is a point $\sigma$ and radii $r, R$ with $\frac{R}{r} = 2D^{3/2}$ for which $\mathcal{B}(\sigma, r) \subset$
   $\tau K \subset \mathcal{B}(\sigma, R)$, where $\mathcal{B}(\sigma, z) \subset \mathbb{R}^D$ is the closed ball with center $\sigma$ and
   radius $z$.
3. Compute a *reduced basis* $b_1, b_2, \ldots, b_D$ for the lattice $\tau\mathbb{Z}^D$: a basis such that
   $\prod_{i=1}^{D} \|b_i\| \leq 2^{D(D-1)/4} \times |\text{determinant}(b_1, b_2, \ldots, b_D)|$, where $\| \|$ denotes the
   Euclidean norm.
4. Find a point $v \in \tau\mathbb{Z}^D$ such that $\|v - \sigma\| \leq \frac{1}{2}\sqrt{D} \max\{\|b_i\| \mid i = 1, \ldots, D\}$.
5. If $v \in \tau K$ then output $\tau^{-1}v$.
6. If $v \notin \tau K$ let $H = \sum_{i=1}^{D-1}(\mathbb{R}b_i)$ be the $(D-1)$-hyperplane spanned by
   $b_1, \ldots, b_{D-1}$.
   **For** each integer $i$ such that $H + ib_D$ intersects $\mathcal{B}(\sigma, R)$ **do** :
       Let $\boldsymbol{K}$ be the intersection of $K$ with $H + ib_D$.
       If $v =$ Lenstra$(\boldsymbol{K})$ is not null, then output $\tau^{-1}(v, ib_D)$.
   Output null.
```

4.1 Time Complexity of Lenstra's Algorithm

Step 1 of Lenstra's algorithm (for details see [15]) requires maximizing $O(2^{2d})$ linear functions on K and sometimes using the Hermite normal form algorithm of Kannan and Bachem [10]; the algorithm in [10] requires the computation of $O(2^{2d})$ determinants of matrices of size $O(2^d) \times O(2^d)$, each of whose entries is encoded with $\log n$ bits, and so it runs in $O(2^{4.7d} \log n)$ time by using the algorithm of [9] for computing the determinants.

Maximizing a linear function $f(y) = f_1 y_1 + f_2 y_2 + \cdots f_{2^d} y_{2^d}$ on K is equivalent to maximizing on K' a linear function that depends only on the 2^d variables y; this latter problem can be written as follows.

$$\begin{aligned}
\text{LP} : \max & \sum_{B_\ell \in S^B} f_\ell y_\ell \\
s.t.\ & y_\ell - \sum_{\substack{C_i \in \mathcal{C} \\ C_i^B = B_\ell}} x_{C_i} \geq 0, \quad \text{for all } B_\ell \in S^B \\
& - \sum_{B_\ell \in S^B} y_\ell \geq -m \\
& \sum_{B_\ell \in S^B} a(B_\ell, j) y_\ell \geq n_j, \quad \text{for all } j = 1, \ldots, \alpha \\
& \sum_{\substack{C_i \in \mathcal{C} \\ C_i^B \in S^B}} a(C_i, j) x_{C_i} \geq n_j, \quad \text{for all } j = \alpha + 1, \ldots, d \\
& y_\ell \geq 0 \ \text{ for all } B_\ell \in S^B; \ x_{C_i} \geq 0, \quad \text{for all } C_i \in \mathcal{C} \text{ such that } C_i^B \in S^B
\end{aligned}$$

LP has $2^d + d + 1$ constraints, but it might have a very large number of variables, so we deal with its dual instead:

$$\begin{aligned}
\text{DLP} : \min \delta_0 m - \sum_{j=1}^{d} (\lambda_j n_j) & \\
s.t.\ \delta_\ell - \delta_0 + \sum_{j=1}^{\alpha} (a(B_\ell, j) \lambda_j) \leq -f_\ell, \ \text{for all } B_\ell \in S^B & \qquad (12) \\
-\delta_\ell + \sum_{j=\alpha+1}^{d} (a(C_i, j) \lambda_j) \leq 0, \ \text{for all } C_i \in \mathcal{C}, & \\
B_\ell \in S^B \text{ s.t. } C_i^B = B_\ell & \qquad (13) \\
\delta_0 \geq 0, \delta_\ell \geq 0 \ \text{ for all } B_\ell \in S^B & \\
\lambda_j \geq 0, \quad j = 1, \ldots, d &
\end{aligned}$$

We use the ellipsoid algorithm [13,7] to solve DLP. Note that DLP has only $2^d + d + 1$ variables, but it has a large number $O(n^d)$ of constraints, so for the ellipsoid's algorithm to solve DLP in time polynomial in N, we need an efficient separation oracle that given a vector $\delta = \langle \lambda_1, \ldots, \lambda_d, \delta_0, \ldots, \delta_{2^d} \rangle$ it either determines that δ is a feasible solution for DLP or it finds a constraint of DLP that is violated by δ.

To design this separation oracle, we can think that each object $o_i \in \mathcal{O}$ has length p_i and value λ_i. Each constraint (12) can be tested in constant time. However, constraints (13) are a bit harder to test. Since a configuration C_i for which $C_i^B = B_\ell \in S^B$ includes small objects of total length at most $\beta - \beta_\ell$, where β_ℓ is the total length of the big objects in B_ℓ, then constraints (13) check that for each $C_i \in \mathcal{C}$ and $B_\ell \in S^B$ such that $C_i^B = B_\ell$, the set of small objects in C_i has total value at most δ_ℓ.

Hence, (as it was also observed by Karmakar and Karp [12]) to determine whether the constraints (13) are satisfied we need to solve an instance of the *knapsack problem* where the input is the set of small objects and the knapsack has capacity $\beta - \beta_\ell$. If the maximum value of any subset of small objects of total length at most $\beta - \beta_\ell$ is larger than δ_ℓ then we know that a constraint of type (13) is violated; furthermore, the solution of the knapsack problem indicates exactly which constraint is not satisfied by δ.

Therefore, a separation oracle for DLP needs to be able to efficiently solve any instance of the knapsack problem formed by a set of objects of $d' \leq d$ different types, where objects of type i have length p_i and value λ_i. This knapsack problem can be formulated as an integer program with a constant number of variables and so it can be solved, for example, by using Kannan's algorithm [11] in $O(d^{9d}(d2^{8d}\log^2 n + \log\beta)^3)$ time.

By Lemmas 2.1 and 8.5 of [20] the basic feasible solutions of DLP are 2^d-vectors of rational numbers whose numerators and denominators have absolute values at most $L = (2^d)!n^{d2^d}$. Therefore, we can use the version of the ellipsoid algorithm described in [7] with precision L^{-1} to solve DLP in time polynomial in $\log n$ and $\log\beta$.

Lemma 5. *The maximum number of bits needed to encode each value in the solution (δ, λ) computed by the ellipsoid algorithm for DLP, is $O(d2^{8d}\log^2 n)$.*

Lemma 6. *The running time of the ellipsoid algorithm when solving DLP is $O(d^{9d}2^{4d}(d2^{8d}\log^2 n + \log\beta)^3\log n)$.*

Lemma 7. *Step 1 of Lenstra's algorithm can be performed in time*

$$O(d^{9d}2^{6d}(d2^{8d}\log^2 n + \log\beta)^3\log n).$$

Step 2 of algorithm Lenstra requires finding a linear transformation τ that maps K into a set τK that is nearly "spherical". To do this, a simplex $\mathcal{S}$ spanned by $O(2^d)$ vertices of K is first found and then it is transformed into a polyhedron of "large" volume through an iterative procedure that at each step modifies $\mathcal{S}$ by replacing one of its vertices with some vertex v of K such that the volume of the resulting polyhedron increases by at least a factor of 3/2. Each vertex v can be found by maximizing 2^{d+1} linear functions on K.

