Structural decompositions and large neighborhoods for node, edge and arc routing problems

Thibaut Vidal

Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro
Rua Marquês de São Vicente, 225 - Gávea, Rio de Janeiro - RJ, 22451-900, Brazil
vidalt@inf.puc-rio.br

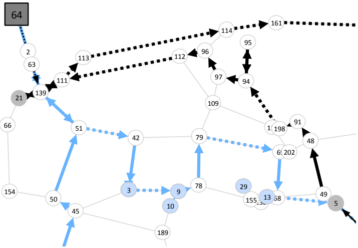
ODYSSEUS
Ajaccio, June 1–5$^{\text{th}}$, 2015

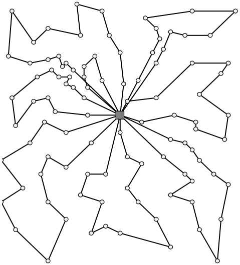
Contents

Contents

Challenges

- Arc routing for home delivery, snow plowing, refuse collection, postal services, among others.
- Bring forth additional challenges beyond "academic" vehicle routing
  - ⇒ *Deciding* on travel directions for services on edges
  - ⇒ Shortest path between services are *conditioned* by service orientations (may also need to include some additional aspects such as turn penalties or delays at intersections).





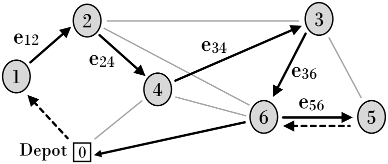
State-of-the-art algorithms

- Until 2010 → Separate streams of research on heuristics for arc and node routing problems. Some of the current state-of-the-art algorithms include:
  - **Capacitated Vehicle Routing Problem (CVRP):** UTS of Cordeau et al. (1997, 2001), AMP of Tarantilis (2005), ILS/ELS of Prins (2009), ES and HGAs of Mester and Bräysy (2007); Nagata and Bräysy (2009); Vidal et al. (2012)...
  - **Capacitated Arc Routing Problem (CARP):** GLS of Beullens et al. (2003), HGA of Lacomme et al. (2001, 2004); Mei et al. (2009), VNS of Polacek et al. (2008), TS of Brandão and Eglese (2008)...
- Arc-routing specific decisions are addressed via a **larger number of enumerative neighborhood classes** : to optimize service orientations.

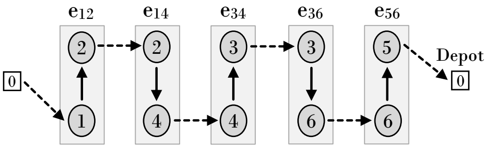
State-of-the-art algorithms

- Two alternative solution representations for the CARP:

R1. Explicit representation of *assignment*, *sequencing* decisions, *service orientations*, and *intermediate paths*.



R2. Explicit representation of *assignment*, *sequencing decisions*, and *service orientations*. *Intermediate paths* have been preprocessed.



State-of-the-art algorithms

- Recent research on combined node, edge and arc routing problems (NEARP – also called mixed capacitated general routing problem MCGRP):
  - Early constructive heuristics: (Pandi and Muralidharan, 1995; Gutiérrez et al., 2002)
  - HGA of Prins and Bouchenoua (2005)
  - SA of Kokubugata et al. (2007)
  - LNS+MIP of Bosco et al. (2014)
  - Remarkable unified metaheuristic: Dell'Amico et al. (2014). Covers a large set of CVRP, CARP, and NEARP benchmark instances. However, *"AILS uses a total of 26 move subtypes: 13 types of 3-opt, 8 types of 2-opt, 2 types of Or-opt, 2 Swap types, and Flip."*

Large neighborhoods

- Interesting large neighborhood from Muyldermans et al. (2005), scarcely used until now : dynamic programming to generate optimal traversal directions for the services of a fixed route

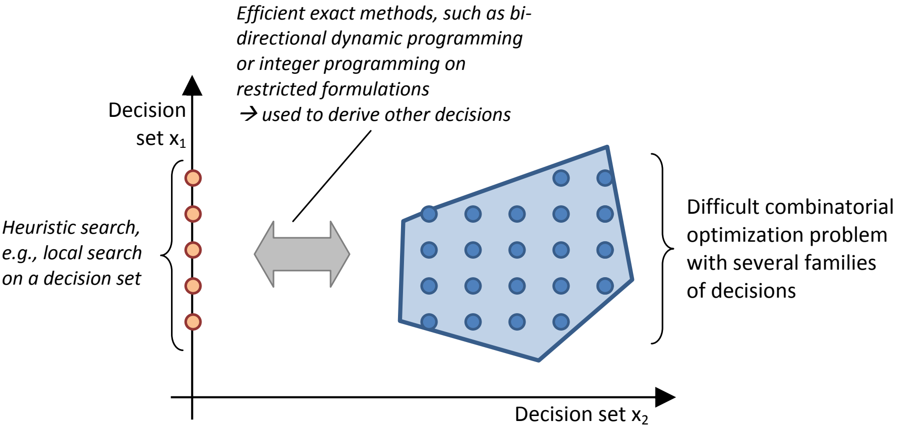
  ⇒ Used as a stand-alone procedures, or combined with a RELOCATE move. Both searches in $\mathcal{O}(n)$

  ⇒ Combined in Irnich (2008) with the neighborhood of Balas and Simonetti (2001), leading to promising results on mail delivery applications.

