Column Generation Algorithms for the Capacitated m-Ring-Star Problem[1]

Edna Ayako Hoshino and Cid Carvalho de Souza

University of Campinas - UNICAMP
Institute of Computing - IC

may, 2008

[1]Capes/PICDT,CNPq – Conselho Nacional de Desenvolvimento Científico e Tecnológico,FAPESP – Fundação de Amparo à Pesquisa do Estado de São Paulo

Capacitated Vehicle Routing Problem (CVRP)

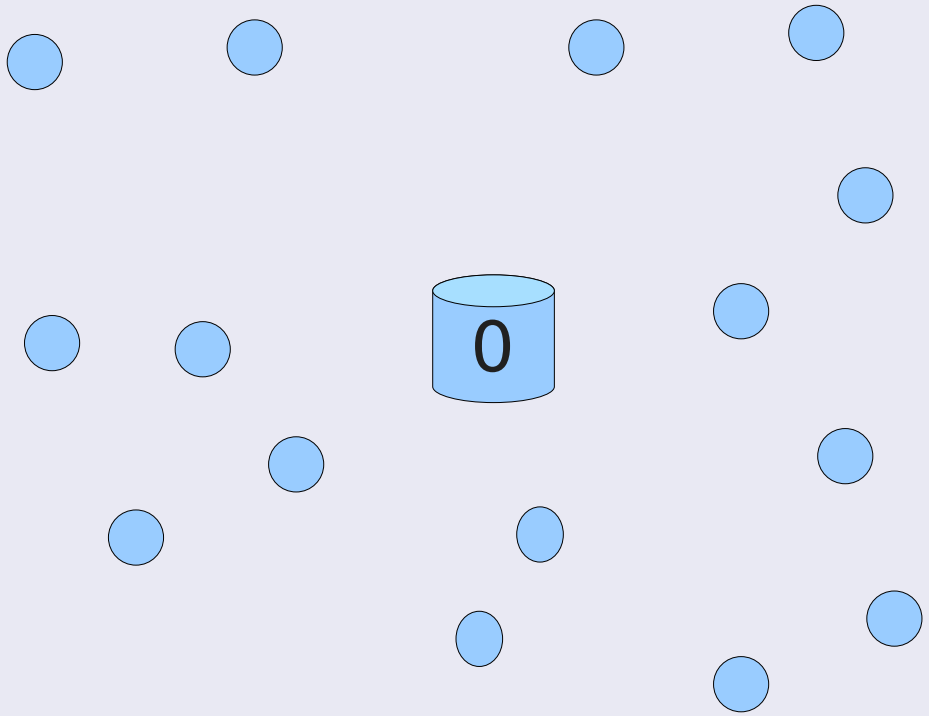


Figure: CVRP instance with $|U| = 15$, $m = 3$ and $Q = 6$.

A Solution for the CVRP

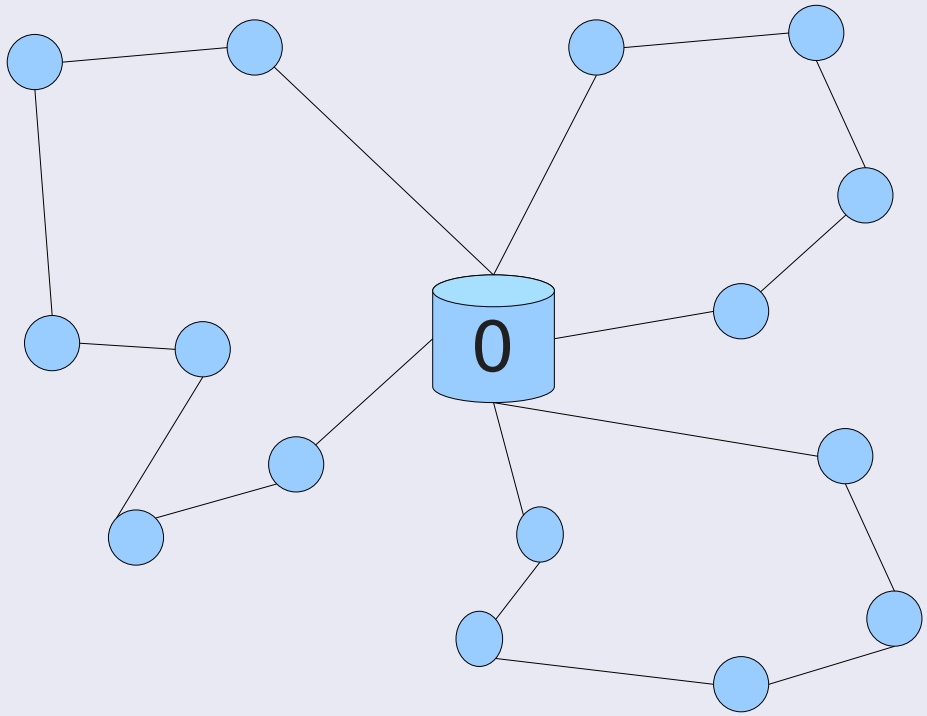


Figure: A solution for the CVRP instance with $m = 3$, $Q = 6$ and graph in Figure 1.

Capacitated Vehicle Routing Problem (CVRP)

Given:

- a graph $G = (V, E)$ where $V = \{0\} \cup U$, (depot 0 and a set of customers U);
- a fleet of m identical vehicles (each of them having a capacity Q);
- costs $c_e \geq 0$, $\forall e \in E$;
- demands $d_i \geq 0, \forall i \in U$.

Capacitated Vehicle Routing Problem (CVRP)

Given:

- a graph $G = (V, E)$ where $V = \{0\} \cup U$, (depot 0 and a set of customers U);
- a fleet of m identical vehicles (each of them having a capacity Q);
- costs $c_e \geq 0$, $\forall e \in E$;
- demands $d_i \geq 0, \forall i \in U$.

The CVRP consists of finding routes for m vehicles such that:

- each route starts and ends at the depot;
- each customer is visited by a single vehicle;
- the total demand of all customers in any route is at most Q, and;
- the sum of the costs of all routes is minimum.

Alternative Solution

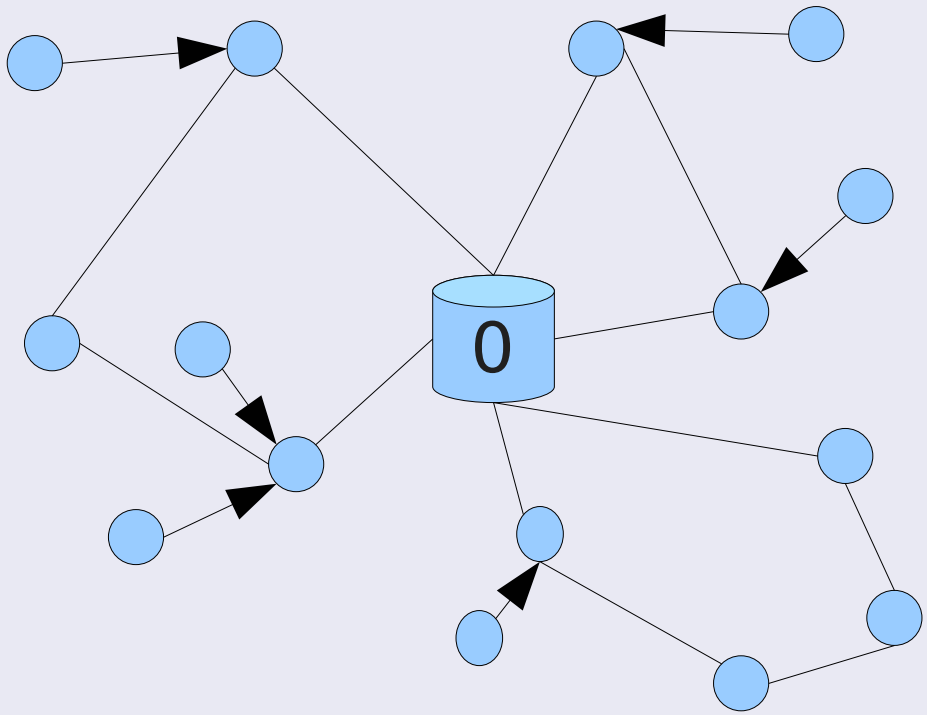


Figure: A solution that allows some customers stay outside of all routes.

Capacitated m-Ring-Star Problem (CmRSP)

Given:

- a mixed graph $G = (V, E \cup A)$ where $V = \{0\} \cup U \cup W$ (depot 0, a set of customers U and a set of Steiner points W);
- unitary demands;
- integer values m and Q;
- route costs $c_e > 0$, $\forall e \in E = \{(i,j) : i,j \in V\}$, satisfying the triangular inequalities;
- connection costs $w_e > 0$, $\forall e \in A \subseteq \{ij : i \in U, j \in U \cup W\}$.

