Journal of Southeast University (English Edition) Vol. 21, No. 1, pp. 92 - 96 Mar. 2005 ISSN 1003—7985

Rolling horizon scheduling algorithm for dynamic vehicle scheduling system

Jia Yongji Gu Hanyu Xi Yugeng

(Institute of Automation, Shanghai Jiaotong University, Shanghai 200030, China)

Abstract: Dynamic exclusive pickup and delivery problem with time windows (DE-PDPTW), a special dynamic vehicle scheduling problem, is proposed. Its mathematical description is given and its static properties are analyzed, and then the problem is simplified as the asymmetrical traveling salesman problem with time windows. The rolling horizon scheduling algorithm (RHSA) to solve this dynamic problem is proposed. By the rolling of time horizon, the RHSA can adapt to the problem's dynamic change and reduce the computation time by dealing with only part of the customers in each rolling time horizon. Then, its three factors, the current customer window, the scheduling of the current customer window and the rolling strategy, are analyzed. The test results demonstrate the effectiveness of the RHSA to solve the dynamic vehicle scheduling problem.

Key words: dynamic vehicle scheduling; rolling horizon scheduling algorithm; exclusive; pickup and delivery problem with time windows (PDPTW)

The vehicle scheduling system is an important portion of the modern logistical system; its optimization is beneficial to reduce logistical costs and strengthen the competitive power of a company. Many researchers in the last two decades have worked on this problem and significant achievements have been achieved. However, almost all researchers have focused on the static problem in which all data are known in advance and do not change during the progress of transportation. With the development in communication and information technology, it has become feasible to deal with real dynamic data immediately. Thus, dynamic vehicle scheduling problem has become attractive to both researchers and practitioners.

Dynamic exclusive pickup and delivery problem with time windows (DE-PDPTW), a special dynamic vehicle scheduling problem, is studied in this paper. It is an NP-hard combinational optimization problem existing extensively in logistical systems. DE-PDPTW is the problem of finding optimal routes for a fleet of vehicles in order to satisfy all the transportation requests. Each vehicle departs from the depot, serves the customers along the optimal route, and then returns to the depot. Each customer request is specified by a pickup location, a delivery location and a load. The pickup location, the delivery location and the depot have specific time windows within which they can be visited. The key difference between DE-PDPTW and pickup and delivery problem with time windows (PDPTW)[1-3] is that in DE-PDPTW, once the vehicle begins to serve a customer, it cannot serve any other customers until having finished serving this customer, that is, the customer has exclusive usage of the vehicle that is serving him.

In DE-PDPTW, many unforeseen situations can arise, such as the dynamic arrival of a new customer, the breakdown of a vehicle, the change of time window, and so on. The rolling horizon scheduling algorithm (RHSA) is proposed in this paper to solve this problem. Using the rolling time horizon, the RHSA can not only adapt to the dynamic change of real problems and get a satisfactory solution but also reduce the computation time by dealing with only part of all the customers in each rolling time horizon.

1 Mathematical Description and Assumptions of DE-PDPTW

1.1 Mathematical description of DE-PDPTW

Let G be the set of customers and n be the number of customers. Each customer appoints a pickup location and a delivery location. So, the number of pickup locations or delivery locations is n respectively. Suppose the pickup location of customer i is point P_i, the delivery location is point D_i and the depot location is point 0. Time window of point i is $[a_i, b_i]$ $(a_i \leqslant b_i)$, where

Received 2004-06-24.

Foundation item: The National Natural Science Foundation of China (No. 60274013).

Biographies: Jia Yongji (1976—), male, graduate; Xi Yugeng (corresponding author), male, professor, ygxi@sjtu.edu.cn.

a_i and b_i are the earliest and latest service times, respectively (a_0, the earliest start time and b_0, the latest end time of the depot). For each customer $i \in G$, the load of weight l_i should be transported from P_i to D_i. Q is the capacity of vehicle v, V is the set of vehicles and m is the number of vehicles. For each two points i, j, $t_{i,j}$ and $c_{i,j}$ represent the travel time and the travel cost from i to j, respectively. s_i, the service time of point i, need not be considered explicitly by the algorithm, for its effects are accounted for in the traveling time $t_{i,j}$ [1]. The optimal objective of DE-PDPTW is to minimize the transportation cost.

The constraints in DE-PDPTW are as follows:

① Time window constraint. The pickup location and the delivery location must be visited within their time windows.

② Visiting constraint. The vehicle picks up loads in the pickup location and unloads in the corresponding delivery location. Each point must be served exactly once.

③ Depot constraint. The vehicle must depart from the depot to a pickup location and return to the depot from a delivery location.

④ Pairs constraint. The pickup location and the delivery location of a customer must be visited by the same vehicle.