Since the volume of the simplex $\mathcal{S}$ is at least $1/(2^d)!$ (see [15]) and by constraint (7) of MILP(m, S^B) the volume of K is at most n^{2^d}, then the number of iterations in the above procedure is at most $\log_{3/2}(n^{2^d}(2^d)!) = O(2^d\log n)$. Therefore, for step 2 of algorithm Lenstra we need to maximize $O(2^d\log n)$ linear functions over K. By using the ellipsoid algorithm to maximize each linear function, the total time needed to perform step 2 of Lenstra's algorithm is $O(d^{9d}2^{5d}(d2^{8d}\log^2 n + \log\beta)^3\log^2 n)$.

For Step 3 we can use the algorithm of Lenstra, Lenstra, and Lovász [14] to find a reduced basis. By Proposition 1.26 in [14] this step can be performed in $O(2^{6d})$ time. Step 4 requires projecting σ over each one of the vectors b_i from the reduced basis and then rounding each component of the projection down to the nearest integer. This step requires $O(2^d)$ multiplications on numbers encoded with $O(2^{6d})$ bits, so this step can be performed in $O(2^{12d})$ time.

Finally, in Step 5, to decide whether $y \in \tau K$, we need to determine whether $y' = \tau^{-1} y \in K$. This requires us to solve MILP when the values of the variables y_ℓ are known: this is just linear program LP when the objective function f is constant and the values for the variables y_ℓ are given. The dual of this linear program is DLP without constraints (12), so we can solve it using the ellipsoid algorithm.

Lemma 8. *The running time of Lenstra's algorithm is*

$$O(d^{9d} 2^{2^{3d}+6d} (d 2^{8d} \log^2 n + \log \beta)^3 \log^2 n).$$

Proof. We have shown that steps 1-5 of the algorithm can be performed in time $T = O(d^{9d} 2^{6d} (d 2^{8d} \log^2 n + \log \beta)^3 \log^2 n)$. As shown in [15] in the "for" loop of step 6 we need to consider $2^{1+2d+2^d(2^d-1)/4}$ different values for i. In each iteration of the "for" loop we perform a recursive call to the algorithm, and the recursive call dominates the running time of every iteration of the loop.

Let $F(D)$ be the time complexity of the algorithm when the input convex set has dimension D. Then, $F(D)$ satisfies the following recurrence:

$$F(D) = T + 2^{1+d+2^d(2^d-1)/4} F(D-1).$$

Therefore, $F(D) = O\left(T (2^{1+d+2^d(2^d-1)/4})^D\right)$. Since $D = 2^d$, the complexity of the algorithm is $O(d^{9d} 2^{2^{3d}+6d} (d 2^{8d} \log^2 n + \log \beta)^3 \log^2 n)$. □

Theorem 2. *A solution for MILP(m^*, S^{B^*}) in which at most $2^d + d + 1$ variables x_{C_i} have positive value can be computed in $O(d^{18d} 2^{2^{3d}+6d} (d 2^{8d} \log^2 n + \log \beta)^3 \log^2 n)$ time.*

Proof of Theorem 1: By Lemma 4 algorithm BinPacking produces a solution for the high multiplicity bin packing problem using at most $OPT + 1$ bins. By Theorem 2 the time complexity of BinPacking is $O(d^{21d} 2^{2^{3d+3}} (\log^2 n + \log \beta)^4)$ as the algorithm needs to solve $O((\varepsilon^{-d})! \log n)$ mixed integer linear programs.

References

1. Coffman Jr., E.G., Garey, M.R., Johnson, D.S.: Approximation algorithms for bin packing: a survey. In: Hochbaum, D.S. (ed.) Approximation algorithms for NP-hard problems, pp. 46–86. PWS Publishing Company (1997)
2. Eisemann, K.: The trim problem. Management Science 3(3), 279–284 (1957)
3. Eisenbrand, F., Shmonin, G.: Carathéorody bounds for integer cones. Operations Research Letters 34, 564–568 (2006)
4. Filippi, C., Agnetis, A.: An asymptotically exact algorithm for the high-multiplicity bin packing problem. Mathematical Programming 104(1), 21–57 (2005)
5. Filippi, C.: On the bin packing problem with a fixed number of object weights. European Journal of Operational Research 181, 117–126 (2007)

6. Gilmore, P.C., Gomory, R.E.: A linear programming approach to the cutting stock problem. Operations Research 9, 849–859 (1961)
7. Grötschel, M., Lovász, L., Schrijver, A.: The ellipsoid method and its consequences in Combinatorial Optimization. Combinatorica 1(2), 169–197 (1981)
8. Hochbaum, D.S., Shamir, R.: Strongly polynomial algorithms for the high multiplicity scheduling problem. Operations Research 39, 648–653 (1991)
9. Kaltofen, E., Villard, G.: On the complexity of computing determinants. Computational Complexity 13(3-4), 91–130 (2004)
10. Kannan, R., Bachem, A.: Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. SIAM Journal on Computing 8, 499–507 (1979)
11. Kannan, R.: Minkowski's convex body theorem and integer programming. Mathematics of Operations Research 12(3), 415–440 (1987)
12. Karmakar, N., Karp, R.M.: An efficient approximation scheme for the one-dimensional bin packing problem. In: Proceedings FOCS, pp. 312–520 (1982)
13. Khachiyan, L.G.: A polynomial algorithm in linear programming. Dokl. Akad. Nauk. SSSR 244, 1093-1096 (1979); English translation: Soviet Math. Dokl. 20, 191–194 (1979)
14. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring Polynomials with rational coefficients. Math. Ann. 261, 515–534 (1982)
15. Lenstra Jr., H.W.: Integer programming with a fixed number of variables. Mathematics of Operations Research 8(4), 538–548 (1983)
16. Lovász, L.: Complexity of algorithms (1998)
17. Marcotte, O.: The cutting stock problem and integer rounding. Mathematical Programming 33, 82–92 (1985)
18. McCormick, S.T., Smallwood, S.R., Spieksma, F.C.R.: A polynomial algorithm for multiprocessor scheduling with two job lengths. Math. Op. Res. 26, 31–49 (2001)
19. Orlin, J.B.: A polynomial algorithm for integer programming covering problems satisfying the integer round-up property. Mathematical Programming 22, 231–235 (1982)
20. Papadimitriou, C.H., Steiglitz, K.: Combinatorial optimization: Algorithms and complexity. Prentice-Hall, Inc., Englewood Cliffs (1982)
21. Schrijver, A.: Theory of Linear and Integer Programming. John Wiley, Chichester (1986)

On the Rank of Cutting-Plane Proof Systems

Sebastian Pokutta[1] and Andreas S. Schulz[2]

[1] Technische Universität Darmstadt, Germany
pokutta@mathematik.tu-darmstadt.de
[2] Massachusetts Institute of Technology, USA
schulz@mit.edu

Abstract. We introduce a natural abstraction of propositional proof systems that are based on cutting planes. This new class of proof systems includes well-known operators such as Gomory-Chvátal cuts, lift-and-project cuts, Sherali-Adams cuts (for a fixed hierarchy level d), and split cuts. The rank of such a proof system corresponds to the number of rounds needed to show the nonexistence of integral solutions. We exhibit a family of polytopes without integral points contained in the n-dimensional 0/1-cube that has rank $\Omega(n/\log n)$ for any proof system in our class. In fact, we show that whenever a specific cutting-plane based proof system has (maximal) rank n on a particular family of instances, then any cutting-plane proof system in our class has rank $\Omega(n/\log n)$ for this family. This shows that the rank complexity of worst-case instances is intrinsic to the problem, and does not depend on specific cutting-plane proof systems, except for log factors. We also construct a new cutting-plane proof system that has worst-case rank $O(n/\log n)$ for any polytope without integral points, implying that the universal lower bound is essentially tight.

Keywords: Cutting planes, proof systems, Gomory-Chvátal cuts, lift-and-project cuts, split cuts.