Contents

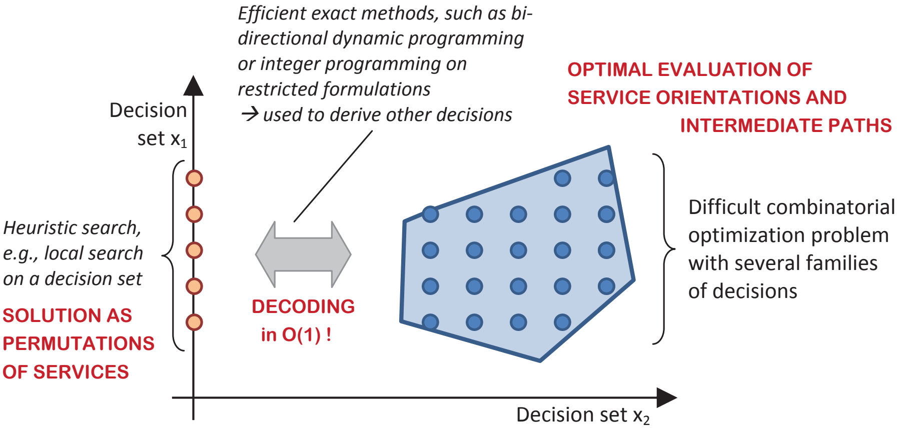
Rationale of this work

- Structural problem decomposition (used naturally in branch-and-price, less explicitly used in heuristics):



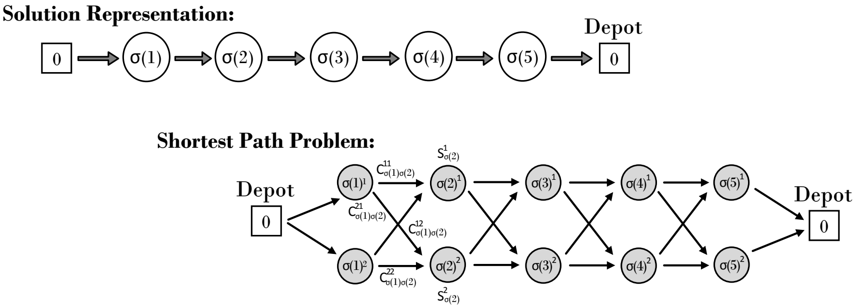
Rationale of this work

- Structural problem decomposition:



Solution representation and decoding

- How to decode/evaluate a solution = deriving optimal orientations for the services ?



- Each service is represented by two nodes, one for each possible orientation. Travel costs c_{ij}^{kl} between (i, j) are conditioned by the orientations (k, l) for departure and arrival.

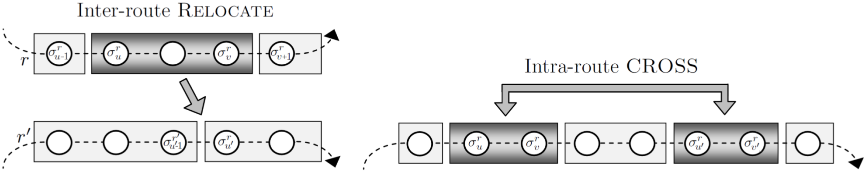
Solution representation and decoding

- Same shortest path subproblem as Muyldermans et al. (2005), but used far beyond it's original scope.
  - Operating a complete problem decomposition : **searching in the space of *service permutations*** (+ depot visits)

    ⇒ **Systematically,** for all solution and move evaluations
  - In very large neighborhoods : Ejections chains and Split algorithm
  - Also used to conceal decisions on *service modes* within the shortest path subproblem, for many variants of arc routing problems
- Evaluated **in $\mathcal{O}(\mathbf{1})$ instead of $\mathcal{O}(\mathbf{n})$**
- And even, using LBs on move evaluations, same average number of elementary operations as a CVRP move...

Seeking low complexity for solution evaluations

- Modern neighborhood-centered heuristics evaluate millions/billions of neighbor solutions during one run.
- Key property of classical routing neighborhoods:
  - Any local-search move involving a bounded number of node relocations or arc exchanges can be assimilated to a concatenation of a bounded number of sub-sequences.
  - Same subsequences appear many times during different moves



  - To decrease the computational complexity, compute auxiliary data on subsequences by induction on concatenation ($\oplus$).

Structural decomposition and route evaluations

Auxiliary data structures = partial shortest paths

Partial shortest path $C(\sigma)[k,l]$ between the first and last service in the sequence σ, for any (entry, exit) direction pair (k,l)

Initialization

For σ_0 with a single visit v_i , $S(\sigma_0)[k,l] = \begin{cases} 0 & \text{if } k = l \\ +\infty & \text{if } k \neq l \end{cases}$

Evaluation

By induction on the concatenation operator:

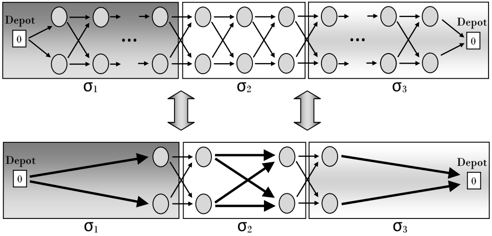
$$C(\sigma_1 \oplus \sigma_2)[k,l] = \min_{x,y} \left\{ C(\sigma_1)[k,x] + c^{xy}_{\sigma_1(|\sigma_1|)\sigma_2(1)} + C(\sigma_2)[y,l] \right\}$$

Arc Routing Problems

- **Pre-processing partial shortest paths in the incumbent solution** – in $\mathcal{O}(n^2)$ before the neighborhood exploration – dramatically simplifies the shortest paths:

Shortest path problem:

Shortest path problem on a reduced graph, using pre-processed labels:



- Only a constant number of edges !

Lower bounds on moves

- Each move evaluation was still taking a bit more operations (constant of $4\times$) than in the classic CVRP.

- Even this can be avoided...
  $\Rightarrow$ by developing lower bounds on the cost of neighbors...

Lower bounds on moves

- Let $\bar{Z}(\sigma)$ be a lower bound on the cost of a route σ
- A move that modifies two routes: $\{\sigma_1, \sigma_2\} \Rightarrow \{\sigma'_1, \sigma'_2\}$ has a chance to be improving if and only if:

$$\Delta_\Pi = \bar{Z}(\sigma'_1) + \bar{Z}(\sigma'_2) - Z(\sigma_1) - Z(\sigma_2) < 0.$$

Lower bounds on moves

- Let $C^{\text{MIN}}(\sigma) = \min_{k,l}\{C(\sigma)[k,l]\}$ the shortest path for the sequence σ between any pair of origin/end orientations.
- Let $c_{ij}^{\text{MIN}} = \min_{k,l}\{c_{ij}^{kl}\}$ be the minimum cost of a shortest path between services i and j, for any orientation.

