Abstract

1 Introduction

In combinatorial optimization the aim is to find the optimal solution in a discrete and usually finite yet large set of solutions. For many specific combinatorial optimization problems such a solution can be found efficiently. For many others, finding optimal or in many cases near optimal solutions is NP-hard. A common approach to deal with such problems is relaxing the discrete solution set into a continuous set, where the optimization problem becomes tractable. Obtaining feasible solutions by means of such a relaxation requires an additional step of rounding the potentially fractional solution of the continuous relaxation into integer solutions.

In this paper, our focus is on linear relaxation of combinatorial optimization problems. Combinatorial optimization was pioneered by Edmonds even before efficient algorithms for solving linear programming problems where introduced by Khachiyan [Kha80] and later by Karmarkar [Kar84]. For problems such as the MINIMUM COST SPANNING TREE PROBLEM there are linear programming relaxations whose basic feasible solutions coincide with integral solutions, i.e. spanning trees. For other problems the value of the linear programming relaxation provides a bound (lower bound for a minimization problem and upper bound for a maximization problem) on the optimal solution. A common and successful approach is to round these (potentially) fractional solutions into integer solutions for the combinatorial optimization problem at hand. The Integrality gap of a linear relaxation of an integer programming problem is the worst case ratio between the objective values of the discrete problem and the continuous problem. Equivalently, the integrality gap of the linear programming relaxation is a limit to the rounding approach: rounding a fractional solution into an integer solution incurs a multiplicative cost proportional to the integrality gap. In this dissertation we study integrality gaps for different combinatorial optimization problems and introduce new rounding algorithms that imply bounds on their respective integrality gaps.

2 Integrality Gap

In this paper we focus on finding solutions to general Integer Linear Programs (IP). Integer Programming (and more generally Mixed Integer Linear Programming) can be used to model many practical optimization problems including scheduling, logistics and resource allocation. Recall that the set of feasible points for a pure IP (henceforth IP) is the set

$$S(A,b) = \{x \in \mathbb{Z}^n \ : \ Ax \geq b\}. \tag{1}$$

If we drop the integrality constraints, we have the linear relaxation of set $S(A, b)$,

$$P(A, b) = \{x \in \mathbb{R}^n \ : \ Ax \geq b\}. \tag{2}$$

Let $I = (A, b)$ denote an instance. Then $S(I)$ and $P(I)$ denote $S(A, b)$ and $P(A, b)$, respectively. Given a linear objective function c, recall that an IP is min $\{cx : \ x \in S(I)\}$. It is NP-hard even to determine if an IP instance has a feasible solution [GJ90]. However, intelligent branch-and-bound strategies allow commercial and open-source MILP solvers to give exact solutions (or near-optimal solution with provable bound) to many specific instances of NP-hard combinatorial optimization problems.

Relaxing the integrality constraints gives the polynomial-time-solvable linear-programming relaxation: min $\{cx : \ x \in P(I)\}$. The optimal value of this linear program (LP), denoted $z_{\mathrm{LP}}(I, c)$, is a lower bound on the optimal value for the IP, denoted $z_{\mathrm{IP}}(I, c)$. The solutions can also provide some useful global structure, even though the fractional values might not directly meaningful.

Many researchers (see [WS11, Vaz01]) have developed polynomial-time LP-based approximation algorithms that find solutions for special classes of IPs whose cost are provably smaller than $C \cdot z_{LP}(I, c)$. The approximation factor C can be a constant or depend on the input parameters of the IP, e.g. $O(\log(n))$ where n is the number of variables in the formulation of the IP (the dimension of the problem). However, for many combinatorial optimization problems there is a limit to such techniques based on LP relaxations, represented by the integrality gap of the IP formulation. Recall that integrality gap $g(I)$ for instance I is defined to be $g(I) = \max_{c \geq 0} \frac{z_{IP}(I,c)}{z_{LP}(I,c)}$. An example of instance specific integrality gap is the integrality gap of the subtour elimination relaxation for the 2-edge-connected spanning multigraph problem on n vertices. The instance is the complete graph on n vertices. Alexander et al. [ABE06] showed the instance specific integrality gap of the subtour elimination relaxation for the 2-edge-connected multigraph problem for instances of the problem with $n = 10$ is at most $\frac{7}{6}$.

This value depends on the constraints in (1). We cannot hope to find solutions for the IP with objective values better than $g(I) \cdot z_{LP}(I, c)$.

More generally we can define the integrality gap for a class of instances $\mathcal{I}$ as follows.

$$g(\mathcal{I}) = \max_{c \geq 0, I \in \mathcal{I}} \frac{z_{IP}(I, c)}{z_{LP}(I, c)}. \tag{3}$$

For example, the aforementioned integrality gap of the subtour elimination relaxation for the 2-edge-connected multigraph problem is at most $\frac{3}{2}$ [Wol80] and at least $\frac{6}{5}$ [ABE06]. Therefore, we cannot hope to obtain an LP-based $(\frac{6}{5} - \epsilon)$-approximation algorithm for this problem using this LP relaxation.

Our methods apply theory connecting integrality gaps to sets of feasible solutions. Instances I with $g(I) = 1$ has $P(I) = \text{conv}(S(I))$, the convex hull of the lattice of feasible points. In this case, $P(I)$ is an *integral* polyhedron. The spanning tree polytope of graph G, $\text{ST}(G)$, and the perfect-matching polytope of graph G, $\text{PM}(G)$, have this property ([Edm70, Edm65]). For such problems there is an algorithm to express vector $x \in P(I)$ as a convex combination of points in $S(I)$ in polynomial time [GLS93].

Proposition 1. *If $g(I) = 1$, then for $x \in P(I)$ there exists $\theta \in [0,1]^k$, where $\sum_{i=1}^k \theta_i = 1$ and $\tilde{x}^i \in S(I)$ for $i \in [k]$ such that $\sum_{i=1}^k \theta_i \tilde{x}^i \leq x$. Moreover, we can find such a convex combination in polynomial time.*

An equivalent way of describing Proposition 1 is the following Theorem of Carr and Vempala [CV04].

Theorem 2 (Carr, Vempala [CV04])**.** *Let $x \in P(I)$. There exists $\theta \in [0,1]^k$ where $\sum_{i=1}^k \theta_i = 1$ and $\tilde{x}^i \in \mathcal{D}(S(I))$ for $i \in [k]$ such that $\sum_{i=1}^k \theta_i \tilde{x}^i \leq Cx$ if and only if $g(I) \leq C$.*

Recall that $\mathcal{D}(P(I))$ is the set of points x' such that there exists a point $x \in P$ with $x' \geq x$, also known as the dominant of $P(I)$. For covering problems the polyhedron is essentially the same as its dominant, but this is not true in general. While there is an exact algorithm for problems with gap 1 as stated in Proposition 1, Theorem 2 is existential, with no construction. To study integrality gaps, we wish to find such a solution constructively: assuming reasonable complexity assumptions, a specific problem $\mathcal{I}$ with $1 < g(\mathcal{I}) < \infty$, and $x \in P(I)$ for some $I \in \mathcal{I}$, can we find $\theta \in [0,1]^k$, where $\sum_{i=1}^k \theta_i = 1$ and $\tilde{x}^i \in S(I)$ for $i \in [k]$ such that $\sum_{i=1}^k \theta_i \tilde{x}^i \leq Cx$ in polynomial time? We wish to find the smallest factor C as possible.

3 Contributions of this paper

We give a general approximation framework for solving binary IPs. Consider the set of point described by sets $S(I)$ and $P(I)$ as in (1) and (2), respectively. Assume in addition that $S(I), P(I) \subseteq [0,1]^n$. For a vector $x \in \mathbb{R}^n_{\geq 0}$ such that $x \in P(I)$, let $\text{supp}(x) = \{i \in [n] : x_i \neq 0\}$. For an integer β let $\{\beta\}^n$ be the vector $y \in \mathbb{R}^n$ with $y_i = \beta$ for $i \in [n]$.