1 Introduction

Cutting planes are a fundamental, theoretically and practically relevant tool in combinatorial optimization and integer programming. Cutting planes help to eliminate irrelevant fractional solutions from polyhedral relaxations while preserving the feasibility of integer solutions. There are several well-known procedures to systematically derive valid inequalities for the integer hull P_I of a rational polyhedron $P = \{x \in \mathbb{R}^n : Ax \leq b\} \subseteq [0,1]^n$ (see, e.g., [8, 9]). This includes Gomory-Chvátal cuts [5, 17, 18, 19], lift-and-project cuts [1], Sherali-Adams cuts [28], and the matrix cuts of Lovász and Schrijver [21], to name just a few. Repeated application of these operators is guaranteed to yield a linear description of the integer hull, and the question naturally arises of how many rounds are, in fact, necessary. This gives rise to the notion of rank. For example, it is known that the Gomory-Chvátal rank of a polytope contained in the n-dimensional 0/1-cube is at most $O(n^2 \log n)$ [14], whereas the rank of all other methods mentioned before is bounded above by n, which is known to be tight (see, e.g., [7, 8]). These convexification procedures can also be viewed as propositional proof systems (e.g., [6, 12, 13]), each using its own rules to prove that a system of linear inequalities with integer coefficients does not have a $0/1$-solution. While exponential lower bounds on the lengths of the proofs were obtained for specific systems (e.g., [3, 13, 25]), there is no general technique available that would work for all propositional proof systems (which would actually prove that NP $\neq$ co-NP). We formalize the concept of an "admissible" cutting-plane proof system (see Definition 1 below for details) and provide a generic framework that comprises all proof systems based on cutting planes mentioned above and that allows us to make general statements on the rank of these proof systems.

Our contribution. Our main contributions are as follows. The introduction of admissible cutting-plane procedures exposes the commonalities of several well-known operators and helps to explain several of their properties on a higher level. It also allows us to uncover much deeper connections. In particular, in the context of cutting-plane procedures as refutation systems in propositional logic, we will show that if an arbitrary admissible cutting-plane procedure has maximal rank n, then so does the Gomory-Chvátal procedure. In addition, the rank of the matrix cuts of Lovász and Schrijver, Sherali and Adams, and Balas, Ceria, and Cornuéjols and that of the split cut operator is at least $n - 1$. In this sense, we show that some of the better known procedures are essentially the weakest possible members in the family of admissible cutting-plane procedures. However, we also provide a family of instances, i.e., polytopes $P \subseteq [0,1]^n$ with empty integer hull, for which the rank of *any* admissible cutting-plane procedure is $\Omega(n/\log n)$. In fact, we show that the rank of any admissible cutting-plane procedure is $\Omega(n/\log n)$ whenever there is an admissible cutting-plane procedure that has maximal rank n. Last not least, we introduce a new cutting-plane procedure whose rank is bounded by $O(n/\log n)$, showing that the lower bound is virtually tight. In the course of our proofs, we also exhibit several interesting structural properties of integer-empty polytopes $P \subseteq [0,1]^n$ with maximal Gomory-Chvátal rank, maximal matrix cut rank, or maximal split rank.

Related work. A lower bound of n for the rank of the Gomory-Chvátal procedure for polytopes $P \subseteq [0,1]^n$ with $P_I = \emptyset$ was established in [6]. This was later used to provide a lower bound of $(1+\epsilon)n$ on the Gomory-Chvátal rank of arbitrary polytopes $P \subseteq [0,1]^n$, showing that in contrast to most other cutting-plane procedures, the Gomory-Chvátal procedure does not have an upper bound of n if $P_I \neq \emptyset$ [14]. The upper bound is known to be n if $P_I = \emptyset$ [2]. Lower bounds of n for the matrix cut operators N_0, N, and N_+ of Balas, Ceria, and Cornuéjols [1], Sherali and Adams [28], and Lovász and Schrijver [21] were given in [7, 10, 16]. Lower bounds for the split cut operator SC were obtained in [11]. These operators (and some strengthenings thereof) have recently regained attention [15, 20, 24], partly due to an interesting connection between the inapproximability of certain combinatorial optimization problems and the integrality gaps of their (LP and SDP) relaxations. For example, in [27] it was shown that the integrality gaps of the vertex cover and the max cut problem remain at least $2 - \epsilon$ after ϵn rounds of the Sherali-Adams operator. A related result for the stronger Lovász-Schrijver operator established an integrality gap of $2 - \epsilon$ after $\Omega(\sqrt{\log n / \log \log n})$ rounds [15]. In [26] it was shown that even for the stronger Lasserre hierarchy [20] one cannot expect to be able to prove the unsatisfiability of certain k-CSP formulas within $\Omega(n)$ rounds. As a result, a $\frac{7}{6} - \epsilon$ integrality gap for the vertex cover problem after $\Omega(n)$ rounds of the Lasserre hierarchy follows. In [4], the strength of the Sherali-Adams operator is studied in terms of integrality gaps for well-known problems like max cut, vertex cover, and sparsest cut, and in [22] integrality gaps for the fractional matching polytope, which has Gomory-Chvátal rank 1, are provided, showing that although the matching problem can be solved in polynomial time, it cannot be approximated well with a small number of rounds of the Sherali-Adams operator. In addition, it was shown that for certain tautologies that can be expressed in first-order logic, the Lovász-Schrijver N_+ rank can be constant, whereas the Sherali-Adams rank grows poly-logarithmically [12].

Outline. Our work complements these results in in a variety of ways. On the one hand, we provide a basic framework (Section 2) that allows us to show that in the case of polytopes $P \subseteq [0,1]^n$ with $P_I = \emptyset$ all admissible cutting-plane procedures exhibit a similar behavior in terms of maximal rank, log factors omitted (Section 4). On the other hand, we define a new cutting-plane procedure that is optimal with respect to the lower bound, i.e., it establishes $P_I = \emptyset$ in $O(n/\log n)$ rounds and thus outperforms well-known cutting-plane procedures in terms of maximal rank (Section 5). We also derive sufficient and necessary conditions on polytopes with maximal rank (Section 3).

Due to space limitations, we have to refer the reader to the literature (e.g., [8]) or to the full version of this paper for the definition of well-established cutting-plane procedures.

2 Admissible Cutting-Plane Proof Systems

Let $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ be a rational polyhedron that is contained in the n-dimensional 0/1-cube; i.e., we assume that $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, and $P \subseteq [0,1]^n$. We use a_i to denote row i of A, and b_i is the corresponding entry on the right-hand side. The integer hull, P_I, of P is the convex hull of all integer points in P, $P_I = \mathrm{conv}(P \cap \{0,1\}^n)$. If F is a face of the n-dimensional unit cube, $[0,1]^n$, then $P \cap F$ can be viewed as the set of those points in P for which certain coordinates have been fixed to 0 or 1. We define $\varphi_F(P)$ as the projection of P onto the space of variables that are not fixed by F. If P, $Q \subseteq [0,1]^n$ are polytopes, we say that $P \cong Q$ if there exists a face F of the n-dimensional unit cube such that $\varphi_F(P) = Q$. Moreover, $[n] = \{1, 2, \ldots, n\}$.

A cutting-plane procedure consists of an operator M that maps P to a closed convex set $M(P)$, which we call the M-closure of P. Any linear inequality that is valid for $M(P)$ is called an M-cut.

Definition 1. *We say that a cutting-plane procedure is* admissible *if M has the following properties:*

(1) *M* ***strengthens*** *P* ***and keeps*** *P_I* ***intact:*** *$P_I \subseteq M(P) \subseteq P$.*
(2) ***Preservation of inclusion:*** *If $P \subseteq Q$, then $M(P) \subseteq M(Q)$, for all polytopes $P, Q \subseteq [0,1]^n$.*
(3) ***Homogeneity:*** *$M(F \cap P) = F \cap M(P)$, for all faces F of $[0,1]^n$.*
(4) ***Single coordinate rounding:*** *If $x_i \leq \epsilon < 1$ (or $x_i \geq \epsilon > 0$) is valid for P, then $x_i \leq 0$ (or $x_i \geq 1$) is valid for $M(P)$.*
(5) ***Commuting with coordinate flips:*** *Let $\tau_i : [0,1]^n \to [0,1]^n$ with $x_i \mapsto (1 - x_i)$ be a coordinate flip. Then $\tau_i(M(P)) = M(\tau_i(P))$.*
(6) ***Short verification:*** *There exists a polynomial p such that for any inequality $cx \leq \delta$ that is valid for $M(P)$ there is a set $I \subseteq [m]$ with $|I| \leq p(n)$ such that $cx \leq \delta$ is valid for $M(\{x : a_i x \leq b_i, i \in I\})$. We call $p(n)$ the* verification degree *of M.*

Note that these conditions are quite natural and are indeed satisfied by all linear cutting-plane procedures mentioned above; proofs and references are given in the full version of this paper. Condition (1) ensures that $M(P)$ is a relaxation of P_I that is not worse than P itself. Condition (2) establishes the monotonicity of the procedure; as any inequality that is valid for Q is also valid for P, the same should hold for the corresponding M-cuts. Condition (3) states that the order in which we fix certain variables to 0 or 1 and apply the operator should not matter. Condition (4) makes sure that an admissible procedure is able to derive the most basic conclusions, while Condition (5) makes certain that the natural symmetry of the 0/1-cube is maintained. Finally, Condition (6) guarantees that admissible cutting-plane procedures cannot be too powerful; otherwise even $M(P) = P_I$ would be included, and the family of admissible procedures would be too broad to derive interesting results. Note also that (6) is akin to an independence of irrelevant alternatives axiom.

