Branch-Cut-and-Price solver for Vehicle Routing Problems

Ruslan Sadykov$^{1,2}$, Issam Tahiri$^{1,2}$
François Vanderbeck$^{2,1}$, Remi Duclos$^1$
Artur Pessoa$^3$, Eduardo Uchoa$^3$

$^1$ Inria Bordeaux, France
$^2$ Université Bordeaux, France
$^3$ Universidade Federal Fluminense, Brazil

ISMP 2018
Bordeaux, France, July 3
Contents

Introduction

Web-based solver

The algorithm
Modern Branch-Cut-and-Price for Vehicle Routing

- Bucket graph-based labelling algorithm for the RCSP pricing
  [Desrosiers et al., 1983] [Righini and Salani, 2006]
  [Sadykov et al., 2017]
- (Dynamic) partially elementary path \((ng\)-path\) relaxation
  [Baldacci et al., 2011] [Roberti and Mingozzi, 2014]
  [Bulhoes et al., 2018]
- Automatic dual price smoothing stabilization [Wentges, 1997]
  [Pessoa et al., 2018b]
- Reduced cost fixing of (bucket) arcs in the pricing problem
  [Ibaraki and Nakamura, 1994] [Irnhich et al., 2010]
  [Sadykov et al., 2017]
- Rounded Capacity Cuts [Laporte and Nobert, 1983]
  [Lysgaard et al., 2004]
- Limited-Memory Rank-1 Cuts [Jepsen et al., 2008]
  [Pecin et al., 2017b] [Pecin et al., 2017c] [Pecin et al., 2017a]
- Enumeration of elementary routes [Baldacci et al., 2008]
  [Contardo and Martinelli, 2014]
- Multi-phase strong branching [Røpke, 2012] [Pecin et al., 2017b]
Motivation

An expert team needs several months of work to implement a state-of-the-art Branch-Cut-and-Price algorithm.

Our objective

An implementation which may be easily used by researchers.
Vehicle Routing Problem

Set $V$ of customers, each $v \in V$ with demand $w_v$, service time $t_v$ and time window $[l_v, u_v]$.

Set $M$ of vehicle types, each $m \in M$ with a depot $v_{|V|+m}$ with $U_m$ vehicles of capacity $W_m$, with vectors of travel costs $c^m$ and travel distances $d^m$.

Variants

- CVRP (with time windows)
- Distance-constrained VRP
- Heterogeneous VRP
- Multi-depot VRP
- Site-dependent VRP
Contents

Introduction

Web-based solver

The algorithm
Logging in

Address

allgo.inria.fr/app/vehiclerouting

One needs to sign up (e-mail and password only) and log in.
Running the solver

Vrp app

Vehicle Routing Problem Solver.

Create a job (remaining quota: 20 GB):

Files:

Files uploaded

A-n37-k6.vrp

Remove

From url:

Version: 1.3  It's the last one by default

Queue: batch (<1 day)

Parameters: --cutOffValue 950  --roundDistances true

Run this job
Parameters

roundDistances — whether to round the distances between clients

Initial upper bound

- By default, jsprit (github.com/graphhopper/jsprit) heuristic is used (Ruin-and-Reconstruct Local Search)
- jspritMaxIteration — number of iterations in the heuristic (default is 500),
- cutOffValue — value specified by the user if given, jsprit heuristic can be turned off (jspritMaxIteration=0)
## Supported problems data formats

<table>
<thead>
<tr>
<th>Problem</th>
<th>Format</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVRP/DCVRP</td>
<td>CVRPLIB format (vrp.atd-lab.inf.puc-rio.br)</td>
<td>[vrp.atd-lab.inf.puc-rio.br]</td>
</tr>
<tr>
<td>VRPTW</td>
<td>[Solomon, 1987] [Gehring and Hombergerer, 2002] format (total distance objective)</td>
<td></td>
</tr>
<tr>
<td>MDVRP/SDVRP</td>
<td>[Cordeau et al., 1997] format</td>
<td></td>
</tr>
<tr>
<td>HVRP</td>
<td>[Taillard, E. D., 1999] format</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[Duhamel et al., 2011] format (explicit distances)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[Pessoa et al., 2018a] format (CVRPLIB)</td>
<td></td>
</tr>
<tr>
<td>Other</td>
<td>Let us know!!!</td>
<td></td>
</tr>
</tbody>
</table>

## Other problems

Extension to other VRP variants can be considered (contact e-mail is ruslan.sadykov@inria.fr)
## Output of the solver

