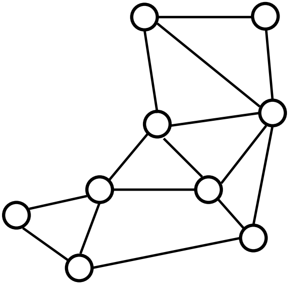
Reformulations and Solution Algorithms for the Maximum Leaf Spanning Tree Problem

Nelson Maculan[1]
Abilio Lucena[1]
Luidi G. Simonetti[2]

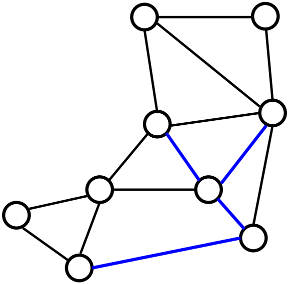
[1]Universidade Federal do Rio de Janeiro
[2]Universidade Estadual de Campinas

August 2009

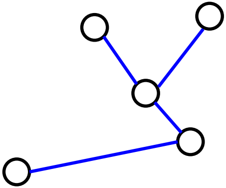
Graph $G = (V, E)$



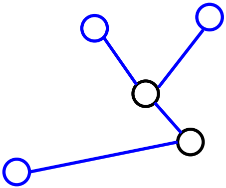
A Non-spanning Tree $\mathcal{T}$ of G



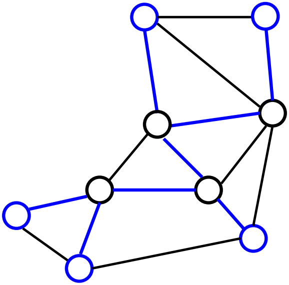
The Subgraph $\mathcal{T}$ of G



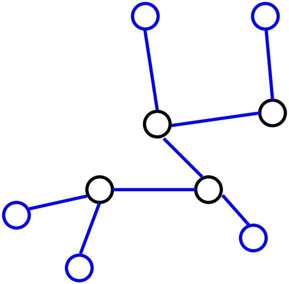
The Leaves of $\mathcal{T}$



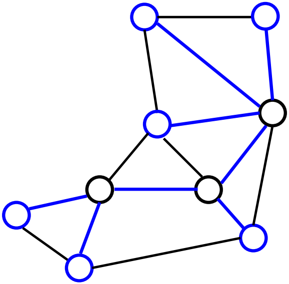
A Spanning Tree $\mathcal{T}$ for $G = (V, E)$



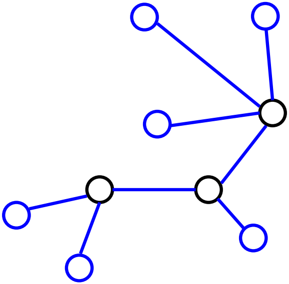
The Subgraph $\mathcal{T}$ of G



A Spanning Tree $\mathcal{T}$ of G with 6 Leaves



A Spanning Tree $\mathcal{T}$ of G with 6 Leaves



The Maximum Leaf Spanning Tree Problem

Given a graph $G = (V, E)$:

find a spanning tree $\mathcal{T}$ of G with as many leaves as possible.

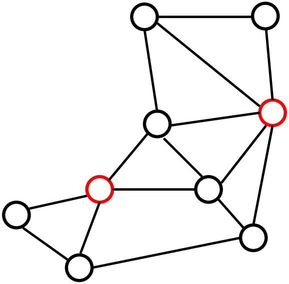
A Closely Related Problem

Minimum Connected Dominating Set Problem:

follows from the Minimum Dominating Set Problem.

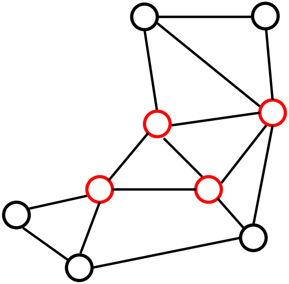
Dominating Set S

Set $S \subset V$ with $i \in V \setminus S$ one edge away from $j \in S$



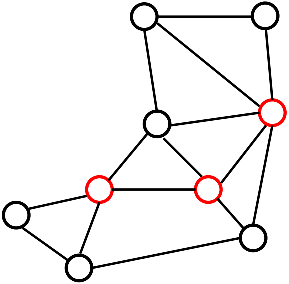
Connected Dominating Set S

Subgraph $G_S = (S, E(S))$ of G is connected

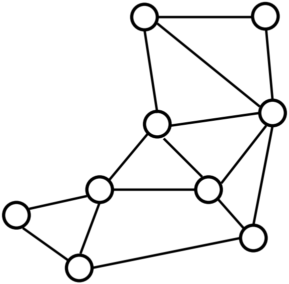


Minimum Connected Dominating Set Problem

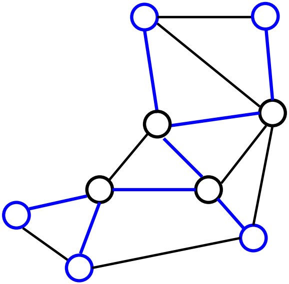
Find connected dominating set $S \subset V$ with $|S|$ as small as possible



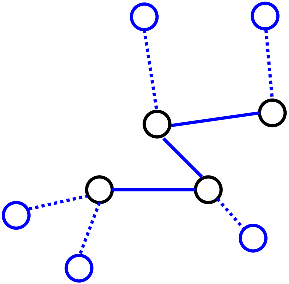
Spanning Trees × Connected Dominating Sets



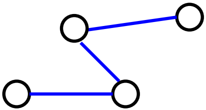
Spanning Trees × Connected Dominating Sets



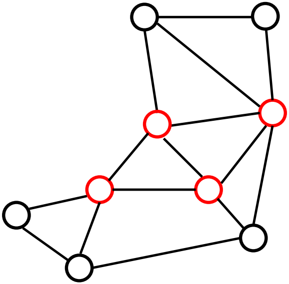
Spanning Trees × Connected Dominating Sets



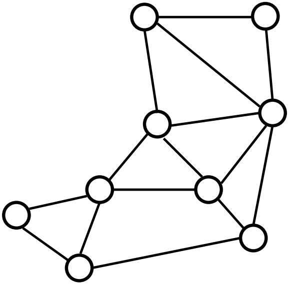
Spanning Trees $\times$ Connected Dominating Sets



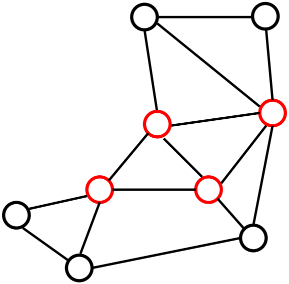
Spanning Trees × Connected Dominating Sets



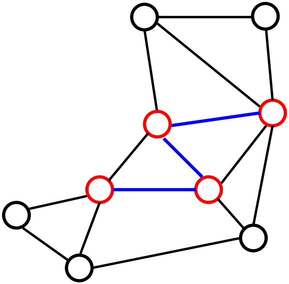
Connected Dominating Sets × Spanning Trees



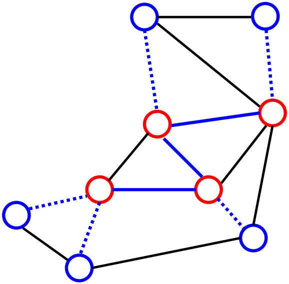
Connected Dominating Sets × Spanning Trees



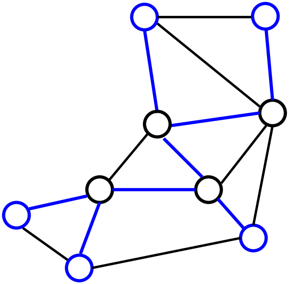
Connected Dominating Sets × Spanning Trees



Connected Dominating Sets × Spanning Trees



Connected Dominating Sets × Spanning Trees



Optimal Solutions

Max-Leaf Spanning Tree Problem

$\times$

Minimum Connected Dominating Set Problem

Given an optimal solution to one :

optimal solution to the other in polynomial time!

