Lp-based methods for solving routing problems

Ángel Corberán

Universitat de València, Spain

Spring School on Integrated Operational Problems
May 14-16, Troyes, France

Outline

1. Introduction
2. Polyhedral Combinatorics applied to some routing problems
   - The Traveling Salesman Problem (TSP)
   - The Stacker Crane Problem (SCP)
   - The Orienteering Arc Routing Problem (OARP)
   - The Generalized Directed Arc Routing Problem (Close Enough ARP)
3. Polyhedral Combinatorics (some theory)

Outline

1. Introduction

2. Polyhedral Combinatorics applied to some routing problems
   - The Traveling Salesman Problem (TSP)
   - The Stacker Crane Problem (SCP)
   - The Orienteering Arc Routing Problem (OARP)
   - The Generalized Directed Arc Routing Problem (Close Enough ARP)

3. Polyhedral Combinatorics (some theory)

Introduction

• From the pioneering works of Dantzig, Edmonds, and others, polyhedral (i.e. LP-based) methods have been successfully applied to the solution of many combinatorial optimization problems.

• Basically, these methods consist of trying to formulate the problem as a linear program and use the existing powerful methods of Linear Programming to solve it.

• The effectiveness of these methods is based on a good understanding of the polyhedron associated with the problem under study.

• In this talk we will briefly introduce some of the concepts and proof techniques of polyhedral theory, and will show how to apply it to the construction of effective optimization algorithms for some routing problems.

Introduction

• Most routing problems can be formulated as

$$\text{Min } \{c^T x : x \in S\},$$

where $x = \{x_1, \ldots, x_n\}$ is a vector of decision variables, $c = \{c_1, \ldots, c_n\} \in R^n_+$ is a vector of objective function coefficients (costs), and $S \subset Z^n$ is a set of feasible solutions.

• Given such a problem, it is natural to define an associated polyhedron

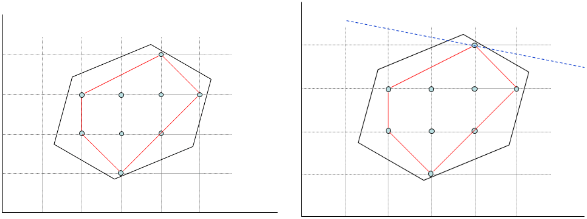
$$Conv(S),$$

the convex hull of the vectors in S.

• Usually, feasible solutions are associated with integer values of the decision variables. In such cases, $Conv(S)$ has integral vertices. Since the objective function is linear, $\text{Min } \{c^T x : x \in S\}$ is equivalent to $\text{Min } \{c^T x : x \in Conv(S)\}$.

Introduction

• It is well known that any polyhedron can be described by a set of linear inequalities, i.e., there is a matrix A and a vector b such that $Conv(S) = \{x \in R^n : Ax \leq b\}$. Hence, at least theoretically, our problem can be solved as a Linear Program.



• Unfortunately, complete linear descriptions of $Conv(S)$ are not known for any NP-hard problem. Only partial descriptions are known buteven partial linear descriptions can provide the basis for powerful algorithms.

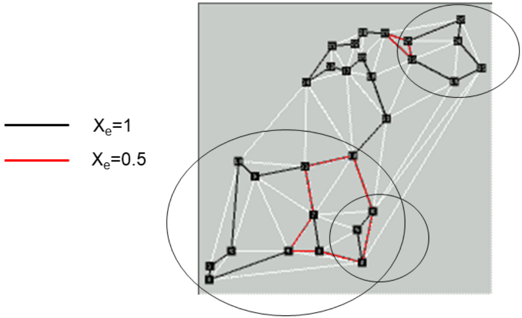
Introduction

- A problem which must be dealt with is that even a partial linear description frequently contains an exponential number of inequalities. Hence, in most cases, it will not be possible to solve an LP including all the inequalities explicitly. For instance, in the TSP:

|V|	Tours	Facets (differents)	Facet families	Facets in a vertex
3	1	0	0	0
4	3	3	1	2
5	12	20	2	10
6	60	100	4	27
7	360	3437	6	196
8	2520	194187	24	2600
9	20160	42,104,442	192	88911
10	181440	≥ 51,043,900,866	≥ 15379	≥ 13,607,980

Introduction

• An alternative is to start with a small subset of the known inequalities and compute the optimal LP solution subject to these constraints. Then, we check if any of the inequalities not in the current LP are violated by the optimal LP solution. If one or more violated inequalities are found, we add one or more of them to the current LP, solve it, and so on. If the LP solution obtained at the end of this process is a feasible solution of our original problem, then it is also optimal for that problem.

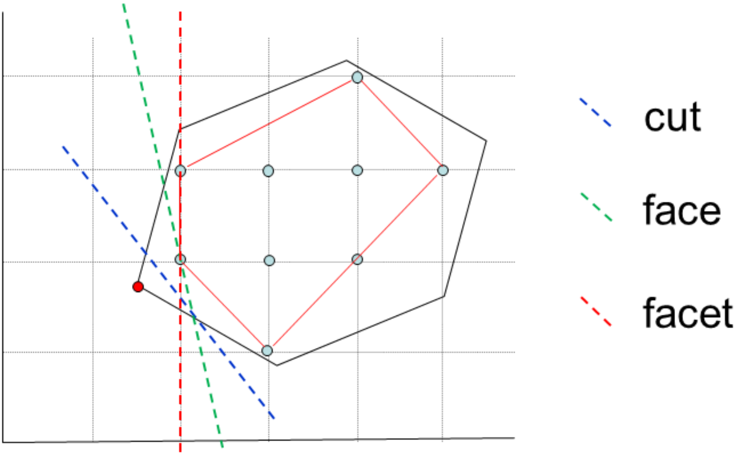


Introduction

• The linear inequalities which are added to the LP at each iteration of this process are called cutting planes (since they cut off the current LP solution).

• The whole procedure is a cutting-plane algorithm and has its origin in the celebrated work of Dantzig, Fulkerson, and Johnson (1954) on the TSP.

Cutting planes



• Note that the cutting-plane approach requires a method for identifying inequalities that are valid for $Conv(S)$ but violated by the current LP solution. Since usually the known valid inequalities fall into certain well-defined classes, for each known class we are faced with the following

Separation Problem: Given a class of valid inequalities and a point $\bar{x} \in R^n$, either find an inequality in this class which is violated by $\bar{x}$, or prove that no such inequality exists.

• Note that we are looking for a hyperplane separating $\bar{x}$ from $Conv(S)$. The separation problem can be solved by an exact or a heuristic method. In the last case, the algorithm may fail to find a violated inequality in the class, even if one exists.

Cutting-plane algorithm scheme

Step 1 (Initialization) Let (LP_0) be a linear relaxation of $Conv(S)$. Set $k = 0$.

Step 2 (LP Solver) Solve (LP_k). Let x^k be an optimal solution to (LP_k).

Step 3 (Separation) Solve the Separation Problem for x^k and some classes of valid inequalities for $Conv(S)$.

- Step 3.1 If $x^k \in S$, the x^k is optimal. Stop.
- Step 3.2 If one or more valid inequalities violated by x^k are found, add them to (LP_k) to define (LP_{k+1}). Set $k := k + 1$ and go to Step 2.
- Step 3.3 If no violated inequality is found, stop.

- If the algorithm ends at Step 3.1: Optimal solution.
- When it ends at Step 3.3 is because

  we don 't know all the inequalities describing $Conv(S)$, or (and)

  we don 't know how to find those which are violated.

  Then:

  $c^T x^k$ is a lower bound (in the Minimization case), and

  we can use the strengthened LP into a branch and bound or

  a branch and cut.

Introduction

• The first cutting-plane algorithm was proposed by Miliotis (1978) for the TSP.

• Grötschel, Jünger and Reinelt (1984) were the first authors using branch and cut (for the solution of the Linear Ordering Problem), but the name was introduced in Padberg and Rinaldi (1987) (for solving the TSP)

Outline

1. Introduction

2. Polyhedral Combinatorics applied to some routing problems
   - The Traveling Salesman Problem (TSP)
   - The Stacker Crane Problem (SCP)
   - The Orienteering Arc Routing Problem (OARP)
   - The Generalized Directed Arc Routing Problem (Close Enough ARP)

3. Polyhedral Combinatorics (some theory)

Routing Problems

Node and arc routing problems are those related to the traversal of some or all the nodes or arcs of a graph.

• Node routing problems: Traveling Salesman Problem (TSP), Capacitated Vehicle Routing Problem (CVRP), ...

• Arc routing problems: Chinese Postman Problem (CPP), Rural Postman Problem (RPP), Capacitated Arc Routing Problem (CARP), ...

