FDT*

Robert Carr[1], Arash Haddadan[2], and Cynthia Phillips[3]

[1] University of New Mexico, Albuquerque NM 87131, USA
bobcarr@unm.

[2] Carnegie Mellon University, Pittsburgh, PA 15213, USA
ahaddada@cmu.edu

[3] Sandia National Laboratories, Albuquerque, NM 87185, USA
caphill@sandia.gov

Abstract. We present a new algorithm for finding a feasible solution for a mixed-integer linear program. The algorithm runs in polynomial time and is guaranteed to find a feasible integer solution provided the integrality gap is bounded. The algorithm is based on convex decomposition of scaled linear- programming relaxations, so in general it provides a suite of integer solutions. Because of the relationship between convex decomposition and integrality gaps, this algorithm can be seen as an approximation algorithm. We apply our algorithm our algorithm to a class of fractional extreme points for two known problems in combinatorial optimization: the 2-edge-connected spanning subgraph problem and the tree augmentation problem. These computational experiments show that our algorithm can be used as a tool to evaluate the integrality gap of integer programs with their linear relaxation. Finally we provide a stronger characterization of integrality gap for a class of covering problems than that of [11].

Keywords: Mixed-integer linear programming · Integrality gap · convex combinations.

1 Introduction

Mixed-integer linear programming (MILP), the optimization of a linear objective function subject to linear and integrality constraints, is a classic NP-hard problem [10]. In fact it is NP-hard in general to determine if there is a feasible solution to an MILP problem [10]. However, intelligent branch-and-bound strategies allow commercial and open-source MIP solvers to give exact solutions (or near-optimal with provable bound) to many specific instances of NP-hard combinatorial optimization problems. Modeling combinatorial optimization problems with MILPs also started the very popular area of LP-based approximation algorithms.

Consider the set of points described by a set

$$S(A,G,b) = \{(x,y) \in \mathbb{Z}^n \times \mathbb{R}^p \ : \ Ax + Gy \geq b\} \tag{1}$$

* Supported by organization x.

and the linear relaxation of set $S(A,G,b)$,

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

To make the notation simpler we use $I = (A,G,b)$, i.e. an instance of the formulation. Then we use $S(I)$ and $P(I)$ to denote $S(A,G,b)$ and $P(A,G,b)$, respectively. Notice that $\min\{cx : \ (x,y) \in S(I)\}$ is in fact an MILP problem and $\min\{cx : \ (x,y) \in P(I)\}$ is lower bound provided from the LP relaxation. Let $z_{IP}(I,c)$ and $z_{LP}(I,c)$ be the optimal solution to these problems, respectively.

Many researchers (see [16,17]) have developed polynomial time algorithms that find solutions for special classes of MILPs whose cost are provably smaller than $C \cdot z_{LP}(I,c)$ for some C which can be a constant number or dependent on some parameter of the MILP, e.g. $O(\log(n))$. However, for many combinatorial optimization problems there is a limit to such techniques. Let $g_I = \max_{c \geq 0} \frac{z_{IP}(I,c)}{z_{LP}(I,c)}$. Parameter g_I depends on set $S(I)$ and the linear constraints in (1) and is known as the *integrality gap* of the formulation for instance I. It is easy to see, that we cannot hope to find solutions for the MILP 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 $\mathcal{I}$. In this case, the integrality gap for problem $\mathcal{I}$ is

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

For example, in the problem of finding a minimum weight 2-edge-connected multigraph, the gap of the the natural formulation, constraining every cut to be crossed at least twice, is at most $\frac{3}{2}$ [18] and at least $\frac{6}{5}$ [1]. Therefore, we cannot hope to obtain an LP-based $(\frac{6}{5} - \epsilon)$-approximation algorithm for this problem using this LP relaxation. If we have $g_I = 1$, $P(I) = \text{conv}(S(I))$ and $P(I)$ is an *integral* polyhedron. There are many interesting examples of such formulations such as spanning tree polytope and the perfect matching polytope [15]. For such problems we know how to write a vector $x \in P(I)$ as convex combination of points in $S(I)$ in polynomial time [12].

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

Now assume $1 < g(I) < \infty$. Denote by $\mathcal{D}(P(I))$ be the set of points (x', y') such that there exists a point $(x,y) \in P$ with $x' \geq x$, also known as the dominant of $P(I)$. Notice that for covering problems the dominant of a polyhedron is essentially the same as its dominant, but for problems that are not covering, the two might be different. We show later, for I with $g(I) < \infty$, the notation of dominant is in fact useful.

