Improved Branch-Cut-and-Price for Capacitated Vehicle Routing

Diego Pecin [1]
Artur Pessoa [2]
Marcus Poggi [1]
Eduardo Uchoa [2]

PUC - Rio de Janeiro [1]
Universidade Federal Fluminense [2]

January, 2014

The Capacitated Vehicle Routing Problem (CVRP)

Instance: Complete graph $G = (V, E)$ with $V = \{0, \ldots, n\}$; vertex 0 is the *depot*, $V_+ = \{1, \ldots, n\}$ is the set of *customers*. Each edge $e \in E$ has a *cost* c_e. Each customer $i \in V_+$ has a *demand* d_i. There is a fleet of K identical *vehicles* with *capacity* Q.

Solution: A set of K routes starting and ending at the depot, attending all customers, and respecting the capacities, with minimal total cost.

- The most classical VRP variant, proposed by Dantzig and Ramser in 1959.

CVRP and Column Generation

Since the 1980's, CG was applied with success on VRPTW instances with narrow time windows.

CVRP is a VRPTW with large time windows $\Rightarrow$ CG was not viewed as a promising approach.

Indeed, pure CG performs poorly on the CVRP.

CVRP and Cut Generation

A lot of polyhedral investigation (inspired by the TSP) was done. Known families of cuts include:

- Rounded Capacity
- Framed Capacities
- Strengthened Combs
- Multistars
- Extended Hypotours

CVRP and Cut Generation

Branch-and-cut became the dominant approach for the CVRP in the 1990s and early 2000s:

- Araque, Kudva, Morin, and Pekny [1994]
- Augerat, Belenguer, Benavent, Corberán, Naddef, and Rinaldi [1995]
- Blasum and Hochstättler [2000]
- Ralphs, Kopman, Pulleyblank, and Trotter Jr. [2003]
- Achuthan, Caccetta, and Hill [2003]
- Lysgaard, Letchford, and Eglese [2004]

CVRP and Cut Generation

The branch-and-cut algorithms may perform well on instances with few vehicles (because they are closer to the TSP?).

Some instances with only 50 customers could not be solved.

Combining Column and Cut Generation

Fukasawa et al. [2006] proposed a Branch-Cut-and-Price (BCP) for the CVRP, solving all the instances from the literature with up to 134 customers.

Since then, the best performing algorithms are based on the combination of column and cut generation.

R. Fukasawa, H. Longo, J. Lysgaard, M. Poggi de Aragão, M. Reis, E. Uchoa, and R.F. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical programming*, 106(3): 491–511, 2006

Recent Literature Review: Fukasawa et al. [2006]

BCP algorithm:

- Columns are associated to q-**routes without** k-**cycles**, a relaxation that allows multiple visits to a customer if at least k other customers are visited in-between.
- Cuts over the edge formulation, the same used in Lysgaard et al. [2004]. Those cuts are **robust** with respect to q-route pricing, not affecting its complexity.
- May automatically switch to pure branch-and-cut if it detects advantage.

Recent Literature Review: Baldacci et al. [2008]

Column-and-cut generation algorithm:

- Columns are associated to elementary routes.
- Also uses **non-robust** Strengthened Capacity and Clique Cuts. Reduce the gaps significantly, but each cut makes the pricing harder.

R. Baldacci, N. Christofides, and A. Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115(2):351–385, 2008

Recent Literature Review: Baldacci et al. [2008]

- Instead of branching, the algorithm finishes by **enumerating** all routes with reduced cost smaller than the gap. A Set Partitioning with those routes is given to a MIP solver.

Solved almost all the instances already solved in the literature, usually taking much less time. A few instances with many customers per vehicle were not solved.

Recent Literature Review: Baldacci et al. [2011]

Improvement over Baldacci et al. [2008]:

- *ng*-**routes**, a new relaxation that is more effective than *q*-routes is introduced.
- Subset Row Cuts (Jepsen et al. [2008]) are used instead of Clique Cuts, less impact on the pricing.

Faster and much more stable, could also solve instances with reasonably many customers per route.

R. Baldacci, A. Mingozzi, and R. Roberti. New route relaxation and pricing strategies for the vehicle routing problem. *Operations Research*, 59(5):1269–1283, 2011

Recent Literature Review: Contardo [2012]

New twists on the use of non-robust cuts and on enumeration:

- The partial elementarity of the routes is enforced by non-robust cuts.
- The enumeration generates a **pool** with up to a few million routes and the **pricing** starts to be performed **by inspection**. At that point, an aggressive separation of non-robust cuts and fixing by reduced costs can lead to small gaps.

Instance M-n151-k12 was solved to optimality in 5.5 hours, setting a new record.

C. Contardo. A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints. Technical report, Archipel-UQAM 5078, Université du Québec à Montréal, Canada, 2012

Recent Literature Review: Røpke [2012]

Back to robust BCP.

- Instead of q-routes without k-cycles, the more effective ng-routes are used.
- A sophisticated and aggressive **strong branching**, greatly reducing the size of the enumeration trees.

In spite of larger root gaps, results comparable with Contardo [2012] and Baldacci et al. [2011].

- M-n151-k12 solved in 5 days.
- Similar algorithm for VRPTW solved all Solomon instances with 100 customers.

S. Røpke. Branching decisions in branch-and-cut-and-price algorithms for vehicle routing problems. *Presentation in Column Generation 2012*, 2012

Overall comparison over the classical benchmark instances

		LLE04			FLL+06			BCM08		
Class	NP	Unsolved	Gap	Time	Unsolved	Gap	Time	Unsolved	Gap	Time
A	22	7	2.06	6638	0	0.81	1961	0	0.20	118
B	20	1	0.61	8178	0	0.47	4763	0	0.16	417
E-M	12	9	2.10	39592	3	1.19	126987	4	0.69	1025
F	3	0	0.06	1016	0	0.14	2398	3		
P	24	8	2.26	11219	0	0.76	2892	2	0.28	187
Total	81	25			3			9		
Machine		Intel Celeron 700MHz			Pentium 4 2.4GHz			Pentium 4 2.6GHz		

		BMR11			Con12			Rop12		
Class	NP	Unsolved	Gap	Time	Unsolved	Gap	Time	Unsolved	Gap	Time
A	22	0	0.13	30	0	0.07	59	0	0.57	53
B	20	0	0.06	67	0	0.05	89	0	0.25	208
E-M	12	3	0.49	303	2	0.30	2807	2	0.96	44295
F	3	1	0.11	164	1	0.06	3	0	0.25	2163
P	24	0	0.23	85	0	0.13	43	0	0.69	280
Total	81	4			3			2		
Machine		Xeon X7350 2.93GHz			Xeon E5462 2.8GHz			Core i7-2620M 2.7GHz		

Book chapter reviewing those recent algorithms

M. Poggi and E. Uchoa. New exact approaches for the capacitated VRP. In P. Toth and D. Vigo, editors, *Vehicle Routing: Problems, Methods, and Applications*, chapter 3. SIAM, second edition, To Appear

Proposed Algorithm

A BCP that borrows from all those recent works and introduces a number of new elements.

- A complex piece of algorithmic engineering, this presentation focus on the individual element that brought more improvement: the *limited memory Subset Row Cuts*

Arc-Load Formulation (ALF)

Binary variable x_{ij}^{q} indicates that some vehicle goes from i to j carrying a load of q units (with the convention that the vehicles are collecting demands).

The routes are paths over an acyclic network $\mathcal{N} = (V_Q, A_Q)$:

- $V_Q = \{(i, q) : i \in V; q = d_i, \ldots, Q\}$
- An arc $(i, j)^q \in A_Q$ goes from (i, q) to $(j, q + d_i)$

Arc-Load Formulation (ALF)

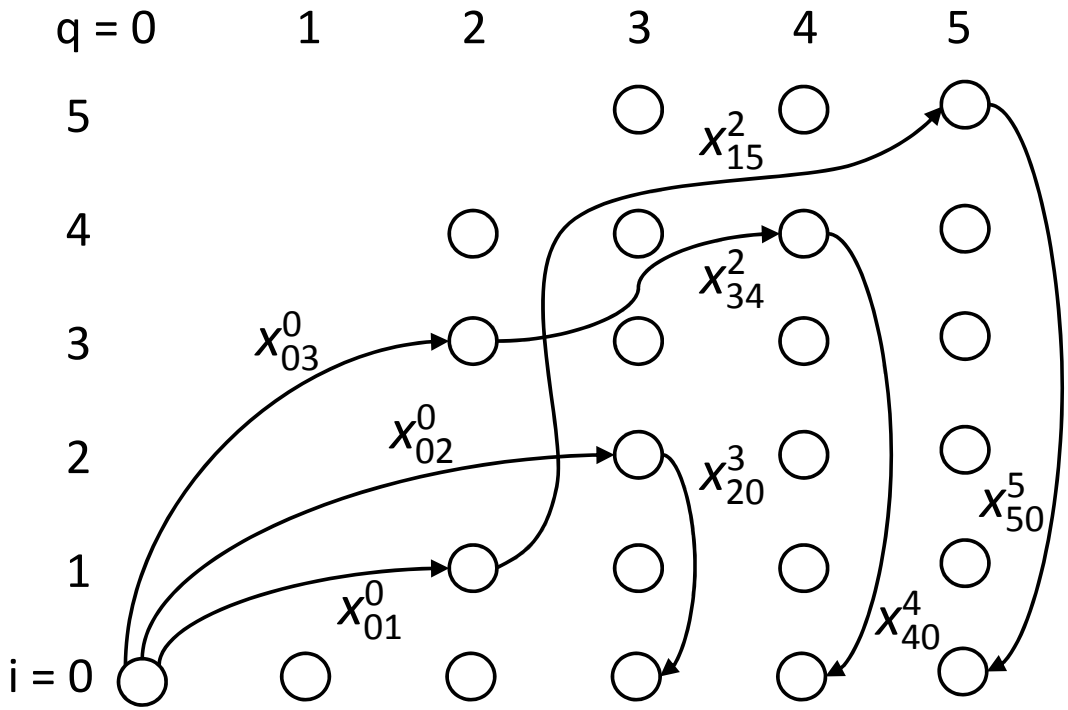


Figure: $n = Q = 5$, $d_1 = d_3 = d_4 = 2, d_2 = d_5 = 3$; routes 0-1-5-0, 0-2-0, 0-3-4-0 are shown

Arc-Load Formulation (ALF)

$$\min \quad \sum_{a^q \in A_Q} c_a x_a^q \tag{1}$$

S.t.

$$\sum_{q=d_i}^{Q} \sum_{a^q \in \delta^+((i,q))} x_a^q = 1, \qquad \forall i \in V_+, \tag{2}$$

$$\sum_{a^0 \in \delta^+((0,0))} x_a^0 = K, \tag{3}$$

$$\sum_{a^{q-d_i} \in \delta^-((i,q))} x_a^{q-d_i} - \sum_{a^q \in \delta^+((i,q))} x_a^q = 0, \qquad \forall (i,q) \in V_Q, \tag{4}$$

$$x_a^q \geq 0, \qquad \forall a^q \in A_Q, \tag{5}$$

$$x \text{ integer}. \tag{6}$$

Dantzig-Wolfe Decomposition

A q-route is a walk that starts and ends at 0, traverses a sequence of customers with total demand at most Q. Not necessarily elementary, but each time a customer is revisited its demand in counted again.