Polyhedral Combinatorics applied to some routing problems

In this Section we will present some results obtained with the LP-based algorithms on some routing problems:

- The Traveling Salesman Problem
- The Stacker Crane Problem
- The Orienteering Arc Routing Problem
- The Generalized Directed Arc Routing Problem (Close Enough ARP)

The Traveling Salesman Problem (TSP)

The Traveling Salesman Problem

Definition

Given a collection of cities and the cost of travel between each pair of them, the traveling salesman problem, or TSP for short, is to find the cheapest way of visiting all of the cities and returning to your starting point.

http://www.math.uwaterloo.ca/tsp/index.html

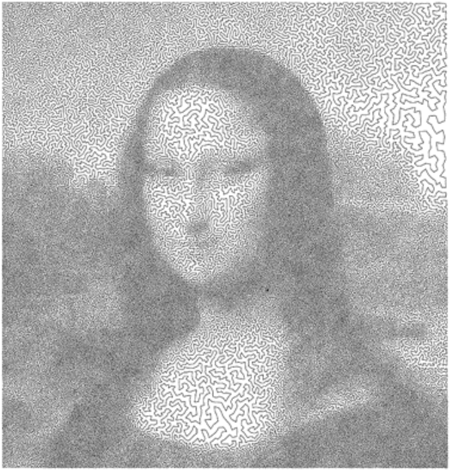
Applegate, Bixby, Chvátal, and Cook (The Traveling Salesman Problem: A Computational Study, 2006)

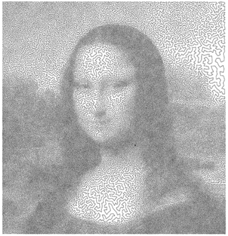
TSP Records

Dantzig, Fulkerson & Johnson	49	(1954)
Grötschel	120	(1977)
Crowder & Padberg	318	(1980)
Padberg & Rinaldi	532	(1987)
Grötschel & Holland	666	(1991)
Padberg & Rinaldi	2392	(1991)
Applegate, Bixby, Chvatal & Cook	7392	(1996)
Applegate, Bixby, Chvatal & Cook	13509	(1998)
Applegate, Bixby, Chvatal & Cook	15112	(2001)
Applegate, Bixby, Chvatal & Cook	24978	(2005)
Applegate, Bixby, Chvatal, Cook, Espinoza, Goycoolea & Helsgaun	85900	(2009)

The Traveling Salesman Problem

In 2009 Robert Bosch created a TSP instance with 100,000 cities giving a representation of the Leonardo da Vinci's Mona Lisa as a continuous line drawing.





The current best known results for the Mona Lisa TSP are

Tour: 5,757,191 (Y. Nagata, 2009)

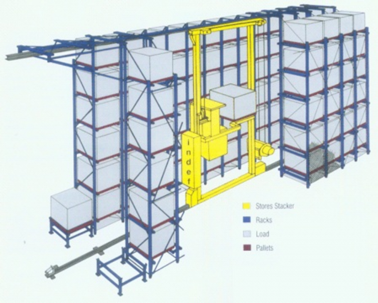
Lower bound: 5,757,084 (Concorde, 2012) A truncated B&C, using an artificial upper bound of 5,757,092. Bound obtained after 11.5 CPU years.

The Stacker Crane Problem (SCP)

The Stacker Crane Problem

Definition

To find the optimal sequence of movements of a crane (or mechanical arm) that has to move objects (for example containers) from a given origin to a given destination.



The Stacker Crane Problem

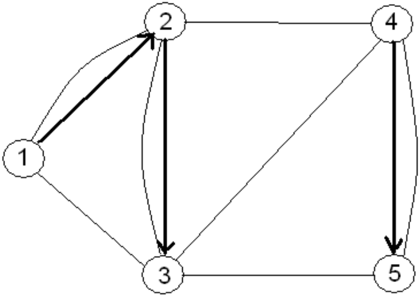
The Stacker Crane Problem (SCP) can be modeled as an Arc Routing Problem, a Pickup and Delivery Problem, or an Asymmetric Traveling Salesman Problem (ATSP).

Frederickson, Hecht, and Kim proposed the SCP in 1978. They defined it as an arc routing problem on a mixed graph $G = (V, E, A)$, where each link (arc or edge) (i, j) has associated a nonnegative cost c_{ij}.

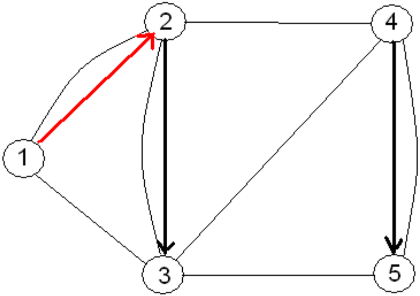
The objective is to find a minimum cost tour traversing at least once all the arcs in A.

They showed the SCP is NP-Hard and proposed a heuristic procedure, called CRANE, which has a worst-case ratio of 9/5 and $O(|V|^3)$ complexity.

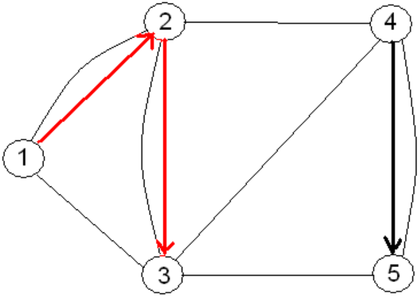
The Stacker Crane Problem



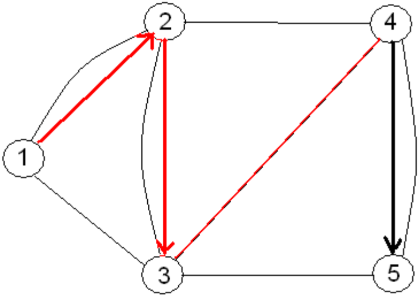
The Stacker Crane Problem



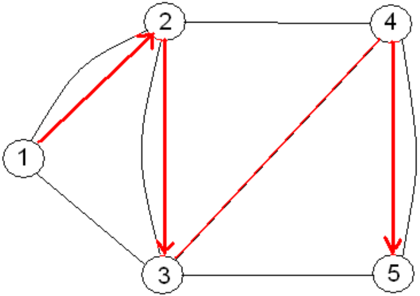
The Stacker Crane Problem



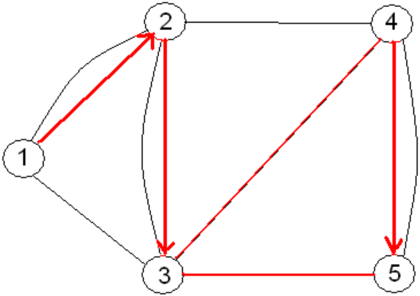
The Stacker Crane Problem



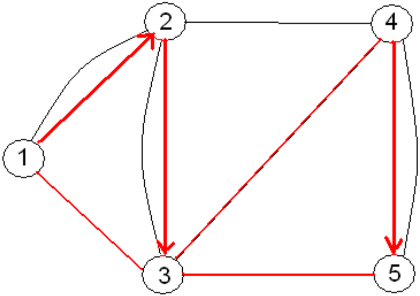
The Stacker Crane Problem



The Stacker Crane Problem



The Stacker Crane Problem



A formulation for the SCP

- Graph $G = (V, A)$ is, in general, disconnected. Let $V_1, V_2, \dots, V_p$ be the vertex sets of its p connected components. We call them *R*-sets.

- d_i^+ is the outdegree of vertex i.
- d_i^- is the indegree of vertex i.

- Let variables x_{ij}, x_{ji} represent the number of times edge $e = (i, j)$ is traversed from i to j and from j to i, respectively.

Ávila, C., Plana, and Sanchis (Networks, 2015)

A formulation for the SCP

Then, the SCP can be formulated as follows:

$$\text{Minimize} \quad \sum_{(i,j)\in E} c_{ij}(x_{ij} + x_{ji})$$

s.t.:

$$
\begin{aligned}
x(\delta^+(S)) &\geq 1, \quad \forall S = \bigcup_{i\in Q} V_i, \quad Q \subset \{1, 2, ..., p\} && (1)\\
x(\delta^+(i)) + d^+(i) &= x(\delta^-(i)) + d^-(i), \quad \forall i \in V && (2)\\
x_{ij}, x_{ji} &\geq 0, \quad \forall (i,j) \in E && (3)\\
x_{ij}, x_{ji} & \quad integer, \quad \forall (i,j) \in E && (4)
\end{aligned}
$$

A branch-and-cut algorithm for the SCP

Initial LP

- A connectivity constraint $x(\delta^+(S)) \geq 1$, for each R-set
- Symmetry equations $x(\delta^+(i)) + d^+(i) = x(\delta^-(i)) + d^-(i), \forall i$

The cutting plane uses separation algorithms for:

- Connectivity constraints (Heuristics and exact algorithm)
- K-C inequalities (Heuristic)
- Path-Bridge Inequalities (Heuristic)
- Asymmetric 2 Path-Bridge inequalities (Heuristic)

The branch-and-cut algorithm:

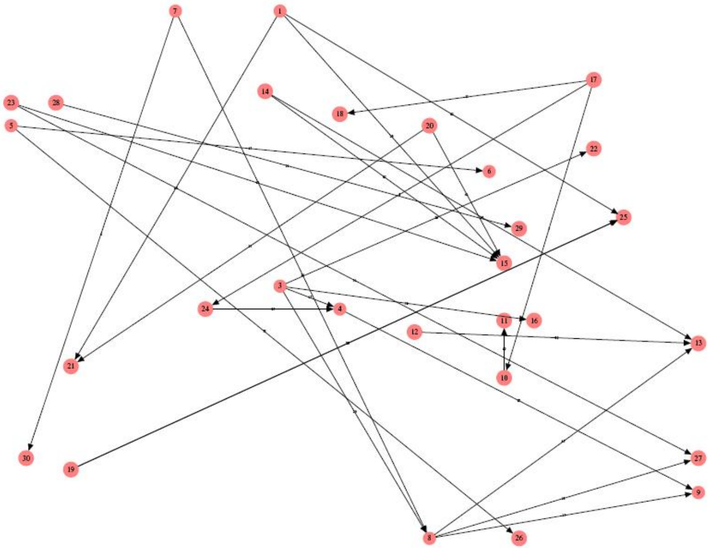
- Coded in C++
- Cplex 12.4
- Time limit: 1 hour.
- Run on a PC Intel Core i7 CPU 3.40 GHz, 16 GB RAM.

