Solving a Complex Waste Collection Routing Problem with Intermediate Disposals

Iliya Markov[1,2], Sacha Varone[2], Michel Bierlaire[1]

[1]Transport and Mobility Laboratory
School of Architecture, Civil and Environmental Engineering
École Polytechnique Fédérale de Lausanne

[2]Haute École de Gestion de Genève
University of Applied Sciences Western Switzerland // HES-SO

hEART 2014

3rd Symposium of the European Association
for Research in Transportation

Leeds, September 10-12, 2014

Overview

Contents

Problem description

- A heterogeneous fixed fleet with different:
  - volume capacities
  - weight capacities
  - fixed costs
  - unit-distance running costs
  - unit-time driver wage rates
  - speeds
  - site dependencies (accessibility constraints)

Problem description

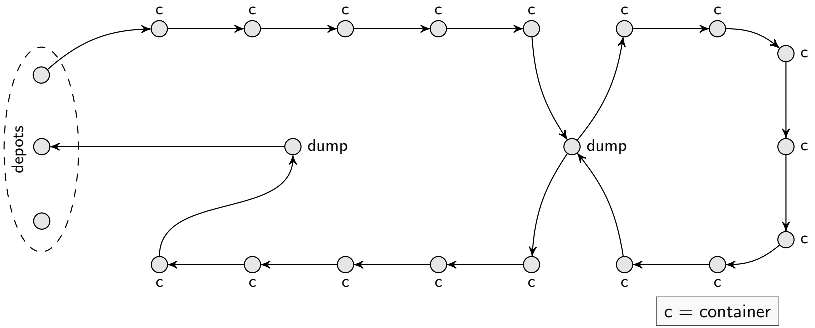
- A heterogeneous fixed fleet with different:
  - volume capacities
  - weight capacities
  - fixed costs
  - unit-distance running costs
  - unit-time driver wage rates
  - speeds
  - site dependencies (accessibility constraints)
- A set of depots
- A set of containers placed at collection points with time windows
- A set of dumps (recycling plants) with time windows

Problem description

- A heterogeneous fixed fleet with different:
  - volume capacities
  - weight capacities
  - fixed costs
  - unit-distance running costs
  - unit-time driver wage rates
  - speeds
  - site dependencies (accessibility constraints)
- A set of depots
- A set of containers placed at collection points with time windows
- A set of dumps (recycling plants) with time windows
- Maximum tour duration, interrupted by a break
- A tour is a sequence of collections and disposals at the available dumps, with a mandatory disposal before the end of the tour
- A tour need not finish at the depot it started from

Problem description

Figure 1: Tour illustration



Contents

Intermediate facilities

- VRP with satellite facilities (Bard et al., 1998)
  - no time windows, no driver break, homogeneous fleet
  - branch-and-cut
- Waste collection VRP (Kim et al., 2006)
  - time windows, driver break, homogeneous fleet
  - simulated annealing
  - Ombuki-Berman et al. (2007) (GA), Benjamin (2011) (VNTS), Buhrkal et al. (2012) (ALNS) improve results by 15-16%
- MDVRPI (Crevier et al., 2007)
  - no time windows, no driver break, homogeneous fleet at single depot
  - SP on a pool of single-depot, multi-depot and inter-depot routes
  - Tarantilis et al. (2008) (h-GLS), Hemmelmayr et al. (2013) (VNS) improve results by 1-3%
  - Muter et al. (2014): branch-and-price, solve sub-instances with 24, 40 and 50 customers

Electric and alternative fuel vehicles

- Recharging VRP (Conrad and Figliozzi, 2011)
  - recharging at customer sites with time windows, homogeneous fleet
  - mathematical model, derived solution bounds
- Green VRP (Erdoğan and Miller-Hooks, 2012)
  - maximum tour duration, no time windows, homogeneous fleet
  - two construction heuristics and an improvement procedure
- E-VRPTW with recharging stations (Schneider et al., 2014a)
  - hierarchical objective, variable recharging times, TW, homog. fleet
  - hybrid VNS/TS
  - improve the results of Erdoğan and Miller-Hooks (2012) by 8-15%
- VRP with intermediate stops (Schneider et al., 2014b)
  - combination of recharging and reloading decisions
  - weighted objective, max tour duration, no time windows, homog. fleet
  - ALNS
  - improve the E-VRPTW results of Schneider et al. (2014a)

