Introduction to Column Generation and hybrid methods for Homecare Routing

Louis-Martin Rousseau

Canada Research Chair in
Healthcare Analytics and Logistics

Plan

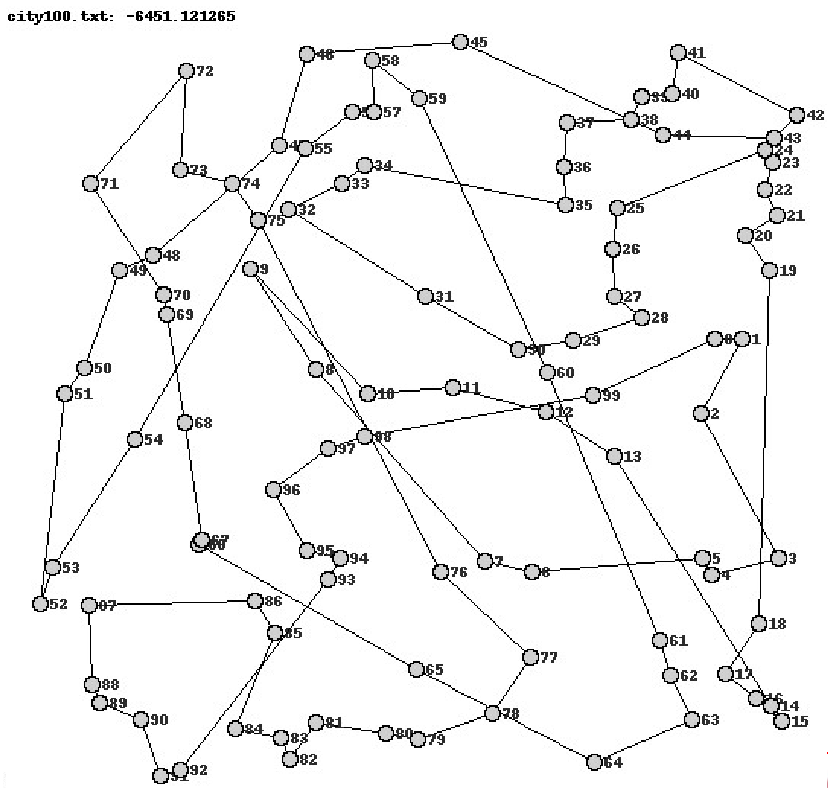
- Génération de coupes (lignes)
  - Illustration sur le problème du voyageur de commerce

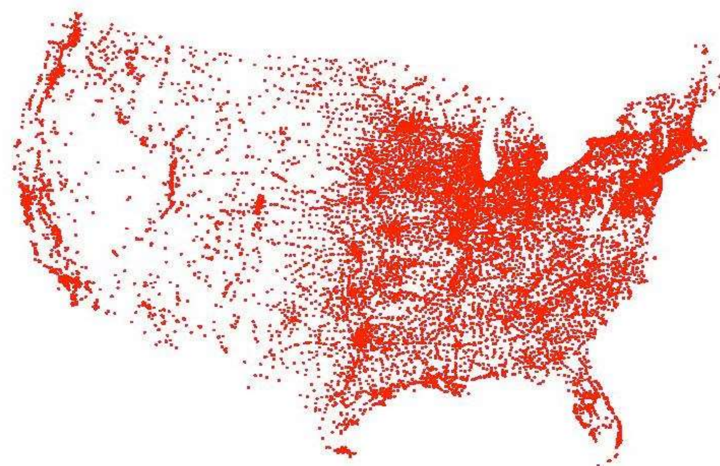
- Génération de variables (colonnes)
  - Illustration sur un problème de tournée de véhicule

Le problème du voyageur de commerce

C'est le problème le plus connu en recherche opérationnelle, celui qui a reçu le plus d'attention et probablement le plus prestigieux;

Étant donné un nombre de points (villes) à visiter et une matrice donnant la distance entre chacune d'elles, donner la tournée qui visite toutes les villes en parcourant la plus petite distance.





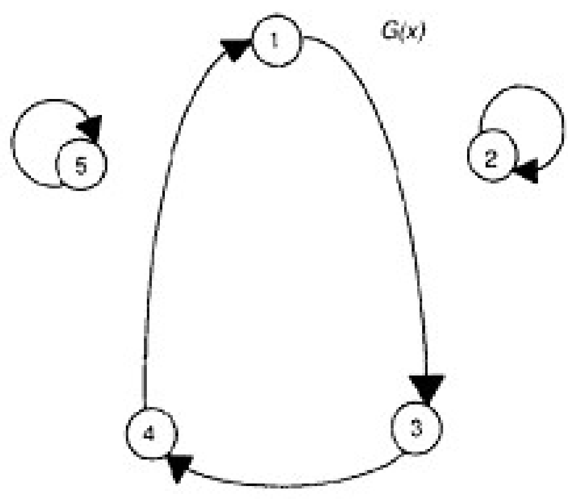
TSP: la structure

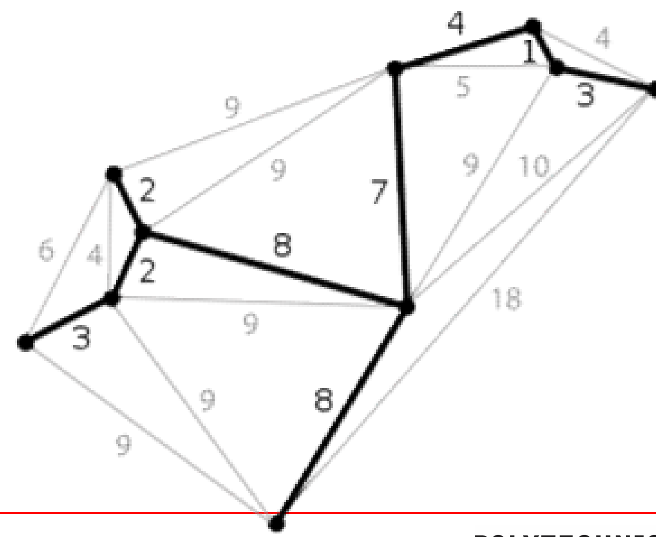
Le problème du voyageur de commerce (ou Traveling Salesman Problem) est un problème combinatoire NP-difficile

- C'est-à-dire qu'il n'existe pas de solution dont le temps de calcul est une fonction polynomiale du nombre de points à visiter.

Le TSP combine deux structures qui elles sont «faciles».

- Le problème d'affectation (le degré de chaque noeud = 2)
- Le problème d'arbre de recouvrement minimum (connectivité)





TSP: Modèle mathématique

Soit:

- x_{ij} une variable binaire qui indique si la route passe directement du point i au point j

  (1 si oui, 0 sinon)
- C_{ij} le coût d'aller directement de i à j.
- S un sous-ensemble non vide des points à visiter.

Les deux modèles suivants sont corrects et équivalents.

$$
\begin{aligned}
\min \sum_{i,j \in N} C_{ij} x_{ij} & \\
\text{sujet à} \sum_{j \in N} x_{ij} &= 1 \;\; \forall i \in N \\
\sum_{i \in N} x_{ij} &= 1 \;\; \forall j \in N \\
\sum_{i \in S, j \notin S} x_{ij} &\geq 1 \;\; \forall S \subset N \\
x_{ij} &\in \{0,1\}, \;\; \forall i,j \in N
\end{aligned}
$$

Contraintes de degré

Contraintes de connectivité

$$
\begin{aligned}
\min \sum_{i,j \in N} C_{ij} x_{ij} & \\
\text{sujet à} \sum_{j \in N} x_{ij} &= 1 \;\; \forall i \in N \\
\sum_{i \in N} x_{ij} &= 1 \;\; \forall j \in N \\
\sum_{i,j \in S} x_{ij} &\leq |S| - 1 \;\; \forall S \subset N \\
x_{ij} &\in \{0,1\}, \;\; \forall i,j \in N
\end{aligned}
$$

TSP: les difficultés

Le problème avec les formulations précédentes est le nombre de contraintes de connectivité.

- Il y en a une pour chaque sous-ensemble possible des points à visiter...
- donc environ $2^{|N|}$

Par contre, celles-ci ne sont peut-être pas toutes utiles. On peut donc les ignorer pour commencer et les ajouter par la suite.

C'est ce qu'on appelle une approche par plans coupants (Cutting Plane Method)

1. On résout d'abord le problème sans ces contraintes
2. On vérifie la solution
   - si celle-ci satisfait toutes les contraintes ignorées, alors elle est optimale. ON ARRÊTE.
   - sinon, on ajoute les contraintes qui sont violées (c'est ce qu'on appelle la séparation)

- ON RETOURNE à 1

An example

Vehicle routing problem

Vehicle routing problem

Customers

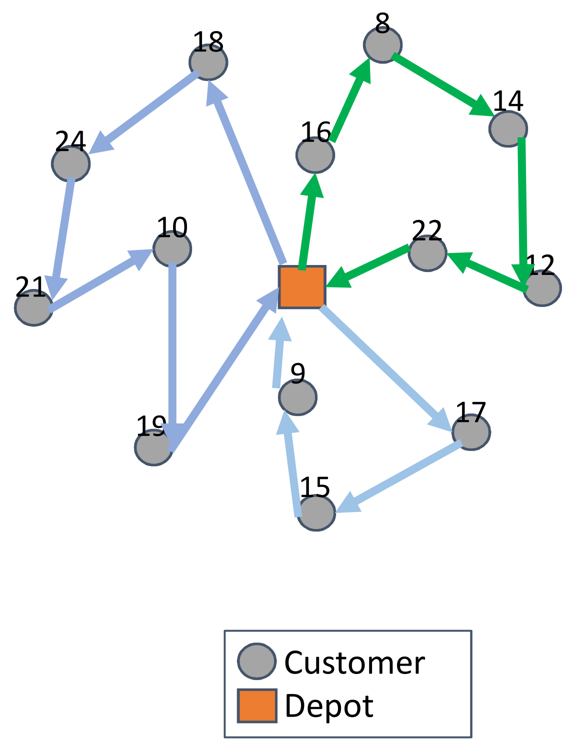
- Demand constraints

Vehicles

- Capacity constraints
- Flow conservation constraints

Objective:

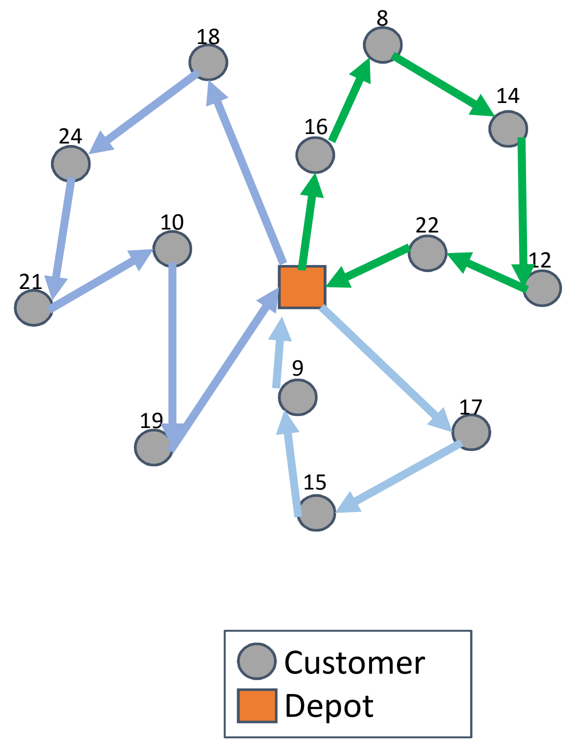
- Find routes that minimize total distance



Vehicle routing problem

Standard mip formulation:

- Scaling issues
- Symmetry
- More complex constraints add even more complexity
- Some constraints can lead to bad linear relaxations.



Enumerate all possible routes

- Much simpler formulation
- Vehicle constraints are implicitly considered in route enumeration
- Better Linear Relaxation

Vehicle routing problem

Enumerate all possible routes

$$\text{Minimize} \quad \sum_{p \in \Omega} c_p \theta_p$$

$$\text{subject to:} \quad \sum_{p \in \Omega} v_{ip} \theta_p = 1 \qquad \forall i \in \mathrm{N}$$

$$\theta_p \in \{0,1\} \qquad \forall p \in \Omega$$

Vehicle routing problem

Enumerate all possible routes

$$
\begin{aligned}
\text{Minimize} \quad & \sum_{p \in \Omega} c_p \theta_p \\
\text{subject to:} \quad & \sum_{p \in \Omega} v_{ip} \theta_p = 1 && \forall i \in \boxed{\mathrm{N}} \\
& \theta_p \in \{0,1\} && \forall p \in \boxed{\Omega}
\end{aligned}
$$

Set of customers

Set of routes

Vehicle routing problem

Enumerate all possible routes

Minimize $$\sum_{p \in \Omega} c_p \theta_p$$

subject to: $$\sum_{p \in \Omega} v_{ip} \theta_p = 1 \qquad \forall i \in \mathrm{N}$$

$$\boxed{\theta_p} \in \{0,1\} \qquad \forall p \in \Omega$$

$$= \begin{cases} \mathbf{0}, & \boldsymbol{if\ route\ p\ is\ used} \\ \mathbf{1}, & \boldsymbol{otherwise} \end{cases}$$

Vehicle routing problem

Enumerate all possible routes

Cost of route p

$$\text{Minimize} \quad \sum_{p\in\Omega} \boxed{c_p}\theta_p$$

$$= \begin{cases} \mathbf{1}, & \textbf{\textit{if route p visits customer i}} \\ \mathbf{0}, & \textbf{\textit{otherwise}} \end{cases}$$

$$\text{subject to:} \quad \sum_{p\in\Omega} \boxed{v_{ip}}\theta_p = 1 \qquad \forall i \in \mathrm{N}$$

$$\boxed{\theta_p} \in \{0,1\} \qquad \forall p \in \Omega$$

$$= \begin{cases} \mathbf{1}, & \textbf{\textit{if route p is used}} \\ \mathbf{0}, & \textbf{\textit{otherwise}} \end{cases}$$

Vehicle routing problem

Enumerate all possible routes

Cost of route p

Minimize $\sum_{p\in\Omega} \boxed{c_p}\theta_p$

$= \begin{cases} 1, & \textit{if route p visits customer i} \\ 0, & \textit{otherwise} \end{cases}$

subject to: $\sum_{p\in\Omega} \boxed{v_{ip}}\theta_p = 1 \qquad \forall i \in N$

$\boxed{\theta_p} \in \{0,1\} \qquad \forall p \in \Omega$

Possibly huge number of routes

$= \begin{cases} 1, & \textit{if route p is used} \\ 0, & \textit{otherwise} \end{cases}$

Vehicle routing problem

Enumerate all possible routes

Minimize $$\sum_{p \in \Omega} \boxed{c_p} x_p$$

Cost of route p

subject to: $$\sum_{p \in \Omega} \boxed{v_{ip}} x_p = 1 \qquad \forall i \in N$$

$$= \begin{cases} \mathbf{1}, & \textbf{\textit{if route p visits customer i}} \\ \mathbf{0}, & \textbf{\textit{otherwise}} \end{cases}$$

$$\boxed{x_p} \in \{0,1\} \qquad \forall p \in \Omega$$

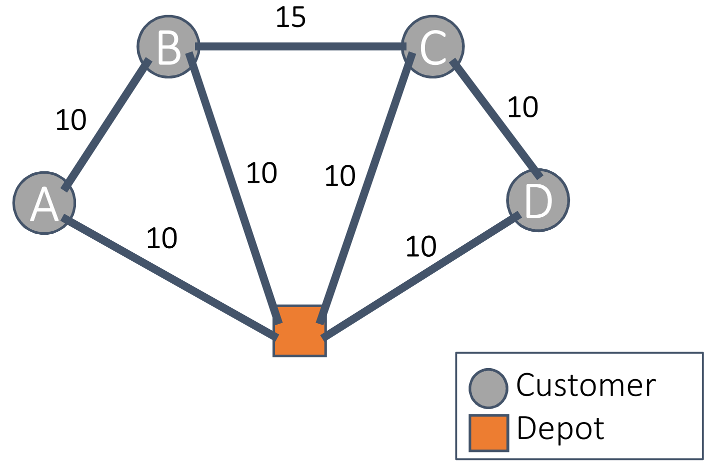
$$= \begin{cases} \mathbf{1}, & \textbf{\textit{if route p is used}} \\ \mathbf{0}, & \textbf{\textit{otherwise}} \end{cases}$$

Possibly huge number of routes

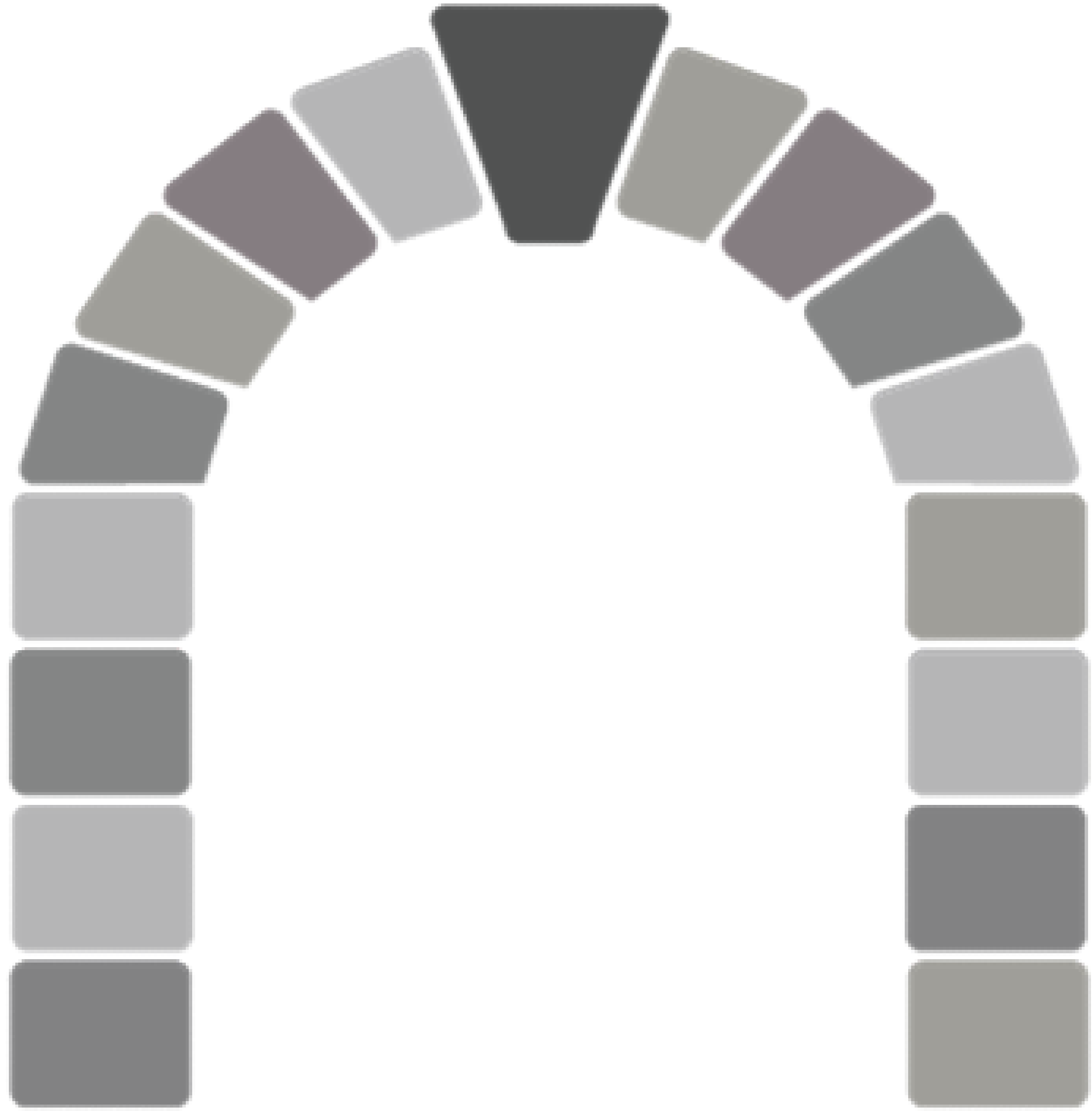
A very small number of routes are interesting

Vehicle routing problem

An example (max 2 clients)



$$
\begin{array}{lcccccccc}
\text{Min} & 20\,x_1 & +20\,x_2 & +20\,x_3 & +20\,x_4 & +30\,x_5 & +30\,x_6 & +35\,x_7 & \\
\text{A :} & x_1 & & & & +x_5 & & & = 1 \\
\text{B :} & & +x_2 & & & +x_5 & & +x_7 & = 1 \\
\text{C :} & & & +x_3 & & & +x_6 & +x_7 & = 1 \\
\text{D :} & & & & +x_4 & & +x_6 & & = 1
\end{array}
$$