Computational Results on SCP instances

First set of SCP instances

14 SCP (drayage) instances similar to the "crane 2" instances proposed in Srour (2010) and Srour & van de Velde (2013) with the following characteristics:

- 5 instances with $n = 100$ points selected from the 100×100 square. 50 origins and 75 destinations.
- 2 instances with $n = 100$ points selected from the $10^6 \times 10^6$ square. 100 origins and 65 destinations.
- 5 instances with $n = 300$ points selected from the 500×500 square. 150 origins and 200 destinations.
- 2 instances with $n = 300$ points selected from the $10^6 \times 10^6$ square. 300 origins and 200 destinations.


7
1
17
14
23
28
18
5
20
22
6
25
29
15
3
24
4
11
16
12
13
21
10
27
30
19
9
8
26

Computational Results on SCP instances

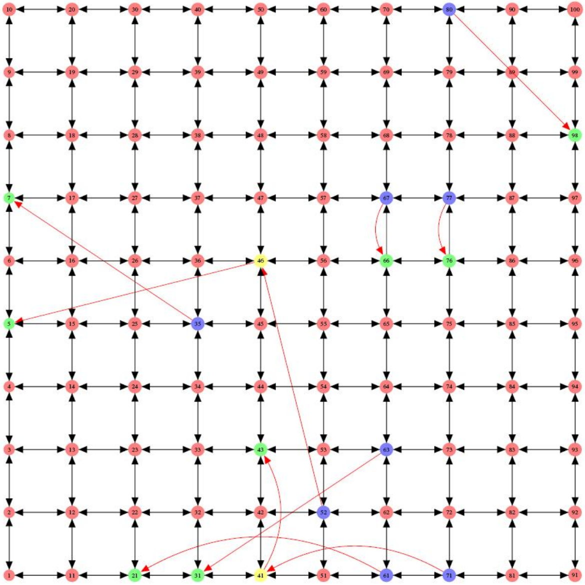
First set: Drayage instances

	Jobs	Gap0 (%)	# of opt.	T(scs)
cranegen100_50_75	100	0.00	5/5	0,1
cranegen100_100_65	100	0.07	2/2	1,2
cranegen300_150_200	300	0.00	2/2	4,4
cranegen300_300_200	300	0.0001	5/5	20,8

Second set of SCP instances

We have generated a new set of SCP instances defined on a grid:

- A grid $x \times x$ is generated.
- All the points $(1,1),(1,2),\ldots,(1,x),(2,1),\ldots(x,x)$ are the vertices of the graph.
- n origins and n destinations are generated for the n jobs (required arcs).
- Non required arcs: all the arcs in the grid.
- Arc costs are computed using the Manhattan distance.


10 20 30 40 50 60 70 80 90 100
9 19 29 39 49 59 69 79 89 99
8 18 28 38 48 58 68 78 88 98
7 17 27 37 47 57 67 77 87 97
6 16 26 36 46 56 66 76 86 96
5 15 25 35 45 55 65 75 85 95
4 14 24 34 44 54 64 74 84 94
3 13 23 33 43 53 63 73 83 93
2 12 22 32 42 52 62 72 82 92
1 11 21 31 41 51 61 71 81 91

Computational Results on SCP instances

Second set of SCP instances

14 SCP instances with the following characteristics:

- 5 instances with $n \in \{10, 50, 100, 200, 300\}$ jobs generated in a grid 50×50.
- 5 instances with $n = 300$ jobs of length at most 5 generated in a grid 50×50.
- 4 instances with $n = 500$ jobs of length at most 5 generated in a grid 100×100.

Computational Results on SCP instances

Second set: SCP instances defined on a grid

	Jobs	Grid	Solved?	Gap0 (%)	Nodes	T(scs)
cranegrid_r1	10	50×50	yes	0,00	3	76,7
cranegrid_r2	50	50×50	yes	0,00	0	0,4
cranegrid_r3	100	50×50	yes	0,00	0	0,7
cranegrid_r4	200	50×50	yes	0,00	0	0,9
cranegrid_r5	300	50×50	yes	0,00	0	1,1
cranegrid_l1	300	50×50	yes	0,00	0	1334,3
cranegrid_l2	300	50×50	yes	0,00	0	6,1
cranegrid_l3	300	50×50	no	–	0	3600
cranegrid_l4	300	50×50	yes	0,00	2	37,7
cranegrid_l5	300	50×50	yes	0,00	0	36,7
cranegrid_l6	500	100×100	yes	0,01	5	451,3
cranegrid_l7	500	100×100	no	–	102	3600
cranegrid_l8	500	100×100	no	0,35	106	3600
cranegrid_l9	500	100×100	yes	0,05	19	620,2

Computational Results on DGRP instances

- Let $G = (V, A)$ be a directed graph.
- Let $V_R \subseteq V$ required vertices and $A_R \subseteq A$ required arcs.
- Directed General Routing Problem: to find a minimum cost tour visiting all the vertices in V_R and traversing all the arcs in A_R.

14×5 DGRP instances proposed by Blais & Laporte (2003):

- Randomly generated directed graphs with $|V| = 5000$ and $|A| = 50000$.
- Different proportions of required vertices and arcs. They are also randomly determinated.
- Arc costs randomly generated on $[10, 110]$.

Blais and Laporte (2003) instances

				Blais & Laporte[1]			Our results	
$\lvert V\rvert$	$\lvert A\rvert$	$\lvert V_R\rvert$	$\lvert A_R\rvert$	$\lvert V_{ATSP}\rvert$	# opt.	Time	# opt.	Time
5000	50000	1000	1000	2000	5/5	125.6	5/5	31.3
5000	50000	1000	1500	2500	5/5	193.6	5/5	51.8
5000	50000	1000	2000	3000	5/5	280.3	5/5	28.3
5000	50000	1000	2500	3500	4/5	374.9	5/5	21.9
5000	50000	1000	3000	4000	0/5	–	5/5	25.5
5000	50000	1500	1000	2500	5/5	183.1	5/5	37.3
5000	50000	2000	1000	3000	5/5	244.7	5/5	36.7
5000	50000	2500	1000	3500	5/5	314.5	5/5	57.4
5000	50000	3000	1000	4000	4/5	396.8	5/5	50.7
5000	50000	0	3000	3000	5/5	303.0	5/5	12.5
5000	50000	500	2500	3000	5/5	300.0	5/5	19.3
5000	50000	1500	1500	3000	5/5	269.2	5/5	32.5
5000	50000	2500	500	3000	5/5	226.1	5/5	57.8
5000	50000	3000	0	3000	4/5	273.9	5/5	347.2

[1] Sun Ultra Sparc Station 10 (resolution time limit: 5 min.)

The Orienteering Arc Routing Problem (OARP)

The Orienteering Arc Routing Problem

• In most routing problems, the objective is to service a given set of customers, with minimum cost.

• In others, the objective is to select some customers with maximum profit from a set of potential customers and to service them.

In Feillet, Dejax & Gendreau (2005) these problems are called routing problems with profits and are classified as:

- **Prize-collecting problems**: there is a lower bound on the total prize collected and the objective is to minimize the total cost.
- **Profitable problems**: the objective is to maximize the difference between the collected profits and the routing costs.
- **Orienteering problems**: there is an upper bound on the cost or length of the route and the collected profits are maximized.

The Orienteering Arc Routing Problem

• In Archetti, C., Plana, Sanchis and Speranza (2014, 2015, and 2016) the Orienteering and the Team Orienteering Arc Routing Problem have been studied.

• The study was motivated by a real life application related to carriers making auctions on the web for transportation services.

• A transportation service is represented by an arc, and consists of reaching a node with an empty truck, filling the truck with load, traversing the arc and downloading the truck completely.

• The carrier has a set of regular customers which need to be served.

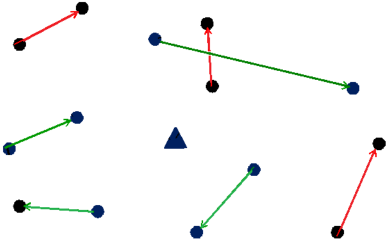
• The carrier has a vehicle or a fleet of vehicles with limited traveling time.

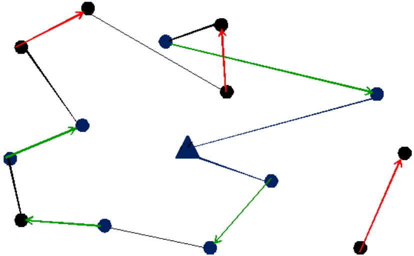
• The carrier looks for additional customers to fully use the traveling time of the vehicles.

The Orienteering Arc Routing Problem

Given a set of regular customers (green arcs) and
given a set of potential customers (red arcs),

we want to select a subset of potential customers with maximum profit that can also be serviced within the vehicle time limit.





The Orienteering Arc Routing Problem

- $G = (V, A)$ is a directed (strongly connected) graph. Vertex 1 is the depot.
- $A_R \subseteq A$ are the required arcs (its service is mandatory).
- $A_P \subseteq A$ are the optional arcs (its service is not mandatory).
- $s_{ij} \geq 0$ is the profit associated with each optional arc $(i, j) \in A_P$.
- $c_{ij} \geq 0$ is the traveling time associated with arc $(i, j) \in A$.
- A vehicle is available with a time limit T_{max}.

The Orienteering Arc Routing Problem

The Orienteering Arc Routing Problem consists of:

- finding a route starting and ending at the depot, such that
- its cost or time is no greater than T_{max},
- all the arcs in A_R are traversed at least once, and
- the sum of the profits of the arcs in A_P traversed is maximum.

We define the following variables:

- For each $(i,j) \in A$

  $x_{ij} =$ number of times that the vehicle traverses arc (i,j).