- Lower bound on the cost of a route $\sigma = \sigma_1 \oplus \cdots \oplus \sigma_X$ composed of a concatenation of X sequences:

$$\bar{Z}(\sigma_1 \oplus \cdots \oplus \sigma_X) = \sum_{j=1}^{X} C^{\text{MIN}}(\sigma_j) + \sum_{j=1}^{X-1} c_{\sigma_j,\sigma_{j+1}}^{\text{MIN}}.$$

- The bound helps to filter a lot of moves ($\geq 90\%$ even when used with granular search)
  - In practice : possible to evaluate a move in the space of service permutations for the CARP with roughly the **same number of elementary operations as the same move for a CVRP!**

Contents

Some preliminary definitions

- **Service:** A visit to a client, which cannot be split, but may be operated in different alternative ways
- **Service Mode:** Alternative way to perform a service, may impact travel or service cost.
  $\Rightarrow$ The set of possible *modes* for a service will be notated M_i

Generalizations via enriched mode definitions

- **CARP – each service has two modes**, one for each possible orientation (curb direction during service).

- Many other mode choices in problem variants:
  - choice of sidewalk and impact on intersection time (postal delivery, refuse collection)
  - lane (snow plowing)
  - parking spot
  - choice of visit location (GVRP and arc routing equivalents)
  - orders of visit clusters, e.g., in a city district (CluVRP and arc routing equivalents)
  - entry-exit of a facility...

Generalizations via enriched mode definitions

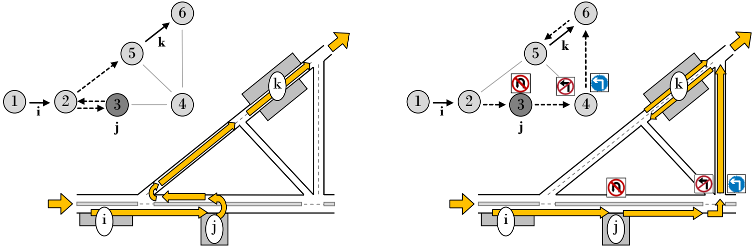
- Now, **node, edge and arc routing problems** are greatly simplified:

NODE	$\vert M_i\vert = 1$	One mode for service;
ARC	$\vert M_i\vert = 1$	One mode for the only feasible service orientation;
EDGE	$\vert M_i\vert = 2$	Two modes, one for each service orientation.

- Route-evaluation subproblem even more efficient since many services are now represented as a single node in the auxiliary graph

Generalizations via enriched mode definitions

- Problems with **turn penalties and delays at intersections** are greatly simplified:
- In previous literature – feasibility issues:
  - Solution of NEARP with turn penalties represented as sequences of services + SPs with turn restrictions between services did not necessarily lead to viable solutions:



  - Because of a **lack of characterization of the arrival edge** when servicing a node

Generalizations via enriched mode definitions

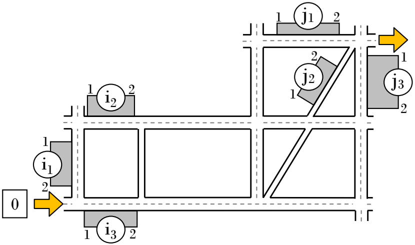
- The needed information can be included in the definition of the mode:

NODE	$\mathbf{\lvert M_i \rvert = p_i}$	$\mathbf{p_i}$ **modes** to specify the arrival direction, where p_i is the in-degree of v_i;
ARC	$\lvert M_i \rvert = 1$	One mode for the only feasible service orientation;
EDGE	$\lvert M_i \rvert = 2$	Two modes, one for each service orientation.

- Then, turn penalties can easily be included in arc costs, in the auxiliary graph
- Done $\Rightarrow$ turn penalties are now optimally addressed (for any fixed sequence of services) without any further change

Generalizations via enriched mode definitions

- Problems with **service clusters** are greatly simplified:

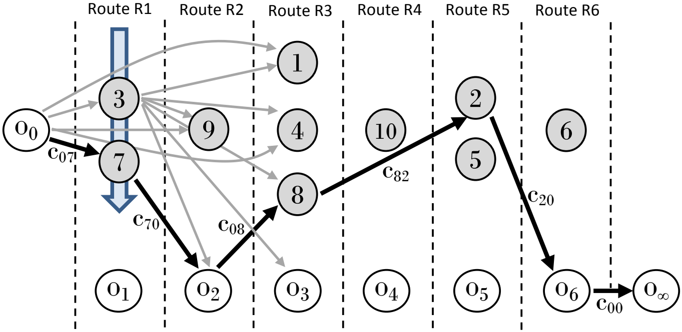


- Problems with **choices of service location** (Generalized routing problems – GVRP) are greatly simplified...
- But also, inserting a break, going to an intermediate facility, recharging electric vehicles... are many ways of choosing a mode when servicing a customer.
  - Keep in mind that in these cases, other resources than cost may be involved ⇒ RCSPs...

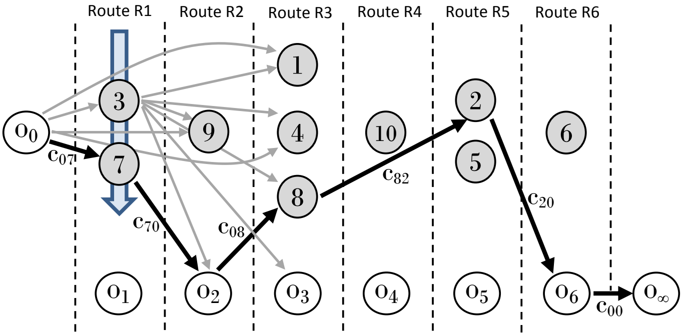
Contents

“Very very” large neighborhoods

- The concept can even be integrated into ejection chains-type neighborhoods to search an **exponential set of solutions** (visit permutations + depots) **in polynomial time** via a shortest-path formulation:



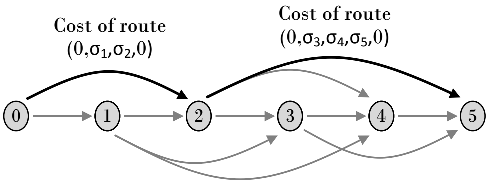
"Very very" large neighborhoods



- The cost c_{ij} of an arc (i, j) corresponds to the difference of cost of $R(j)$ when *removing service* j and *inserting service* i with minimum cost in the route.