- The paths in $\mathcal{N}$ starting and ending at 0 are q-routes.
- Let Ω be the set of q-routes. For each $r \in \Omega$ define a non-negative variable λ_r and coefficients a_{rq}^{ij}, for each $(i,j)^q \in A_Q$, indicating whether (i,j) is traversed with load q in route r.

Equations (4) can be replaced by:

$$\sum_{r \in \Omega} a_{rq}^{ij} \lambda_r = x_{ij}^q, \quad \forall (i,j)^q \in A_Q. \tag{7}$$

Dantzig-Wolfe Master

Substituting the x variables and relaxing the integrality:

$$\text{(DWM)} \quad \min \quad \sum_{r \in \Omega} \left(\sum_{(i,j)^q \in A_Q} a_{rq}^{ij} c_{ij} \right) \lambda_r \tag{8}$$

S.t.

$$\sum_{r \in \Omega} \left(\sum_{(i,j)^q \in \delta^+(\{i\})} a_{rq}^{ij} \right) \lambda_r = 1, \ \forall i \in V_+, \tag{9}$$

$$\sum_{r \in \Omega} \left(\sum_{(i,j)^q \in \delta^+(\{0\})} a_{rq}^{ij} \right) \lambda_r = K, \tag{10}$$

$$\lambda_r \geq 0 \qquad \forall r \in \Omega. \tag{11}$$

Dantzig-Wolfe Master

$$\text{(DWM)} \quad \min \quad \sum_{r \in \Omega} c_r \lambda_r \tag{12}$$

S.t.

$$\sum_{r \in \Omega} a_i^r \lambda_r = 1, \qquad \forall i \in V_+, \tag{13}$$

$$\sum_{r \in \Omega} \lambda_r = K, \tag{14}$$

$$\lambda_r \geq 0 \qquad \forall r \in \Omega. \tag{15}$$

a_i^r is the number of times that customer i appears in route r.

Adding Robust Cuts to the DWM

The l-th cut with format

$$\sum_{(i,j)^q \in A_Q} \alpha_{ij}^{lq} x_{ij}^q \geq b_l$$

can be included in the DWM as

$$\sum_{r \in \Omega} \Big(\sum_{(i,j)^q \in A_Q} \alpha_{ij}^{lq} a_{ij}^{rq} \Big) \lambda_r \geq b_l.$$

Suppose there are n_R constraints in the DWM, including (9),(10), constraint l has dual variable π_l. The reduced cost of an arc $(i,j)^q$ is:

$$\bar{c}_{ij}^q = c_{ij} - \sum_{l=1}^{n_R} \alpha_{ij}^{lq} \pi_l. \tag{16}$$

The pricing subproblem is finding shortest paths in $\mathcal{N}$ with respect to the arc reduced costs. Can be done in $O(n^2 Q)$ time.

Redefining the Set of Routes

A stronger formulation would be obtained if Ω is redefined as the set of elementary routes. On the other hand, the pricing would become a strongly NP-hard problem.

The definition of Ω should be a balance between formulation strength and impact on the pricing.

As in other recent works, we chose *ng*-routes:

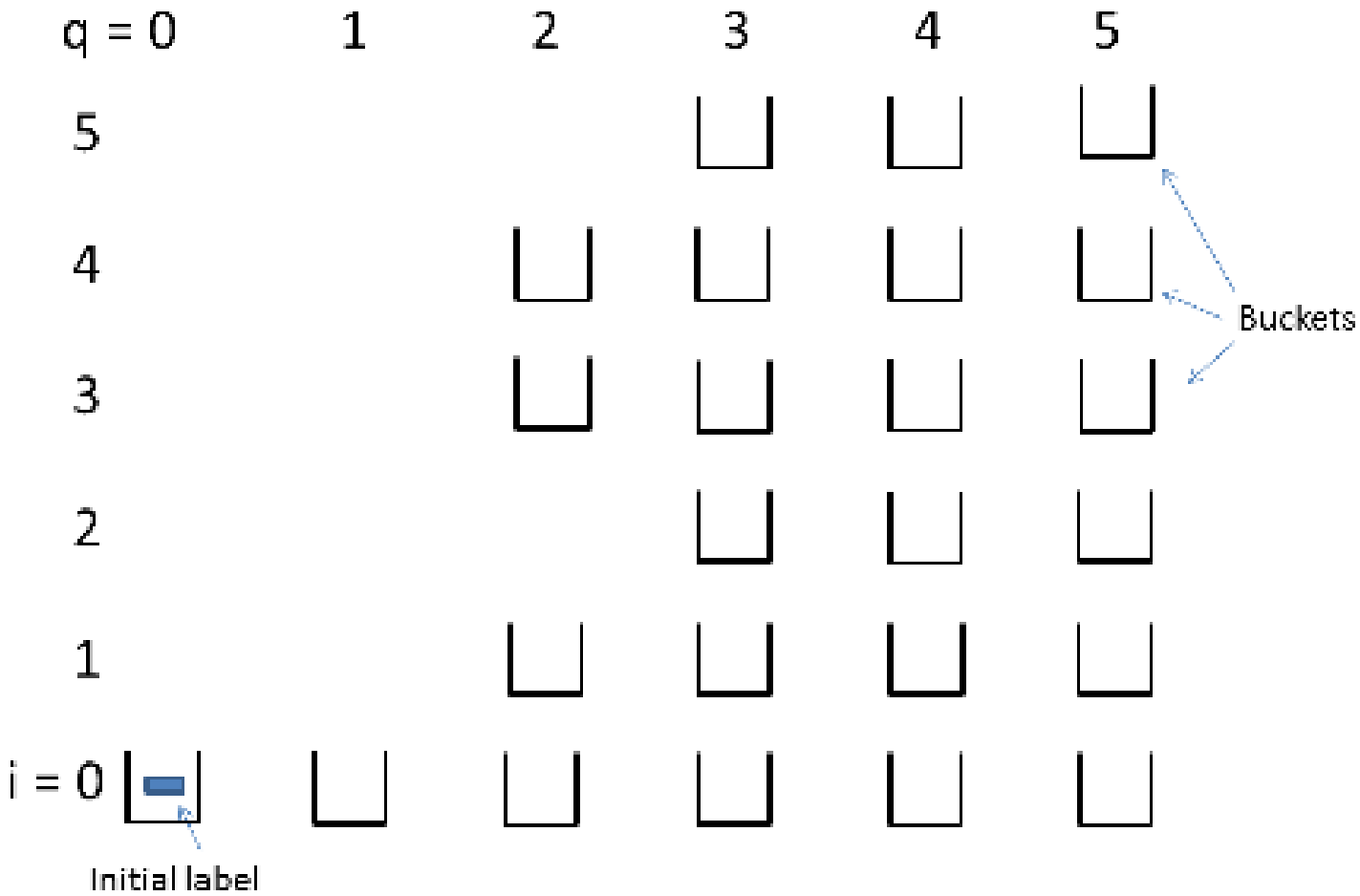
- For each $i \in V_+$, let $NG(i) \subseteq V_+$ be the neighborhood of i, its ng closest customers. A ng-route can only revisit customer i after it passes by another costumer j such that $i \notin NG(j)$.
- Using $ng = n$, ng-routes are elementary. We used $ng = 8$.

The Labeling Algorithm

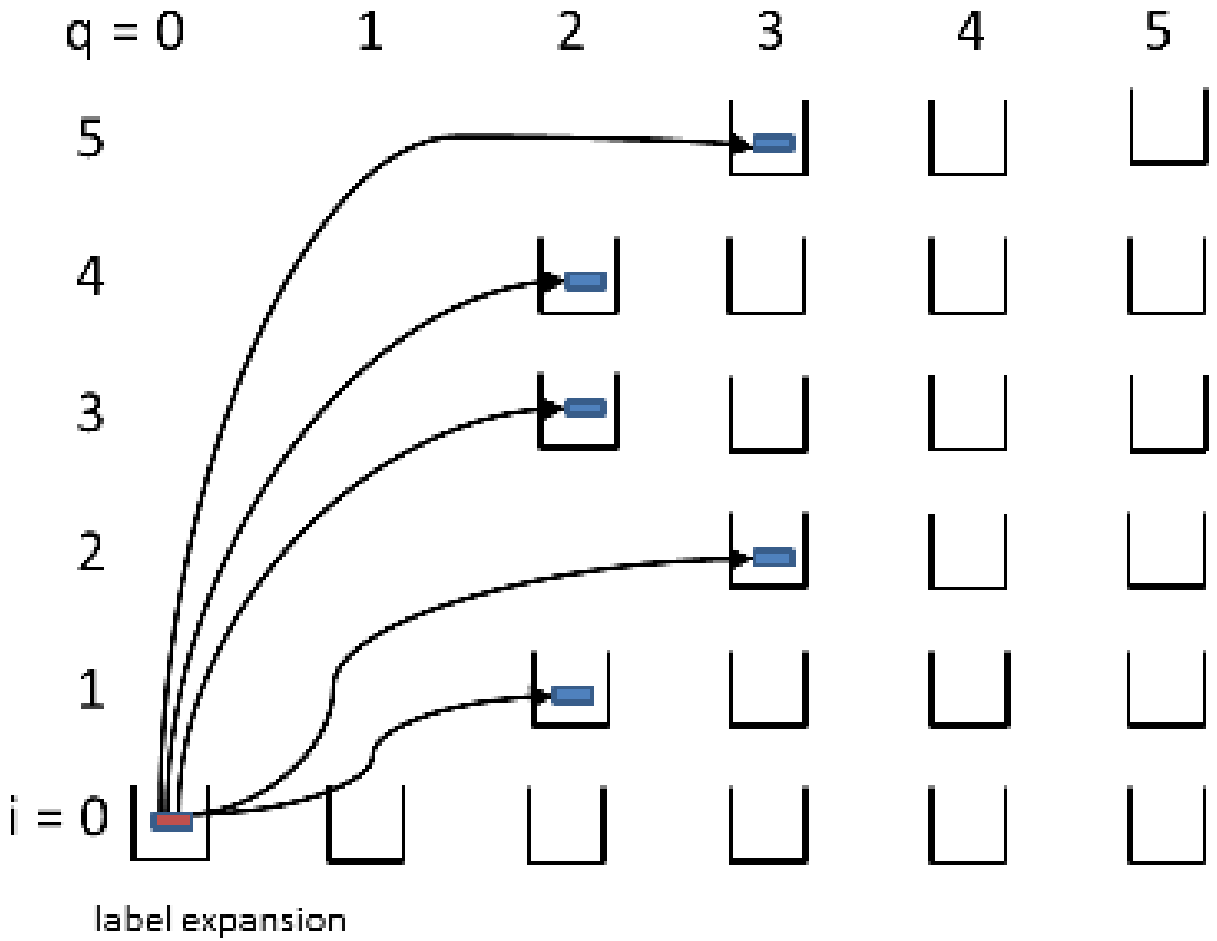
The pricing is done by a labeling dynamic programming algorithm.