- For each $(i,j) \in A_P$

$$y_{ij} = \begin{cases} 1, & \text{if the vehicle services arc (i,j)} \\ 0, & \text{otherwise.} \end{cases}$$

The Orienteering Arc Routing Problem

Then, the OARP can be formulated as follows:

$$\text{Maximize} \quad \sum_{(i,j)\in A_P} s_{ij} y_{ij}$$

$$\sum_{j\in V\setminus\{i\}} x_{ij} = \sum_{j\in V\setminus\{i\}} x_{ji} \quad \forall i \in V \tag{1}$$

$$\sum_{i\in V\setminus S,\ j\in S} x_{ij} \geq 1 \quad \forall S \subset V\setminus\{1\} \ \text{ with } \ A_R(S) \neq \emptyset \tag{2}$$

$$\sum_{i\in V\setminus S,\ j\in S} x_{ij} \geq y_a \quad \forall S \subset V\setminus\{1\},\ \forall a \in A_P(S) \tag{2'}$$

$$x_{ij} \geq 1 \quad \forall (i,j) \in A_R \tag{3}$$

$$x_{ij} \geq y_{ij} \quad \forall (i,j) \in A_P \tag{3'}$$

$$\sum_{(i,j)\in A} c_{ij} x_{ij} \leq T_{max} \tag{4}$$

$$x_{ij} \geq 0 \text{ and integer} \quad \forall (i,j) \in A \tag{5}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in A_P \tag{6}$$

Initial LP:

$$\text{Maximize} \quad \sum_{(i,j)\in A_P} s_{ij} y_{ij}$$

$$\sum_{j\in V\setminus\{i\}} x_{ij} = \sum_{j\in V\setminus\{i\}} x_{ji} \quad \forall i\in V \tag{1}$$

$$\sum_{i\in V\setminus S,\ j\in S} x_{ij} \geq 1 \quad \forall S\subset V\setminus\{1\} \quad R\text{-component} \tag{2}$$

$$x_{ij} \geq 1 \quad \forall (i,j)\in A_R \tag{3}$$

$$x_{ij} \geq y_{ij} \quad \forall (i,j)\in A_P \tag{3'}$$

$$\sum_{(i,j)\in A} c_{ij} x_{ij} \leq T_{max} \tag{4}$$

$$x_{ij} \geq 0 \quad \forall (i,j) \in A \tag{5}$$

$$y_{ij} \in [0,1] \quad \forall (i,j) \in A_P \tag{6}$$

Cutting plane strategy

We use separation algorithms for the following inequalities:

1. Connectivity (heuristic).
2. Connectivity (exact).
3. KC and 2-PB (heuristic)

Computational results on OARP instances

- Run with a time limit of 1 hour.
- OARP instances randomly generated in a 1000×1000 square. Each vertex is incident with 4 entering arcs and 4 leaving arcs. $p1 = 0.2, 0.4, 0.6, 0.8$ (probability of required or optional arc) and $p2 = 0, 0.25, 0.50, 0.75$ (the percentage of required arcs among the "service" arcs).
- The OARP instances have $1000 \leq |V| \leq 2000$ and $7000 \leq |A| \leq 14000$.

Computational results on OARP instances

Set	$\|A_R\|$	$\|A_P\|$	# opt	Gap0	Gap	% profit	Nodes	Time
1000_2_0	0	1436.7	4/5	0.1405	0.61	80.1	1507.6	1050.0
1000_2_2	368.0	1068.7	5/5	0.0021		83.8	396.6	225.8
1000_2_5	715.5	721.2	5/5	0.0042		81.2	343.0	174.5
1000_2_7	1072.5	364.2	5/5	0.0110		89.2	100.8	54.3
1000_4_0	0	2761.0	5/5	0.0009		80.0	397.0	372.7
1000_4_2	677.5	2083.5	5/5	0.0013		81.5	716.0	567.6
1000_4_5	1384.5	1376.5	5/5	0.0015		85.8	775.0	551.1
1000_4_7	2063.7	697.2	5/5	0.0049		90.1	395.6	263.5
1000_6_0	0	4198.5	5/5	0.0002		88.6	441.6	610.0
1000_6_2	1044.5	3154.0	5/5	0.0003		83.8	656.2	697.8
1000_6_5	2097.5	2101.0	5/5	0.0008		86.9	301.2	391.2
1000_6_7	3168.7	1029.7	5/5	0.0029		87.5	382.6	410.3
1000_8_0	0	5592.0	5/5	0.0001		83.4	211.8	583.2
1000_8_2	1420.2	4171.7	5/5	0.0002		94.2	481.2	603.7
1000_8_5	2782.5	2809.5	5/5	0.0005		91.2	973.8	1239.2
1000_8_7	4219.7	1372.2	5/5	0.0017		85.4	530.8	715.7
Average	1313.4	2183.6	79/80	0.0108	0.61	85.8	538.2	531.9

Table: Results on instances with 1000 vertices and 7000 arcs

Computational results on OARP instances

Set	$\|A_R\|$	$\|A_P\|$	# opt	Gap0	Gap	% profit	Nodes	Time
1500_2_0	0	2182.0	3/5	0.0049	0.006	88.4	965.6	913.6
1500_2_2	538.6	1643.4	5/5	0.0007		87.5	107.2	186.1
1500_2_5	1089.8	1092.2	5/5	0.0021		82.8	582.2	641.0
1500_2_7	1631.8	550.2	5/5	0.0070		88.0	272.2	297.2
1500_4_0	0	4186.2	5/5	0.0002		86.4	229.0	546.8
1500_4_2	1043.8	3142.4	5/5	0.0006		90.1	221.0	497.8
1500_4_5	2080.4	2105.8	5/5	0.0007		90.0	470.0	738.8
1500_4_7	3133.8	1052.4	5/5	0.0023		84.7	653.2	1009.0
1500_6_0	0	6269.6	4/5	0.0004	0.001	82.2	387.4	1095.4
1500_6_2	1573.4	4696.2	5/5	0.0001		90.7	321.8	769.7
1500_6_5	3114.0	3155.6	5/5	0.0003		88.6	789.8	1874.5
1500_6_7	4701.4	1568.2	4/5	0.0014	0.001	93.0	570.2	1029.7
1500_8_0	0	8411.6	5/5	0.0001		85.5	203.0	1810.5
1500_8_2	2114.6	6297.0	5/5	0.0001		86.7	215.8	724.1
1500_8_5	4195.6	4216.0	5/5	0.0001		81.8	177.4	1120.1
1500_8_7	6323.2	2088.4	5/5	0.0005		83.1	409.2	1818.6
Average	1971.3	3291.1	76/80	0.0014	0.003	85.7	423.4	947.6

Table: Results on instances with 1500 vertices and 10500 arcs

Computational results on OARP instances

Set	$\|A_R\|$	$\|A_P\|$	# opt	Gap0	Gap	% profit	Nodes	Time
2000_2_0	0	2893.0	2/5	0.0120	0.0158	86.3	422.0	870.2
2000_2_2	720.4	2172.6	4/5	0.0006	0.0008	84.7	318.8	909.0
2000_2_5	1467.4	1425.6	5/5	0.0009		85.1	374.4	820.5
2000_2_7	2173.6	719.4	5/5	0.0049		85.7	372.0	617.5
2000_4_0	0	5537.6	3/5	0.0002	0.0004	91.5	325.3	1959.9
2000_4_2	1382	4155.6	5/5	0.0004		88.4	693.2	2180.1
2000_4_5	2789.4	2748.2	4/5	0.0006	0.0010	86.7	587.0	1705.5
2000_4_7	4147.4	1390.2	5/5	0.0016		84.3	611.8	1764.2
2000_6_0	0	8345.4	4/5	0.0003	0.0002	81.6	272.5	2947.6
2000_6_2	2065.8	6279.6	5/5	0.0001		77.7	166.4	898.7
2000_6_5	4139.2	4206.2	4/5	0.0002	0.0004	86.9	238.3	975.2
2000_6_7	6302	2043.4	5/5	0.0007		84.8	183.8	1984.0
2000_8_0	0	11182.6	2/5	0.0003	0.0004	88.1	212.5	3163.4
2000_8_2	2785.8	8396.8	5/5	0.0000		79.7	232.6	1506.9
2000_8_5	5608.6	5574.0	4/5	0.0002	0.0006	91.0	189.3	2548.3
2000_8_7	8395.2	2787.4	2/5	0.0013	0.0015	85.2	551.5	3209.0
Average	2623.6	4366.1	64/80	0.0015	0.0018	85.5	359.0	1878.4

Table: Results on instances with 2000 vertices and 14000 arcs

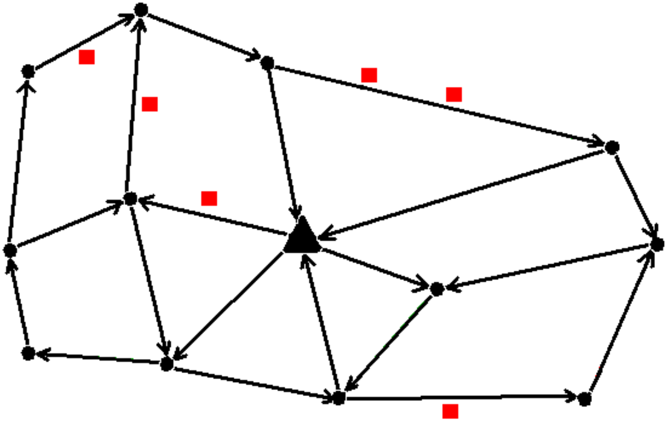
The Generalized Directed Arc Routing Problem (Close Enough ARP)

The Generalized Directed Arc Routing Problem (Close Enough ARP)

• In most arc routing problems, the service to a given customer is usually modeled as the traversal of a given arc (or edge) of the graph.