An intuitive view of

Column Generation

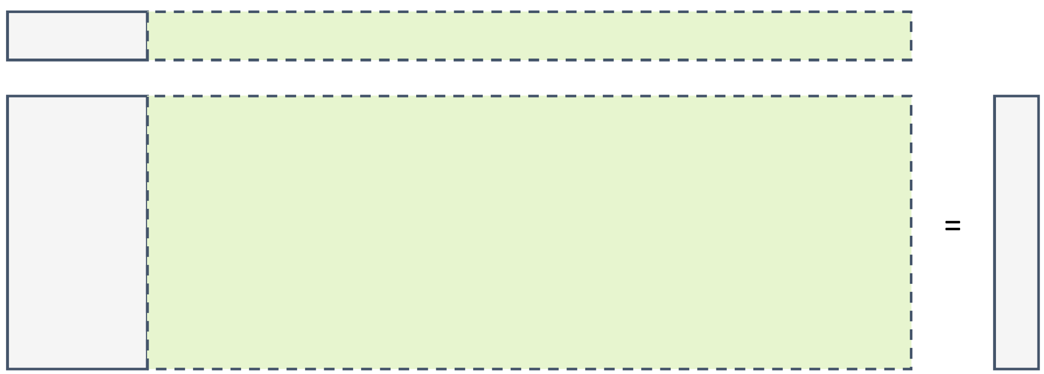
Column Generation

Solve linear programs with a lot of variables

Column Generation

Solve linear programs with a lot of variables

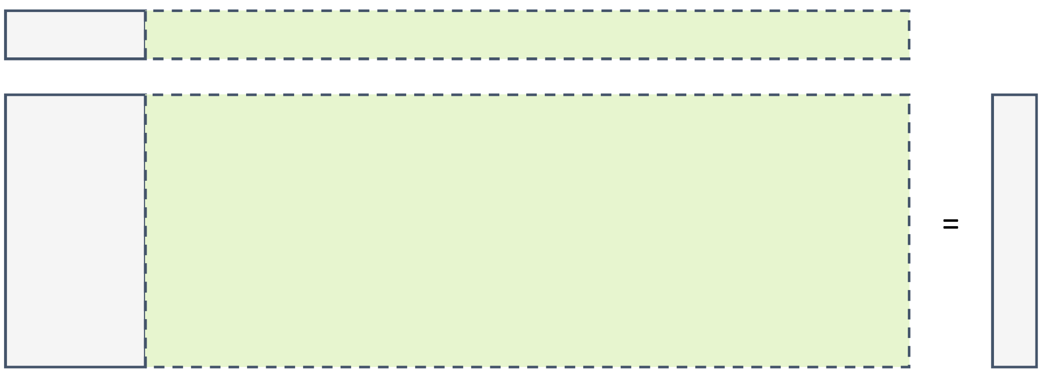
- Solve with a subset of variables



Column Generation

Solve linear programs with a lot of variables

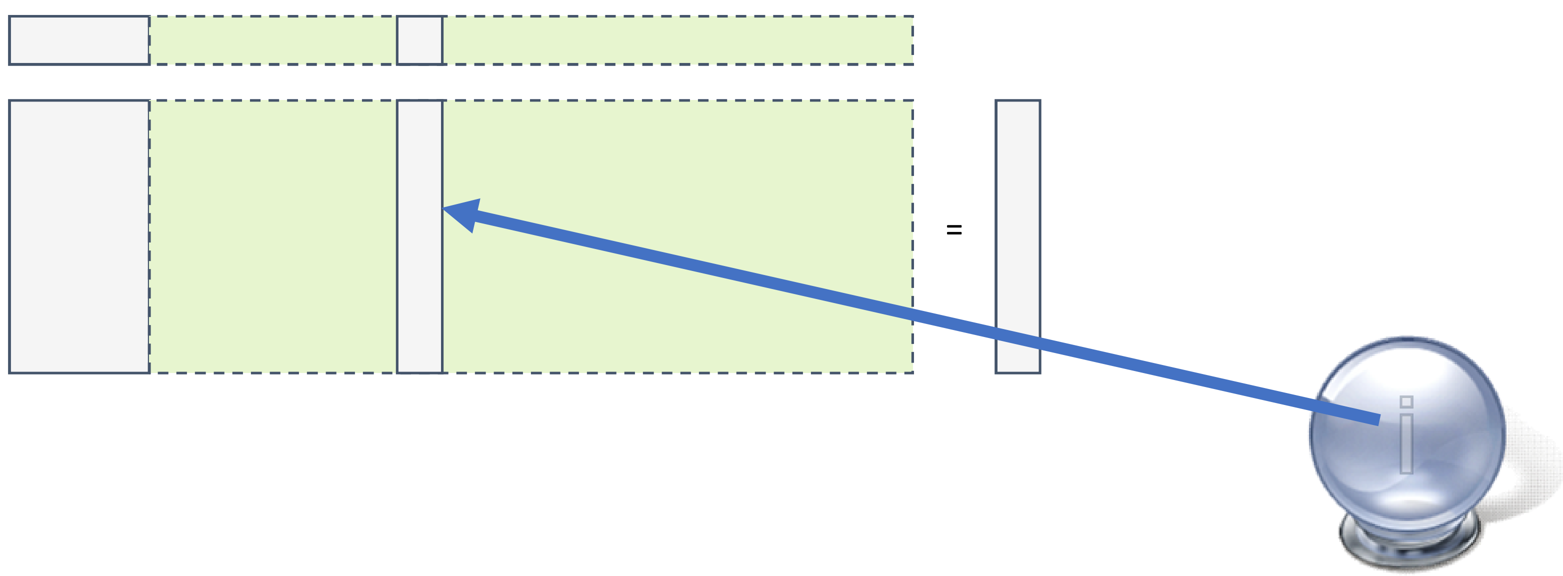
- Solve with a subset of variables



Column Generation

Solve linear programs with a lot of variables

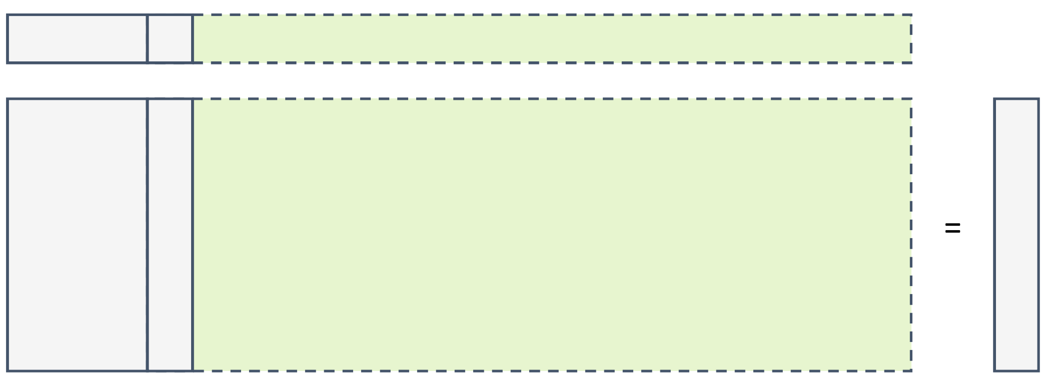
- Solve with a subset of variables



Column Generation

Solve linear programs with a lot of variables

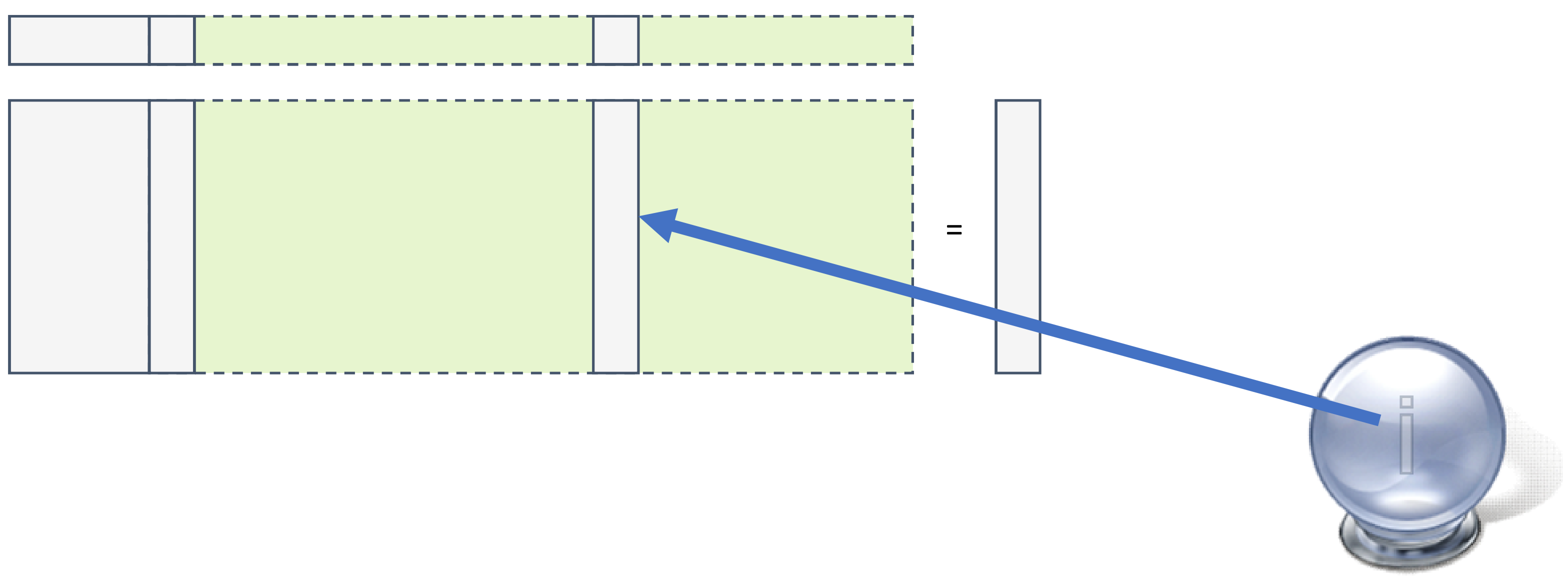
- Solve with a subset of variables



Column Generation

Solve linear programs with a lot of variables

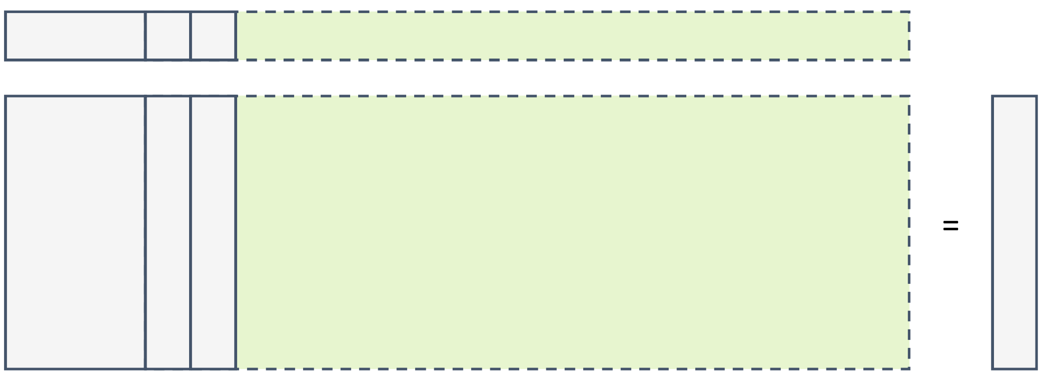
- Solve with a subset of variables



Column Generation

Solve linear programs with a lot of variables

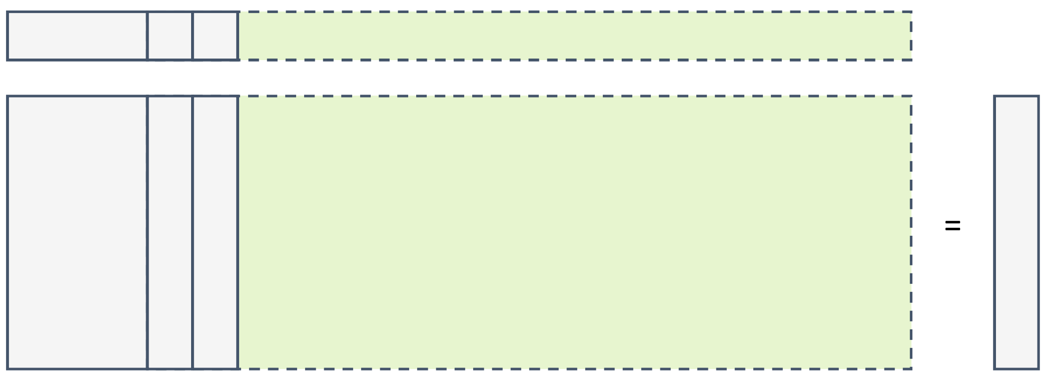
- Solve with a subset of variables



Column Generation

Solve linear programs with a lot of variables

- Solve with a subset of variables

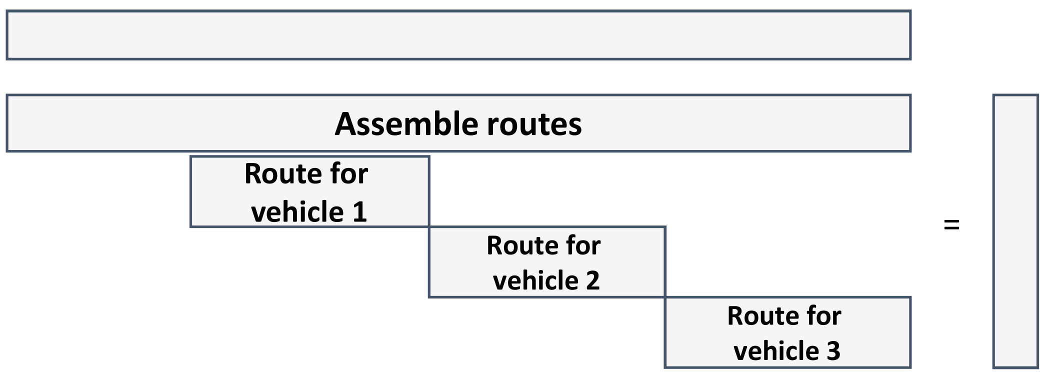


Column Generation

When to use column generation?

Column Generation

When to use column generation?

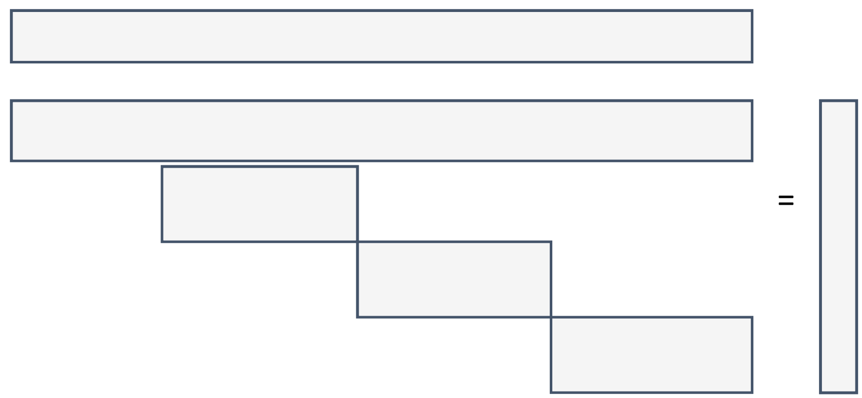


Column Generation

When to use column generation?

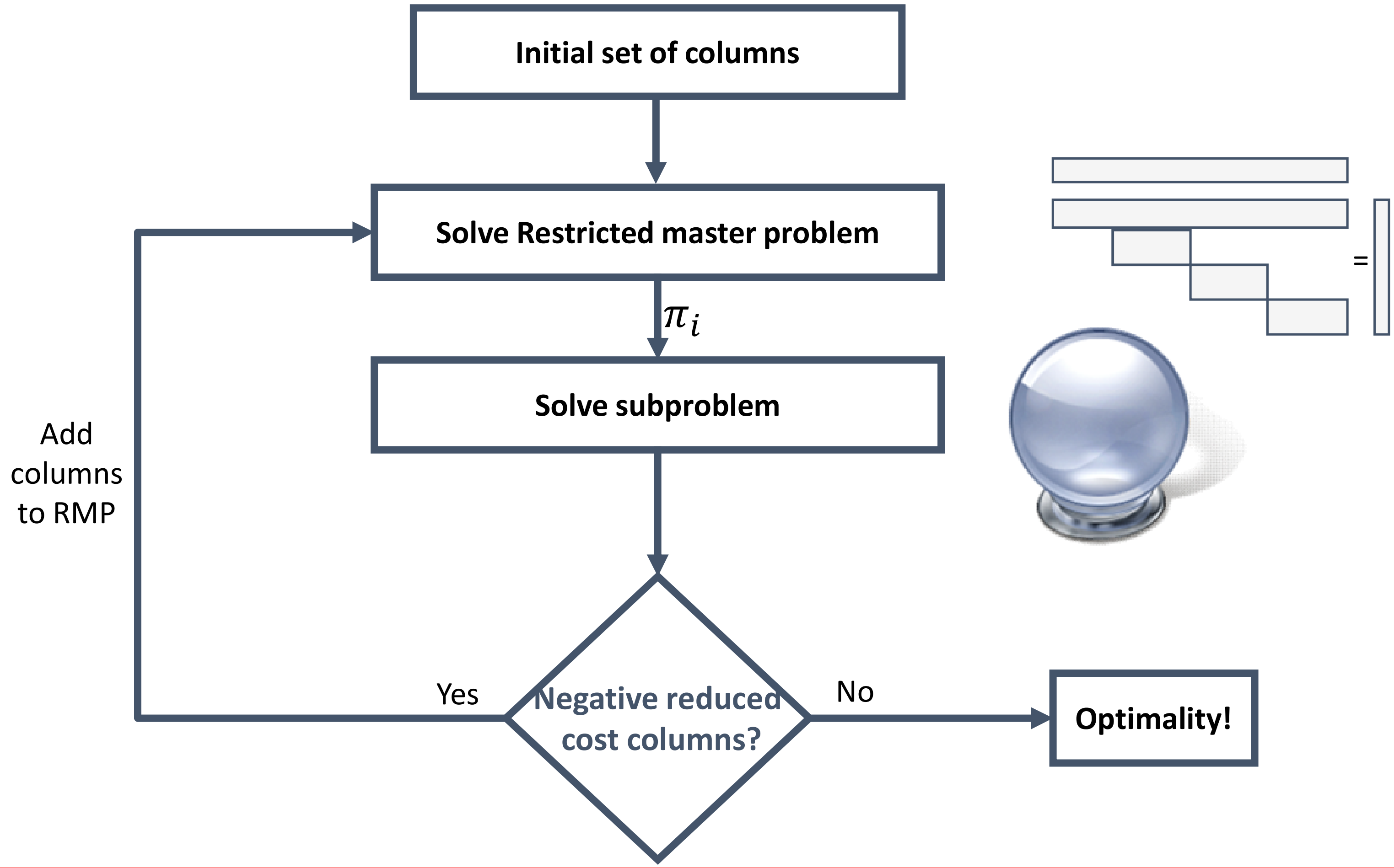
Works well generally on:

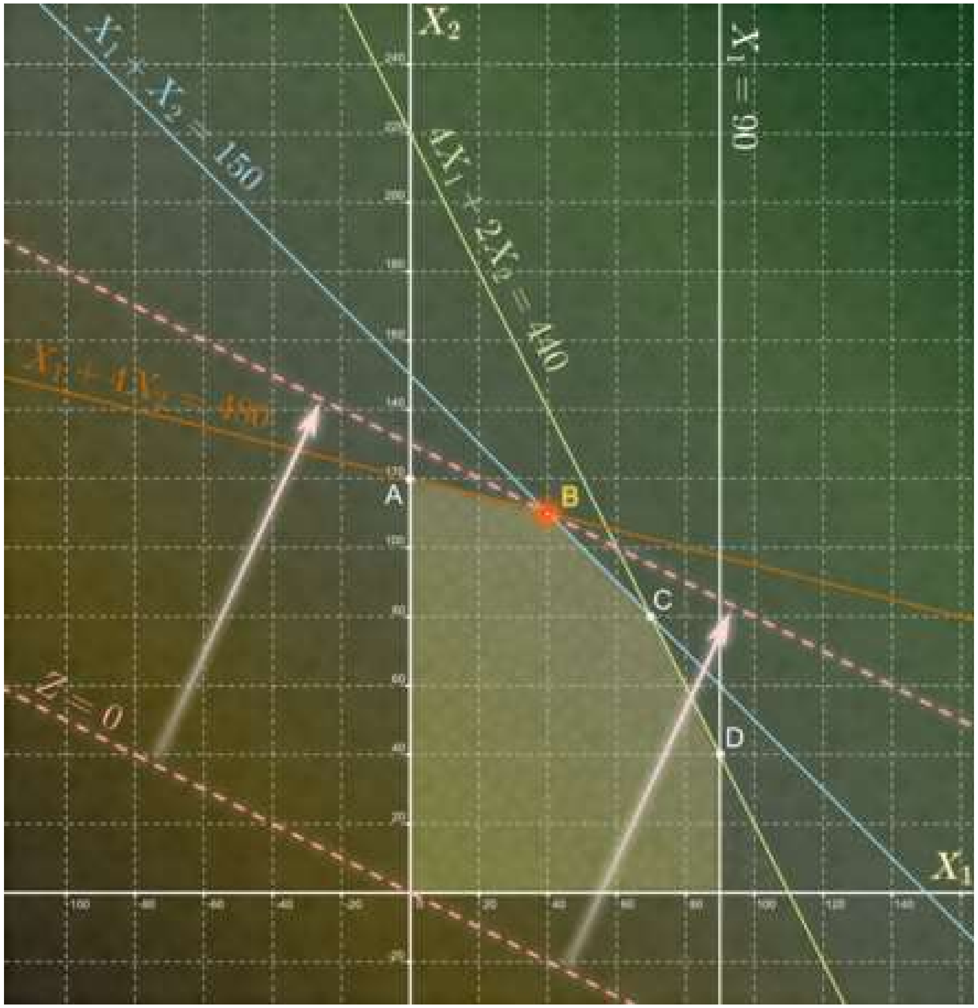
- Vehicle routing
- Airline Scheduling
- Shift Scheduling
- Jobshop Scheduling
- …



Worked the best when part of the problem has an underlying structure: Network, Hypergraph, knapsack, etc…

Column Generation



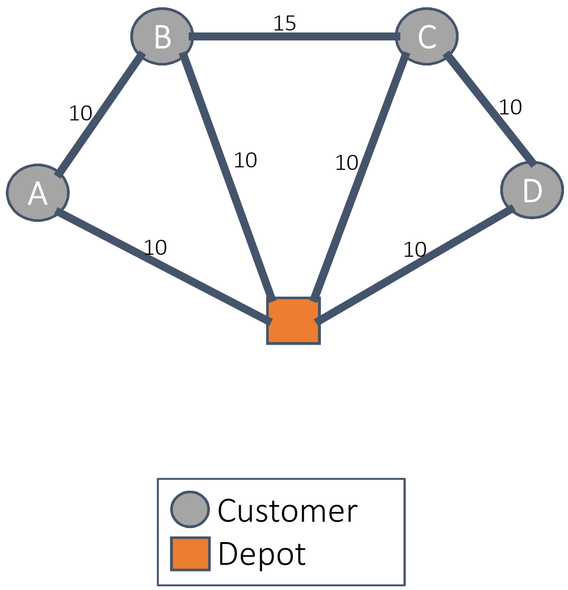


Master Probelm for the

Vehicle routing problem

Vehicle routing problem

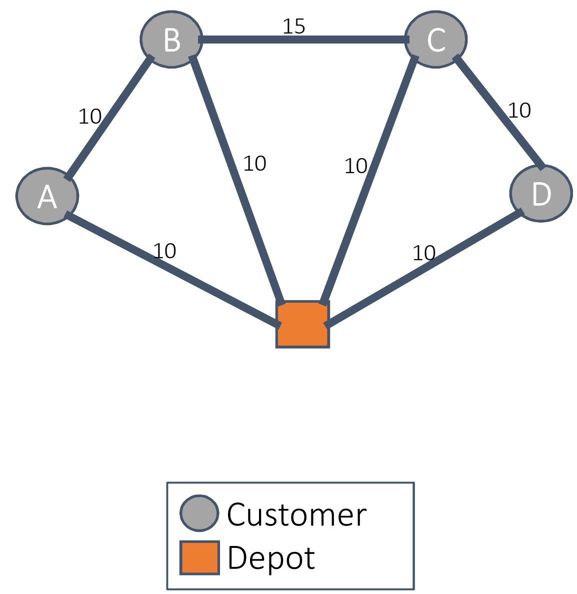
An example (max 2 clients)



Vehicle routing problem

An example (max 2 clients)

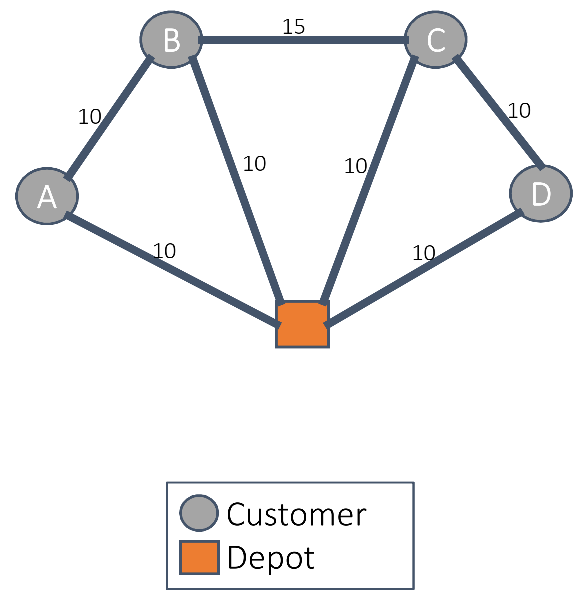
$$
\begin{array}{llllll}
\text{Min} & 20\,x_1 & +20\,x_2 & +20\,x_3 & +20\,x_4 & \\
\text{A :} & x_1 & & & & = 1 \\
\text{B :} & & x_2 & & & = 1 \\
\text{C :} & & & x_3 & & = 1 \\
\text{D :} & & & & x_4 & = 1
\end{array}
$$



Vehicle routing problem

An example (max 2 clients)

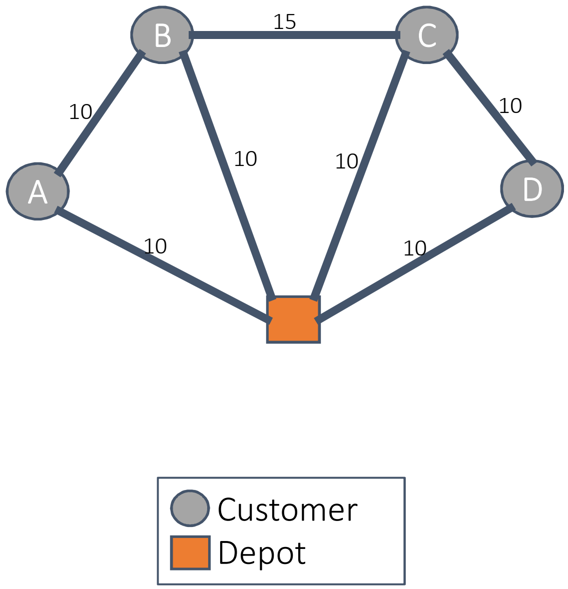
	x_1	x_2	x_3	x_4	
Min	20	20	20	20	
A :	1				= 1
B :		1			= 1
C :			1		= 1
D :				1	= 1



Vehicle routing problem

An example (max 2 clients)

	x_1	x_2	x_3	x_4		
$\hat{c}$	0	0	0	0		π_i
A :	1				= 1	20
B :		1			= 1	20
C :			1		= 1	20
D :				1	= 1	20
	1	1	1	1	$\boxed{80}$	

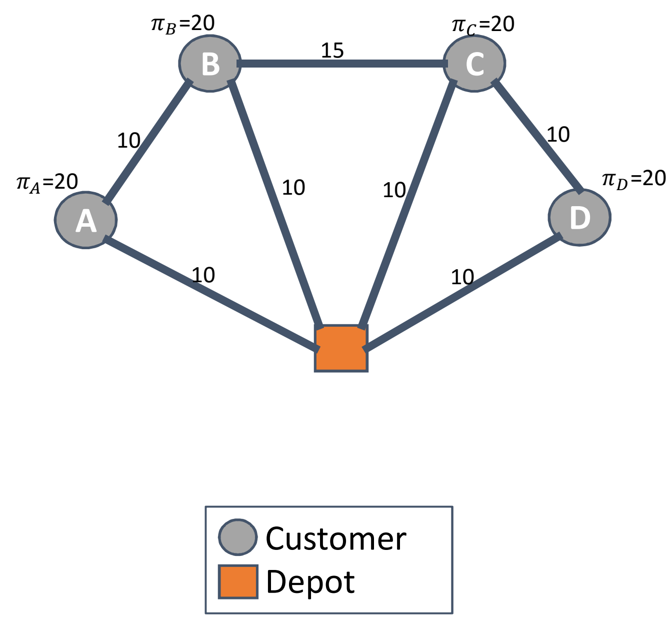


Vehicle routing problem

An example (max 2 clients)

	x_1	x_2	x_3	x_4		
$\hat{c}$	0	0	0	0		π_i
A :	1				= 1	20
B :		1			= 1	20
C :			1		= 1	20
D :				1	= 1	20
	1	1	1	1	$\boxed{80}$	