“Very very” large neighborhoods

- Using this problem decomposition and route evaluation procedure in the **“Split” algorithm** leads to **another very large neighborhood**.



- **Still in $\mathcal{O}(\mathbf{n^2})$**
- Already known as Split “with flips” from Prins et al. (2009).

Contents

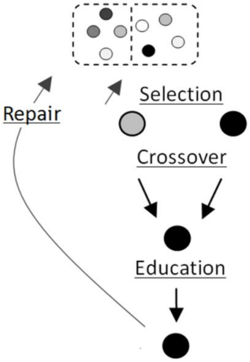
Metaheuristics

- Integration into two state-of-the-art metaheuristics:
- The iterated local search variant (ILS) of Prins (2009)
  - Produces n_C offspring from the incumbent solution and selects the best
  - Search is restarted n_P times, each run terminates after n_I consecutive iterations
  - I added the possibility to use penalized infeasible solutions (not in the original version of the algorithm).
- The unified hybrid genetic search (UHGS) of Vidal et al. (2012, 2014)

UHGS

Classic genetic algorithm components: population, selection, crossover, and

1. Efficient **local-improvement** procedure. Replaces random mutation
2. Management of **penalized infeasible solutions**
3. Individual evaluation: **solution quality** and **contribution to population diversity**



Metaheuristics

Local improvement procedure used in both methods:
Very standard neighborhoods:

- RELOCATE, SWAP, CROSS, 2-OPT and 2-OPT*.
  - Exploration in random order
  - First improvement policy
  - Restrictions of moves to K$^{\text{TH}}$ closest customers
    $\Rightarrow$ Number of neighbors in $\mathcal{O}(n)$
  - + one attempt of ejection chain on any local minimum.

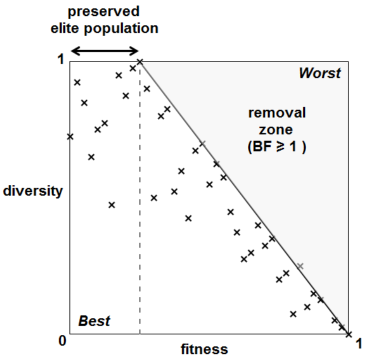
Penalized infeasible solutions:

- Simple linear combination of the excess of load, distance or other resource constraints on routes.
  - Penalty coefficients are adapted during the search.

UHGS – Biased fitness: combining ranks in terms of solution cost $C(I)$ and contribution to the population diversity $D(I)$, measured as a distance to other individuals :

$$BF(I) = C(I) + \left(1 - \frac{nbElite}{popSize - 1}\right) D(I)$$

- Used for parents selection
  - $\Rightarrow$ Balancing quality with innovation to promote a more thorough exploration of the search space.
- Used during selection of survivors
  - $\Rightarrow$ Removing individuals with worst $BF(I)$ still guarantees elitism



Experimental setting

- Literature on CARP and NEARP built around several sets of well-known benchmark instances:

	#	**Reference**	$\|N_R\|$	$\|E_R\|$	$\|A_R\|$	n	**Specificities**
CARP:							
GDB	(23)	Golden et al. (1983)	0	[11,55]	0	[11,55]	Random graphs; Only required edges
VAL	(34)	Benavent et al. (1992)	0	[39,97]	0	[39,97]	Random graphs; Only required edges
BMCV	(100)	Beullens et al. (2003)	0	[28,121]	0	[28,121]	Intercity road network in Flanders
EGL	(24)	Li and Eglese (1996)	0	[51,190]	0	[51,190]	Winter-gritting application in Lancashire
EGL-L	(10)	Brandão and E. (2008)	0	[347,375]	0	[347,375]	Larger winter-gritting application
NEARP:							
MGGDB	(138)	Bosco et al. (2012)	[3,16]	[1,9]	[4,31]	[8,48]	From CARP instances GBD
MGVAL	(210)	Bosco et al. (2012)	[7,46]	[6,33]	[12,79]	[36,129]	From CARP instances VAL
CBMix	(23)	Prins and B. (2005)	[0,93]	[0,94]	[0,149]	[20,212]	Randomly generated planar networks
BHW	(20)	Bach et al. (2013)	[4,50]	[0,51]	[7,380]	[20,410]	From CARP instances GDB, VAL, & EGL
DI-NEARP	(24)	Bach et al. (2013)	[120,347]	[120,486]	0	[240,833]	Newspaper and media product distribution

Experimental setting

- To prevent any possible over-tuning
  $\Rightarrow$ using the original parameters of the metaheuristics
- Single core: Xeon 3.07 GHz CPU with 16 GB of RAM
- Single termination criterion on all instances
  $\Rightarrow$ scaled to reach a similar CPU time as previous competitive algorithms.

Experimental setting

- For each benchmark set, we collected the best three solution methods in the literature (some are heavily tailored for specific benchmark sets).

BE08	Brandão and Eglese (2008)	HKSG12	Hasle et al. (2012)	MTY09	Mei et al. (2009)
BLMV14	Bosco et al. (2014)	LPR01	Lacomme et al. (2001)	PDHM08	Polacek et al. (2008)
BMCV03	Beullens et al. (2003)	MLY14	Mei et al. (2014)	TMY09	Tang et al. (2009)
DHDI14	Dell'Amico et al. (2014)	MPS13	Martinelli et al. (2013)	UFF13	Usberti et al. (2013)

- Comparison with the proposed metaheuristics, which are searching the space of service permutations (our methods are not fine-tuned for any of these instance sets).

Experimental setting

- Reporting the average and best solution on 10 runs.
- All Gap(%) values measured from the current best known solutions (BKS)
- Warning – time measures for some previous algorithms: using known optimal solutions to trigger termination, or reporting the time to reach the best solution
  - Dependent on exogenous information
  - Not the complete search time
- Hence, two columns for time measures:
  $\Rightarrow$ "T" for total CPU time when available,
  $\Rightarrow$ "T*" for time to reach final solution.