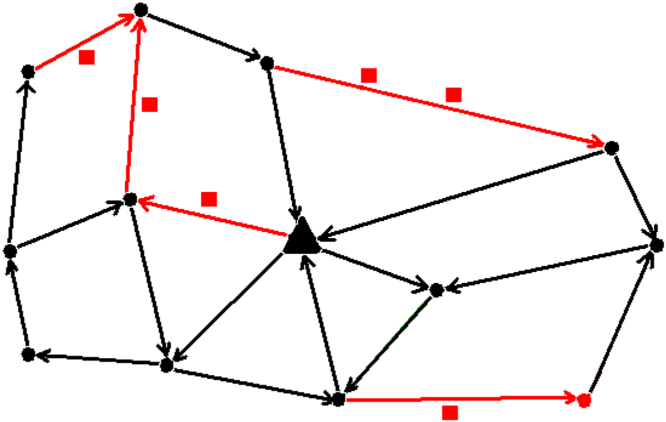
• For example, consider the meter reading problem (gas, electricity, water): to read the meter, the house has to be visited and the corresponding street has to be traversed.

The Generalized Directed Arc Routing Problem (Close Enough ARP)



The Generalized Directed Arc Routing Problem (Close Enough ARP)

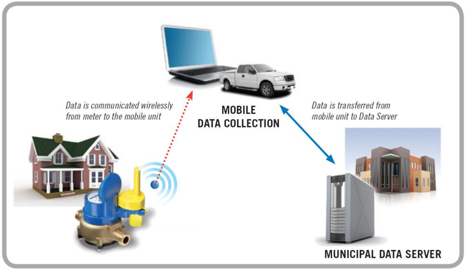
Then, the objective is to traverse a given set of (required) arcs at minimum cost



The Generalized Directed Arc Routing Problem (Close Enough ARP)

Now suppose that:

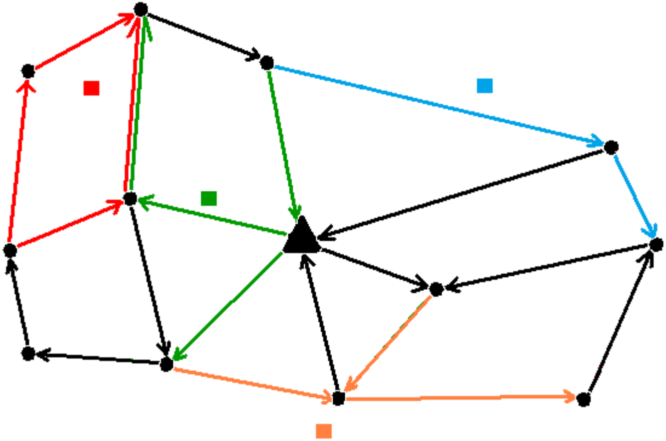
- Each meter has a RFID (Radio Frequency IDentification) tag.
- A RFID reader can read the data of any meter located closer than a given distance r.



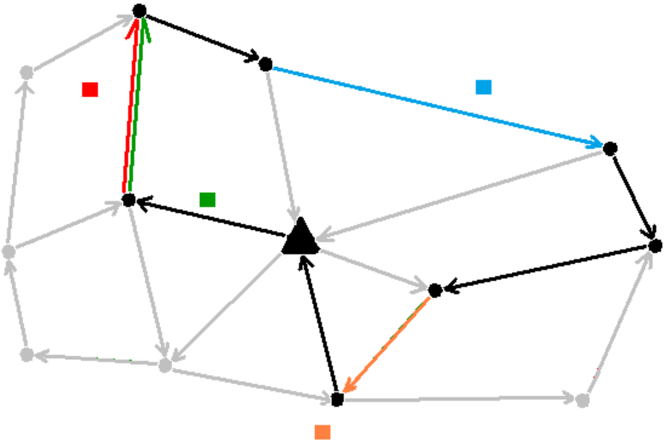
The Generalized Directed Arc Routing Problem (Close Enough ARP)

- Now, the service (meter reading) is not modeled as the traversal of a given street,
- The service is modeled as the traversal of a close-enough street (to the customer).

The Generalized Directed Arc Routing Problem



The Generalized Directed Arc Routing Problem



The Generalized Directed Arc Routing Problem

- Drexl (2007): Defines the problem as a generalization of the Directed Rural Postman Problem (DRPP) in his PhD Thesis.
- Shuttleworth, Golden, Smith & Wasil (2008): Define the problem from the perspective of its practical application. "Advances in Meter Readings: Heuristic Solution of the Close-Enough Traveling Salesman problem over a Street Network".
- Hà, Bostel, Langevin y Rousseau (2013): Formulation, branch-and-cut and very good results on instances with more than 500 vertices and 1500 arcs.

The Generalized Directed Arc Routing Problem

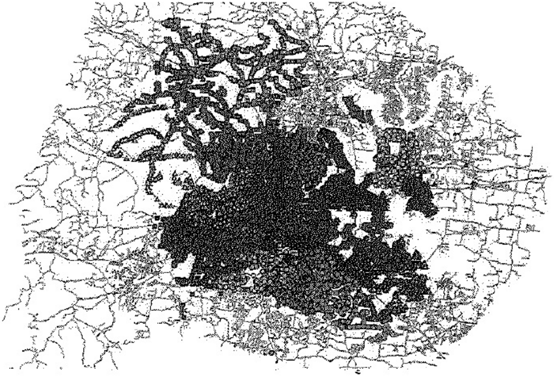


Figure: Real life instance with 150000 customers and 18 zones

The Generalized Directed Arc Routing Problem

Shuttleworth et al. (2008) propose several heuristics, obtaining:

- In a single zone: 24% improvement if $r = 500$ feet,
  18% improvement if $r = 350$ feet.
- Global improvement: 15% in length and 20% in time ($r = 500$).

The Generalized Directed Arc Routing Problem

- Let $G = (V, A)$ be a directed (strongly connected) graph. Vertex 1 is the depot.
- There is a cost $c_{ij} \geqslant 0$ for each arc $(i, j) \in A$.
- There is a family $\mathbb{H} = \{H_1, ..., H_L\}$, of "customers", each $H_c \subseteq A$

$$H_1 = \{a_{i_1}, \ldots, a_{i_{m1}}\} \quad H_2 = \{a_{j_1}, \ldots, a_{j_{m2}}\} \quad \ldots \quad H_L = \{a_{k_1}, \ldots, a_{k_{mk}}\}$$

The Generalized Directed Rural Postman Problem consists of

- finding a route starting and ending at the depot, such that
- it traverses at least one arc in each set H_c ,
- the length of the route is minimum.

The GDRPP generalizes

- The Directed Rural Postman Problem (DRPP) when $H_c = \{a_{ij}\}$ for each required arc (i, j)
- The Windy Rural Postman Problem (WRPP) when $H_c = \{a_{ij}, a_{ji}\}$ for each required edge (i, j)
- The Generalized Arc Routing Problem (GARP) when the H_c are disjoint sets (Fernández, 2013, undirected case)

Notation

- $A_R = H_1 \cup \cdots \cup H_L$ are the arcs that can service a customer
- Given $S \subset V$, $\quad A(S) = \{(i,j) \in A| \ i,j \in S\}$
- Given $S_1, S_2 \subset V$, $\quad (S_1 : S_2) = \{(i,j) \in A| \ i \in S_1, j \in S_2\}$
- $\delta^+(S) = \big(S : V \backslash S\big), \quad \delta^-(S) = \big(V \backslash S : S\big)$
- $\delta(S) = \delta^+(S) \cup \delta^-(S)$

Ávila, C. Plana, and Sanchis (Transportation Science, 2016)

We define the following variables:

- For each $(i,j) \in A$,

  $x_{ij} =$ number of times that arc (i,j) is traversed .

- For each $(i,j) \in A_R$,

  $$y_{ij} = \begin{cases} 1, & \text{if arc (i,j) is serviced (chosen to do the service) ,} \\ 0, & \text{otherwise.} \end{cases}$$

GDARP Problem formulation

$$\text{Minimize} \quad \sum_{(i,j)\in A} c_{ij} x_{ij}$$

$$x\left(\delta^{+}(i)\right) = x\left(\delta^{-}(i)\right) \quad \forall i \in V \tag{10}$$

$$x\left(\delta^{-}(S)\right) \geq 1 \quad \forall S \subset V \setminus \{1\}, \exists H_c \subset A_R(S) \tag{12}$$

$$x\left(\delta^{-}(S)\right) \geq y_a \quad \forall S \subset V \setminus \{1\},\ \forall a \in A_R(S) \tag{13}$$

$$x_{ij} \geq y_{ij} \quad \forall (i,j) \in A_R, \tag{14}$$

$$y\left(H_c\right) \geq 1 \quad \forall H_c \tag{15}$$

$$x_{ij} \geq 0 \text{ and integer} \quad \forall (i,j) \in A \tag{16}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in A_R \tag{17}$$

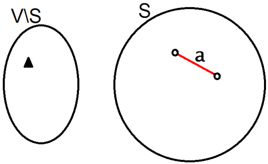
GDRPP polyhedron

We define $GDRPP(G)$ as the convex hull of all the vectors $(x, y) \in R^{|A|+|A_R|}$ satisfying inequalities (10) to (17).