**App:**
VRP App

**Version:**
1.3

**Parameters:**
`--cutoffValue 950 --roundDistances true --jspritMaxIteration 0`

**Status:**
done

**Queue:**
batch (<1 day)

### Files:

**solution_A-n37-k6.txt** 636 Bytes

Best found solution of value 949:
Vehicle 1 of type 1: depot -> 16 -> 35 -> 25 -> 7 -> depot, demand = 99, time = 104
Vehicle 3 of type 1: depot -> 33 -> 2 -> 28 -> 23 -> 22 -> 12 -> 11 -> 10 -> 4 -> depot
Vehicle 4 of type 1: depot -> 24 -> 29 -> 36 -> 6 -> 14 -> depot, demand = 88, time =
Vehicle 5 of type 1: depot -> 18 -> 26 -> 21 -> 9 -> 19 -> 31 -> depot, demand = 97, time =
Vehicle 6 of type 1: depot -> 17 -> 34 -> 1 -> 3 -> 5 -> 8 -> 20 -> depot, demand = 89

Solution is feasible

**outputSummary_A-n37-k6.txt** 101 Bytes

Model: A-n37-k6.vrp
bestDualBound: 949
bestIncumbent: 949
bcCountNodeProc: 1

TimeMain: 8.24 seconds

**A-n37-k6.vrp** 808 Bytes

**algo.log** 109 Bytes

```
--cutoffValue 950 --roundDistances true --jspritMaxIteration 0
Solution found

=== ALLGO JOB SUCCESS ===
```

**BalPTree_A-n37-k6.pdf** 4.24 KB
Example of the branching tree

N_1 (56.7s) [8455.08,8662.00]

N_2 (2m20s) [8658.17,8662.00] PPN_0 >= 15

N_3 (3m5s) [8510.47,8662.00] PPN_0 <= 14

N_8 (4m58s) BOUND [8662.00] AggX_0_54_55 >= 1

N_9 (5m19s) [8658.90,8662.00] AggX_0_54_55 <= 0

N_4 (3m11s) [8662.00] PPN_1 <= 0

N_5 (4m34s) [8644.29,8662.00] PPN_1 >= 1

N_10 (5m39s) BOUND [8662.00] PPN_0 >= 13

N_11 (5m53s) BOUND [8661.81] PPN_0 <= 12

N_6 (4m44s) BOUND [8662.00] AggX_0_2_57 <= 0

N_7 (4m45s) BOUND [8662.00] AggX_0_2_57 >= 1
Remarks

- Academic use only
- Your instance file is kept on the server
- Third party software used:
  - IBM ILOG Cplex — LP and MIP solver (replacement by Cbc is coming!)
  - jsprit (github.com/graphhopper/jsprit) — heuristic vehicle routing solver
  - LEMON Graph library (http://lemon.cs.elte.hu/trac/lemon)
  - Boost C++ libraries (www.boost.org)
  - CVRPSEP — RCC separator [Lysgaard et al., 2004]
- The solver is in beta version, please report us all issues!
Performance

State-of-the-art exact solver for all the problems!


Remarks

- CVRP performance is on pair with [Pecin et al., 2017b]
- Performance is very dependent on the initial upper bound
Contents

Introduction

Web-based solver

The algorithm
Set partitioning (master) formulation

- $R_m$ — set of feasible elementary routes for a type $m$ vehicle
- $a^r_v$ — number of times vertex $v$ appear in route $r$.
- $c_r$ — cost of route $r$
- Binary variable $\lambda_r = 1$ if and only if route $r \in R_m$ is used by a vehicle of type $m$

$$\min \sum_{m \in M} \sum_{r \in R_m} c_r \lambda_r$$

$$\sum_{m \in M} \sum_{r \in R_m} a^r_v \lambda_r = 1, \quad \forall v \in V,$$

$$\sum_{r \in R_m} \lambda_r \leq U_m, \quad \forall m \in M,$$

$$\lambda_r \in \{0, 1\}, \quad \forall r \in R_m, \forall m \in M.$$
Pricing sub-problem for a vehicle type $m$

Given dual solution $\pi$ of the restricted master problem, the pricing problem is

$$\min_{r \in \mathbb{R}^m} \bar{c}_r = \sum_{v, v' \in V'_m} \left( c^m_{(v,v')} - \pi v' \right) x_{(v,v')}$$

i.e. the elementary shortest path problem with capacity and time resources.

Labels
Each label $L = (v^L, \bar{c}^L, w^L, t^L, V^L)$ represents a partial route. It dominates another label $L'$ if

$$v^L = v^{L'}, \bar{c}^L \leq \bar{c}^{L'}, w^L \leq w^{L'}, t^L \leq t^{L'}, V^L \subseteq V^{L'}$$
Partial relaxation of the elementarity: *ng*-paths

[Baldacci et al., 2011]

For each vertex $v \in V$, define a memory $\mathcal{M}_v$ of vertices which “remember” $v$. If $v^L \not\in \mathcal{M}_v$, $v$ is removed from $\mathcal{V}^L$.

Sets $\mathcal{V}^L$ are smaller $\Rightarrow$ stronger domination

Small neighbourhoods (of size $\approx 8$-10) produce a tight relaxation of elementarity constraints for most instances.