⑤ Precedence constraint. The pickup location must be visited prior to its corresponding delivery location.

⑥ Capacity constraint. The loads in a vehicle cannot exceed the vehicle's capacity.

⑦ Exclusive constraint. When the vehicle picks up loads in the customer's pickup location, it must drive directly to the corresponding delivery location to unload.

1.2 Assumptions of DE-PDPTW

In order to simplify the problem, some assumptions are made as follows:

① A portion of or all the customers are known before the vehicle scheduling. The new customers arrive gradually in the progress of transportation and their data is known by the vehicle scheduling system once they arrive.

② The customers that are already known but have not yet been served can change. Once changed, the new data is known by the vehicle scheduling system.

③ If the vehicle has finished serving all the customers assigned to it, it must stay at the current point until new customers are assigned to it or return to the depot when the work period is over.

④ Once the vehicle begins to serve one customer, it must not leave its current destination.

⑤ There is no event influencing the vehicle and the vehicle speed is constant.

2 Static Properties of DE-PDPTW

Only a brief introduction of the static properties of DE-PDPTW is given in this section and the details are referred to Ref. [4].

Property 1 The objective function is only determined by the vehicle's empty traveling cost and is independent of its fixed cost.

Where, the empty traveling cost is the cost of the vehicle traveling without any load in order to serve another customer after having finished with the current customer, and the fixed cost is the obligatory cost of the vehicle serving a customer. In all feasible solutions, the fixed cost is constant and does not influence the objective function and can be omitted.

Property 2 The pickup location P and corresponding delivery location D of each customer can be considered as a node N. The cost between two nodes is the cost of empty traveling cost from the delivery location of the first node to the pickup location of the second one.

Property 3 The time window $[a_N, b_N]$ of node N is that if $b_P \leqslant a_D - t_N$, then $a_N = b_N = b_P$; otherwise, $a_N = \max\{a_P, a_D - t_N\}$, $b_N = \min\{b_P, b_D - t_N\}$.

Where, $[a_P, b_P]$ is the time window of pickup location, $[a_D, b_D]$ is the time window of delivery location and t_N is the traveling time from the pickup location to the delivery location.

Property 4 The distance between any two nodes N_i and N_j is asymmetrical, that is, $d_{N_iN_j} = d_{D_iP_j} \neq d_{N_jN_i} = d_{D_jP_i}$.

Therefore, DE-PDPTW can be considered as an asymmetrical traveling salesman problem with time windows: a fleet of vehicles departs from the depot and visits n nodes $N_i (i = 1, 2, \ldots, n)$ and then returns to the depot, such that each node is visited exactly once during the time window $[a_{N_i}, b_{N_i}]$.

3 RHSA to Solve DE-PDPTW

DE-PDPTW is an NP-hard problem and it is impossible to get the exact optimal solution directly. Therefore, only a heuristic algorithm is considered to attain the satisfactory solution. By referring to the fun-

damental theory of the rolling horizon method and its application to job-shop scheduling problems[5], the RHSA to solve DE-PDPTW is proposed in this paper. By the rolling of the time horizon, the algorithm can not only adapt to the dynamic change of the problem and achieve its satisfactory solution, but also reduce the computing time by dealing with only part of the customers in each rolling time horizon.

3.1 Current customer window

The rolling window, which is the core of the RHSA, is the current customer window C_w, which is the set of a certain number of customers being dealt with currently. Assume that C is the set of customers already served, while $\overline{C}$ is the set of customers not yet served but whose requests have already arrived before the current time t_c. The customer requests arriving after t_c will be added into $\overline{C}$ at their arrival time and wait to be served in the future.

At each rolling of the RHSA, only C_w is scheduled and part or all of the customers in it are assigned to vehicles to be served according to the scheduling result. The rolling of C_w is implemented by deleting the customers already served from it and adding the customers not yet served into it. When the transportation service is nearly finished, the number of customers in C_w will reduce gradually until there are no customers left.

The number of customers in C_w, the number of customers added into C and the rule to add the customers not served into C_w are the three factors that have the greatest influence on the performance of the RHSA.

3.1.1 The number of customers in C_w

Let y be the number of customers in C_w. If y is relatively large, the scale of the sub-problems to be solved at each rolling of the RHSA is also relatively large and the solution is near the optimal solution, but more computation time is needed. If y is relatively small, the scale of sub-problems is also relatively small and it is against to attain the satisfactory solution. Thus, y is a key factor to the performance of the RHSA.

3.1.2 The number of customers added into C

Let z be the number of customers added into C. If z is relatively large, that is, more customers are assigned to vehicles at each rolling, the rolling times of RHSA can decrease and the computation speed is increased but the solution quality is greatly decreased. When z is relatively small, the solution quality is increased but much more rolling times are needed. So, z is also a key factor to the performance of the RHSA.