- $GDRPP(G)$ is an unbounded polyhedron if G is strongly connected.
- $dim(GDRPP(G)) = |A| + |A_R| - |V| + 1$ iff $|H_c| \geq 2, \forall H_c \in \mathbb{H}$.
- The following inequalities are facet-defining under mild conditions:
  - Trivial inequalities $x_{ij} \leq 0$, $y_{ij} \leq 0$, $y_{ij} \leq 1$.
  - Traversing inequalities $x_{ij} \geq y_{ij}$.
  - Service inequalities $\sum_{(i,j)\in H_c} y_{ij} \geq 1$.

Connectivity constraints

Connectivity constraints (2) can be improved.

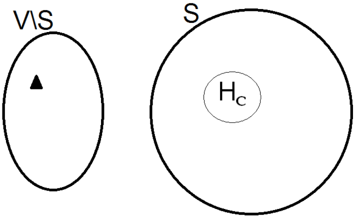


$$x\left(\delta^{-}(S)\right) \geq y_a$$

$$\forall S \subset V \setminus \{1\}, \ \forall a \in A_R(S)$$

Connectivity constraints

Connectivity constraints: If there is a set $H_c \subseteq A(S) \cup \delta(S)$,



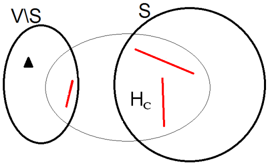
$x\left(\delta^{-}(S)\right) \geq 1$ is valid

and facet-defining.

Connectivity constraints

Connectivity constraints: (2)

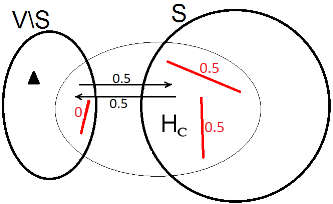
If there is NO set $H_c \subseteq A(S) \cup \delta(S)$, the inequality



$\cancel{x\left(\delta^-(S)\right) \geq 1}$ is not valid

Connectivity constraints: (2)

If there is NO set $H_c \subseteq A(S) \cup \delta(S)$, the inequality



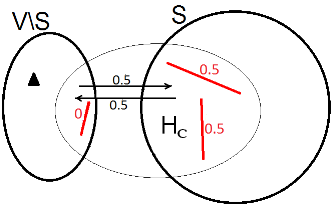
$\cancel{x\left(\delta^{-}(S)\right) \geq 1}$ is not valid

$x\left(\delta^{-}(S)\right) \geq y_a$ is valid

but not violated

Connectivity constraints: (2b)

However, these other Connectivity constraints (2b) are also valid and violated:



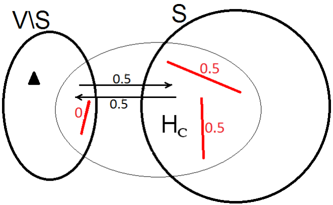
$$x\left(\delta^{-}(S)\right) \geq 1 - y(H_c \cap A(V \setminus S))$$

They define facets of GDRPP(G).

Connectivity inequalities

Connectivity inequalities (summary):

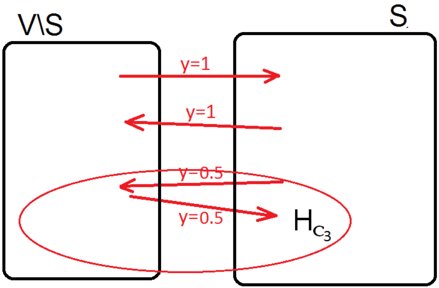
- $x\left(\delta^{-}(S)\right) \geq y_a$ when there is NO a subset $H_c \subseteq A(S) \cup \delta(S)$.
- $x\left(\delta^{-}(S)\right) \geq 1$ when there is a $H_c \subseteq A(S) \cup \delta(S)$.



New connectivity inequalities:

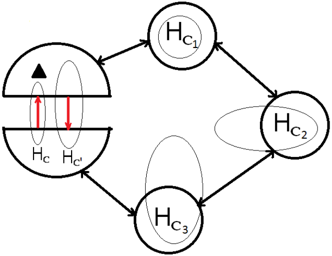
$$x\left(\delta^{-}(S)\right) \geq 1 - y(H_c \cap A(V \setminus S))$$

Parity inequalities



$$x(\delta(S)) \geq 2y_{a_1} - 1 +$$
$$+2y_{a_2} - 1 +$$
$$+1 - 2y(H_{c_3} \setminus \delta(S)) +$$
$$+1$$

Required arcs as well as whole subsets H_c are allowed in $\delta(S)$.



$$
\begin{aligned}
f(x) \geq\ & 2\Big(1 - 2y(H_c \setminus (M_0 : M_4))\Big) + \\
& +2\Big(1 - 2y(H_{c'} \setminus (M_0 : M_4))\Big) + \\
& +2 - 2y(H_{c_1} \setminus A(M_1)) + \\
& +2 - 2y(H_{c_2} \setminus A(M_2)) + \\
& +2 - 2y(H_{c_3} \setminus A(M_3)) +
\end{aligned}
$$

Required arcs outside subsets M_i $i = 1, ..., K$ are allowed.

Dominance inequalities

Dominance inequalities (Ha et al., 2013):

(based on those proposed by Gendreau, Laporte and Semet, 1997, for the Covering Tour Problem)

Let $a_1, a_2 \in A_R$ such that

$$\Big\{H \in \mathbb{H} \; : \; a_1 \in H\Big\} \subseteq \Big\{H \in \mathbb{H} \; : \; a_2 \in H\Big\}.$$

Then $y_{a_1} + y_{a_2} \leq 1$.

Dominance inequalities are not valid inequalities for the GDRPP.

A branch-and-cut algorithm

Initial LP

- Symmetry equations (1): $x(\delta^+(i)) = x(\delta^-(i)), \ \forall i \in V$
- One connectivity constraint (2b) $x(\delta^-(H_c)) \geq 1, \quad$ for each $H_c \in \mathbb{H}$
- Traversing inequalities (3): $x_{ij} \geq y_{ij}, \forall (i,j) \in A_R$
- Service inequalities (4): $y(H_c) \geq 1, \quad$ for each $H_c \in \mathbb{H}$
- Dominance inequalities $y_{a_1} + y_{a_2} \leq 1$
- Trivial inequalities $x_{ij} \geq 0, \ 0 \leq y_{ij} \leq 1$

A branch-and-cut algorithm

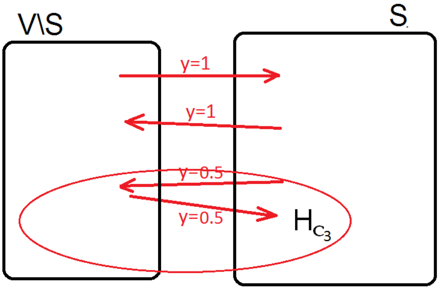
Separation algorithms for connectivity inequalities :

Let (y^*, x^*) be the fractional solution at any iteration of the cutting-plane procedure.

- Heuristic algorithm: Check the connected components induced in G by the arcs a with $y^*_a > \varepsilon$ (ε = 0, 0.25, 0.5, 0.75)
- Exact algorithm
- “Reduced” exact algorithm

Separation algorithms for parity inequalities:

Let (y^*, x^*) be the fractional solution at any iteration of the cutting-plane procedure.
Check the connected components $G(S)$ induced by the arcs a with $y^*_a > \varepsilon$ such that $y^*(\delta(S))$ is odd



$$x(\delta(S)) \geq 2y_{a_1} - 1 +$$
$$+2y_{a_2} - 1 +$$
$$+1 - 2y(H_{C_3} \setminus \delta(S)) +$$
$$+1$$

A branch-and-cut algorithm

We use separation algorithms for K-C inequalities:

Let (y^*, x^*) be the fractional solution at any iteration of the cutting-plane procedure.

- Heuristic algorithm based on the one for K-C constraints in the General Routing Problem in C., Letchford & Sanchis (2000).

Heuristic to obtain feasible solutions:

Let (y^*, x^*) be the fractional solution at any iteration of the cutting-plane procedure.

• First a subset $A' \subseteq A_R$ of arcs with a large value of y^*_a and forming a feasible solution of the Set Covering Problem is selected.

• Then, a DGRP instance in graph G with required arcs A' and the depot as a required vertex is solved.

A branch-and-cut algorithm

- Cplex 12.4 with zero-half cuts.
- Time limit: 2 hours.
- Run on a PC Intel Core i7 CPU 3.40 GHz, 16 GB RAM.

5×8 GDRPP instances from Ha et al. (2013)

- $|V|$ vertices randomly generated in a unit square.
- $|A|$ arcs randomly generated trying to imitate real networks.
- $|A| \times t$ customers randomly positioned in the square, where $t = 0.5, 1, 5, 10$.
- $|V| = 500$ and $|A| = 1500$ (for $r = 150$) or $|V| = 500$ and $|A| = 1000$ (for $r = 200$)

Computational results on GDARP instances

Table: Comparison with Ha et al. results

	$\|V\|$	$\|A\|$	$\|\mathbb{H}\|$	Ha et al.[1] #of opt	Ha et al.[1] Time	Our Results #of opt	Our Results Time	Gap0 impr.
ce200-0.5	500	1000	500	3	4155,6	5	245,7	2,54
ce200-1	500	1000	1000	4	2447,8	5	88,6	2,11
ce200-5	500	1000	5000	5	315,1	5	28,3	0,87
ce200-10	500	1000	10000	5	82,3	5	20,3	0,79
ce150-0.5	500	1500	750	0	7202,9	5	830,5	2,04
ce150-1	500	1500	1500	2	4499,9	5	1235,5	1,43
ce150-5	500	1500	7500	5	154,5	5	49,2	0,47
ce150-10	500	1500	15000	5	205,9	5	50,1	0,26

(1) CPU at 2.4GHz with 6GB RAM.