- Each $(i, q) \in V_Q$ defines a **bucket**.
- A **label** $L(P)$ is a data structure representing a partial path P, with cost $\bar{c}(P)$. All labels corresponding to paths ending in i with load q are kept in bucket $B(i, q)$.
- An initial label representing a null path is put in $B(0, 0)$.
- Labels are **expanded** producing other labels.
- Dominated labels should be removed along the algorithm.

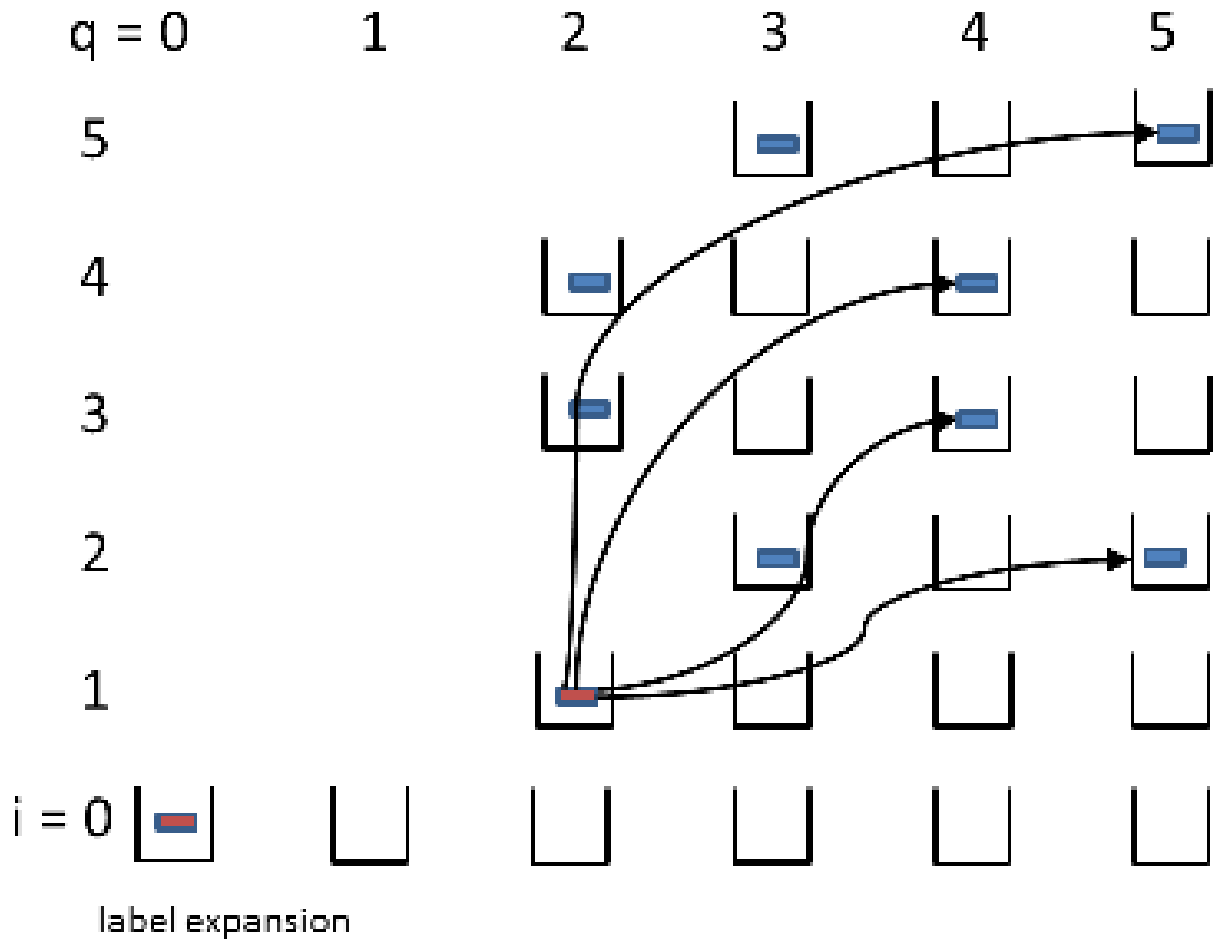
The Labeling Algorithm



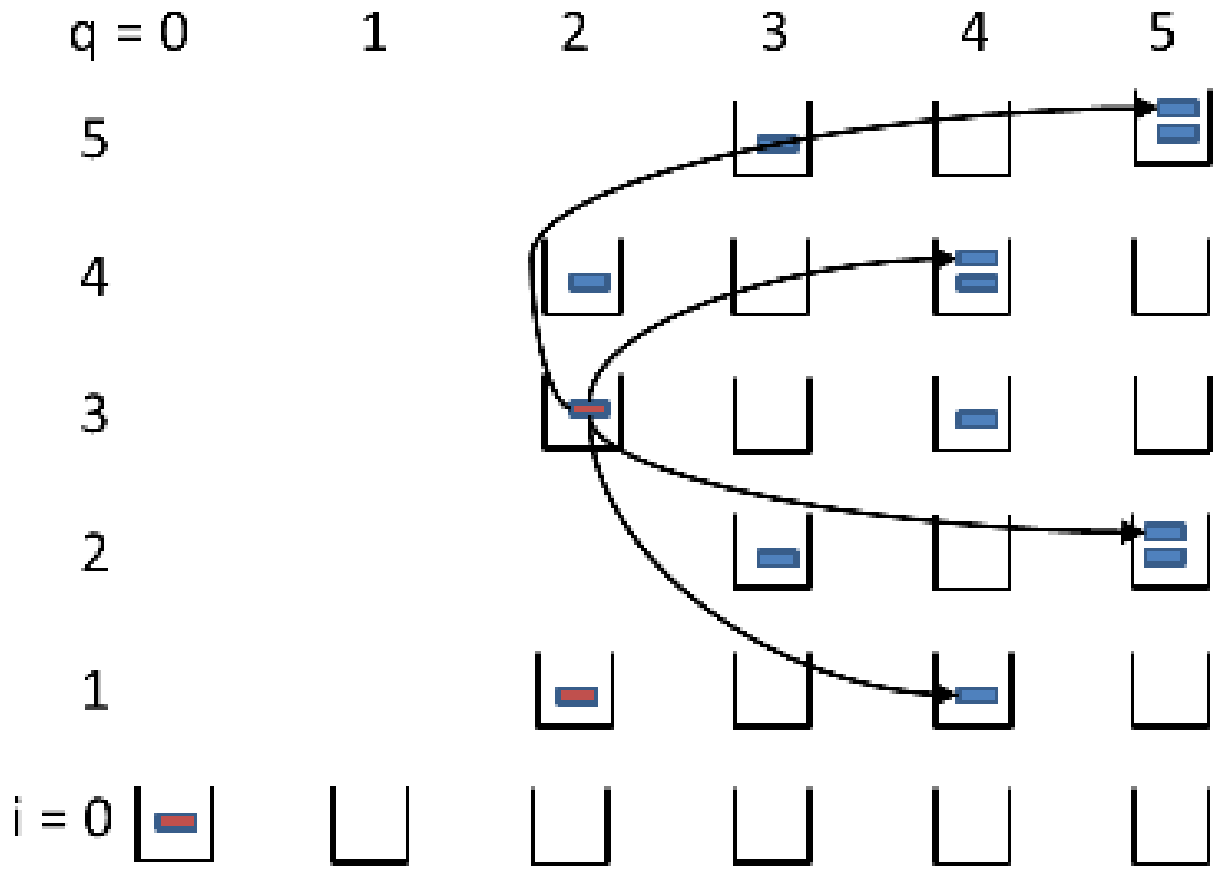
The Labeling Algorithm



The Labeling Algorithm

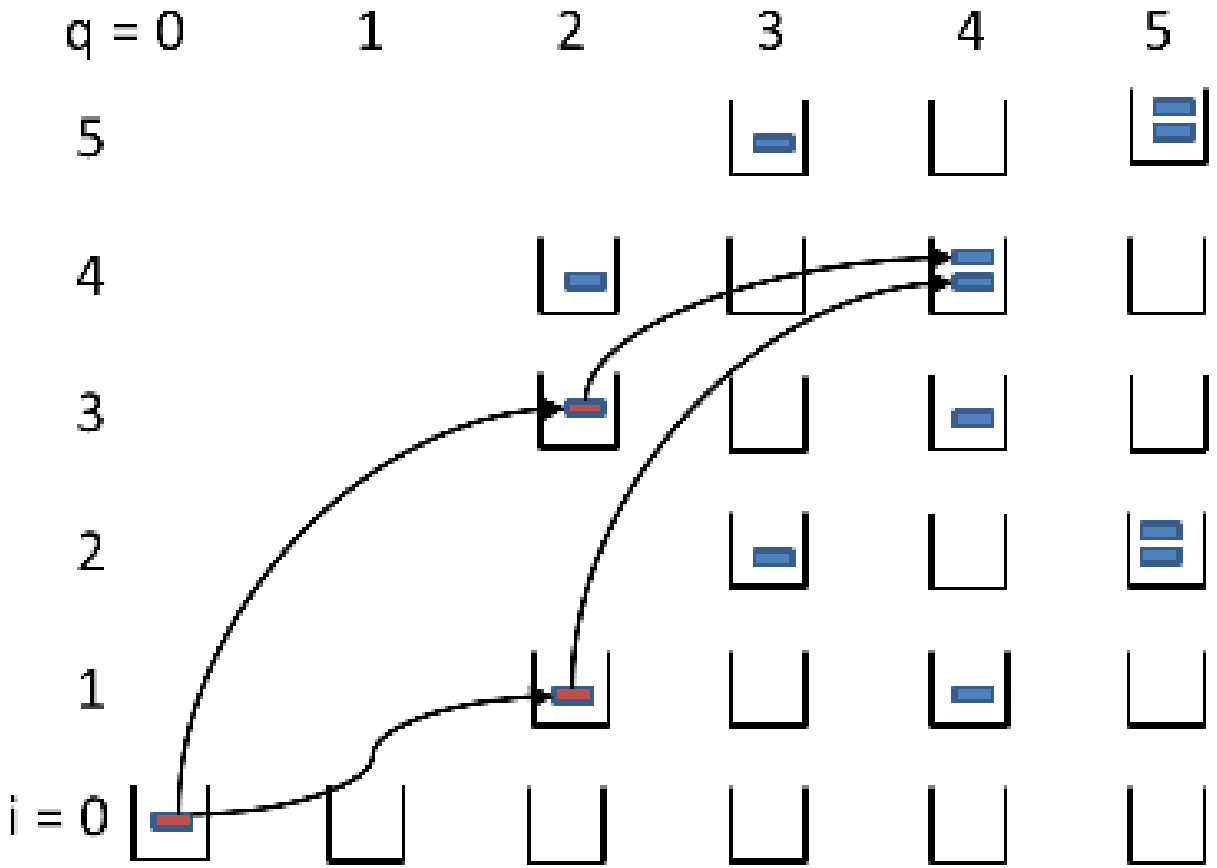


The Labeling Algorithm



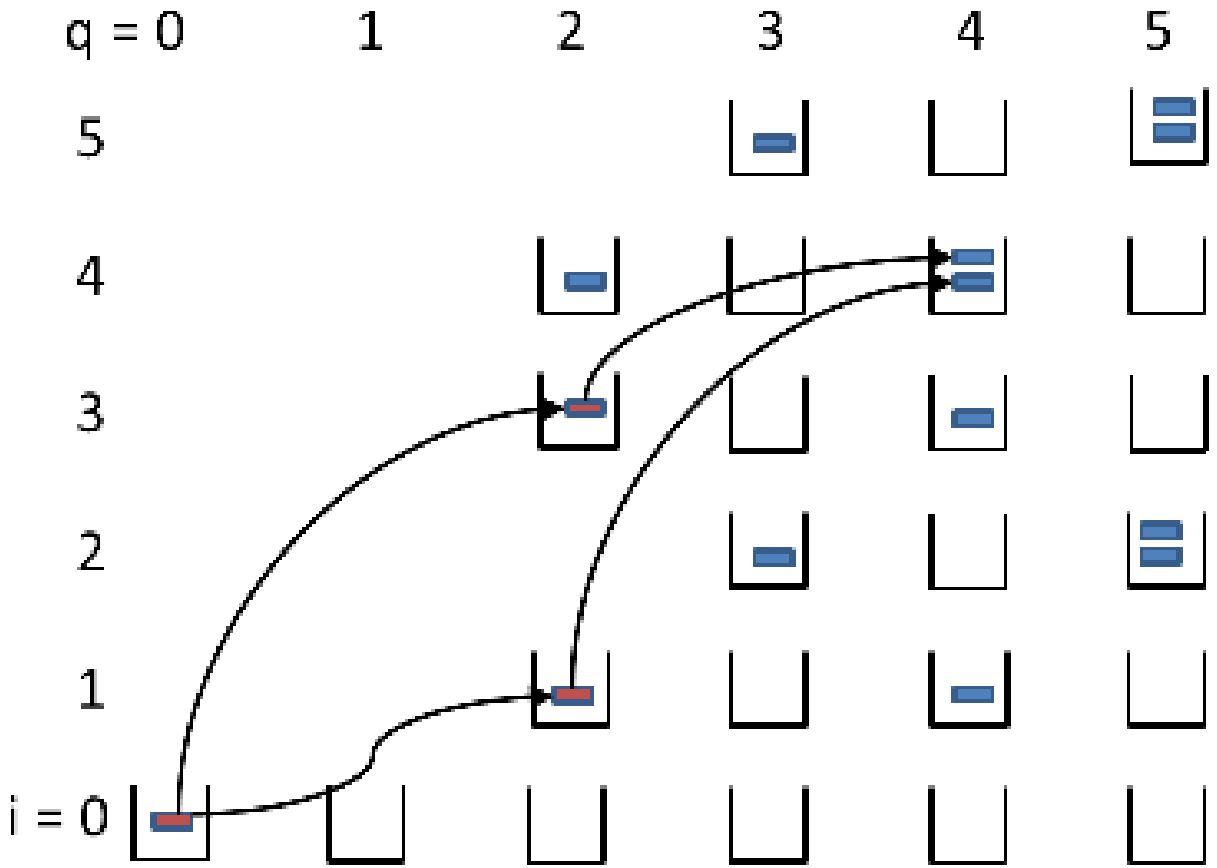
Do both labels need to be kept in bucket (4,4)?

The Labeling Algorithm



The labels represent partial paths 0-1-4 and 0-3-4

The Labeling Algorithm



Do both labels need to be kept in bucket (4,4)? Depends on the definition of Ω

Domination in the Labeling Algorithm

Rule

A label $L(P_1)$ dominates another label $L(P_2)$ in the same bucket if $\bar{c}(P_1) \leq \bar{c}(P_2)$ and every valid completion of P_2 is also a valid completion for P_1.

- q-routes: at most one non-dominated label per bucket
- q-routes without k-cycles: at most $k!$ non-dominated labels in each bucket
- ng-routes: Let Π_1 and Π_2 be the forbidden immediate extensions of paths P_1 and P_2, due to NG-sets. If $\Pi_1 \subseteq \Pi_2$ then every valid completion of P_2 is also a valid completion for P_1. At most 2^{ng-1} labels per bucket. $ng = 8$ is a safe choice for avoiding an exponential proliferation of labels.

Subset Row Cuts (SRCs)

Given $C \subseteq V_+$ and a multiplier p, the (C, p)-Subset Row Cut is:

$$\sum_{r \in \Omega} \left\lfloor p \sum_{i \in C} a_i^r \right\rfloor \lambda_r \leq \lfloor p|C| \rfloor \tag{17}$$

Non-robust cut obtained by a Chvátal-Gomory rounding of $|C|$ constraints in the DWM.

M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, 56(2):497–511, 2008

Interesting SRCs

Given an SRC with base set C, for each integer d, define y_C^d as the sum of all variables λ_r such that $\sum_{i \in C} a_i^r = d$.