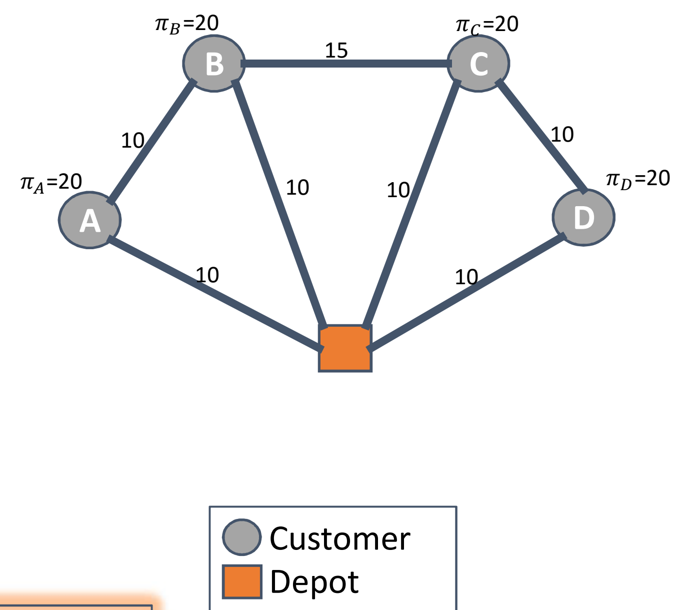
π_i : Marginal price of visiting customer *I*



Vehicle routing problem

An example (max 2 clients)

	x_1	x_2	x_3	x_4		
$\hat{c}$	0	0	0	0		π_i
A :	1				= 1	20
B :		1			= 1	20
C :			1		= 1	20
D :				1	= 1	20
	1	1	1	1	$\boxed{80}$	



π_i : Marginal price of visiting customer *I*

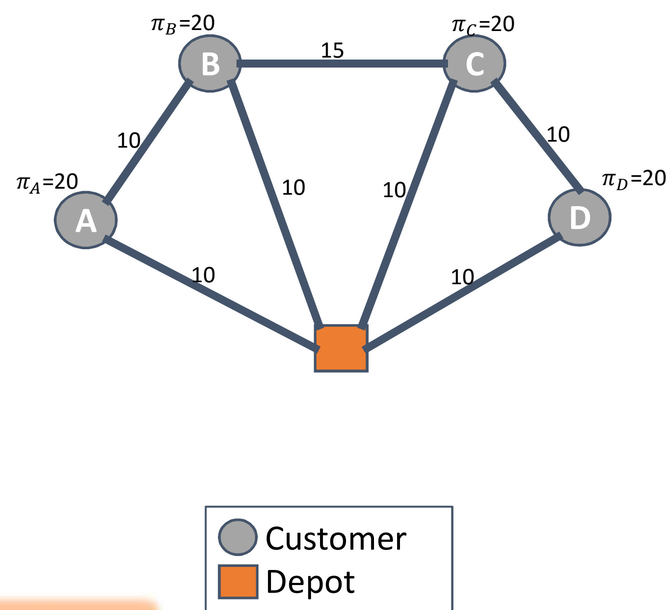
Can I find a route such that:

$$c < \sum \pi_i$$

Vehicle routing problem

An example (max 2 clients)

	x_1	x_2	x_3	x_4		π_i
ĉ	0	0	0	0		
A :	1				= 1	20
B :		1			= 1	20
C :			1		= 1	20
D :				1	= 1	20
	1	1	1	1	$\boxed{80}$	



π_i : Marginal price of visiting customer *I*

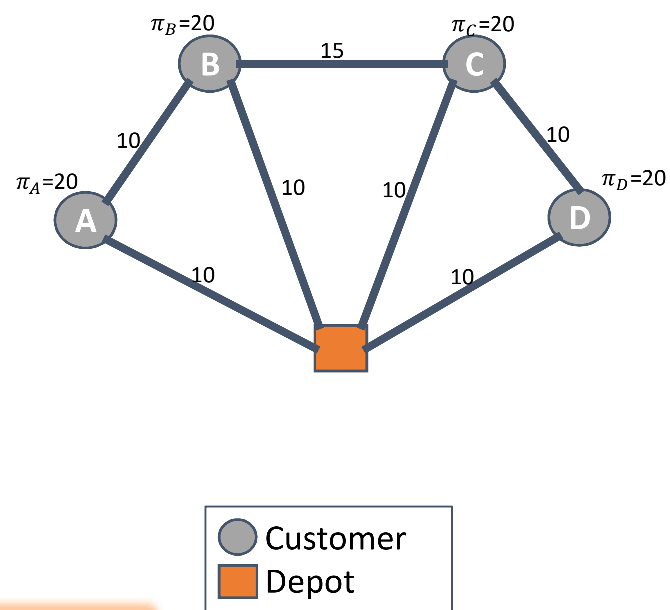
Can I find a route such that:

$$c - \sum \pi_i < 0$$

Vehicle routing problem

An example (max 2 clients)

	x_1	x_2	x_3	x_4		
$\hat{c}$	0	0	0	0		π_i
A :	1				= 1	20
B :		1			= 1	20
C :			1		= 1	20
D :				1	= 1	20
	1	1	1	1	$\boxed{80}$	



π_i : Marginal price of visiting customer *I*

Can I find a route such that:

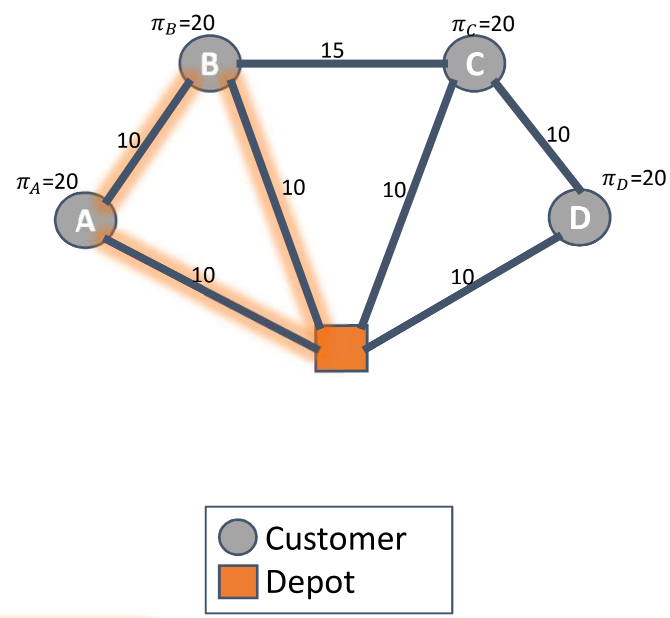
$$c - \sum \pi_i < 0$$

Reduced cost!

Vehicle routing problem

An example (max 2 clients)

	x_1	x_2	x_3	x_4		π_i
ĉ	0	0	0	0		
A :	1				= 1	20
B :		1			= 1	20
C :			1		= 1	20
D :				1	= 1	20
	1	1	1	1	80	



π_i : Marginal price of visiting customer *I*

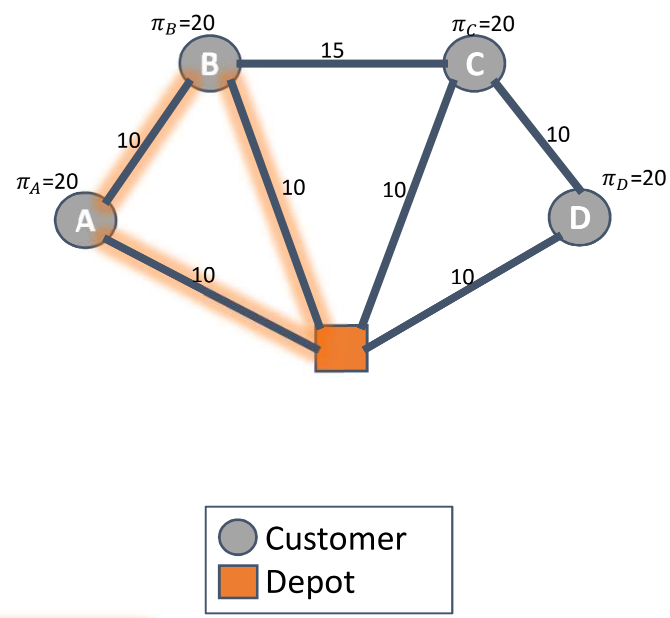
Can I find a route such that:

$$c - \sum \pi_i < 0$$

Vehicle routing problem

An example (max 2 clients)

	x_1	x_2	x_3	x_4	x_5		
ĉ	0	0	0	0	-10		π_i
A :	1				1	= 1	20
B :		1			1	= 1	20
C :			1			= 1	20
D :				1		= 1	20
	1	1	1	1		$\boxed{80}$	



π_i : Marginal price of visiting customer *I*

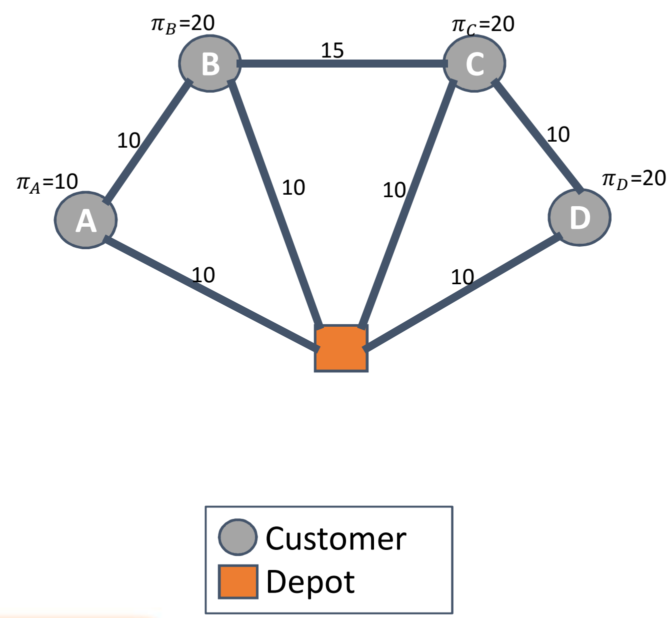
Can I find a route such that:

$$c - \sum \pi_i < 0$$

Vehicle routing problem

An example (max 2 clients)

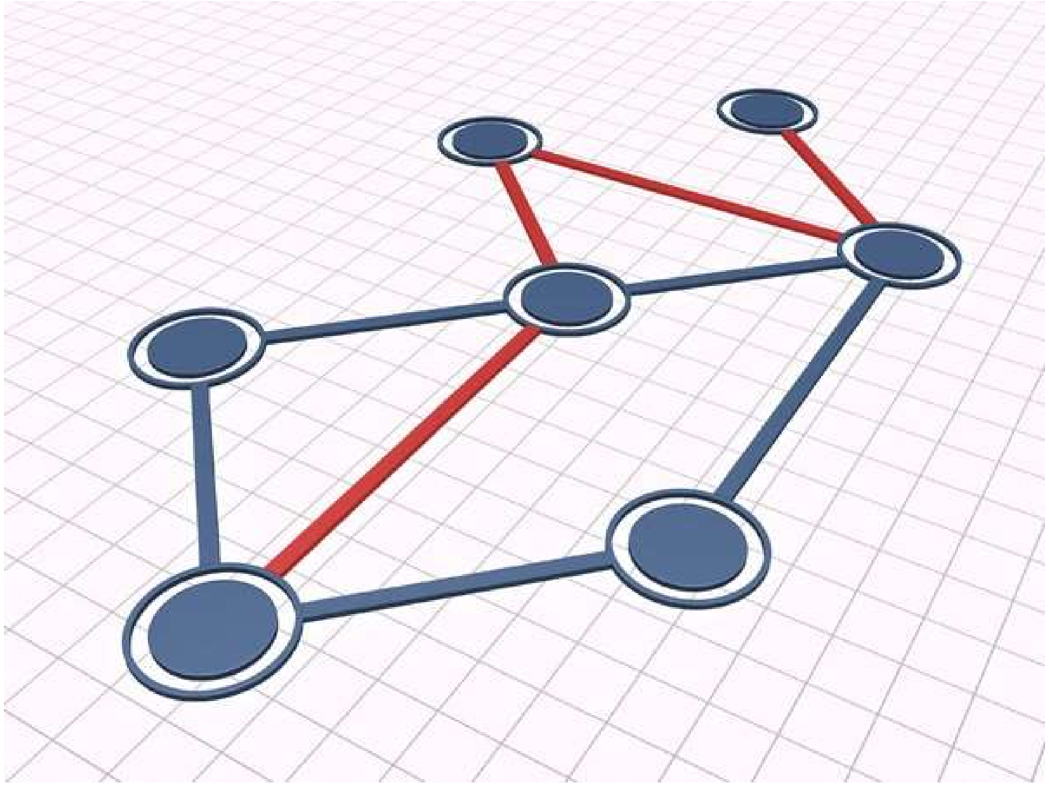
	x_1	x_2	x_3	x_4	x_5		
ĉ	10	0	0	0	0		π_i
A :	1				1	= 1	10
B :		1			1	= 1	20
C :			1			= 1	20
D :				1		= 1	20
		0	1	1	1	$\boxed{70}$	



π_i : Marginal price of visiting customer *I*

Can I find a route such that:

$$c - \sum \pi_i < 0$$



Sub Probelm for the

Vehicle routing problem

General Subproblem

Implicit representation of all variables

- Every possible solution to the subproblem is a variable

Optimization objective:

→ find variable with (the most) negative reduced cost

$$\text{Min } \hat{c} = c - \sum_i a_i \pi_i \qquad a_i = \begin{cases} 1, & \textit{if customer } i \textit{ is visited} \\ 0, & \textit{otherwise} \end{cases}$$

$$c = \sum_x c_x x$$

General Subproblem

Implicit representation of all variables

- Every possible solution to the subproblem is a variable

Optimization objective:

→ find variable with (the most) negative reduced cost

$$\text{Min } \hat{c} = \sum_x c_x x - \sum_i \pi_i a_i \qquad a_i = \begin{cases} 1, & \textit{if customer } i \textit{ is visited} \\ 0, & \textit{otherwise} \end{cases}$$

Subproblem

Implicit representation of all variables

- Every possible solution to the subproblem is a variable

Optimization objective:

→ find variable with (the most) negative reduced cost

$$\text{Min } \hat{c} = \sum_x c_x x - \sum_i \pi_i a_i$$

$$a_i = \begin{cases} 1, & \textit{if customer i is visited} \\ 0, & \textit{otherwise} \end{cases}$$

Subject to: Capacity constraints

Flow conservation constraints

Shortest-path problem with resource constraints:
Dynamic programming

Resources Constraint SPP

Resource $r = 1,\ldots,R$

Resource consumption $t^r_{ij} > 0$ on each arc.

Resources window$[a^r_i, b^r_i]$ at each node

- Resources level cannot go above b^r_i when node v_i is reached
- If t^r_{ij} is below a^r_i when node path reaches v_i then is it set to a^r_i

Resources Constraint SPP - DP

Dynamic Programming Algorithm

- L_i : list of labels associated with node v_i
- label $l = (c, T^1, \ldots, T^R)$ where
  - a label represents a partial path from v_0 to v_i
  - c is the cost of the label or
  - T^r is the consumption level of resource r
  - v(l) is the node which to which l is associated

Resources Constraint SPP - DP

Extending a label $l = (c, T^1_i, \ldots, T^R_i)$ from v_i to v_j

- Create a label $(c + c_{ij}, T^1 + t^1_{ij}, \ldots, T^R + t^R_{ij})$
  - Making sure we respect $[a^1_j, b^1_j], \ldots, [a^R_j, b^R_j]$
- Insert the label in the list of labels associated with v_j
- Apply Dominance Rules
  - Without such rules, the algorithm would enumerates all possible paths
- Resources constraints make sure the algorithm terminates

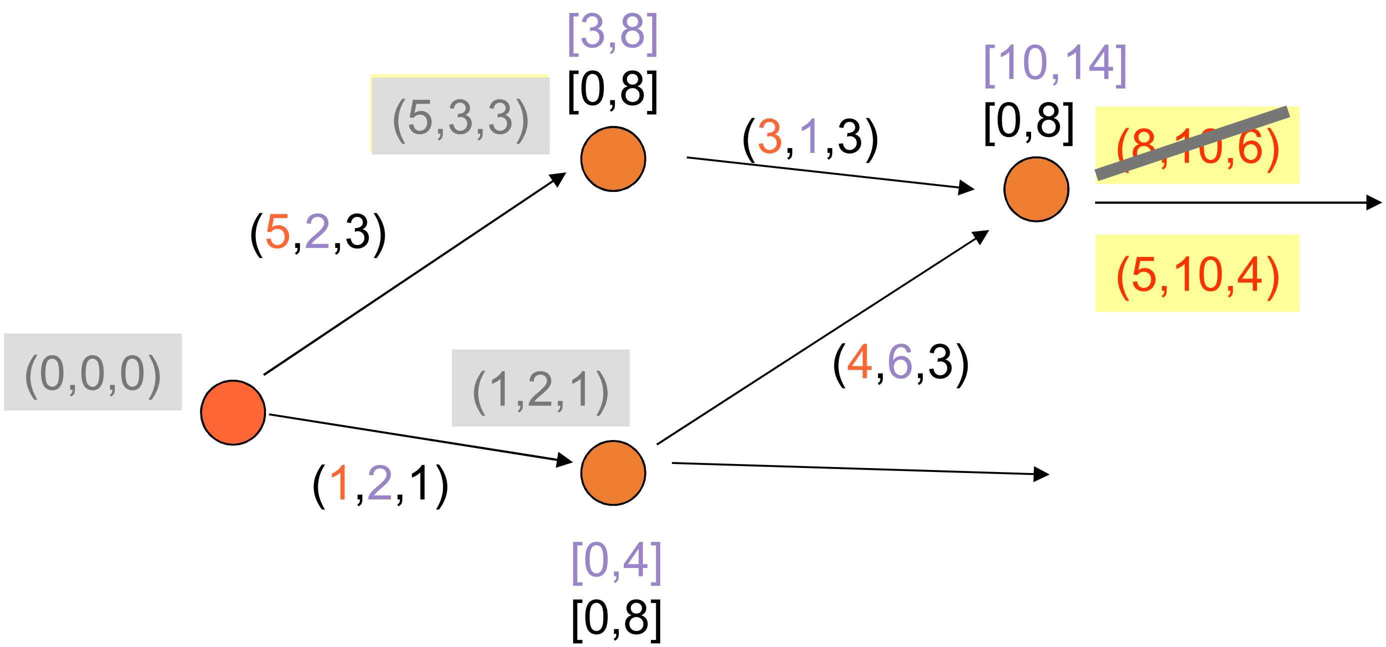
Resources Constraint SPP - DP

Dominance Rules: l_1 dominates l_2 iff :

- $c(l_1) <= c(l_2)$
- Every feasible future extensions of l_2 will be feasible for l_1
  - *Most often* we check that $T^r(l_1) <= T^r(l_2)$ for all r

Dominance: an example

label : (c, time, capacity)



Subproblem – Constraint Programming

"Arc Flow" model

Objectives:

- Minimize: $\sum_i (\text{ReducedCost}(i, S_i))$

Variables:

- $S_i \in N$ — Successor of node i
- $V_i \in \{False, True\}$ — Node i visited by current path
- $l_i \in [0..Capacity]$ — Truck load after visit of node i

Constraints:

- $S_i = i \rightarrow V_i = False$ — S-V Coherence constraints
- AllDiff(S) — Conservation of flow
- Circuit(S) — SubTour elimination constraint
- $S_i = j \rightarrow l_i + D_j = l_j$ — Capacity constraints

+ Redundant Constraints from work on TSP(TW)

Subproblem – Constraint Programming

"Position" model

Objectives:

- Minimize: $\sum_k$ (ReducedCost(P_k, P_{k+1}))

Variables:

- $P_k \in N$ Node visited a position k
- $L_k \in [0..Capacity]$ Truck load after visiting position k

Constraints:

- **AllDiff(P)** **Elementarity of the path**
- $L_{k+1} = L_k + D_{Pk}$ Capacity constraints
- $P_k = \text{depot} \rightarrow P_{k+1} = \text{depot}$ Padding at the end of path

Can you compare these models?

"Arc Flow" model

Objectives:

- Minimize: $\sum_i$ (ReducedCost(i, S_i))

Variables:

- $S_i \in N$
- $V_i \in$ {False,True}
- $l_i \in$ [0..Capacity]

Constraints:

- $S_i = i \rightarrow V_i$ =False
- AllDiff(S)
- Circuit(S)
- $S_i = j \rightarrow l_i + D_j = l_j$

"Position" model

Objectives:

- Minimize: $\sum_k$ (ReducedCost(P_k, P_{k+1}))

Variables:

- $P_k \in N$
- $L_k \in$ [0..Capacity]

Constraints:

- AllDiff(P)
- $L_{k+1} = L_k + D_{Pk}$
- P_k = depot $\rightarrow P_{k+1}$ = depot

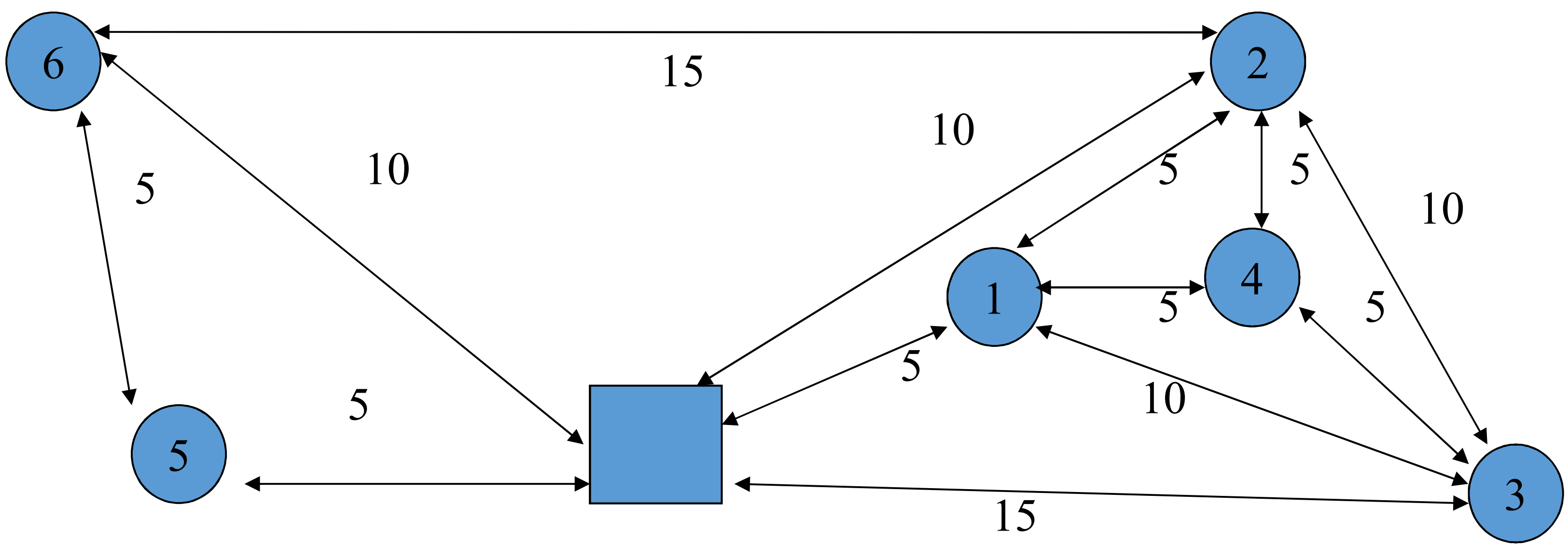
Column generation

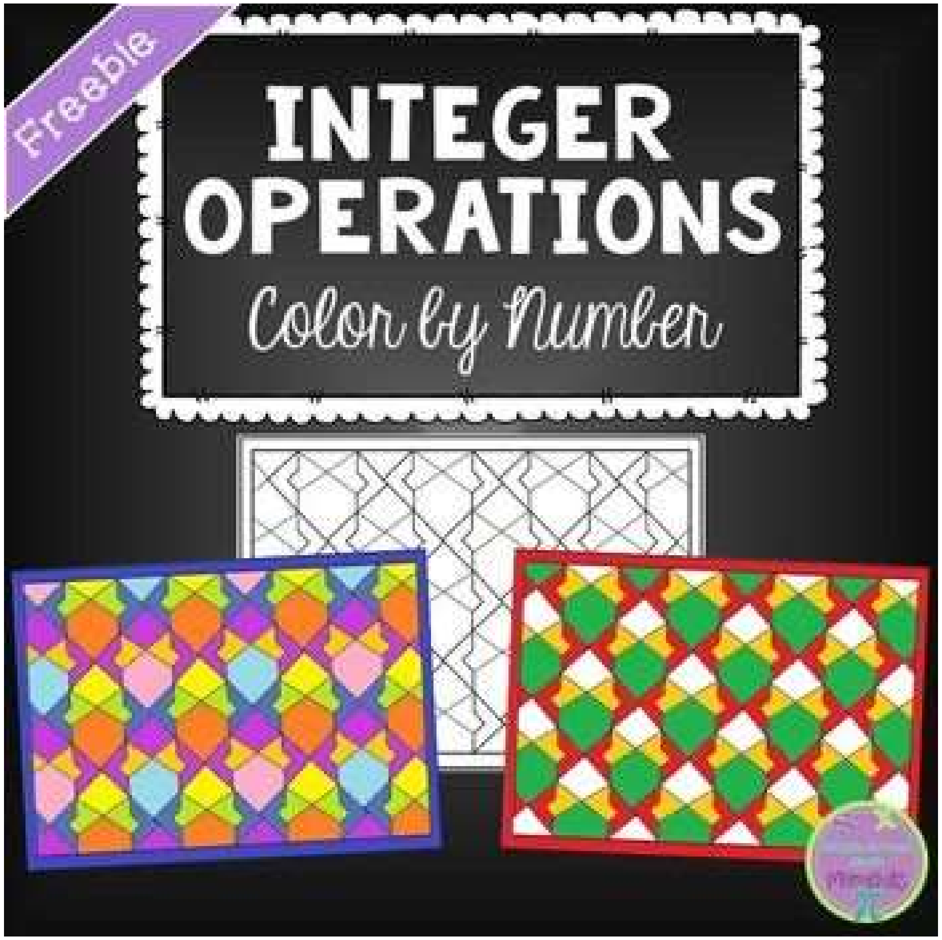
In Practice

"I expect you all to be independent, innovative, critical thinkers who will do exactly as I say!"