Table: Cuts added

	$\lvert V \rvert$	$\lvert A \rvert$	$\lvert \mathbb{H} \rvert$	Conn.	Parity	K-C	Z-H
ce200-0.5	500	1000	500	9892,0	1311,6	2,8	175,0
ce200-1	500	1000	1000	4985,0	997,8	24,0	165,6
ce200-5	500	1000	5000	1093,0	668,0	6,8	121,8
ce200-10	500	1000	10000	429,8	583,4	3,0	124,8
ce150-0.5	500	1500	750	8118,2	2405,2	39,4	413,4
ce150-1	500	1500	1500	4871,8	1897,0	23,2	371,8
ce150-5	500	1500	7500	403,4	745,8	6,2	166,6
ce150-10	500	1500	15000	412,2	734,6	3,6	173,6

Computational results on mixed GDARP instances

5×8 GDRPP instances proposed by Hà et al. (2013) from two Mixed RPP instances:

- MB537 with $|V| = 500$, $|E| = 364$, $|A| = 476$
- MB547 with $|V| = 500$, $|E| = 351$, $|A| = 681$
- Now, r is defined as the average cost of all the arcs.

Computational results on mixed GDARP instances

Table: Comparison with Ha et al. results (mixed graph instances)

	$\|V\|$	$\|A\|$	$\|\mathbb{H}\|$	Ha et al.[1] #of opt	Ha et al.[1] Time	Our Results #of opt	Our Results Time	Gap0 impr.
MB0537-0.5	500	1204	400	0	7200,7	5	456,3	0,16
MB0537-1	500	1204	800	0	7201,5	5	368,5	0,20
MB0537-5	500	1204	4000	3	3937,8	5	223,8	0,19
MB0537-10	500	1204	8000	5	2418,3	5	233,4	0,18
MB0547-0.5	500	1383	520	0	7201,1	4	2854,3	1,17
MB0547-1	500	1383	1040	0	7202,2	5	1515,2	0,78
MB0547-5	500	1383	5200	4	1639,9	5	232,1	0,19
MB0547-10	500	1383	10400	5	756,2	5	131,5	0,18

(1) CPU at 2.4GHz with 6GB RAM.

Computational results on undirected GDARP instances

12×3 instances generated from RPP instances:

- UR500 with $298 \leqslant |V| \leqslant 499$, $597 \leqslant |A| \leqslant 1526$ and $1 \leqslant |\mathbb{H}| \leqslant 99$.
- UR750 with $452 \leqslant |V| \leqslant 749$, $915 \leqslant |A| \leqslant 2314$ and $1 \leqslant |\mathbb{H}| \leqslant 140$.
- UR1000 with $605 \leqslant |V| \leqslant 1000$, $2289 \leqslant |A| \leqslant 3083$ and $1 \leqslant |\mathbb{H}| \leqslant 204$.
- Each R-connected component of the original graph defines a "customer" (in this way, the H_c are connected and disjoint subsets, GARP instances).

Computational results (undirected graph instances)

	$\lvert V \rvert$	$\lvert A \rvert$	$\lvert \mathbb{H} \rvert$	Gap0 (%)	Gap (%)	Solved	Nodes	Time (s)
UR500	446.0	2257.8	35.3	0.24	0.00	12/12	320.2	790.0
UR750	665.7	3396.8	55.7	0.32	1.18	10/12	1590.0	3247.0
UR1000	882.2	4580.8	74.8	3.58	4.34	2/12	1262.0	6016.3

Outline

1. Introduction
2. Polyhedral Combinatorics applied to some routing problems
   - The Traveling Salesman Problem (TSP)
   - The Stacker Crane Problem (SCP)
   - The Orienteering Arc Routing Problem (OARP)
   - The Generalized Directed Arc Routing Problem (Close Enough ARP)
3. Polyhedral Combinatorics (some theory)

Polyhedral Combinatorics

• A polyhedron P is a set of the form $\{x \in R^n : Ax \leq b\}$

• A polyhedron P is of dimension k, $dim(P) = k$, if the maximum number of affinely independent points in P is $k+1$. It is full-dimensional if $dim(P) = n$.

• Let $A^{=}\{i : a^i x = b_i, \forall x \in P\}$ and $(A^{=}, b^{=})$ the associated rows of (A, b). Then $dim(P) = n - rank((A^{=}, b^{=})$.

Polyhedral Combinatorics

• Vectors $x^1, \ldots, x^k \in R^n$ are linearly independent if the unique solution to $\sum_i \alpha_i x^i = 0$ is $\alpha_i = 0, \forall i$.

• Vectors $x^1, \ldots, x^k \in R^n$ are affinely independent if the unique solution to $\sum_i \alpha_i x^i = 0$ and $\sum_i \alpha_i = 0$ is $\alpha_i = 0, \forall i$.

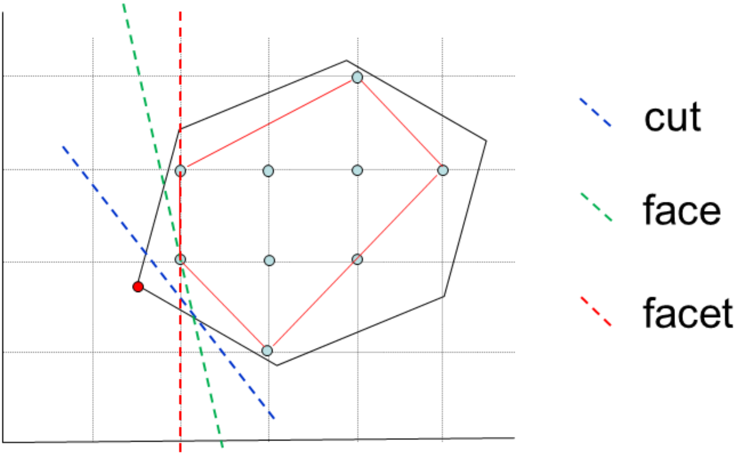
• Linear independence implies affine independence but not viceversa.

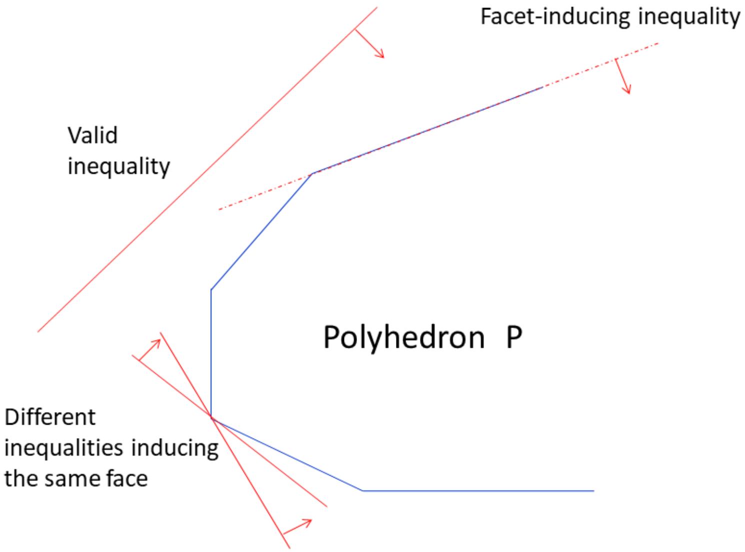
• Vectors $x^1, \ldots, x^k \in R^n$ are affinely independent if and only if $x^2 - x^1, \ldots, x^k - x^1$ are linearly independent.

• If $0 \notin aff(P)$, i.e., if P is contained in a hyperplane $\{x \in R^n : ax = a_0\}$, with $a_0 \neq 0$, then $dim(P)$ is the maximum number of linearly independent points in P minus 1 (linear independence and affine independence are in this case equivalent).

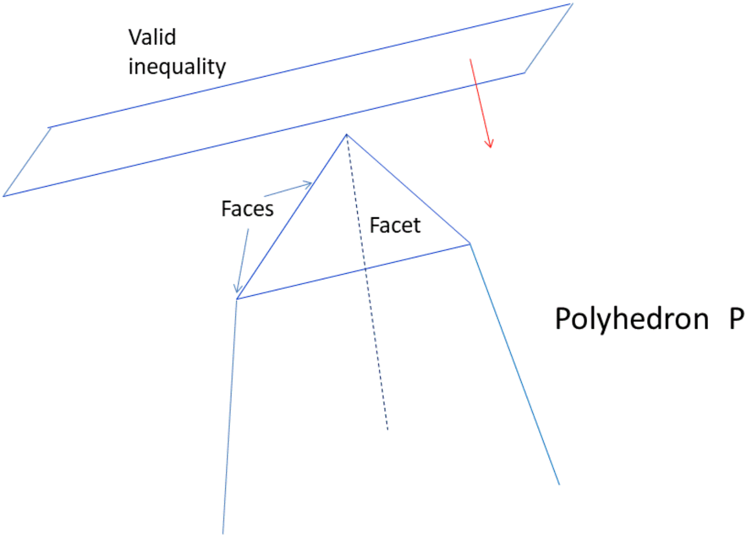
- The inequality (π, π_0) is a valid inequality for P if $\pi x \leq \pi_0, \forall x \in P$.
- If (π, π_0) is a valid inequality for P, $F = \{x \in P : \pi x = \pi_0\}$ is called a face of P.
- A face is said to be a proper face if $F \neq \emptyset$ and $F \neq P$.
- A face F is said to be a facet of P if $dim(F) = dim(P) - 1$.
- Facets are all we need to describe polyhedra.