The cuts where $|C| = 3$ and $p = 1/2$ are called **3-Subset Row Cuts (3SRCs)**, expressed as:

$$y_C^2 + y_C^3 + 2y_C^4 + 2y_C^5 + \ldots \leq 1.$$

- Used in Contardo [2012] and Baldacci et al. [2011]
- Potentially very effective

Interesting SRCs

$|C| = 1$ and $p = 1/2$, **1-Subset Row Cuts (1SRCs)**:

$$y_C^2 + y_C^3 + 2y_C^4 + \ldots \leq 0.$$

- Forbid routes that revisit a certain vertex
- Similar cut used in Contardo [2012]

Interesting SRCs

$|C| = 4$ and $p = 2/3$, **4SRCs**:

$$y_C^2 + 2y_C^3 + 2y_C^4 + 3y_C^5 + 4y_C^6 + \ldots \leq 2.$$

$|C| = 5$ and $p = 1/3$, **5,1SRCs**:

$$y_C^3 + y_C^4 + y_C^5 + 2y_C^6 + \ldots \leq 1.$$

$|C| = 5$ and $p = 1/2$, **5,2SRCs**:

$$y_C^2 + y_C^3 + 2y_C^4 + 2y_C^5 + 3y_C^6 + \ldots \leq 2.$$

limited memory Subset Row Cuts (lm-SRCs)

Given $C \subseteq V_+$, a memory set M, $C \subseteq M \subseteq V_+$, and a multiplier p, the limited memory (C, M, p)-Subset Row Cut is:

$$\sum_{r \in \Omega} \alpha(C, M, p, r)\lambda_r \leq \lfloor p|C| \rfloor, \tag{18}$$

where the coefficient of a route r is computed as:

```
1: function α(C, M, p, r)
2: coeff ← 0, state ← 0
3: for every vertex i ∈ r (in order) do
4:     if i ∉ M then
5:         state ← 0
6:     else if i ∈ C then
7:         state ← state + p
8:         if state ≥ 1 then
9:             coeff ← coeff + 1, state ← state − 1
10: return coeff
```

limited memory Subset Row Cuts (lm-SRCs)

1: **function** $\alpha(C, M, p, r)$
2: $coeff \leftarrow 0$, $state \leftarrow 0$
3: **for every vertex** $i \in r$ (in order) **do**
4: **if** $i \notin M$ **then**
5: $state \leftarrow 0$
6: **else if** $i \in C$ **then**
7: $state \leftarrow state + p$
8: **if** $state \geq 1$ **then**
9: $coeff \leftarrow coeff + 1$, $state \leftarrow state - 1$
10: **return** $coeff$

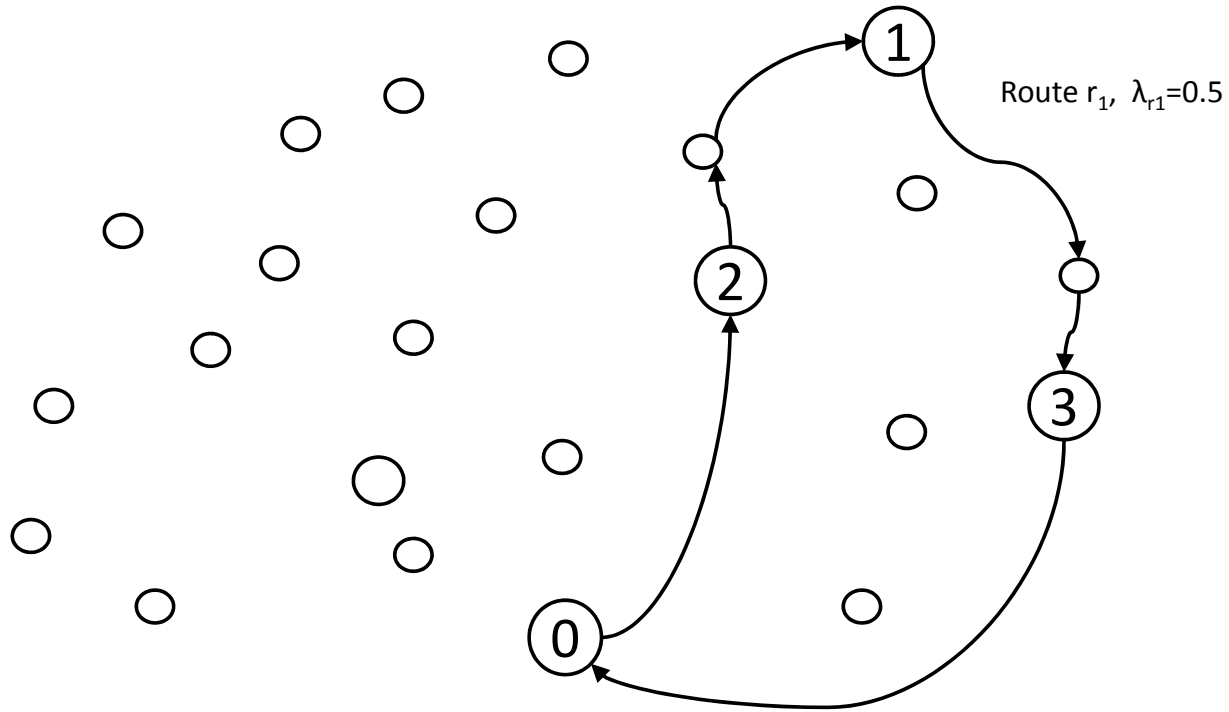
- If $M = V_+$, the function returns $\lfloor p \sum_{i \in C} a_i^r \rfloor$
- Otherwise, the lm-SRC may be a weakening of the corresponding SRC

Separation of lm-SRCs

If a violated (C, p)-SRC exists, it finds a minimal set M such that the lm-(C, M, p)-SRC has the same violation.