We introduce the *Fractional Decomposition Tree Algorithm* (FDT) which is a polynomial-time algorithm that given a point $x \in P(I)$ produces a convex combination of feasible points in $S(I)$ that are dominated by a "factor" C of x in the coordinates corresponding to x. If $C = g(I)$, it would be optimal. However we can only guarantee a factor of $g(I)^{|\text{supp}(x)|}$. FDT relies on iteratively solving linear programs that are about the same size as the description of $P(I)$.

Theorem 3. *Assume $1 \leq g(I) < \infty$. The Fractional Decomposition Tree (FDT) algorithm, given $x^* \in P(I)$, produces in polynomial time $\lambda \in [0,1]^k$ and $z^1, \dots, z^k \in S(I)$ such that $k \leq |\operatorname{supp}(x^*)|$, $\sum_{i=1}^k \lambda_i z^i \leq \min(Cx^*, \{1\}^n)$, and $\sum_{i=1}^k \lambda_i = 1$. Moreover, $C \leq g(I)^{|\operatorname{supp}(x^*)|}$.*

A subroutine of the FDT, called the DomToIP algorithm, finds feasible solutions to any IP with finite gap. This can be of independent interest, especially in proving that a model has unbounded gap.

Theorem 4. *Assume $1 \leq g(I) < \infty$. The DomToIP algorithm finds $\hat{x} \in S(I)$ in polynomial time.*

For a generic IP instance I it is NP-hard to even decide if the set of feasible solutions $S(I)$ is empty or not. There are a number of heuristics for this purpose, such as the feasibility pump algorithm [FGL05, FS09]. These heuristics are often very effective and fast in practice, however, they can sometimes fail to find a feasible solution. Moreover, these heuristics do not provide any bounds on the quality of the solution they find.

Here is how the FDT algorithm works in a high level: in iteration i the algorithm maintains a convex combination of vectors in $\mathcal{D}(L(I))$ that have a 0 or 1 value for coordinates indexed $0, \dots, i-1$. Let y be a vector in the convex combination in iteration i of the algorithm. We solve a linear programming problem that gives us $\theta \in [0,1]$ and $y^0, y^1 \in \mathcal{D}(L(I))$ such that $g(I)y \geq \theta_1 y^0 + (1-\theta)y^1$ and $y^0_i = 0$ and $y^1_i = 1$. We then replace y in the convex combination with $\frac{\theta}{g(I)}y^0 + \frac{1-\theta}{g(I)}y^1$. Repeating this for every vector in the convex combination from previous iteration yields a convex combination of points that is "more" integral. If in any iteration there are too many points in the convex combination we solve a linear programming problem that "prunes" the convex combination. At the end we find a convex combination of integer solutions $\mathcal{D}(L(I))$. For each such solution z we invoke the DomToIP algorithm (see Section 4) to find $z' \in S(I)$ where $z' \leq z$.

One can extend the FDT algorithm for binary IPs into covering $\{0,1,2\}$ IPs by losing a factor $2^{|\operatorname{supp}(x)|}$ on top of the loss for FDT. In order to eradicate this extra factor, we need to treat the coordinate i with $x_i = 1$ differently. We focus on the 2-EDGE-CONNECTED MULTIGRAPH GRAPH PROBLEM (2EC): Given a graph $G = (V, E)$ and $c \in \mathbb{R}^E_{\geq 0}$ find a 2-edge-connected multi-subgraph (henceforth a multigraph) of G with minimum cost. The natural linear programming relaxation for this problem is

$$\min\{cx \ : \ x(\delta(U)) \geq 2 \ \text{ for } \emptyset \subset U \subset V, \ x \in [0,2]^E\} \tag{4}$$

We denote the feasible region of this LP by Subtour(G). Let 2EC(G) be the convex hull of incidence vectors of 2-edge-connected multigraphs of graph G. Following the definition in

(3) have

$$g(\text{2EC}) = \max_{c\geq 0, G} \frac{\min_{x\in \text{2EC}(G)} cx}{\min_{x\in \text{Subtour}(G)} cx}. \tag{5}$$

Theorem 5. *Let $G = (V, E)$ and x be an extreme point of* $\text{Subtour}(G)$*. The FDT algorithm for 2EC produces $\lambda \in [0, 1]^k$ and 2-edge-connected multigraphs $F_1, \ldots, F_k$ such that $k \leq 2|V| - 1$, $\sum_{i=1}^k \lambda_i \chi^{F_i} \leq \min(Cx, \{2\}^n)$, and $\sum_{i=1}^k \lambda_i = 1$. Moreover, $C \leq g(\text{2EC})^{|E_x|}$.*

3.1 Experiments.

Although the bound guaranteed in both Theorems 3 and 5 are very large, we show that in practice, the algorithm works very well for network design problems described above. We show how one might use FDT to investigate the integrality gap for such well-studied problems.

3.1.1 Minimum vertex cover problem

TODO

3.1.2 Tree augmentation problem

In the Tree Augmentation Problem (TAP) we are given a graph $G = (V, E)$, a tree T. We also have a cost vector $c \in \mathbb{R}^{E\setminus T}_{\geq 0}$. A subset F of $E \setminus T$ is called a *feasible augmentation* if $(V, T \cup F)$ is a 2-edge-connected graph. In TAP we seek the minimum cost feasible augmentation. The natural linear programming relaxation for TAP is

$$\min\{cx \ : \sum_{\ell \in \text{cov}(e)} x_\ell \geq 1 \text{ for } e \in T,\ x \in [0, 1]^{E\setminus T}\}. \tag{6}$$

where $\text{cov}(e)$ is set of edges $\ell \in E \setminus T$ such that e is in the unique cycle of $T \cup \{\ell\}$. We call the LP above the cut-LP. The integrality gap of the cut-LP is known to be between $\frac{3}{2}$ [CKKK08] and 2 [FJ81]. We create random fractional extreme points of the cut-LP and round them using FDT. For the instances that we create the blow-up factor is always below $\frac{3}{2}$ providing an upper bound for such instances.

3.1.3 2-edge-connected multigraph problem

Known polyhedral structure makes it easier to study integrality gaps for such problems. We use the idea of fundamental extreme point [CR98, BC11, CV04] to create the "hardest" LP solutions to decompose.

There are fairly good bounds for the integrality gap for TSP or 2EC. Benoit and Boyd [BB08] used a quadratic program to show the integrality gap of the subtour elimination relaxation for the TSP, $g(\text{TSP})$, is at most $\frac{20}{17}$ for graphs with at most 10 vertices. Alexander et al. [ABE06] used the same ideas to provide an upper bound of $\frac{7}{6}$ for $g(\text{2EC})$ on graphs with at most 10 vertices.

A *Carr-Vempala point* x is a fractional point in the subtour elimination relaxation where the fractional edges of x, edges e with $0 < x_e < 1$, form a Hamiltonian cycle. For 2EC we show that the integrality gap is at most $\frac{6}{5}$ for Carr-Vempala points with at most 12 vertices on the Hamiltonian cycle formed by the fractional edges. For Carr-Vempala points we assume that 1-edges are replaced by long paths of 1-edges making these points into potentially harder to round instances.

3.2 Notations

For vectors $x, y \in \mathbb{R}_n$ we say x dominates y if $x_i \geq y_i$ for $i = 1, \ldots, n$. For $m \times n$ matrix A, let A_j be the j-th row of A and A^j be the j-th column of A. For a set S of vectors in $\mathbb{R}_n$, $\text{conv}(S)$ is the convex hull of all the points in S.