Other VRP related

- Heterogeneous fixed fleet VRP (HFFVRP)
  - proposed by Taillard (1996)
  - best exact solutions by Baldacci and Mingozzi (2009)
  - best heuristic solutions by Subramanian et al. (2012) and Penna et al. (2013)
- Flexible assignment of start and end depot
  - Kek et al. (2008): a case study in Singapore finds significant benefits

Contributions

- Integration of dynamic start and end depot assignment into VRP-IF
  - consideration of relocation costs
- Integration of heterogeneous fixed fleet into VRP-IF
  - challenges posed by intermediate facility visits
- Integration of other side constraints
- Benchmarking to several classes of simpler problems from the literature and state of practice
  - MDVRPI (Crevier et al., 2007)
  - HFFVRP (Taillard, 1996)
  - optimal solutions, state of practice, etc...

Contents

Sets

O'	$=$ set of origins	O''	$=$ set of destinations
D	$=$ set of dumps	P	$=$ set of containers
N	$= O' \cup O'' \cup D \cup P$		
K	$=$ set of vehicles		

Parameters

π_{ij}	$=$ length of edge (i, j)
α_{ijk}	$=$ 1 if edge (i, j) is accessible for vehicle k, 0 otherwise
τ_{ijk}	$=$ travel time of vehicle k on edge (i, j)
ϵ_i	$=$ service duration at point i
$[\lambda_i, \mu_i]$	$=$ time window lower and upper bound at point i
H	$=$ maximum tour duration
η	$=$ maximum continuous work limit after which a break is due
δ	$=$ break duration
ρ_i^v, ρ_i^w	$=$ volume and weight pickup quantity at point i
Ω_k^v, Ω_k^w	$=$ volume and weight capacity of vehicle k
ϕ_k	$=$ fixed cost of vehicle k
β_k	$=$ unit-distance running cost of vehicle k
θ_k	$=$ unit-time wage rate of vehicle k

Decision variables: binary

$$x_{ijk} = \begin{cases} 1 & \text{if vehicle } k \text{ traverses edge } (i,j) \\ 0 & \text{otherwise} \end{cases}$$

$$b_{ijk} = \begin{cases} 1 & \text{if vehicle } k \text{ takes a break on edge } (i,j) \\ 0 & \text{otherwise} \end{cases}$$

$$y_{k} = \begin{cases} 1 & \text{if vehicle } k \text{ is used} \\ 0 & \text{otherwise} \end{cases}$$

Decision variables: continuous

S_{ik} = start-of-service time of vehicle k at point i

Q_{ik}^{v} = cumulative volume on vehicle k at point i

Q_{ik}^{w} = cumulative weight on vehicle k at point i

$$\text{Min} \quad f = \sum_{k\in K}\left(\phi_k y_k + \beta_k \sum_{i\in N}\sum_{j\in N}\pi_{ij}x_{ijk} + \theta_k\left(\sum_{j\in O''}S_{jk} - \sum_{i\in O'}S_{ik}\right)\right) \tag{1}$$

$$\text{s.t.} \quad \sum_{k\in K}\sum_{j\in D\cup P} x_{ijk} = 1, \qquad \forall i \in P \tag{2}$$

$$\sum_{i\in O'}\sum_{j\in N} x_{ijk} = y_k, \qquad \forall k \in K \tag{3}$$

$$\sum_{i\in D}\sum_{j\in O''} x_{ijk} = y_k, \qquad \forall k \in K \tag{4}$$

$$\sum_{i\in N} x_{ijk} = 0, \qquad \forall k \in K, j \in O' \tag{5}$$

$$\sum_{j\in N} x_{ijk} = 0, \qquad \forall k \in K, i \in O'' \tag{6}$$

$$\sum_{i\in N\setminus O''} x_{ijk} = \sum_{i\in N\setminus O'} x_{jik}, \qquad \forall k \in K, j \in D\cup P \tag{7}$$

$$x_{ijk} \leqslant \alpha_{ijk}, \qquad \forall k \in K, i \in N\setminus O'', j \in N\setminus O' \tag{8}$$