A analogous result to Proposition 1 is the following theorem due to Carr and Vempala [2], which is a generalization of Goemans proof for blocking polyhedra [11].

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

In contrast to Proposition 1 which implies exact algorithms for problems with a gap of 1, Theorem 1 does not imply an approximation algorithm, since it does not suggest how to find such a convex combination in polynomial time. This points to an interesting open question.

Question 1. Assume reasonable complexity assumptions (such as UGC or P $\neq$ NP). Given instance I with $1 < g_I < \infty$ and $(x, y) \in P(I)$, can we find $\theta \in [0,1]^k$, where $\sum_{i=1}^k \theta_i = 1$ and $(\tilde{x}^i, \tilde{y}^i) \in \mathcal{D}(\text{conv}(S(I)))$ for $i = 1, \ldots, k$ such that $\sum_{i=1}^k \theta_i \tilde{x}^i \leq g_I x$ in polynomial time?

This seems to be a very hard question. A more specific question is of more interest.

Question 2. Assume reasonable complexity assumptions, a specific problem $\mathcal{I}$ with $1 < g_{\mathcal{I}} < \infty$, and $(x, y) \in P(I)$ for some $I \in \mathcal{I}$, can we find $\theta \in [0,1]^k$, where $\sum_{i=1}^k \theta_i = 1$ and $(\tilde{x}^i, \tilde{y}^i) \in S(I)$ for $i = 1, \ldots, k$ such that $\sum_{i=1}^k \theta_i \tilde{x}^i \leq g_{\mathcal{I}} x$ in polynomial time?

Although Question 2 is wide open, for some problems there are polynomial time algorithms closing the gap. For example, for generalized Steiner forest problem [13] gave an LP-based 2-approximation algorithm. The gap for this problem is also lower bounded by 2. Same holds for the set covering problem [14]. In fact, for set cover the approximation algorithm achieving the same factor as the integrality gap lower bound, is a *randomized rounding* algorithm. Raghavan and Thompson [14] showed that this technique achieves provably good approximation for many combinatorial optimization problems.

If we relax Question 1 (resp. Question 2), but multiplying $g(I)$ (resp. $g(\mathcal{I})$) by a factor C, they are still very interesting, since they will provide upper bounds on the integraltiy gap the instance (resp. the problem). The results in this paper serve this purpose.

We give a general approximation framework for solving $\{0,1\}$-MILPs. Consider the set of point described by a set

$$S(I) = \{(x, y) \in \mathbb{R}^{n \times p} \ : \ Ax + Gy \geq b, \ x \in \{0,1\}\}, \tag{4}$$

and a linear relaxation of set S

$$P(I) = \{(x, y) \in \mathbb{R}^{n \times p} \ : \ Ax + Gy \geq b, \ 0 \leq x \leq 1\}. \tag{5}$$

For a vector $x \in \mathbb{R}^n$ such that $(x, y) \in P(I)$ for some $y \in \mathbb{R}^p$, let $\text{supp}(x) = \{i \in \{1, \ldots, n\} : x_i \neq 0\}$.

The *Fractional Decomposition Tree* algorithm (henceforth FDT) is a polynomial time algorithm that given a point $x \in P(I)$ produces a convex combination of feasible point in $S(I)$ that are dominates by a "factor" of x. If this factor is g_I, it would settle Question 2, however we can only guarantee a factor of $g_I^{|\operatorname{supp}(x)|}$. FDT relies on iteratively solving linear programs that are about the same size as the description of $P(I)$.

Theorem 2. *Assume $1 \leq g_I < \infty$. The Fractional Decomposition Tree (FDT) algorithm , given $(x^*, y^*) \in P(I)$, produces in polynomial time $\lambda \in [0,1]^k$ and $(z^1, w^1), \ldots, (z^k, w^k) \in S(I)$ such that $k \leq |\operatorname{supp}(x^*)|$, $\sum_{i=1}^k \lambda_i z^i \leq Cx^*$, and $\sum_{i=1}^k \lambda_i = 1$. Moreover, $C \leq g_I^{|\operatorname{supp}(x^*)|}$.*

Notice, that the FDT algorithm needs to find feasible solutions to any MILP with finite gap. This can be of independent interest.

Theorem 3. *Assume $1 \leq g_I < \infty$. The DomToIP algorithm finds $(\hat{x}, \hat{y}) \in S(I)$ in polynomial time.*

Note that for general I it is NP-hard to even decide if $S(I)$ is empty or not. There are a number of heuristics for this purpose, such as the feasibility pump heuristic [8,9]. These heuristics are often very effective and fast in practice, however, they can sometimes fail in finding a feasible solution. In addition these algorithms do not provide any bounds on the quality of the solution they find.