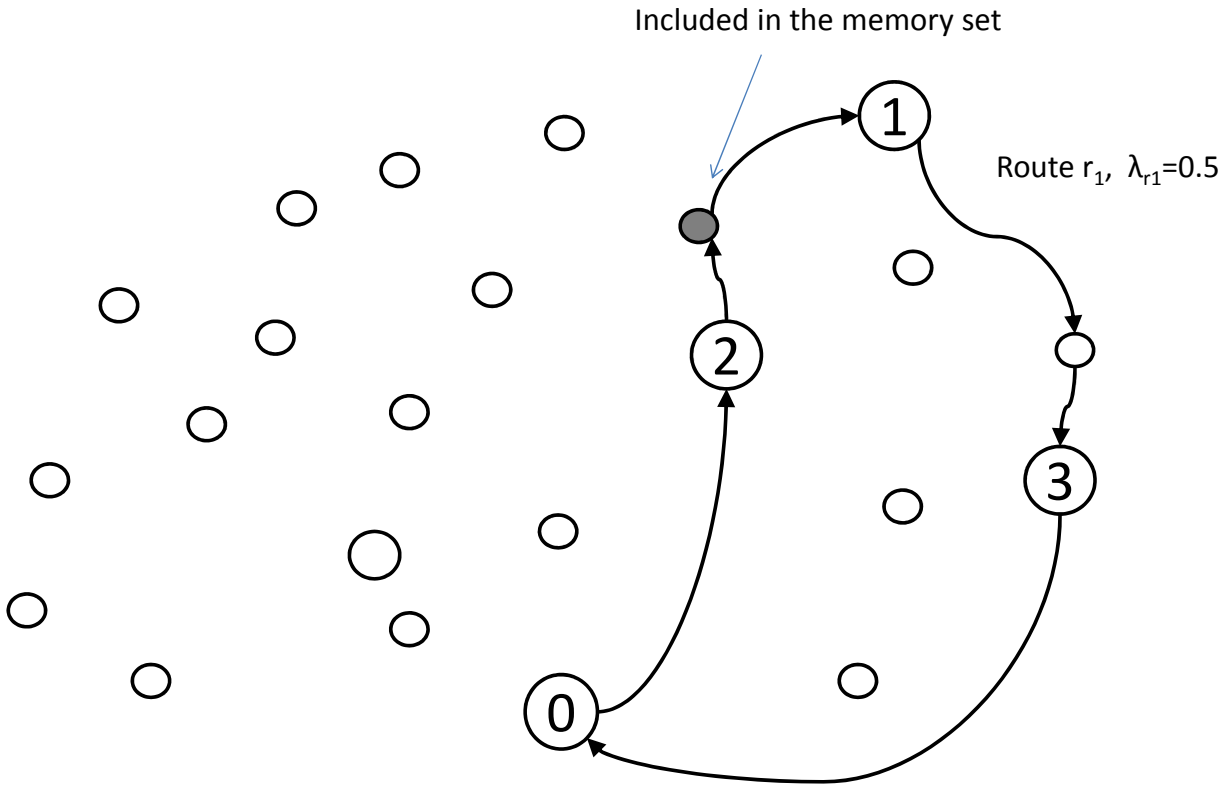
- Eventually (perhaps in more iterations), the lower bounds obtained with the lm-SRCs will be the same that would be obtained with the SRCs.

Separation of Im-SRCs



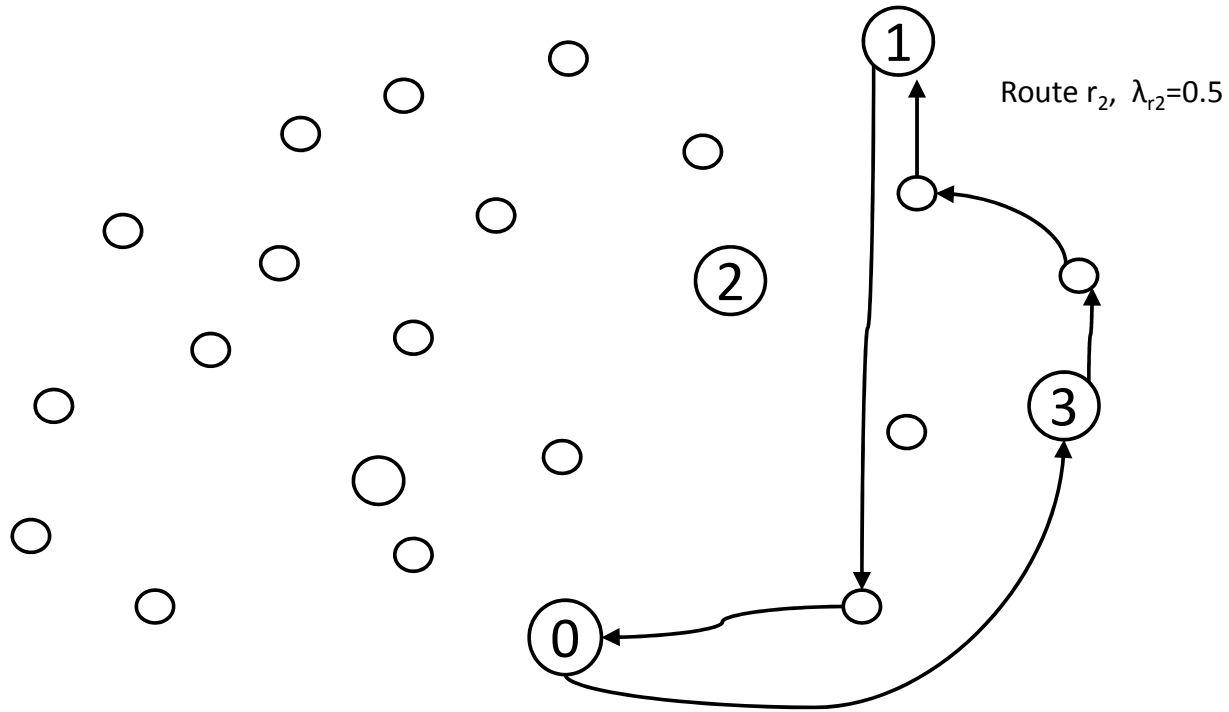
λ_{r1} has coefficient 1 in the 3-SRC with $C = \{1, 2, 3\}$

Separation of Im-SRCs



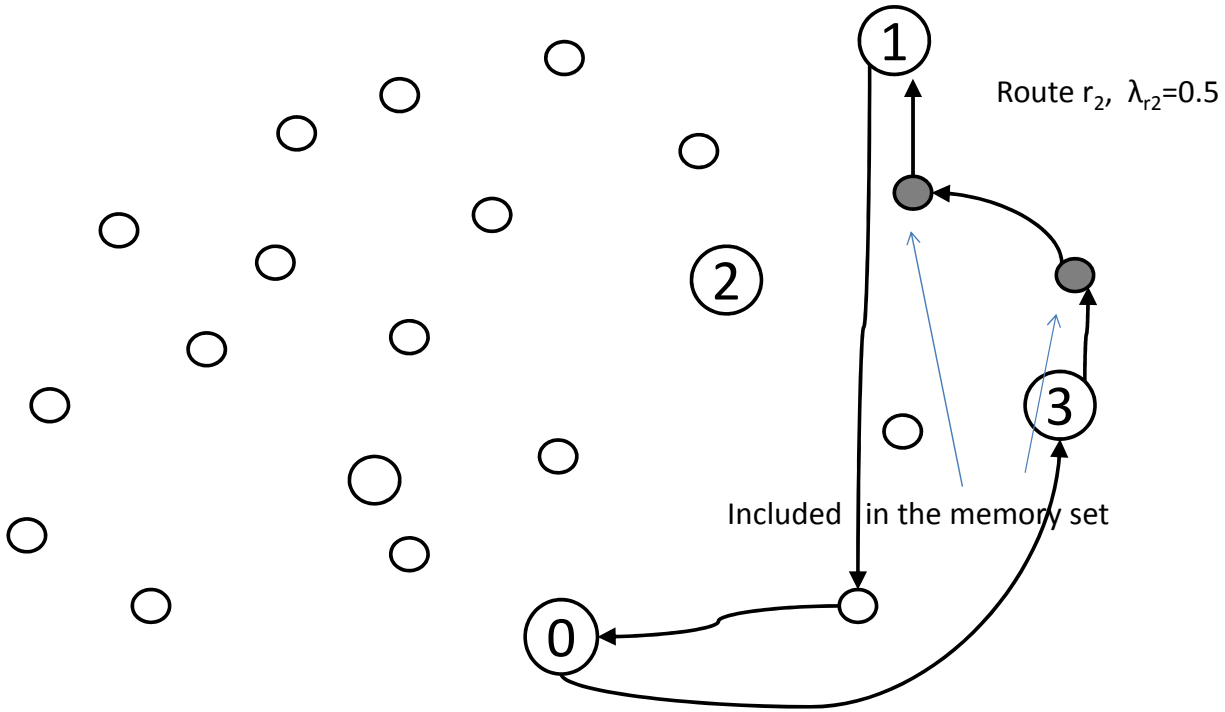
λ_{r1} still has coefficient 1 in the Im 3-SRC

Separation of Im-SRCs



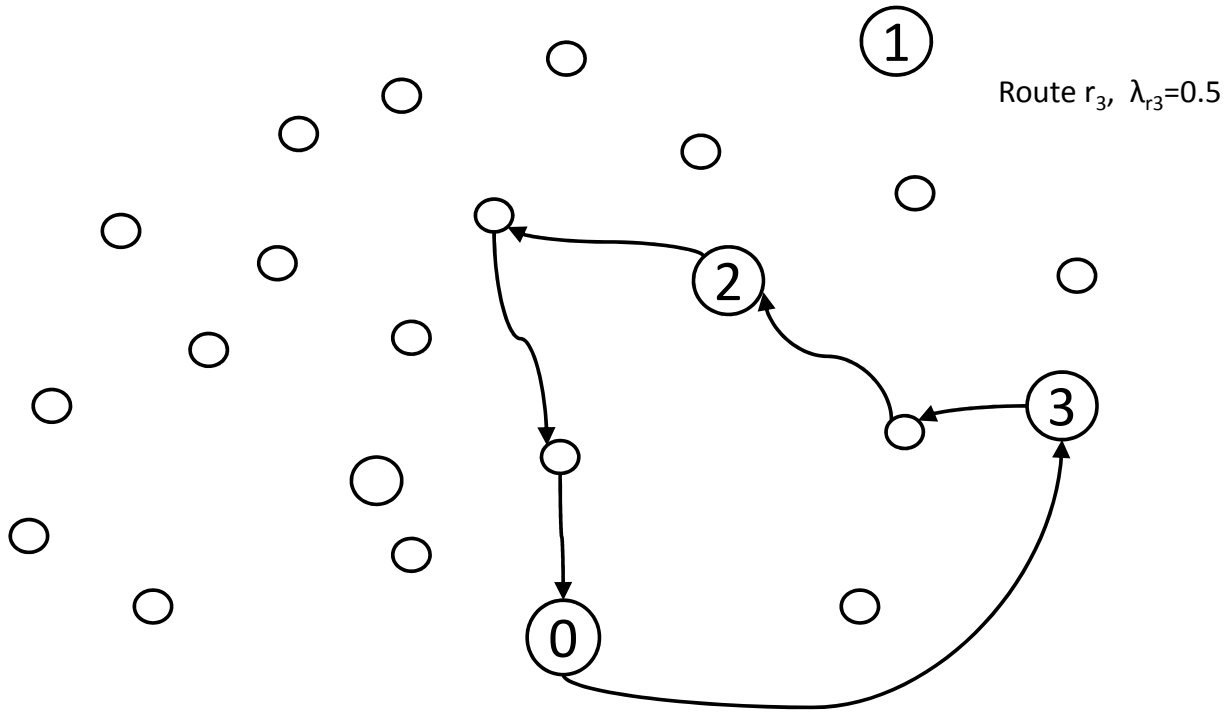
λ_{r2} has coefficient 1 in the 3-SRC with $C = \{1, 2, 3\}$