Why Investigate These Two Problems?

- Quite challenging *NP*-hard problems
- Model some telecommunication network design applications
- Model some circuit layout design applications

Why Investigate These Two Problems?

- Quite challenging *NP*-hard problems
- Model some telecommunication network design applications
- Model some circuit layout design applications

Previous Work for Maximum Leaf STP

- Garey and Johnson [1979]: problem is *NP*-hard
- Lu and Ravi [1992,1998]: factor of 3 approximation algorithm
- Galbiati, Maffioli and Morzenti [1994]: problem is *SNP*-hard (there exists ϵ such that finding $(1+\epsilon)$-approximation algorithm is *NP*-hard)
- Solis-Oba [1998]: factor of 2 approximation algorithm
- Fernandes and Gouveia [1998]: formulation for fixed number of leaves
- Fujie [2003]: formulation and branch-and-bound algorithm
- Fujie [2004]: new formulation and polyhedral study

Previous Work for Maximum Leaf STP

- Garey and Johnson [1979]: problem is *NP*-hard
- Lu and Ravi [1992,1998]: factor of 3 approximation algorithm
- Galbiati, Maffioli and Morzenti [1994]: problem is *SNP*-hard
  (there exists ϵ such that finding $(1+\epsilon)$-approximation algorithm is *NP*-hard)
- Solis-Oba [1998]: factor of 2 approximation algorithm
- Fernandes and Gouveia [1998]: formulation for fixed number of leaves
- Fujie [2003]: formulation and branch-and-bound algorithm
- Fujie [2004]: new formulation and polyhedral study

Previous Work for Minimum Connected DSP

- Garey and Johnson [1979]: problem is *NP*-hard
- Guha and Khuller [1998]: Approximation algorithms
- Cheng, Huang, Li, Wu, and Du [2006]:
  Polynomial-time approximation scheme
- Chen, Lujubić and Raghavan [2007], accepted for publication, 2008:
  The Regenerator Location Problem

Previous Work for Minimum Connected DSP

- Garey and Johnson [1979]: problem is *NP*-hard
- Guha and Khuller [1998]: Approximation algorithms
- Cheng, Huang, Li, Wu, and Du [2006]:
  Polynomial-time approximation scheme
- Chen, Lujubić and Raghavan [2007], accepted for publication, 2008:
  The Regenerator Location Problem

Outline for the Presentation

Formulation + reformulations for Max. Leaf STP

- A formulation from the literature
- A reinforced direct graph reformulation
- Recasting the problem as Steiner Arborescence Problem

A heuristic for the Minimum Connected Dominating Set Problem

Pre-processing tests

Computational results

Conclusions and Future Work

Formulation of Fujie [2003,2004]

Basic Idea: Spanning Tree Polytope + Leaf Imposing Constraints

Variables involved:

- $x = \{x_e \in \{0,1\} : e \in E\}$: to identify tree edges
- $z = \{z_i \in \{0,1\} : i \in V\}$: to identify tree leaves

Notation used:

- $\delta(i) \subseteq E$, $i \in V$: edges incident to i
- $E(S) \subseteq E$, $S \subseteq V$: edges with both end vertices in S

Formulation of Fujie [2003,2004]

Basic Idea: Spanning Tree Polytope + Leaf Imposing Constraints

Variables involved:

- $x = \{x_e \in \{0,1\} : e \in E\}$: to identify tree edges
- $z = \{z_i \in \{0,1\} : i \in V\}$: to identify tree leaves

Notation used:

- $\delta(i) \subseteq E$, $i \in V$: edges incident to i
- $E(S) \subseteq E$, $S \subseteq V$: edges with both end vertices in S

Formulation of Fujie [2003,2004]

$$\max \sum_{i \in V} z_i$$

$$\sum_{e \in E} x_e = |V| - 1$$

$$\sum_{e \in E(S)} x_e \leq |S| - 1 \qquad S \subset V, |S| \geq 2$$

$$\sum_{e \in \delta(i)} x_e + (|\delta(i)| - 1) z_i \leq |\delta(i)| \qquad i \in V$$

$$x_e \geq 0 \qquad e \in E$$

$$z_i \in \{0, 1\} \qquad i \in V$$

Follows from a formulation to

"Minimal Spanning Trees with a Constraint on the Number of Leaves"

by Fernandes and Gouveia [1998]

Formulation of Fujie [2003,2004]

$$\max \sum_{i \in V} z_i$$

$$\sum_{e \in E} x_e = |V| - 1$$

$$\sum_{e \in E(S)} x_e \leq |S| - 1 \qquad S \subset V, |S| \geq 2$$

$$\sum_{e \in \delta(i)} x_e + (|\delta(i)| - 1) z_i \leq |\delta(i)| \qquad i \in V$$

$$x_e \geq 0 \qquad e \in E$$

$$z_i \in \{0, 1\} \qquad i \in V$$

Follows from a formulation to
"Minimal Spanning Trees with a Constraint on the Number of Leaves"
by Fernandes and Gouveia [1998]

Formulation of Fujie [2003,2004]

$$\max \sum_{i \in V} z_i$$

$$\sum_{e \in E} x_e = |V| - 1$$

$$\sum_{e \in E(S)} x_e \leq |S| - 1 \qquad S \subset V, |S| \geq 2$$

$$\sum_{e \in \delta(i)} x_e + (|\delta(i)| - 1) z_i \leq |\delta(i)| \qquad i \in V$$

$$x_e \geq 0 \qquad e \in E$$

$$z_i \in \{0, 1\} \qquad i \in V$$

Follows from a formulation to
"Minimal Spanning Trees with a Constraint on the Number of Leaves"
by Fernandes and Gouveia [1998]

LP Relaxation of Fujie's Formulation

$$\max \sum_{i \in V} z_i$$

$$\text{s.t.} \sum_{e \in E} x_e = |V| - 1$$

$$\sum_{e \in E(S)} x_e \leq |S| - 1 \qquad S \subset V, |S| \geq 2 \qquad (1)$$

$$\sum_{e \in \delta(i)} x_e + (|\delta(i)| - 1) z_i \leq |\delta(i)| \qquad i \in V \qquad (2)$$

$$x_e \geq 0 \qquad e \in E$$

$$z_i \geq 0 \qquad i \in V$$

$x_e \leq 1$ is implied by (1) for $|S| = 2$

$z_i \leq 1$, $i \in V$ is implied by (2)

LP Relaxation of Fujie's Formulation

Since one is maximizing $\sum_{i \in V} z_i$:

$$z_i \leq \left[\frac{|\delta(i)| - \sum_{e \in \delta(i)} x_e}{|\delta(i)| - 1} \right], i \in V,$$

must hold as equality, at optimality!