A cutting-plane operator can be applied iteratively; we define $M^{(i+1)}(P) := M(M^{(i)}(P))$.[1] For consistency, we let $M^{(1)}(P) := M(P)$ and $M^{(0)}(P) := P$. Obviously, $P_I \subseteq M^{(i+1)}(P) \subseteq M^{(i)}(P) \subseteq \cdots \subseteq M^{(1)}(P) \subseteq M^{(0)}(P) = P$. In general, it is not clear whether there exists a finite $k \in \mathbb{Z}_+$ such that $P_I = M^{(k)}(P)$. However, we will see that for polytopes $P \subseteq [0,1]^n$ with $P_I = \emptyset$ this follows from properties (3) and (4). In this sense, every admissible cutting-plane procedure can be viewed as a system for proving the unsatisfiability of propositional formulas in conjunctive normal form (which can be naturally represented as systems of integer inequalities), which is the setting considered here. The rank of P with respect to M is the minimal $k \in \mathbb{Z}_+$ such that $P_I = M^{(k)}(P)$. We write $\text{rk}_M(P) = k$ (and drop the index M if it is clear from the context).

In the following, we put together some useful properties of admissible cutting-plane procedures that follow almost directly from the definition. We define $L \cap M$ as $(L \cap M)(P) := L(P) \cap M(P)$ for all polytopes $P \subseteq [0,1]^n$.

[1] To avoid technical difficulties, we assume that $M(P)$ is again a rational polytope whenever we apply M repeatedly.

Lemma 2. *Let L and M be admissible cutting-plane procedures. Then $L \cap M$ is an admissible cutting-plane procedure.*

In other words, admissible cutting-plane procedures are closed under intersection. Moreover, whenever M is admissible, $\text{rk}(Q) \leq \text{rk}(P)$ for $Q \subseteq P \subseteq [0,1]^n$ with $P_I = \emptyset$:

Lemma 3. *Let M be admissible and consider polytopes $Q \subseteq P \subseteq [0,1]^n$. Then $rk(Q) \leq rk(P)$ if $Q_I = P_I$.*

2.1 Universal Upper Bounds

We will now show that there is a natural upper bound on $\text{rk}(P)$ for any admissible cutting-plane procedure M whenever $P_I = \emptyset$, and that this upper bound is attained if and only if the rank of $P \cap F$ is maximal for all faces F of $[0,1]^n$. The proof of the first result is similar to that for the Gomory-Chvátal procedure [2, Lemma 3] and is omitted from this extended abstract.

Theorem 4. *Let M be an admissible cutting-plane procedure and let $P \subseteq [0,1]^n$ be a polytope of dimension d with $P_I = \emptyset$. If $d = 0$, then $M(P) = \emptyset$. If $d > 0$, then $rk(P) \leq d$.*

The following lemma states that $\text{rk}(P)$ is "sandwiched" between the largest rank of P intersected with a facet of the 0/1-cube and that number plus one.

Lemma 5. *Let M be an admissible cutting-plane procedure and let $P \subseteq [0,1]^n$ be a polytope with $P_I = \emptyset$. Then $k \leq rk(P) \leq k+1$, where $k = \max_{(i,l)\in[n]\times\{0,1\}} rk(P \cap \{x_i = l\})$. Moreover, if there exist $i \in [n]$ and $l \in \{0,1\}$ such that $rk(P \cap \{x_i = l\}) < k$, then $rk(P) = k$.*

Proof. Clearly, $k \leq \text{rk}(P)$, by Lemma 3. For the right-hand side of the inequality, observe that $M^k(P) \cap \{x_i = l\} = M^k(P \cap \{x_i = l\}) = \emptyset$ by Property (3) of Definition 1. It follows that $x_i < 1$ and $x_i > 0$ are valid for $M^k(P)$ for all $i \in [n]$. Hence $x_i \leq 0$ and $x_i \geq 1$ are valid for $M^{k+1}(P)$ for all $i \in [n]$ and we can deduce $M^{k+1}(P) = \emptyset$, i.e., $\text{rk}(P) \leq k+1$. It remains to prove that $\text{rk}(P) = k$ if there exist $i \in [n]$ and $l \in \{0,1\}$ such that $\text{rk}(P \cap \{x_i = l\}) =: h < k$. Without loss of generality assume that $l = 1$; otherwise apply the corresponding coordinate flip. Then $M^h(P) \cap \{x_i = l\} = \emptyset$ and thus $x_i < 1$ is valid for $M^h(P)$ and as $h < k$ we can derive that $x_i \leq 0$ is valid for $M^k(P)$. It follows now that $M^k(P) = M^k(P) \cap \{x_i = 0\} = M^k(P \cap \{x_i = 0\}) = \emptyset$, which implies $\text{rk}(P) \leq k$; the claim follows. □

Interestingly, one can show that $\text{rk}(P \cap \{x_i = l\}) = k$ for all $(i,l) \in [n] \times \{0,1\}$ is not sufficient for $\text{rk}(P) = k+1$. We immediately obtain the following corollary:

Corollary 6. *Let M be an admissible cutting-plane procedure and let $P \subseteq [0,1]^n$ be a polytope with $P_I = \emptyset$. Then $rk(P) = n$ if and only if $rk(P \cap F) = k$ for all k-dimensional faces F of $[0,1]^n$ with $1 \leq k \leq n$.*

Proof. One direction follows by induction from Lemma 5; the other direction is trivial. □

3 Polytopes $P \subseteq [0,1]^n$ with $P_I = \emptyset$ and Maximal Rank

We will now study polytopes $P \subseteq [0,1]^n$ with $P_I = \emptyset$ and $\mathrm{rk}_M(P) = n$, where M is an admissible cutting-plane procedure. For lack of space, most proofs are omitted from this section. They can be found in the full version of this paper. We use e to denote the all-ones vector of appropriate dimension. For a face F of the 0/1-cube, we define $\frac{1}{2}e^F$ to be fixed to 0 or 1 according to F and to 1/2 on all other coordinates. Let $\mathrm{Int}(P)$ denote the interior of P, and $\mathrm{RInt}_F(P)$ denote the interior of $\varphi_F(P)$. We use F_k to denote the set of all n-dimensional points that have exactly k coordinates equal to 1/2, and the other $n-k$ coordinates are 0 or 1. The polytope B_n is defined as

$$B_n := \left\{x \in [0,1]^n \mid \sum_{i\in S} x_i + \sum_{i\in[n]\setminus S}(1-x_i) \geq 1 \quad \text{for all } S \subseteq [n]\right\}$$

for all $n \in \mathbb{Z}_+$. Note that B_n contains no integer points, and if F is a k-dimensional face of $[0,1]^n$, then $B_n \cap F \cong B_k$. Moreover, $B_n = \mathrm{conv}(F_2)$.

Lemma 7. *Let M be admissible. Then rk$(B_n) \leq n-1$.*

We first consider two-dimensional polytopes $P \subseteq [0,1]^2$ with maximal rank.