Separation of Im-SRCs



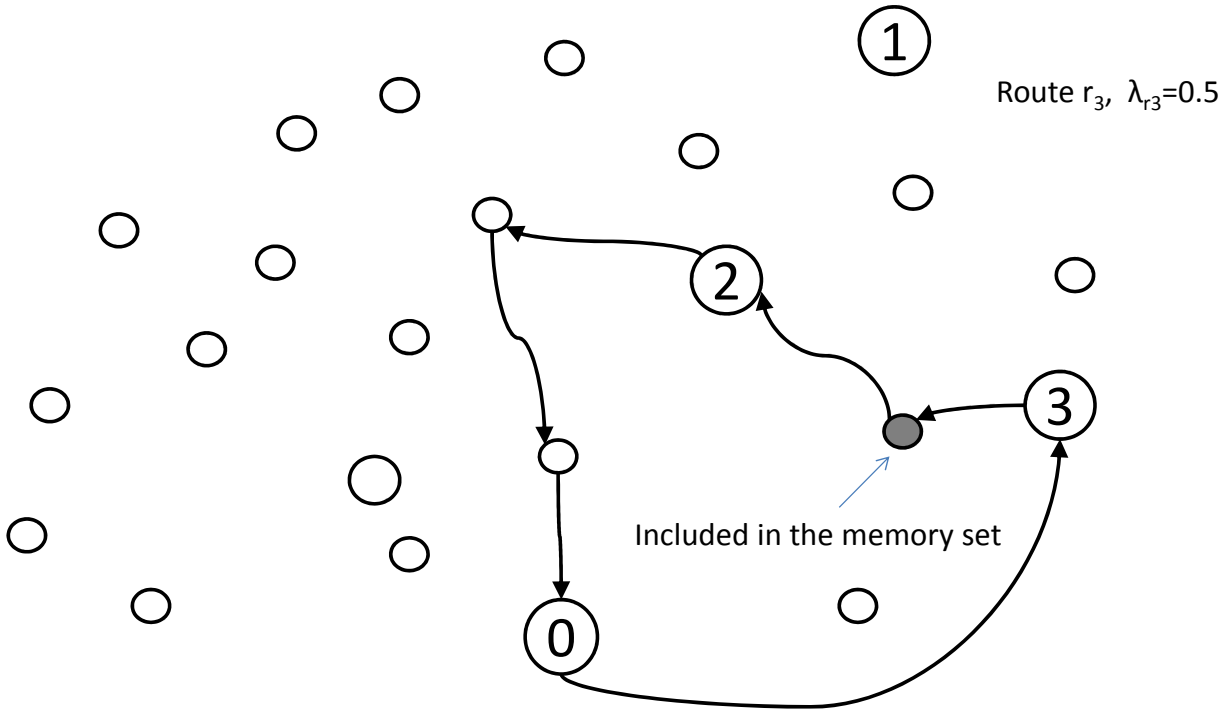
λ_{r2} still has coefficient 1 in the Im 3-SRC

Separation of Im-SRCs



λ_{r3} has coefficient 1 in the 3-SRC with $C = \{1, 2, 3\}$

Separation of Im-SRCs



λ_{r3} still has coefficient 1 in the Im 3-SRC

Why it is good to reduce the set M as much as possible?

Suppose that nS lm-SRCs are in the DWM, cut s has dual variable σ_s. Those variables **penalize** the reduced cost of some paths. $S(P)$ is the nS-dimensional vector of states of a partial path P, calculated by the α function.

Rule

A label $L(P_1)$ dominates another label $L(P_2)$ in the same bucket if $\bar{c}(P_1) \leq \bar{c}(P_2) + \sum_{1 \leq s \leq nS : S(P_1)[s] > S(P_2)[s]} \sigma_s$ and every valid completion for P_2 is also a valid completion for P_1.

- When $S(P_1)[s] > S(P_2)[s]$, it is possible that a completion for P_1 is penalized by σ_s but the same completion for P_2 is not.

Why it is good to reduce the set M as much as possible?

Rule

A label $L(P_1)$ dominates another label $L(P_2)$ in the same bucket if $\bar{c}(P_1) \leq \bar{c}(P_2) + \sum_{1 \leq s \leq n_S : S(P_1)[s] > S(P_2)[s]} \sigma_s$ and every valid completion for P_2 is also a valid completion for P_1.

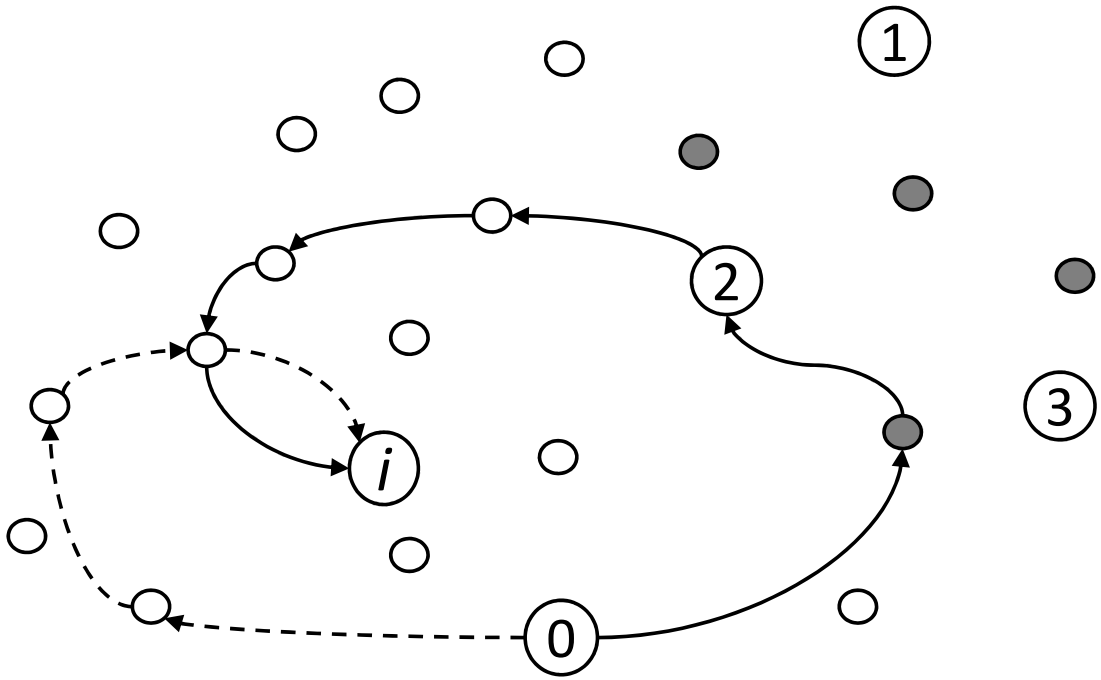
The separation of a few dozen SRCs already makes the domination much less frequent.

- The labeling algorithm is subject to an early exponential proliferation of labels

The separation of lm-SRCs with minimal M sets mitigates this effect. A lm-SRC only affects the dominance in buckets of its M.

- Many more cuts can be separated before the pricing becomes intractable

Why it is good to reduce the set M as much as possible?



Solid path may only dominate the dashed path because the lm 3-SRC $\{1, 2, 3\}$ is already forgotten at i.

New Algorithm Summary: Cuts

- Robust cuts
  - Rounded Capacity
  - Strengthened Comb
  - Path-ECC
- Non-robust cuts
  - Im-SRC
- Post-enumeration cuts
  - SRC
  - Clique

New Algorithm Summary: Pricing

The most critical part of the BCP.

- The label setting dynamic programming algorithm handles:
  - *ng*-routes
  - lm-SRCs (1SRCs, 3SRCs, 4SRCs, 5SRCs)
- Features:
  - Bidirectional Search - Differs a little from Righini and Salani [2006], concatenation not always in the middle.
  - Some implementation tricks, like data structures over bitmaps, etc.
  - Fast and effective heuristics. Exact pricing called a few times per node.

Decremental State Space Relaxation (DSSR) (Boland et al. [2006] and Righini and Salani [2008]) was tried, inconclusive results.

New Algorithm Summary: Non-robustness control

Even with all the care in their separation, Im-SRCs are indeed "non-robust":

- The pricing may be handling several hundreds Im-SRCs efficiently. Then, in some node of the tree, it suddenly becomes 100 or even 1000 times slower!
- In those cases it is necessary to **roll back**, removing the offending cuts. The node lower bound decreases, but the BCP does not halt.