Comparison with previous literature

Variant	Bench.	n	Author	Runs	Avg.	Best	T	T*	CPU
CARP	GDB	[11,55]	TMY09	*30*	0.009%	0.000%	0.11	—	Xe 2.0G
			BMCV03	*1*	0.000%	—	—	0.03	P-II 500M
			MTY09	*1*	0.000%	—	—	0.01	Xe 2.0G
			ILS	*10*	0.002%	0.000%	0.16	0.03	Xe 3.07G
			UHGS	*10*	**0.000%**	**0.000%**	0.22	0.01	Xe 3.07G
	VAL	[39,97]	MTY09	*1*	0.142%	—	—	0.11	Xe 2.0G
			LPR01	*1*	0.126%	—	2.00	—	P-III 500M
			BMCV03	*1*	0.060%	—	—	1.36	P-II 500M
			ILS	*10*	0.054%	0.024%	0.68	0.16	Xe 3.07G
			UHGS	*10*	**0.048%**	**0.021%**	0.82	0.08	Xe 3.07G
	BMCV	[28,121]	BE08	*1*	0.156%	—	—	1.08	P-M 1.4G
			MTY09	*1*	0.073%	—	—	0.35	Xe 2.0G
			BMCV03	*1*	0.036%	—	2.57	—	P-II 450M
			ILS	*10*	0.027%	0.000%	0.82	0.22	Xe 3.07G
			UHGS	*10*	**0.007%**	**0.000%**	0.87	0.11	Xe 3.07G
	EGL	[51,190]	PDHM08	*10*	0.624%	—	30.0	8.39	P-IV 3.6G
			UFF13	*15*	0.560%	0.206%	13.3	—	I4 3.0G
			MTY09	*1*	0.553%	—	—	2.10	Xe 2.0G
			ILS	*10*	0.236%	0.106%	2.35	1.33	Xe 3.07G
			UHGS	*10*	**0.153%**	**0.058%**	4.76	3.14	Xe 3.07G
	EGL-L	[347,375]	BE08	*1*	4.679%	—	—	17.0	P-M 1.4G
			MPS13	*10*	2.950%	2.523%	20.7	—	I5 3.2G
			MLY14	*30*	1.603%	0.895%	33.4	—	I7 3.4G
			ILS	*10*	0.880%	0.598%	23.6	15.4	Xe 3.07G
			UHGS	*10*	**0.645%**	**0.237%**	36.5	27.5	Xe 3.07G

Comparison with previous literature

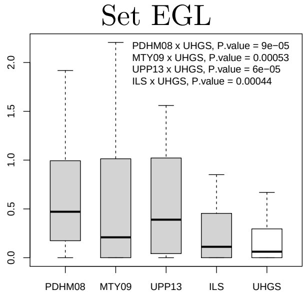
Variant	Bench.	n	Author	Runs	Avg.	Best	T	T*	CPU
NEARP	MGGDB	[8,48]	BLMV14	*1*	1.342%	—	0.31	—	Xe 3.0G
			DHDI14	*1*	0.018%	—	60.0	0.86	CPU 3G
			ILS	*10*	0.010%	0.000%	0.13	0.03	Xe 3.07G
			UHGS	*10*	**0.015%**	**0.000%**	0.16	0.01	Xe 3.07G
	MGVAL	[36,129]	BLMV14	*1*	2.620%	—	16.7	—	Xe 3.0G
			DHDI14	*1*	0.071%	—	60.0	3.69	CPU 3G
			ILS	*10*	0.067%	0.019%	1.18	0.32	Xe 3.07G
			UHGS	*10*	**0.045%**	**0.011%**	1.20	0.17	Xe 3.07G
	CBMix	[20,212]	HKSG12	*2*	—	3.076%	120	56.9	CPU 3G
			BLMV14	*1*	2.697%	—	44.7	—	Xe 3.0G
			DHDI14	*1*	0.884%	—	60.0	19.6	CPU 3G
			ILS	*10*	0.733%	0.363%	2.46	1.48	Xe 3.07G
			UHGS	*10*	**0.381%**	**0.109%**	4.56	3.08	Xe 3.07G
	BHW	[20,410]	HKSG12	*2*	—	1.949%	120	60.1	CPU 3G
			DHDI14	*1*	0.555%	—	60.0	21.4	CPU 3G
			ILS	*10*	0.440%	0.196%	5.22	2.90	Xe 3.07G
			UHGS	*10*	**0.208%**	**0.077%**	7.95	5.87	Xe 3.07G
	DI-NEARP	[240,833]	HKSG12	*2*	—	1.639%	120	93.0	CPU 3G
			DHDI14	*1*	0.536%	—	60.0	36.3	CPU 3G
			ILS	*10*	0.199%	0.084%	30.0	21.3	Xe 3.07G
			UHGS	*10*	**0.139%**	**0.055%**	29.6	16.7	Xe 3.07G

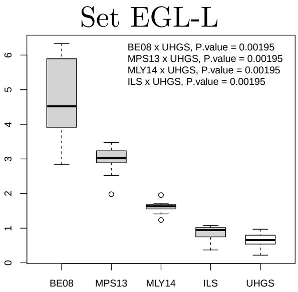
Comparison with previous literature

- New neighborhoods lead to much better solutions $\rightarrow$ even ILS already produces better solutions than previous literature
- UHGS goes further in performance $\rightarrow$ up to 0.503% and 0.958% improvement on the large instance sets
- Some BKSs for large CARP instances have been improved by up to 2.275% !
- Average standard deviation in $[0.000\%, 0.292\%]$
- On the CARP benchmark sets, 187/191 BKS have been matched or improved. 153/155 known optimal solutions were found
- For the NEARP, 408/409 BKS have been matched or improved. All 217 known optimal solutions found.