4 Finding a Feasible Solution

Consider an instance $I = (A, b)$ of the IP formulation. Define sets $S(I)$ and $P(I)$ as in (1) and (2), respectively. Assume $S(I) \subseteq \{0,1\}^n$ and $P(I) \subseteq [0,1]^n$. For simplicity in the notation we denote $P(I), S(I)$, and $g(I)$ with P, S, and g for this section and the next section. Also, for both sections we assume $t = |\,\text{supp}(x)|$. Without loss of generality we can assume $x_i = 0$ for $i = t+1, \ldots, n$.

In this section we prove Theorem 4. In fact, we prove a stronger result.

Lemma 6. *Given $\tilde{x} \in \mathcal{D}(P)$ and $\tilde{x} \in \{0,1\}^n$, there is an algorithm (the DomToIP algorithm) that finds $\bar{x} \in S$ in polynomial time, such that $\bar{x} \leq \tilde{x}$.*

Notice that Lemma 6 implies Theorem 4, since it is easy to obtain an integer point in $\mathcal{D}(P)$: rounding up any fractional point in P gives us a point in $\mathcal{D}(P)$.

4.1 Proof of Lemma 6: The DomToIP Algorithm

We start by introducing an algorithm that "fixes" the variables iteratively, starting from from the first coordinate and ending at the t-th coordinate. Suppose we run the algorithm for $\ell \in \{0, \ldots, t-1\}$ iterations and in each iteration we find $x^{(\ell)} \in \mathcal{D}(P)$ such that $x_i^{(\ell)} \in \{0,1\}$ for $i = 1, \ldots, \ell$. Notice that we can set $x^{(0)} = \tilde{x}$. Now consider the following linear program.

The variables of this linear program are the $z \in \mathbb{R}^n$ variables.

$$
\begin{aligned}
\text{DomToIP}(x^{(\ell)}) \qquad & \min && z_{\ell+1} && && (7)\\
& \text{s.t.} && Az \geq b && && (8)\\
& && z_j = x_j^{(\ell)} && j = 1, \ldots, \ell && (9)\\
& && z_j \leq x_j^{(\ell)} && j = \ell+1, \ldots, n && (10)\\
& && z \;\geq 0 && && (11)
\end{aligned}
$$

If the optimal value to DomToIP($x^{(\ell)}$) is 0, then let $x_{\ell+1}^{(\ell+1)} = 0$. Otherwise if the optimal value is strictly positive let $x_{\ell+1}^{(\ell+1)} = 1$. Let $x_j^{(\ell+1)} = x_j^{(\ell)}$ for $j \in [n] \setminus \{\ell+1\}$ (See Algorithm 1).

The above procedure suggests how to find $x^{(\ell+1)}$ from $x^{(\ell)}$. The DomToIP algorithm initializes with $x^{(0)} = \tilde{x}$ and iteratively calls this procedure in order to obtain $x^{(t)}$.

Algorithm 1: The DomToIP algorithm

Input: $\tilde{x} \in \mathcal{D}(P)$, $\tilde{x} \in \{0,1\}^n$
Output: $x^{(t)} \in S$, $x^{(t)} \leq \tilde{x}$

$x^{(0)} \leftarrow \tilde{x}$
for $\ell = 0$ ***to*** $t-1$ **do**
    $x^{(\ell+1)} \leftarrow x^{(\ell)}$
    $\eta \leftarrow$ optimal value of DomToIP($x^{(\ell)}$)
    if $\eta = 0$ **then**
        $x_{\ell+1}^{(\ell+1)} \leftarrow 0$
    else
        $x_{\ell+1}^{(\ell+1)} \leftarrow 1$
    end
end

We prove that indeed $x^{(t)} \in S$. First, we need to show that in any iteration $\ell = 0, \ldots, t-1$ of DomToIP that DomToIP($x^{(\ell)}$) is feasible. We show something stronger. For $\ell = 0, \ldots, t-1$ let

$$
\begin{aligned}
\text{LP}^{(\ell)} &= \{z \in P \;:\; z \leq x^{(\ell)} \text{ and } z_j = x_j^{(\ell)} \text{ for } j \in [\ell]\}, \text{ and}\\
\text{IP}^{(\ell)} &= \{z \in \text{LP}^{(\ell)} \;:\; z \in \{0,1\}^n\}.
\end{aligned}
$$

Notice that if $\text{LP}^{(\ell)}$ is a non-empty set then DomToIP($x^{(\ell)}$) is feasible. We show by induction on ℓ that $\text{LP}^{(\ell)}$ and $\text{IP}^{(\ell)}$ are not empty sets for $\ell = 0, \ldots, t-1$. First notice that $\text{LP}^{(0)}$

is clearly feasible since by definition $x^{(0)} \in \mathcal{D}(P)$, meaning there exists $z \in P$ such that $z \leq x^{(0)}$. By Theorem 2, there exists $\tilde{z}^i \in S$ and $\theta_i \geq 0$ for $i \in [k]$ such that $\sum_{i=1}^k \theta_i = 1$ and $\sum_{i=1}^k \theta_i \tilde{z}^i \leq gz$. Hence, $\sum_{i=1}^k \theta_i \tilde{z}^i \leq gz \leq gx^{(0)}$. So if $x_j^{(0)} = 0$, then $\sum_{i=1}^k \theta_i \tilde{z}_j^i = 0$, which implies that $\tilde{z}_j^i = 0$ for all $i \in [k]$ and $j \in [n]$ where $x_j^{(0)} = 0$. Hence, $\tilde{z}^i \leq x^{(0)}$ for $i \in [k]$. Therefore $\tilde{z}^i \in \mathrm{IP}^{(0)}$ for $i \in [k]$, which implies $\mathrm{IP}^{(0)} \neq \emptyset$.

Now assume $\mathrm{IP}^{(\ell)}$ is non-empty for some $\ell \in [t-2]$. Since $\mathrm{IP}^{(\ell)} \subseteq \mathrm{LP}^{(\ell)}$ we have $\mathrm{LP}^{(\ell)} \neq \emptyset$ and hence the DomToIP$(x^{(\ell)})$ has an optimal solution z^*.

We consider two cases. In the first case, we have $z_{\ell+1}^* = 0$. In this case we have $x_{\ell+1}^{(\ell+1)} = 0$. Since $z^* \leq x^{(\ell+1)}$, we have $z^* \in \mathrm{LP}^{(\ell+1)}$. Also, $z^* \in P$. By Theorem 2 there exists $\tilde{z}^i \in S$ and $\theta_i \geq 0$ for $i \in [k]$ such that $\sum_{i=1}^k \theta_i = 1$ and $\sum_{i=1}^k \theta_i \tilde{z}^i \leq gz^*$. We have $\sum_{i=1}^k \theta_i \tilde{z}^i \leq gz^* \leq gx^{(\ell+1)}$. So for $j \in [n]$ where $x_j^{(\ell+1)} = 0$, we have $z_j^i = 0$ for $i \in [k]$. This implies $\tilde{z}^i \leq x^{(\ell+1)}$ for $i = 1, \ldots, k$. Hence, there exists $z \in S$ such that $z \leq x^{(\ell+1)}$. We claim that $z \in \mathrm{IP}^{(\ell+1)}$. If $z \notin \mathrm{IP}^{(\ell+1)}$ we must have $1 \leq j \leq \ell$ such that $z_j < x_j^{(\ell+1)}$, and thus $z_j = 0$ and $x_j^{(\ell+1)} = 1$. Without loss of generality assume j is minimum number satisfying $z_j < x_j^{(\ell+1)}$. Consider iteration j of the DomToIP algorithm. Notice that $z \leq x^{(\ell+1)} \leq x^{(j)}$. We have $x_j^{(j)} = 1$ which implies when we solved DomToIP$(x^{(j-1)})$ the optimal value was strictly larger than zero. However, z is a feasible solution to DomToIP$(x^{(j-1)})$ and gives an objective value of 0. This is a contradiction, so $z \in \mathrm{IP}^{(\ell+1)}$.