Theorem 8. *Let M be admissible and let $P \subseteq [0,1]^2$ be a polytope with $P_I = \emptyset$ so that rk$(P) = 2$. Then*

(1) $\frac{1}{2}e \in \mathit{Int}(P)$, *and*
(2) $P \cap \{x_i = l\} \neq \emptyset$ *for all* $(i,l) \in [2] \times \{0,1\}$.

If M is the Gomory-Chvátal operator, P', we can obtain the stronger statement that $P' = \{\frac{1}{2}e\}$. In general, $\frac{1}{2}e \notin M(P)$. More specifically, in [23] the following characterization of polytopes $P \subseteq [0,1]^n$ with $P_I = \emptyset$ and maximal Gomory-Chvátal rank rk_{GC} was shown:

Theorem 9. *Let $P \subseteq [0,1]^n$ be a polytope with $P_I = \emptyset$. Then the following are equivalent:*

(1) $\mathrm{rk}_{GC}(P) = n$.
(2) $B_n = P'$.
(3) $F \cap P \neq \emptyset$ *for all one-dimensional faces F of* $[0,1]^n$.

We now prove a similar, but slightly weaker version for generic admissible cutting-plane procedures. This weakening is a direct consequence of the fact that, in general, $\frac{1}{2}e \in \mathrm{Int}(P)$ fails to imply $\frac{1}{2}e \in M(P)$.

Theorem 10. *Let M be admissible and let $P \subseteq [0,1]^n$ be a polytope with $P_I = \emptyset$ and rk$(P) = n$. Then*

(1) $P \cap F \neq \emptyset$ *for all one-dimensional faces F of* $[0,1]^n$.
(2) *For* $v \in F_2$ *and* $F := \bigcap_{(i,l)\in[n]\times\{0,1\}: v_i = l, l \neq \frac{1}{2}} \{x_i = l\}$, $v \in \mathit{RInt}_F(P)$.
(3) $B_n \subseteq P$.

3.1 Results for Gomory-Chvátal Cuts, Matrix Cuts, and Split Cuts

We immediately obtain the following corollary from Theorem 10, which shows that the Gomory-Chvátal procedure is, in some sense, weakest possible: Whenever the rank of some admissible cutting-plane procedure is maximal, then so is the Gomory-Chvátal rank. More precisely:

Corollary 11. *Let M be admissible and let $P \subseteq [0,1]^n$ be a polytope with $P_I = \emptyset$ and $\operatorname{rk}_M(P) = n$. Then $\operatorname{rk}_{GC}(P) = n$.*

Proof. By Theorem 10 (1) we have that $P \cap F \neq \emptyset$ for all one-dimensional faces F of $[0,1]^n$. With Theorem 9 we therefore obtain $\operatorname{rk}_{GC}(P) = n$. □

Note that Corollary 11 does not hold for polytopes $P \subseteq [0,1]^n$ with $P_I \neq \emptyset$: Let $P_n = \{x \in [0,1]^n \mid \sum_{i\in[n]} x_i \geq \frac{1}{2}\}$. Then $(P_n)_I = \{x \in [0,1]^n \mid \sum_{i\in[n]} x_i \geq 1\} \neq \emptyset$. In [7, Section 3] it was shown that $\operatorname{rk}_{GC}(P_n) = 1$, but $\operatorname{rk}_{N_0}(P_n) = n$.

We can also derive a slightly weaker relation between the rank of matrix cuts, split cuts, and other admissible cutting-plane procedures. First we will establish lower bounds for the rank of B_n. The following result was provided in [7, Lemma 3.3] for matrix cuts and in [8, Lemma 6] for split cuts.

Lemma 12. *Let $P \subseteq [0,1]^n$ be a polytope and let $F_k \subseteq P$. Then $F_{k+1} \subseteq N_+(P)$ and $F_{k+1} \subseteq SC(P)$.*

This yields:

Lemma 13. *Let $M \in \{N_0, N, N_+, SC\}$. Then $\operatorname{rk}_M(B_n) = n-1$.*

Proof. As $B_n = \operatorname{conv}(F_2)$, Lemma 12 implies that $F_n \subseteq M^{(n-2)}(B_n)$, and thus $\operatorname{rk}(B_n) \geq n-1$. Together with Lemma 7 it follows that $\operatorname{rk}(B_n) = n-1$. □

We also obtain the following corollary that shows that the M-rank with $M \in \{N_0, N, N_+, SC\}$ is at least $n-1$ whenever it is n with respect to any other admissible cutting-plane procedure.

Corollary 14. *Let L be an admissible cutting-plane procedure, let $M \in \{N_0, N, N_+, SC\}$, and let $P \subseteq [0,1]^n$ be a polytope with $P_I = \emptyset$ and $\operatorname{rk}_L(P) = n$. Then $\operatorname{rk}_M(P) \geq n-1$ and, if P is half-integral, then $\operatorname{rk}_M(P) = n$.*

Proof. If $\operatorname{rk}_L(P) = n$, then $B_n \subseteq P$ by Theorem 10 and $\operatorname{rk}_M(B_n) = n-1$ by Lemma 13. So the first part follows from Lemma 3. In order to prove the second part observe that $P \cap F \neq \emptyset$ for all one-dimensional faces F of $[0,1]^n$. Thus $P \cap F \cong F_2$ for all two-dimensional faces F of $[0,1]^n$ and, by Lemma 12, $M(P) \cap F = \{\frac{1}{2}e^F\}$. Therefore $B_n \subseteq M(P)$. The claim now follows from Lemma 13. □

We will now consider the case where $P \subseteq [0,1]^n$ is half-integral with $P_I = \emptyset$ in detail. The polytope $A_n \subseteq [0,1]^n$ is defined by

$$A_n := \Big\{x \in [0,1]^n \mid \sum_{i\in S} x_i + \sum_{i\in[n]\setminus S} (1-x_i) \geq \frac{1}{2} \quad \text{for all } S \subseteq [n]\Big\}.$$

Lemma 15. *Let $P \subseteq [0,1]^n$ be a half-integral polytope with $P_I = \emptyset$. Then*

$$rk_M(P) = n \text{ if and only if } P = A_n,$$

with $M \in \{GC, N_0, N, N_+, SC\}$.

Proof. It suffices to show that $P = A_n$ if and only if $\mathrm{rk}_{GC}(P) = n$. Let $\mathrm{rk}_{GC}(P) = n$. By Theorem 10 we have that $P \cap F \neq \emptyset$ for all one-dimensional faces F of $[0,1]^n$ and as $P_I = \emptyset$ and P half-integral, it follows that $P \cap F = \{\frac{1}{2}e^F\}$ and $P = A_n$. For the other direction, observe that if $P = A_n$, then $P \cap F \neq \emptyset$ for all one-dimensional faces F of $[0,1]^n$ and, by Theorem 9, we therefore have $\mathrm{rk}_{GC}(P) = n$. $\square$

Hence, in the case of half-integral polytopes without integral points there is exactly one polytope that realizes the maximal rank for the classical operators. Combining Corollary 11 and Lemma 15, we obtain:

Corollary 16. *Let $P \subseteq [0,1]^n$ be a half-integral polytope with $P_I = \emptyset$, and let M be an admissible cutting-plane procedure. Then $\mathrm{rk}_M(P) = n$ implies $P = A_n$.*

For half-integral polytopes $P \subseteq [0,1]^2$ with $P_I = \emptyset$ the matrix cut operators, the split cut operator, and the Gomory-Chvátal procedure are actually identical.

Lemma 17. *Let $P \subseteq [0,1]^2$ be a half-integral polytope with $P_I = \emptyset$. Then $N_0(P) = N(P) = N_+(P) = SC(P) = P'$.*

Note that Lemma 15 and Lemma 17 are in strong contrast to the case where $P \subseteq [0,1]^n$ is a half-integral polytope with $P_I \neq \emptyset$: In the remark after Corollary 11, the polytope $P_n = \{x \in [0,1]^n \mid \sum_{i \in [n]} x_i \geq \frac{1}{2}\}$ has $\mathrm{rk}_{GC}(P_n) = 1$ but $\mathrm{rk}_{N_0}(P_n) = n$ as shown in [7, Theorem 3.1]. On the other hand, the polytope $P = \mathrm{conv}(\{(0,0),(1,0),(\frac{1}{2},1)\}) \subseteq [0,1]^2$ has $\mathrm{rk}_{N_0}(P) = 1$, but $\mathrm{rk}_{GC}(P) = 2$ as P is half-integral and $\frac{1}{2}e \in \mathrm{Int}(P)$.