DIY in Excell + CP Solver

- Solve the following VRP problem using ColGen, knowing that
  - A route can visit at most 4 customers





Branch-and-price

Obtaining integer solutions

Branch-and-price

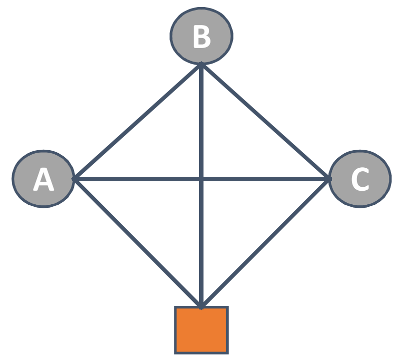
Column generation + MIP : Branch-and-price

- How to obtain integer solutions?
  - Branch-and-bound -> solve LP relaxation at each node
  - Branch-and-price -> column generation to solve LP relaxation at each node

Branch-and-price

Vehicle routing problem

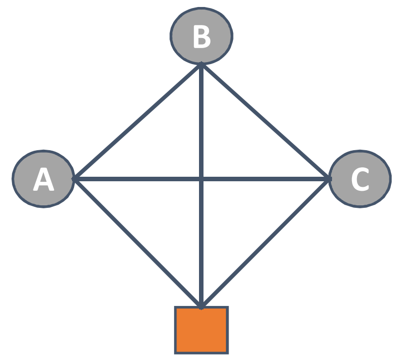
- Max 2 customers
- Cost of all arc : 1



Branch-and-price

Vehicle routing problem

- Max 2 customers
- Cost of all arc : 1

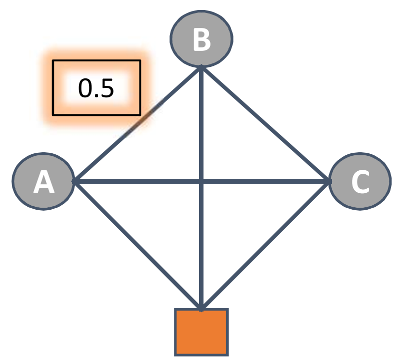


	x_1	x_2	x_3	
Min	3	3	3	
A :	1	1		= 1
B :	1		1	= 1
C :		1	1	= 1
OptSol:	0.5	0.5	0.5	$\boxed{4.5}$

Branch-and-price

Vehicle routing problem

- Max 2 customers
- Cost of all arc : 1

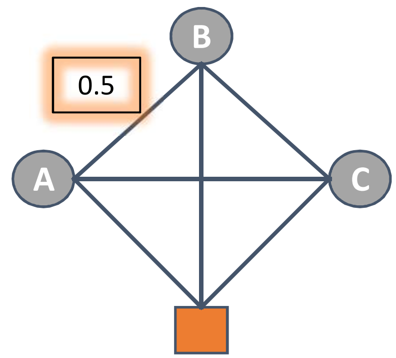


	x_1	x_2	x_3	
Min	3	3	3	
A :	1	1		= 1
B :	1		1	= 1
C :		1	1	= 1
OptSol:	0.5	0.5	0.5	$\boxed{4.5}$

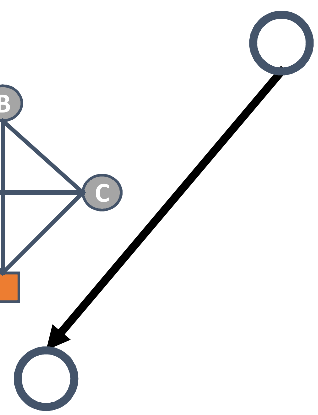
Branch-and-price

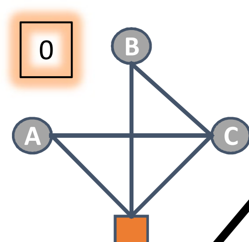
Vehicle routing problem

- Max 2 customers
- Cost of all arc : 1



	x_1	x_2	x_3	
Min	3	3	3	
A :	1	1		= 1
B :	1		1	= 1
C :		1	1	= 1
OptSol:	0.5	0.5	0.5	$\boxed{4.5}$



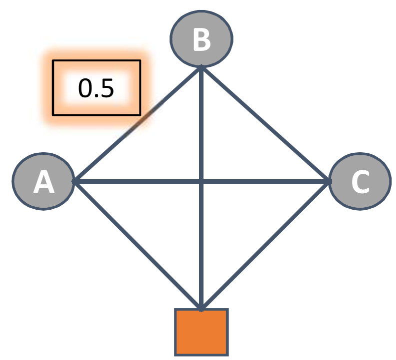


	x_2	x_3	
Min	3	3	
A :	1		= 1
B :		1	= 1
C :	1	1	= 1

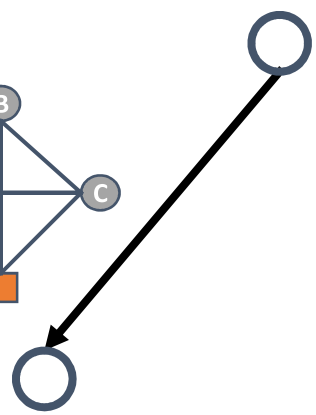
Branch-and-price

Vehicle routing problem

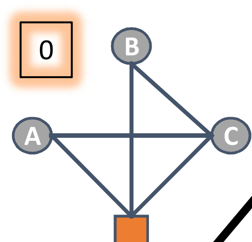
- Max 2 customers
- Cost of all arc : 1



	x_1	x_2	x_3	
Min	3	3	3	
A :	1	1		= 1
B :	1		1	= 1
C :		1	1	= 1
OptSol:	0.5	0.5	0.5	$\boxed{4.5}$



	x_4
	2
A :	1
B :	
C :	

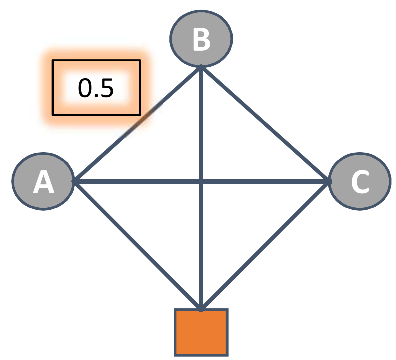


	x_1	x_2	x_3	
Min	3	3	3	
A :	1	1		= 1
B :	1		1	= 1
C :		1	1	= 1

Branch-and-price

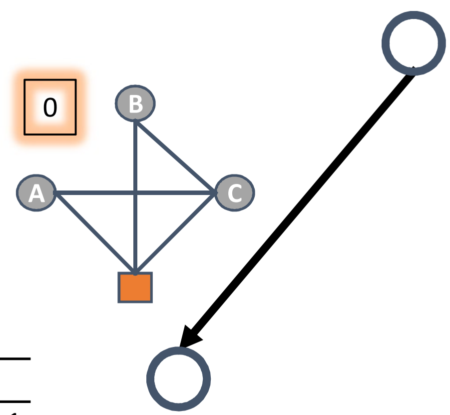
Vehicle routing problem

- Max 2 customers
- Cost of all arc : 1



	x_1	x_2	x_3	
Min	3	3	3	
A :	1	1		= 1
B :	1		1	= 1
C :		1	1	= 1
OptSol:	0.5	0.5	0.5	$\boxed{4.5}$

	x_4
	2
A :	1
B :	
C :	

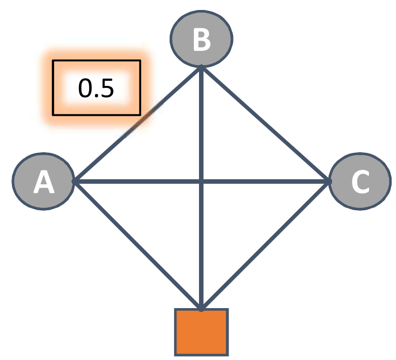


	x_1	x_2	x_3	x_4	
Min	3	3	3	2	
A :	1	1		1	= 1
B :	1		1		= 1
C :		1	1		= 1
OptSol:			1	1	$\boxed{5}$

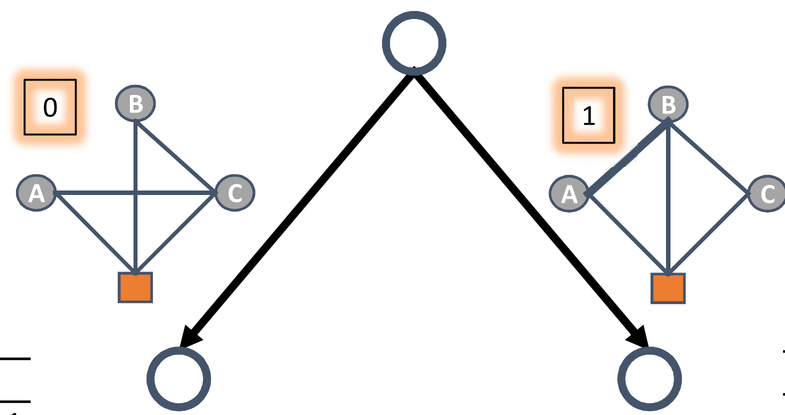
Branch-and-price

Vehicle routing problem

- Max 2 customers
- Cost of all arc : 1



	x_1	x_2	x_3	
Min	3	3	3	
A :	1	1		= 1
B :	1		1	= 1
C :		1	1	= 1
OptSol:	0.5	0.5	0.5	$\boxed{4.5}$



	x_1	x_2	x_3	x_4	
Min	3	3	3	2	
A :	1	1		1	= 1
B :	1		1		= 1
C :		1	1		= 1
OptSol:			1	1	$\boxed{5}$

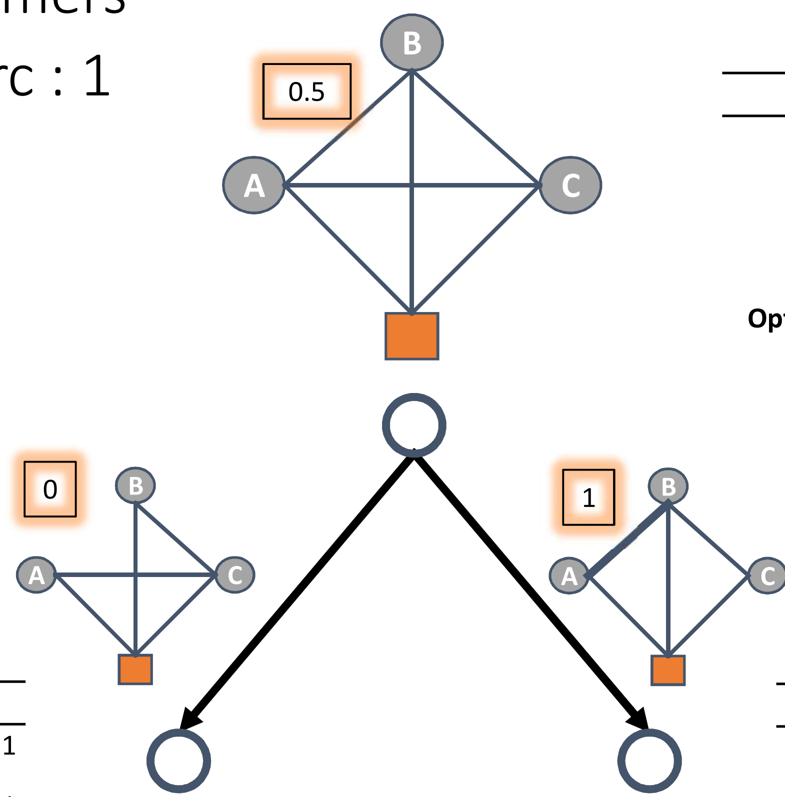
	x_1	x_2	x_3	
Min	3	3	3	
A :	1	1		= 1
B :	1		1	= 1
C :		1	1	= 1

Branch-and-price

Vehicle routing problem

- Max 2 customers
- Cost of all arc : 1

Why branch on arc-flow variables?



	x_1	x_2	x_3	
Min	3	3	3	
A :	1	1		= 1
B :	1		1	= 1
C :		1	1	= 1
OptSol:	0.5	0.5	0.5	$\boxed{4.5}$

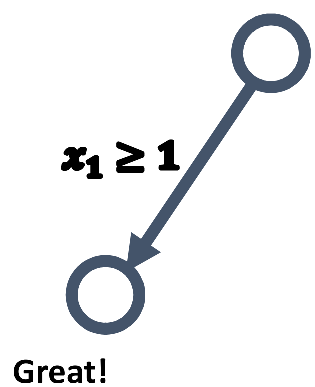
	x_1	x_2	x_3	x_4	
Min	3	3	3	2	
A :	1	1		1	= 1
B :	1		1		= 1
C :		1	1		= 1
OptSol:			1	1	$\boxed{5}$

	x_1	x_2	x_3	x_5	
Min	3	3	3	2	
A :	1	1			= 1
B :	1		1		= 1
C :		1	1	1	= 1
OptSol:	1			1	$\boxed{5}$

Branch-and-price

Branching possibilities

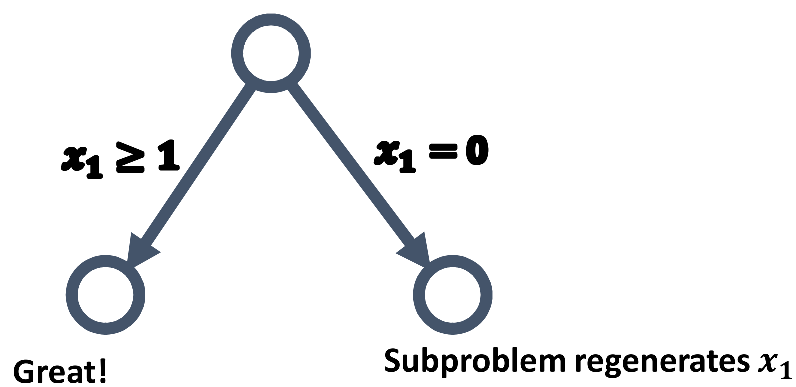
- Branch on master variables



Branch-and-price

Branching possibilities

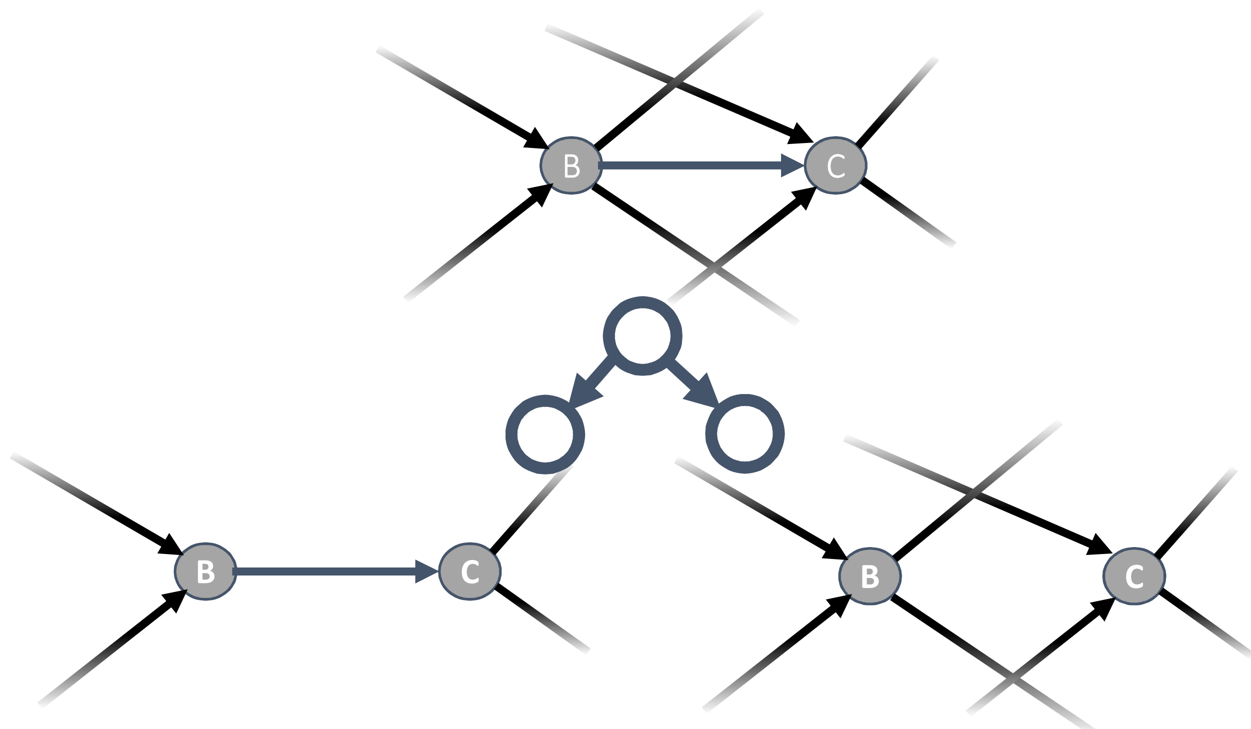
- Branch on master variables



Branch-and-price

Branching possibilities

- Branch on master variables… NO!
- Branch on subproblem variables

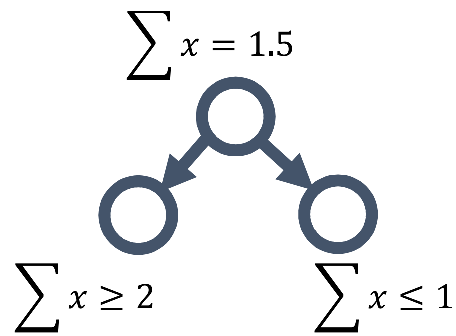


Branch-and-price

Branching possibilities

- Branch on master variables… NO!
- Branch on subproblem variables
- Branch on the master problem constraints
  - BUT adding a constraints c **requires** its dual value π_c must be handled in the subproblems
  - Example: Branch on the total number of vehicle used