Substituting for z in the objective function:

$$\begin{array}{ll} \max & \sum_{i \in V} \frac{|\delta(i)|}{|\delta(i)| - 1} - \sum_{e=(i,j) \in E} \left(\frac{1}{|\delta(i)| - 1} + \frac{1}{|\delta(j)| - 1} \right) x_e \\ \text{s.t.} & x \in \text{Spanning Tree Polytope} \end{array}$$

LP relaxation bound could be computed efficiently!

Depth-first-search based branch-and-bound algorithm - Fujie [2003]

Does not use lower bound given by upper bounding spanning tree

LP Relaxation of Fujie's Formulation

Since one is maximizing $\sum_{i \in V} z_i$:

$$z_i \leq \left[\frac{|\delta(i)| - \sum_{e \in \delta(i)} x_e}{|\delta(i)| - 1} \right], i \in V,$$

must hold as equality, at optimality!

Substituting for z in the objective function:

$$\begin{aligned} \max \quad & \sum_{i \in V} \frac{|\delta(i)|}{|\delta(i)| - 1} - \sum_{e=(i,j) \in E} \left(\frac{1}{|\delta(i)| - 1} + \frac{1}{|\delta(j)| - 1} \right) x_e \\ \text{s.t.} \quad & x \in \text{Spanning Tree Polytope} \end{aligned}$$

LP relaxation bound could be computed efficiently!

Depth-first-search based branch-and-bound algorithm - Fujie [2003]

Does not use lower bound given by upper bounding spanning tree

LP Relaxation of Fujie's Formulation

Since one is maximizing $\sum_{i \in V} z_i$:

$$z_i \leq \left[\frac{|\delta(i)| - \sum_{e \in \delta(i)} x_e}{|\delta(i)| - 1}\right], i \in V,$$

must hold as equality, at optimality!

Substituting for z in the objective function:

$$\begin{array}{ll} \max & \displaystyle\sum_{i \in V} \frac{|\delta(i)|}{|\delta(i)| - 1} - \sum_{e=(i,j) \in E} \left(\frac{1}{|\delta(i)| - 1} + \frac{1}{|\delta(j)| - 1} \right) x_e \\ \text{s.t.} & x \in \text{Spanning Tree Polytope} \end{array}$$

LP relaxation bound could be computed efficiently!

Depth-first-search based branch-and-bound algorithm - Fujie [2003]

Does not use lower bound given by upper bounding spanning tree

LP Relaxation of Fujie's Formulation

Since one is maximizing $\sum_{i \in V} z_i$:

$$z_i \leq \left[\frac{|\delta(i)| - \sum_{e \in \delta(i)} x_e}{|\delta(i)| - 1}\right], i \in V,$$

must hold as equality, at optimality!

Substituting for z in the objective function:

$$\begin{array}{ll} \max & \displaystyle\sum_{i \in V} \frac{|\delta(i)|}{|\delta(i)| - 1} - \sum_{e=(i,j) \in E} \left(\frac{1}{|\delta(i)| - 1} + \frac{1}{|\delta(j)| - 1}\right) x_e \\ \text{s.t.} & x \in \text{Spanning Tree Polytope} \end{array}$$

LP relaxation bound could be computed efficiently!

Depth-first-search based branch-and-bound algorithm - Fujie [2003]

Does not use lower bound given by upper bounding spanning tree

LP Relaxation of Fujie's Formulation

Since one is maximizing $\sum_{i \in V} z_i$:

$$z_i \leq \left\lfloor \frac{|\delta(i)| - \sum_{e \in \delta(i)} x_e}{|\delta(i)| - 1} \right\rfloor, i \in V,$$

must hold as equality, at optimality!

Substituting for z in the objective function:

$$\begin{array}{ll} \max & \sum_{i \in V} \frac{|\delta(i)|}{|\delta(i)| - 1} - \sum_{e=(i,j) \in E} \left(\frac{1}{|\delta(i)| - 1} + \frac{1}{|\delta(j)| - 1} \right) x_e \\ \text{s.t.} & x \in \text{Spanning Tree Polytope} \end{array}$$

LP relaxation bound could be computed efficiently!

Depth-first-search based branch-and-bound algorithm - Fujie [2003]

Does not use lower bound given by upper bounding spanning tree

Details of Fujie's Branch-and-Bound Algorithm

Heuristics used:

- At the root node of the enumeration tree:
  - Breadth-First-Search (BFS) Spanning Tree Heuristic (Fujie [2003])
  - Factor of 3 Approximation Heuristic (Lu and Ravi [1998])
  - Factor of 2 Approximation Heuristic (Solis-Oba [1998])
- At nodes other than the root:
  - BFS Spanning Tree Heuristic, if certain conditions are met

Details of Fujie's Branch-and-Bound Algorithm

Heuristics used:

- At the root node of the enumeration tree:
  - Breadth-First-Search (BFS) Spanning Tree Heuristic (Fujie [2003])
  - Factor of 3 Approximation Heuristic (Lu and Ravi [1998])
  - Factor of 2 Approximation Heuristic (Solis-Oba [1998])
- At nodes other than the root:
  - BFS Spanning Tree Heuristic, if certain conditions are met

Enhancements to Fujie's Algorithm

At every enumeration tree node: lower bounds from upper bound computation

Additional speed-ups through the use of
(new) Minimum Connected Dominating Set heuristic

Number of enumeration tree nodes dropped to about half

Computational results presented for enhanced version of Fujie's algorithm

Facet Defining Inequalities for Fujie's Formulation

Inequality

$$\sum_{e \in \delta(i)} x_e + (|\delta(i)| - 1) z_i \leq |\delta(i)| \qquad i \in V \tag{3}$$

is used in the formulation to characterize spanning tree leaves

For $F \subseteq \delta(i)$, where $|F| \geq 2$: inequality (3) is the particular case for $F = \delta(i)$!

For the general case, where $F \subset \delta(i)$, for $i \in V$ and $|F| \geq 2$:

$$\sum_{e \in F} x_e + (|F| - 1) z_i \leq |F| \tag{4}$$

is facet defining inequalities for corresponding polytope - Fujie[2004]

If $n_i = |\delta(i)|$, inequalities in (4) could be separated for i in $O(n_i \log n_i)$ time

Facet Defining Inequalities for Fujie's Formulation

Inequality

$$\sum_{e \in \delta(i)} x_e + (|\delta(i)| - 1) z_i \leq |\delta(i)| \qquad i \in V \tag{3}$$

is used in the formulation to characterize spanning tree leaves

For $F \subseteq \delta(i)$, where $|F| \geq 2$: inequality (3) is the particular case for $F = \delta(i)$!