$$
\begin{aligned}
\text{s.t.}\quad & Q^v_{ik} \leqslant \Omega^v_k, && \forall k \in K, i \in P && (9)\\
& Q^w_{ik} \leqslant \Omega^w_k, && \forall k \in K, i \in P && (10)\\
& Q^v_{ik} = 0, && \forall k \in K, i \in N \setminus P && (11)\\
& Q^w_{ik} = 0, && \forall k \in K, i \in N \setminus P && (12)\\
& Q^v_{ik} + \rho^v_j \leqslant Q^v_{jk} + \left(1 - x_{ijk}\right) M, && \forall k \in K, i \in N \setminus O'', j \in P && (13)\\
& Q^w_{ik} + \rho^w_j \leqslant Q^w_{jk} + \left(1 - x_{ijk}\right) M, && \forall k \in K, i \in N \setminus O'', j \in P && (14)\\
& S_{ik} + \epsilon_i + \delta b_{ijk} + \tau_{ijk} \leqslant S_{jk} + \left(1 - x_{ijk}\right) M, && \forall k \in K, i \in N \setminus O'', j \in N \setminus O' && (15)\\
& \left(S_{ik} - \sum_{m \in O'} S_{mk}\right) + \epsilon_i - \eta \leqslant \left(1 - b_{ijk}\right) M, && \forall k \in K, i \in N \setminus O'', j \in N \setminus O' && (16)\\
& \eta - \left(S_{jk} - \sum_{m \in O'} S_{mk}\right) \leqslant \left(1 - b_{ijk}\right) M, && \forall k \in K, i \in N \setminus O'', j \in N \setminus O' && (17)\\
& b_{ijk} \leqslant x_{ijk}, && \forall k \in K, i, j \in N && (18)\\
& \left(\sum_{j \in O''} S_{jk} - \sum_{i \in O'} S_{ik}\right) - \eta \leqslant \left(\sum_{\substack{i \in N \setminus O'' \\ j \in N \setminus O'}} b_{ijk}\right) M, && \forall k \in K && (19)
\end{aligned}
$$

$$\text{s.t.} \quad \lambda_i \sum_{j \in N \setminus O'} x_{ijk} \leqslant S_{ik} \leqslant \mu_i \sum_{j \in N \setminus O'} x_{ijk}, \qquad \forall k \in K, i \in N \setminus O'' \tag{20}$$

$$\sum_{j \in O''} S_{jk} - \sum_{i \in O'} S_{ik} \leqslant \mathrm{H}, \qquad \forall k \in K \tag{21}$$

$$x_{ijk}, y_k, b_{ijk} \in \{0, 1\}, \qquad \forall k \in K, i, j \in N \tag{22}$$

$$Q^v_{ik}, Q^w_{ik}, S_{ik} \geqslant 0, \qquad \forall k \in K, i \in N \tag{23}$$

Extension: Relocation decisions

$$z_{ijk} = \begin{cases} 1 & \text{if } i \text{ is the origin and } j \text{ the destination of vehicle } k \\ 0 & \text{otherwise} \end{cases}$$

Ψ = weight of relocation term

$$\text{Min} \quad f = \text{Objective (1)} + \Psi \sum_{k \in K} \sum_{i \in O'} \sum_{j \in O''} \left(\beta_k \pi_{ji} + \theta_k \tau_{jik}\right) z_{ijk} \tag{24}$$

s.t. Constraints (2) to (23)

$$\sum_{m \in P} x_{imk} + \sum_{m \in D} x_{mjk} - 1 \leqslant z_{ijk}, \qquad \forall k \in K, i \in O', j \in O'' \tag{25}$$

$$z_{ijk} = \{0, 1\}, \qquad \forall k \in K, i \in O', j \in O'' \tag{26}$$

Contents

Contents

Speed-up rules

Zero fixing

$$\text{s.t.}\quad x_{iik} = 0, \qquad \forall k \in K, i \in N \tag{27}$$

$$x_{ijk} = 0, \qquad \forall k \in K, i \in O', j \in D \cup O'' \tag{28}$$

$$x_{ijk} = 0, \qquad \forall k \in K, i \in P, j \in O'' \tag{29}$$

$$x_{ijk} = 0, \qquad \forall k \in K, i \in D, j \in D\colon i \neq j \tag{30}$$