Best branching for shift scheduling problem



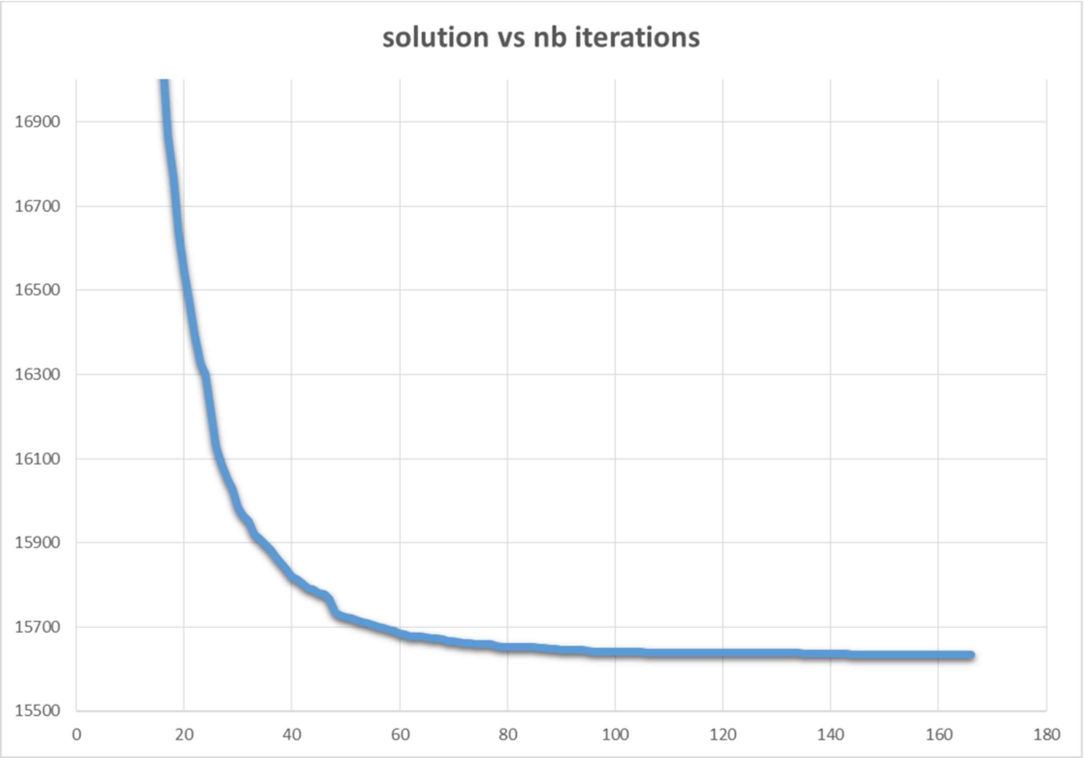
Applied column generation

Main Challenges

Applied column generation

Evolution of costs

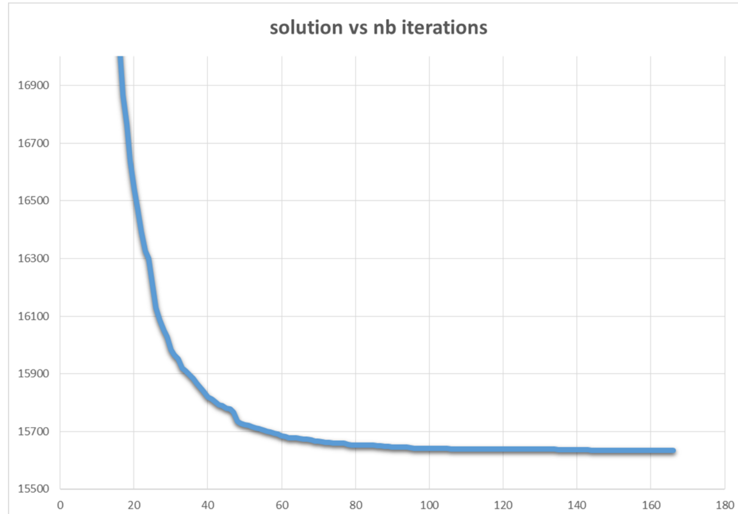
- Long convergence time



Applied column generation

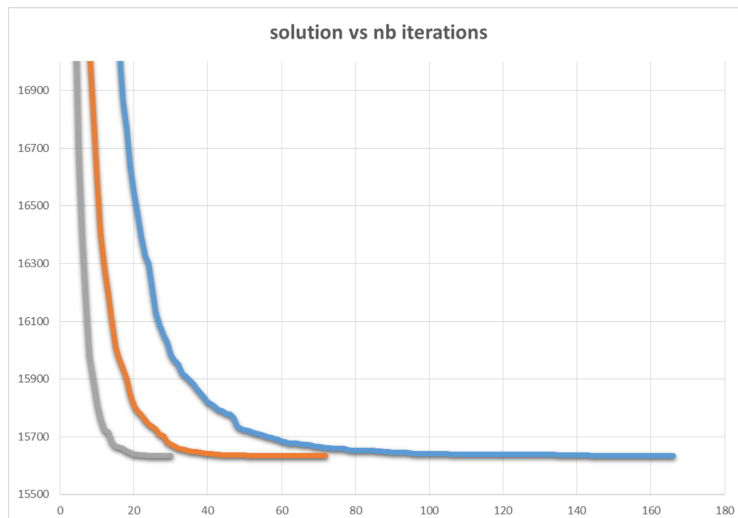
Evolution of costs

- Long convergence time



Speed-up techniques

- Spend more time to generate new columns
- Delete variables in RMP



Applied column generation

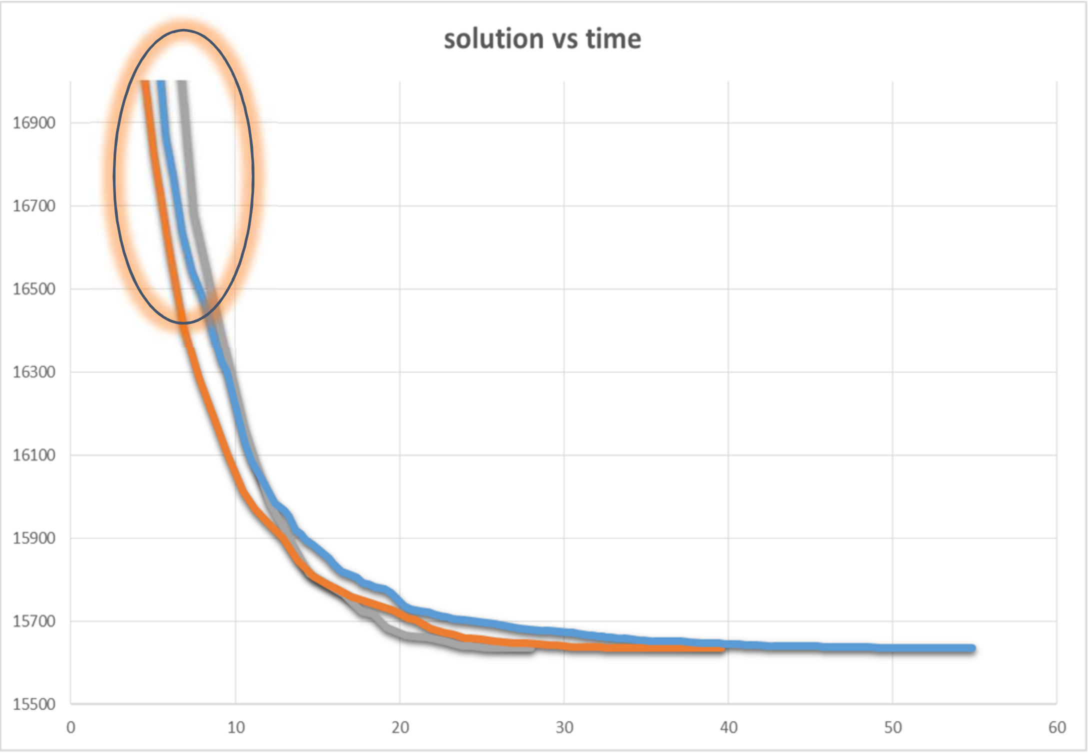
Evolution of costs

- Long convergence time

Speed-up techniques

- Spend more time to generate new columns
- Delete variables in RMP

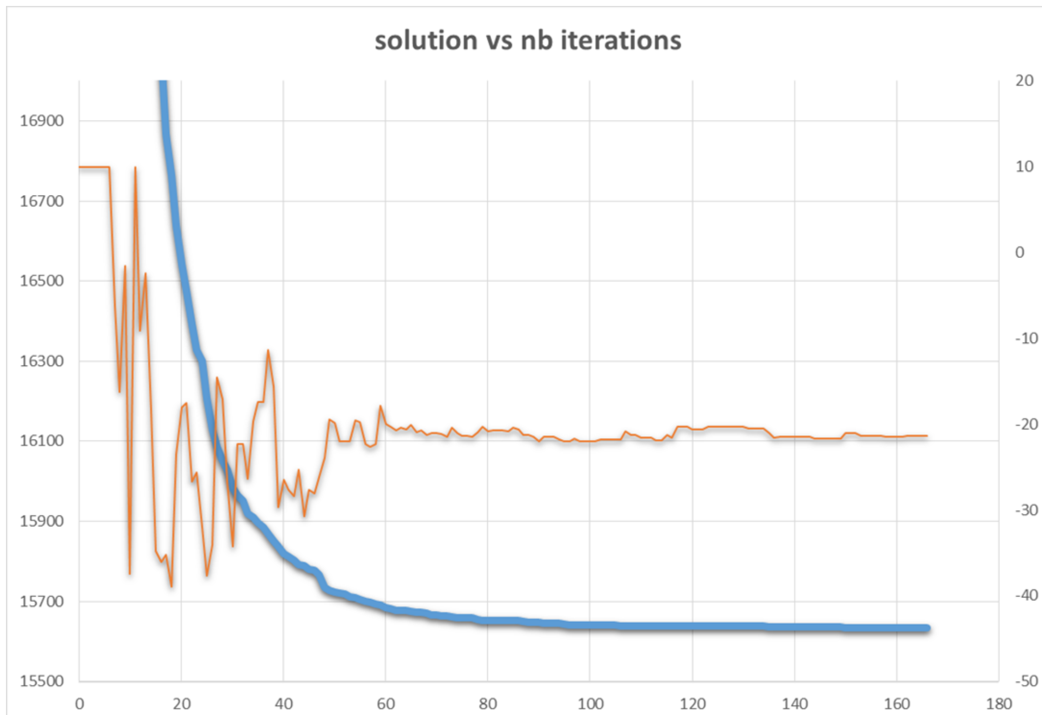
Balance between subproblems and master problem



Applied column generation

Stabilization

- Duals are extreme points
- Master problem is degenerated
- Tail-off effect is due to difficulty finding the right dual vector



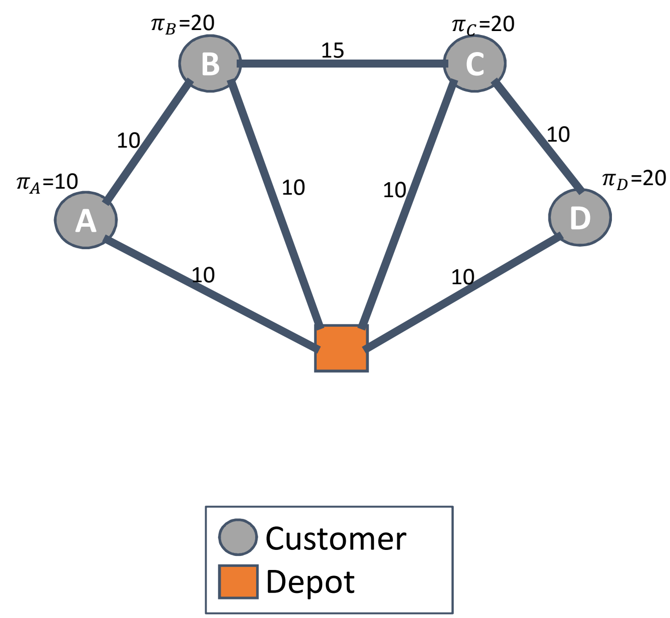
A quick look at

Stabilization issues

Column Generation

Stabilization

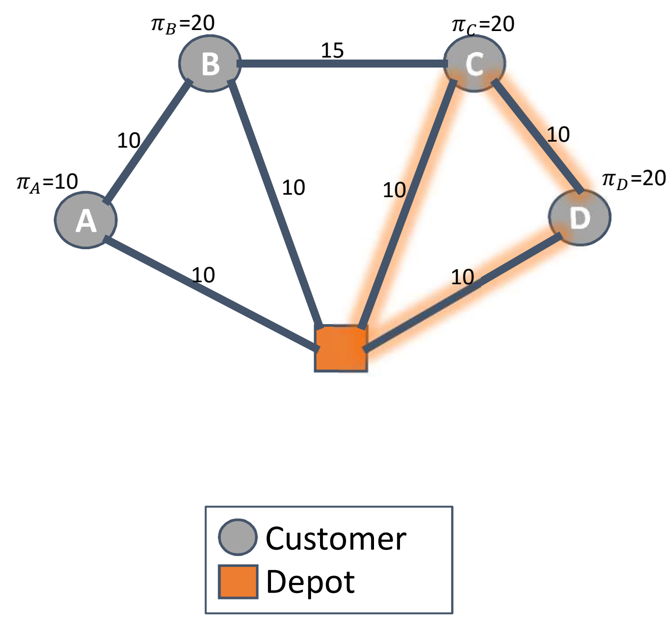
	x_1	x_2	x_3	x_4	x_5		
ĉ	10	0	0	0	0		π_i
A :	1				1	= 1	10
B :		1			1	= 1	20
C :			1			= 1	20
D :				1		= 1	20
		0	1	1	1	$\boxed{70}$	



Column Generation

Stabilization

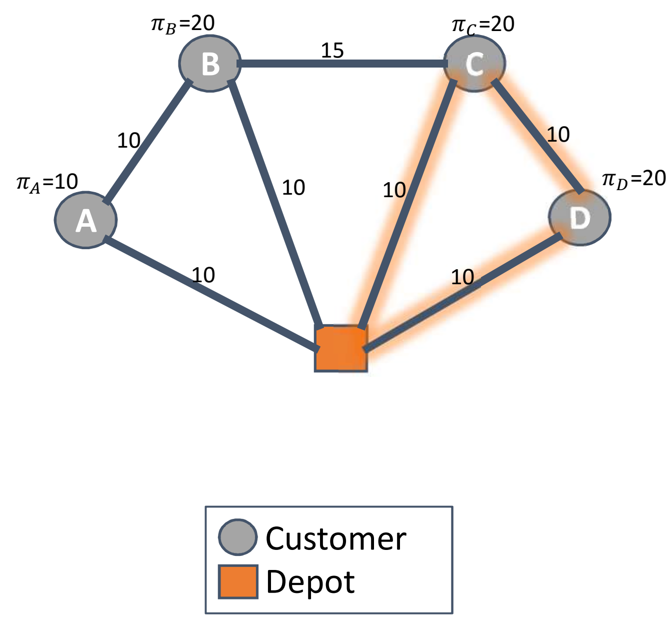
	x_1	x_2	x_3	x_4	x_5		
ĉ	10	0	0	0	0		π_i
A :	1				1	= 1	10
B :		1			1	= 1	20
C :			1			= 1	20
D :				1		= 1	20
		0	1	1	1	$\boxed{70}$	



Column Generation

Stabilization

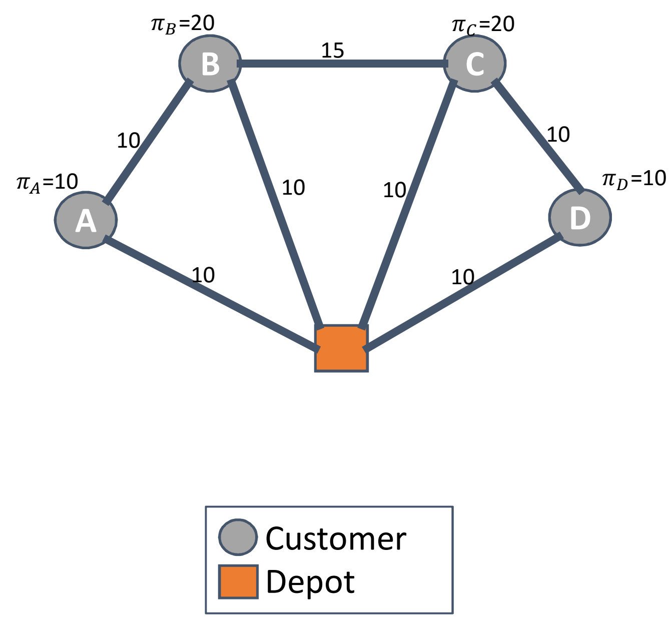
	x_1	x_2	x_3	x_4	x_5	x_6		
ĉ	10	0	0	0	0	-10		π_i
A :	1				1		= 1	10
B :		1			1		= 1	20
C :			1			1	= 1	20
D :				1		1	= 1	20
		0	1	1	1		$\boxed{70}$	



Column Generation

Stabilization

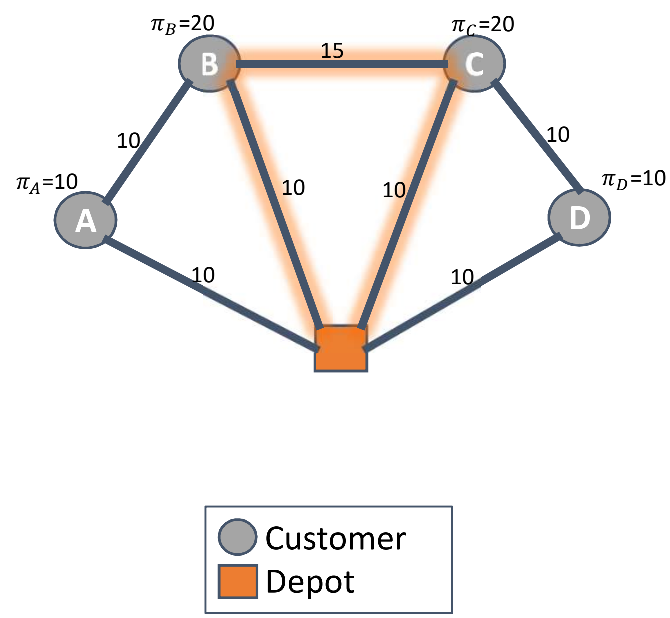
	x_1	x_2	x_3	x_4	x_5	x_6		
ĉ	10	0	0	10	0	0		π_i
A :	1				1		= 1	10
B :		1			1		= 1	20
C :			1			1	= 1	20
D :				1		1	= 1	10
		0	0		1	1	$\boxed{60}$	



Column Generation

Stabilization

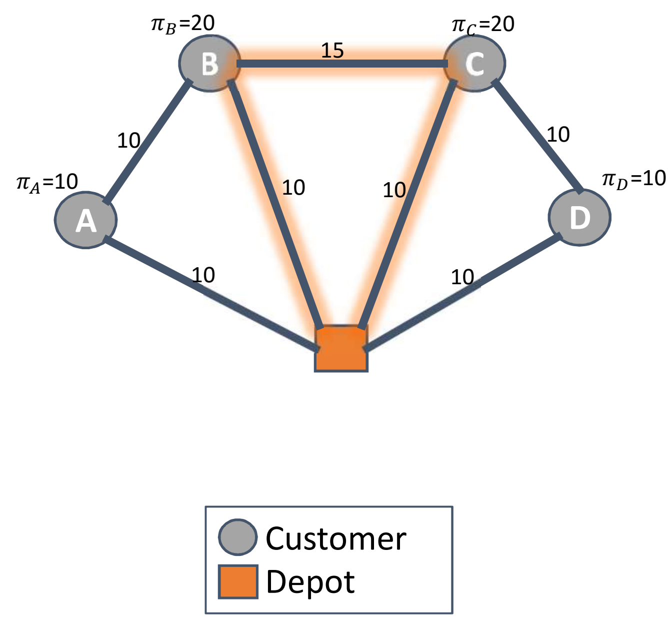
	x_1	x_2	x_3	x_4	x_5	x_6		
ĉ	10	0	0	10	0	0		π_i
A :	1				1		= 1	10
B :		1			1		= 1	20
C :			1			1	= 1	20
D :				1		1	= 1	10
		0	0		1	1	$\boxed{60}$	



Column Generation

Stabilization

	x_1	x_2	x_3	x_4	x_5	x_6	x_7		
$\hat{c}$	10	0	0	10	0	0	-5		π_i
A :	1				1			= 1	10
B :		1			1		1	= 1	20
C :			1			1	1	= 1	20
D :				1		1		= 1	10
		0	0		1	1		$\boxed{60}$	

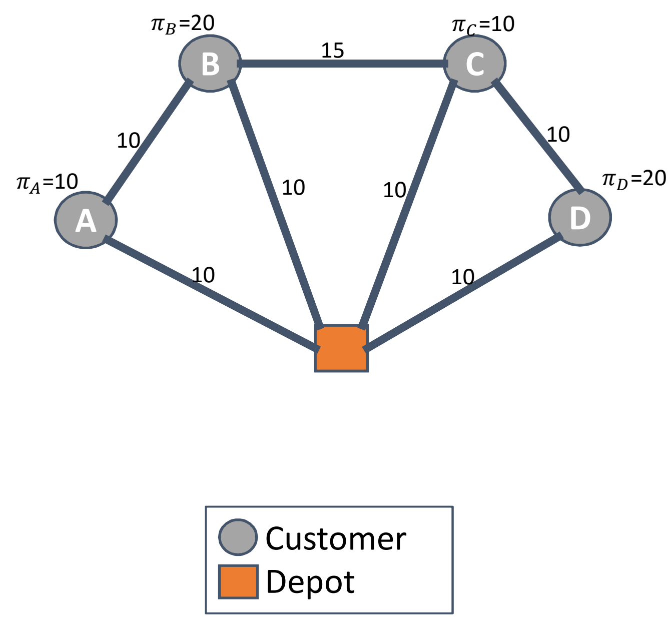


Column Generation

Stabilization

BUT THIS COLUMN IS USELESS

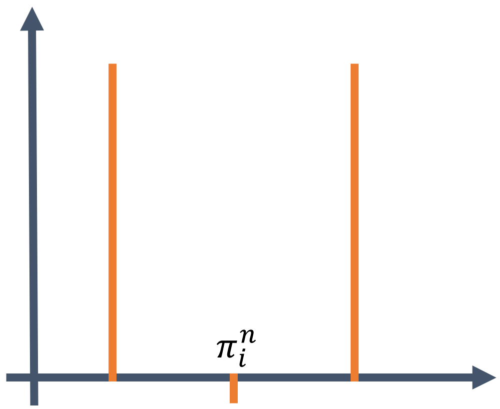
	x_1	x_2	x_3	x_4	x_5	x_6	x_7		
ĉ	10	0	10	0	0	0	5		π_i
A :	1				1			= 1	10
B :		1			1		1	= 1	20
C :			1			1	1	= 1	10
D :				1		1		= 1	20
		0		0	1	1		$\boxed{60}$	



Column Generation

Stabilization!

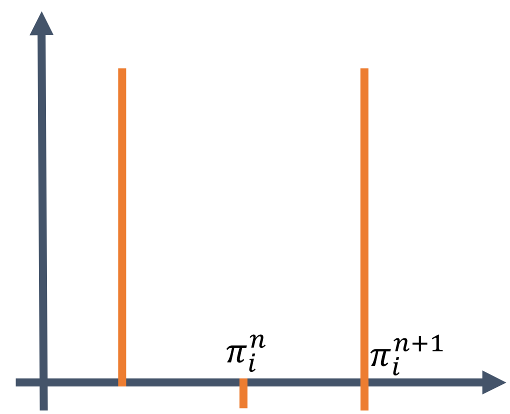
- What to do?
- Popular technique
  - Box penalization



Column Generation

Stabilization!

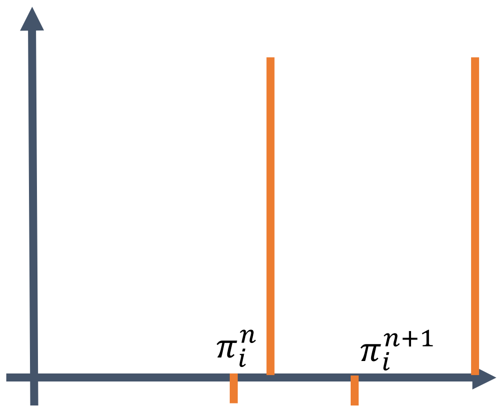
- What to do?
- Popular technique
  - Box penalization



Column Generation

Stabilization!

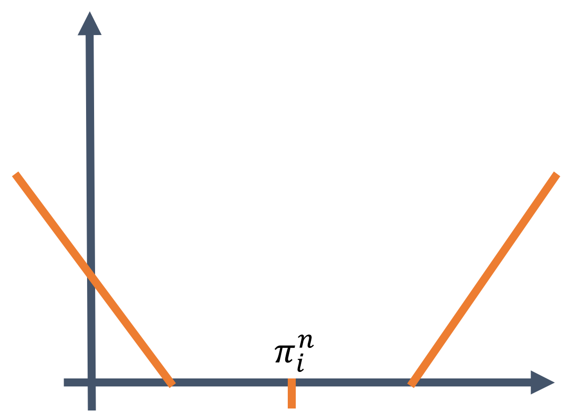
- What to do?
- Popular technique
  - Box penalization



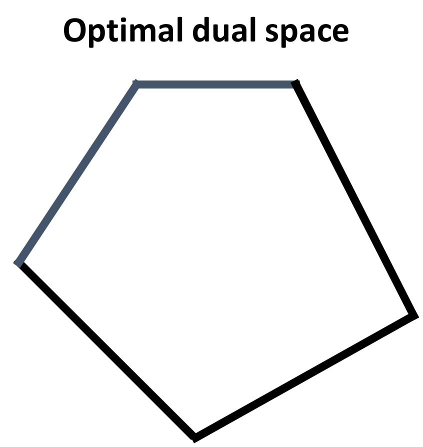
Column Generation

Stabilization!

- What to do?
- Popular technique
  - Box penalization



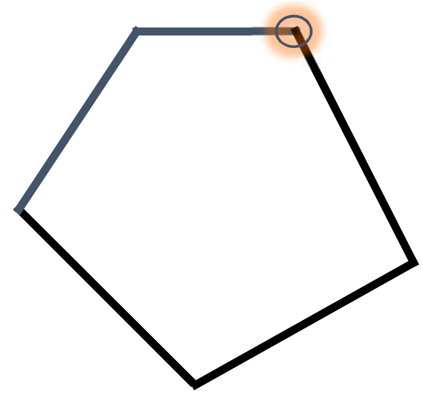
Column Generation



Stabilization!

- What to do?
- Popular technique
  - Box penalization
- Interior point stabilization

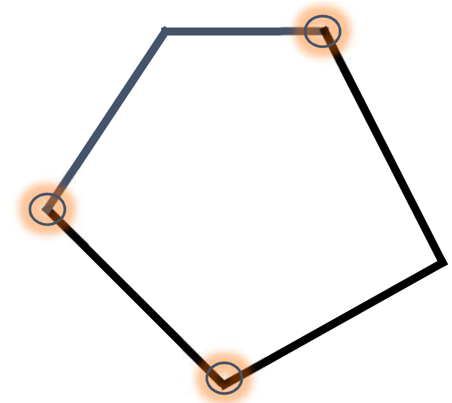
Column Generation



Stabilization!

- What to do?
- Popular technique
  - Box penalization
- Interior point stabilization
  - Adding a variable to the primal is equivalent to adding a cut to the dual

Column Generation

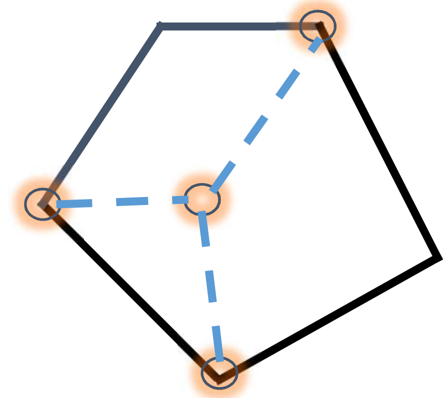


Stabilization!

- What to do?
- Popular technique
  - Box penalization
- Interior point stabilization
  - Find multiple dual optimal extreme points

Column Generation

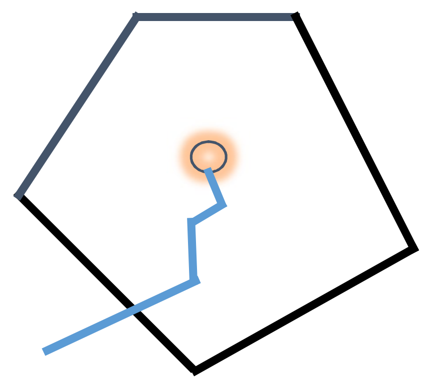
	Average time	Average nb Iterations
Unstabilized	384.4 s	72.6
Box penalization	389.1 s	61.0
IPS	277.9 s	37.1



Stabilization!

- What to do?
- Popular technique
  - Box penalization
- Interior point stabilization
  - Find multiple dual optimal extreme points
    - Do a linear combination

Column Generation



Stabilization!

