A Study of Neighborhood Structures for the Multiple Depot Vehicle Scheduling Problem*

Benoît Laurent[1,2] and Jin-Kao Hao[2]

[1] Perinfo SA, Strasbourg, France
[2] LERIA, Université d'Angers, Angers, France
blaurent@perinfo.com, hao@info.univ-angers.fr

Abstract. This paper introduces the "block moves" neighborhood for the Multiple Depot Vehicle Scheduling Problem. Experimental studies are carried out on a set of benchmark instances to assess the quality of the proposed neighborhood and to compare it with two existing neighborhoods using shift and swap. The "block moves" neighborhood can be beneficial for any local search algorithm.

1 Introduction

Given $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$ a set of trips, a fleet of vehicles housed in $\mathcal{K} = \{n+1, n+2, \ldots, n+m\}$ depots, each having a limited capacity r_k, the Multiple Depot Vehicle Scheduling Problem (MDVSP) consists of determining a least-cost feasible vehicle schedule. Each trip t_i is defined by an origin and a destination with associated starting and ending times (s_i, e_i). We denote by τ_{ij}, the travel time from the end location of trip t_i to the starting location of trip t_j. Two trips are said compatible if t_j can be achieved right after t_i by the same vehicle, i.e. $e_i + \tau_{ij} \leq s_j$. Transfers without passengers are called deadhead trips. Transfers either to come from or return to the depot are the pull-out and pull-in trips respectively. The set of vehicles can also be defined by $\mathcal{V} = \{1, 2, \ldots, p\}$. The fleet is supposed to be homogeneous such that trips can be performed by any vehicle. In a valid schedule, the trips performed by each vehicle are pairwise compatible. The vehicles start from and return to their depot. Finally, the number of vehicles at each depot does not exceed the depot capacity.

The main objective of the MDVSP is to minimize the number of vehicles in use. Other objectives aim to avoid non-commercial tasks that induce traveling and/or waiting costs. They are denoted by c_{ij} for deadhead trips connecting t_i and t_j, by c_{ki} for the pull-out trip starting at depot d to reach trip t_i and by c_{jk} for the pull-in trip from t_j to return to the depot d. Note that fixed vehicle costs are added to pull-in and pull-out trips costs.

The MDVSP is NP-hard when two depots at least are considered [1]. The literature offers a panel of solution methods. Early works focused on heuristic algorithms. Exact algorithms have been proposed since the end of the 1980s (for

* Partially supported by the French Research Ministry (CIFRE No. 176/2004).

T. Stützle, M. Birattari, and H.H. Hoos (Eds.): SLS 2007, LNCS 4638, pp. 197–201, 2007.

a thorough survey, see [2] and [3]. In [4], Large Neighborhood Search (LNS) and Tabu Search (TS) were employed for the first time on the MDVSP.

The main contribution of this paper is the introduction and an in-depth study of a new neighborhood schema for the MDVSP. This new neighborhood, called "block moves", is based on the notion of ejection chains. We report comparative studies of the two neighborhoods used in the TS algorithm of [4] and of our "block moves" neighborhood.

2 Solution Approach

2.1 Decision Variables, Domains and Constraints

The set of decision variables is the set of trips $\mathcal{T}$, the domain $\mathcal{D}_i$ associated to each variable t_i corresponding to vehicles. A configuration is an assignment of vehicles in $\mathcal{V}$ to trips in $\mathcal{T}$. It can be represented as a vector of integers:

$$\sigma = (\sigma(t_1), \sigma(t_2), \ldots, \sigma(t_n)) \qquad (\sigma \in \mathcal{V}^n)$$

The search space Ω is then defined as the set of all such assignments.

To obtain a feasible configuration σ, the compatibility constraint between trips must be satisfied: $\forall\ (t_i, t_j) \in \mathcal{T}^2,\ \sigma(t_i) = \sigma(t_j),\ e_i + \tau_{ij} \leq s_j$

2.2 Evaluation Function

The evaluation function measures the quality of a solution σ. In addition to the costs previously described, it comprises a penalty term for constraint violations.

$$\forall \sigma \in \Omega, \quad f(\sigma) = w_c f_c(\sigma) + f_o(\sigma) \tag{1}$$

where $w_c > 0$ is the weight associated to the constraints violations, f_c the number of violations detected in σ, f_o the value of the objective function on σ.

2.3 Initial State

The initial solution is built by means of a greedy algorithm, relying on a Forward Checking procedure. At each step, the choice of variables follows the min-domain heuristic, ties being broken randomly. Each selected variable (a trip) is labeled by a possible vehicle with a priority given to already employed vehicles.