Time windows

$$\text{s.t.}\quad x_{ijk} = 0, \qquad \forall k \in K, i \in P \cup D, j \in P \cup D\colon \lambda_i + \epsilon_i + \tau_{ijk} > \mu_j \tag{31}$$

Start-of-service time bounds

$$\text{s.t.}\quad \sum_{j \in O''} S_{jk} - \sum_{i \in O'} S_{ik} \geqslant \min_{\substack{m_1 \in O' \\ m_2 \in P \\ m_3 \in D \\ m_4 \in O''}} \left(\tau_{m_1 m_2 k} + \epsilon_{m_2} + \tau_{m_2 m_3 k} + \epsilon_{m_3} + \tau_{m_3 m_4 k}\right) y_k, \qquad \forall k \in K \tag{32}$$

$$S_{ik} \leqslant \max_{m \in P} \left(\mu_m - \tau_{imk}\right) y_k, \qquad \forall k \in K, i \in O' \tag{33}$$

$$S_{jk} \geqslant \min_{m \in D} \left(\lambda_m + \epsilon_m + \tau_{mjk}\right) \sum_{m \in D} x_{mjk}, \qquad \forall k \in K, j \in O'' \tag{34}$$

Speed-up rules

Symmetry breaking

$K' \subset K$ represent a subset of identical vehicles
$k'_g \in K'$
$g \in 1, \ldots, |K'|$ introduces a simple ordering of the elements of K'.

$$\text{s.t.} \quad \sum_{i \in P} \sum_{j \in P \cup D} \rho_i^v x_{ijk'_g} \geqslant \sum_{i \in P} \sum_{j \in P \cup D} \rho_i^v x_{ijk'_{g+1}}, \qquad \forall g \in 1, \ldots, (|K'| - 1) \qquad (35)$$

$$\sum_{i \in P} \sum_{j \in P \cup D} \rho_i^w x_{ijk'_g} \geqslant \sum_{i \in P} \sum_{j \in P \cup D} \rho_i^w x_{ijk'_{g+1}}, \qquad \forall g \in 1, \ldots, (|K'| - 1) \qquad (36)$$

Dump visits

$$\text{s.t.} \quad \sum_{i \in P} x_{ijk} \leqslant 1, \qquad \forall k \in K, j \in D \qquad (37)$$

$$\sum_{i \in D} \sum_{j \in P} x_{ijk} \leqslant |D| - 1, \qquad \forall k \in K \qquad (38)$$

Contents

Construction

- Feasibility is defined in terms of four criteria:
  - time-window feasibility } temporal feasibility
  - duration feasibility } temporal feasibility
  - capacity feasibility
  - accessibility feasibility

Algorithm 1: Temporal feasibility algorithm

Data: tour k as a sequence of points $1, \ldots, n$ after a change
Result: start-of-service times, waiting times and temporal feasibility of tour k

```
set S_1k to earliest possible;
for i = 2...n in tour k do
    // Calculate tentative start-of-service times
    S_ik = S_(i-1)k + ε_(i-1) + τ_(i-1)ik;
    // Insert break
    if S_(i-1)k < S_1k + η and S_ik + ε_i > S_1k + η then
        S_ik = S_ik + δ;
    end
    // Calculate waiting times
    if S_ik < λ_i then
        w_ik = λ_i − S_ik;
        S_ik = λ_i;
    else
        w_ik = 0;
    end
end
```

set S_{1k} to earliest possible;
for $i = 2 \ldots n$ in tour k **do**
- // Calculate tentative start-of-service times
- $S_{ik} = S_{(i-1)k} + \epsilon_{i-1} + \tau_{(i-1)ik}$;
- // Insert break
- **if** $S_{(i-1)k} < S_{1k} + \eta$ **and** $S_{ik} + \epsilon_i > S_{1k} + \eta$ **then**
  - $S_{ik} = S_{ik} + \delta$;
- **end**
- // Calculate waiting times
- **if** $S_{ik} < \lambda_i$ **then**
  - $w_{ik} = \lambda_i - S_{ik}$;
  - $S_{ik} = \lambda_i$;
- **else**
  - $w_{ik} = 0$;
- **end**

