

# 1 Solution Method

Describe Local search algorithms and refer some papers. These algorithms can have difficulties moving from one solution space to another, even when this is considered in the metaheuristics.

A way to deal with this could be to, refer to some paper that tries to implement some form of heuristic to search larger neighbourhoods. In this paper we proposed a heuristic is based on Shaw (1997) Large Neighbourhood Search (LNS)

## 2 ALNS on PDPMTWF

Describe here the difference between this model and standard methods like ALNS and other LNS for PDPTW

### 2.1 Request shuffling

To quickly do series of local searches and reach locally optimal solutions we have designed the following heuristics to efficiently search in local neighbourhood.

#### 2.1.1 2-opt

The 2-opt operator used in this paper is based on the 2-opt-L operator from ref: Carcassee2006. It selects a random vehicle with more than 2 orders (it breaks out after not finding any fitting vehicle after  $o$  tries, where  $o$  is the amount of orders in the problem). For the selected vehicle it divides up the route of the vehicle in 3 parts. All orders up until the index  $i$  of the vehicle route are inserted normally. Orders from the index  $j+1$  until the end of the route are also inserted normally. then finally orders from the index  $i+1$  until index  $j$  are inserted in inverse order. If the new route has a smaller cost than the original route the route the route is remembered as the new best route. When all reverses have been performed on the current route, the best route is selected as the new route. This operation is continued until no improvement can be made ie. the best possible schedule for the selected vehicle has been found.

### 2.2 Removal and reinsertion heuristics

#### 2.2.1 Remove Worst and Insert Greedy

#### 2.2.2 Remove Similar and Regret Insert

The removal heuristic used in this operator is based on Shaw (1997) with slight modifications based on our PDPMTWF problem. It removes orders that share specific similar qualities, as the basic idea is that by replacing these orders by eachother we find new, hopefully better, solutions. Another good reason to remove similar orders is that it could be advantageous to reinsert these orders together on the same vehicle since they share alot of the same properties in regards to distance, time etc. We define a relatedness factor  $r_{ij}$  which represents how much the order  $i$  is related to the order  $j$ . The lower the value of  $r_{ij}$  the more the two orders  $i$  and  $j$  are related. The relatedness of two orders we base here on the following properties: a distance property, a weight property, an overlapping timewindow property, a property indicating if the same vehicles can be used to serve each request, and finally if the orders belong to the same factory.

The relatedness factor is given by the following equation:

$$r_{ij} = \psi(D_{ij} + D_{(i+n)(j+n)}) + \omega|Q_i - Q_j| + \phi(1 - \frac{|V_i \cap V_j|}{\max(|V_i|, |V_j|)}) + \tau G_{ij} + \chi(U_{ij} + U_{(i+n)(j+n)}) \quad (1)$$

$D_{ij}$ ,  $Q_i$ , are defined in the problem formulation section and these values have been normalised to result in a value between  $[0..1]$ .  $V_i$  is the set of vehicles that can serve order  $i$ . The parameter  $G_{ij}$  is 1 if  $i$  belong to another factory than  $j$  and 0 if they belong to the same factory.  $U_{ij}$  is the timewindows at the pickup and delivery location, equal to 0 if the two orders have identical time windows and 1 if not. It corresponds to the sum of overlapping time windows divided by the overlapping span of the two time window sets. It can be formulated as follows

$$U_{ij} = 1 - \frac{\sum_{\substack{p \in \pi_i \\ o \in \pi_j \\ \frac{T_{ip} \leq T_{jo}}{T_{jo} \leq T_{ip}}}} (\min(\overline{T_{ip}}, \overline{T_{jo}}) - \max(\underline{T_{ip}}, \underline{T_{jo}}))}{\max(\max_{p \in \pi_i} \overline{T_{ip}}, \max_{o \in \pi_j} \overline{T_{jo}}) - \min(\min_{p \in \pi_i} \underline{T_{ip}}, \min_{o \in \pi_j} \underline{T_{jo}}) - \sum_{\substack{p \in \pi_i \\ o \in \pi: q_j \\ \frac{T_{ip} \geq T_{j(o-1)}}{T_{jo} \geq T_{i(p-1)}}}} (\min(\underline{T_{ip}}, \underline{T_{jo}}) - \max(\overline{T_{i(p-1)}}, \overline{T_{j(o-1)}}))} \quad (2)$$

Here  $\overline{T_{ip}}$  and  $\underline{T_{ip}}$  are the upper and lower time windows defined in the problem formulation.

Thus the relatedness measure is given a value  $0 \leq r_{ij} \leq 2\psi + \omega + \phi + \tau + \chi$ . We have chosen the following values in this paper  $\psi = 0.7$ ,  $\omega = 1.0$ ,  $\phi = 0.8$ ,  $\tau = 0.3$ ,  $\chi = 0.3$ .

### 2.2.3 Remove Randomly and insert First

Removes randomly between 2-7 orders (adjusted to the instance size) and reinserts them in randomly their first possible position. This operator is used to jump from one neighbourhood to another and is trying to search for possible solutions regardless of the cost they produce.

## References

- P. Shaw. A new local search algorithm providing high quality solutions to vehicle routing problems. *APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK*, 1997.