Cutting planes




Facet-inducing inequality
Valid
inequality
Polyhedron P
Different
inequalities inducing
the same face

Polyhedral Combinatorics



- There are two basic methods to prove that a given inequality $\pi x \leq \pi_0$ is facet-defining of P.
- In both cases, one first has to check that $\pi x \leq \pi_0$ is valid and that P is not contained in $\{x : \pi x = \pi_0\}$.
- The 1st (direct) method consists of finding $k = dim(P)$ affinely independent vectors $x^1, \ldots, x^k$ satisfying $\pi x^i = \pi_0, \forall i$.
- The 2nd (indirect) method consists of assuming the existence of a (stronger) valid inequality $dx \leq d_0$ with $\{x \in P : \pi x = \pi_0\} \subseteq \{x \in P : dx = d_0\}$ and prove they both are equivalent.

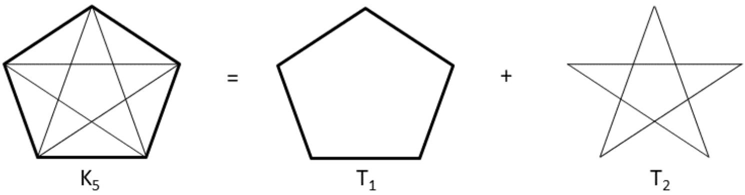
TSP polyhedron

• Let $K_n = (V, E)$ be the complete graph on n vertices and let $TSP(K_n)$ be the convex hull of all the TSP tours.

• $TSP(K_n)$ is a polytope (a bounded polyhedron).

• Lemma: If $|V| = 2k + 1$, there are k edge-disjoint tours $T_1, \dots, T_k$ such that $E = \cup T_i$. If $|V| = 2k$, there are $k - 1$ edge-disjoint tours $T_1, \dots, T_{k-1}$ and an edge-disjoint matching M such that $E = M \cup (\cup T_i)$.

Consider, for instance, $|V| = 5$:

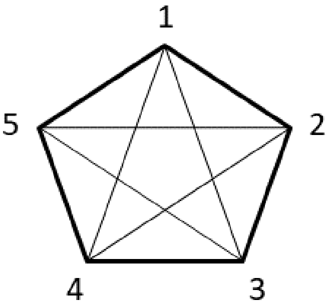


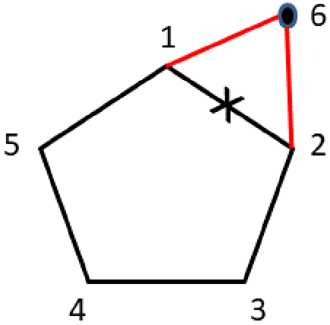
TSP polyhedron

Theorem: $dim(TSP(K_n)) = |E| - |V|$.

Assume $n = 6$. We have to prove that $dim(TSP(K_6))$ = 15 - 6 = 9.

The subgraph K_5 induced by the 5 first vertices of G is the union of 2 tours T_1 and T_2 of length 5 (= n-1).





TSP polyhedron

Add

Remove

	(1,2)	(2,3)	(3,4)	(4,5)	(1,5)	(1,3)	(1,4)	(2,4)	(2,5)	(3,5)	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)
	1	1	1	1	1						1	1			
	1	1	1	1	1							1	1		
T_1	1	1	1	1	1								1	1	
	1	1	1	1	1									1	1
	1	1	1	1	1						1				1
						1	1	1	1	1	1		1		
						1	1	1	1	1	1			1	
T_2						1	1	1	1	1		1		1	
						1	1	1	1	1		1			1
						1	1	1	1	1			1		1

TSP polyhedron

(1,2)	(2,3)	(3,4)	(4,5)	(1,5)	(1,3)	(1,4)	(2,4)	(2,5)	(3,5)	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)
	1	1	1	1						1	1			
1		1	1	1							1	1		
1	1		1	1								1	1	
1	1	1		1									1	1
1	1	1	1							1				1
						1	1	1	1	1		1		
					1		1	1	1	1			1	
					1	1		1	1		1		1	
					1	1	1		1		1			1
					1	1	1	1				1		1

Matrix $\mathbf{1} - \mathbf{I}$ is non-singular. Therefore, the whole matrix is also non-singular and the 10 rows (tours) are linearly (and affinely) independents, so $dim(TSP(K_6))$=10-1=9.

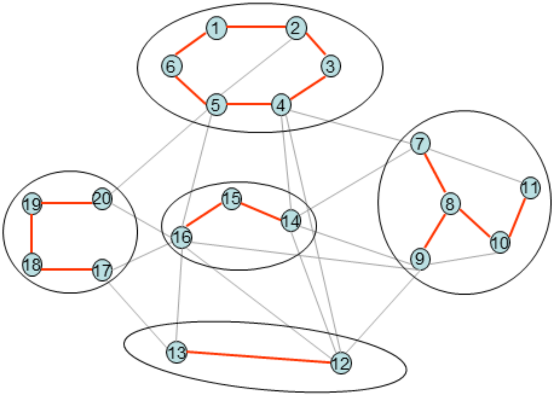
RPP polyhedron

The Rural Postman Problem (RPP) is a generalization of the Chinese Postman Problem that consists of, given $G = (V, E)$ and $E_R \subseteq E$, finding a minimum length tour traversing at least once every edge in E_R (required edges).

Usually $G_R = (V, E_R)$ is non connected. Let $V_1, \ldots, V_p$ the sets of vertices of its connected components (R-sets).

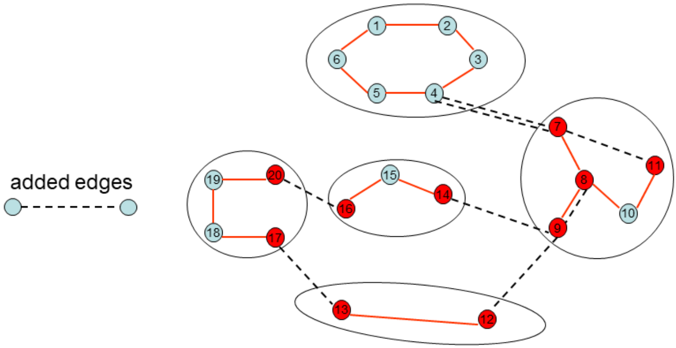
Proposed by Orloff (1974). It is NP-hard (Lenstra & Rinnooy Kan, 1976)

RPP polyhedron



Equivalent augmentation problem: Add to G_R a set of edges with total minimum cost such that the resulting graph is connected and even.

RPP polyhedron



Tour for the RPP: a connected and even graph

A tour for the RPP minus E_R is a semitour for the RPP

RPP polyhedron

We define x_e as the number of copies of e to be added to G_R to obtain a connected and even graph.

Then, the RPP can be formulated as follows:

$$\text{Minimize} \quad \sum_{e \in E} c_e x_e$$

$$\begin{aligned} x(\delta(S)) &\geq 2, \quad \forall S \subset V : \delta_R(S) = \emptyset && (5) \\ x(\delta(i)) &\equiv |\delta_R(i)|, \quad \forall i \in V && (6) \\ x_e &\geq 0, \quad \forall e \in E && (7) \\ x_e & \quad integer, \quad \forall e \in E && (8) \end{aligned}$$

RPP polyhedron

Let $RPP(G)$ be the convex hull of all the semitours for the RPP in $G = (V, E)$. $RPP(G)$ is a polyhedron.

Theorem: $dim(RPP(G)) = |E|$ iff G is connected.

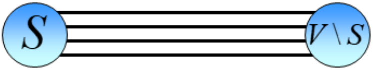
If G is connected, there is at least a tour (and therefore a semitour) for the RPP: x.

From x, we construct $|E|$ different semitours as follows:

For each edge e, consider the vector $x + 2z_e$, where $z_e \in R^{|E|}$ is the unit vector with a 1 in position e.

Obviously, these $1 + |E|$ vectors are affinely independent and, hence, $RPP(G)$ is full-dimensional.

RPP polyhedron



$$x(\delta(S)) \geq 2$$

Theorem: Connectivity inequalities define facets of $RPP(G)$ iff $G(S)$ and $G(V \setminus S)$ are connected.

If $G(S)$ and $G(V \setminus S)$ are connected, it is possible to build an RPP tour x in $G(S)$ and another tour x' in $G(V \setminus S)$ such that jointly traverse at least once all the required edges.

Assume $\delta(S) = \{e_1, \ldots, e_k\}$. Given e_1, we construct a tour y^1 by adding two copies of e_1 to $x + x'$. Repeat this for $e_2, \ldots, e_k$ to obtain k tours for the RPP, $y^1, \ldots, y^k$, all of them satisfying $y(\delta(S)) = 2$.

By substracting x^R to $y^1, \ldots, y^k$, we obtain k semitours $x^1, \ldots, x^k$, all of them satisfying $x(\delta(S)) = 2$. From x^k , for example, we construct $|E| - k$ more vectors $x^k + 2z_e$, $\forall e \in E \setminus \delta(S)$, which are semitours satisfying $x(\delta(S)) = 2$.

RPP polyhedron

The incidence matrix of these $k + |E| - k$ semitours is non-singular and, therefore, the $|E|$ semitours are linearly (and affinely) independent.

e_1	e_2			e_k	e_{k+1}	e_{k+2}			$e_{\|E\|}$
2	0			0	0	0			0
0	2			0	0	0			0
0	0	2		0	0	0			0
0	0		2	0	0	0			0
0	0			2	0	0			0
0	0			2	2	0			0
0	0			2	0	2			0
0	0			2	0	0	2		0
0	0			2	0	0		2	0
0	0			2	0	0			2

And this is, if not all, enough for today.

Thanks for your attention !