end

Algorithm 1: Temporal feasibility algorithm, cont'd

```
// Check time window feasibility
if S_ik ≤ μ_i, ∀i then
    // Forward time slack reduction
    for i = n...2 in tour k do
        S'_(i−1)k = S_(i−1)k;
        S_(i−1)k = min(S_(i−1)k + w_ik, μ_(i−1));
        w_(i−1)k = w_(i−1)k + (S_(i−1)k − S'_(i−1)k);
        w_ik = w_ik − (S_(i−1)k − S'_(i−1)k);
    end
    w_1k = 0;
    // Check duration feasibility
    if S_nk − S_1k ≤ H then
        tour k is temporally feasible;
    else
        tour k is (duration) infeasible;
    end
else
    tour k is (time-window) infeasible;
end
```

Construction

- Feasibility is defined in terms of four criteria:
  - time-window feasibility
  - duration feasibility
  - capacity feasibility
  - accessibility feasibility

(time-window feasibility and duration feasibility } temporal feasibility)

Construction

- Feasibility is defined in terms of four criteria:
  - time-window feasibility } temporal feasibility
  - duration feasibility } temporal feasibility
  - capacity feasibility
  - accessibility feasibility
- Tour construction – feasibility preserving greedy insertion:
  - Tours are constructed sequentially, starting with the cheapest truck
  - At every iteration an unassigned container is inserted at the point that yields the smallest increase in the objective value
  - When container insertions would violate capacity, a dump is inserted using the same logic as long as it recovers capacity feasibility
  - A dump insertion should allow for at least one subsequent temporally feasible container insertion

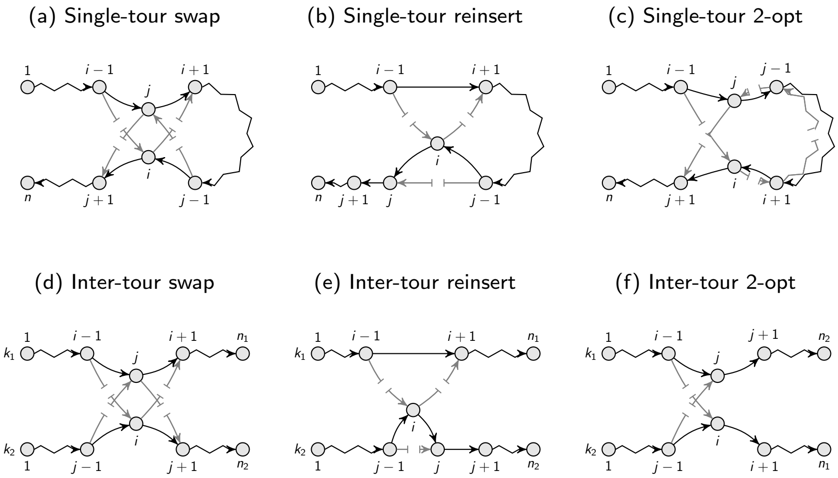
Algorithm 2: Feasibility preserving greedy insertion

Define: P is a set of unassigned containers; D is a set of dumps
Data: seed tour k as cheapest sequence of origin, container, dump, destination
Result: tour k as a sequence of points I

```
while P ≠ ∅ do
    // Best container insertion
    [bestP, bestI] = argmin(p, i){insertCost(p, i) | ∀p ∈ P, i ∈ I : insertIsFeasible(p, i)};
    if ∃ bestP then
        insert(bestP, bestI); P = P \ bestP;
    else
        // Best dump insertion
        [bestD, bestI] = argmin(d, i){insertCost(d, i) | ∀d ∈ D, i ∈ I : insertIsFeasible(d, i)};
        if ∃ bestD then
            insert(bestD, bestI);
            // Check after-dump container insertion
            [bestP′, bestI′] = argmin(p, i){insertCost(p, i) | ∀p ∈ P, i ∈ I : insertIsFeasible(p, i)};
            if ∃ bestP′ then
                insert(bestP′, bestI′); P = P \ bestP′;
            else
                remove(bestD);
                break;
            end
        else
            break;
        end
    end
end
```