In general, the change of y and z also have great influence on the performance of the RHSA. If y increases, it will obviously attain a better solution, for more customers' data is considered at each rolling, but the computation time also increases. If z increases, the algorithm can attain the solution faster, for the number of sub-problems decreases, but the solution quality also decreases.

In order to get the optimal performance of the RHSA, y and z must be set to optimization. However, their values are dependent on the concrete problem to be solved and can be specified only by experience and experiments.

3.1.3 Rule to add customers not served into C_w

When C_w is updated, a certain number of customers are selected and added into C_w. Generally speaking, the rules to add customers not served into C_w are as follows: ① First arrived, first served; ② Nearest to the current customer, first served; ③ Tightest in the time window, first served. Apparently, rule ① does not consider the relationship among the customers and its solution, in general, it is worse than that obtained by rule ② and rule ③. Rule ② considers the customers' space relationship while rule ③ considers the customers' time relationship. In general, the customer who is the nearest to the current customer (Its pickup location is the nearest to the delivery location of the current customer) or the tightest in the time window (Its latest service time is the tightest) should be served first.

Therefore, the priority factor of customer $N_i \in \overline{C}$ is defined as $u_i = \alpha(b_{N_i} - t_c) + \beta d_{N_i}$. Where b_{N_i} is the latest service time of N_i, t_c is the current time and d_{N_i} is the distance from the current customer to the customer N_i. α and β are the weighted coefficients to denote the time factor and the distance factor, respectively. The larger α is, the greater the weight of the time factor, while the larger β is, the greater the weight of the distance factor. $\alpha = 0$ indicates that only the distance factor, rule ②, is used to select the customers not served while $\beta = 0$ indicates that only the time factor, rule ③, is used.

When C_w is updated, in ascending order of priority factors of customers not served, the customers are selected in turn from $\overline{C}$ and added into C_w. Repeat this

process until the number of customers in C_w reaches y or there are no customers in $\overline{C}$.

3.2 Scheduling of the current customer window

At each rolling of the RHSA, the current customer window is rescheduled once, that is, a static exclusive PDPTW is solved once. In this paper, a tabu search algorithm is used to schedule the current customer window but this algorithm differs from the tabu search algorithm in Ref. [6] in the following aspects: ① Since the vehicle can serve only one customer at a time, the load must satisfy the vehicle's capacity constraint. As to the customers whose loads exceed the vehicle's capacity, they are divided into several different customers whose loads are no more than the vehicle's capacity. ② It is enough to detect only the time window $[a_N, b_N]$ of node N. ③ When the customer's pickup location is inserted, it is unnecessary to find the optimal position to insert the customer's delivery location and the delivery location is just inserted behind the corresponding pickup location.

Except for the above, the scheduling algorithm of the current customer window is the same as the one proposed in Ref. [6], and the details of the algorithm are referred to Ref. [6].

3.3 Rolling strategy

At beginning ($C = C_w = \Phi$, $\overline{C} \neq \Phi$), y customers are selected from $\overline{C}$ (If there are fewer than y customers in $\overline{C}$, select all the customers in $\overline{C}$) to compose C_w. Then C_w is scheduled and z customers are assigned to vehicles to be served according to the scheduling result. When a period passes, if z customers have already been served, they are removed from C_w to C, and other customers are selected from $\overline{C}$ and added into C_w in ascending order of customers' priority factors until the number of customers in C_w is y or there are no customers in $\overline{C}$. And then C_w is rescheduled again. This process is executing repeatedly until all the customers have been served.

The shortcoming of the rolling strategy above is that the algorithm reacts slowly to unexpected events, such as vehicle breakdown, time window changes, and so on. So, the event-driven rolling strategy[5] is used in the RHSA. Events like vehicle breakdown or time window change are defined as key events. When key events occur, the current customer window is rescheduled immediately.

4 Test

To test the performance of the RHSA, cases with real problem properties are created randomly. The program is programmed in C and tested in a PC of P4 1.6 GHz, 128 MB memory. In the tests, the customer number is 100 and the transportation cost is the traveling distance of all the vehicles and its unit is kilometer (km). Computation time is the CPU time of the scheduling algorithm and its unit is millisecond (ms). The weighted coefficients, α and β, are 1 and 10, respectively.

4.1 Flexibility of the RHSA

Tab. 1 shows the comparison between the RHSA and a static scheduling algorithm in the situation where all customers' data are known in advance and do not change during the progress of transportation. The solution obtained by the RHSA is slightly worse than the one obtained by the static scheduling algorithm, but the computation time is only 1/8 that of the static scheduling algorithm.