Capacitated m-Ring-Star Problem (CmRSP)

Given:

- a mixed graph $G = (V, E \cup A)$ where $V = \{0\} \cup U \cup W$ (depot 0, a set of customers U and a set of Steiner points W);
- unitary demands;
- integer values m and Q;
- route costs $c_e > 0$, $\forall e \in E = \{(i,j) : i,j \in V\}$, satisfying the triangular inequalities;
- connection costs $w_e > 0$, $\forall e \in A \subseteq \{ij : i \in U, j \in U \cup W\}$.

The CmRSP consists of finding m Q-ring-stars with minimum costs and covering all customers.

Capacitated m-Ring-Star Problem (CmRSP)

Given:

- a mixed graph $G = (V, E \cup A)$ where $V = \{0\} \cup U \cup W$ (depot 0, a set of customers U and a set of Steiner points W);
- unitary demands;
- integer values m and Q;
- route costs $c_e > 0$, $\forall e \in E = \{(i,j) : i,j \in V\}$, satisfying the triangular inequalities;
- connection costs $w_e > 0$, $\forall e \in A \subseteq \{ij : i \in U, j \in U \cup W\}$.

The CmRSP consists of finding m Q-ring-stars with minimum costs and covering all customers.

- a pair (R, S) is a Q-ring-star if:
  - $R \subseteq E$ is a cycle passing by the depot 0;
  - $ij \in S \subseteq A$ such that $i \notin V[R]$ and $j \in V[R]$;
  - $|U \cap (V[R] \cup V[S])| \leq Q$.

Capacitated m-Ring-Star Problem (CmRSP)

Given:

- a mixed graph $G = (V, E \cup A)$ where $V = \{0\} \cup U \cup W$ (depot 0, a set of customers U and a set of Steiner points W);
- unitary demands;
- integer values m and Q;
- route costs $c_e > 0$, $\forall e \in E = \{(i,j) : i,j \in V\}$, satisfying the triangular inequalities;
- connection costs $w_e > 0$, $\forall e \in A \subseteq \{ij : i \in U, j \in U \cup W\}$.

The CmRSP consists of finding m Q-ring-stars with minimum costs and covering all customers.

- a pair (R, S) is a Q-ring-star if:
  - $R \subseteq E$ is a cycle passing by the depot 0;
  - $ij \in S \subseteq A$ such that $i \notin V[R]$ and $j \in V[R]$;
  - $|U \cap (V[R] \cup V[S])| \leq Q$.
- We say that a Q-ring-star covers a customer i if $i \in V[R] \cup V[S]$.

Example of a Q-ring-star

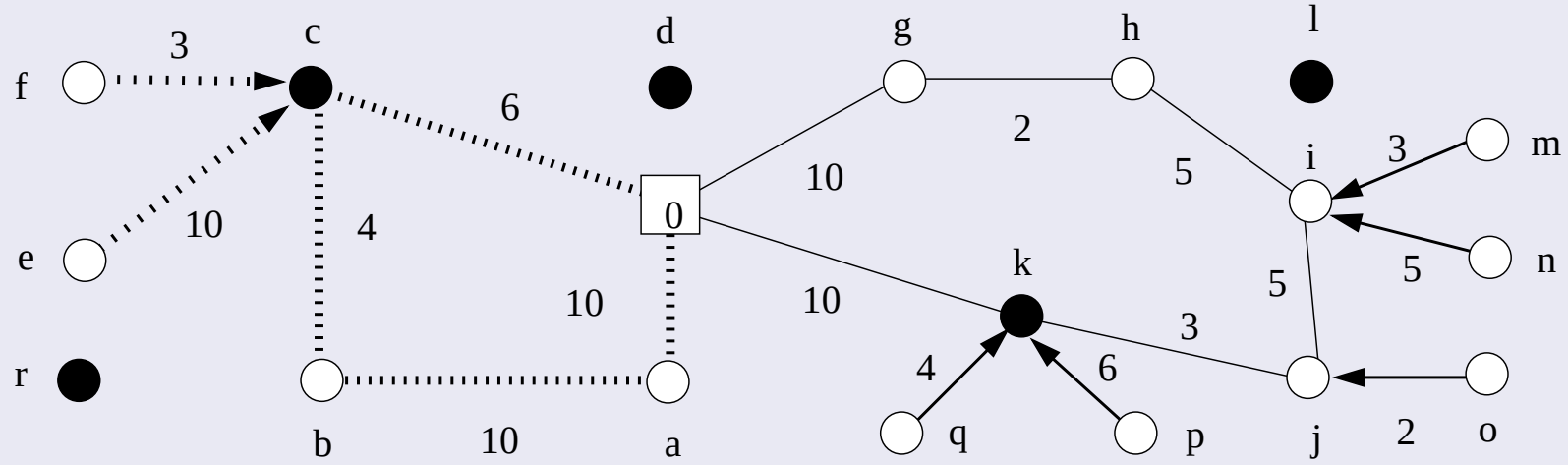


Figure: Two 9-*ring-stars* $(\{(0,g),(g,h),(h,i),(i,j),(j,k),(k,0)\},\{mi,ni,oj,pk,qk\})$ and $(\{(0,a),(a,b),(b,c),(c,0)\},\{(ec,fc\})$.

Example of a CmRSP Instance

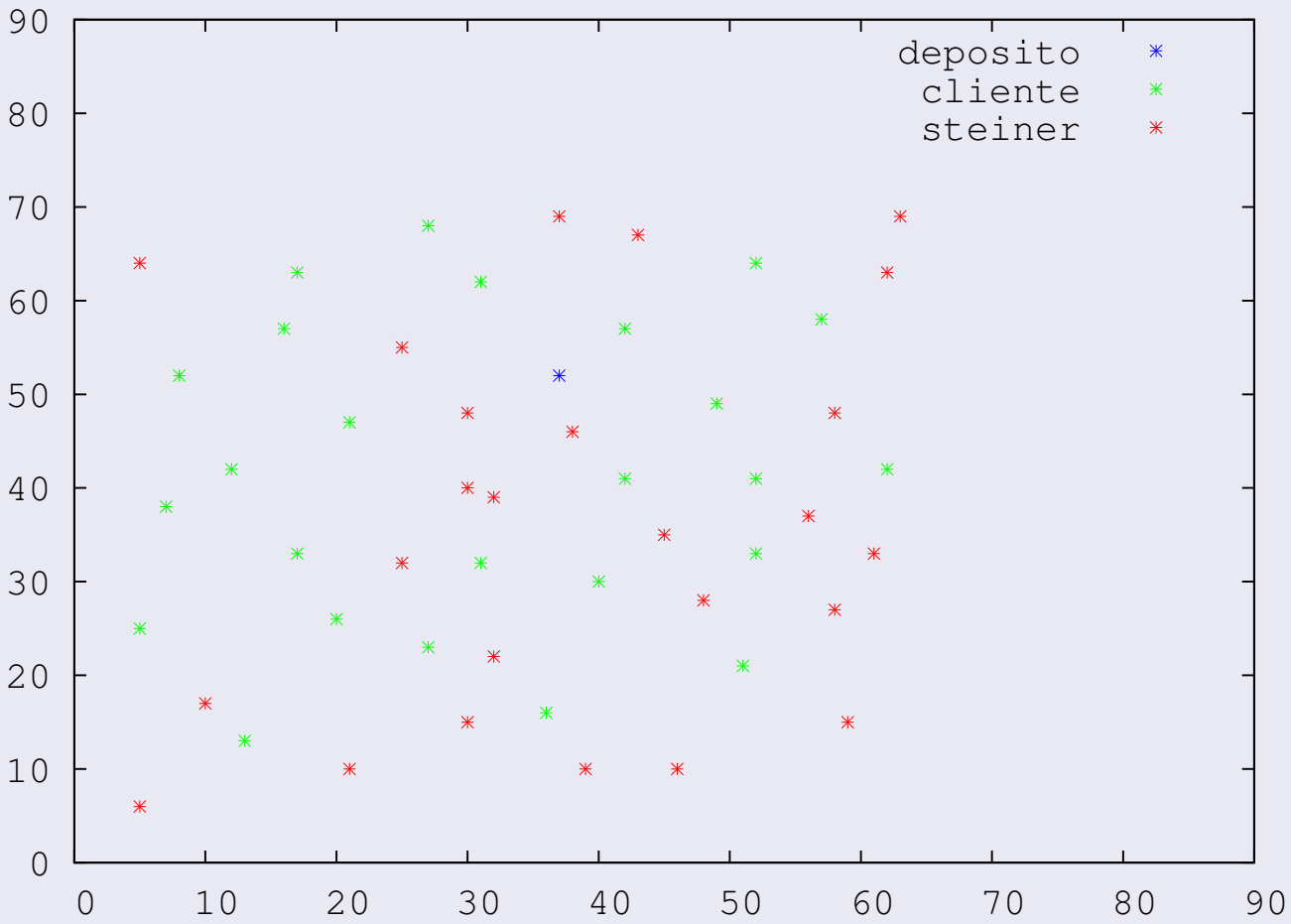


Figure: Instance eil51.tsp with $|U| = 25$.

