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 cosidered 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 shuffeling

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: Carcasse2006. It selects a random vehicle with more than 2 orders (it breaks out after not finding any fitting vehicle after o tires, 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 inserting these orders together we find new, hopefully better, solutions. 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 factor, a time factor, a factor indicating if the same vehicles can be used to serve each request, and finally if the same vehicle is currently both orders.

The relatedness factor is given by the following equation:

$$r_{ij} = \psi(D_{ij} + D_{(i+n)(j+n)}) + \chi(|t_i - t_j| + |t_{(i+n)} - t_{(j+n)}|) + \phi(1 - \frac{|V_i \cap V_j|}{max(|V_i|, |V_j|)} \quad (1)$$

All values have been normalised to have a value between [0..1]. Here V_i is the set of vehicles that can serve order i. Thus the relatedness measure $0 \leq r_{ij} \leq 2(\psi + \chi) + \phi + \omega$ We have chosen the following values in this paper $\psi = 0.7$, $\chi = 0.3$, $\phi = 0.8$, $\omega = 1.0$.

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.