One can trivially extend the FDT algorithm for binary MILPs into covering $\{0,1,2\}$-MILPs by losing a factor $2^{|\operatorname{supp}(x)|}$. In order to eradicate this factor, we need to treat the coordinate i with $x_i = 1$ differently. For the 2-edge-connected subgraph problem we show how this can be done. In this problem we want to find the minimum weight 2-edge-connected subgraph (which can multiple copies of each edge) in a graph $G = (V, E)$ with respect to weights $c \in \mathbb{R}^E_{\geq 0}$. For this problem the natural linear programming relaxation is

$$2\mathrm{EC}(G) = \{x \in \mathbb{R}^E_{\geq 0} \ : \ x(\delta(S)) \geq 2 \text{ for } \emptyset \subset S \subset V)\}. \tag{6}$$

We prove the following theorem.

Theorem 4. *Let $G = (V, E)$ and x be an extreme point of $2\mathrm{EC}(G)$. The FDT algorithm for 2EC produces $\lambda \in [0,1]^k$ and 2-edge-connected multigraphs $F_1, \ldots, F_k$ such that $k \leq 2|V| - 1$, $\sum_{i=1}^k \lambda_i \chi^{F_i} \leq Cx^*$, and $\sum_{i=1}^k \lambda_i = 1$. Moreover, $C \leq g(2\mathrm{EC})^k$, where $g(2\mathrm{EC})$ is the integrality gap of the 2-edge-connected multigraph problem with respect to formulation in (6)*

Although the bound guaranteed in both Theorems 2 and 4 are very large for large problems, we show that in practice, the algorithm works very well providing approximation ratios that are far better than the theoretical bound in the theorem statements. We examine FDT for binary MILPs for problems such as the tree augmentation problem (TAP) and we apply FDT for 2EC on some interesting and "hard to decompose" points in the linear relaxation. Our computational results show that the FDT algorithm is a good tool to evaluate the integrality gap of integer programming formulations.

Evaluating the integrality gap for instances of interesting problems such as TSP or 2EC have been done by others as well. For instance, Benoit and Boyd [6] used a quadratic programming model to show that the integrality gap for TSP is at most $\frac{20}{17}$ for instances with at most 10 vertices. Also, Alexander et. al [5] used the same ideas to provide an upper bound of $\frac{7}{6}$ for 2EC on instances with at most 10 vertices. For 2EC we can show that the integrality gap of 2EC is at most $\frac{6}{5}$ for fundamental vertices with 12 vertices. In fact, since FDT can be applied to different problem, we can use it to evaluate the integrality gap of other well-known problem. For example, for TAP, we create random fractional extreme points 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.

2 FDT on binary MIPs

In this section we present the fractional decomposition tree (FDT) algorithm and prove its correctness.

Consider a formulation instance $I = (A, G, b)$. Define sets $S(I)$ and $P(I)$ as in (4) and (5), respectively. Also assume we are given a point $(x^*, y^*) \in P(I)$. For instance, (x^*, y^*) 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 2. One key ingredient in proving Theorem 2 is Theorem 3 that we will also prove in this section.

Although the theoretical worst-case upper bound on C in the statement of Theorem 2 can be very large, we will show that in practice C can be really small and hence FDT can provide good LP-based approximation algorithms in many cases. We also remark that if $g_I = 1$, then the algorithm will give an exact decomposition of any feasible solution. In the remainder of this section we explain the FDT algorithm and prove Theorem 2. For simplicity in the notation we denote $P(I), S(I)$, and g_I with P, S, and g, respectively. Let $t = |\operatorname{supp}(x^*)|$. We assume without loss of generality that $\operatorname{supp}(x^*) = \{1, \ldots, t\}$.

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 $(\bar{x}, \bar{y})$ 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. We will later see how we can turn these into points in S. We begin with the following lemmas that show how the FDT algorithm branches on a variable.