Example of a Q-ring-star

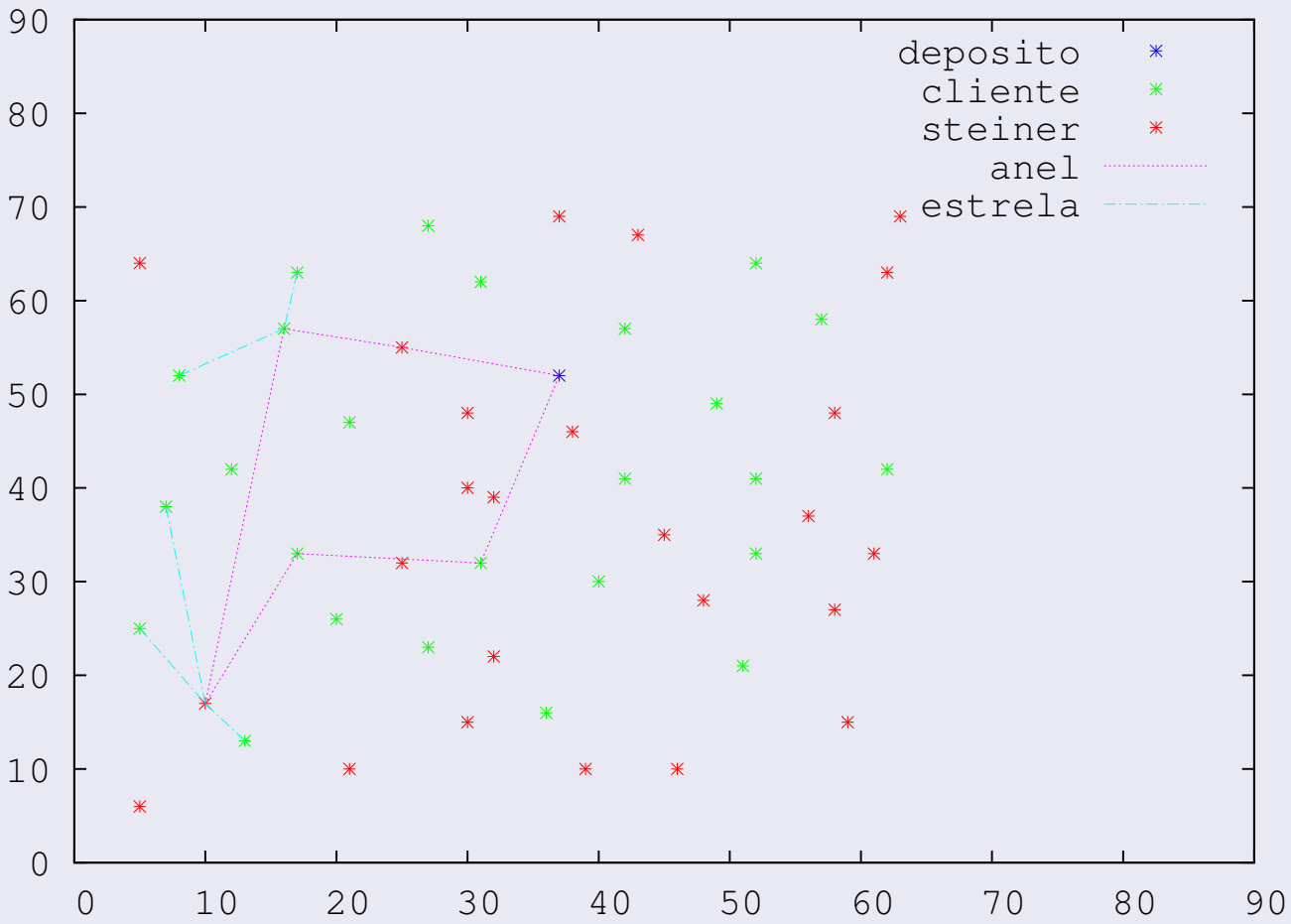


Figure: 8-ring-star for the instance in Figure 5.

A Solution for the CmRSP instance

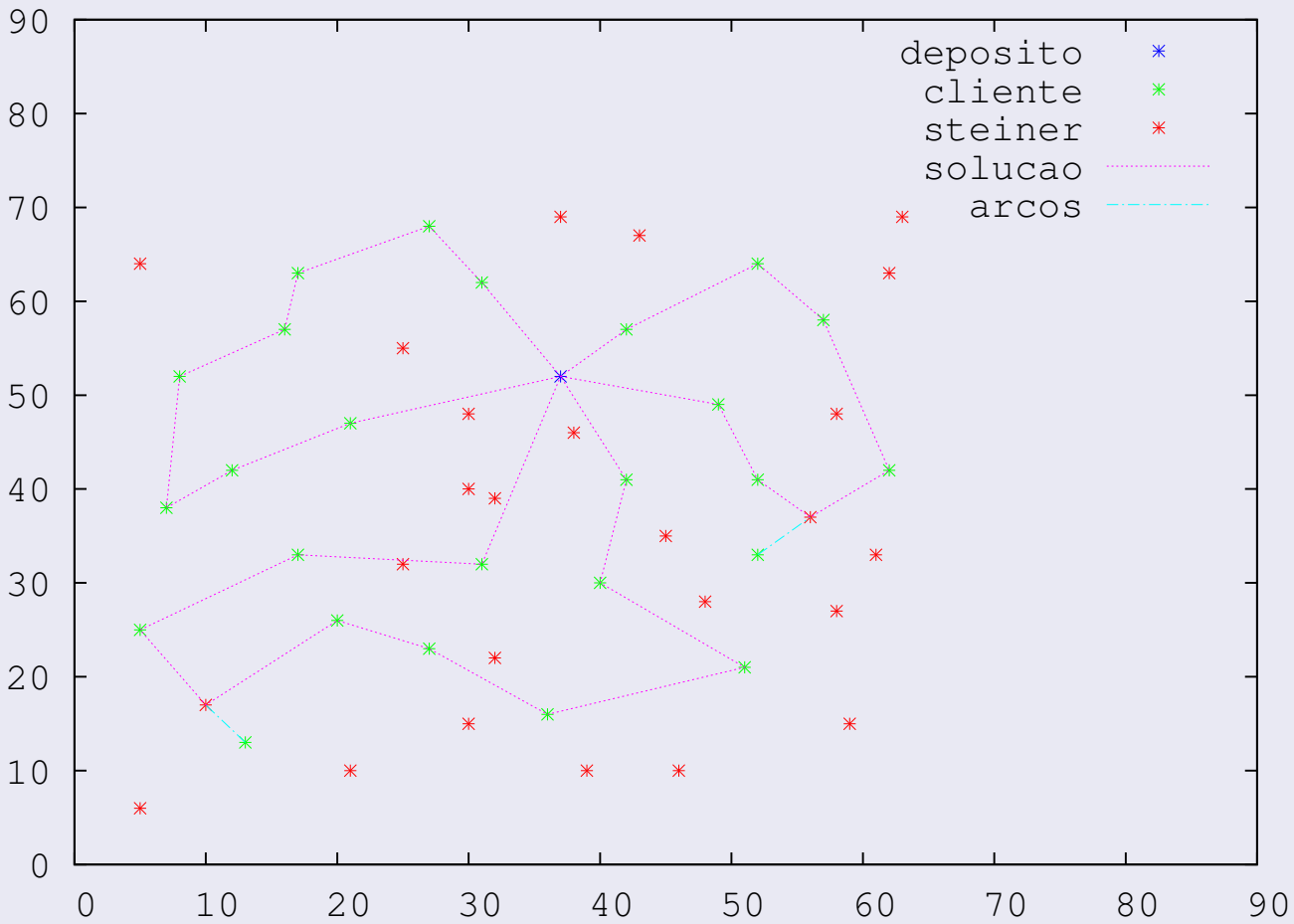


Figure: A solution for eil51.tsp with $m = 3$ e $Q = 10$.