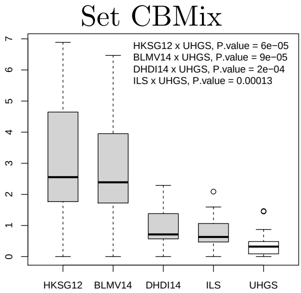
Comparison with previous literature

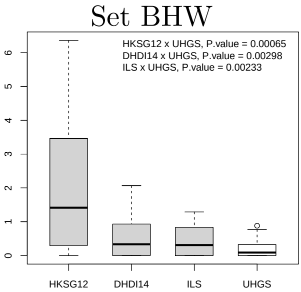
- Boxplot visualizations of Gap(%) of various methods on large-scale instances:
- Gray colors indicate a significant difference of performance, as highlighted by pairwise Wilcoxon tests with adequate correction for multiplicity

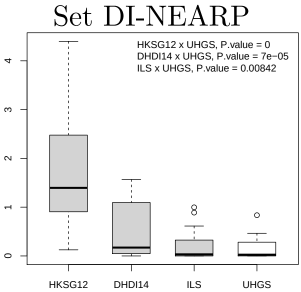




Comparison with previous literature

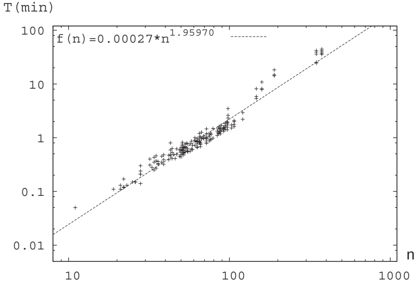


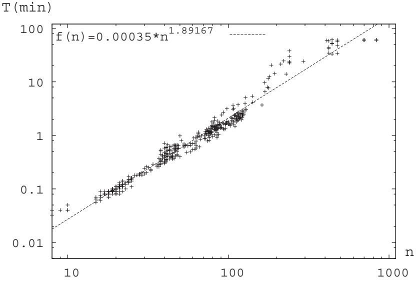




Scalability

- Growth of the CPU time of UHGS as a function of the number of services, for the CARP instances (left figure) and NEARP instances (right figure). Log-log scale.





- A linear fit, with a least square regression, has been performed on the sample after logarithmic transformation:
  $\Rightarrow$ CPU time appears to grow in $\mathcal{O}(n^2)$

To reduce or not to reduce

- Previous slides: investigated whether methods using combined neighborhoods – with optimal choices of service orientations – can outperform methods based on more traditional neighborhoods

- Now analyzing whether relying on a problem reduction from CARP to CVRP (Martinelli et al., 2013) with a classical routing metaheuristic can be profitable.

- The reduction increases the number of services by $\times 2$.
  - Half of the edges of a CVRP solution, with a large fixed negative cost, directly determine the service orientations in the associated CARP solution.

To reduce or not to reduce

- Applied the same ILS and UHGS on the transformed instances, now using a classical move evaluation for the CVRP.

	Gap(%)		T(min)	
	ILS	$\mathrm{ILS}_{\mathrm{CVRP}}$	ILS	$\mathrm{ILS}_{\mathrm{CVRP}}$
GDB	0.002%	0.000%	0.16	0.59
VAL	0.054%	0.061%	0.68	2.39
BMCV	0.027%	0.044%	0.82	2.79
EGL	0.236%	0.345%	2.35	8.50
EGL-L	0.880%	1.411%	23.6	60.0

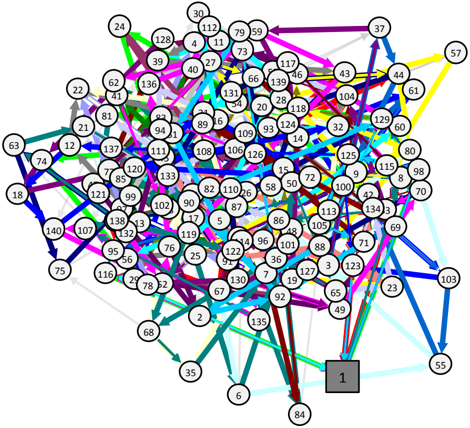
	Gap(%)		T(min)	
	UHGS	$\mathrm{UHGS}_{\mathrm{CVRP}}$	UHGS	$\mathrm{UHGS}_{\mathrm{CVRP}}$
GDB	0.000%	0.000%	0.22	0.72
VAL	0.048%	0.048%	0.82	2.98
BMCV	0.007%	0.014%	0.87	3.02
EGL	0.153%	0.200%	4.76	12.65
EGL-L	0.645%	1.001%	36.5	59.7

- Significantly lower solution quality and higher CPU time when relying on the decomposition.

- Heuristics for the CARP are worth studying...

Addressing problems with turn penalties

- Final experiment about CARP and NEARP with turn penalties
  - A **must-have** in various sectors of application, but more scarcely studied in the routing community.
- Lack of reasonable benchmark sets, previous instances based on random graphs:

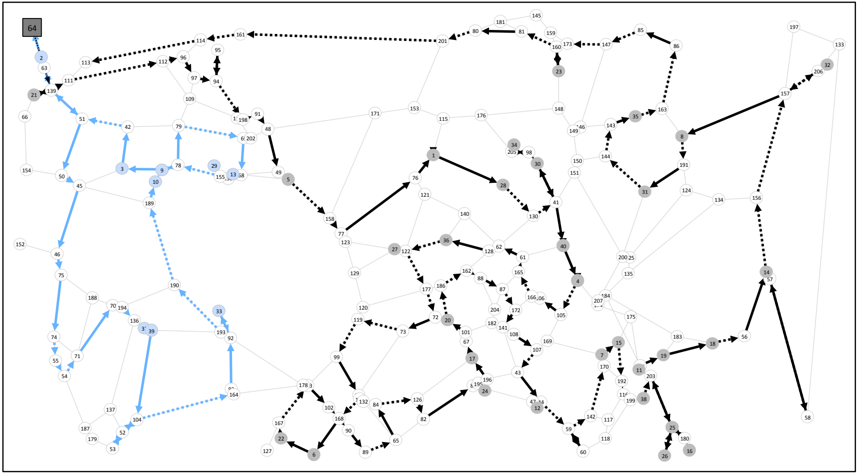


Addressing problems with turn penalties

- Hence, also generating new benchmark instances to investigate the problem
- Extension of DI-NEARP (Bach et al., 2013), adding turn penalties ⇒ 28 instances with 240–833 services.
  - Application of media products distribution in Nordic countries
  - Edge distances are available but no node coordinates
- How to produce realistic turn penalties?
  - Reconstructing a plausible planar layout for each instance, with the FM^3 algorithm of Hachul and Jünger (2005)
    ⇒ efficiently evaluates a force equilibrium, based on desired distances to obtain 2D node coordinates
  - 5γ for U-turns, 3γ for left turns, γ for intersection crossing
  - γ calibrated for turn penalties to scale to 30% of solution cost, (realistic according to analyses of Nielsen et al. 1998)