- What to do?
- Popular technique
  - Box penalization
- Interior point stabilization
  - Find multiple dual optimal extreme points
    - Do a linear combination
  - Simple idea: barrier algorithm without crossover

Back to the Primal

Finding good solution fast: An Homecare Application

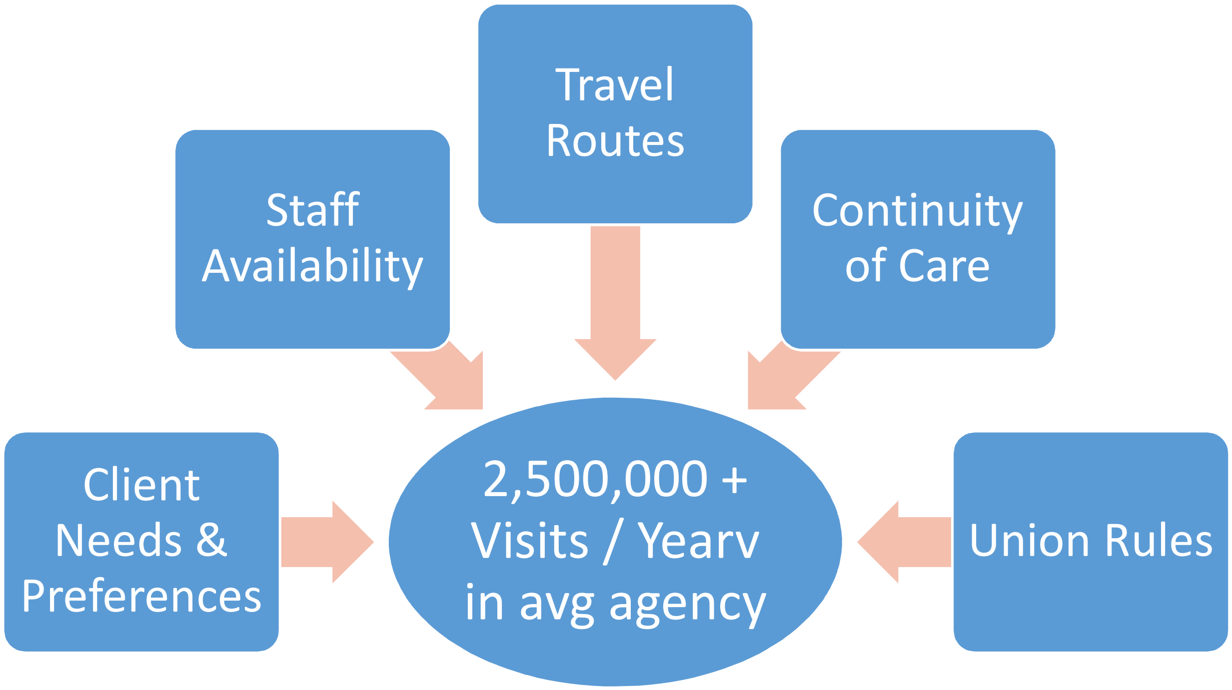
Problem Definition

- Problem Definition
- Mathematical Formulation
- Resolution Method
- Computation Results
- Conclusion

The home care in Canada

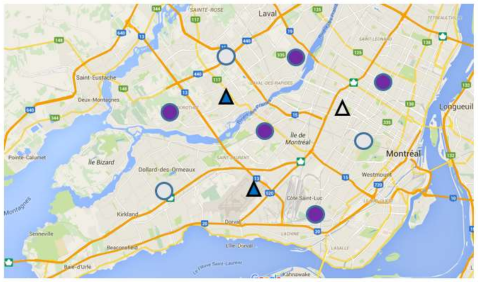
- People want to stay at home as long as possible
- In 2012, approximately 2.2 million people relied on home care services
- For the same cares, a patient at home costs 90% less than a patient at the hospital
- Homecare services is one of the fastest growing market in the US and Canada

The Scheduling Challenge



An example

Monday



Tuesday

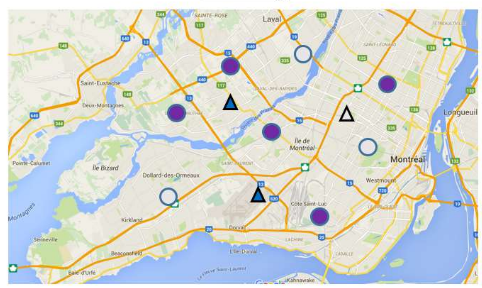
Wednesday

Thursday

Friday

Saturday

Sunday



● Available day for the patient

▲ Work day of the nurse

○ △ Patient/Nurse not available

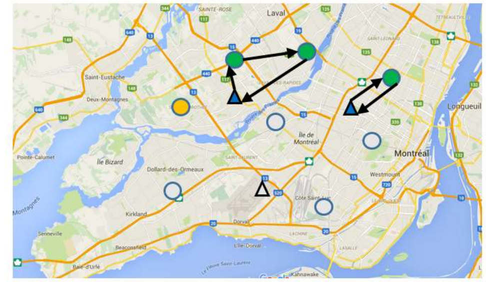
Each patient needs 3 visits

An example

Monday

Tuesday

Wednesday

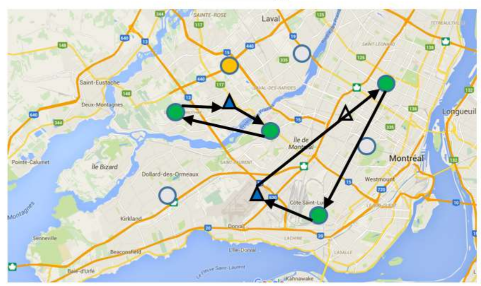


Thursday

Friday

Saturday

Sunday



Unused available day

Used available day

Patient/Nurse not available

Problem Definition

- This Homcare routing problem (HHCRSP) can be described as mix between an assignment problem

Hard constraints	Soft constraints
• Mandatory requirements : nurse skills, type of care, … • Forbidden nurses	• Continuity of care • Optional requirements

Problem Definition

- The HHCRSP can be described as mix between an assignment problem and a multi-attributes VRP

Hard constraints	Soft constraints
• Mandatory requirements : nurse skills, type of care, … • Forbidden nurses • Time windows • Available days • Workdays • Time-dependent travel time	• Continuity of care • Optional requirements • Travel time • Min/Max worktime week • Min/Max worktime workday • Number of visits over the week

Problem Definition

- The HHCRSP can be described as mix between an assignment problem and a multi-attributes VRP

Hard constraints	Soft constraints
• Mandatory requirements : nurse skills, type of care, …	• Continuity of care
• Forbidden nurses	• Optional requirements
• Time windows	• Travel time
• Available days	• Min/Max worktime week
• Workdays	• Min/Max worktime workday
• Time-dependent travel time	• Number of visits over the week

Objective function = weighted sum

Mathematical Formulation

- Problem Definition
- Mathematical Formulation
- Resolution Method
- Computation Results
- Conclusion

Formulation

- The HHCRSP can be formulated as a set partitioning problem

- The decision variables correspond to the feasible routes for each nurse for each one of his/her workdays

Set partitioning model

Use the route ω

P : Patients
N : Nurses
Ω : Routes

$$\underset{x}{\text{minimize}} \quad \sum_{\omega\in\Omega} c_\omega x_\omega + C.\sum_{n\in N}(o_n+u_n) + U.\sum_{p\in P} z_p$$

$$\text{subject to} \quad \sum_{\omega\in\Omega_d} a_{\omega,p}x_\omega \le 1 \qquad \forall_{p\in P},\forall_{d\in A_d}$$

$$\sum_{\omega\in\Omega} a_{\omega,p}x_\omega + z_p = n_p \qquad \forall_{p\in P}$$

$$\sum_{\omega\in\Omega_d\cap\Omega_n} x_\omega \le 1 \qquad \forall_{n\in N},\forall_{d\in W_d}$$

$$\sum_{\omega\in\Omega} l_\omega x_\omega + u_n \ge min_n \qquad \forall_{n\in N}$$

$$\sum_{\omega\in\Omega} l_\omega x_\omega - o_n \le max_n \qquad \forall_{n\in N}$$

$$x_\omega \in \{0,1\} \qquad \forall_{\omega\in\Omega}$$

$$z_p \in \mathbb{N} \qquad \forall_{p\in P}$$

$$o_n, u_n \ge 0 \qquad \forall_{n\in N}$$

Set partitioning model

Overtime of the nurse

P : Patients
N : Nurses
Ω : Routes

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & \sum_{\omega \in \Omega} c_\omega x_\omega + C. \sum_{n \in N} (o_n + u_n) + U. \sum_{p \in P} z_p & \\
\text{subject to} \quad & \sum_{\omega \in \Omega_d} a_{\omega,p} x_\omega \leq 1 & \forall_{p \in P}, \forall_{d \in A_d} \\
& \sum_{\omega \in \Omega} a_{\omega,p} x_\omega + z_p = n_p & \forall_{p \in P} \\
& \sum_{\omega \in \Omega_d \cap \Omega_n} x_\omega \leq 1 & \forall_{n \in N}, \forall_{d \in W_d} \\
& \sum_{\omega \in \Omega} l_\omega x_\omega + u_n \geq min_n & \forall_{n \in N} \\
& \sum_{\omega \in \Omega} l_\omega x_\omega - o_n \leq max_n & \forall_{n \in N} \\
& x_\omega \in \{0,1\} & \forall_{\omega \in \Omega} \\
& z_p \in \mathbb{N} & \forall_{p \in P} \\
& o_n, u_n \geq 0 & \forall_{n \in N}
\end{aligned}
$$

Set partitioning model

Under-used time of the nurse

P : Patients
N : Nurses
Ω : Routes

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & \sum_{\omega\in\Omega} c_\omega x_\omega + C.\sum_{n\in N}(o_n + u_n) + U.\sum_{p\in P} z_p \\
\text{subject to} \quad & \sum_{\omega\in\Omega_d} a_{\omega,p} x_\omega \le 1 && \forall_{p\in P}, \forall_{d\in A_d} \\
& \sum_{\omega\in\Omega} a_{\omega,p} x_\omega + z_p = n_p && \forall_{p\in P} \\
& \sum_{\omega\in\Omega_d\cap\Omega_n} x_\omega \le 1 && \forall_{n\in N}, \forall_{d\in W_d} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega + u_n \ge min_n && \forall_{n\in N} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega - o_n \le max_n && \forall_{n\in N} \\
& x_\omega \in \{0,1\} && \forall_{\omega\in\Omega} \\
& z_p \in \mathbb{N} && \forall_{p\in P} \\
& o_n, u_n \ge 0 && \forall_{n\in N}
\end{aligned}
$$

Set partitioning model

Non-scheduled visits

P : Patients
N : Nurses
Ω : Routes

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & \sum_{\omega \in \Omega} c_\omega x_\omega + C.\sum_{n \in N} (o_n + u_n) + U.\sum_{p \in P} z_p \\
\text{subject to} \quad & \sum_{\omega \in \Omega_d} a_{\omega,p} x_\omega \leq 1 && \forall_{p \in P}, \forall_{d \in A_d} \\
& \sum_{\omega \in \Omega} a_{\omega,p} x_\omega + z_p = n_p && \forall_{p \in P} \\
& \sum_{\omega \in \Omega_d \cap \Omega_n} x_\omega \leq 1 && \forall_{n \in N}, \forall_{d \in W_d} \\
& \sum_{\omega \in \Omega} l_\omega x_\omega + u_n \geq min_n && \forall_{n \in N} \\
& \sum_{\omega \in \Omega} l_\omega x_\omega - o_n \leq max_n && \forall_{n \in N} \\
& x_\omega \in \{0,1\} && \forall_{\omega \in \Omega} \\
& z_p \in \mathbb{N} && \forall_{p \in P} \\
& o_n, u_n \geq 0 && \forall_{n \in N}
\end{aligned}
$$

Set partitioning model

P : Patients
N : Nurses
Ω : Routes

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & \sum_{\omega\in\Omega} c_\omega x_\omega + C.\sum_{n\in N}(o_n + u_n) + U.\sum_{p\in P} z_p \\
\text{subject to} \quad & \sum_{\omega\in\Omega_d} a_{\omega,p} x_\omega \le 1 && \forall_{p\in P}, \forall_{d\in A_d} \\
& \sum_{\omega\in\Omega} a_{\omega,p} x_\omega + z_p = n_p && \forall_{p\in P} \\
& \sum_{\omega\in\Omega_d\cap\Omega_n} x_\omega \le 1 && \forall_{n\in N}, \forall_{d\in W_d} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega + u_n \ge min_n && \forall_{n\in N} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega - o_n \le max_n && \forall_{n\in N} \\
& x_\omega \in \{0,1\} && \forall_{\omega\in\Omega} \\
& z_p \in \mathbb{N} && \forall_{p\in P} \\
& o_n, u_n \ge 0 && \forall_{n\in N}
\end{aligned}
$$

Max 1 visit per day

Set partitioning model

P : Patients
N : Nurses
Ω : Routes

$$\underset{x}{\text{minimize}} \quad \sum_{\omega \in \Omega} c_\omega x_\omega + C. \sum_{n \in N} (o_n + u_n) + U. \sum_{p \in P} z_p$$

$$\text{subject to} \quad \sum_{\omega \in \Omega_d} a_{\omega,p} x_\omega \leq 1 \qquad \forall_{p \in P}, \forall_{d \in A_d}$$

Max 1 visit per day

$$\sum_{\omega \in \Omega} a_{\omega,p} x_\omega + z_p = n_p \qquad \forall_{p \in P}$$

Nb visits per week

$$\sum_{\omega \in \Omega_d \cap \Omega_n} x_\omega \leq 1 \qquad \forall_{n \in N}, \forall_{d \in W_d}$$

$$\sum_{\omega \in \Omega} l_\omega x_\omega + u_n \geq min_n \qquad \forall_{n \in N}$$

$$\sum_{\omega \in \Omega} l_\omega x_\omega - o_n \leq max_n \qquad \forall_{n \in N}$$

$$x_\omega \in \{0,1\} \qquad \forall_{\omega \in \Omega}$$

$$z_p \in \mathbb{N} \qquad \forall_{p \in P}$$

$$o_n, u_n \geq 0 \qquad \forall_{n \in N}$$

Set partitioning model

P : Patients
N : Nurses
Ω : Routes

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & \sum_{\omega\in\Omega} c_\omega x_\omega + C.\sum_{n\in N}(o_n + u_n) + U.\sum_{p\in P} z_p \\
\text{subject to} \quad & \sum_{\omega\in\Omega_d} a_{\omega,p} x_\omega \le 1 && \forall_{p\in P}, \forall_{d\in A_d} \\
& \sum_{\omega\in\Omega} a_{\omega,p} x_\omega + z_p = n_p && \forall_{p\in P} \\
& \sum_{\omega\in\Omega_d\cap\Omega_n} x_\omega \le 1 && \forall_{n\in N}, \forall_{d\in W_d} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega + u_n \ge min_n && \forall_{n\in N} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega - o_n \le max_n && \forall_{n\in N} \\
& x_\omega \in \{0,1\} && \forall_{\omega\in\Omega} \\
& z_p \in \mathbb{N} && \forall_{p\in P} \\
& o_n, u_n \ge 0 && \forall_{n\in N}
\end{aligned}
$$

Max 1 visit per day

Nb visits per week

Route per day

Set partitioning model

P : Patients
N : Nurses
Ω : Routes

$$\underset{x}{\text{minimize}} \quad \sum_{\omega\in\Omega} c_\omega x_\omega + C.\sum_{n\in N}(o_n + u_n) + U.\sum_{p\in P} z_p$$

$$\text{subject to} \quad \sum_{\omega\in\Omega_d} a_{\omega,p} x_\omega \leq 1 \qquad \forall_{p\in P}, \forall_{d\in A_d}$$

Max 1 visit per day

$$\sum_{\omega\in\Omega} a_{\omega,p} x_\omega + z_p = n_p \qquad \forall_{p\in P}$$

Nb visits per week

$$\sum_{\omega\in\Omega_d\cap\Omega_n} x_\omega \leq 1 \qquad \forall_{n\in N}, \forall_{d\in W_d}$$

Route per day

$$\sum_{\omega\in\Omega} l_\omega x_\omega + u_n \geq min_n \qquad \forall_{n\in N}$$

Minimum worktime

$$\sum_{\omega\in\Omega} l_\omega x_\omega - o_n \leq max_n \qquad \forall_{n\in N}$$

Maximum worktime

$$x_\omega \in \{0,1\} \qquad \forall_{\omega\in\Omega}$$

$$z_p \in \mathbb{N} \qquad \forall_{p\in P}$$

$$o_n, u_n \geq 0 \qquad \forall_{n\in N}$$

Ways to solve the problem

- Find the routes in a reasonable computation time is complex, the possibilities are :

  - Solve a heuritistic Branch-And-Price using a column generation → Does not allow a current primal solution

  - Adapt a metaheuristic framework and add it some enhancements to make it the most efficient

Outline

- Problem Definition
- Mathematical Formulation
- Resolution Method
- Computation Results
- Conclusion

Methodology

- Our algorithm is based on 2 main components :
  - An ALNS-based framework
  - A heuristic concentration method

Adaptive Large Neighborhood Search

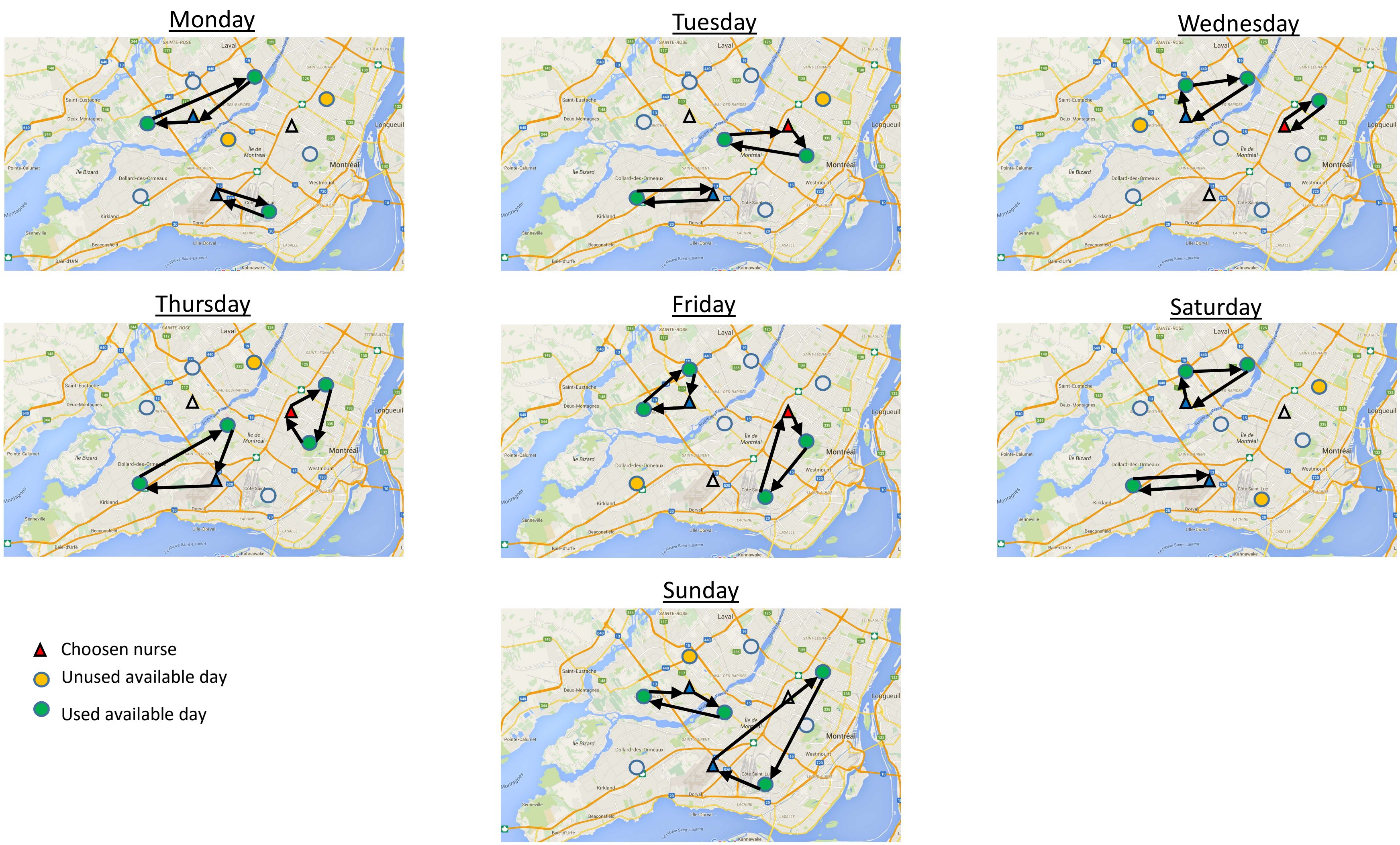
- ALNS: introduced by Ropke and Pisinger in 2006

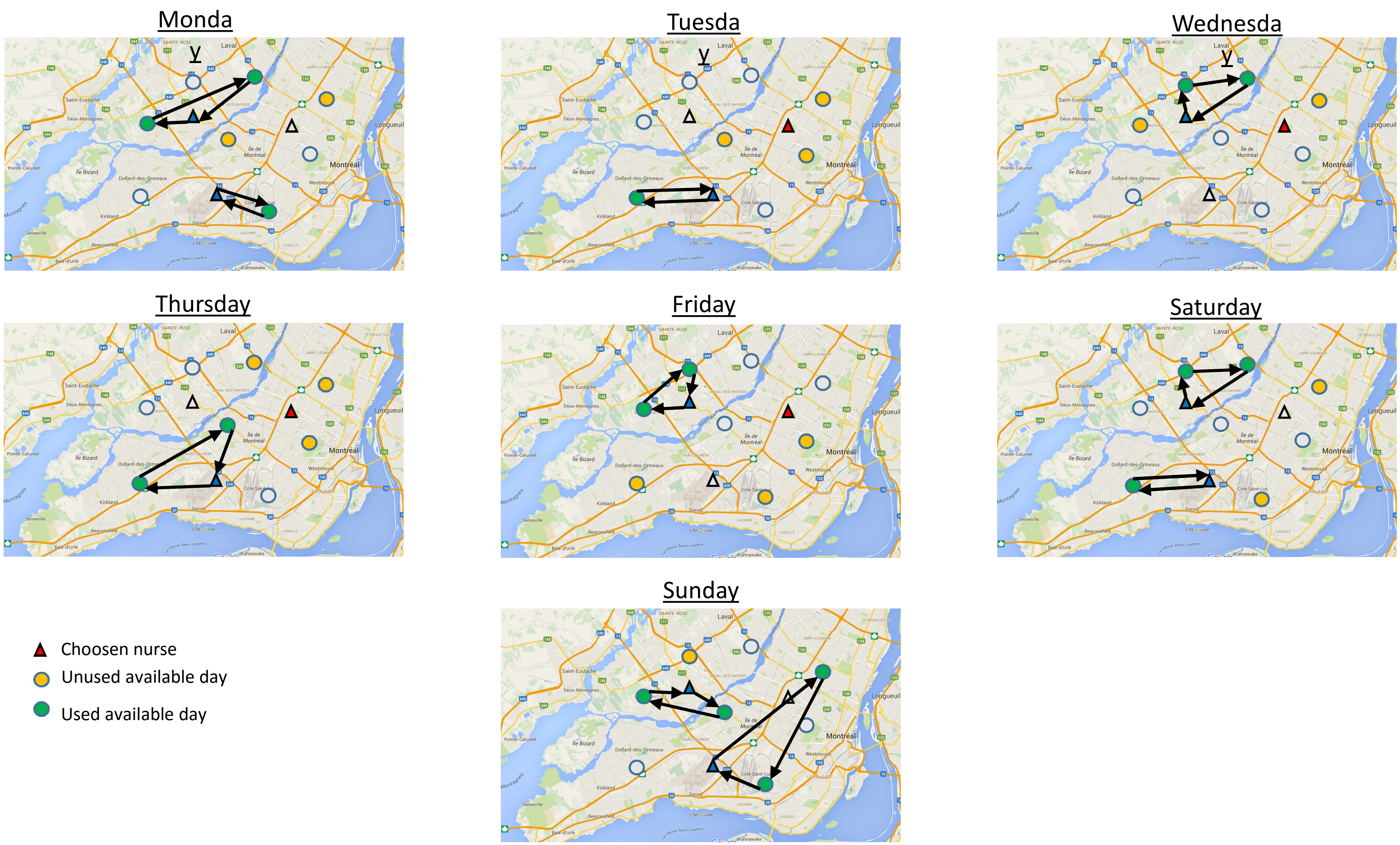
- Considers :
  - A large number of visits
  - A large set of constraints

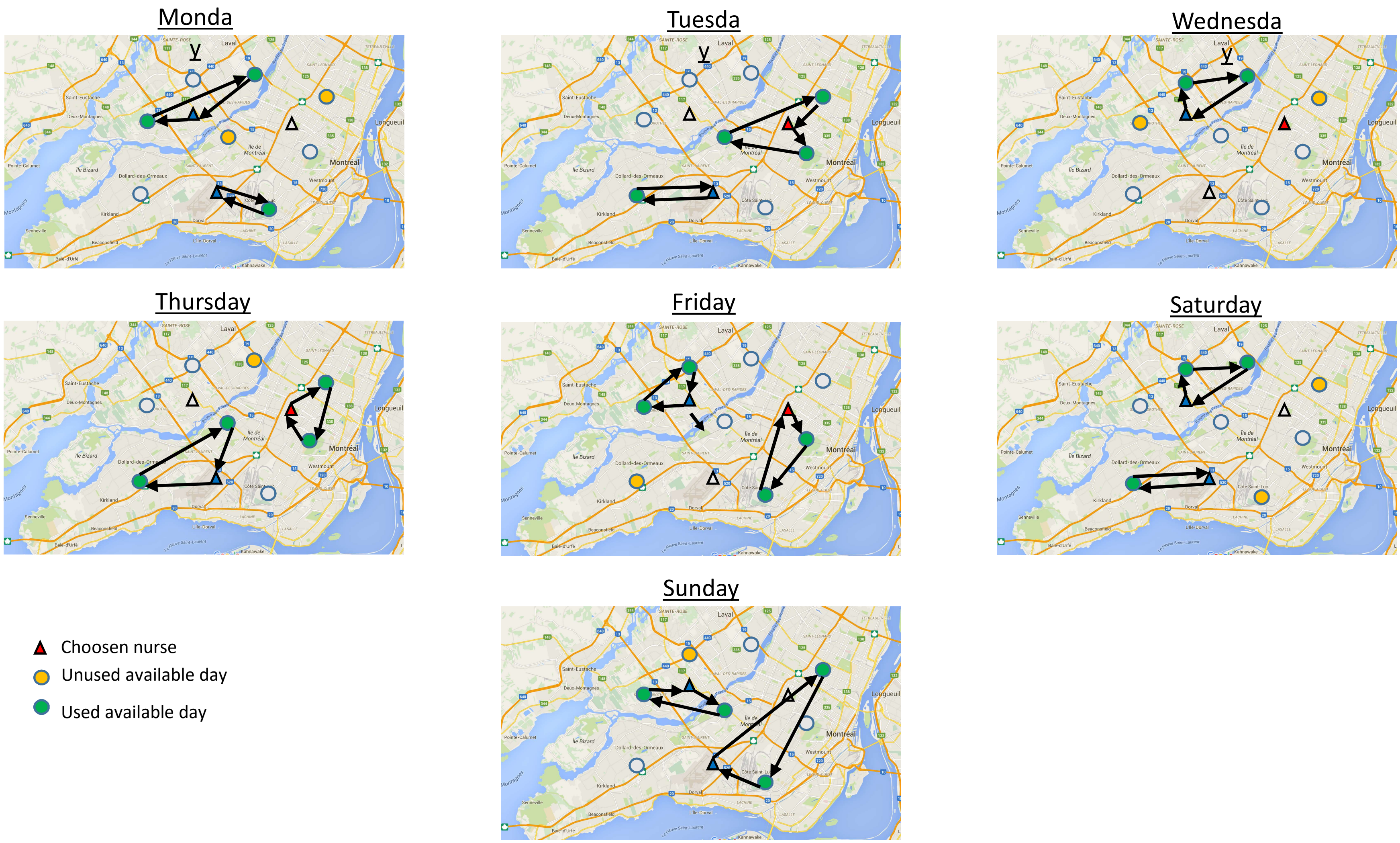
- Allows to test different operators associated with different strategies

Algorithm 1: LNS Heuristic.