Integer Programming (IP) Formulation

Formulation proposed by Baldacci et al and used into a *branch-and-cut* algorithm.

$$\text{(BC)} \quad \min \sum_{e \in E} c_e x_e + \sum_{ij \in A} w_{ij} z_{ij}$$

$$\text{subject to} \quad \sum_{e \in \delta(0)} x_e = 2m \tag{1}$$

$$\sum_{e \in \delta(i)} x_e = 2y_i, \forall i \in V \setminus \{0\} \tag{2}$$

$$\sum_{ij \in A} z_{ij} + y_i = 1, \forall i \in U \tag{3}$$

$$\sum_{e \in \delta(S)} x_e \geq \frac{2}{Q} \sum_{i \in U} \sum_{j \in S : ij \in A} z_{ij}, \forall S \subseteq V \setminus \{0\} : S \neq \{\} \tag{4}$$

$$y \in \{0,1\}^{|V'|}, z_{ij} \in \{0,1\}, x_{ij} \in \{0,1\}. \tag{5}$$

Motivations

- Practical applications arising in telecommunications and logistics;
  - Large fiber optics networks design;
  - Logistics of product distribution;
  - School bus allocation.
- Just one exact algorithm for CmRSP is reported, to our knowledge;
- In general, set covering models provide tight relaxations.

Motivations

- Practical applications arising in telecommunications and logistics;
  - Large fiber optics networks design;
  - Logistics of product distribution;
  - School bus allocation.
- Just one exact algorithm for CmRSP is reported, to our knowledge;
- In general, set covering models provide tight relaxations.

Our Objective

- Evaluate the use of a set covering model for the CmRSP together a column generation algorithm to solve it.

Column Generation

$$
\begin{aligned}
(\mathrm{P}) \quad & \min c\lambda \\
\text{s.a.} \quad & A\lambda \geq 1, \forall i \in N && (6) \\
& \lambda \in \{0,1\}^p && (7)
\end{aligned}
$$

Column Generation

$$
\begin{aligned}
(\mathrm{P}) \quad & \min c\lambda \\
\text{s.a.} \quad & A\lambda \geq 1, \forall i \in N \qquad (6) \\
& \lambda \in \{0,1\}^p \qquad (7)
\end{aligned}
$$

- p is exponential in $|N|$, i.e., the total number of columns is big!

Column Generation

$$
\begin{aligned}
(\mathrm{P}) \quad & \min c\lambda \\
\text{s.a.} \quad & A\lambda \geq 1, \forall i \in N \qquad (6) \\
& \lambda \in \{0,1\}^p \qquad (7)
\end{aligned}
$$

- p is exponential in $|N|$, i.e., the total number of columns is big!
- Consider just a few number of the columns and **construct other columns implicitly** by solving the *pricing problem*.

Column Generation

$$
\begin{aligned}
(\mathrm{P}) \quad & \min c\lambda \\
\text{s.a.} \quad & A\lambda \geq 1, \forall i \in N \qquad (6) \\
& \lambda \in \{0,1\}^p \qquad (7)
\end{aligned}
$$

- p is exponential in $|N|$, i.e., the total number of columns is big!
- Consider just a few number of the columns and **construct other columns implicitly** by solving the *pricing problem*.
- $\overline{c}_p = c_p - \pi A_p$ (π: dual variables corresponding to constraints (6));

Column Generation

$$
\begin{aligned}
(\mathrm{P}) \quad & \min c\lambda \\
\text{s.a.} \quad & A\lambda \geq 1, \forall i \in N \qquad (6) \\
& \lambda \in \{0,1\}^p \qquad (7)
\end{aligned}
$$

- p is exponential in $|N|$, i.e., the total number of columns is big!
- Consider just a few number of the columns and **construct other columns implicitly** by solving the *pricing problem*.
- $\overline{c}_p = c_p - \pi A_p$ (π: dual variables corresponding to constraints (6));
- Pricing problem consists of finding $p \in P$ that minimizes $\overline{c}_p$.

Main Idea

- each column represents a Q-ring-star;
- a solution for the CmRSP consists of m columns such that:
  - each customer $i \in U$ is covered by some column;
  - the sum of the costs associated with each column is minimum.

Main Idea

- each column represents a Q-ring-star;
- a solution for the CmRSP consists of m columns such that:
  - each customer $i \in U$ is covered by some column;
  - the sum of the costs associated with each column is minimum.

$$\begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & \dots & & n \end{matrix} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & & 0 & 0 \\ 1 & 0 & 1 & 0 & & 0 & 1 \\ 1 & 1 & 1 & 0 & & 1 & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 & 0 \\ 1 & 0 & 1 & 1 & & 0 & 0 \\ 0 & 0 & 1 & 0 & & 0 & 1 \\ 0 & 1 & 1 & 0 & & 0 & 1 \end{bmatrix} \end{array} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{bmatrix} >= \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Figure: Example of a matrix of coefficients in a set covering model.

Set Covering Formulation

Consider

- $P = \{(R,S) | (R,S)$ is a Q-ring-star $\}$;
- $r^p \in \mathbb{Z}_+^{|U|+|W|+|E|}$ and $s^p \in \mathbb{Z}_+^{|U|+|A|}$: the characteristic vectors of the ring R and of the star S of a ring-star $p = (R,S)$;
- $u_e = 1$, if $e \in E \setminus \delta(0)$, otherwise, $u_e = 2$;
- $c_p = \sum_{e \in R(p)} c_e + \sum_{e \in S(p)} w_e$: the cost of a ring-star p.

Set Covering Formulation

$$\text{(F1)} \quad \min \sum_{p \in P} c_p \lambda_p$$

$$\text{subject to} \quad \sum_{p \in P} \lambda_p = m \tag{8}$$

$$\sum_{p \in P} (r_i^p + s_i^p)\lambda_p \geq 1, \forall i \in U \tag{9}$$

$$\sum_{p \in P} \sum_{e \in \delta_i} r_e^p \lambda_p \leq 1, \forall i \in V \setminus \{0\} \tag{10}$$

$$\sum_{p \in P} r_e^p \lambda_p \leq u_e, \forall e \in E \tag{11}$$

$$\sum_{p \in P} s_{ij}^p \lambda_p \leq 1, \forall ij \in A \tag{12}$$

$$\lambda_p \in \{0, 1\}, \forall p \in P \tag{13}$$

The Pricing Problem

Consider

- π, μ, ν, β and α: dual variables corresponding to constraints (8), (9), (10), (11) and (12);
- $\tilde{c}_e = c_e - \beta_e$;
- $\tilde{w}_{ij} = w_{ij} - \alpha_{ij} - \mu_i$;
- $\tilde{p}_i = \mu_i$;
- $\overline{c}_p = \sum_{e \in E} \tilde{c}_e r_e^p + \sum_{ij \in A} \tilde{w}_{ij} s_{ij}^p - \sum_{i \in U} \tilde{p} i r_i^p + \pi$.

The pricing problem consists of $\min_{p \in P} \overline{c}_p$.

Pricing Problem

Pricing Problem is NP-hard

The *profitable tour problem* can be reduced in polinomial time to the pricing problem ($W = \emptyset, Q = |V|$, $\tilde{w}_{ij} = \sum_{e \in E} \tilde{c}_e$).

Pricing Problem

Pricing Problem is NP-hard

The *profitable tour problem* can be reduced in polinomial time to the pricing problem ($W = \emptyset, Q = |V|$, $\tilde{w}_{ij} = \sum_{e \in E} \tilde{c}_e$).

Alternative

Relax the pricing problem to allow vertex repetition inside a ring-star!

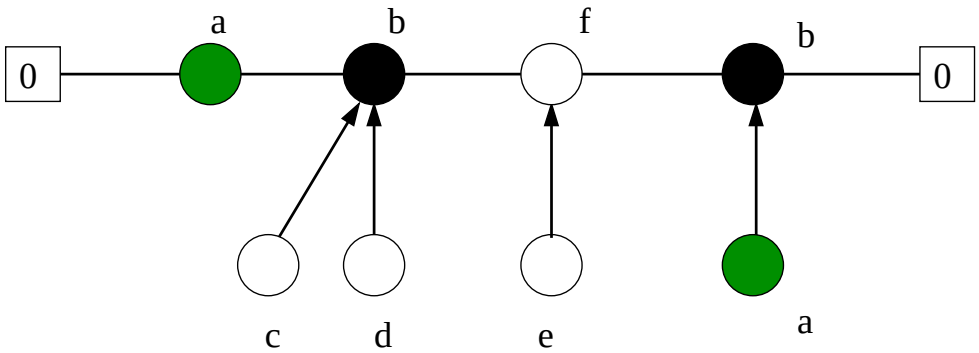


Figure: An example of a relaxed ring-star.