New Algorithm Summary: Miscellanea

- Strong Branching:
  - Hierarchical, 3 levels
  - Uses history of past branchings
  - Aggressive, up to 200 candidates can be tested
  - Uses estimates of the subtree size determining the SB effort in each node
- Enumeration:
  - Performed when the node gap is sufficiently small for generating a pool with less than 20M routes
  - Ordinary branching occurs in enumerated nodes
  - The MIP solver only finishes the node when the pool has less than 20K routes
- Branch-and-cut: In the root node, when column generation is having severe convergence problems, it may try BC

Overall comparison over the classical benchmark instances

		BMR11			Con12			Rop12		
Class	NP	Unsolved	Gap	Time	Unsolved	Gap	Time	Unsolved	Gap	Time
A	22	0	0.13	30	0	0.07	59	0	0.57	53
B	20	0	0.06	67	0	0.05	89	0	0.25	208
E-M	12	3	0.49	303	2	0.30	2807	2	0.96	44295
F	3	1	0.11	164	1	0.06	3	0	0.25	2163
P	24	0	0.23	85	0	0.13	43	0	0.69	280
Total	81	4			3			2		
Machine		Xeon X7350 2.93GHz			Xeon E5462 2.8GHz			Core i7-2620M 2.7GHz		

		This BCP		
Class	NP	Unsolved	Gap	Time
A	22	0	0.03	7
B	20	0	0.04	8
E-M	12	0	0.14	8395
F	3	0	0.28	390
P	24	0	0.08	24
Total	81	0		
Machine		Core i7-3770 3.4GHz		

Detailed Results: M-n151-k12

Algo	Machine	Root LB	Final LB	Total Time
BMR11	X7350 2.93GHz	1004.3	1004.3	380
Contardo12	E5462 2.8GHz	1012.5	**1015**	19699
Ropke12	i7-2620M 2.7GHz	1001.5	**1015**	417146
This BCP	i7-3770 3.4GHz	1013.7	**1015**	349

Detailed Results: M-n200-k17

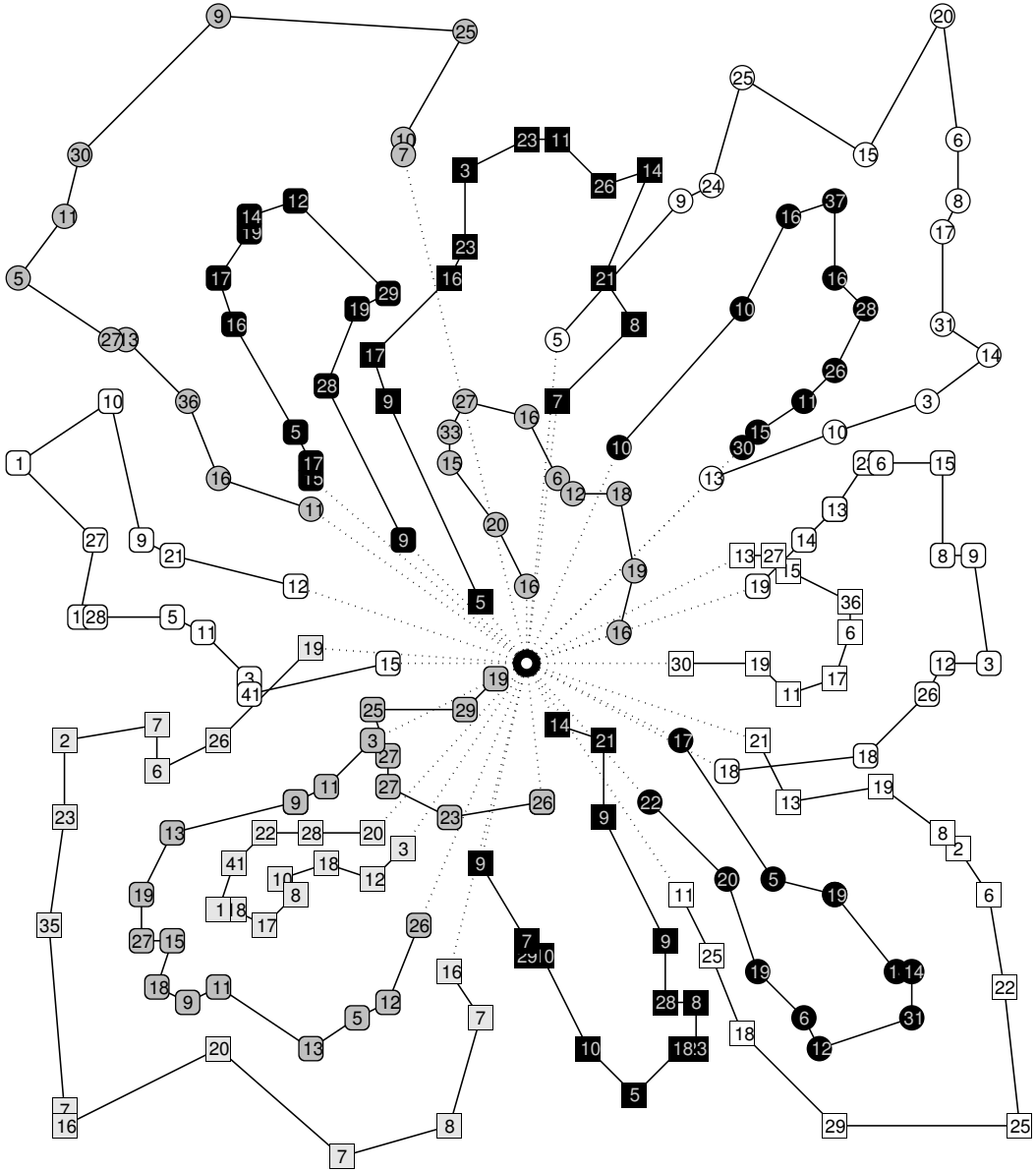
Algo	Machine	Root LB	Final LB	Total Time
BMR11	X7350 2.93GHz	1258.7	1258.7	436
Contardo12	E5462 2.8GHz	1265.1	1265.1	34350
Ropke12	i7-2620M 2.7GHz	1255.3	1261.4	7200
This BCP	i7-3770 3.4GHz	1271.0	**1275**	10230

Detailed Results: M-n200-k16

Algo	Machine	Root LB	Final LB	Total Time
BMR11	X7350 2.93GHz	1256.6	1256.6	319
Contardo12	E5462 2.8GHz	1263.0	1263.0	265588
Ropke12	i7-2620M 2.7GHz	1253.0	1258.2	7200
This BCP	i7-3770 3.4GHz	1266.0	**1274**	89772

Previous upper bound: 1278.

Optimal solution of M-n200-k16 ($Q = 200$), cost 1274



M-n200 with fractional costs (a.k.a. C5)

In the heuristics literature it is usual not to round the edge costs and not to fix the number of vehicles.

Algorithm	Best Sol
Rochat and Taillard [1995]	1291.45
Pisinger and Røpke [2007]	1297.12
Mester and Bräysy [2007]	**1291.29**
Nagata and Bräysy [2009]	1291.45
Vidal et al. [2012]	1291.45
Subramanian et al. [2012]	1291.45

The optimal value is indeed 1291.29, proved in 57718 seconds.

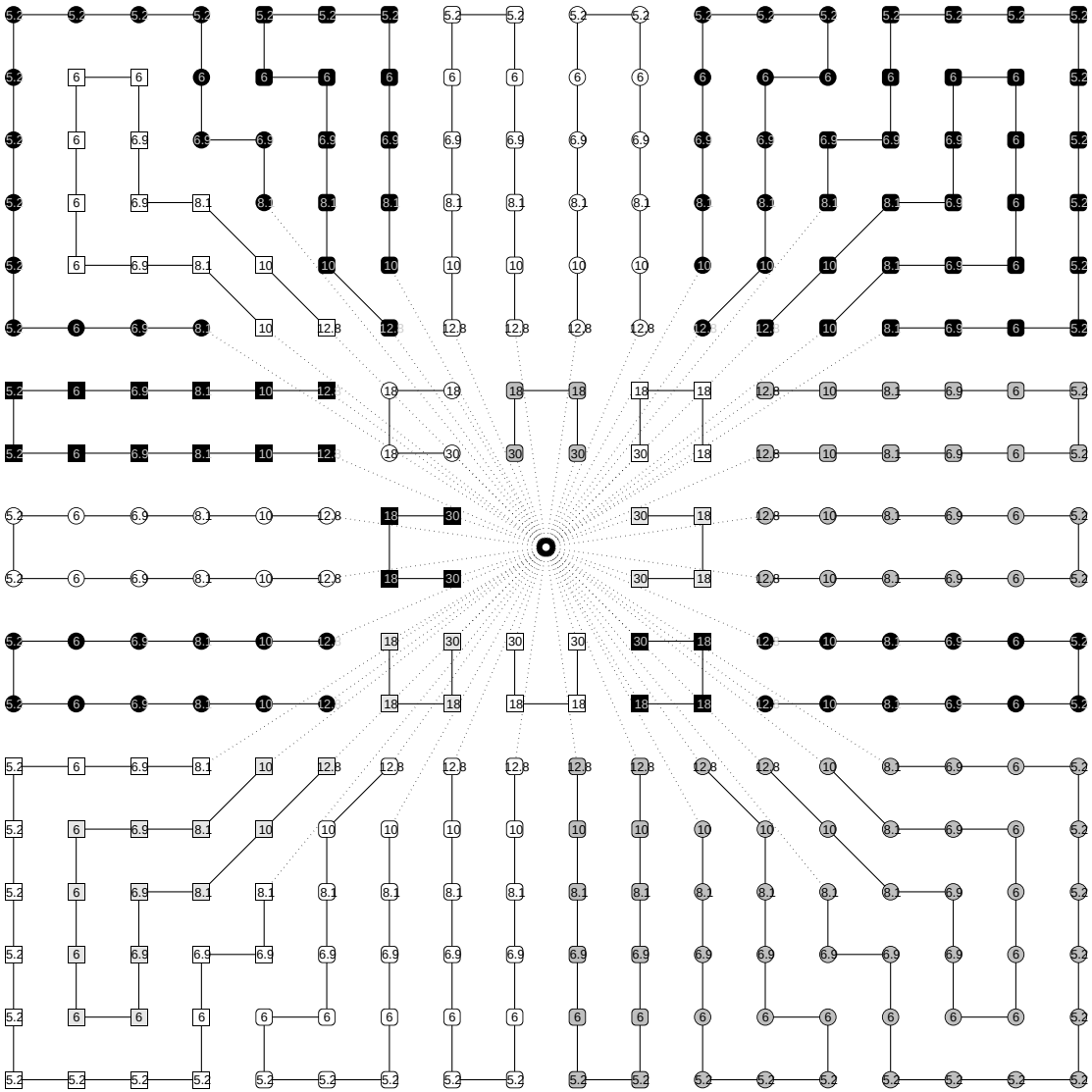
Golden's Instances

Golden, Wasil, Kelly, and Chao [1998] proposed 12 instances, ranging from 240 to 483 customers.