Now for the second case, assume $z_{\ell+1}^* > 0$. We have $x_{\ell+1}^{(\ell+1)} = 1$. Notice that for each point $z \in \mathrm{LP}^{(\ell)}$ we have $z_{\ell+1} > 0$, so for each $z \in \mathrm{IP}^{(\ell)}$ we have $z_{\ell+1} > 0$, i.e. $z_{\ell+1} = 1$. This means that $z \in \mathrm{IP}^{(\ell+1)}$, and $\mathrm{IP}^{(\ell+1)} \neq \emptyset$.

Now consider $x^{(t)}$. Let z be the optimal solution to $\mathrm{LP}^{(t-1)}$. If $x_t^{(t)} = 0$, we have $x^{(t)} = z$, which implies that $x^{(t)} \in P$, and since $x^{(t)} \in \{0,1\}^n$ we have $x^{(t)} \in S$. If $x_t^{(t)} = 1$, it must be the case that $z_t > 0$. By the argument above there is a point $z' \in \mathrm{IP}^{(t-1)}$. We show that $x^{(t)} = z'$. For $j \in [t-1]$ we have $z'_j = x_j^{(t-1)} = x_j^{(t)}$. We just need to show that $z'_t = 1$. Assume $z'_t = 0$ for contradiction, then $z' \in \mathrm{LP}^{(t-1)}$ has objective value of 0 for DomToIP$(x^{(t-1)})$, this is a contradiction to z being the optimal solution. This concludes the proof of Lemma 6.

5 FDT on Binary IPs

Assume we are given a point $x^* \in P$. For instance, x^* can be the optimal solution of minimizing a cost function cx over set P, which provides a lower bound on $\min_{(x,y) \in S(I)} cx$. In this section, we prove Theorem 3 by describing the Fractional Decomposition Tree (FDT) algorithm. We also remark that if $g(I) = 1$, then the algorithm will give an exact decomposition of any feasible solution.

The FDT algorithm grows a tree similar to the classic branch-and-bound search tree for integer programs. Each node represents a partially integral vector $\bar{x}$ in $\mathcal{D}(P)$ together with a multiplier $\bar{\lambda}$. The solutions contained in the nodes of the tree become progressively more integral at each level. In each level of the tree, the algorithm maintain a conic combination of points with the properties mentioned above. Leaves of the FDT tree contain solutions with integer values for all the x variables that dominate a point in P. In Lemma 6 we saw how to turn these into points in S.

Branching on a node. We begin with the following lemmas that show how the FDT algorithm branches on a variable.

Lemma 7. *Given $x' \in \mathcal{D}(P)$ and $\ell \in [n]$ where $x'_\ell < 1$, we can find in polynomial time vectors $\hat{x}^0, \hat{x}^1$ and scalars $\gamma_0, \gamma_1 \in [0,1]$ such that: (i) $\gamma_0 + \gamma_1 \geq 1/g$, (ii) $\hat{x}^0$ and $\hat{x}^1$ are in P ,(iii) $\hat{x}^0_\ell = 0$ and $\hat{x}^1_\ell = 1$, (iv) $\gamma_0\hat{x}^0 + \gamma_1\hat{x}^1 \leq x'$.*

Proof. Consider the following linear program which we denote by LPC(ℓ, x'). The variables of LPC(ℓ, x') are γ_0, γ_1 and x^0 and x^1.

$$
\begin{aligned}
\text{LPC}(\ell, x') \qquad \max \quad & \lambda_0 + \lambda_1 && && (12)\\
\text{s.t.} \quad & Ax^j \geq b\lambda_j && \text{for } j = 0,1 && (13)\\
& 0 \leq x^j \leq \lambda_j && \text{for } j = 0,1 && (14)\\
& x^0_\ell = 0,\ x^1_\ell = \lambda_1 && && (15)\\
& x^0 + x^1 \leq x' && && (16)\\
& \lambda_0, \lambda_1 \geq 0 && && (17)
\end{aligned}
$$

Let x^0, x^1, and γ_0, γ_1 be an optimal solution solution to the LP above. Let $\hat{x}^0 = x^0/\gamma_0$, $\hat{x}^1 = x^1/\gamma_1$. This choice satisfies (ii), (iii), (iv). To show that (i) is also satisfied we prove the following claim.

Claim 1. *We have $\gamma_0 + \gamma_1 \geq 1/g$.*

Proof. We show that there is a feasible solution that achieves the objective value of $\frac{1}{g}$. By Theorem 2 there exists $\theta \in [0,1]^k$, with $\sum_{i=1}^k \theta_i = 1$ and $\tilde{x}^i \in S$ for $i \in [k]$ such that $\sum_{i=1}^k \theta_i \tilde{x}^i \leq gx'$. So

$$x' \geq \sum_{i=1}^{k} \frac{\theta_i}{g}\tilde{x}^i = \sum_{i \in [k]: \tilde{x}^i_\ell = 0} \frac{\theta_i}{g}\tilde{x}^i + \sum_{i \in [k]: \tilde{x}^i_\ell = 1} \frac{\theta_i}{g}\tilde{x}^i. \tag{18}$$

For $j = 0,1$, let $x^j = \sum_{i \in [k]: \tilde{x}^i_\ell = j} \frac{\theta_i}{g}\tilde{x}^i$. Also let $\lambda_0 = \sum_{i \in [k]: \tilde{x}^i_\ell = 0} \frac{\theta_i}{g}$ and $\lambda_1 = \sum_{i \in [k]: \tilde{x}^i_\ell = 1} \frac{\theta_i}{g}$. Note that $\lambda_0 + \lambda_1 = 1/g$. Constraint (16) is satisfied by Inequality (18). Also, for $j = 0,1$

we have

$$Ax^j = \sum_{i\in[k],\tilde{x}^i_\ell=j} \frac{\theta_i}{g} A\tilde{x}^i \geq b \sum_{i\in[k],\tilde{x}^i_\ell=j} \frac{\theta_i}{g} = b\lambda_j. \tag{19}$$

Hence, Constraints (13) holds. Constraint (15) also holds since x^0_ℓ is obviously 0 and $x^1_\ell = \sum_{i\in[k]:\tilde{x}^i_\ell=1} \frac{\theta_i}{g} = \lambda_1$. The rest of the constraints trivially hold. $\Diamond$

This concludes the proof of Lemma 7. $\square$

We now show if x' in the statement of Lemma 7 is partially integral, we can find solutions with more integral components.

Lemma 8. *Given $x' \in \mathcal{D}(P)$ where $x'_1, \ldots, x'_{\ell-1} \in \{0,1\}$ and $x'_\ell < 1$ for some $\ell \geq 1$ we can find in polynomial time vectors $\hat{x}^0, \hat{x}^1$ and scalars $\gamma_0, \gamma_1 \in [0,1]$ such that: (i) $1/g \leq \gamma_0 + \gamma_1 \leq 1$, (ii) $\hat{x}^0$ and $\hat{x}^1$ are in $\mathcal{D}(P)$, (iii) $\hat{x}^0_\ell = 0$ and $\hat{x}^1_\ell = 1$, (iv) $\gamma_0\hat{x}^0 + \gamma_1\hat{x}^1 \leq x'$,(v) $\hat{x}^i_j \in \{0,1\}$ for $i = 0,1$ and $j \in [\ell-1]$.*

Proof. By Lemma 7 we can find $\bar{x}^0$, $\bar{x}^1$, γ_0 and γ_1 that satisfy (i), (ii), (iii), and (iv). We define $\hat{x}^0$ and $\hat{x}^1$ as follows. For $i = 0,1$, for $j \in [\ell-1]$, let $\hat{x}^i_j = \lceil \bar{x}^i_j \rceil$, for $j = \ell, \ldots, t$ let $\hat{x}^i_j = \bar{x}^i_j$.