For the general case, where $F \subset \delta(i)$, for $i \in V$ and $|F| \geq 2$:

$$\sum_{e \in F} x_e + (|F| - 1) z_i \leq |F| \tag{4}$$

is facet defining inequalities for corresponding polytope - Fujie[2004]

If $n_i = |\delta(i)|$, inequalities in (4) could be separated for i in $O(n_i \log n_i)$ time

Facet Defining Inequalities for Fujie's Formulation

Inequality

$$\sum_{e \in \delta(i)} x_e + (|\delta(i)| - 1) z_i \leq |\delta(i)| \qquad i \in V \tag{3}$$

is used in the formulation to characterize spanning tree leaves

For $F \subseteq \delta(i)$, where $|F| \geq 2$: inequality (3) is the particular case for $F = \delta(i)$!

For the general case, where $F \subset \delta(i)$, for $i \in V$ and $|F| \geq 2$:

$$\sum_{e \in F} x_e + (|F| - 1) z_i \leq |F| \tag{4}$$

is facet defining inequalities for corresponding polytope - Fujie[2004]

If $n_i = |\delta(i)|$, inequalities in (4) could be separated for i in $O(n_i \log n_i)$ time

Facet Defining Inequalities for Fujie's Formulation

Inequality

$$\sum_{e \in \delta(i)} x_e + (|\delta(i)| - 1) z_i \leq |\delta(i)| \qquad i \in V \tag{3}$$

is used in the formulation to characterize spanning tree leaves

For $F \subseteq \delta(i)$, where $|F| \geq 2$: inequality (3) is the particular case for $F = \delta(i)$!

For the general case, where $F \subset \delta(i)$, for $i \in V$ and $|F| \geq 2$:

$$\sum_{e \in F} x_e + (|F| - 1) z_i \leq |F| \tag{4}$$

is facet defining inequalities for corresponding polytope - Fujie[2004]

If $n_i = |\delta(i)|$, inequalities in (4) could be separated for i in $O(n_i \log n_i)$ time

Facet Defining Inequalities for Fujie's Formulation

Inequality

$$\sum_{e \in \delta(i)} x_e + (|\delta(i)| - 1) z_i \leq |\delta(i)| \qquad i \in V \tag{3}$$

is used in the formulation to characterize spanning tree leaves

For $F \subseteq \delta(i)$, where $|F| \geq 2$: inequality (3) is the particular case for $F = \delta(i)$!

For the general case, where $F \subset \delta(i)$, for $i \in V$ and $|F| \geq 2$:

$$\sum_{e \in F} x_e + (|F| - 1) z_i \leq |F| \tag{4}$$

is facet defining inequalities for corresponding polytope - Fujie[2004]

If $n_i = |\delta(i)|$, inequalities in (4) could be separated for i in $O(n_i \log n_i)$ time

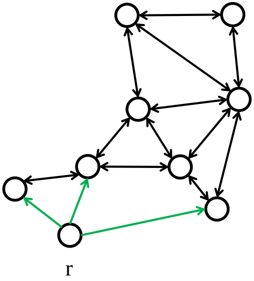
Directed Graph Reformulation for Fujie's Formulation

Suggested (with minor differences) in Fernandes and Gouveia [1998]

Build directed graph $D = (r, V, A)$ from $G = (V, E)$

- $\{(r, j) : \ \forall e = [r, j] \vee e = [j, r], e \in E\}$
- $\{(i, j) \wedge (j, i) : \ \forall e = [i, j] \in E, \ i, j \in V \setminus \{r\}\}$

Variables involved: $\{y_{ij} : (i, j) \in A\}$ and $\{z_i : i \in V\}$



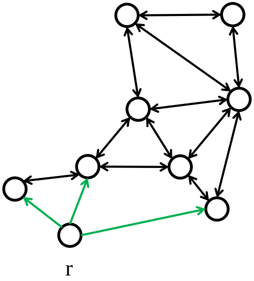
Directed Graph Reformulation for Fujie's Formulation

Suggested (with minor differences) in Fernandes and Gouveia [1998]

Build directed graph $D = (r, V, A)$ from $G = (V, E)$

- $\{(r, j) : \forall e = [r, j] \vee e = [j, r], e \in E\}$
- $\{(i, j) \wedge (j, i) : \forall e = [i, j] \in E, \; i, j \in V \setminus \{r\}\}$

Variables involved: $\{y_{ij} : (i, j) \in A\}$ and $\{z_i : i \in V\}$



Directed Graph Reformulation for Fujie's Formulation

$$\max \sum_{i \in V} z_i$$

$$\text{s.t.} \sum_{a \in A} y_a = |V| - 1$$

$$\sum_{a \in \delta^-(j)} y_a = 1 \qquad j \in V \setminus \{r\}$$

$$\sum_{a \in A(S)} y_a \leq |S| - 1 \qquad S \subset V, |S| \geq 2$$

$$\sum_{a \in \delta^+(i)} y_a + (|\delta^-(i)| - 1) z_i \leq |\delta^-(i)| - 1 \qquad i \in V \setminus \{r\}$$

$$\sum_{a \in \delta^+(r)} y_a + (|\delta^+(r)| - 1) z_r \leq |\delta^+(r)|$$

$$y_a \geq 0 \qquad a \in A$$

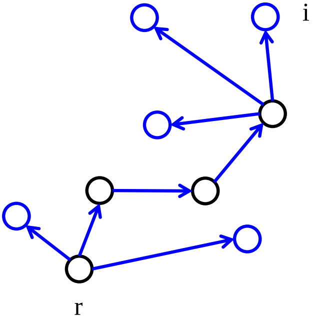
$$z_i \in \{0, 1\} \qquad i \in V$$

Same LP relaxation bound as Fujie's undirected graph formulation!

r-Rooted Spanning Arborescence $\mathcal{T}$ of $D = (V, A)$

For a spanning arborescence of D:

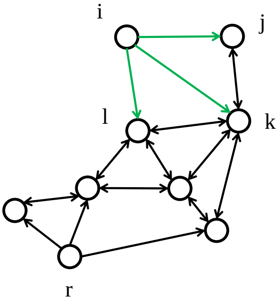
$i \in V \setminus \{r\}$ is leaf implying iff it has no outwards pointing arcs



Advantage over Fujie's formulation

- Inequalities $y_a + z_i \leq 1, \quad a = (i,j) \in A, \quad i \in V \setminus \{r\}$ impose this condition
- Very few inequalities that considerably strengthen the reformulation!
- No counterpart inequalities for Fujie's undirected graph formulation

$x_{[i,j]} + z_i \leq 1$, $x_{[i,k]} + z_i \leq 1$ and $x_{[i,l]} + z_i \leq 1$ are valid for vertex i