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

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

$$
\begin{aligned}
\text{Branching}(\ell, x', y') \qquad \max \quad & \lambda_0 + \lambda_1 && && (7)\\
\text{s.t.} \quad & Ax^j + Gy^j \geq b\lambda_j && \text{for } j = 0,1 && (8)\\
& 0 \leq x^j \leq \lambda_j && \text{for } j = 0,1 && (9)\\
& x^0_\ell = 0,\ x^1_\ell = \lambda_1 && && (10)\\
& x^0 + x^1 \leq x' && && (11)\\
& \lambda_0, \lambda_1 \geq 0 && && (12)
\end{aligned}
$$

Let $(x^0, y^0), (x^1, y^1)$, and γ_0, γ_1 be an optimal solution solution to the LP above. Let $(\hat{x}^0, \hat{y}^0) = (\frac{x^0}{\gamma_0}, \frac{y^0}{\gamma_0})$, $(\hat{x}^1, \hat{y}^1) = (\frac{x^1}{\gamma_1}, \frac{y^1}{\gamma_1})$. Observe that (ii), (iii), (iv) are satisfied with this choice. In order to show that (i) is also satisfied we prove the following claim.

Claim. We have $\gamma_0 + \gamma_1 \geq \frac{1}{g}$.

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

$$
x' \geq \sum_{i=1}^{k} \frac{\theta_i}{g} \tilde{x}^i = \sum_{i \in [k]: \tilde{x}^i_\ell = 0} \frac{\theta_i}{g} \tilde{x}^i + \sum_{i \in [k]: \tilde{x}^i_\ell = 1} \frac{\theta_i}{g} \tilde{x}^i \tag{13}
$$

For $j = 0,1$, let $(x^j, y^j) = \sum_{i \in [k]: \tilde{x}^i_\ell = j} \frac{\theta_i}{g} (\tilde{x}^i, \tilde{y}^i)$. 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 $\lambda_0 + \lambda_1 = \frac{1}{g}$. Constraint (11) is satisfied by Inequality (13). Also, for $j = 0,1$ we have

$$
Ax^j + Gy^j = \sum_{i \in [k], \tilde{x}^i_\ell = j} \frac{\theta_i}{g} (A\tilde{x}^i + G\tilde{y}^i) \geq b \sum_{i \in [k], \tilde{x}^i_\ell = j} \frac{\theta_i}{g} = b\lambda_j. \tag{14}
$$

Hence, Constraints (8) holds. Constraint (10) 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. This concludes the proof.

We now show if x' in the statement of Lemma 1 is partially integral, we can find solutions with more integral components.

Lemma 2. *Given $(x', y') \in \mathcal{D}(P)$, such that $x'_1, \ldots, x'_{\ell-1} \in \{0,1\}$ for some $\ell \geq 1$, we can find in polynomial time vectors $(\hat{x}^0, \hat{y}^0), (\hat{x}^1, \hat{y}^1)$ and scalars $\gamma_0, \gamma_1 \in [0,1]$ such that: (i) $\frac{1}{g} \leq \gamma_0 + \gamma_1 \leq 1$, (ii) $(\hat{x}^0, \hat{y}^0)$ and $(\hat{x}^1, \hat{y}^1)$ 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 $j = 1, \ldots, \ell - 1$.*

By Lemma 1 we can find $(\bar{x}^0, \bar{y}^0)$, $(\bar{x}^1, \bar{y}^1)$, 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 $j = 1, \ldots, \ell - 1$, 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 $(\hat{x}^0, \bar{y}^0)$, $(\hat{x}^1, \bar{y}^1)$, 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 $\gamma_0 \hat{x}^0_j + \gamma_1 \hat{x}^1_j \leq g x'_j$. 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 (11) 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.

Let us define the FDT algorithm more formally. The 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 3. *There is a polynomial time algorithm that produces sets $L_0, \ldots, L_t$ of pairs of $(x, y) \in \mathcal{D}(P)$ together with multipliers λ with the following properties for $i = 0, \ldots, t$: (a) If $(x, y) \in L_i$, then $x_j \in \{0, 1\}$ for $j = 1, \ldots, i$, i.e. the first i coordinates of a solution in level i are integral, (b) $\sum_{[(x,y),\lambda] \in L_i} \lambda \geq \frac{1}{g^i}$, (c) $\sum_{[(x,y),\lambda] \in L_i} \lambda x \leq x^*$, (d) $|L_i| \leq t$.*