- Tour improvement - local search admitting intermediate infeasibility:
  - The cost of an infeasible solution is multiplied by *infPenalty*; the latter is increased by *infStepUp* for an infeasible incumbent, and decreased by *infStepDown* for a feasible incumbent
  - Three neighborhoods - swap, reinsert and 2-opt, each applying single- and inter-tour operators
  - Vehicle reassignment evaluations, origin-destination reassignment evaluations, and capacity recovery are performed at every *recoverFreq* number of iterations
  - Capacity recovery reassigns dumps in case capacity feasibility is violated, or there are too many dumps in a tour
  - A solution with the same objective value is not admitted more than once for every *cycleFreq* number of iterations
  - The resulting tour schedule is the best found during all iterations

Figure 2: Neighborhood operators



Algorithm 3: Local search heuristic

Define: K is the set of all available vehicles
Data: set of constructed tours $K' \in K$
Result: set of improved tours $K'' \in K$

```
setBanList();
setNeighborhood(); resetCurrentNeighbor();
for maxIter do
    for maxOpIter do
        N = generateNeighborSample();
        currentNeighbor = min(n){cost(n) | ∀n ∈ N : cost(n) ∉ banList};
        updateBanList();
        if reached recoverFreq then
            reassignVehiclesRecoverCapacity();
            improveIndividually();
            updateBanList();
        end
        if reached maxOpNonImpIter then
            changeNeighborhood(); resetCurrentNeighbor();
            break;
        end
        changeNeighborhood(); resetCurrentNeighbor();
    end
    if reached maxNonImpIter then
        break;
    end
end
```

Contents

Table 1: Randomly generated instances (10 runs per instance)

Instance	# of tours	Heuristic: Objective average	Heuristic: Runtime avg (s.)	Solver: Objective	Solver: LB avg	Solver: MIP gap avg(%)	Solver: Relaxation	Solver: Runtime avg (s.)	Opt gap(%)
i1	1	214.85	0.25	214.85	214.85	0.00	11.25	688.69	0.00
i1_wtw	1	252.83	0.19	252.83	252.83	0.00	95.63	1.97	0.00
i1_ntw	2	394.82	0.44	394.82	394.82	0.00	169.30	0.59	0.00
i2	1	249.32	0.21	249.32	249.30	0.01	58.79	778.58	0.00
i2_wtw	1	257.58	0.17	257.58	257.58	0.00	119.75	2.01	0.00
i2_ntw	2	439.77	0.65	439.77	439.77	0.00	217.32	2.01	0.00
i3	1	240.13	0.21	240.13	240.12	0.01	14.93	1724.26	0.00
i3_wtw	1	245.46	0.17	245.46	245.46	0.00	45.63	2.28	0.00
i3_ntw	2	444.59	0.59	444.59	444.59	0.00	76.17	1.22	0.00
i4	1	138.64	0.16	138.64	138.64	0.00	4.08	2720.74	0.00
i4_wtw	1	140.20	0.20	140.20	140.20	0.00	4.08	5.73	0.00
i4_ntw	1	179.54	0.21	179.54	179.54	0.00	19.99	1.79	0.00
i5	1	220.77	0.21	220.77	220.76	0.01	37.89	1404.74	0.00
i5_wtw	1	233.21	0.17	233.21	233.21	0.00	83.94	1.48	0.00
i5_ntw	2	405.62	0.57	405.62	405.62	0.00	105.23	1.83	0.00

Table 2: MDVRPI instances of Crevier et al. (2007) (10 runs per instance)