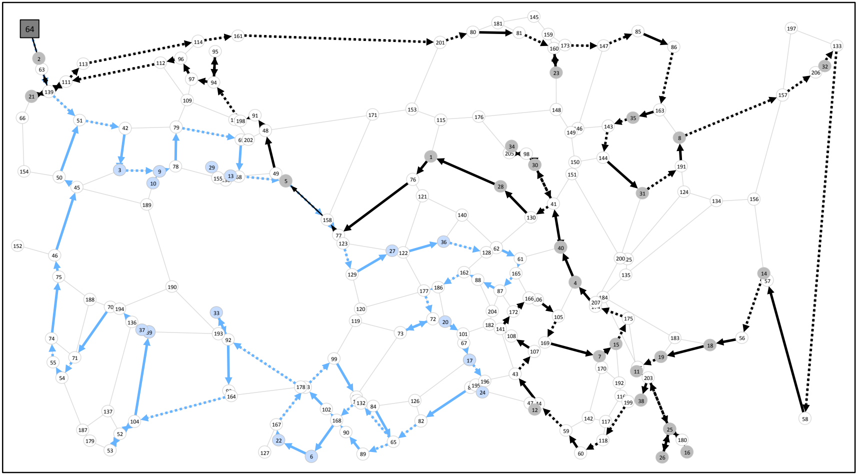
Addressing problems with turn penalties

- Sample solution with small turn penalties:
  - $\gamma = 0.25$, distance $= 4286$:



Addressing problems with turn penalties

- Sample solution with slightly larger turn penalties:
  - $\gamma = 0.5$, distance of 4336:

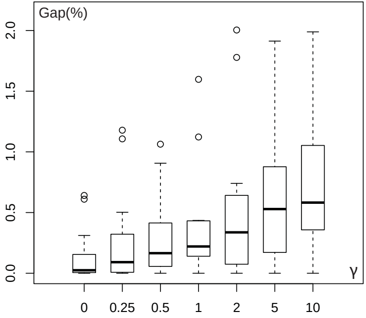


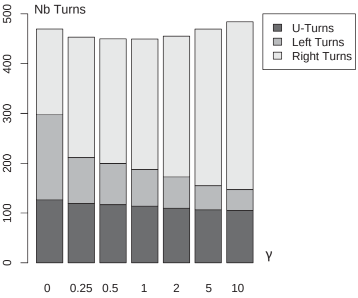
Addressing problems with turn penalties

γ	Gap (%)	T	Cost	Distance	Nb Turns			
					U-turns	Left	Right	All
0	0.141%	50.68	25076.61	25076.61	126.24	170.85	172.35	469.44
0.25	0.280%	51.32	27500.70	25164.44	119.40	91.72	241.98	453.10
0.5	0.281%	51.65	29806.22	25250.74	116.79	82.77	250.17	449.73
1	0.373%	51.74	34339.29	25451.40	113.87	73.91	261.63	449.41
2	0.511%	51.77	43103.49	25986.19	109.84	62.54	282.69	455.06
5	0.607%	51.90	68258.91	27243.48	106.31	48.52	314.51	469.34
10	0.752%	51.92	109011.41	28534.13	105.23	42.01	336.76	484.00

- To assess method performance, Gap(%) measured between average solutions and BKS produced by long runs.
- Gap and standard deviation remain moderate, usually good sign
- CPU time is moderate ($\approx$ 50min for 833 services).
  - Straightforward parallelization, or reduction of termination criterion if more speed is needed.

Addressing problems with turn penalties





- Turn penalties seem to lead to slightly more difficult problems
- Remarkable reductions of left turns or U-turns even with very small penalties.
- A few turns cannot be avoided, due to the graph topology

Contents

Conclusions

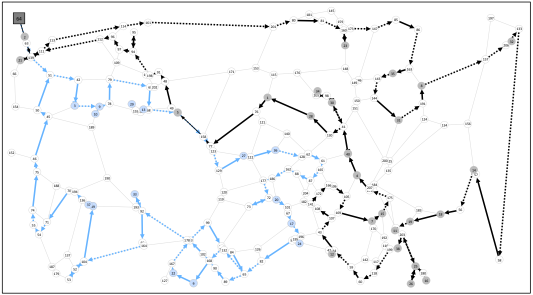
- Studied a neighborhood that was scarcely used in the past $\Rightarrow$ leads to a decomposition of problem structure, to **conceal arc routing difficulties**
- We made is efficient, systematic and general
- Interesting complexity properties
  $\rightarrow$ a kind of "free lunch".
- Many opportunities of problem generalizations
- State-of-the-art results for all known CARP and NEARP benchmark sets
- Connecting further arc and node routing worlds

Perspectives

- Open doors for research
- New instances for problems with turn penalties, challenging
- Perspectives: look for similar structural decompositions
  $\Rightarrow$ cases with more resources
  $\Rightarrow$ other combinatorial optimization problems
  $\Rightarrow$ further connections with branch-cut-price

Thank You I

THANK YOU FOR YOUR ATTENTION !



Technical report, instances, detailed results and slides available at:
`http://w1.cirrelt.ca/~vidalt/en/publications-thibaut-vidal.html`

And references after this slide...

Thank You II

Bach, L., G. Hasle, S. Wø hlk. 2013. A lower bound for the node, edge, and arc routing problem. *Computers & Operations Research* **40**(4) 943–952.

Balas, E., N. Simonetti. 2001. Linear time dynamic-programming algorithms for new classes of restricted TSPs: A computational study. *INFORMS Journal on Computing* **13**(1) 56–75.

Benavent, E., V. Campos, A. Corberán, E. Mota. 1992. The capacitated arc routing problem: lower bounds. *Networks* **22**(7) 669–690.

Beullens, P., L. Muyldermans, D. Cattrysse, D. Van Oudheusden. 2003. A guided local search heuristic for the capacitated arc routing problem. *European Journal of Operational Research* **147**(3) 629–643.