Relaxed Pricing Problem

We analyzed three relaxations:

- BPr: any repetition are allowed;
- BPkc: k-cycles are prohibited inside the component R of a ring-star (R, S);
  - k-**cycles** are cycles with length less than or equal to k;
- BPks: k-stream are prohibited inside a ring-star (R, S);
  - A **stream** is a string of vertices in $V[R]$ and $V[S]$ of a ring-star (R, S) with a fixed order;
  - k-**streams** are cycle with length less than or equal to k inside a stream;

Relaxed Pricing Problem

We analyzed three relaxations:

- BPr: any repetition are allowed;
- BPkc: k-cycles are prohibited inside the component R of a ring-star (R, S);
  - k-**cycles** are cycles with length less than or equal to k;
- BPks: k-stream are prohibited inside a ring-star (R, S);
  - A **stream** is a string of vertices in $V[R]$ and $V[S]$ of a ring-star (R, S) with a fixed order;
  - k-**streams** are cycle with length less than or equal to k inside a stream;

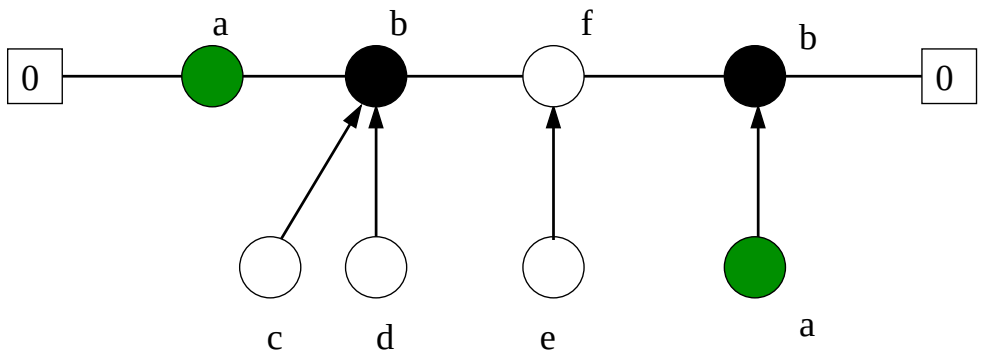


Figure: An example of a 2-cycle and a stream *acdbefab*.

Relaxed Pricing Problem

We analyzed three relaxations:

- BPr: any repetition are allowed;
- BPkc: k-cycles are prohibited inside the component R of a ring-star (R, S);
  - k-**cycles** are cycles with length less than or equal to k;
- BPks: k-stream are prohibited inside a ring-star (R, S);
  - A **stream** is a string of vertices in $V[R]$ and $V[S]$ of a ring-star (R, S) with a fixed order;
  - k-**streams** are cycle with length less than or equal to k inside a stream;

All these relaxations can be solved in pseudo-polynomial time using dynamic programming.

Solving the Relaxed Pricing Problem

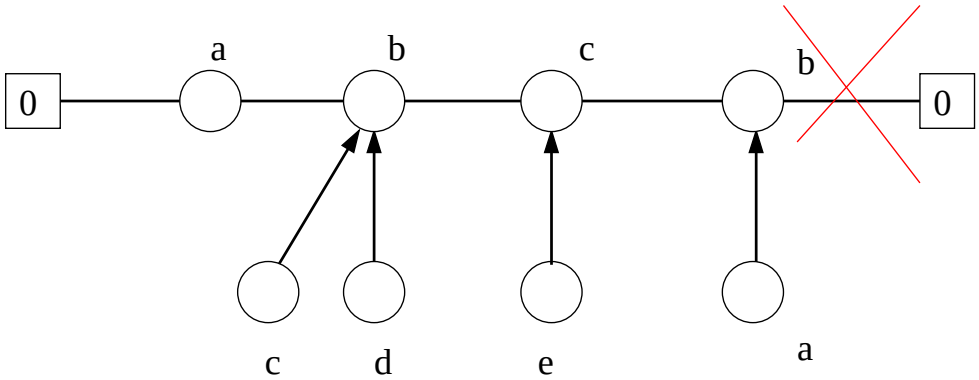


Figure: An example of a (q, j)-walk-star.

- $F(j, q)$: minimum weight of a (q, j)-walk-star.

Solving the Relaxed Pricing Problem

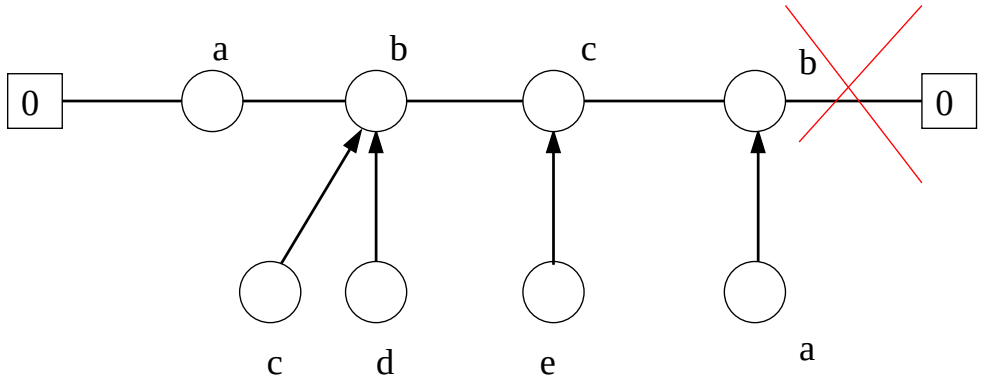


Figure: An example of a (q, j)-walk-star.

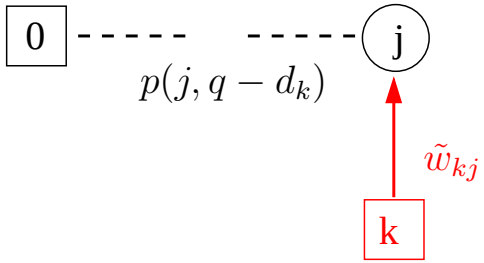
- $F(j, q)$: minimum weight of a (q, j)-walk-star.

Idea

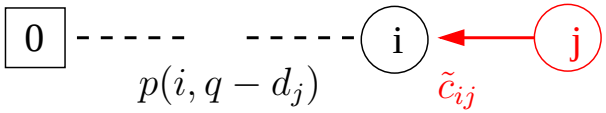
The minimum weight relaxed ring-star can be found by:

$$\min_{j \in V', q \in [1..Q]} F(j, q) + \tilde{c}_{j0}$$

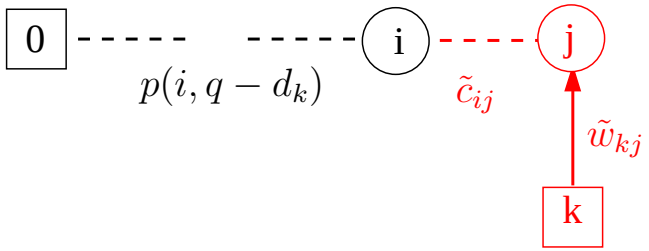
Computing $F(j, q)$



(a) adding a connection arc



(b) adding an edge



(c) adding a connection arc and an edge

Computing $F(j, q)$

$$F(j,q) = \min \begin{cases} \min\limits_{k \in U : j \in C_k} F(j, q-1) + \tilde{w}_{kj}, \\ \begin{cases} \min\limits_{i \in V, i \neq j} F(i, q-1) + \tilde{c}_{ij} + \tilde{p}_j & , \text{if } j \in U \\ \min\limits_{i \in V, i \neq j, k \in U : j \in C_k} F(i, q-1) + \tilde{w}_{kj} + \tilde{c}_{ij} & , \text{if } j \in W. \end{cases} \end{cases}$$

where $F(0,0) = 0$, $F(j,0) = \infty$ and $F(0, q > 0) = \infty$.

Computing $F(j, q)$

$$F(j,q) = \min \begin{cases} \min\limits_{k \in U : j \in C_k} F(j, q-1) + \tilde{w}_{kj}, \\ \begin{cases} \min\limits_{i \in V, i \neq j} F(i, q-1) + \tilde{c}_{ij} + \tilde{p}_j & , \text{if } j \in U \\ \min\limits_{i \in V, i \neq j, k \in U : j \in C_k} F(i, q-1) + \tilde{w}_{kj} + \tilde{c}_{ij} & , \text{if } j \in W. \end{cases} \end{cases}$$

where $F(0,0) = 0$, $F(j,0) = \infty$ and $F(0, q > 0) = \infty$.

Complexity

The computation of $F(j, q)$ is $O(|V|)$, so, the relaxed pricing problem can be solved in $O(|V|^2 Q)$.