We now show that $\hat{x}^0$, $\hat{x}^1$, γ_0, and γ_1 satisfy all the conditions. Note that conditions (i), (ii), (iii), and (v) are trivially satisfied. Thus we only need to show (iv) holds. We need to show that $\gamma_0\hat{x}^0_j + \gamma_1\hat{x}^1_j \leq gx'_j$. If $j = \ell, \ldots, t$, then this clearly holds. Hence, assume $j \leq \ell - 1$. By the property of x' we have $x'_j \in \{0,1\}$. If $x'_j = 0$, then by Constraint (16) we have $\bar{x}^0_j = \bar{x}^1_j = 0$. Therefore, $\hat{x}^i_j = 0$ for $i = 0,1$, so (iv) holds. Otherwise if $x'_j = 1$, then we have $\gamma_0\hat{x}^0_j + \gamma_1\hat{x}^1_j \leq \gamma_0 + \gamma_1 \leq 1 \leq x'_j$. Therefore (v) holds. $\square$

Growing and Pruning FDT tree. The FDT algorithm maintains nodes L_i in iteration i of the algorithm. The nodes in L_i correspond to the nodes in level L_i of the FDT tree. The points in the leaves of the FDT tree, L_t, are points in $\mathcal{D}(P)$ and are integral for all integer variables.

Lemma 9. *There is a polynomial time algorithm that produces sets $L_0, \ldots, L_t$ of pairs of $x \in \mathcal{D}(P)$ together with multipliers λ with the following properties for $i = 0, \ldots, t$: (a) If $x \in L_i$, then $x_j \in \{0,1\}$ for $j \in [i]$, i.e. the first i coordinates of a solution in level i are integral, (b) $\sum_{[x,\lambda]\in L_i} \lambda \geq \frac{1}{g^i}$, (c) $\sum_{[x,\lambda]\in L_i} \lambda x \leq x^*$, (d) $|L_i| \leq t$.*

Proof. We prove this lemma using induction but one can clearly see how to turn this proof into a polynomial time algorithm. Let L_0 be the set that contains a single node (*root of the FDT tree*) with x^* and multiplier 1. It is easy to check all the requirements in the lemma are satisfied for this choice.

Suppose by induction that we have constructed sets $L_0, \dots, L_i$. Let the solutions in L_i be x^j for $j \in [k]$ and λ_j be their multipliers, respectively. For each $j \in [k]$ if $x^j_{i+1} = 1$ we add the pair (x^j, λ_j) to L'. Otherwise, applying Lemma 8 (setting $x' = x^j$ and $\ell = i + 1$) we can find x^{j0}, x^{j1}, λ^0_j and λ^1_j with the properties (i) to (v) in Lemma 8. Add the pairs $(x^{j0}, \lambda_j\lambda^0_j)$ and $(x^{j1}, \lambda_j\lambda^1_j)$ to L'. It is easy to check that set L' is a suitable candidate for L_{i+1}, i.e. set L' satisfies (a), (b) and (c). However we can only ensure that $|L'| \leq 2k \leq 2t$, and might have $|L'| > t$. We call the following linear program Pruning(L'). Let $L' = \{[x^1, \gamma_1], \dots, [x^{|L'|}, \gamma_{|L'|}]\}$. The variables of Pruning(L') are scalar variables θ_j for each node j in L'.

$$\text{Pruning}(L') \qquad \{\max \sum_{j=1}^{|L'|} \theta_j \; : \; \sum_{j=1}^{|L'|} \theta_j x^j_i \leq x^*_i \text{ for } i \in [t],\ \theta \geq 0\} \tag{20}$$

Notice that $\theta = \gamma$ is in fact a feasible solution to Pruning(L'). Let θ^* be the optimal vertex solution to this LP. Since the problem is in $\mathbb{R}^{|L'|}$, θ^* has to satisfy $|L'|$ linearly independent constraints at equality. However, there are only t constraints of type $\sum_{j=1}^{|L'|} \theta_j x^j_i \leq x^*_i$. Therefore, there are at most t coordinates of θ^*_j that are non-zero. Set L_{i+1} which consists of x^j for $j = 1, \dots, |L'|$ and their corresponding multipliers θ^*_j satisfy the properties in the statement of the lemma. Notice that, we can discard the nodes in L_{i+1} that have $\theta^*_j = 0$, so $|L_{i+1}| \leq t$. Also, since θ^* is optimal and γ is feasible for Pruning(L'), we have $\sum_{j=1}^{|L'|} \theta^*_j \geq \sum_{j=1}^{|L'|} \gamma_j \geq \frac{1}{g^{i+1}}$.

□

From leaves of FDT to feasible solutions. For the leaves of the FDT tree, L_t, we have that every solution x in L_t has $x \in \{0,1\}^n$ and $x \in \mathcal{D}(P)$. By applying Lemma 6 we can obtain a point $x' \in S$ such that $x' \leq x$. This concludes the description of the FDT algorithm and proves Theorem 3. See Algorithm 2 for a summary of the FDT algorithm.

Algorithm 2: Fractional Decomposition Tree Algorithm

Input: $P = \{x \in \mathbb{R}^n : Ax \geq b\}$ and $S = \{x \in P : x \in \{0,1\}^n\}$ such that $g = \max_{c \in \mathbb{R}^n_+} \frac{\min_{x \in S} cx}{\min_{x \in P} cx}$ is finite, $x^* \in P$

Output: $z^i \in S$ and $\lambda_i \geq 0$ for $i \in [k]$ such that $\sum_{i=1}^k \lambda_i = 1$, and $\sum_{i=1}^k \lambda_i z^i \leq g^t x^*$

$L^0 \leftarrow [x^*, 1]$
for $i = 1$ ***to*** t **do**
    $L' \leftarrow \emptyset$
    for $[x, \lambda] \in L^i$ **do**
        Apply Lemma 8 to obtain $[\hat{x}^0, \gamma_0]$ and $[\hat{x}^1, \gamma_1]$
        $L' \leftarrow L' \cup \{[\hat{x}^0, \lambda \cdot \gamma_0]\} \cup \{[\hat{x}^1, \lambda \cdot \gamma_1]\}$
    end
    Apply Lemma 9 to prune L' to obtain L^{i+1}.
end
for $[x, \lambda] \in L^t$ **do**
    Apply Algorithm 1 to x to obtain $z \in S$
    $F \leftarrow F \cup \{[z, \lambda]\}$
end
return F

6 FDT for 2EC

In Section 5 our focus was on binary IPs. In this section, in an attempt to extend FDT to {0,1,2} problems we introduce an FDT algorithm for a 2-edge-connected multigraph problem. Given a graph $G = (V, E)$ a multi-subset of edges F of G is a 2-edge-connected multigraph of G if for each set $\emptyset \subset U \subset V$, the number of edge in F that have one endpoint in U and one not in U is at least 2. Recall that in the 2EC, we are given non-negative costs on the edges of G and the goal is to find the minimum cost 2-edge-connected multigraph of G. The natural linear programming relaxation is $\min\{cx \ : \ x \in \text{Subtour}(G)\}$ where $\text{Subtour}(G) = \{x \in [0,2]^E : x(\delta(U)) \geq 2 \text{ for } \emptyset \subset U \subset V\}$. Notice that, no optimal solution ever takes 3 copies of an edge in 2EC, hence we assume that we can take an edge at most 2 times, hence in this chapter (unlike in the previous chapters) we work with a bounded version of $\text{Subtour}(G)$. Notice that $\mathcal{D}(\text{Subtour}(G)) \cap [0,2]^E = \text{Subtour}(G)$. Thus, we also assume a multigraph can contain at most 2 copies of any edge in the graph. We want to prove Theorem 5.