4 A Universal Lower Bound

We now establish a universal lower bound on the rank of admissible cutting-plane procedures. Our approach makes use of inequalities as certificates for non-membership:

Definition 18. *Let $cx \leq \delta$ with $c \in \mathbb{Z}^n$ and $\delta \in \mathbb{Z}$ be an inequality. The* violation set *$V(c,\delta) := \{x \in \{0,1\}^n : cx > \delta\}$ is the set of* 0/1 *points for which $cx \leq \delta$ serves as a certificate of infeasibility.*

The following observation is an essential building block in establishing the lower bound:

Lemma 19. *Let M be an admissible cutting-plane procedure, and let $P \subseteq [0,1]^n$ be a polytope. Let $cx \leq \delta$ with $c \in \mathbb{Z}^n$ and $\delta \in \mathbb{Z}$ be a valid inequality for $M(P)$ whose certificate of $M(P)$-validity depends only on $\{c_i x \leq \delta_i : i \in I\}$, where I is an index set and $c_i x \leq \delta_i$ with $c_i \in \mathbb{Z}^n$ and $\delta_i \in \mathbb{Z}$ is valid for P, for all $i \in I$. Then $V(c,\delta) \subseteq \bigcup_{i \in I} V(c_i,\delta_i)$.*

Proof. The proof is by contradiction. Suppose there is $x_0 \in \{0,1\}^n$ such that $x_0 \in V(c,\delta) \setminus \bigcup_{i\in I} V(c_i,\delta_i)$. We define $Q := [0,1]^n \cap \bigcap_{i\in I}\{x : c_i x \leq \delta_i\}$. Note that $x_0 \in Q_I$. On the other hand, by Property (6) of Definition 1, $cx \leq \delta$ is valid for $M(Q)$ as well. Thus $x_0 \notin M(Q)$ as $cx \leq \delta$ is valid for $M(Q)$ and $x_0 \in V(c,\delta)$. But then $Q_I \not\subseteq M(Q)$ and therefore M is not admissible, a contradiction. □

This lemma can be interpreted as follows: Together, a set of inequalities $c_i x \leq \delta_i$ certifies that a certain set of 0/1 points is not contained in P. The cutting plane procedure combines these inequalities into a new one, $cx \leq \delta$, that certifies that a (hopefully large) subset of the set of 0/1 points is not contained in P. The fact that we will exploit in order to establish a lower bound is that an admissible cutting-plane procedure can access at most a polynomial number of inequalities in the derivation of a single new inequality. If we now had a polytope $P \subseteq [0,1]^n$ with $|V(a,\beta)|$ small for all inequalities $ax \leq \beta$ in a linear description of P, we could estimate how many rounds it takes to generate an inequality $cx \leq \delta$ so that $V(c,\delta) = \{0,1\}^n$. The following observation characterizes $P_I = \emptyset$ in terms of a violation set $V(c,\delta)$.

Lemma 20. *Let $P \subseteq [0,1]^n$ be a polytope. Then $P_I = \emptyset$ if and only if there exists an inequality $cx \leq \delta$ valid for P_I with $V(c,\delta) = \{0,1\}^n$.*

Proof. Clearly, if there exists an inequality $cx \leq \delta$ valid for P_I with $V(c,\delta) = \{0,1\}^n$, then $P_I = \emptyset$. For the other direction, $ex \leq -1$ is valid for $P_I = \emptyset$, and $V(e,-1) = \{0,1\}^n$. □

Next we establish an upper bound on the growth of the size of $V(c,\delta)$.

Lemma 21. *Let M be admissible with verification degree $p(n)$. Further, let $P = \{x : Ax \leq b\} \subseteq [0,1]^n$ be a polytope with $P_I = \emptyset$ and define $k := \max_{i\in[m]} |V(a_i,b_i)|$. If $cx \leq \delta$ has been derived by M from $Ax \leq b$ within ℓ rounds, then $|V(c,\delta)| \leq p(n)^\ell k$.*

Proof. The proof is by induction on the number ℓ of rounds. For $\ell = 1$, $cx \leq \delta$ can be derived with the help of at most $p(n)$ inequalities $\{a_i x \leq b_i\}$ from the original system $Ax \leq b$. By Lemma 19, it follows that $V(c,\delta) \subseteq \bigcup_i V(a_i,b_i)$ and thus $|V(c,\delta)| \leq \sum_i |V(a_i,b_i)| \leq p(n)k$. Now consider the case $\ell > 1$. The derivation of $cx \leq \delta$ involves at most $p(n)$ inequalities $\{c_i x \leq \delta_i\}$ each of which has been derived in at most $\ell - 1$ rounds. By Lemma 19, it follows that $V(c,\delta) \subseteq \bigcup_i V(c_i,\delta_i)$ and thus $|V(c,\delta)| \leq \sum_i |V(c_i,\delta_i)| \leq p(n)(p(n)^{\ell-1}k) \leq p(n)^\ell k$. □

We are ready to prove a universal lower bound on the rank of admissible cutting-plane procedures:

Theorem 22. *Let $k \in \mathbb{Z}_+$ be fixed, and let M be admissible with verification degree $p(n)$. Further, let $P = \{x : Ax \leq b\} \subseteq [0,1]^n$ be a polytope with $P_I = \emptyset$ such that $P \cap F \neq \emptyset$ for all k-dimensional faces F of $[0,1]^n$. Then, for all $n \geq 2k$, $rk(P) \in \Omega(n/\log n)$.*

Proof. We will first show that if $P \cap F \neq \emptyset$ for all k-dimensional faces F of $[0,1]^n$ and $cx \leq \delta$ is a valid inequality for P, then $cx \leq \delta$ can cut off at most $(2n)^k$ 0/1 points, i.e., $|V(c,\delta)| \leq (2n)^k$. Without loss of generality, we may assume that $c \geq 0$ and that $c_i \geq c_j$ whenever $i \leq j$; otherwise we can apply coordinate flips and variable permutations. Define $l := \min\{j \in [n] : \sum_{i=1}^{j} c_i > \delta\}$. Suppose $l \leq n-k$. Define $F := \bigcap_{i=1}^{n-k}\{x_i = 1\}$. Observe that $\dim(F) = k$ and $cx > \delta$ for all $x \in F$ as $l \leq n-k$. Thus $P \cap F = \emptyset$, which contradicts our assumption that $P \cap F \neq \emptyset$ for all k-dimensional faces F of $[0,1]^n$. Therefore, $k \geq n-l+1$. By the choice of l, every 0/1 point x_0 cut off by $cx \leq \delta$ has to have at least l coordinates equal to 1. The number ζ of 0/1 points of dimension n with this property is bounded by

$$\zeta \leq 2^{n-l}\binom{n}{l} \leq 2^k\binom{n}{n-l} \leq 2^k\binom{n}{k-1} \leq 2^k n^k \leq (2n)^k.$$

Note that the third inequality holds as $k \leq n/2$, by assumption. It follows that $|V(c,\delta)| \leq (2n)^k$.