Solving the Relaxed Pricing Problem with k-cycle and k-stream Elimination

- Similar idea proposed by Irnich and Villeneuve to solve the non-elementary shortest path with resource constraint;
- They used *label setting* algorithm to enumerate all resource feasible paths;
- Some dominance rules are used to remove resource feasible paths that are non-useful (paths that can not be extended to obtain an optimal solution);
- They proposed an algorithm, called Intersection Algorithm, to recognize useful paths.

Digraph of Intersection

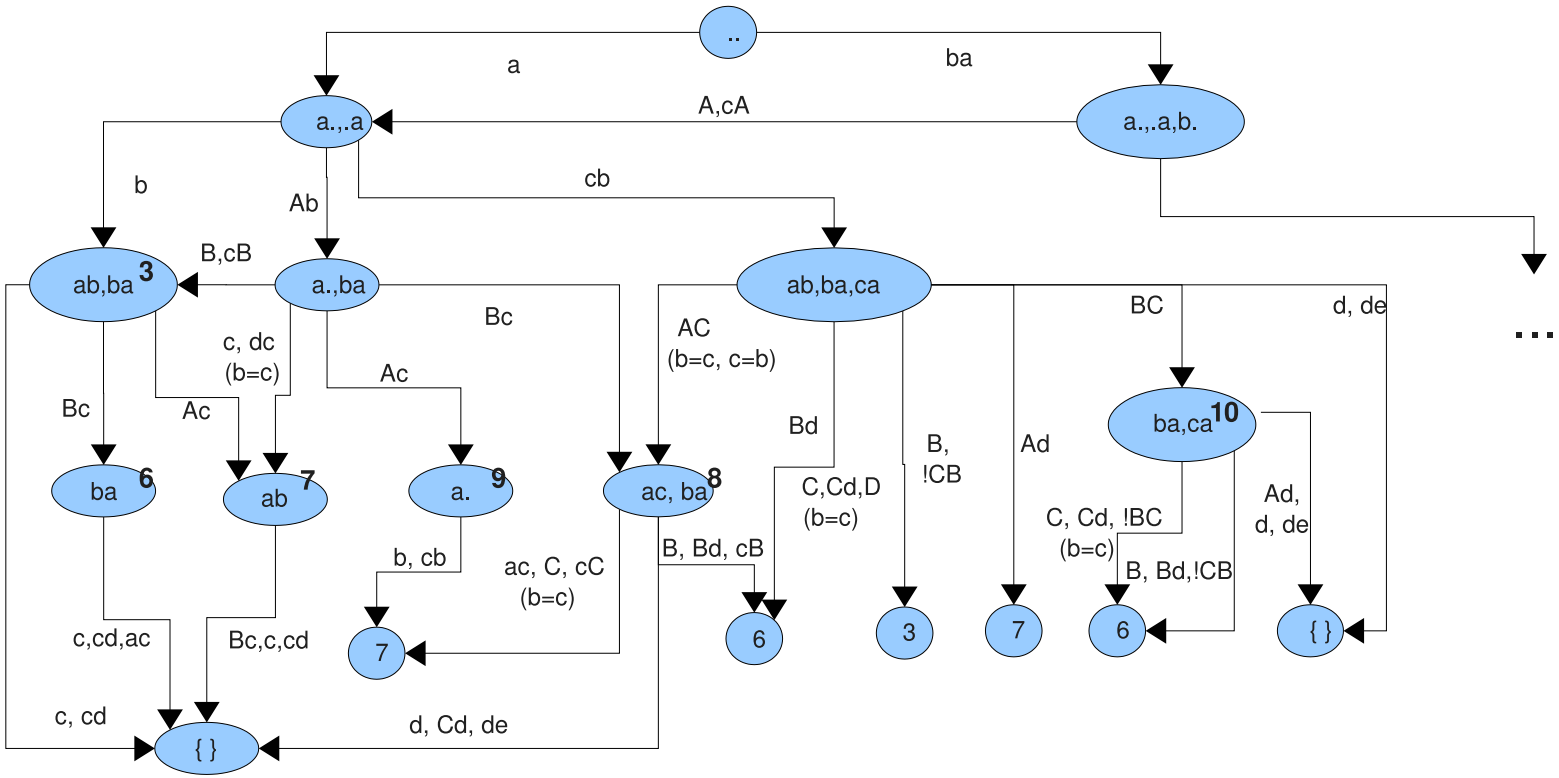


Figure: Part of the intersection digraph for $k = 3$.

We proposed to use a deterministic finite automaton to recognize an useful path.

Computational Environment

- Pentium IV 2.66GHz,900MB of RAM;
- language C, XPRESS optimizer library [a].

[a] XPRESS is a product of Dash Corporation.

Implementation

- We have implemented three *branch-and-price* algorithm, one for each kind of relaxation.
- The BC code is an implementation for the *branch-and-cut* algorithm proposed by Baldacci et al.

Branch-and-Price

Initial basis

We introduced artificial variables with huge costs whose columns form an identity matrix.

To accelerate convergence, we also included a set of columns corresponding to *ring-stars* that are part of a feasible solution generated by a naïve heuristic inspired in Mauttone et al.

Branch-and-Price

Initial basis

We introduced artificial variables with huge costs whose columns form an identity matrix.
To accelerate convergence, we also included a set of columns corresponding to *ring-stars* that are part of a feasible solution generated by a naïve heuristic inspired in Mauttone et al.

Branching rule

$$\sum_{e \in \delta(S)} \sum_{p \in P} r_e^p \lambda_p \geq 2 \left\lceil \frac{|\{i \in U : C_i \subseteq S\}|}{Q} \right\rceil, S \subseteq V \setminus \{0\}, S \neq \emptyset.$$

Branch-and-Price

Initial basis

We introduced artificial variables with huge costs whose columns form an identity matrix.
To accelerate convergence, we also included a set of columns corresponding to *ring-stars* that are part of a feasible solution generated by a naïve heuristic inspired in Mauttone et al.

Branching rule

$$\sum_{e \in \delta(S)} \sum_{p \in P} r_e^p \lambda_p \geq 2 \left\lceil \frac{|\{i \in U : C_i \subseteq S\}|}{Q} \right\rceil, S \subseteq V \setminus \{0\}, S \neq \emptyset.$$

Node selection

The classical best-bound strategy is used to select the next node to be explored during the enumeration procedure.

Bounding

- No heuristics were developed to compute feasible solutions during the enumeration. Primal bounds must correspond to IP solutions.
- Dual bounds are directly obtained from the linear relaxations or from Lasdon's formula.

Bounding

- No heuristics were developed to compute feasible solutions during the enumeration. Primal bounds must correspond to IP solutions.
- Dual bounds are directly obtained from the linear relaxations or from Lasdon's formula.

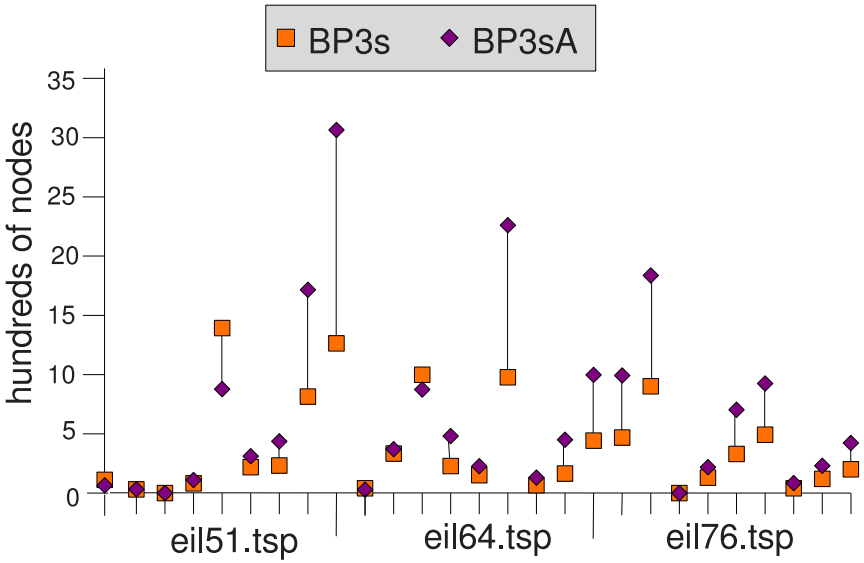
Pricing

- all columns with negative reduced cost associated to non relaxed *ring-stars*;
- for each $j \in V \setminus \{0\}$, the most negative reduced cost column related to the minimum weight relaxed (q, j)-*walk-star*.