Theorem 5. *Let $G = (V, E)$ and x be an extreme point of* $\text{Subtour}(G)$*. The FDT algorithm* *for 2EC produces* $\lambda \in [0,1]^k$ *and 2-edge-connected multigraphs* $F_1, \ldots, F_k$ *such that* $k \leq 2|V|-1$, $\sum_{i=1}^k \lambda_i \chi^{F_i} \leq \min(Cx, \{2\}^n)$, *and* $\sum_{i=1}^k \lambda_i = 1$. *Moreover,* $C \leq g(\mathrm{2EC})^{|E_x|}$.

We do not know the exact value for $g(\mathrm{2EC})$, but we know $\frac{6}{5} \leq g(\mathrm{2EC}) \leq \frac{3}{2}$ [ABE06, Wol80]. The FDT algorithm for 2EC is very similar to the one for binary IPs, but there are some differences as well. A natural thing to do is to have three branches for each node of the FDT tree, however, the branches that are equivalent to setting a variable to 1, might need further decomposition. That is the main difficulty when dealing with $\{0,1,2\}$-IPs.

First, we need a branching lemma. Observe that the following branching lemma is essentially a translation of Lemma 7 for $\{0,1,2\}$ problems except for one additional clause.

Lemma 10. *Given* $x \in \mathrm{Subtour}(G)$, *and* $e \in E$ *we can find in polynomial time vectors* x^0, x^1 *and* x^2 *and scalars* γ_0, γ_1, *and* γ_2 *such that: (i)* $\gamma_0 + \gamma_1 + \gamma_2 \geq 1/g(\mathrm{2EC})$, *(ii)* x^0, x^1, *and* x^2 *are in* $\mathrm{Subtour}(G)$, *(iii)* $x^0_e = 0$, $x^1_e = 1$, *and* $x^2_e = 2$, *(iv)* $\gamma_0 x^0 + \gamma_1 x^1 + \gamma_2 x^2 \leq x$, *(v) for* $f \in E$ *with* $x_f \geq 1$, *we have* $x^j_f \geq 1$ *for* $j = 0,1,2$.

Proof. Consider the following LP with variables λ_j and x^j for $j = 0,1,2$.

$$
\begin{aligned}
\max \quad & \sum_{j=0,1,2} \lambda_j && && (21)\\
\text{s.t.} \quad & x^j(\delta(U)) \geq 2\lambda_j && \text{for } \emptyset \subset U \subset V, \text{ and } j = 0,1,2 && (22)\\
& 0 \leq x^j \leq 2\lambda_j && \text{for } j = 0,1,2 && (23)\\
& x^j_e = j \cdot \lambda_j && \text{for } j = 0,1,2 && (24)\\
& x^j_f \geq \lambda_j && \text{for } f \in E \text{ where } x_f \geq 1, \text{ and } j = 0,1,2 && (25)\\
& x^0 + x^1 + x^2 \leq x && && (26)\\
& \lambda_0, \lambda_1, \lambda_2 \geq 0 && && (27)
\end{aligned}
$$

Let x^j, γ_j for $j = 0,1,2$ be an optimal solution solution to the LP above. Let $\hat{x}^j = x^j/\gamma_j$ for $j = 0,1,2$ where $\gamma_j > 0$. If $\gamma_j = 0$, let $\hat{x}^j = 0$. Observe that (ii), (iii), (iv), and (v) are satisfied with this choice. We can also show that $\gamma_0 + \gamma_1 + \gamma_2 \geq 1/g(\mathrm{2EC})$, which means that (i) is also satisfied. The proof is similar to the proof of the claim in Lemma 7, but we need to replace each $f \in E$ with $x_f \geq 1$ with a suitably long path to ensure that Constraint (25) is also satisfied.

Claim 2. *We have* $\gamma_0 + \gamma_1 + \gamma_2 \geq \frac{1}{g(\mathrm{2EC})}$.

Proof. Suppose for contradiction $\sum_{j=0,1,2} \gamma_j = \frac{1}{g(\mathrm{2EC})} - \epsilon$ for some $\epsilon > 0$. Construct graph G' by removing edge f with $x_f \geq 1$ and replacing it with a path P_f of length $\lceil \frac{2}{\epsilon} \rceil$. Define $x'_h = x_h$ for each edge h such that $x_h < 1$. For each $h \in P_f$ let $x'_h = x_f$ for all f with

$x_f \geq 1$. It is easy to check that $x' \in \text{Subtour}(G')$. By Theorem 2 there exists $\theta \in [0,1]^k$, with $\sum_{i=1}^k \theta_i = 1$ and 2-edge-connected multigraphs F'_i of G' for $i = 1, \dots, k$ such that $\sum_{i=1}^k \theta_i \chi^{F'_i} \leq g(2\text{EC})x'$.

Note that each F'_i contains at least one copy of every edge in any path P_f, except for at most one edge in the path. We will obtain 2-edge-connected multigraphs $F_1, \dots, F_k$ of G using $F'_1, \dots, F'_k$, respectively. To obtain F_i first remove all P_f paths from F'_i. Suppose there is an edge h in P_f such that $\chi_h^{F'_i} = 0$, this means that for any edge $p \in P_f$ such that $p \neq h$, $\chi_p^{F'_i} = 2$. In this case, let $\chi_f^{F_i} = 2$, i.e. add two copies of f to F_i. If there are at least one edge $h \in P_f$ with $\chi_h^{F'_i} = 1$, let $\chi_f^{F_i} = 1$, i.e. add one copy of f to F_i. If for all edges $h \in P_f$, we have $\chi_h^{F'_i} = 2$, then let $\chi_f^{F_i} = 2$. For $f \in E$ with $x_f < 1$ we have

$$\sum_{i=1}^k \theta_i \chi_f^{F_i} = \sum_{i=1}^k \theta_i \chi_f^{F'_i} \leq g(2\text{EC})x'_f = g(2\text{EC})x_f. \tag{28}$$

In addition for $f \in E$ with $x_f \geq 1$ we have $\chi_f^{F_i} \leq \frac{\sum_{h \in P_f} \chi_h^{F'_i}}{\lceil \frac{2}{\epsilon} \rceil - 1}$ by construction.

$$\begin{aligned}
\sum_{i=1}^k \theta_i \chi_f^{F_i} &\leq \sum_{i=1}^k \theta_i \frac{\sum_{h \in P_f} \chi_h^{F'_i}}{\lceil \frac{2}{\epsilon} \rceil - 1} \\
&= \frac{\sum_{h \in P_f} \sum_{i=1}^k \theta_i \chi_h^{F'_i}}{\lceil \frac{2}{\epsilon} \rceil - 1} \\
&\leq \frac{\sum_{h \in P_f} g(2\text{EC})x'_h}{\lceil \frac{2}{\epsilon} \rceil - 1} \\
&= \frac{\sum_{h \in P_f} g(2\text{EC})x_f}{\lceil \frac{2}{\epsilon} \rceil - 1} \\
&= \frac{\lceil \frac{2}{\epsilon} \rceil}{\lceil \frac{2}{\epsilon} \rceil - 1} g(2\text{EC})x_f.
\end{aligned}$$