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^*, y^*) 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 $L_0, \ldots, L_i$. Let the solutions in L_i be (x^j, y^j) for $j = 1, \ldots, k$ and λ_j be their multipliers, respectively. For each $j = 1, \ldots, k$ by Lemma 2 (setting $(x', y') = (x^j, y^j)$ and $\ell = i + 1$) we can find $(x^{j0}, y^{j0}), (x^{j1}, y^{j1})$ and λ^0_j, λ^1_j with the properties (i) to (v) in Lemma 2. Define L' to be the set of nodes with solutions $(x^{j0}, y^{j0}), (x^{j1}, y^{j1})$ and multipliers $\lambda_j \lambda^0_j$, $\lambda_j \lambda^1_j$, respectively, for $j = 1, \ldots, k$. 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 $L' = \{[(x^1, y^1), \gamma_1], \ldots, [(x^{2k}, y^{2k}), \gamma_{2k}]\}$. The variables of Pruning(L') is a scalar variable θ_j for each node j in L'.

$$\text{Pruning}(L') \qquad \max \quad \sum_{j=1}^{2k} \theta_j \tag{15}$$

$$\text{s.t.} \quad \sum_{j=1}^{2k} \theta_j x^j_i \leq x^*_i \qquad \text{for } i = 1, \ldots, t \tag{16}$$

$$\theta \geq 0 \tag{17}$$

Notice that $\theta = \gamma$ is in fact a feasible solution to Pruning(L'). Let θ^* be the optimal vertex point solution to this LP. Since the problem is in $\mathbb{R}^{2k}$, θ^* has to satisfy $2k$ linearly independent constraints at equality. However, there are only

t constraints of type (16). Therefore, there are at most t coordinates of θ^*_j that are non-zero. We claim that L_{i+1} which consists of (x^j, y^j) for $j = 1, \dots, 2k$ 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 $\sum_{j=1}^{k} \theta^*_j \geq \sum_{j=1}^{2k} \gamma_j \geq \frac{1}{g^{i+1}}$. The leaves of the FDT tree, L_t, have the property that every solution (x, y) in L_t has $x \in \{0,1\}^n$ and $(x, y) \in \mathcal{D}(P)$. Our goal to now replace these leaves solutions in $\mathcal{D}(P)$ with solutions in S. To this end, we need to reduce some variables with value of 1 into variables of value 0 and obtain feasibility in the meantime. We introduce a procedure that given a point $(x, y) \in \mathcal{D}(P)$ with $x \in \{0,1\}^n$ outputs a point $z \leq x$ with $z \in \{0,1\}^n$ such that (z, w) is in S for some vector $w \in \mathbb{R}^p$.

The algorithm "fixes" the variables iteratively, from x_1 to x_t. Suppose we run the algorithm for $\ell \in \{0, \dots, t-1\}$ iterations and we have $(x^{(\ell)}, y^{(\ell)}) \in \mathcal{D}(P)$ such that $x^{(\ell)}_i \in \{0,1\}$ for $i = 1, \dots, t$. Now consider the following linear program. The variables of this LP are the $z \in \mathbb{R}^n$ variables and $w \in \mathbb{R}^p$.

$$\text{DominantToFeasbile}(x^{(\ell)}) \qquad \min \quad z_{\ell+1} \tag{18}$$
$$\text{s.t.} \quad Az + Gw \geq b \tag{19}$$
$$z_j = x^{(\ell)}_j \quad j = 1, \dots, \ell \tag{20}$$
$$z_j \leq x^{(\ell)}_j \quad j = \ell+1, \dots, n \tag{21}$$
$$z \geq 0 \tag{22}$$

If the optimal value to DominantToFeasbile$(x^{(\ell)})$ 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 $x^{(\ell+1)}_j = x^{(\ell)}_j$ for $j \in \{1, \dots, n\} \setminus \{\ell+1\}$. This procedure is the DomToIP algorithm. The following lemma proves Theorem 3.

Lemma 4. *Given $(x, y) \in \mathcal{D}(P)$, DomToIP algorithm correctly finds $(x^{(t)}, y^{(t)}) \in S$ in polynomial time.*

First, we need to show that in any iteration $\ell = 0, \dots, t-1$ of Algorithm **??** the DominantToFeasbile(x^ℓ) is feasible. We show something stronger. For $\ell = 0, \dots, t-1$ let

$$\text{LP}^{(\ell)} = \{(z, w) \in P \ : \ z \leq x^{(\ell)} \text{ and } z_j = x^{(\ell)}_j \text{ for } j = 1, \dots, \ell\}, \text{ and}$$
$$\text{IP}^{(\ell)} = \{(z, w) \in \text{LP}^{(\ell)} \ : \ z \in \{0,1\}^n\}.$$