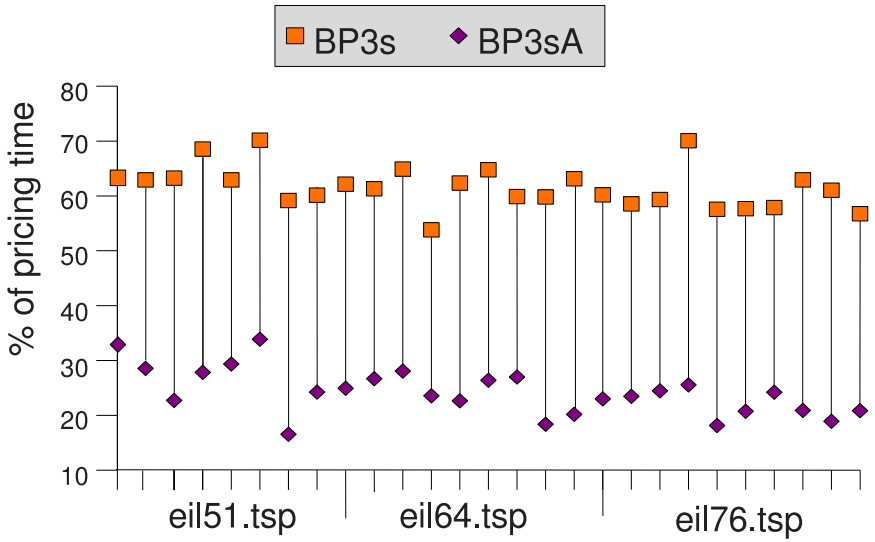
Table: Comparison between BPr, BP3c, BP3s, BP3sA and BC codes.

	BPr × BP3c			BP3c × BP3s		
instance	GAP	TIME	OPT	GAP	TIME	OPT
eil40.tsp	1.0	2.8	2/8	*	1.6	1/9
eil51.tsp	1.0	1.4	0/2	0.4	19.9	4/6
eil64.tsp	1.4	*	0/0	1.3	*	4/4
eil76.tsp	0.6	*	1/1	2.5	48.7	0/1

	BP3s × BP3sA			BC × BP3sA		
instance	GAP	TIME	OPT	GAP	TIME	OPT
eil40.tsp	*	3.3	0/9	*	0.4	0/9
eil51.tsp	0.2	3.5	0/6	-0.6	0.6	0/6
eil64.tsp	0.1	2.3	0/4	0.1	0.4	0/4
eil76.tsp	0.2	2.4	0/1	11.8	*	1/1

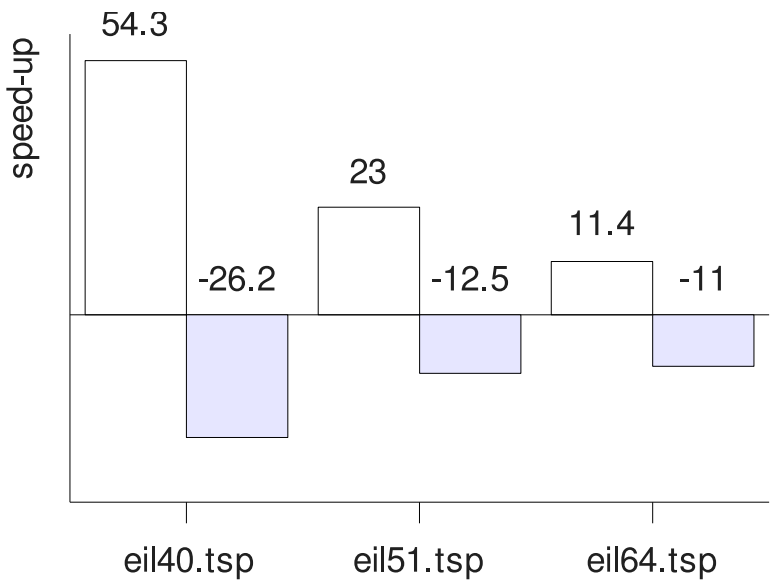


(a) nodes processed per instance

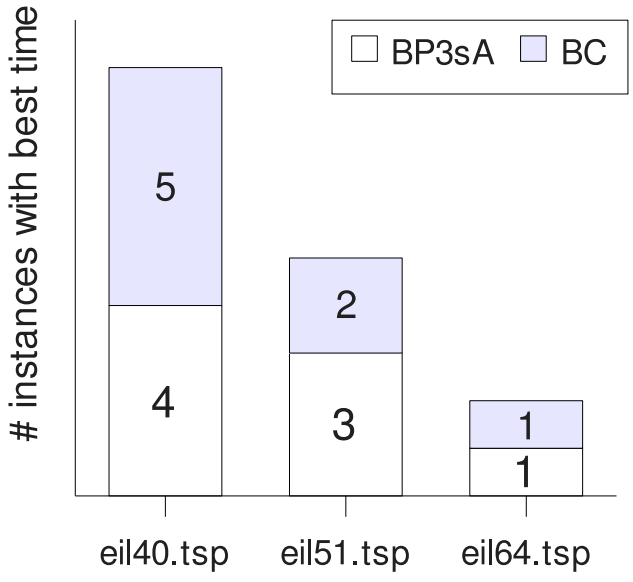


(b) Pricing time per instance

Figure: Comparison between `BP3s` and `BP3sA` codes.

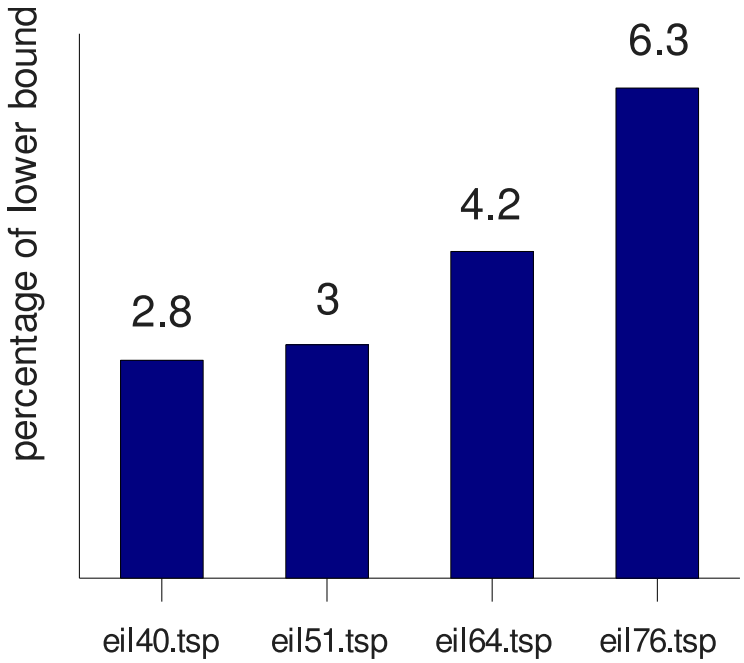


(a) # instances solved faster

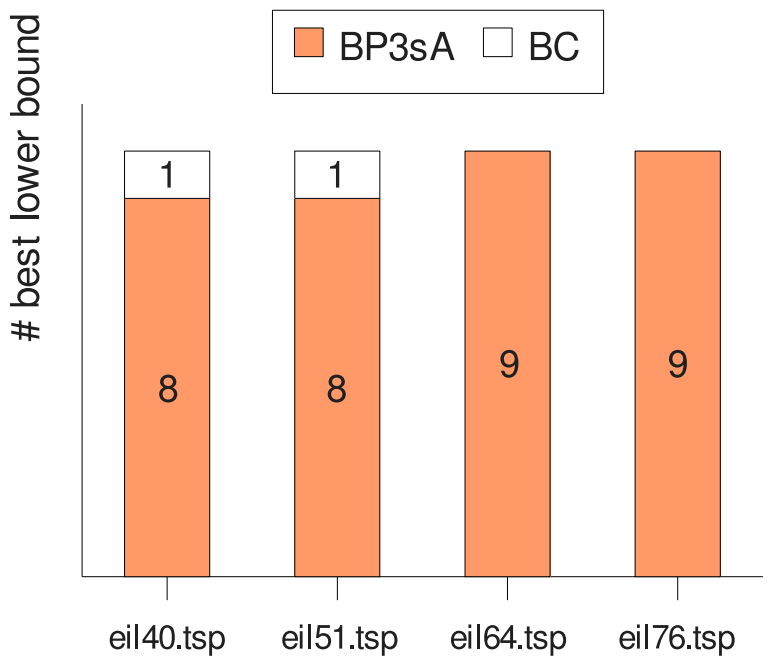


(b) Speedup: instances solved by both

Figure: Comparison between BP3sA and BC.