As we have seen, any inequality $\pi x \leq \pi_0$ valid for P can cut off at most $(2n)^k$ 0/1-points. In order to prove that $P_I = \emptyset$, we have to derive an infeasibility certificate $cx \leq \delta$ with $V(c,\delta) = \{0,1\}^n$, by Lemma 20. Thus, $|V(c,\delta)| = 2^n$ is a necessary condition for $cx \leq \delta$ to be such a certificate. If $cx \leq \delta$ is derived in ℓ rounds by M from $Ax \leq b$ then, by Lemma 21, we have that $|V(c,\delta)| \leq p(n)^\ell (2n)^k$. Hence, $\ell \in \Omega(n/\log n)$ and, therefore, $\mathrm{rk}(P) \in \Omega(n/\log n)$. □

Note that the result can be easily generalized to non-fixed k if k is growing slowly enough as a function of n, e.g., $k \in O(\log n)$. Theorem 22 implies that, in contrast to the case where $P_I \neq \emptyset$, when dealing with polytopes with $P_I = \emptyset$, the property of having high/maximal rank is *universal*, i.e., it is a property of the polytope and not the particular cutting-plane procedure used. We immediately obtain the following corollary:

Corollary 23. *Let M be admissible. Then rk$(B_n) \in \Omega(n/\log n)$ and rk$(A_n) \in \Omega(n/\log n)$.*

Proof. It is sufficient to observe that $A_n \cap F \neq \emptyset$ for all one-dimensional faces F of $[0,1]^n$ and $B_n \cap F \neq \emptyset$ for all two-dimensional faces F of $[0,1]^n$. The claim then follows from Theorem 22. □

For $k \in \mathbb{N}$, it is also easy to see that $\frac{1}{2}e \in M^k(B_n)$ whenever $M^k(B_n) \neq \emptyset$. This is because B_n is symmetric with respect to coordinate flips and thus $\frac{1}{2}e$ is obtained by averaging over all points in $M^k(B_n)$. The next corollary relates all cutting plane procedures in terms of maximal rank:

Corollary 24. *Let $P \subseteq [0,1]^n$ be a polytope with $P_I = \emptyset$ and let L, M be two admissible cutting-plane procedures. If $\mathrm{rk}_L(P) = n$, then $\mathrm{rk}_M(P) \in \Omega(n/\log n)$.*

Proof. If $\mathrm{rk}_L(P) = n$*, then, by Theorem* 10*, we have that* $P \cap F \neq \emptyset$ *for all one-dimensional faces* F *of* $[0,1]^n$*. The claim now follows from Theorem* 22*.* □

In this sense, modulo log-factors, all admissible cutting-plane procedures are of similar strength, at least as far as proving 0/1-infeasibility of a system of linear inequalities is concerned.

5 A Rank Optimal Cutting-Plane Procedure

Traditional convexification procedures such as Gomory-Chvátal or lift-and-project have worst-case rank n, and thus one might wonder if the lower bound of $\Omega(n/\log n)$ in Theorem 22 is tight. We will now construct a new, admissible cutting-plane procedure that is asymptotically optimal with respect to this bound.

Definition 25. *Let* $P \subseteq [0,1]^n$ *be a polytope. The cutting-plane procedure "+" is defined as follows. Let* $\tilde{J} \subseteq [n]$ *with* $|\tilde{J}| \leq \lceil \log n \rceil$ *and let* $I \subseteq \tilde{I} \subseteq [n]$ *with* $\tilde{I} \cap \tilde{J} = \emptyset$*. If there exists* $\epsilon > 0$ *such that*

$$\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I} (1 - x_i) + \sum_{i \in J} x_i + \sum_{i \in \tilde{J} \setminus J} (1 - x_i) \geq \epsilon$$

is valid for P *for all* $J \subseteq \tilde{J}$*, then we add the inequality* $\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I} (1 - x_i) \geq 1$*, and we call this inequality a "+-cut." Furthermore, we define* P^+ *to be the set of points in* P *that satisfy all +-cuts.*

Let us first prove that +-cuts are indeed valid; i.e., they do not cut off any integer points contained in P. At the same time, the proof of the following lemma helps to establish that the +-operator satisfies Property (6) of Definition 1.

Lemma 26. *Let* $P \subseteq [0,1]^n$ *be a polytope. Every +-cut is valid for* P_I*.*

Proof. For $\tilde{J} \subseteq [n]$ with $|\tilde{J}| \leq \lceil \log n \rceil$, and $I \subseteq \tilde{I} \subseteq [n]$ with $\tilde{I} \cap \tilde{J} = \emptyset$, let $\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I} (1 - x_i) \geq 1$ be the corresponding +-cut. Using Farkas' Lemma and Carathéodory's Theorem, we can verify in polynomial time that $\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I} (1 - x_i) + \sum_{i \in J} x_i + \sum_{i \in \tilde{J} \setminus J} (1 - x_i) \geq \epsilon$ is valid for P for all $J \subseteq \tilde{J}$. Note that we have at most $2^{\lceil \log n \rceil} \in O(n)$ of these initial inequalities.

Now we round up all right-hand sides to 1, which leaves us with inequalities that are valid for P_I. By induction, we can verify that

$$\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I} (1 - x_i) + \sum_{i \in J} x_i + \sum_{i \in J_0 \setminus J} (1 - x_i) \geq 1$$

is valid for P_I with $J_0 = \tilde{J} \setminus \{i_0\}$, $i_0 \in \tilde{J}$, and $J \subseteq J_0$. For this, consider

$$\begin{array}{rl} & \frac{1}{2}\Big(\sum_{i\in I} x_i + \sum_{i\in \tilde{I}\setminus I}(1-x_i) + \sum_{i\in J} x_i + \sum_{i\in J_0\setminus J}(1-x_i) + x_{i_0} \geq 1\Big) \\ + & \frac{1}{2}\Big(\sum_{i\in I} x_i + \sum_{i\in \tilde{I}\setminus I}(1-x_i) + \sum_{i\in J} x_i + \sum_{i\in J_0\setminus J}(1-x_i) + (1-x_{i_0}) \geq 1\Big) \\ \hline & \sum_{i\in I} x_i + \sum_{i\in \tilde{I}\setminus I}(1-x_i) + \sum_{i\in J} x_i + \sum_{i\in J_0\setminus J}(1-x_i) \geq \frac{1}{2} \end{array}$$

We can again round up the right-hand side and iteratively repeat this process until $|J_0| = 0$. □

The "+"-operator is indeed admissible. In addition to (6), properties (1), (2), (4), and (5) clearly hold. It remains to prove (3). Let F be a k-dimensional face of $[0,1]^n$ and let $P \subseteq [0,1]^n$ be a polytope. Without loss of generality, we can assume that F fixes the last $n-k$ coordinates to 0. Clearly, $(P\cap F)^+ \subseteq P^+ \cap F$. For the other direction, let $\sum_{i\in I} x_i + \sum_{i\in\tilde{I}\setminus I}(1-x_i) \geq 1$ be a +-cut valid for $(P\cap F)^+$ with $I \subseteq \tilde{I} \subseteq [n]$. Then there exists $\epsilon > 0$ such that

$$\sum_{i\in I} x_i + \sum_{i\in \tilde{I}\setminus I}(1-x_i) + \sum_{i\in J} x_i + \sum_{i\in \tilde{J}\setminus J}(1-x_i) \geq \epsilon$$

is valid for $P \cap F$ with $J \subseteq \tilde{J} \subseteq [n]$, $|\tilde{J}| \leq \lceil \log n \rceil$, and $\tilde{I} \cap \tilde{J} = \emptyset$. By Farkas' Lemma, there exists $\tau \geq 1$ such that

$$\sum_{i\in I} x_i + \sum_{i\in \tilde{I}\setminus I}(1-x_i) + \sum_{i\in J} x_i + \sum_{i\in \tilde{J}\setminus J}(1-x_i) + \tau(\sum_{k+1\leq i\leq n} x_i) \geq \epsilon$$

with $J \subseteq \tilde{J} \subseteq [n]$, $|\tilde{J}| \leq \lceil \log n \rceil$ and $\tilde{I} \cap \tilde{J} = \emptyset$ is valid for P. Hence, so is the weaker inequality

$$\sum_{i\in I} x_i + \sum_{i\in \tilde{I}\setminus I}(1-x_i) + \sum_{i\in J} x_i + \sum_{i\in \tilde{J}\setminus J}(1-x_i) + \sum_{k+1\leq i\leq n} x_i \geq \frac{\epsilon}{\tau} \ .$$

By Definition 25,

$$\sum_{i\in I} x_i + \sum_{i\in \tilde{I}\setminus I}(1-x_i) + \sum_{k+1\leq i\leq n} x_i \geq 1$$

is valid for P^+. Restricting it to the face F, we get that

$$\sum_{i\in I} x_i + \sum_{i\in \tilde{I}\setminus I}(1-x_i) \geq 1$$

is valid for $P^+ \cap F$; thus, property (3) holds.