Function LNS($s \in \{solutions\}, q \in \mathbb{N}$)
  solution $s_{best} = s$;
  repeat
    $s' = s$;
    remove q requests from s'
    reinsert removed requests into s';
    if $(f(s') < f(s_{best}))$ **then**
      $s_{best} = s'$;
    if accept(s', s) **then**
      $s = s'$;
  until stop-criterion met
return s_{best};


Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
Choosen nurse
Unused available day
Used available day

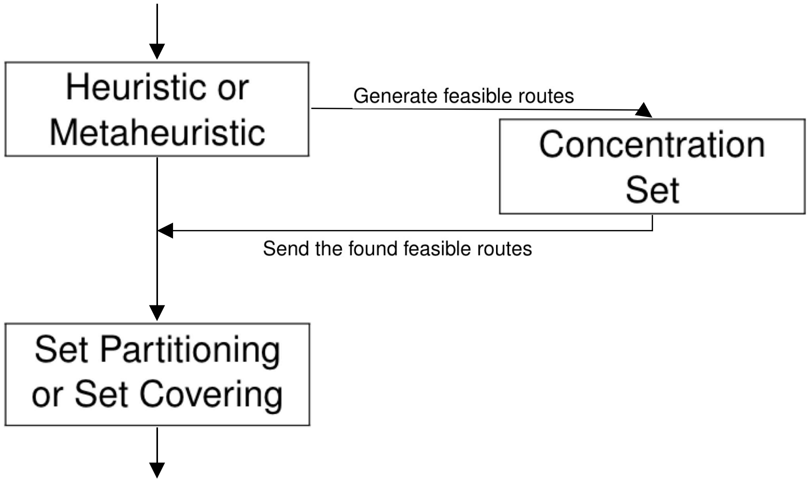

Monda
y
Tuesda
y
Wednesda
y
Thursday
Friday
Saturday
Sunday
Choosen nurse
Unused available day
Used available day


Monda
y
Tuesda
y
Wednesda
y
Thursday
Friday
Saturday
Sunday
Choosen nurse
Unused available day
Used available day
Laval
Montréal
Longueuil

Heuristic concentration

- The heuristic concentration principle has been proposed by Rosing et al. in 1996

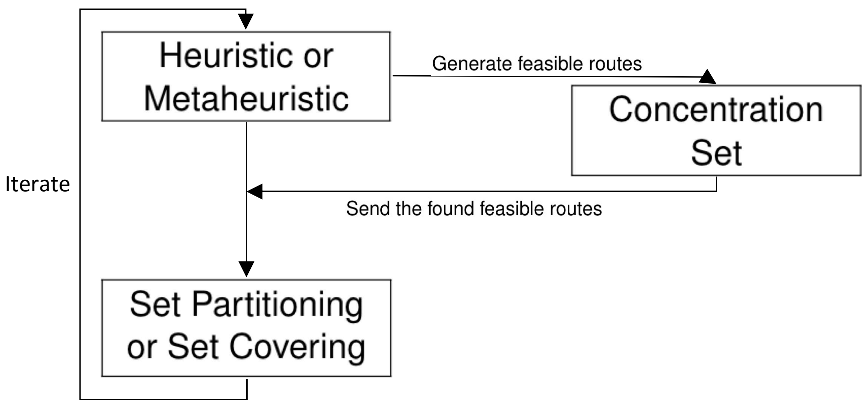
- The goal is to keep the generated feasible routes during the heuristic or metaheuristic then use these routes in the resolution of a set partitioning



Heuristic concentration

- Our version of the HC is close to the one developed by Subramanian et al. in 2013. They implemented an ILS-RVND + set part method

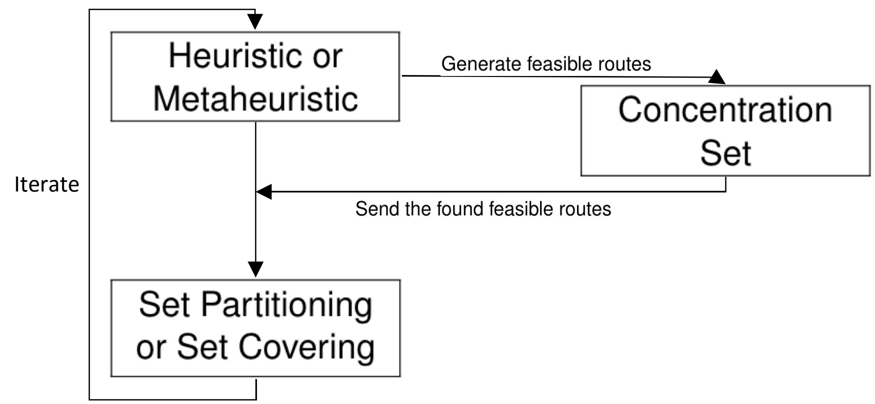
- They iteratively call the set partitioning to quickly guide the search to a good solution



Heuristic concentration

- Our version of the HC is close to the one developed by Subramanian et al. in 2013. They implemented an ILS-RVND + set part method

- They iteratively call the set partitioning to quickly guide the search to a good solution

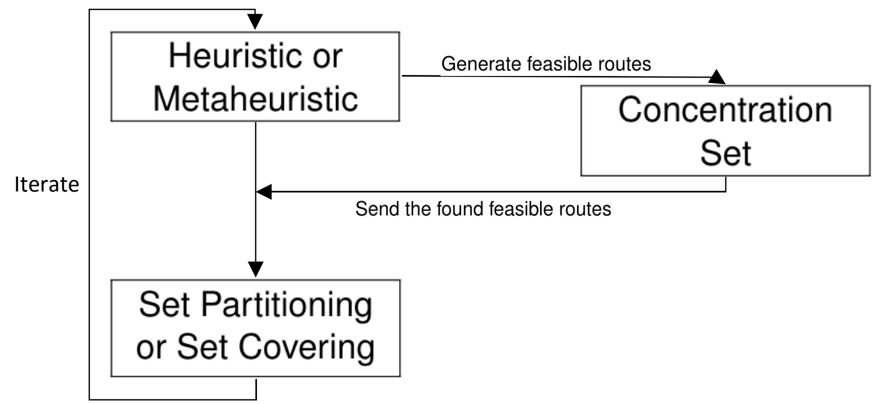


<u>PROBLEM</u> : Set partitioning in MIP = **Slow !**

Heuristic concentration

- Our version of the HC is close to the one developed by Subramanian et al. in 2013. They implemented an ILS-RVND + set part method

- They iteratively call the set partitioning to quickly guide the search to a good solution



<u>PROBLEM</u> : Set partitioning in MIP = **Slow !**

<u>SOLUTION</u> : **Relax it !**

Overview of the method

Find an initial solution ;

while *No termination criteria met* **do**

    $s \leftarrow currentSolution$;

    Select and apply a destroy operator on s ;

    Select and apply a repair operator on s ;

    Analyze the solution s ;

    if *A end of segment is met* **then**

        Do the relaxed HC method ;

        Apply the local search ;

        Reset the operators' scores ;

    end

end

Return the best solution found ;

Overview of the method

Find an initial solution ;

while *No termination criteria met* **do**

- $s \leftarrow currentSolution$;
- Select and apply a destroy operator on s ;
- Select and apply a repair operator on s ;
- Analyze the solution s ;
- **if** *A end of segment is met* **then**
  - Do the relaxed HC method ;
  - Apply the local search ;
  - Reset the operators' scores ;
- **end**

end

Return the best solution found ;

Find an initial solution heuristically

Overview of the method

```
Find an initial solution ;
while No termination criteria met do
    s ← currentSolution ;
    Select and apply a destroy operator on s ;
    Select and apply a repair operator on s ;
    Analyze the solution s ;
    if A end of segment is met then
        Do the relaxed HC method ;
        Apply the local search ;
        Reset the operators' scores ;
    end
end
Return the best solution found ;
```

Find an initial solution heuristically

Remove a subset of the visits

Overview of the method

Find an initial solution ;

while *No termination criteria met* **do**

  $s \leftarrow currentSolution$;

  Select and apply a destroy operator on s ;

  Select and apply a repair operator on s ;

  Analyze the solution s ;

  if *A end of segment is met* **then**

    Do the relaxed HC method ;

    Apply the local search ;

    Reset the operators' scores ;

  end

end

Return the best solution found ;

Find an initial solution heuristically

Remove a subset of the visits

Insert the non-scheduled visits

Overview of the method

Find an initial solution ;
while *No termination criteria met* **do**
  $s \leftarrow currentSolution$;
  Select and apply a destroy operator on s ;
  Select and apply a repair operator on s ;
  Analyze the solution s ;
  if *A end of segment is met* **then**
    Do the relaxed HC method ;
    Apply the local search ;
    Reset the operators' scores ;
  end
end
Return the best solution found ;

Find an initial solution heuristically

Remove a subset of the visits

Insert the non-scheduled visits

Update the best / current solutions

Overview of the method

```
Find an initial solution ;
while No termination criteria met do
    s ← currentSolution ;
    Select and apply a destroy operator on s ;
    Select and apply a repair operator on s ;
    Analyze the solution s ;
    if A end of segment is met then
        Do the relaxed HC method ;
        Apply the local search ;
        Reset the operators' scores ;
    end
end
Return the best solution found ;
```

Find an initial solution heuristically

Remove a subset of the visits

Insert the non-scheduled visits

Update the best / current solutions

Apply a heuristic concentration

Overview of the method

```
Find an initial solution ;
while No termination criteria met do
    s ← currentSolution ;
    Select and apply a destroy operator on s ;
    Select and apply a repair operator on s ;
    Analyze the solution s ;
    if A end of segment is met then
        Do the relaxed HC method ;
        Apply the local search ;
        Reset the operators' scores ;
    end
end
Return the best solution found ;
```

Find an initial solution heuristically

Remove a subset of the visits

Insert the non-scheduled visits

Update the best / current solutions

Apply a heuristic concentration

Apply a local search

Overview of the method

Find an initial solution ;
while *No termination criteria met* **do**
  $s \leftarrow currentSolution$;
  Select and apply a destroy operator on s ;
  Select and apply a repair operator on s ;
  Analyze the solution s ;
  if *A end of segment is met* **then**
    Do the relaxed HC method ;
    Apply the local search ;
    Reset the operators' scores ;
  end
end
Return the best solution found ;

Find an initial solution heuristically

Remove a subset of the visits

Insert the non-scheduled visits

Update the best / current solutions

Apply a heuristic concentration

Apply a local search

Update the operators' scores

Overview of the method

```
Find an initial solution ;
while No termination criteria met do
    s ← currentSolution ;
    Select and apply a destroy operator on s ;
    Select and apply a repair operator on s ;
    Analyze the solution s ;
    if A end of segment is met then
        Do the relaxed HC method ;
        Apply the local search ;
        Reset the operators' scores ;
    end
end
Return the best solution found ;
```

- *Find an initial solution heuristically*
- *Remove a subset of the visits*
- *Insert the non-scheduled visits*
- *Update the best / current solutions*
- *Apply a heuristic concentration*
- *Apply a local search*
- *Update the operators' scores*

Overview of the method

Find an initial solution ;
while *No termination criteria met* **do**
  $s \leftarrow currentSolution$;
  Select and apply a destroy operator on s ;
  Select and apply a repair operator on s ;
  Analyze the solution s ;
  if *A end of segment is met* **then**
    Do the relaxed HC method ;
    Apply the local search ;
    Reset the operators' scores ;
  end
end
Return the best solution found ;

Find an initial solution heuristically

Remove a subset of the visits

Insert the non-scheduled visits

Update the best / current solutions

Apply a heuristic concentration

Apply a local search

Update the operators' scores

Relaxed heuristic concentration

$$
\begin{array}{lll}
\underset{x}{\text{minimize}} & \sum_{\omega\in\Omega} c_\omega x_\omega + C.\sum_{n\in N}(o_n+u_n) + U.\sum_{p\in P} z_p & \\
\text{subject to} & \sum_{\omega\in\Omega_d} a_{\omega,p} x_\omega \le 1 & \forall_{p\in P}, \forall_{d\in A_d} \\
& \sum_{\omega\in\Omega} a_{\omega,p} x_\omega + z_p = n_p & \forall_{p\in P} \\
& \sum_{\omega\in\Omega_d\cap\Omega_n} x_\omega \le 1 & \forall_{n\in N}, \forall_{d\in W_d} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega + u_n \ge min_n & \forall_{n\in N} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega - o_n \le max_n & \forall_{n\in N} \\
& x_\omega \in \{0,1\} & \forall_{\omega\in\Omega} \\
& z_p \in \mathbb{N} & \forall_{p\in P} \\
& o_n, u_n \ge 0 & \forall_{n\in N}
\end{array}
$$

Relaxed heuristic concentration

$$
\begin{array}{lll}
\underset{x}{\text{minimize}} & \displaystyle\sum_{\omega\in\Omega} c_\omega x_\omega + C.\sum_{n\in N}(o_n+u_n) + U.\sum_{p\in P} z_p & \\
\text{subject to} & \displaystyle\sum_{\omega\in\Omega_d} a_{\omega,p} x_\omega \le 1 & \forall_{p\in P}, \forall_{d\in A_d} \\
& \displaystyle\sum_{\omega\in\Omega} a_{\omega,p} x_\omega + z_p = n_p & \forall_{p\in P} \\
& \displaystyle\sum_{\omega\in\Omega_d\cap\Omega_n} x_\omega \le 1 & \forall_{n\in N}, \forall_{d\in W_d} \\
& \displaystyle\sum_{\omega\in\Omega} l_\omega x_\omega + u_n \ge min_n & \forall_{n\in N} \\
& \displaystyle\sum_{\omega\in\Omega} l_\omega x_\omega - o_n \le max_n & \forall_{n\in N} \\
& x_\omega \in \{0,1\} & \forall_{\omega\in\Omega} \\
& z_p \in \mathbb{N} & \forall_{p\in P} \\
& o_n, u_n \ge 0 & \forall_{n\in N}
\end{array}
$$

$$
\begin{array}{lll}
\underset{x}{\text{minimize}} & \displaystyle\sum_{\omega\in\Omega} c_\omega x_\omega + C.\sum_{n\in N}(o_n+u_n) + U.\sum_{p\in P} z_p & \\
\text{subject to} & \displaystyle\sum_{\omega\in\Omega_d} a_{\omega,p} x_\omega \le 1 & \forall_{p\in P}, \forall_{d\in A_d} \\
& \displaystyle\sum_{\omega\in\Omega} a_{\omega,p} x_\omega + z_p = n_p & \forall_{p\in P} \\
& \displaystyle\sum_{\omega\in\Omega_d\cap\Omega_n} x_\omega \le 1 & \forall_{n\in N}, \forall_{d\in W_d} \\
& \displaystyle\sum_{\omega\in\Omega} l_\omega x_\omega + u_n \ge min_n & \forall_{n\in N} \\
& \displaystyle\sum_{\omega\in\Omega} l_\omega x_\omega - o_n \le max_n & \forall_{n\in N} \\
& \boxed{\begin{array}{l} x_\omega \in [0,1] \\ z_p \ge 0 \\ o_n, u_n \ge 0 \end{array}} & \begin{array}{l} \forall_{\omega\in\Omega} \\ \forall_{p\in P} \\ \forall_{n\in N} \end{array}
\end{array}
$$

Relaxed heuristic concentration

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & \sum_{\omega\in\Omega} c_\omega x_\omega + C.\sum_{n\in N}(o_n+u_n) + U.\sum_{p\in P} z_p \\
\text{subject to} \quad & \sum_{\omega\in\Omega_d} a_{\omega,p} x_\omega \le 1 && \forall_{p\in P}, \forall_{d\in A_d} \\
& \sum_{\omega\in\Omega} a_{\omega,p} x_\omega + z_p = n_p && \forall_{p\in P} \\
& \sum_{\omega\in\Omega_d\cap\Omega_n} x_\omega \le 1 && \forall_{n\in N}, \forall_{d\in W_d} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega + u_n \ge min_n && \forall_{n\in N} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega - o_n \le max_n && \forall_{n\in N} \\
& x_\omega \in \{0,1\} && \forall_{\omega\in\Omega} \\
& z_p \in \mathbb{N} && \forall_{p\in P} \\
& o_n, u_n \ge 0 && \forall_{n\in N}
\end{aligned}
$$

$$\Downarrow$$

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & \sum_{\omega\in\Omega} c_\omega x_\omega + C.\sum_{n\in N}(o_n+u_n) + U.\sum_{p\in P} z_p \\
\text{subject to} \quad & \sum_{\omega\in\Omega_d} a_{\omega,p} x_\omega \le 1 && \forall_{p\in P}, \forall_{d\in A_d} \\
& \sum_{\omega\in\Omega} a_{\omega,p} x_\omega + z_p = n_p && \forall_{p\in P} \\
& \sum_{\omega\in\Omega_d\cap\Omega_n} x_\omega \le 1 && \forall_{n\in N}, \forall_{d\in W_d} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega + u_n \ge min_n && \forall_{n\in N} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega - o_n \le max_n && \forall_{n\in N} \\
& \boxed{x_\omega \in [0,1]} && \forall_{\omega\in\Omega} \\
& \boxed{z_p \ge 0} && \forall_{p\in P} \\
& \boxed{o_n, u_n \ge 0} && \forall_{n\in N}
\end{aligned}
$$

We then call a constructive heuristic based on the LP solution

```
L : list of the route sorted by decreasing order of x_ω
forall route ω in L do
    forall visit v in ω do
        if the schedule of the visit is possible then
            schedule the visit in the route ω ;
        end
    end
end
```

Heuristic Concentration

Best Solution
Route 1
Route 2
Route 3

Concentration Set
Route 1
Route 2
Route 3

Iteration : 0

Heuristic Concentration

Best Solution
Route 1
Route 4
Route 3

Concentration Set
Route 1
Route 2
Route 3
Route 4
Route 5

Iteration : 145

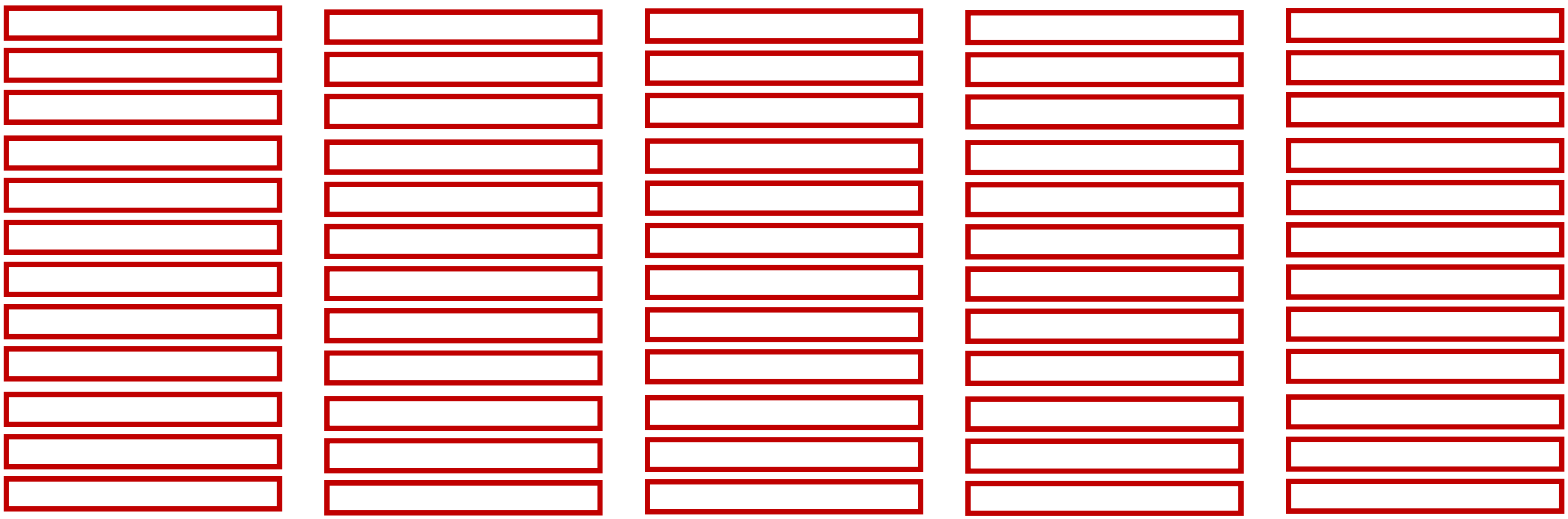
Heuristic Concentration

Best Solution	Concentration Set
Route 7	*Route 1*
Route 4	*Route 2*
Route 12	*Route 3*
	Route 4
	Route 5
	Route 6
	Route 7
	Route 8
	Route 9
	Route 10
	Route 11
	Route 12

Iteration : 319

Heuristic Concentration

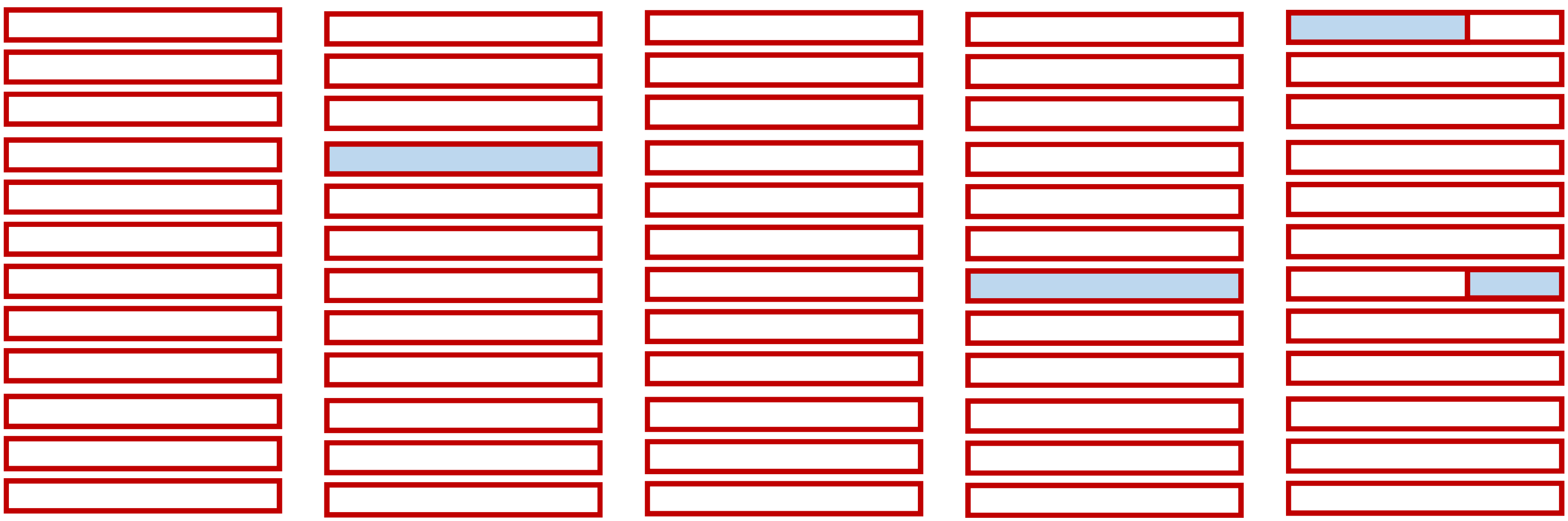
Concentration Set



Iteration : 1000 → Solve the relaxed set partitioning

Heuristic Concentration

Concentration Set



Relaxed set partitioning solution

Heuristic Concentration

New Solution

Heuristic Concentration Selection

Route 11

Route 32

Route 45

Route 75

Iteration : 1000

Heuristic Concentration

New Solution	Heuristic Concentration Selection
Route 11	*Route 11*
Route 32	*Route 32*
	Route 45
	Route 75

Iteration : 1000

Heuristic Concentration

New Solution	
Route 11	
Route 32	
Route 45	?