Memories $\mathcal{M}_v$ are dynamically augmented

[Roberti and Mingozzi, 2014] [Bulhoes et al., 2018]

---

Non-robust rank-1 cuts [Jepsen et al., 2008] [Pecin et al., 2017b]

[Pecin et al., 2017c] [Pecin et al., 2017a]

Each cut $\eta \in \mathcal{N}$ is obtained by a Chvátal-Gomory rounding of a set $C_\eta \subseteq V$ of set packing constraints using a vector of multipliers $\rho^n$ ($0 < \rho^n_v < 1$, $v \in C_\eta$)

$$\sum_{m \in M} \sum_{r \in R_m} \left[ \sum_{v \in C_\eta} \rho^n_v a^r_v \right] \lambda_r \leq \left[ \sum_{v \in C_\eta} \rho^n_v \right]$$

- An active cut $\eta \in \mathcal{N}$ adds one resource in the RCSP pricing
- Limited-memory technique [Pecin et al., 2017b] is critical to reduce the impact on the pricing problem difficulty: for each cut, a memory (on vertices or on arcs) is defined at the separation, making the resource local

---

Pricing: structure of RCSPP instances

- A complete directed graph.
- Unrestricted in sign reduced costs on arcs
- Two global (capacity and time) resources with non-negative continuous resource consumption
- Up to $\approx 500 - 1000$ of (more or less) local binary or (small) integer resources
- Many different optimal (or near-optimal) solutions.

We want to
Find a walk minimizing the total reduced cost respecting the resource constrains, as well as many other (up to 1000) different near-optimal feasible walks
Pricing: original graph
Pricing: the bucket graph [Sadykov et al., 2017]
Pricing: the bucket graph [Sadykov et al., 2017]
Pricing: the bucket graph [Sadykov et al., 2017]
Bucket graph labeling algorithm [Sadykov et al., 2017]

Features:

- Bidirectional search + concatenation  
  [Righini and Salani, 2006]
- If the bucket graph is acyclic the algorithm is label setting
- Otherwise, it becomes label correcting. Buckets in the same strongly connected component are treated together.
- The bucket graph structure helps to reduce the number of label dominance checks and speed up concatenation
- Arcs in the bucket graph can be fixed by reduced cost, generalizing the fixing schemes in  
  [Ibaraki and Nakamura, 1994] [Irnich et al., 2010]

A bucket graph based labeling algorithm with application to vehicle routing.  
Cadernos do LOGIS 7, Universidade Federal Fluminense.
Fixing of bucket arcs by reduced cost [Sadykov et al., 2017]

A sufficient condition to fix a bucket arc \((\vec{b}, (\vec{v}_1, \vec{v}_2), \vec{b})\)

No pair \((\vec{L}, \vec{L})\) of labels at vertices \(\vec{v}^L = \vec{v}_1, \vec{v}^L = \vec{v}_2\), in “gray” buckets producing a path by concatenation along arc \((\vec{v}_1, \vec{v}_2)\) with reduced cost smaller than the current primal-dual gap.
Enumeration of elementary paths [Baldacci et al., 2008]
[Contardo and Martinelli, 2014]

- After each reduced cost fixing, we try to enumerate into a pool all elementary paths with reduced cost smaller than the current primal-dual gap in each graph $G^k$
- A labeling algorithm is used for enumeration
- If $G^k$ is enumerated, the pricing can be done by inspection
- If all graphs are enumerated and the total number of paths is “small”, the problem can be finished by a MIP solver

An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts.

A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints.
Dual price smoothing stabilization [Wentges, 1997]

- \( \bar{\pi} \) — current dual solution of the restricted master
- \( \pi^* \) — dual solution giving the best Lagrangian bound so far
- We solve the pricing problem using the dual vector

\[
\pi' = (1 - \alpha) \cdot \bar{\pi} + \alpha \cdot \pi^*,
\]

where \( \alpha \in [0, 1) \).

- Parameter \( \alpha \) is automatically adjusted in each column generation iteration using the sub-gradient of the Lagrangian function at \( \pi' \) [Pessoa et al., 2018b].

---


Branching

Strong branching [Røpke, 2012] [Pecin et al., 2017b]

- Multi-strategy
- Branching history (pseudo-costs)
- Multi-phase

Branching strategies

- Number of vehicles
- Assignment of customers to vehicle types
- Participation of arcs in routes

---


Questions?
Suggestions?


A tabu search heuristic for periodic and multi-depot vehicle routing problems.  
*Networks*, 30(2):105–119.

Plus court chemin avec contraintes d’horaires.  

Efficient frameworks for greedy split and new depth first search split procedures for routing problems.  

Parallelization of a two-phase metaheuristic for routing problems with time windows.  


References IV


New enhancements for the exact solution of the vehicle routing problem with time windows.

Improved branch-cut-and-price for capacitated vehicle routing.

Limited memory rank-1 cuts for vehicle routing problems.


References VI