Further Strengthening the Directed Graph Reformulation

Maximum Leaf STP facet defining inequalities characterized by Fujie [2004]:

$$\sum_{e \in F} x_e + (|F| - 1) z_i \leq |F| \qquad i \in V, F \subseteq \delta(i), \text{ for } |F| \geq 2$$

could be written as

$$\sum_{a \in F} y_a + \sum_{a=(j,i) \in A | (i,j) \in F} y_a + (|F| - 1) z_i \leq |F|, \; i \in V, \;\; F \subseteq \delta^+(i), \;\; |F| \geq 2$$

Further Strengthening the Directed Graph Reformulation

Maximum Leaf STP facet defining inequalities characterized by Fujie [2004]:

$$\sum_{e \in F} x_e + (|F| - 1) z_i \leq |F| \qquad i \in V, F \subseteq \delta(i), \text{ for } |F| \geq 2$$

could be written as

$$\sum_{a \in F} y_a + \sum_{a=(j,i) \in A | (i,j) \in F} y_a + (|F| - 1) z_i \leq |F|, \; i \in V, \;\; F \subseteq \delta^+(i), \;\; |F| \geq 2$$

LP Relaxation Bound Improvements

$\|V\|$	dens.	opt.	no ineqs.	just Fujie's	just ours	Fujie's + ours
	5	95	105.04	103.15	97.86	97.51
120	20	112	116.39	116.19	114.93	114.93
	50	116	118.42	118.34	118.12	118.12

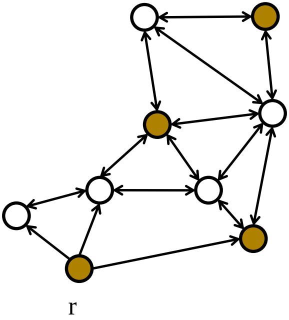
Directed Graph Reformulation

Branch-and-Cut Algorithm implemented

Steiner Arborescence Reformulation for Max. Leaf STP

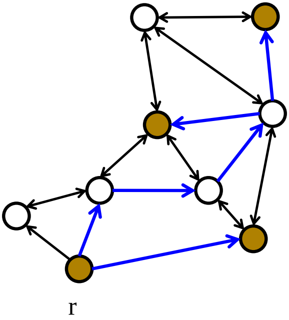
Steiner Arborescence Problem:

r-rooted tree $\mathcal{T}$, for directed graph $D = (V, A, T, r)$, must contain T, where $r \in T$ and $T \subset V$



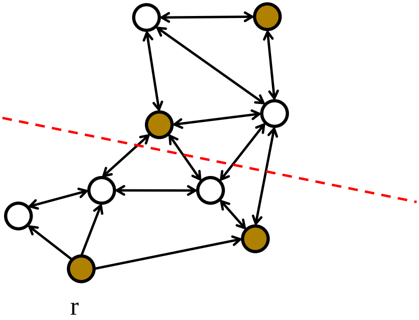
Steiner Arborescence Reformulation for Max. Leaf STP

Steiner Arborescence $\mathcal{T}$



Steiner Arborescence Reformulation for Max. Leaf STP

For $S \subset V$, where $r \in S$ and $(V \setminus S) \cap T \neq \emptyset$:
at least one arc in cut-set $\delta(S, V \setminus S)$ must be used at a feasible solution



Steiner Arborescence Reformulation for Max. Leaf STP

Cut-Set Formulation for Steiner Arborescence Problem
defined over directed graph $D = (V, A, r, T)$

$$\text{minimize} \sum_{[i,j] \in A} c_{ij} w_{ij}$$

subject to

$$\sum_{(i,j) \in \delta(S, V \setminus S)} w_{ij} \geq 1, \quad r \in S, (V \setminus S) \cap T \neq \emptyset$$

$$w_{ij} \in \{0, 1\}, \text{ for } (i,j) \in A$$

Suggested by Aneja [1980], Wong [1989]

Steiner Arborescence Reformulation for Max. Leaf STP

- Luidi Simonetti Phd thesis, 2008:
  - Hop-Constrained Minimum Spanning Tree Problem +
    Diameter-Constrained Minimum Spanning Tree Problem:
    Gouveia, Simonetti and Uchoa [2006,2007,2008]
  - Maximum Leaf Spanning Tree Problem:
    Lucena, Maculan and Simonetti [2008]
- The Regeneration Location Problem:
  (Minimum Connected Dominating Set Problem):
  Chen, Lujubić and Raghavan [2007], accepted for publication, 2008

Steiner Arborescence Reformulation for Max. Leaf STP

For simplicity, for $G = (V, E)$, denote $V = \{v_1, v_2, \ldots v_n\}$

Maximum Leaf STP reformulation as Steiner Arborescence Problem
over directed graph $D_s = (V_s, A_s, T, r)$

Building $D_s = (V_s, A_s, T, r)$ from $G = (V, E)$

V_s is partitioned into V_s^0, V_s^1 and V_s^2:

- $V_s^0 = \{v_0\}$: $r = v_0$ is an artificial "root" vertex
- $V_s^1 = \{v_i^1 : v_i \in V\}$: "copy" of V
- $V_s^2 = \{v_i^2 : v_i \in V\}$: "copy" of V
- $|V_s| = 2|V| + 1$

A_s is defined as:

- $\{(v_0, v_i^1) : \ v_i \in V\}$: (only one is to be used)
- $\{(v_i^1, v_j^1) \wedge (v_j^1, v_i^1) : [v_i, v_j] \in E\}$: (to identify dominating set vertices of $\mathcal{T}$)
- $\{(v_i^1, v_j^2) : [v_i, v_j] \in E\}$: (to identify spanning tree leaves of $\mathcal{T}$)
- $\{(v_i^1, v_i^2) : v_i \in V\}$: (simply used as a modeling device)

Terminal vertices for Steiner Arborescence: $T = \{v_0\} \cup V_s^2$

Building $D_s = (V_s, A_s, T, r)$ from $G = (V, E)$

V_s is partitioned into V_s^0, V_s^1 and V_s^2:

- $V_s^0 = \{v_0\}$: $r = v_0$ is an artificial "root" vertex
- $V_s^1 = \left\{v_i^1 : v_i \in V\right\}$: "copy" of V
- $V_s^2 = \left\{v_i^2 : v_i \in V\right\}$: "copy" of V
- $|V_s| = 2|V| + 1$

A_s is defined as:

- $\{(v_0, v_i^1) : \ v_i \in V\}$: (only one is to be used)
- $\{(v_i^1, v_j^1) \wedge (v_j^1, v_i^1) : [v_i, v_j] \in E\}$: (to identify dominating set vertices of $\mathcal{T}$)
- $\{(v_i^1, v_j^2) : [v_i, v_j] \in E\}$: (to identify spanning tree leaves of $\mathcal{T}$)
- $\{(v_i^1, v_i^2) : v_i \in V\}$: (simply used as a modeling device)

Terminal vertices for Steiner Arborescence: $T = \{v_0\} \cup V_s^2$

Building $D_s = (V_s, A_s, T, r)$ from $G = (V, E)$

V_s is partitioned into V_s^0, V_s^1 and V_s^2:

- $V_s^0 = \{v_0\}$: $r = v_0$ is an artificial "root" vertex
- $V_s^1 = \left\{v_i^1 : v_i \in V\right\}$: "copy" of V
- $V_s^2 = \left\{v_i^2 : v_i \in V\right\}$: "copy" of V
- $|V_s| = 2|V| + 1$

A_s is defined as:

- $\{(v_0, v_i^1) : \quad v_i \in V\}$: (only one is to be used)
- $\{(v_i^1, v_j^1) \wedge (v_j^1, v_i^1) : [v_i, v_j] \in E\}$: (to identify dominating set vertices of $\mathcal{T}$)
- $\{(v_i^1, v_j^2) : [v_i, v_j] \in E\}$: (to identify spanning tree leaves of $\mathcal{T}$)
- $\{(v_i^1, v_i^2) : v_i \in V\}$: (simply used as a modeling device)

Terminal vertices for Steiner Arborescence: $T = \{v_0\} \cup V_s^2$

Building $D_s = (V_s, A_s, T, r)$ from $G = (V, E)$

V_s is partitioned into V_s^0, V_s^1 and V_s^2:

- $V_s^0 = \{v_0\}$: $r = v_0$ is an artificial "root" vertex
- $V_s^1 = \left\{v_i^1 : v_i \in V\right\}$: "copy" of V
- $V_s^2 = \left\{v_i^2 : v_i \in V\right\}$: "copy" of V
- $|V_s| = 2|V| + 1$

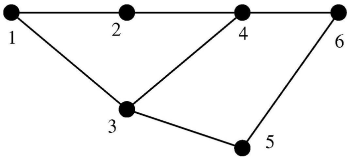
A_s is defined as:

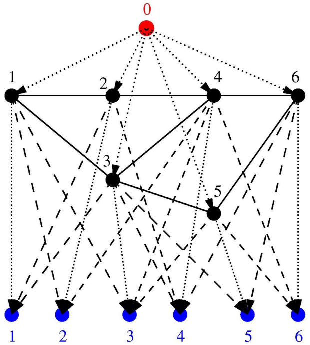
- $\{(v_0, v_i^1) : \; v_i \in V\}$: (only one is to be used)
- $\{(v_i^1, v_j^1) \wedge (v_j^1, v_i^1) : [v_i, v_j] \in E\}$: (to identify dominating set vertices of $\mathcal{T}$)
- $\{(v_i^1, v_j^2) : [v_i, v_j] \in E\}$: (to identify spanning tree leaves of $\mathcal{T}$)
- $\{(v_i^1, v_i^2) : v_i \in V\}$: (simply used as a modeling device)

Terminal vertices for Steiner Arborescence: $T = \{v_0\} \cup V_s^2$

Graph $G_s = (V_s, A_s, T, r = v_0)$

$$
\begin{aligned}
V_s &= \{v_0\} \cup \{v_i^k : 1 \leq k \leq 2,\ v_i \in V\} \\
A_s &= \{(v_0, v_j^1) : v_j \in V\} \cup \{(v_i^1, v_j^1), (v_j^1, v_i^1) : [v_i, v_j] \in E\} \\
&\quad \cup \{(v_i^1, v_i^2) : v_i \in V\} \cup \{(v_i^1, v_j^2) \wedge (v_j^1, v_i^2) : [v_i, v_j] \in E\}
\end{aligned}
$$



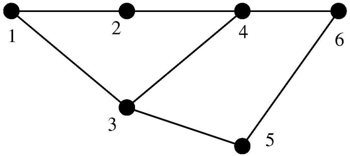


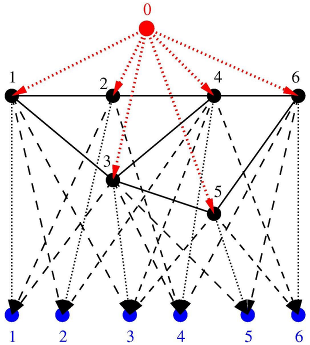
Graph $G_s = (V_s, A_s, T, r = v_0)$

$$V_s = \{v_0\} \cup \{v_i^k : 1 \leq k \leq 2,\ v_i \in V\}$$

$$A_s = \{(v_0, v_j^1) : v_j \in V\} \cup \{(v_i^1, v_j^1), (v_j^1, v_i^1) : [v_i, v_j] \in E\}$$

$$\cup \{(v_i^1, v_i^2) : v_i \in V\} \cup \{(v_i^1, v_j^2) \wedge (v_j^1, v_i^2) : [v_i, v_j] \in E\}$$



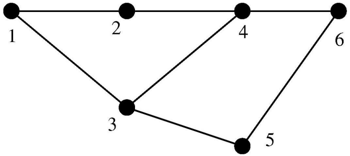


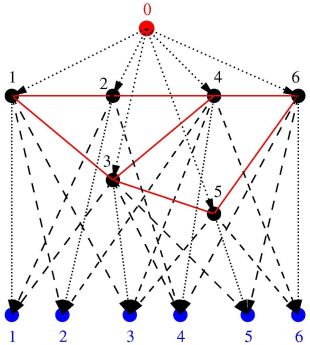
Graph $G_s = (V_s, A_s, T, r = v_0)$

$$V_s = \{v_0\} \cup \{v_i^k : 1 \leq k \leq 2,\ v_i \in V\}$$

$$A_s = \{(v_0, v_j^1) : v_j \in V\} \cup \{(v_i^1, v_j^1), (v_j^1, v_i^1) : [v_i, v_j] \in E\}$$

$$\cup\ \{(v_i^1, v_i^2) : v_i \in V\} \cup \{(v_i^1, v_j^2) \wedge (v_j^1, v_i^2) : [v_i, v_j] \in E\}$$



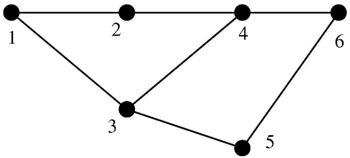


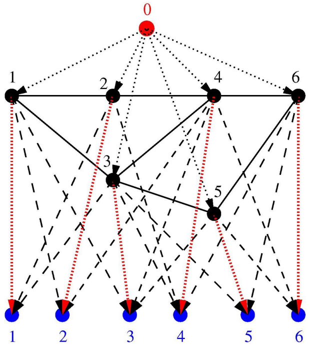
Graph $G_s = (V_s, A_s, T, r = v_0)$

$$V_s = \{v_0\} \cup \{v_i^k : 1 \leq k \leq 2,\ v_i \in V\}$$

$$A_s = \{(v_0, v_j^1) : v_j \in V\} \cup \{(v_i^1, v_j^1), (v_j^1, v_i^1) : [v_i, v_j] \in E\}$$

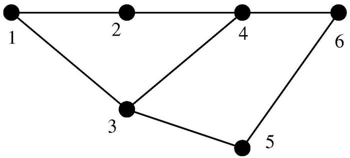
$$\cup\ \{(v_i^1, v_i^2) : v_i \in V\} \cup \{(v_i^1, v_j^2) \wedge (v_j^1, v_i^2) : [v_i, v_j] \in E\}$$