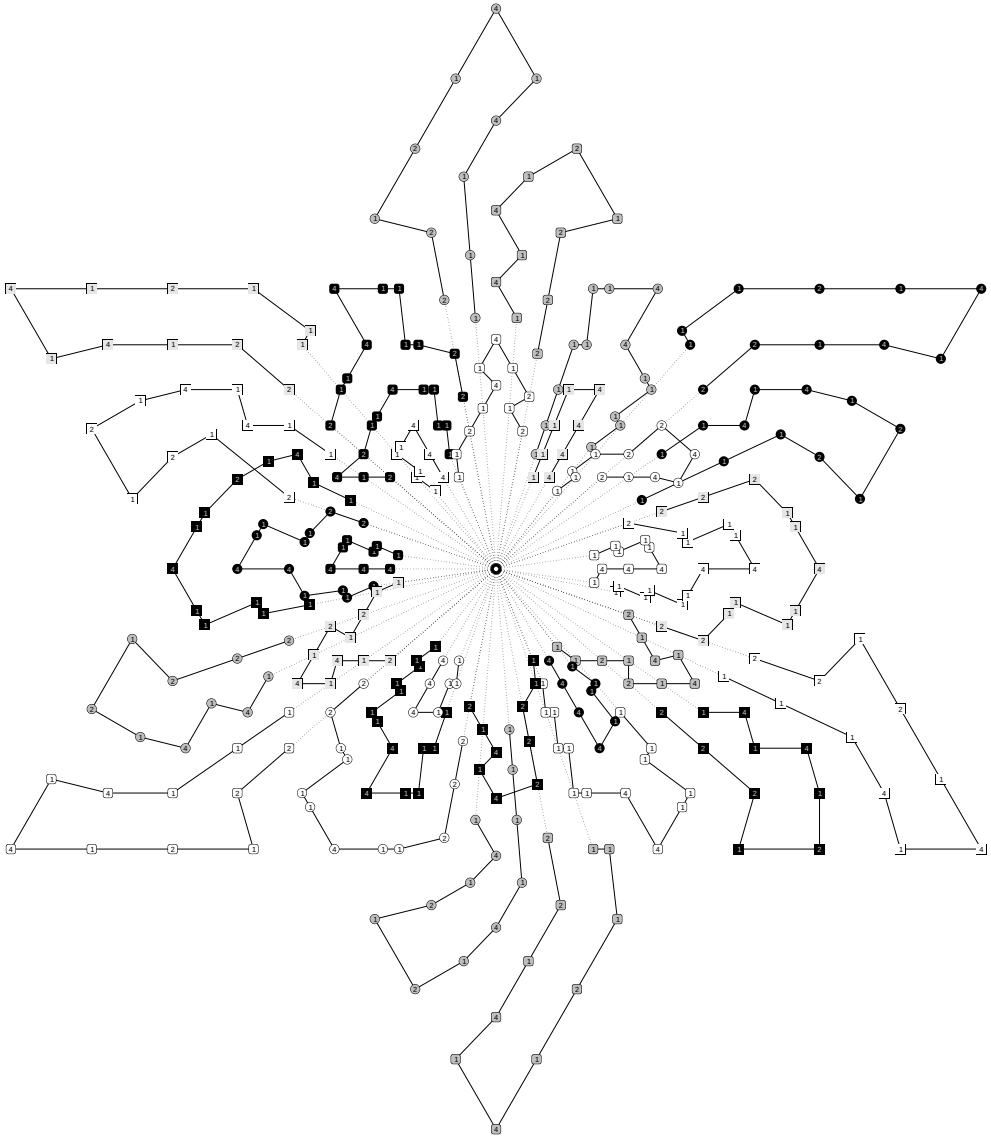
- Appear frequently in the literature on heuristic methods
- Until now, considered to be far beyond the reach of exact methods

Four instances could be solved, those with 240, 300, 320, and 360 customers.

Optimal solution of Golden_14 (320 customers, $Q = 1000$), cost 1080.55, 30 routes



Optimal solution of Golden_19 (360 customers, $Q = 20$), cost 1365.60, 33 routes



New Instances

The development of exact methods for the CVRP needs new test instances:

- The usual set of 81 benchmark instances is outdated
  - 72 instances have no more than 80 customers and are now very easy
- The 12 instances by Golden et al. are not up to that role:
  - Too few
  - Too homogeneous
  - Too artificial (the customers are located in concentric geometric figures around the depot, lots of symmetries)

New Instances

We (together with Anand Subramanian and Thibaut Vidal) are proposing a new set test instances:

- 50 instances between 100 and 330 customers, the more interesting range for current exact methods
- 50 instances between 335 and 1000 customers, at the moment, mainly for heuristic methods

New Instances

The new instances were generated by a sampling method, trying to cover a wide range of characteristics:

- Depot Location
  - Random
  - Central
  - Corner
- Customer Location
  - Random
  - Clustered
  - Random-Clustered

  The clustering was devised to mimic the growth of a metropolitan area.

New Instances

The new instances were generated by a sampling method, trying to cover a wide range of characteristics:

- Demand Distribution
  - Unitary
  - Small values, small variance
  - Small values, big variance
  - Big values, small variance
  - Big values, big variance
  - Depending on costumer position
  - A few values close to Q, many small values
- Average number of customers per route
  - Between 4 and 25

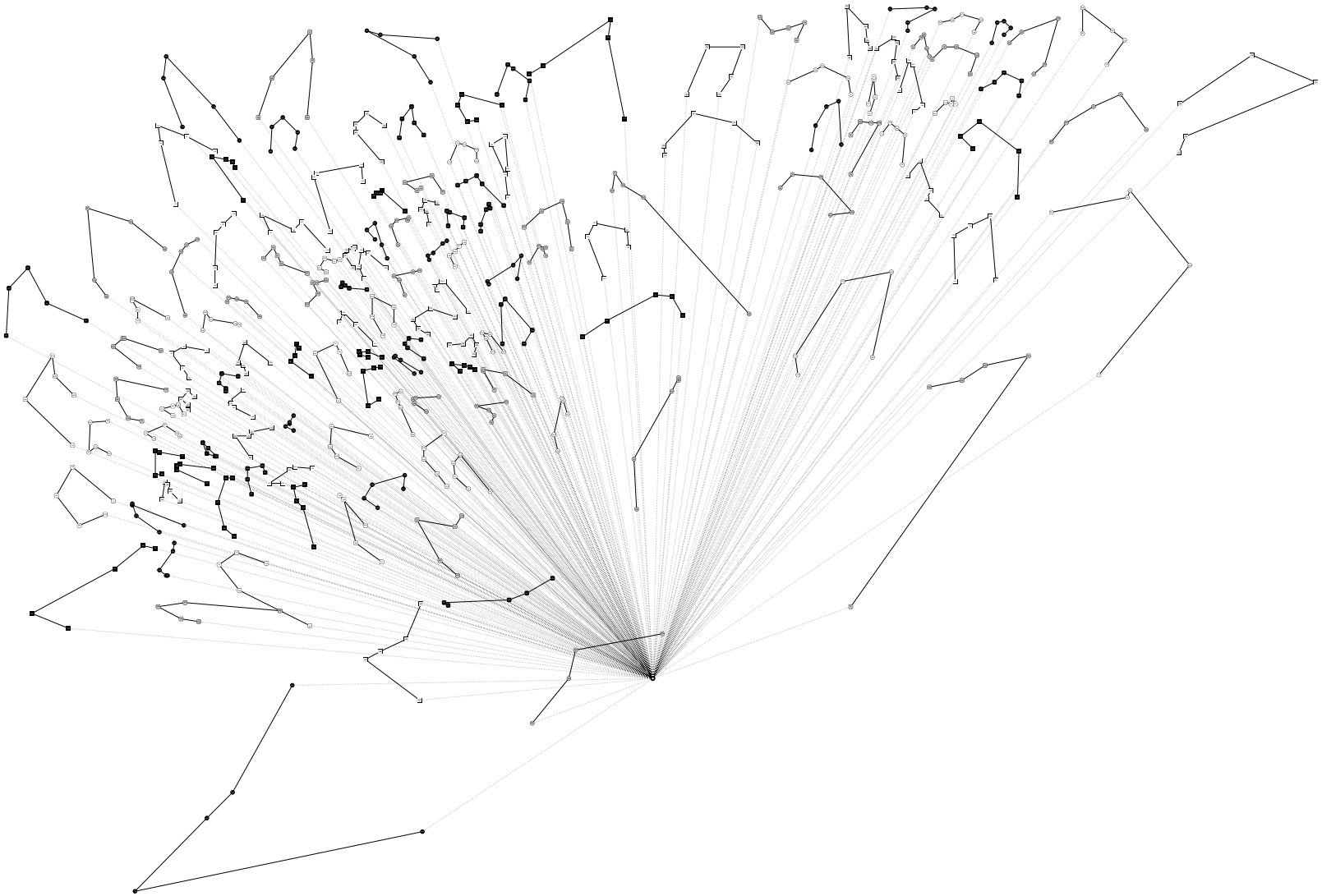
A report describing the instances (and a web page) will be released soon.

Preliminary Results on the New Instances

Most "typical instances" with up to 300 customers can be solved to optimality by the BCP

- Instances with long routes (more than 18 customers) are considerably harder. With 200 customers they already can be very challenging (to all currently known methods, including pure branch-and-cut).
- Instances with only short routes (less than 7 customers) are considerably easier. Large instances can be solved.

Optimal solution of X655 (unitary demands, $Q = 5$), cost 106780, 131 routes



Thank you for your attention!

N. Achuthan, L. Caccetta, and S. Hill. An improved branch-and-cut algorithm for the capacitated vehicle routing problem. *Transportation Science*, 37:153–169, 2003.

J. Araque, G. Kudva, T. Morin, and J. Pekny. A branch-and-cut algorithm for the vehicle routing problem. *Annals of Operations Research*, 50:37–59, 1994.

P. Augerat, J. Belenguer, E. Benavent, A. Corberán, D. Naddef, and G. Rinaldi. Computational results with a branch and cut code for the capacitated vehicle routing problem. Technical Report 949-M, Université Joseph Fourier, Grenoble, France, 1995.

R. Baldacci, N. Christofides, and A. Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115(2):351–385, 2008.

R. Baldacci, A. Mingozzi, and R. Roberti. New route relaxation and pricing strategies for the vehicle routing problem. *Operations Research*, 59(5):1269–1283, 2011.

U. Blasum and W. Hochstättler. Application of the branch and cut method to the vehicle routing problem. Technical Report ZPR2000-386, Zentrum fur Angewandte Informatik Köln, 2000.

N. Boland, J. Dethridge, and I. Dumitrescu. Accelerated label setting algorithms for the elementary resource constrained shortest path problem. *Operations Research Letters*, 34(1): 58–68, 2006.

C. Contardo. A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints. Technical report, Archipel-UQAM 5078, Université du Québec à Montréal, Canada, 2012.

R. Fukasawa, H. Longo, J. Lysgaard, M. Poggi de Aragão, M. Reis, E. Uchoa, and R.F. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical programming*, 106(3):491–511, 2006.

M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, 56(2):497–511, 2008.

J. Lysgaard, A. Letchford, and R. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100:423–445, 2004.

M. Poggi and E. Uchoa. New exact approaches for the capacitated VRP. In P. Toth and D. Vigo, editors, *Vehicle Routing: Problems, Methods, and Applications*, chapter 3. SIAM, second edition, To Appear.

T. Ralphs, L. Kopman, W. Pulleyblank, and L. Trotter Jr. On the capacitated vehicle routing problem. *Mathematical Programming*, 94:343–359, 2003.

G. Righini and M. Salani. Symmetry helps: bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, 3(3):255–273, 2006.

G. Righini and M. Salani. New dynamic programming algorithms for the resource constrained elementary shortest path problem. *Networks*, 51(3):155–170, 2008.

S. Røpke. Branching decisions in branch-and-cut-and-price algorithms for vehicle routing problems. *Presentation in Column Generation 2012*, 2012.