Notice that if LP$^{(\ell)}$ is a non-empty set then DominantToFeasbile$(x^{(\ell)})$ is feasible. We show by induction on ℓ that LP$^{(\ell)}$ and IP$^{(\ell)}$ are not empty sets for $\ell = 0, \dots, t-1$. First notice that LP$^{(0)}$ is clearly feasible since by definition $(x^{(0)}, y^{(0)}) \in \mathcal{D}(P)$, meaning there exists $(z, w) \in P$ such that $z \leq x^{(0)}$. By Theorem 1, there exists $(\tilde{z}^i, \tilde{w}^i) \in S$ and $\theta_i \geq 0$ for $i = 1, \dots, k$ 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 $x^{(0)}_j = 0$, then $\sum_{i=1}^{k} \theta_i \tilde{z}^i_j = 0$, which implies that $\tilde{z}^i_j = 0$ for all $i = 1, \ldots, k$ and $j = 1, \ldots, n$ where $x^{(0)}_j = 0$. Hence, $z^i \leq x^{(0)}$ for $i = 1, \ldots, k$. Therefore $(\tilde{z}^i, \tilde{w}^i) \in \mathrm{IP}^{(0)}$ for $i = 1, \ldots, k$, which implies $\mathrm{IP}^{(0)} \neq \emptyset$.

Now assume $\mathrm{IP}^{(\ell)}$ is non-empty for some $\ell \in \{0, \ldots, t-2\}$. Since $\mathrm{IP}^{(\ell)} \subseteq \mathrm{LP}^{(\ell)}$ we have $\mathrm{LP}^{(\ell)} \neq \emptyset$ and hence the DominantToFeasbile$(x^{(\ell)})$ has an optimal solution (z^*, w^*).

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 $(z^*, w^*) \in \mathrm{LP}^{(\ell+1)}$. Also, $(z^*, w^*) \in P$. By Theorem 1 there exists $(\tilde{z}^i, \tilde{w}^i) \in S$ and $\theta_i \geq 0$ for $i = 1, \ldots, k$ such that $\sum_{i=1}^{k} \theta_i = 1$ and $\sum_{i=1}^{k} \theta_i \tilde{z}^i \leq g z^*$. We have

$$\sum_{i=1}^{k} \theta_i \tilde{z}^i \leq g z^* \leq g x^{(\ell+1)} \tag{23}$$

So for $j \in \{1, \ldots, n\}$ where $x^{(\ell+1)}_j = 0$, we have $z^i_j = 0$ for $i = 1, \ldots, k$. Hence, $\tilde{z}^i \leq x^{(\ell+1)}$ for $i = 1, \ldots, k$. Hence, there exists $(z, w) \in S$ such that $z \leq x^{(\ell+1)}$. We claim that $(z, w) \in \mathrm{IP}^{(\ell+1)}$. If $(z, w) \notin \mathrm{IP}^{(\ell+1)}$ we must have $1 \leq j \leq \ell$ such that $z_j < x^{(\ell+1)}_j$, and thus $z_j = 0$ and $x^{(\ell+1)}_j = 1$. Without loss of generality assume j is minimum number satisfying $z_j < x^{(\ell+1)}_j$. Consider iteration j of Algorithm **??**. Notice that $z \leq x^{(\ell+1)} \leq x^{(j)}$. We have $x^{(j)}_j = 1$ which implies when we solved DominantToFeasbile$(x^{(j-1)})$ the optimal value was strictly larger than zero. However, (z, w) is a feasible solution to DominantToFeasbile$(x^{(j-1)})$ and gives an objective value of 0. This is a contradiction, so $(z, w) \in \mathrm{IP}^{(\ell+1)}$.

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 $(z, w) \in \mathrm{IP}^{(\ell+1)}$, and $\mathrm{IP}^{(\ell+1)} \neq \emptyset$.

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

3 Computational experiments with FDT

We implement FDT to two covering problem. First, the tree argumentation problem (TAP) where given a tree $T = (V, E)$, and a set of link L between the vertices in V and costs $c \in \mathbb{R}^L_{\geq 0}$ we with to find minimum cost subset L' of L such that $T + L'$ is 2-edge-connected. For $\ell \in L$, let P_ℓ be the

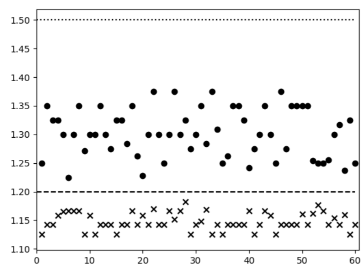

Fig. 1. Christofides' algorithm vs FDT on all fundamental extreme points with 10 vertices.