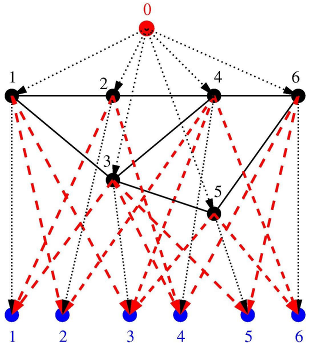




Graph $G_s = (V_s, A_s, T, r = v_0)$

$$
\begin{aligned}
V_s &= \{v_0\} \cup \{v_i^k : 1 \le k \le 2,\ v_i \in V\} \\
A_s &= \{(v_0, v_j^1) : v_j \in V\} \cup \{(v_i^1, v_j^1), (v_j^1, v_i^1) : [v_i, v_j] \in E\} \\
&\quad \cup \{(v_i^1, v_i^2) : v_i \in V\} \cup {\color{red}\{(v_i^1, v_j^2) \wedge (v_j^1, v_i^2) : [v_i, v_j] \in E\}}
\end{aligned}
$$





Steiner Arborescence Reformulation for Max. Leaf STP

(Quite sophisticated) Branch-and-Cut Algorithm

implemented for the Steiner Arborescence Problem

Pre-processing Tests for $G = (V, E)$

Degree one: if $|\delta(i)| = 1$, vertex i must be leaf implying (from the literature)

Articulation vertex: if i is an articulation vertex for G, it can not be a leaf

Vertex dominance: for $i, j \in V$ with $\Gamma(j) \subseteq \Gamma(i)$,
optimal solution with j leaf implying

Lower Bounds (Feasible Solutions) for Maximum Leaf STP

Lower bound = number of leaves in a (any) spanning tree of G

- Factor of 3 approximation algorithms: Lu and Ravi [1992,1998] (performs Local Search)
- Factor of 2 approximation algorithm: Solis-Oba [1998] (no good in practice)
- Depth-first-search strategy: Build $|V|$ spanning trees of G, Fujie [2003]
- Connected dominating set heuristic: Lucena, Maculan and Simonetti [2008] (also in Chen, Lujubić and Raghavan [2007])

Upper bounding procedure in Fujie [2003]:

"good quality" spanning trees in dual information driven approach!

The LM&S and CL&R Mininum Connected DS Heuristic

Graph $G = (V, E)$, connected dominating set $\mathcal{D}$, and leaf set $\mathcal{L}$

Initialization:

- Select "seed" $r \in V$ and place it into $\mathcal{L}$ together with all vertices in $\Gamma(r)$
- Select $i \in \mathcal{L}$ with the largest $|\Gamma(i)|$.
- Remove i from $\mathcal{L}$ and place it into $\mathcal{D}$.
- Introduce all vertices in $(\Gamma(i) \setminus \mathcal{L})$ into $\mathcal{L}$.

Iteration $k \geq 2$:

- Select a $i \in \mathcal{L}$ with the largest $|\Gamma(i) \setminus (\mathcal{L} \cup \mathcal{D})|$.
- Remove i from $\mathcal{L}$ and place it in $\mathcal{D}$.
- Place all vertices in $\Gamma(i) \setminus (\mathcal{L} \cup \mathcal{D})$ into $\mathcal{L}$.
- Stop when all vertices of V are either in $\mathcal{L}$ or $\mathcal{D}$.

The LM&S and CL&R Mininum Connected DS Heuristic

Graph $G = (V, E)$, connected dominating set $\mathcal{D}$, and leaf set $\mathcal{L}$

Initialization:

- Select "seed" $r \in V$ and place it into $\mathcal{L}$ together with all vertices in $\Gamma(r)$
- Select $i \in \mathcal{L}$ with the largest $|\Gamma(i)|$.
- Remove i from $\mathcal{L}$ and place it into $\mathcal{D}$.
- Introduce all vertices in $(\Gamma(i) \setminus \mathcal{L})$ into $\mathcal{L}$.

Iteration $k \geq 2$:

- Select a $i \in \mathcal{L}$ with the largest $|\Gamma(i) \setminus (\mathcal{L} \cup \mathcal{D})|$.
- Remove i from $\mathcal{L}$ and place it in $\mathcal{D}$.
- Place all vertices in $\Gamma(i) \setminus (\mathcal{L} \cup \mathcal{D})$ into $\mathcal{L}$.
- Stop when all vertices of V are either in $\mathcal{L}$ or $\mathcal{D}$.

The LM&S and CL&R Mininum Connected DS Heuristic

Graph $G = (V, E)$, connected dominating set $\mathcal{D}$, and leaf set $\mathcal{L}$

Initialization:

- Select "seed" $r \in V$ and place it into $\mathcal{L}$ together with all vertices in $\Gamma(r)$
- Select $i \in \mathcal{L}$ with the largest $|\Gamma(i)|$.
- Remove i from $\mathcal{L}$ and place it into $\mathcal{D}$.
- Introduce all vertices in $(\Gamma(i) \setminus \mathcal{L})$ into $\mathcal{L}$.

Iteration $k \geq 2$:

- Select a $i \in \mathcal{L}$ with the largest $|\Gamma(i) \setminus (\mathcal{L} \cup \mathcal{D})|$.
- Remove i from $\mathcal{L}$ and place it in $\mathcal{D}$.
- Place all vertices in $\Gamma(i) \setminus (\mathcal{L} \cup \mathcal{D})$ into $\mathcal{L}$.
- Stop when all vertices of V are either in $\mathcal{L}$ or $\mathcal{D}$.

Computational Results

			B&B			Dir			SAP		
$\|V\|$	dens.	opt.	LP bound	# nodes	T(s)	LP bound	# nodes	T(s)	LP bound	# nodes	T(s)
	10	15	17.83	291	0.12	15.57	1	0.01	15.80	1	0.04
	20	23	26.04	5055	0.33	24.48	7	0.1	23.95	1	0.12
30	30	26	27.44	842	0.24	27.05	1	0.03	26.13	5	26.7
	50	27	28.46	307	0.19	28.13	3	0.09	27.94	1	1.28
	70	28	28.83	1	0.16	28.73	1	0.01	28.00	1	0.26
	5	19	21.50	265	0.94	19.00	1	0.02	19.00	1	0.09
	10	38	42.16	82599	4.54	39.75	41	0.82	38.86	38	94
50	20	43	46.58	225771	16.9	45.22	77	1.32	44.48	57	1827
	30	45	47.59	38155	5.97	46.80	39	1.21	46.08	43	22424
	50	47	48.45	3050	4	48.18	13	0.51	47.36	-	-
	70	48	48.82	5	1.64	48.62	1	0.09	48.00	1	2.08
	5	43	49.57	9068999	313	44.45	53	0.99	43.56	19	103
	10	57	63.18	-	-	59.20	174	4.73	58.60	-	-
70	20	63	66.22	-	-	65.04	607	16.3	64.37	-	-
	30	65	67.62	4113677	536	66.91	35	2.9	66.15		-
	50	67	68.52	33058	25.3	68.14	7	1.33	-	-	-
	70	68	68.76	2661	10.8	68.68	5	1.92	68	1	8.55