Therefore, since $\frac{\lceil \frac{2}{\epsilon} \rceil}{\lceil \frac{2}{\epsilon} \rceil - 1} \geq 1$, we have

$$x \geq \sum_{i \in [k]: \chi_e^{F_i} = 0} \frac{\theta_i (\lceil \frac{2}{\epsilon} \rceil - 1)}{g(2\text{EC}) \lceil \frac{2}{\epsilon} \rceil} \chi^{F_i} + \sum_{i \in [k]: \chi_e^{F_i} = 1} \frac{\theta_i (\lceil \frac{2}{\epsilon} \rceil - 1)}{g(2\text{EC}) \lceil \frac{2}{\epsilon} \rceil} \chi^{F_i} + \sum_{i \in [k]: \chi_e^{F_i} = 2} \frac{\theta_i (\lceil \frac{2}{\epsilon} \rceil - 1)}{g(2\text{EC}) \lceil \frac{2}{\epsilon} \rceil} \chi^{F_i}. \tag{29}$$

Let $x^j = \sum_{i \in [k]: \chi_e^{F_i} = j} \frac{\theta_i (\lceil \frac{2}{\epsilon} \rceil - 1)}{g(2\text{EC}) \lceil \frac{2}{\epsilon} \rceil} \chi^{F_i}$ and $\theta_j = \sum_{i \in [k]: \chi_e^{F_i} = j} \frac{\theta_i (\lceil \frac{2}{\epsilon} \rceil - 1)}{g(2\text{EC}) \lceil \frac{2}{\epsilon} \rceil}$ for $j = 0, 1, 2$. It is easy to check that x^j , θ_j for $j = 0, 1, 2$ is a feasible solution to the LP above. Notice that $\sum_{j=0,1,2} \theta_j = \frac{\lceil \frac{2}{\epsilon} \rceil - 1}{g(2\text{EC}) \lceil \frac{2}{\epsilon} \rceil}$. By assumption, we have $\frac{\lceil \frac{2}{\epsilon} \rceil - 1}{g(2\text{EC}) \lceil \frac{2}{\epsilon} \rceil} \leq \frac{1}{g(2\text{EC})} - \epsilon$, which is a contradiction. $\Diamond$

This concludes the proof. $\square$

In contrast to FDT for binary IPs where we round up the fractional variables that are already branched on at each level, in FDT for 2EC we keep all coordinates as they are and perform a rounding procedure at the end. Formally, let L_i for $i = 1, \ldots, |\operatorname{supp}(x^*)|$ be collections of pairs of feasible points in $\operatorname{Subtour}(G)$ together with their multipliers. Let $t = |\operatorname{supp}(x^*)|$ and assume without loss of generality that $\operatorname{supp}(x^*) = \{e_1, \ldots, e_t\}$.

Lemma 11. *The FDT algorithm for 2EC in polynomial time produces sets $L_0, \ldots, L_t$ of pairs $x \in \mathrm{2EC}(G)$ together with multipliers λ with the following properties for $i \in [t]$: (a) If $x \in L_i$, then $x_{e_j} = 0$ or $x_{e_j} \geq 1$ for $j = 1, \ldots, i$, (b) $\sum_{(x,\lambda)\in L_i} \lambda \geq \frac{1}{g(\mathrm{2EC})^i}$, (c) $\sum_{(x,\lambda)\in L_i} \lambda x \leq x^*$, (d) $|L_i| \leq t$.*

The proof is similar to Lemma 9, but we need to use property (v) in Lemma 10 to prove that (a) also holds.

Proof. We proceed by induction on i. Define $L_0 = \{(x^*, 1)\}$. It is easy to check all the properties are satisfied. Now, suppose by induction we have L_{i-1} for some $i = 1, \ldots, t$ that satisfies all the properties. For each solution x^ℓ in L_{i-1} apply Lemma 10 on x^ℓ and e_i to obtain $x^{\ell j}$ and $\lambda_{\ell j}$ for $j = 0, 1, 2$. Let L' be the collection that contains $(x^{\ell j}, \lambda_\ell \cdot \lambda_{\ell j})$ for $j = 0, 1, 2$, when applied to all (x^ℓ, λ_ℓ) in L_{i-1}. Similar to the proof in Lemma 9 one can check that L_i satisfies properties (b), (c). We now verify property (a). Consider a solution x^ℓ in L_{i-1}. For $e \in \{e_1, \ldots, e_{i-1}\}$ if $x^\ell_e = 0$, then by property (iv) in Lemma 10 we have $x^{\ell j} = 0$ for $j = 0, 1, 2$. Otherwise by induction we have $x^\ell_e \geq 1$ in which case property (v) in Lemma 10 ensures that $x^{\ell j}_e \geq 1$ for $j = 0, 1, 2$. Also, $x^{\ell j}_{e_i} = j$, so $x^{\ell j}_{e_i} = 0$ or $x^{\ell j}_{e_i} \geq 1$ for $j = 0, 1, 2$.

Finally, if $|L'| \leq t$ we let $L_i = L'$, otherwise apply $\operatorname{Pruning}(L')$ to obtain L_i. $\square$

Consider the solutions x in L_t. For each variable e we have $x_e = 0$ or $x_e \geq 1$.

Lemma 12. *Let x be a solution in L_t. Then $\lfloor x \rfloor \in \operatorname{Subtour}(G)$.*

Proof. Suppose not. Then there is a set of vertices $\emptyset \subset U \subset V$ such that $\sum_{e\in\delta(U)} \lfloor x_e \rfloor < 2$. Since $x \in \operatorname{Subtour}(G)$ we have $\sum_{e\in\delta(U)} x_e \geq 2$. Therefore, there is an edge $f \in \delta(U)$ such that x_f is fractional. By property (a) in Lemma 11, we have $1 < x_f < 2$. Therefore, there is another edge h in $\delta(U)$ such that $x_h > 0$, which implies that $x_h \geq 1$. But in this case $\sum_{e\in\delta(U)} \lfloor x_e \rfloor \geq \lfloor x_f \rfloor + \lfloor x_h \rfloor \geq 2$. This is a contradiction. $\square$

The FDT algorithm for 2EC iteratively applies Lemmas 10 and 11 to variables $x_1, \ldots, x_t$ to obtain leaf point solutions L_t. Finally, we just need to apply Lemma 12 to obtain the 2-edge-connected multigraphs from every solution in L_t. Notice that since x is an extreme point we have $t \leq 2|V| - 1$ [BP90]. By Lemma 11 we have

$$\sum_{(x,\lambda)\in L_t} \frac{\lambda}{\sum_{(x,\lambda)\in L_t} \lambda} \lfloor x \rfloor \leq \frac{1}{\sum_{(x,\lambda)\in L_t} \lambda} \sum_{(x,\lambda)\in L_t} \lambda x \leq g^t_{\text{2EC}} x^*.$$

7 Computational Experiments with FDT

We ran FDT on two network design problems: TAP and 2EC.

FDT on randomly generated instances of TAP. Recall that in TAP we are given a tree $T = (V, E)$, and a set of links L between vertices in V and costs $c \in \mathbb{R}^L_{\geq 0}$. A feasible augmentation is $L' \subseteq L$ such that $T + L'$ is 2-edge-connected. In TAP we wish to find the minimum-cost feasible augmentation. The integrality gap of the cut-LP for TAP is defined as

$$g(\text{TAP}) = \max_{c \in \mathbb{R}^L_{\geq 0}} \frac{\min_{x \in \text{TAP}(T,L)} cx}{\min_{x \in \text{CUT}(T,L)} cx}.$$

We know $\frac{3}{2} \leq g(\text{TAP}) \leq 2$ [FJ81, CKKK08]. Notice that $\min_{x \in \text{TAP}(T,L)} cx$ is a binary IP. We ran binary FDT on a set of 264 fractional extreme points of randomly generated instances of TAP. Table 1 shows FDT found solutions better than the integrality-gap lower bound for most instances.