Instance	Hemmelmayr et al. (2013) Best	Hemmelmayr et al. (2013) Average	Hemmelmayr et al. (2013) Average runtime(s.)	This work Best	This work Average	This work Average runtime(s.)	Gap best(%)	Gap average(%)
a1	1179.79	1180.57	85.20	1189.18	1202.89	21.12	0.80	1.89
b1	1217.07	1217.07	383.40	1217.07	1231.33	190.62	0.00	1.17
c1	1866.76	1867.96	1224.00	1885.57	1910.21	712.35	1.01	2.26
d1	1059.43	1059.43	94.20	1059.43	1071.19	19.33	0.00	1.11
e1	1309.12	1309.12	373.20	1309.12	1333.99	157.02	0.00	1.90
f1	1570.41	1573.05	1536.00	1576.81	1597.78	1148.62	0.41	1.57
g1	1181.13	1183.32	202.80	1186.59	1202.28	72.50	0.46	1.60
h1	1545.50	1548.61	876.60	1559.21	1571.26	531.82	0.89	1.46
i1	1922.18	1923.52	2014.80	1933.30	1956.97	1224.14	0.58	1.74
j1	1115.78	1115.78	166.80	1119.39	1139.20	66.34	0.32	2.10
k1	1576.36	1577.96	873.60	1581.23	1598.25	555.05	0.31	1.29
l1	1863.28	1869.70	2128.80	1880.93	1903.15	1435.59	0.95	1.79
a2	997.94	997.94	73.80	997.94	998.90	37.81	0.00	0.10
b2	1291.19	1291.19	384.60	1294.77	1343.87	217.86	0.28	4.08
c2	1715.60	1715.84	900.60	1731.60	1756.83	432.03	0.93	2.39
d2	1856.84	1860.92	1808.40	1863.97	1884.91	1031.17	0.38	1.29
e2	1919.38	1922.81	2958.60	1939.02	1979.30	1621.11	1.02	2.94
f2	2230.32	2233.43	4274.40	2273.17	2291.38	2451.33	1.92	2.59
g2	1152.92	1153.17	222.60	1153.21	1167.65	77.96	0.02	1.26
h2	1575.28	1575.28	939.60	1583.12	1601.21	506.46	0.50	1.65
i2	1919.74	1922.24	2515.20	1927.44	1958.01	1402.32	0.40	1.86
j2	2247.70	2250.21	4402.80	2259.99	2291.22	3056.50	0.55	1.82
Avg			1292.73			771.32	0.53	1.81

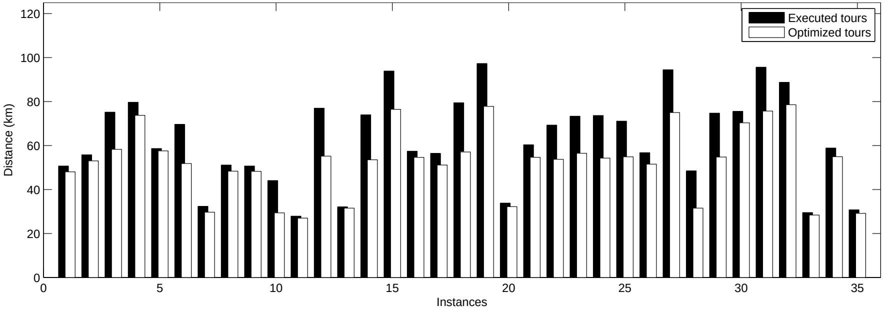
Table 3: HFFVRP instances of Taillard (1996) (10 runs per instance)

Inst-ance	Subramanian et al. (2012)			This work				
	Best	Average	Average runtime(s.)	Best	Average	Average runtime(s.)	Gap best(%)	Gap average(%)
13	3185.09	3186.32	1.99	3231.85	3257.29	28.89	1.47	2.23
14	10107.53	10110.61	1.29	10127.35	10630.97	43.73	0.20	5.15
15	3065.29	3065.29	1.77	3106.30	3117.85	27.02	1.34	1.71
16	3265.41	3273.15	1.67	3353.44	3361.73	32.81	2.70	2.71
17	2076.96	2081.55	5.95	2145.71	2180.39	82.82	3.31	4.75
18	3743.58	3758.83	16.47	3971.54	4040.45	71.59	6.09	7.49
19	10420.34	10421.05	15.80	10462.42	10778.23	199.91	0.40	3.43
20	4761.26	4822.16	16.87	4893.51	4909.26	132.60	2.78	1.81
Avg			7.73			77.42	2.28	3.66

Industry instances

- 35 instances of tours executed in the canton of Geneva
- 7 to 38 containers per tour, up to 4 dump visits per tour
- LS heuristic improvements range from 1.73% to 34.91%, on average 14.75%

Figure 3: Executed vs. optimized tours (average of 10 runs per instance)



Contents

- MILP model and local search heuristic that completely capture problem features
- Heuristic performs well compared to optimal solutions on small random instances and the state of practice
- Very good solutions compared to BKS to benchmark instances of much restricted versions of the problem at hand
- Future work:
  - model reformulation for better benchmarking
  - improvement of the vehicle reassignment evaluation procedure to obtain better results on HFFVRP instances
  - part of a larger inventory routing framework

Thank you

Questions?