3 Neighborhood Structures

3.1 Existing Neighborhood Structures

We first describe the neighborhoods, $\mathcal{N}_{shift}$ and $\mathcal{N}_{swap}$, embedded in the Tabu Search of [4]. In $\mathcal{N}_{shift}$, a neighbor is obtained by transferring a trip to another vehicle. The swap move implies two trips t_i and t_j, accomplished by two different vehicles v and v'. It consists in moving t_i from v to v' and t_j from v' to v. The respective size of $\mathcal{N}_{shift}$ and $\mathcal{N}_{swap}$ is $O(np)$ and $O(n^2)$. Each time a change concerns the first or the last trip of a vehicle, we check if a transfer of the entire sequence of trips to an available vehicle of another depot would be profitable.

3.2 Block-Moves Neighborhood

The block-moves neighborhood is a parameterized structure, based on ejection chains. The principle of ejection chains was introduced in [5].

Let bl be the size of the "block", that is the number ($bl \geq 1$) of trips that will initiate the sequence of moves. Our neighborhood mechanism, called $\mathcal{N}_{bl\_moves}$, consists in moving bl consecutive trips handled by the same vehicle v to another vehicle v'. These ejection moves often cause constraints violations that will trigger repair attempts. For each conflicting trip, we scan and retain the best vehicle, that may receive it. If no such vehicle exists, the conflicts remain. As in $\mathcal{N}_{shift}$ and $\mathcal{N}_{swap}$, complete transfers of trips to vehicles belonging to other depots are evaluated. The size of $\mathcal{N}_{bl\_moves}(\sigma)$ depends on the number of vehicles running at least bl trips in the current configuration σ.

The rationale behind $\mathcal{N}_{bl\_moves}$ is the following. First, it prevents the search from being stuck in local optima because of some conflicts that could be repaired as soon as they arise. Second, behind the notion of "block moves", we aim at preserving the good properties of the configuration, typically the trips that fit well together.

4 Computational experiments

4.1 Benchmarks and Experimental Settings

Our experiments rely on the benchmarks proposed in [4] and generated as in [6]. In these instances, $n \in \{500, 1000, 1500\}$, $m \in \{4, 8\}$ and the cost incurred for the use of a vehicle is 10000. We heavily penalized constraints violations (100000 each) to discourage the exploration of infeasible regions.

$\mathcal{N}_{bl\_moves}$ is by definition parameterized by bl. In our experiments, bl ranges from 1 to 5. Enlarging this domain was considered useless since the average number of trips per vehicle is approximately equal to 4.2.

4.2 Statistical Study of the Neighborhoods

To assess the performance differences between $\mathcal{N}_{shift}$, $\mathcal{N}_{swap}$ and $\mathcal{N}_{bl\_moves}$, we computed some statistics on the neighborhood of the initial solutions issued as described in section 2. Table 1 gathers the average values (over 20 independent runs) on the instances m4n1000s0 to m4n1000s4, arbitrary chosen since all results were similar.

The second column displays the size of the neighborhoods, the two next ones the percentages of improving (Imp.) and deteriorating moves (Det.) respectively. The percentage of plateau moves, almost negligible, is omitted. The five next columns focus on the improving moves. They contain the values of the average improvement (Avg), the standard deviation (Sd), the best improvement (Best), the Hamming distance (d_H) between σ and σ' and the associated Standard deviation (Sd). The average value and the distance to the best known solution are indicated in the table caption (respectively, Avg value and Gap). These

Table 1. m4n1000 (heuristic) - Avg. Value = 2574683.67 - Gap (%) = 3.88

				Improvements				
	Size	Imp. (%)	Det. (%)	Avg	Sd	Best	d_H	Sd
shift	429000	0.41	99.34	167.50	(156.24)	1036.22	2.71	(2.52)
swap	497582	0.17	99.83	186.27	(177.59)	1161.66	3.67	(2.34)
1_moves	427052	6.64	93.30	275.09	(245.77)	2046.83	4.95	(3.04)
2_moves	323991	5.01	94.93	326.66	(289.90)	2144.54	7.15	(3.22)
3_moves	204345	4.66	95.02	321.09	(304.14)	2158.02	8.12	(3.55)
4_moves	113503	5.07	94.48	296.62	(302.21)	2094.69	8.42	(3.67)
5_moves	58289	5.54	94.16	299.03	(304.72)	2018.69	9.21	(3.73)