Computational Results

$\|V\|$	dens.	opt.	B&B LP bound	B&B # nodes	B&B T(s)	Dir LP bound	Dir # nodes	Dir T(s)	SAP LP bound	SAP # nodes	SAP T(s)
30	10	15	17.83	291	0.12	15.57	1	0.01	15.80	1	0.04
	20	23	26.04	5055	0.33	24.48	7	0.1	23.95	1	0.12
	30	26	27.44	842	0.24	27.05	1	0.03	26.13	5	26.7
	50	27	28.46	307	0.19	28.13	3	0.09	27.94	1	1.28
	70	28	28.83	1	0.16	28.73	1	0.01	28.00	1	0.26
50	5	19	21.50	265	0.94	19.00	1	0.02	19.00	1	0.09
	10	38	42.16	82599	4.54	39.75	41	0.82	38.86	38	94
	20	43	46.58	225771	16.9	45.22	77	1.32	44.48	57	1827
	30	45	47.59	38155	5.97	46.80	39	1.21	46.08	43	22424
	50	47	48.45	3050	4	48.18	13	0.51	47.36	-	-
	70	48	48.82	5	1.64	48.62	1	0.09	48.00	1	2.08
70	5	43	49.57	9068999	313	44.45	53	0.99	43.56	19	103
	10	57	63.18	-	-	59.20	174	4.73	58.60	-	-
	20	63	66.22	-	-	65.04	607	16.3	64.37	-	-
	30	65	67.62	4113677	536	66.91	35	2.9	66.15		-
	50	67	68.52	33058	25.3	68.14	7	1.33	-	-	-
	70	68	68.76	2661	10.8	68.68	5	1.92	68	1	8.55

Computational Results

$\lvert V \rvert$	dens.	opt.	B&B			Dir			SAP		
			LP bound	# nodes	T(s)	LP bound	# nodes	T(s)	LP bound	# nodes	T(s)
100	5	76	83.42	-	-	79.36	605	24.5	78.57	-	-
	10	87	93.27	-	-	89.52	135	9.36	-	-	-
	20	92	96.56	-	-	94.85	1025	86.1	-	-	-
	30	94	97.39	-	-	96.68	1753	258	-	-	-
	50	96	98.36	348389	213	98.03	479	132	-	-	-
	70	97	98.76	9091	50.5	98.64	121	154	-	-	-
120	5	95	105.04	-	-	97.77	24	2.65	-	-	-
	10	107	113.16	-	-	109.83	869	65.4	-	-	-
	20	112	116.39	-	-	114.93	2401	393	-	-	-
	30	114	117.40	-	-	116.69	2301	653	-	-	-
	50	116	118.42	571335	435	118.12	1297	815	-	-	-
	70	117	118.72	13791	97	118.63	137	356	-	-	-
150	5	124	135.11	-	-	128.74	31077	2954	-	-	-
	10	136	142.81	-	-	139.59	6089	3247	-	-	-
	20	141	146.81	-	-	145.12	173425	61639	-	-	-
	30	144	147.38	-	-	146.67	3043	2617	-	-	-
	50	146	148.38	2104992	2190	148.10	1755	2756	-	-	-
	70	147	148.72	21625	301	148.63	219	1828	-	-	-
200	50	196	198.32	-	-	198.07	3125	20155	-	-	-
	70	197	198.73	38215	1322	198.63	253	8154	-	-	-

Computational Results

$\lvert V \rvert$	dens.	opt.	B&B			Dir			SAP		
			LP bound	# nodes	T(s)	LP bound	# nodes	T(s)	LP bound	# nodes	T(s)
100	5	76	83.42	-	-	79.36	605	24.5	78.57	-	-
	10	87	93.27	-	-	89.52	135	9.36	-	-	-
	20	92	96.56	-	-	94.85	1025	86.1	-	-	-
	30	94	97.39	-	-	96.68	1753	258	-	-	-
	50	96	98.36	348389	213	98.03	479	132	-	-	-
	70	97	98.76	9091	50.5	98.64	121	154	-	-	-
120	5	95	105.04	-	-	97.77	24	2.65	-	-	-
	10	107	113.16	-	-	109.83	869	65.4	-	-	-
	20	112	116.39	-	-	114.93	2401	393	-	-	-
	30	114	117.40	-	-	116.69	2301	653	-	-	-
	50	116	118.42	571335	435	118.12	1297	815	-	-	-
	70	117	118.72	13791	97	118.63	137	356	-	-	-
150	5	124	135.11	-	-	128.74	31077	2954	-	-	-
	10	136	142.81	-	-	139.59	6089	3247	-	-	-
	20	141	146.81	-	-	145.12	173425	61639	-	-	-
	30	144	147.38	-	-	146.67	3043	2617	-	-	-
	50	146	148.38	2104992	2190	148.10	1755	2756	-	-	-
	70	147	148.72	21625	301	148.63	219	1828	-	-	-
200	50	196	198.32	-	-	198.07	3125	20155	-	-	-
	70	197	198.73	38215	1322	198.63	253	8154	-	-	-

Conclusions

Fujie's algorithm:

- Weaker upper (LP) bounds
- Demanding in terms of memory
- Faster for some high density instances

Directed graph reformulation:

- Stronger LP bounds than original undirected formulation
- Faster in 31 out of 37 instances tested
- Attempted to solve the largest number of test instances

SAP reformulation:

- Stronger LP bounds than other formulations
- Highly demanding in terms of CPU time

Conclusions

Fujie's algorithm:

- Weaker upper (LP) bounds
- Demanding in terms of memory
- Faster for some high density instances

Directed graph reformulation:

- Stronger LP bounds than original undirected formulation
- Faster in 31 out of 37 instances tested
- Attempted to solve the largest number of test instances

SAP reformulation:

- Stronger LP bounds than other formulations
- Highly demanding in terms of CPU time

Conclusions

Fujie's algorithm:

- Weaker upper (LP) bounds
- Demanding in terms of memory
- Faster for some high density instances

Directed graph reformulation:

- Stronger LP bounds than original undirected formulation
- Faster in 31 out of 37 instances tested
- Attempted to solve the largest number of test instances

SAP reformulation:

- Stronger LP bounds than other formulations
- Highly demanding in terms of CPU time

Conclusions

Fujie's algorithm:

- Weaker upper (LP) bounds
- Demanding in terms of memory
- Faster for some high density instances

Directed graph reformulation:

- Stronger LP bounds than original undirected formulation
- Faster in 31 out of 37 instances tested
- Attempted to solve the largest number of test instances

SAP reformulation:

- Stronger LP bounds than other formulations
- Highly demanding in terms of CPU time

Future Work

Symmetry breaking

Additional strong valid inequalities

Fine tuning of branching strategy

Future Work

Symmetry breaking

Additional strong valid inequalities

Fine tuning of branching strategy

Future Work

Symmetry breaking

Additional strong valid inequalities

Fine tuning of branching strategy

Future Work

Symmetry breaking

Additional strong valid inequalities

Fine tuning of branching strategy