Operations Research B

Lars Burghardt, Alexander Wördekemper, Ahmad Hashemi

22.02.2017

Contents

1 Initial solution

For our initial solution we decided to use a sequential construction algorithm with a hill climbing algorithm introduced in [$HoMu12$]. We have chosen these algorithms because they seemed to produce good and fast initial solutions in the literature. We had to do some little adjustments because we could not open as many routes as we want because we have a fix number of vehicles. In line 1 we choose a random customer from a list of customers which are not assigned to a route yet and insert him into the current route. In line 2 the hill climbing algorithm tries to improve the route with the new inserted customer. In line 3 we check if the new route is feasible. If it is feasible the customer stays in the route, else the customer gets removed again. Line 5 says that if none of the left customers can be added to the route without making it infeasible the algorithm starts again with a new route. After all vehicles have their route we check if the solution is feasible. If it is feasible we choose it as our initial solution. Otherwise we restart the whole sequential construction algorithm.

In line 1 of the hill climbing algorithm we have a for loop for each pair of locations in the route. In line 2 we switch the locations if the latter location has a later upper time window. In line 3 we calculate the new cost function. The cost function is dependant of the route duration, the number of time window violations and the number of capacity violations. They are weighted with w1, w2 and w3 equal to 1. If the new route is better we keep the route in line 4, else we switch it back (line 5).

Algorithm 1 Sequential construction

```
 1: Initialize an empty solution $s$
 2: repeat
 3:     Initialize an empty route $r$
 4:     for (All unassigned customers) do
 5:         Get a random next customer $i$
 6:         Insert the customer $i$ at the end of the current route $r$
 7:         Call HC (Algorithm 2) to improve route $r$
 8:         if (route $r$ is feasible) then
 9:             Mark $i$ as inserted
10:         else
11:             Remove $i$ from route $r$
12:     Add route $r$ to solution $s$
13: until (All customers have been inserted OR $|s| = |Vehicles|$)
```

Algorithm 2 Hill climbing

```
1: Given a route $r$
2: repeat
3:     for (Each possible pair of locations) do
4:         if (The latter location is more urgent in its upper time window bound) then
5:             Swap the current two locations in $r$ to get a new route $r'$
6:             Calculate $cost(r)$ and $cost(r')$
7:             if ($cost(r) - cost(r') > 0$) then
8:                 $r \leftarrow r'$
9: until (Done) {Stop when no improvement achieved in the previous pass}
```

1.1 Different approaches

As we found out that we did not find initial solutions for larger instances we tried some different approaches to solve the problem

1.1.1 Removing the customer with the largest distance

The sequential construction algorithm adds random customers to the route and after the root gets improved by the hill climbing algorithm we remove the customer we had just added if the new route is infeasible. Instead of removing the just added customer we tried to remove the customer which needed the most time on the route. For every customer and it's destination we added the distance from the last node to the customer/destination and from the customer/destination to the next node. We removed the customer which needed the most time. In experiments we found out that this approach was no improvement compared to the one we had before.

2 Large neighborhood search

Algorithm 3 LNS $(maxSize, range, iterations, probability)$

```
1: Given a solution s
2: $current \leftarrow s$
3: for $(i \leftarrow 2; i \leq masSize - range; i \leftarrow i + 1)$ do
4:     for $(j \leftarrow 0; j \leq range; j \leftarrow j + 1)$ do
5:         for $(k \leftarrow 0; k < iterations; k \leftarrow k + 1)$ do
6:             $new \leftarrow$ Remove randomly $i + j$ customer in $current$
7:             for (All removed customer) do
8:                 Add customer to a randomly selected route $r$ in $new$
9:                 Call HC (Algorithm 2) to improve route $r$
10:            $pr \leftarrow$ random number between 0 and 1
11:            if ($new$ is feasible solution) then
12:                if $(cost(new) < cost(current)$ OR $pr < probability)$ then
13:                    $current = new$
14:                    if $(cost(current) < cost(s))$ then
15:                        $s = current$
```

3 Relevant work in the literature

ToDo

4 How to compile and run the code

ToDo

5 Experimental investigation of our approach's components

ToDo

6 Experimental investigation of our approach's performance

ToDo

7 Literature

[$HoMu$12] M. I. Hosny and C. L.Mumford, "Constructing initial solutions for the multiple vehicle pickup and delivery problem with time windows", Journal of King Saud University, Computer and Information Sciences, vol. 24, no. 1, pp. 59–69, 2012.

[$JaHe$11] S. Jain , P. Van Hentenryck, "Large neighborhood search for dial-a-ride problems", In: Principles and practice of constraint programming, Notes in computer science, vol. 6876. Springer, 2011.