	$C \in [1.1, 1.2]$	$C \in (1.2, 1.3]$	$C \in (1.3, 1.4]$	$C \in (1.4, 1.5]$
TAP	36	66	170	10

Table 1: The scale factor C for FDT run on 264 randomly generated TAP instances with fractional extreme points: 138 instances have 74 variables. The rest have 250.

Computational comparison between Christofides' algorithm and FDT for 2EC on Carr-Vempala points. We implemented the polyhedral version of Christofides' algorithm [Wol80]. In particular, we implemented the O-join augmentation in Christofides' algorithm, in a way that minimizes the average usage of every edge in the O-join augmentation across the convex combination of spanning trees. In particular, let $x \in \text{SEP}(G_x)$. It is easy to check that $\frac{n-1}{n} x \in \text{ST}(G_x)$, hence we can write $x = \sum_{i=1}^{k} \lambda_i \chi^{T_i}$ where T_i is spanning tree of G_x, $\sum_{i=1}^{k} \lambda_i = 1$, and $\lambda_i \geq 0$ for $i \in [k]$. Let O_i be the set of odd degree vertices of

T_i. We then solve the following LP that allows us to find parity corrections that are good for the whole convex combination.

$$\min\{\alpha : \sum_{i=1}^{k} \lambda_i y^i = \alpha \cdot x, \qquad (30)$$
$$y^i(\delta(U)) \geq 1 \text{ for } U \subseteq V(G_x), |V \cap O_i| \text{ odd}, \ y^i \in [0,1]^{E_x} \text{ for } i \in [k]\}.$$

The variables in the above LP are $y^i \in \mathbb{R}^{E_x}_{\geq 0}$ for $i \in [k]$. For each $i \in [k]$ we have $y^i \in \mathcal{D}(O_i\text{-}\mathrm{JOIN}(G_x))$. This formulation allows the instance specific approximation ratio of Christofides' algorithm to be below $\frac{3}{2}$. Recall that a Carr-Vempala point consists of a Hamiltonian cycle of fractional edges. Figure 1 shows FDT's solutions on all Carr-Vempala points with at most 10 vertices on the Hamiltonian cycle formed by the fractional edges are always better than those from the polyhedral version of Christofides' algorithm. In more details, in Figure 1 the horizontal axis of the plot is indexed with the 60 Carr-Vempala points that we considered. For each Carr-Vempala point x, there are two data points. The value of the first data point depicted by a circle on the vertical axis is $\frac{n-1}{n} + \alpha$ where n is the number of vertices in the Hamiltonian cycle formed by fractional edges of x and α is the optimal solution to (30). The value of the second data point depicted by a cross on the vertical axis is C where C is obtained from applying Theorem 5 to x. In other words, Figure 1 is comparing the upper bounds on the instances specific integrality gap certified by Christofides' algorithm and FDT algorithm for 2EC.

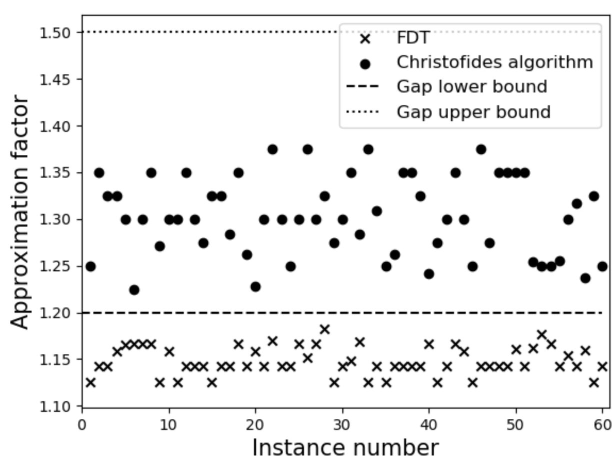

Figure 1: Polyhedral version of Christofides' algorithm vs FDT on all Carr-Vempala points with 10 vertices on the Hamiltonian cycle of the fractional-edges.

FDT for 2EC on Carr-Vempala points. We ran FDT for 2EC on 963 fractional extreme points of Subtour(G). We enumerated all (fractional) Carr-Vempala points with 10 and 12 vertices. Table 2 shows that again FDT found solutions better than the integrality-gap lower bound for most instances.

	$C \in [1.08, 1.11]$	$C \in (1.11, 1.14]$	$C \in (1.14, 1.17]$	$C \in (1.17, 1.2]$
2EC	79	201	605	78

Table 2: FDT for 2EC implemented applied to all Carr-Vempala with 10 or 12 vertices. A Carr-Vempala point with k vertices has $\frac{3k}{2}$ edges. Thus, the upper bound provided by Theorem 5 is $g(\text{2EC})^{3k/2}$. The lower bound on $g(\text{2EC})$ is $\frac{6}{5}$.

References

[ABE06] Anthony Alexander, Sylvia Boyd, and Paul Elliott-Magwood. On the integrality gap of the 2-edge connected subgraph problem. Technical report, TR-2006-04, SITE, University of Ottawa, 2006.

[BB08] Geneviève Benoit and Sylvia Boyd. Finding the exact integrality gap for small Traveling Salesman Problems. *Mathematics of Operations Research*, 33(4):921–931, 2008.

[BC11] Sylvia Boyd and Robert Carr. Finding low cost TSP and 2-matching solutions using certain half-integer subtour vertices. *Discrete Optimization*, 8(4):525 – 539, 2011.

[BP90] S Boyd and W. R. Pulleyblank. Optimizing over the subtour polytope of the travelling salesman problem. *Mathematical Programming*, 49:163–187, 1990.

[CCZ14] Michele Conforti, Gerard Cornuejols, and Giacomo Zambelli. *Integer Programming*. Springer Publishing Company, Incorporated, 2014.

[CKKK08] J. Cheriyan, H. Karloff, R. Khandekar, and J. Könemann. On the integrality ratio for tree augmentation. *Operations Research Letters*, 36(4):399 – 401, 2008.

[CR98] Robert Carr and R. Ravi. A new bound for the 2-edge connected subgraph problem. In *IPCO*, 1998.

[CV04] Robert Carr and Santosh Vempala. On the Held-Karp relaxation for the asymmetric and symmetric traveling salesman problems. *Mathematical Programming*, 100(3):569–587, Jul 2004.

[Edm65] Jack Edmonds. Maximum matching and a polyhedron with 0, 1-vertices. *Journal of Research of the National Bureau of Standards B*, 69:125–130, 1965.

[Edm70] Jack Edmonds. *Submodular Functions, Matroids, and Certain Polyhedra*, pages 11–26. Springer Berlin Heidelberg, Berlin, Heidelberg, 1970.

[FGL05] Matteo Fischetti, Fred Glover, and Andrea Lodi. The feasibility pump. *Mathematical Programming*, 104(1):91–104, Sep 2005.

[FJ81] Greg N. Frederickson and Joseph. Ja'Ja'. Approximation algorithms for several graph augmentation problems. *SIAM Journal on Computing*, 10(2):270–283, 1981.

[FS09] Matteo Fischetti and Domenico Salvagnin. Feasibility pump 2.0. *Mathematical Programming Computation*, 1(2):201–222, Oct 2009.

[GJ90] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., USA, 1990.

[GLS93] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2. Second corrected edition edition, 1993.

[Kar84] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, Dec 1984.

[Kha80] L.G. Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1):53 – 72, 1980.

[Vaz01] Vijay V. Vazirani. *Approximation Algorithms*. Springer-Verlag, Berlin, Heidelberg, 2001.

[Wol80] Laurence A. Wolsey. *Heuristic analysis, linear programming and branch and bound*, pages 121–134. Springer Berlin Heidelberg, Berlin, Heidelberg, 1980.

[WS11] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, USA, 1st edition, 2011.