Bosco, A., D. Laganà, R. Musmanno, F. Vocaturo. 2012. Modeling and solving the mixed capacitated general routing problem. *Optimization Letters* **7**(7) 1451–1469.

Bosco, A., D. Laganà, R. Musmanno, F Vocaturo. 2014. A matheuristic algorithm for the mixed capacitated general routing problem. *Networks* **64**(4) 262–281.

Brandão, J., R. Eglese. 2008. A deterministic tabu search algorithm for the capacitated arc routing problem. *Computers & Operations Research* **35**(4) 1112–1126.

Cordeau, J.-F., M. Gendreau, G. Laporte. 1997. A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks* **30**(2) 105–119.

Thank You III

Cordeau, J.-F., G. Laporte, A. Mercier. 2001. A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational Research Society* **52**(8) 928–936.

Dell'Amico, M., G. Hasle, J.C.D. Diaz, M. Iori. 2014. An adaptive iterated local search for the mixed capacitated general routing problem. Tech. rep.

Golden, B.L., J.S. DeArmon, E.K. Baker. 1983. Computational experiments with algorithms for a class of routing problems. *Computers & Operations Research* **10**(1) 47–59.

Gutiérrez, J.C.A., D. Soler, A. Hervás. 2002. The capacitated general routing problem on mixed graphs. *Revista Investigacion Operacional* **23**(1) 15–26.

Hachul, S., M. Jünger. 2005. Drawing large graphs with a potential-field-based multilevel algorithm. J. Pach, ed., *Graph Drawing*, *LNCS*, vol. 3383. Springer, 285–295.

Hasle, G., O. Kloster, M. Smedsrud, K. Gaze. 2012. Experiments on the node, edge, and arc routing problem. Tech. rep., SINTEF, Oslo, Norway.

Irnich, S. 2008. Solution of real-world postman problems. *European Journal of Operational Research* **190**(1) 52–67.

Kokubugata, H., A. Moriyama, H. Kawashima. 2007. A practical solution using simulated annealing for general routing problems with nodes, edges, and arcs. LNCS, Springer Berlin Heidelberg, 136–149.

Thank You IV

Lacomme, P., C. Prins, W. Ramdane-Chérif. 2001. A genetic algorithm for the capacitated arc routing problem and its extensions. E.J.W. Boers, ed., *Applications of Evolutionary Computing*, lncs ed. Springer Berlin Heidelberg, 473–483.

Lacomme, P., C. Prins, W. Ramdane-Cherif. 2004. Competitive memetic algorithms for arc routing problems. *Annals of Operations Research* **131**(1-4) 159–185.

Li, L.Y.O., R.W. Eglese. 1996. An interactive algorithm for vehicle routeing for winter-gritting. *Journal of the Operational Research Society* **47**(2) 217–228.

Martinelli, R., M. Poggi, A. Subramanian. 2013. Improved bounds for large scale capacitated arc routing problem. *Computers & Operations Research* **40**(8) 2145–2160.

Mei, Y., X. Li, X. Yao. 2014. Cooperative co-evolution with route distance grouping for large-scale capacitated arc routing problems. *IEEE Transactions on Evolutionary Computation* **18**(3) 435–449.

Mei, Y., K. Tang, X. Yao. 2009. A global repair operator for capacitated arc routing problem. *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics* **39**(3) 723–734.

Mester, D., O. Bräysy. 2007. Active-guided evolution strategies for large-scale capacitated vehicle routing problems. *Computers & Operations Research* **34**(10) 2964–2975.

Muyldermans, L., P. Beullens, D. Cattrysse, D. Van Oudheusden. 2005. Exploring Variants of 2-Opt and 3-Opt for the General Routing Problem. *Operations Research* **53**(6) 982–995.

Thank You V

Nagata, Y., O. Bräysy. 2009. Edge assembly-based memetic algorithm for the capacitated vehicle routing problem. *Networks* **54**(4) 205–215.

Nielsen, O.A., R.D. Frederiksen, N. Simonsen. 1998. Using expert system rules to establish data for intersections and turns in road networks. *International Transactions in Operational Research* **5**(6) 569–581.

Pandi, R., B. Muralidharan. 1995. A capacitated general routing problem on mixed networks. *Computers & Operations Research* **22**(5) 465–478.

Polacek, M., K.F. Doerner, R.F. Hartl, V. Maniezzo. 2008. A variable neighborhood search for the capacitated arc routing problem with intermediate facilities. *Journal of Heuristics* **14**(5) 405–423.

Prins, C. 2009. A GRASP - evolutionary local search hybrid for the vehicle routing problem. F.B. Pereira, J. Tavares, eds., *Bio-inspired algorithms for the vehicle routing problem*. Springer, 35–53.

Prins, C., S. Bouchenoua. 2005. A memetic algorithm solving the VRP, the CARP, and more general routing problems with nodes, edges and arcs. W. Hart, N. Krasnogor, J. Smith, eds., *Recent advances in memetic algorithms*. Springer, 65–85.

Prins, C., N. Labadi, M. Reghioui. 2009. Tour splitting algorithms for vehicle routing problems. *International Journal of Production Research* **47**(2) 507–535.

Tang, K., Y. Mei, X. Yao. 2009. Memetic algorithm with extended neighborhood search for capacitated arc routing problems. *IEEE Transactions on Evolutionary Computation* **13**(5) 1151–1166.

Thank You VI

Tarantilis, C.D. 2005. Solving the vehicle routing problem with adaptive memory programming methodology. *Computers & Operations Research* **32**(9) 2309–2327.

Usberti, F.L., P.M. França, A.L.M. França. 2013. GRASP with evolutionary path-relinking for the capacitated arc routing problem. *Computers & Operations Research* **40**(12) 3206–3217.

Vidal, T., T.G. Crainic, M. Gendreau, N. Lahrichi, W. Rei. 2012. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research* **60**(3) 611–624.

Vidal, T., T.G. Crainic, M. Gendreau, C. Prins. 2014. A unified solution framework for multi-attribute vehicle routing problems. *European Journal of Operational Research* **234**(3) 658–673.