(a) Percentage of increase in lower bound at the root node



(b) Total of instances with best lower bound at the root node

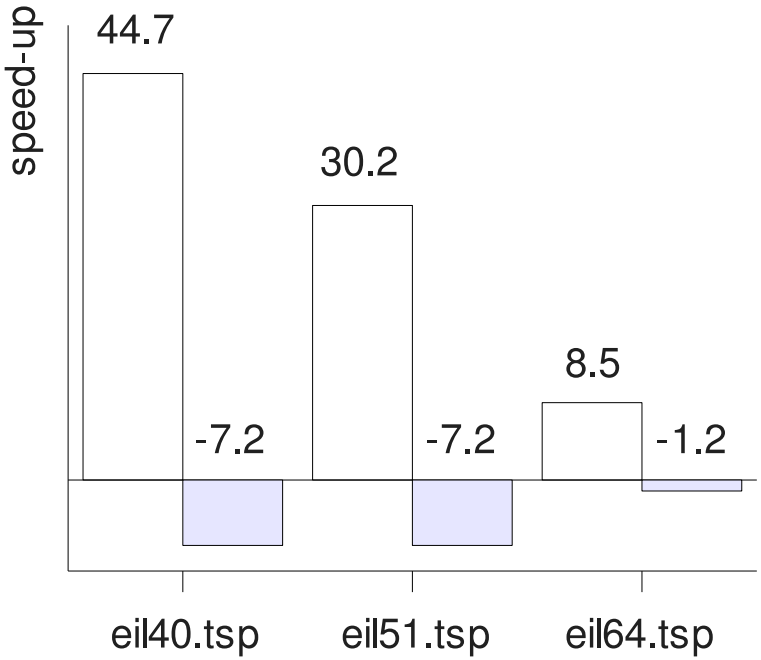
Figure: Lower bound comparative between `BP3sA` and `BC`.

Preliminary Results for Branch-and-Cut-and-Price (BCP)

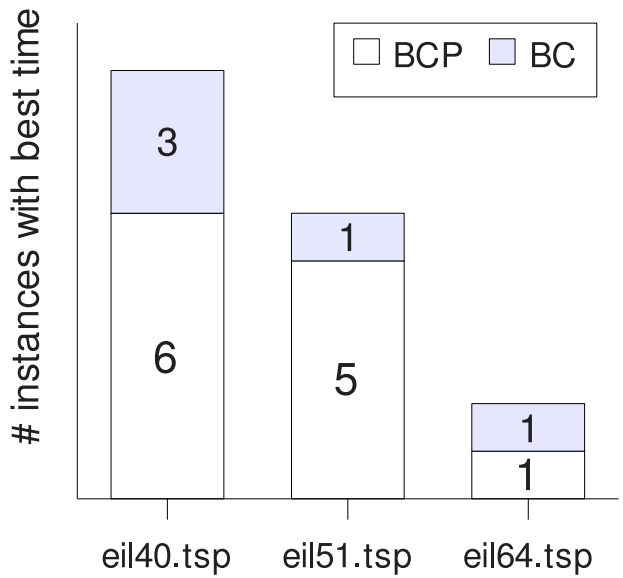
Table: Comparison between BP3sA, BC and BCP

	BP3sA × BCP			BC × BCP		
instance	GAP	TIME	OPT	GAP	TIME	OPT
eil40.tsp	*	1.7	0/9	*	2.6	0/9
eil51.tsp	0.9	1.8	1/7	0.0	0.8	0/7
eil64.tsp	0.2	2.2	0/4	0.1	2.5	0/4
eil76.tsp	0.1	1.0	0/1	0.2	*	1/1

Preliminary Results for Branch-and-Cut-and-Price (BCP)



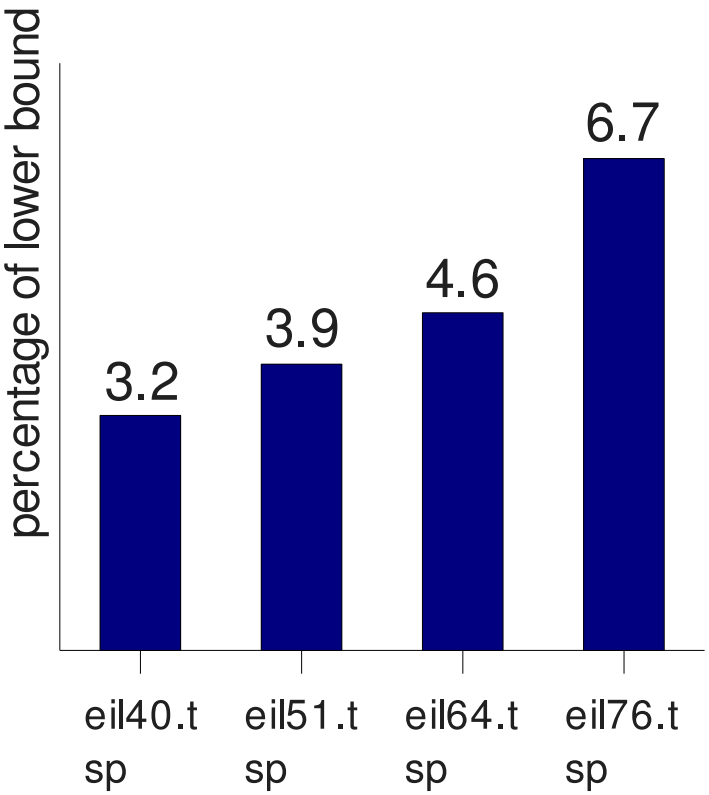
(a) # instances solved faster



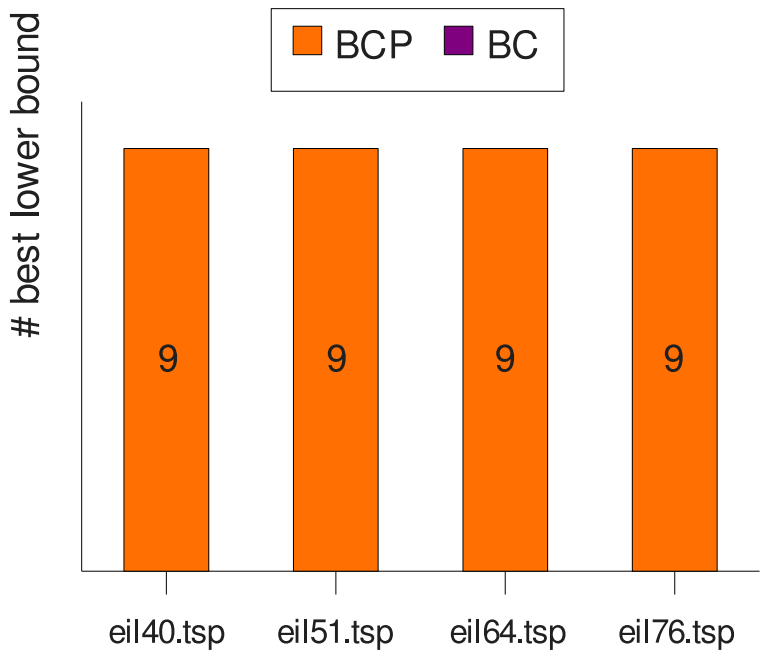
(b) Speedup: instances solved by both

Figure: Comparison between BCP and BC.

Preliminar Results for Branch-and-Cut-and-Price (BCP)



(a) Percentage of increase in lower bound at the root node



(b) Total of instances with best lower bound at the root node

Figure: Lower bound comparative between BCP and BC.

Conclusions

- The more complex are the structures forbidden in the relaxation, the larger is the number of instances that are solved to optimality;

Conclusions

- The more complex are the structures forbidden in the relaxation, the larger is the number of instances that are solved to optimality;
- more stringent relaxations lead to higher speedups in running times (early pruning);

Conclusions

- The more complex are the structures forbidden in the relaxation, the larger is the number of instances that are solved to optimality;
- more stringent relaxations lead to higher speedups in running times (early pruning);
- Using deterministic finite automaton was important to reduce the pricing time;

Conclusions

- The more complex are the structures forbidden in the relaxation, the larger is the number of instances that are solved to optimality;
- more stringent relaxations lead to higher speedups in running times (early pruning);
- Using deterministic finite automaton was important to reduce the pricing time;
- Set covering model presented better dual bounds;

Conclusions

- The more complex are the structures forbidden in the relaxation, the larger is the number of instances that are solved to optimality;
- more stringent relaxations lead to higher speedups in running times (early pruning);
- Using deterministic finite automaton was important to reduce the pricing time;
- Set covering model presented better dual bounds;
- The BP3sA and BC do not dominate each other and some instances are better suited for one or the other algorithm.

Further Works

- Introduce primal heuristic (to improve primal dual);
- Remove columns during branch-and-price (to reduce LP time);
- Add strong-branching (to obtain better dual bound early);
- Implement a *branch-and-cut-and-price* (on going).

The End

Questions?