Heuristic Concentration Selection
Route 11
Route 32
Route 45
Route 75

Iteration : 1000

Heuristic Concentration

New Solution

Route 11

Route 32

Route 45

Heuristic Concentration Selection

Route 11

Route 32

Route 45

Route 75

→ And we analyse the new solution

Iteration : 1000

Overview of the method

```
Find an initial solution ;
while No termination criteria met do
    s ← currentSolution ;
    Select and apply a destroy operator on s ;
    Select and apply a repair operator on s ;
    Analyze the solution s ;
    if A end of segment is met then
        Do the relaxed HC method ;
        Apply the local search ;
        Reset the operators' scores ;
    end
end
Return the best solution found ;
```

Find an initial solution heuristically

Remove a subset of the visits

Insert the non-scheduled visits

Update the best / current solutions

Apply a heuristic concentration

Apply a local search

Update the operators' scores

Classic ALNS operators

Classic Destroy operators :

Worst removal → Visits which cost the most

Classic ALNS operators

Classic Destroy operators :

Worst removal → Visits which cost the most

Random Removal → Randomly select q visits

Classic ALNS operators

Classic Destroy operators :

- Worst removal → Visits which cost the most
- Random Removal → Randomly select q visits
- Related removal → Randomly select a visit and remove it and the q-1 most related

Classic ALNS operators

Classic Destroy operators :

- Worst removal → Visits which cost the most
- Random Removal → Randomly select q visits
- Related removal → Randomly select a visit and remove it and the q-1 most related

Classic Repair operators :

- Greedy heuristic → Scheduled at lowest cost

Classic ALNS operators

Classic Destroy operators :

- Worst removal → Visits which cost the most
- Random Removal → Randomly select q visits
- Related removal → Randomly select a visit and remove it and the q-1 most related

Classic Repair operators :

- Greedy heuristic → Scheduled at lowest cost
- Regret-2/Regret-3 → Take into account the regret after insertion

New Operators

New Destroy operators :

Random Patient → Randomly select a patient and remove all his visits

New Operators

New Destroy operators :

Random Patient → Randomly select a patient and remove all his visits

Flexible patient → Remove the most flexible : Nb_available / Nb_visits

New Operators

New Destroy operators :

Random Patient → Randomly select a patient and remove all his visits

Flexible patient → Remove the most flexible : Nb_available / Nb_visits

New Repair operators :

Random Patient → Randomly select a patient and schedule all his visits

New operators

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & \sum_{\omega\in\Omega} c_\omega x_\omega + C.\sum_{n\in N}(o_n+u_n) + U.\sum_{p\in P} z_p \\
\text{subject to} \quad & \sum_{\omega\in\Omega_d} a_{\omega,p} x_\omega \le 1 && \forall_{p\in P}, \forall_{d\in A_d} \\
& \boxed{\sum_{\omega\in\Omega} a_{\omega,p} x_\omega + z_p = n_p} && \forall_{p\in P} \\
& \sum_{\omega\in\Omega_d\cap\Omega_n} x_\omega \le 1 && \forall_{n\in N}, \forall_{d\in W_d} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega + u_n \ge min_n && \forall_{n\in N} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega - o_n \le max_n && \forall_{n\in N} \\
& x_\omega \in [0,1] && \forall_{\omega\in\Omega} \\
& z_p \ge 0 && \forall_{p\in P} \\
& o_n, u_n \ge 0 && \forall_{n\in N}
\end{aligned}
$$

New operators

$$
\begin{array}{lll}
\underset{x}{\text{minimize}} & \sum_{\omega \in \Omega} c_\omega x_\omega + C. \sum_{n \in N} (o_n + u_n) + U. \sum_{p \in P} z_p & \\
\text{subject to} & \sum_{\omega \in \Omega_d} a_{\omega,p} x_\omega \leq 1 & \forall_{p \in P}, \forall_{d \in A_d} \\
& \boxed{\sum_{\omega \in \Omega} a_{\omega,p} x_\omega + z_p = n_p} & \forall_{p \in P} \\
& \sum_{\omega \in \Omega_d \cap \Omega_n} x_\omega \leq 1 & \forall_{n \in N}, \forall_{d \in W_d} \\
& \sum_{\omega \in \Omega} l_\omega x_\omega + u_n \geq min_n & \forall_{n \in N} \\
& \sum_{\omega \in \Omega} l_\omega x_\omega - o_n \leq max_n & \forall_{n \in N} \\
& x_\omega \in [0, 1] & \forall_{\omega \in \Omega} \\
& z_p \geq 0 & \forall_{p \in P} \\
& o_n, u_n \geq 0 & \forall_{n \in N}
\end{array}
$$

New operators

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & \sum_{\omega\in\Omega} c_\omega x_\omega + C. \sum_{n\in N}(o_n + u_n) + U. \sum_{p\in P} z_p & \\
\text{subject to} \quad & \sum_{\omega\in\Omega_d} a_{\omega,p} x_\omega \leq 1 & \forall_{p\in P}, \forall_{d\in A_d} \\
& \boxed{\sum_{\omega\in\Omega} a_{\omega,p} x_\omega + z_p = n_p} & \forall_{p\in P} \\
& \sum_{\omega\in\Omega_d\cap\Omega_n} x_\omega \leq 1 & \forall_{n\in N}, \forall_{d\in W_d} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega + u_n \geq min_n & \forall_{n\in N} \\
& \sum_{\omega\in\Omega} l_\omega x_\omega - o_n \leq max_n & \forall_{n\in N} \\
& x_\omega \in [0,1] & \forall_{\omega\in\Omega} \\
& z_p \geq 0 & \forall_{p\in P} \\
& o_n, u_n \geq 0 & \forall_{n\in N}
\end{aligned}
$$

Dual power

Focus on the highest dual values !

New Operators

New Destroy operators :

Random Patient → Randomly select a patient and remove all his visits
Flexible patient → Remove the most flexible : Nb_available / Nb_visits
Dual Patient → Remove the patients with **the lowest dual value**

New Repair operators :

Random Patient → Randomly select a patient and schedule all his visits
Dual Patient → Prioritize the patient with the highest dual values

Outline

- Problem Definition
- Mathematical Formulation
- Resolution Method
- Computation Results
- Conclusion

Instances generation

- We have generated 3 sets of 20 pseudo-instances

Instance	Patient	Visits	Nurse	Workdays
Small	40	120	5	25
Medium	80	225	10	45
Large	150	430	20	90

Table 1: Instances' characteristics

- The algorithm is implemented in C++, the set partitioning calls Cplex and each instance runs during 10 minutes / 10^5 iterations

Experiments: Impact of the new operators

	Classic	All	
		Gap	CPU
Small	512169,0577	**-9,38%**	<4 min
Medium	613572,3348	**-6,48%**	10 min
Large	799746,4565	**-8,19%**	10 min
Mean		**-8,01%**	

Table 1: Evolution of the costs with the new operators

Experiments: Impact of the set partitioning

	Classic	All		All + Set Part	
		Gap	CPU	Gap	CPU
Small	512169,0577	-9,38%	<4 min	**-15,29%**	<6 min
Medium	613572,3348	-6,48%	10 min	**-18,32%**	10 min
Large	799746,4565	-8,19%	10 min	**-18,70%**	10 min
Mean		-8,01%		**-17,44%**	

Table 2: Evolution of the costs with the set partitioning

Experiments: Impact of the dual operators

	Classic	All		All + Set Part		All + SP + Dual	
		Gap	CPU	Gap	CPU	Gap	CPU
Small	512169,0577	-9,38%	<4 min	-15,29%	<6 min	-15,28%	<6 min
Medium	613572,3348	-6,48%	10 min	-18,32%	10 min	-18,29%	10 min
Large	799746,4565	-8,19%	10 min	-18,70%	10 min	**-20,53%**	10 min
Mean		-8,01%		-17,44%		**-18,03%**	

Table 3: Evolution of the costs with the dual operators

Analysis of the operators

$$\boxed{\text{Can we remove some useless operators ?}}$$

Analysis of the operators

$\boxed{\text{Can we remove some useless operators ?}}$

Goal : Keep the top-3 destroy and repair operators

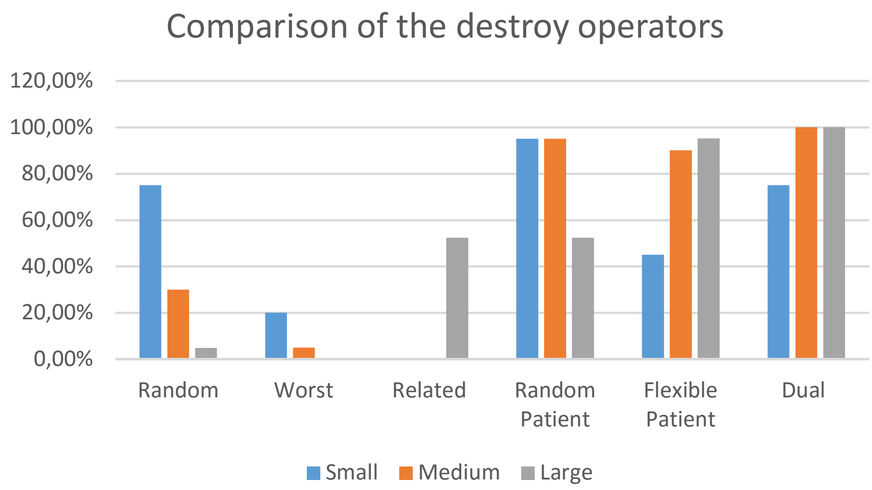
Analysis of the operators

$\boxed{\text{Can we remove some useless operators ?}}$

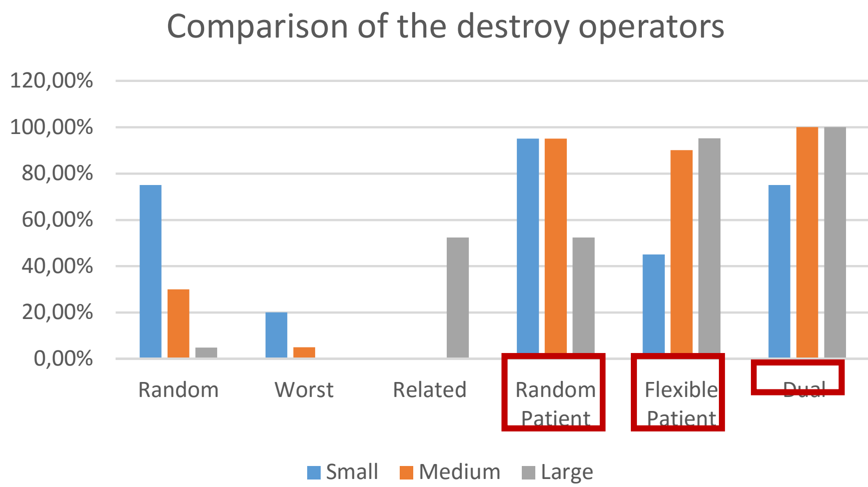
Goal : Keep the top-3 destroy and repair operators

Idea : Keep the operators which are the less often rejected at the end of the iteration

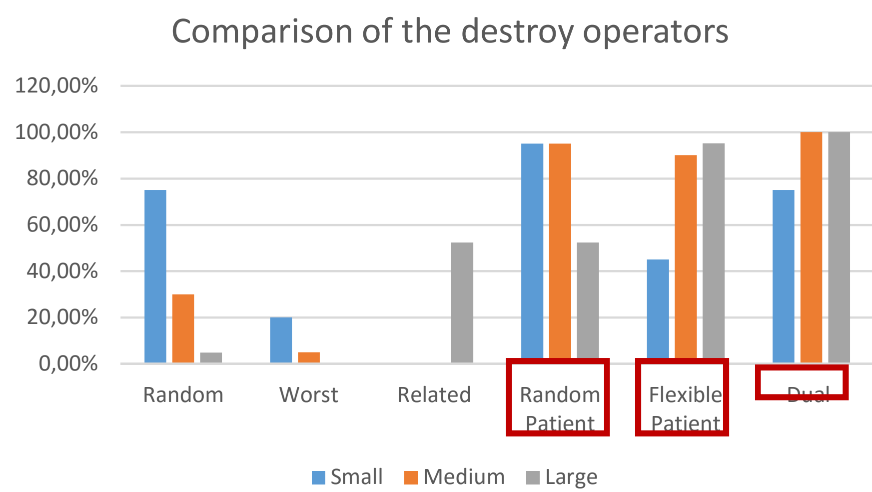
Analysis of the operators

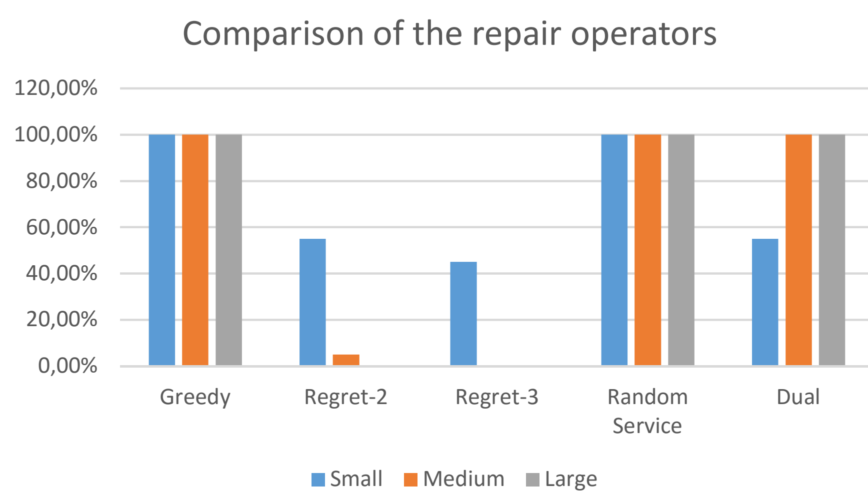


Analysis of the operators

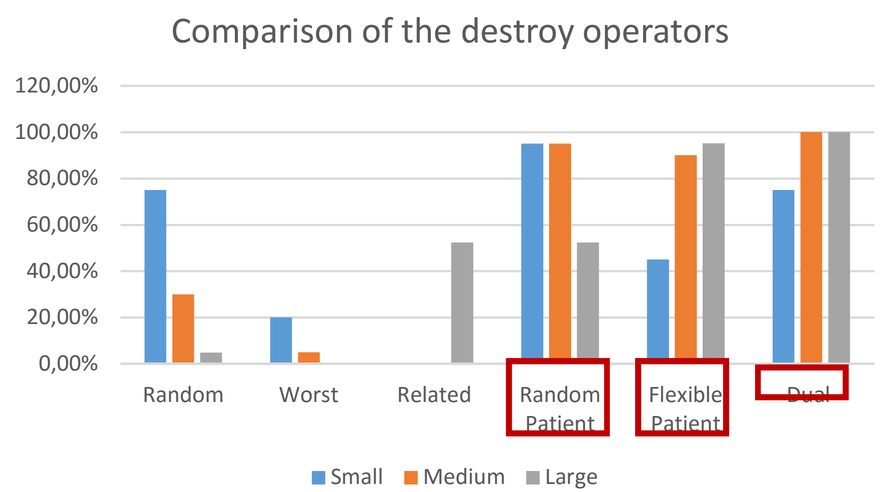


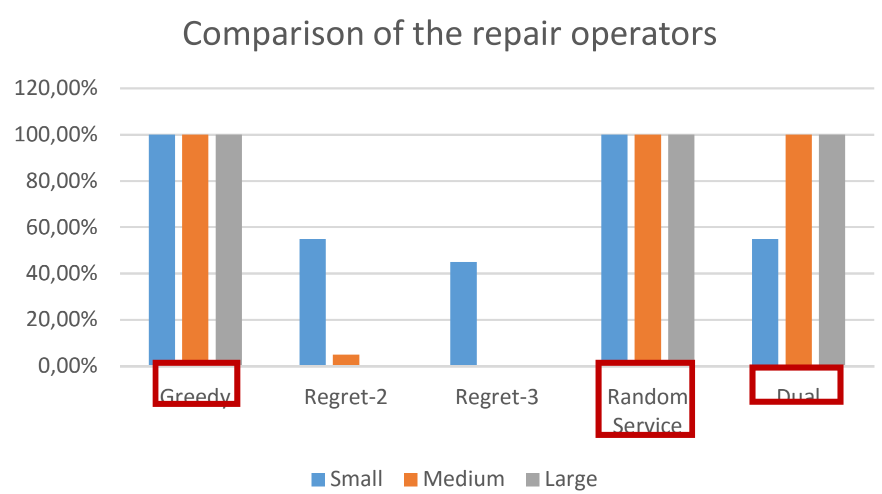
Analysis of the operators





Analysis of the operators





Experiments: Selection of the best operators

	Classic	All		All + Set Part		All + SP + Dual		Selected	
		Gap	CPU	Gap	CPU	Gap	CPU	Gap	CPU
Small	512169,0577	-9,38%	<4 min	-15,29%	<6 min	-15,28%	<6 min	-14,86%	<4 min
Medium	613572,3348	-6,48%	10 min	-18,32%	10 min	-18,29%	10 min	**-18,86%**	10 min
Large	799746,4565	-8,19%	10 min	-18,70%	10 min	-20,53%	10 min	**-20,92%**	10 min
Mean		-8,01%		-17,44%		-18,03%		**-18,22%**	

Table 4: Evolution of the costs with the selected operators

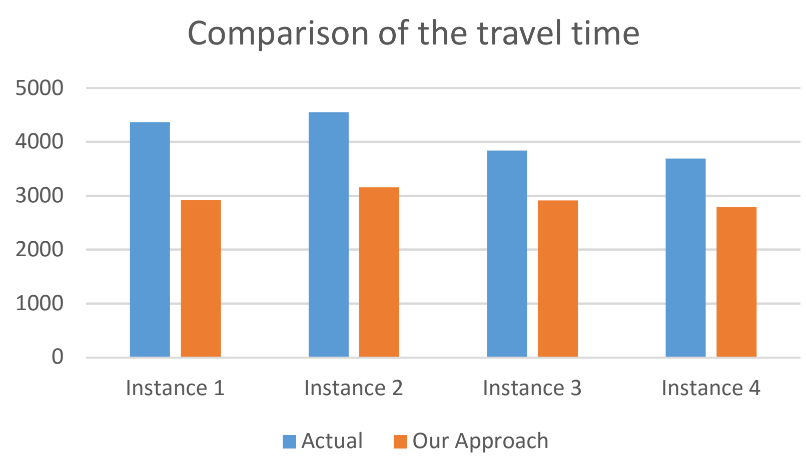
Real instances

We have taken 4 real instances corresponding to 1 week of work

Name	Patient	Visit	Nurse	Workday
Instance 1	149	325	11	40
Instance 2	137	340	11	40
Instance 3	145	311	11	35
Instance 4	146	324	11	40

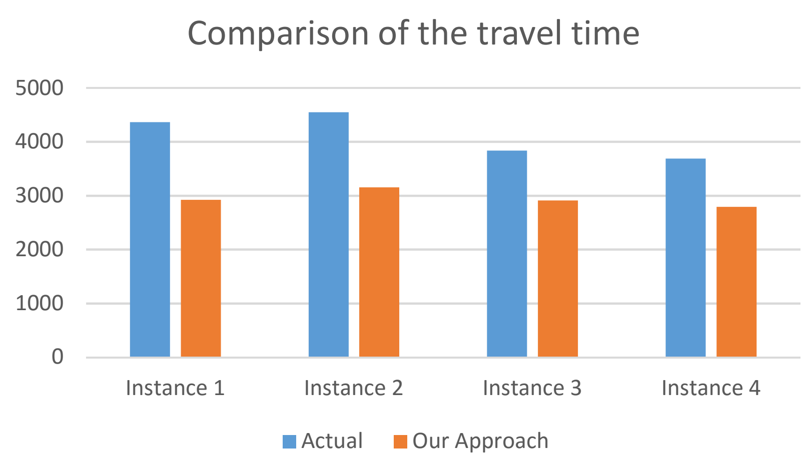
Table 5: Real instances

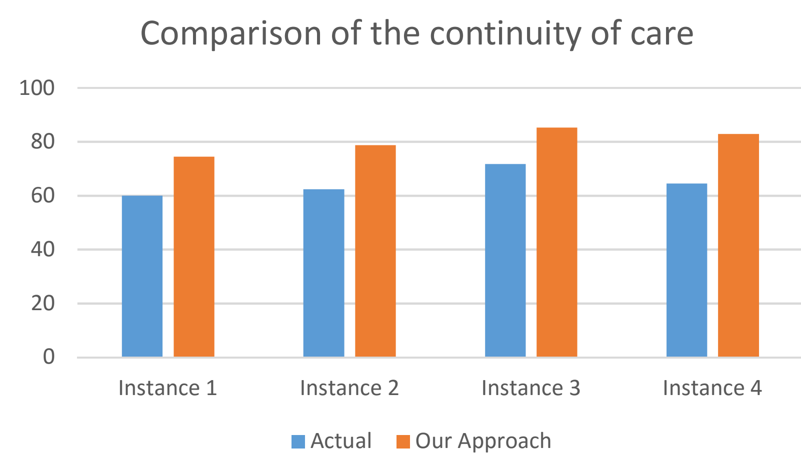
Real instances' results



Reduction of the travel time by **28,31%** in comparaison with the actual solution

Real instances' results



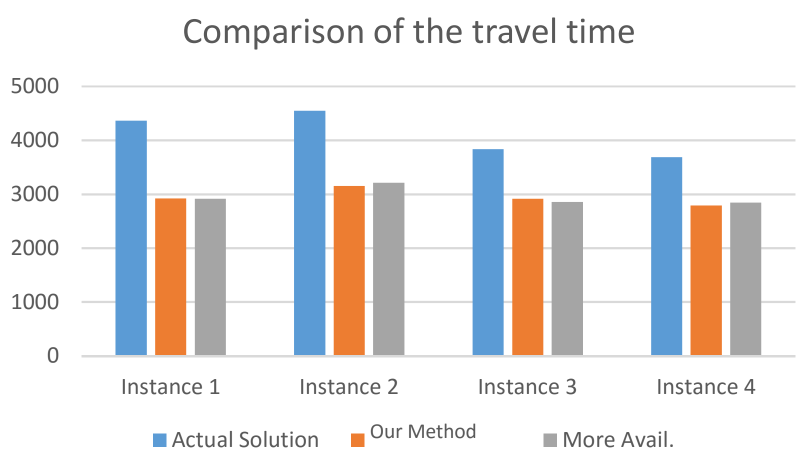


Reduction of the travel time by **28,31%** in comparaison with the actual solution

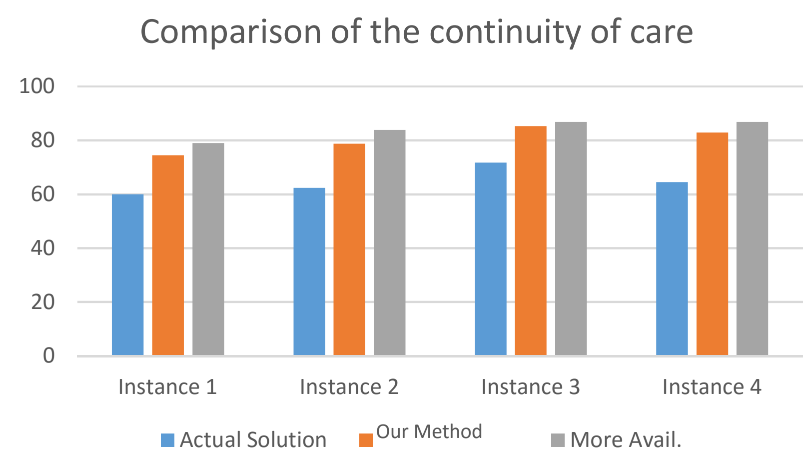
Increase of the fidelity by **15,70%** in comparaison with the actual solution

Real instances’ results

+ 1 available day for 40% of the patients



Reduction of the travel time by **28,03%** in comparaison with the actual solution



Increase of the fidelity by **19,44%** in comparaison with the actual solution