In the following we will show that, for any given polytope $P \subseteq [0,1]^n$ with $P_I = \emptyset$, $\text{rk}_+(P) \in O(n/\log n)$. This is a direct consequence of the following lemma; we use $P^{(k)}$ to denote the k-th closure of the "+" operator.

Lemma 27. *Let $P \subseteq [0,1]^n$ be a polytope with $P_I = \emptyset$. Then $\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I}(1 - x_i) \geq 1$ with $I \subseteq \tilde{I} \subseteq [n]$, $|\tilde{I}| \geq n - k\lceil \log n \rceil$ is valid for $P^{(k+1)}$.*

Proof. The proof is by induction on k. Let $k = 0$. As $P_I = \emptyset$, there exists $\epsilon > 0$ such that $\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I}(1 - x_i) \geq \epsilon$ is valid for P for all $I \subseteq \tilde{I} = [n]$. Thus $\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I}(1 - x_i) \geq 1$ is valid for P^+ for all $I \subseteq \tilde{I} = [n]$. Consider now $k \geq 1$. Then $\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I}(1 - x_i) \geq 1$ is valid for $P^{(k)}$ for all $I \subseteq \tilde{I} \subseteq [n]$ with $|\tilde{I}| \geq n - (k-1)\lceil \log n \rceil$. Now consider $I \subseteq \tilde{I} \subseteq [n]$ with $n - k\lceil \log n \rceil \leq |\tilde{I}| < n - (k-1)\lceil \log n \rceil$. Pick $\tilde{J} \subseteq [n]$ such that $|\tilde{J}| \leq \lceil \log n \rceil$, $\tilde{I} \cap \tilde{J} = \emptyset$, and $|\tilde{I} \cup \tilde{J}| \geq n - (k-1)\lceil \log n \rceil$. Then for all $J \subseteq \tilde{J}$ we have that $\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I}(1 - x_i) + \sum_{i \in J} x_i + \sum_{i \in \tilde{J} \setminus J}(1 - x_i) \geq 1$ is valid for $P^{(k)}$ by induction hypothesis. We may conclude that $\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I}(1 - x_i) \geq 1$ is valid for $P^{(k+1)}$ by Definition 25. □

We are ready to establish an upper bound on the rank of the "+" operator:

Theorem 28. *Let $P \subseteq [0,1]^n$ be a polytope with $P_I = \emptyset$. Then $\text{rk}_+(P) \in O(n/\log n)$.*

Proof. It suffices to derive the inequalities $x_i \geq 1$ and $x_i \leq 0$. By Lemma 27 we have that $\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I}(1 - x_i) \geq 1$ with $I \subseteq \tilde{I} \subseteq [n]$, $|\tilde{I}| \geq n - k\lceil \log n \rceil$ is valid for $P^{(k+1)}$. Thus $\sum_{i \in I} x_i + \sum_{i \in \tilde{I} \setminus I}(1 - x_i) \geq 1$ with $I \subseteq \tilde{I} = \{i\}$ is valid for $k \geq (n-1)/\lceil \log n \rceil$. Observe that for $I = \{i\}$ and $I = \emptyset$ we obtain that $x_i \geq 1$ and $x_i \leq 0$ are valid for $P^{(k+1)}$, respectively. □

References

[1] Balas, E., Ceria, S., Cornuejols, G.: A lift-and-project cutting plane algorithm for mixed 0-1 programs. Mathematical Programming 58, 295–324 (1993)

[2] Bockmayr, A., Eisenbrand, F., Hartmann, M., Schulz, A.: On the Chvátal rank of polytopes in the 0/1 cube. Discrete Applied Mathematics 98, 21–27 (1999)

[3] Bonet, M., Pitassi, T., Raz, R.: Lower bounds for cutting planes proofs with small coefficients. In: Proceedings of the 27th Annual ACM Symposium on Theory of Computing, pp. 575–584 (1995)

[4] Charikar, M., Makarychev, K., Makarychev, Y.: Integrality gaps for Sherali-Adams relaxations. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, pp. 283–292 (2009)

[5] Chvátal, V.: Edmonds polytopes and a hierarchy of combinatorial problems. Discrete Mathematics 4, 305–337 (1973)

[6] Chvátal, V., Cook, W., Hartmann, M.: On cutting-plane proofs in combinatorial optimization. Linear algebra and its applications 114, 455–499 (1989)

[7] Cook, W., Dash, S.: On the matrix-cut rank of polyhedra. Mathematics of Operations Research 26, 19–30 (2001)

[8] Cornuéjols, G.: Valid inequalities for mixed integer linear programs. Mathematical Programming 112, 3–44 (2008)
[9] Cornuejols, G., Li, Y.: Elementary closures for integer programs. Operations Research Letters 28, 1–8 (2001)
[10] Cornuéjols, G., Li, Y.: A connection between cutting plane theory and the geometry of numbers. Mathematical Programming 93, 123–127 (2002)
[11] Cornuéjols, G., Li, Y.: On the rank of mixed 0,1 polyhedra. Mathematical Programming 91, 391–397 (2002)
[12] Dantchev, S.: Rank complexity gap for Lovász-Schrijver and Sherali-Adams proof systems. In: Proceedings of the 39th Annual ACM Symposium on Theory of Computing, pp. 311–317 (2007)
[13] Dash, S.: An exponential lower bound on the length of some classes of branch-and-cut proofs. Mathematics of Operations Research 30, 678–700 (2005)
[14] Eisenbrand, F., Schulz, A.: Bounds on the Chvátal rank of polytopes in the 0/1-cube. Combinatorica 23, 245–261 (2003)
[15] Georgiou, K., Magen, A., Pitassi, T., Tourlakis, I.: Integrality gaps of 2-o(1) for vertex cover SDPs in the Lovász-Schrijver hierarchy. In: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, pp. 702–712 (2007)
[16] Goemans, M., Tuncel, L.: When does the positive semidefiniteness constraint help in lifting procedures? Mathematics of Operations Research 26, 796–815 (2001)
[17] Gomory, R.: Outline of an algorithm for integer solutions to linear programs. Bulletin of the American Mathematical Society 64, 275–278 (1958)
[18] Gomory, R.: Solving linear programming problems in integers. In: Bellman, R., Hall, M. (eds.) Proceedings of Symposia in Applied Mathematics X, pp. 211–215. American Mathematical Society, Providence (1960)
[19] Gomory, R.: An algorithm for integer solutions to linear programs. In: Recent Advances in Mathematical Programming, pp. 269–302. McGraw-Hill, New York (1963)
[20] Lasserre, J.: 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)
[21] Lovász, L., Schrijver, A.: Cones of matrices and set-functions and 0-1 optimization. SIAM Journal on Optimization 1, 166–190 (1991)
[22] Mathieu, C., Sinclair, A.: Sherali-Adams relaxations of the matching polytope. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, pp. 293–302 (2009)
[23] Pokutta, S., Schulz, A.: A note on 0/1 polytopes without integral points and with maximal rank (2009) (preprint)
[24] Pokutta, S., Schulz, A.: On the connection of the Sherali-Adams closure and border bases (2009) (submitted)
[25] Pudlak, P.: On the complexity of propositional calculus. In: Sets and Proofs, Invited Papers from Logic Colloquium '97, pp. 197–218. Cambridge University Press, Cambridge (1999)
[26] Schoenebeck, G.: Linear level Lasserre lower bounds for certain k-CSPs. In: Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science, pp. 593–602 (2008)
[27] Schoenebeck, G., Trevisan, L., Tulsiani, M.: Tight integrality gaps for Lovász-Schrijver LP relaxations of vertex cover and max cut. In: Proceedings of the 39th Annual ACM Symposium on Theory of Computing, pp. 302–310 (2007)
[28] Sherali, H., Adams, W.: A hierarchy of relaxations between the continuous and convex representations for zero-one programming problems. SIAM Journal on Discrete Mathematics 3, 411–430 (1990)

Author Index