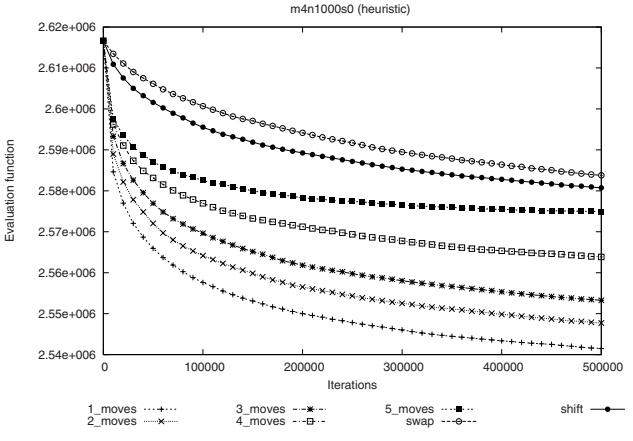

Fig. 1. Evolution of the cost function according to the neighborhood mechanism

percentages must not be compared to those of [4] in which solutions values have been purged of vehicles costs. Here, all costs are taken into account.

We observe that $\mathcal{N}_{bl\_moves}$ clearly outperforms the other neighborhoods independently of the value of *bl*. The probability of obtaining an improving neighbor in $\mathcal{N}_{bl\_moves}$ is much higher than in $\mathcal{N}_{shift}$ or $\mathcal{N}_{swap}$. Moreover, the average and best improvements procured by $\mathcal{N}_{bl\_moves}$ are about twice superior to the values related to the shift and swap neighborhoods.

To further investigate the neighborhoods behavior, we observe the profile of the evaluation function during the search of a *first improvement Descent*[1] algorithm, using the competing neighborhoods. The stop criterion is based on the number of iterations elapsed since the last improvement. This number was set to the size of the neighborhood. Figure 1 shows, for the instance m4n1000s0, the mean evolution (on 20 runs) of the evaluation function (y-axis) during a search process (x-axis). One observes that the $\mathcal{N}_{bl\_moves}$ neighborhood always lead to a better convergence with respect to the $\mathcal{N}_{shift}$ or $\mathcal{N}_{swap}$ neighborhoods.

Concerning the influence of the *bl* parameter, Figure 1 shows that it might be not worthwhile to consider blocks of size strictly greater than 3 for these

[1] A best improvement descent would be too time-comsuming.

instances. This value is certainly relative to the average number of trips per vehicle (approximately 4.2).

Finally, let us indicate making use of a portfolio of 3 simple *Descent* algorithms based on the *bl_moves* neighborhood ($bl \in \{1, 2, 3\}$), cyclically applied in a diversification scheme (see [7]), we obtained results of good quality. For example, on the instances containing 4 depots and 500 trips, the average gap from optimality is of 6.195 again 10.919 for TS [4].

5 Conclusion

In this paper, we investigated for the first time in the context of the MDVSP, a fundamental component of any local search algorithm, namely the neighborhood structure. We designed a new parameterized neighborhood schema, called "block-moves" and compared it with existing neighborhoods.

The computational study carried out on a set of artificial instances clearly shows the advantage of our "block-moves" neighborhood. A portfolio of the most effective neighborhood structures is being exploited to tackle the MDVSP with promising results. We are convinced that integrating the "block-moves" neighborhood in other frameworks can be beneficial.

References

1. Bertossi, A., Carraresi, P., Gallo, G.: On some matching problems arising in vehicle scheduling models. Networks 17, 271–281 (1987)
2. Odoni, A.R., Rousseau, J.M., Wilson, N.H.: Models in urban and air transportation, vol. 6, pp. 107–150. Elsevier Science, North-Holland, Amsterdam (1994)
3. Desaulniers, G., Hickman, M.: Public transit, pp. 69–120. Elsevier Science, North-Holland, Amsterdam (2007)
4. Pepin, A.S., Desaulniers, G., Hertz, A., Huisman, D.: Comparison of heuristic approaches for the multiple vehicle scheduling problem. Technical Report EI2006-34, Economie Institute, Erasmus University Rotterdam, Rotterdam (2006)
5. Glover, F.: Ejection chains, reference structures and alternating path methods for traveling salesman problems. Discrete Applied Mathematics 65(1-3), 223–253 (1996)
6. Carpenato, G., Dell'Amico, M., Fischetti, M., Toth, P.: A branch and bound algorihm for the multiple depot vehicle scheduling problem. Networks 19, 531–548 (1989)
7. Di Gaspero, L., Schaerf, A.: Neighborhood portfolio approach for local search applied to timetabling problems. Journal of Mathematical Modeling and Algorithms 5(1), 65–89 (2006)