set of edges in the unique path between the endpoints of ℓ in T. For TAP, $S(\mathrm{TAP}) = \{x \in \mathbb{Z}^L_{\geq 0} : \sum_{\ell: e \in P_\ell} x_\ell \geq 1, \text{ for } e \in E\}$. Relaxing the integrality constraint we get $P(\mathrm{TAP})$. It is been shown that $\frac{3}{2} \leq g(\mathrm{TAP}) \leq 2$ [3,4]. We applied the binary FDT algorithm on a set of 264 fractional extreme points of $P(\mathrm{TAP})$. The result are summarized in Table 1. Next we implemented the

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

Table 1. FDT implemented applied to 264 randomly generated TAP instances with fractional extreme points: 138 of the 264 have 74 variables, so the theoretical guarantee of Theorem 2 is at least $(1.5)^{74}$. For the rest, the number of variables in 250.

FDT for 2EC on a 963 fractional extreme points of $2\mathrm{EC}(G)$. These points are obtained by considering all fundamental vertices with 10 and 12 vertices (See [2] for the definition of fundamental vertices). The results are summarized in Table 2. We also implemented polyhedral version of Christofides' algorithm [18] and

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

Table 2. FDT for 2EC implemented applied to all fundamental extreme points with 10 and 12 vertices. The number of variables for a fundamental extreme point with k vertices is $\frac{3k}{2}$ and the lower bound on $g(2\mathrm{EC})$ is $\frac{6}{5}$

compared its performance on fundamental extreme points with 10 vertices. The result are in Figure 1.

References

1. Robert Carr and R. Ravi. A new bound for the 2-edge connected subgraph problem. In *Proceedings of the 6th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pages 112–125, London, UK, UK, 1998. Springer-Verlag.
2. 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.
3. Greg N. Frederickson and Joseph Ja'Ja'. Approximation algorithms for several graph augmentation problems. *SIAM Journal on Computing*, 10(2):270–283, 1981.
4. Cheriyan, Joseph and Karloff, Howard and Khandekar, Rohit and Könemann, Jochen. On the integrality ratio for tree augmentation. *Operations Research Letters*, 36(4):399–401, 2008.
5. Anthony Alexander, Sylvia Boyd, and Paul Elliott-Magwood. On the Integrality Gap of the 2-Edge Connected Subgraph Problem. Technical report, University of Ottawa, Ottawa, Canada, 04 2006.
6. Geneviève Benoit and Sylvia Boyd. Finding the exact integrality gap for small traveling salesman problems. *Mathematics of Operations Research*, 33(4):921–931, 2008.
7. Gérard Cornuéjols, Jean Fonlupt, and Denis Naddef. The traveling salesman problem on a graph and some related integer polyhedra. *Mathematical Programming*, 33(1):1–27, Sep 1985.
8. Matteo Fischetti, Fred Glover, and Andrea Lodi. The feasibility pump. *Mathematical Programming*, 104(1):91–104, Sep 2005.
9. Matteo Fischetti and Domenico Salvagnin. Feasibility pump 2.0. *Mathematical Programming Computation*, 1(2):201–222, Oct 2009.
10. M.R. Garey and D.S. Johnson. *Computers and Intractability – A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, CA, 1979.
11. Michel X. Goemans. Worst-case comparison of valid inequalities for the TSP. *Math. Program.*, 69(2):335–349, August 1995.
12. Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2. Second corrected edition edition, 1993.
13. Kamal Jain. A factor 2 approximation algorithm for the generalized steiner network problem. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, FOCS '98, pages 448–, Washington, DC, USA, 1998. IEEE Computer Society.
14. Prabhakar Raghavan and Clark D. Tompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, Dec 1987.
15. A. Schrijver. *Combinatorial Optimization - Polyhedra and Efficiency*. Springer, 2003.
16. Vijay V. Vazirani. *Approximation Algorithms*. Springer-Verlag, Berlin, Heidelberg, 2001.
17. David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, New York, NY, USA, 1st edition, 2011.
18. Laurence A. Wolsey. *Heuristic analysis, linear programming and branch and bound*, pages 121–134. Springer Berlin Heidelberg, Berlin, Heidelberg, 1980.