Baldacci, R. and Mingozzi, A. (2009). A unified exact method for solving different classes of vehicle routing problems. *Mathematical Programming*, 120(2):347–380.

Bard, J. F., Huang, L., Dror, M., and Jaillet, P. (1998). A branch and cut algorithm for the VRP with satellite facilities. *IIE Transactions*, 30(9):821–834.

Benjamin, A. M. (2011). *Metaheuristics for the Waste Collection Vehicle Routing Problem with Time Windows*. PhD thesis, Department of Mathematical Sciences, Brunel University, London, UK.

Buhrkal, K., Larsen, A., and Ropke, S. (2012). The waste collection vehicle routing problem with time windows in a city logistics context. *Procedia-Social and Behavioral Sciences*, 39:241–254.

Conrad, R. G. and Figliozzi, M. A. (2011). The recharging vehicle routing problem. In Doolen, T. and Aken, E. V., editors, *Proceedings of the 2011 Industrial Engineering Research Conference*, Reno, NV, USA.

Crevier, B., Cordeau, J.-F., and Laporte, G. (2007). The multi-depot vehicle routing problem with inter-depot routes. *European Journal of Operational Research*, 176(2):756–773.

Erdoğan, S. and Miller-Hooks, E. (2012). A green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 48(1):100–114. Select Papers from the 19th International Symposium on Transportation and Traffic Theory.

Hemmelmayr, V., Doerner, K. F., Hartl, R. F., and Rath, S. (2013). A heuristic solution method for node routing based solid waste collection problems. *Journal of Heuristics*, 19(2):129–156.

Kek, A. G., Cheu, R. L., and Meng, Q. (2008). Distance-constrained capacitated vehicle routing problems with flexible assignment of start and end depots. *Mathematical and Computer Modelling*, 47:140–152.

Kim, B. I., Kim, S., and Sahoo, S. (2006). Waste collection vehicle routing problem with time windows. *Computers & Operations Research*, 33(12):3624–3642.

Muter, I., Cordeau, J.-F., and Laporte, G. (2014). A branch-and-price algorithm for the multidepot vehicle routing problem with interdepot routes. *Transportation Science*, 48(3):425–441.

Ombuki-Berman, B. M., Runka, A., and Hanshar, F. T. (2007). Waste collection vehicle routing problem with time windows using multi-objective genetic algorithms. In *Proceedings of the Third IASTED International Conference on Computational Intelligence*, CI '07, pages 91–97, Anaheim, CA, USA.

Penna, P., Subramanian, A., and Ochi, L. (2013). An iterated local search heuristic for the heterogeneous fleet vehicle routing problem. *Journal of Heuristics*, 19(2):201–232.

Schneider, M., Stenger, A., and Goeke, D. (2014a). The electric vehicle-routing problem with time windows and recharging stations. *Transportation Science*, in press.

Schneider, M., Stenger, A., and Hof, J. (2014b). An adaptive VNS algorithm for vehicle routing problems with intermediate stops. Technical Report LPIS-01/2014, Technical University Darmstadt, Darmstadt, Germany.

Subramanian, A., Penna, P. H. V., Uchoa, E., and Ochi, L. S. (2012). A hybrid algorithm for the heterogeneous fleet vehicle routing problem. *European Journal of Operational Research*, 221(2):285–295.

Taillard, É. D. (1996). A heuristic column generation method for the heterogeneous fleet VRP. *Publication CRT-96-13*.

Tarantilis, C. D., Zachariadis, E. E., and Kiranoudis, C. T. (2008). A hybrid guided local search for the vehicle-routing problem with intermediate replenishment facilities. *INFORMS Journal on Computing*, 20(1):154–168.