Tab. 1 Comparisons between RHSA and static scheduling algorithm

Algorithm	Transportation cost/km	Computation time/ms
Static scheduling algorithm	3 213	2 984
RHSA	3 248	356

Moreover, the static scheduling algorithm cannot trace the change of problems, while the RHSA can adapt to the changes by adjusting the customers' sequence and serving time according to changes of problems. Assume that the time window of a pickup location of a customer is delayed for 60 min or that a vehicle is delayed for 30 min because of a traffic jam, the static scheduling algorithm cannot trace this change and fails, while the RHSA can obtain a new solution.

4.2 Influencing factors of the RHSA

The influencing factors of the RHSA are y and z. If z remains constant, y changes, the influence of y on the performance of the RHSA is shown in Tab. 2, where the value of z is 3. If y remains constant, z changes, the influence of z on the performance of the RHSA is shown in Tab. 3, where the value of y is 10.

Tab. 2 The influence of y

y	Transportation cost/km	Computation time/ms
5	4 265	2 366
8	3 521	2 852
10	3 213	2 984
12	3 213	4 429
20	3 185	6 038

Tab. 3 The influence of z

z	Transportation cost/km	Computation time/ms
1	2 946	13 568
2	2 985	6 267
3	3 213	2 984
4	3 962	2 556
5	5 325	2 353

Tab. 2 illustrates that when z remains constant and y increases, the RHSA can find a better solution, while the computation time also increases. When y is 10, the RHSA yields the best performance. Although a better solution can be found if y is over 10, the improvement in solution is little, while the computation time increases greatly. And in Tab. 3, when y remains constant and z increases, the RHSA can find a solution quickly, although the solution quality decreases. When z is 3, the RHSA yields the best performance. Although the solution can be found more quickly if z is greater than 3, the solution quality decreases greatly. If z is less than 3, using so much computation time to get only a slightly better solution is also unsatisfactory.

5 Conclusion

The vehicle scheduling system is an important piece of the modern logistical system. DE-PDPTW, a special dynamic vehicle scheduling problem, is proposed in this paper. Its mathematical description is given and its static properties are analyzed, and then the problem is simplified as an asymmetrical traveling salesman problem with time windows. The RHSA is proposed to solve DE-PDPTW and its three factors, the current customer window, the scheduling of the current customer window and the rolling strategy, are analyzed. The test results demonstrate the effectiveness of the proposed algorithm to solve the dynamic vehicle scheduling problem.

References

[1] Nanry W P, Barnes J W. Solving the pickup and delivery problem with time windows using reactive tabu search [J]. *Transportation Research*, *Part B*, 2000, **34**B(2): 107 – 121.

[2] Lau H C, Liang Z. Pickup and delivery with time windows: algorithms and test case generation[A]. In: *Proceedings of the* 13*th IEEE International Conference on Tools with Artificial Intelligence* [C]. Dallas, Texas: IEEE Computer Society, 2001. 333 – 340.

[3] Li H, Lim A. A metaheuristic for the pickup and delivery problem with time windows [A]. In: *Proceedings of the* 13*th IEEE International Conference on Tools with Artificial Intelligence* [C]. Dallas, Texas: IEEE Computer Society, 2001. 160 – 167.

[4] Jia Yongji, Gu Hanyu, Xi Yugeng. Analysis and algorithm of single-vehicle exclusive pickup and delivery problem with time windows [J]. *Journal of Shanghai Jiaotong University* (*Natural Science Edition*), to appear in 2005, **39**(3). (in Chinese)

[5] Fang Jian, Xi Yugeng. A periodic and event-driven rolling horizon job shop scheduling strategy [J]. *Control and Decision*, 1997, **12**(3): 159 – 162. (in Chinese)

[6] Jia Yongji, Gu Hanyu, Xi Yugeng. Quick taboo search algorithm for solving PDPTW problem [J]. *Control and Decision*, 2004, **19**(1): 57 – 60. (in Chinese)

动态车辆调度系统的滚动时域调度算法

贾永基　谷寒雨　席裕庚

(上海交通大学自动化研究所, 上海 200030)

摘要: 提出了一类特殊的动态车辆调度问题——动态独占性带时间窗口装卸货问题. 给出了问题的数学描述, 分析了其静态性质, 并把问题简化为不对称带时间窗口旅行商问题. 提出了求解该动态问题的滚动时域调度算法, 通过时域的不断滚动, 不仅可以跟踪问题的动态变化, 还由于每次滚动只对部分客户进行处理, 可以减少问题的求解时间. 并分析了算法的 3 个要素: 当前客户窗口、当前客户窗口的调度和滚动策略. 测试结果验证了算法在求解动态车辆调度问题中的有效性.

关键词: 动态车辆调度; 滚动时域调度算法; 独占性; 带时间窗口装卸货问题

中图分类号: TP301