A FDT for 2EC

In Section 2 our focus was on binary MIPs. 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. In the 2EC problem, we are given non-negative costs on the edge of G and the goal is to find the minimum cost 2-edge-connected multigraph of G. 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. The natural linear programming relaxation is $\text{2EC}(G) = \{x \in [0,2]^E : x(\delta(U)) \geq 2 \text{ for } \emptyset \subset U \subset V\}$. Notice that $\mathcal{D}(\text{2EC}(G)) \cap [0,2]^E = \text{2EC}(G)$, since 2EC is a covering problem. We want to prove Theorem 4. We do not know the exact value for g_{2EC}, but we know $\frac{6}{5} \leq g_{\text{2EC}} \leq \frac{3}{2}$ [1,18]. Also, we need to remark that polyhedral version of Christofides' algorithm provides a $\frac{3}{2}$-approximation for 2EC, i.e. we already have an algorithm with $C \leq \frac{3}{2}$. However, we showed in Section 3 that in practice the constant C for the FDT algorithm for 2EC is much better than $\frac{3}{2}$ for fundamental extreme points with 10 vertices.

The FDT algorithm for 2EC is very similar to the one for binary MILPs, 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 $\{0,1,2\}$-MILPs.

First, we need a branching lemma. Observe that the following branching lemma is essentially a translation of Lemma 1 for 0,1,2 problems expect for one additional clause.

Lemma 5. *Given $x \in \text{2EC}(G)$, 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) $\gamma_0 + \gamma_1 + \gamma_2 \geq \frac{1}{g_{\text{2EC}}}$, (ii) x^0, x^1, and x^2 are in $\text{2EC}(G)$, (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 $j = 0, 1, 2$.*

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

$$\max \sum_{j=0,1,2} \lambda_j \tag{24}$$

$$\text{s.t.} \quad x^j(\delta(U)) \geq 2\lambda_j \qquad \text{for } \emptyset \subset U \subset V, \text{ and } j = 0,1,2 \tag{25}$$

$$0 \leq x^j \leq 2\lambda_j \qquad \text{for } j = 0,1,2 \tag{26}$$

$$x^j_e = j \qquad \text{for } j = 0,1,2 \tag{27}$$

$$x^j_f \geq j \qquad \text{for } f \in E \text{ where } x_f \geq 1, \text{ and } j = 0,1,2 \tag{28}$$

$$x^0 + x^1 + x^2 \leq x \tag{29}$$

$$\lambda_0, \lambda_1, \lambda_2 \geq 0 \tag{30}$$

Let x^j, γ_j for $j = 0, 1, 2$ be an optimal solution solution to the LP above. Let $\hat{x}^j = \frac{x^j}{\gamma_j}$ 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 $\gamma_0 + \gamma_1 + \gamma_2 \geq \frac{1}{g_{2\mathrm{EC}}}$, which means that (i) is also satisfied. The proof is similar to the proof of the claim in Lemma 1, but we need to replace each $f \in E$ with $x_f \geq 0$ with a suitably long path to ensure that Constraint (28) is also satisfied. We skip the details.

In contrast to FDT for binary MIPs 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 preform 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 $2\mathrm{EC}(G)$ 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 6. *The FDT algorithm for 2EC in polynomial time produces sets $L_0, \ldots, L_t$ of pairs $x \in 2\mathrm{EC}(G)$ together with multipliers λ with the following properties. (a) If $x \in L_i$, then $x_{e_j} = 0$ or $x_{e_j} \geq 1$ for $j = 1, \ldots, i$, (b) $\sum_{(x,\lambda)\in L_i} \lambda \geq \frac{1}{g_{2\mathrm{EC}}^i}$, (c) $\sum_{(x,\lambda)\in L_i} \lambda x \leq x^*$, (d) $|L_i| \leq t$.*

The proof is similar to Lemma 3, but we need to use property (i) in Lemma 5 to prove that (a) also holds.

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

Lemma 7. *Let x be a solution in L_t. Then $\lfloor x \rfloor \in 2\mathrm{EC}(G)$.*

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 $x \in 2\mathrm{EC}(G)$ 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 6, 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.

The FDT algorithm for 2EC iteratively applies Lemmas 5 and 6 to variables $x_1, \ldots, x_t$ to obtain leaf point solutions L_t. Then, we just need to apply Lemma 7 to obtain the 2-edge-connected multigraphs from every solution in L_t. Notice that since x is an extreme point we have $t \leq 2|V| - 1$ [7]. By Lemma 6 we have

$$\sum_{(x,\lambda)\in L_t} \frac{\lambda}{\sum_{(x,\lambda)\in L_t} \lambda} \lfloor x \rfloor \leq \frac{1}{\sum_{(x,\lambda)\in L_t} \lambda} \sum_{(x,\lambda)\in L_t} \lambda x \leq g_{2\mathrm{EC}}^t x^*.$$