IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 4, April 2015.
www.ijiset.com

ISSN 2348 – 7968

Demand Responsive Transport

Pankaj Basnal, Vickey Singh
Computer Engineering
Army Institute of Technology
Pune, Maharashtra-411015
pakajbasnal_11421@aitpune.edu.in
vickey_11262@aitpune.edu.in

Ambikesh Shukla, Deepak Kumar
Computer Engineering
Army Institute of Technology
Pune, Maharashtra-411015
ambikeshkumarshukla_11423@aitpune.edu.in
deepakkumarsingh_11419@aitpune.edu.in

Asst. Prof. Anup Kadam
Computer Engineering
Army Institute of Technology
Pune, Maharashtra-411015
akadam@aitpune.edu.in

***Abstract*—This paper introduces a sequential construction algorithm for the Demand Responsive Transport (DRT). The algorithm utilizes a simple routing heuristic to create individual vehicle routes. Inserting a request in a route is either based on a first acceptance criterion, in which a request is inserted in the first route where a feasible insertion is found, or a best acceptance criterion, in which a request is inserted in the estimated best route for insertion.**

***Keywords*—Pickup and delivery, Vehicle routing, Demand responsive transport, Heuristics, Construction Algorithms**

I. Introduction

Demand responsive transport (DRT) is a form of shared-ride public transportation service that is responsive to the requests of passengers. The problem deals with a number of customer requests that are to be served by a fleet of vehicles, while a number of constraints must be observed. Each vehicle has a limited capacity (the capacity constraint). A vehicle route usually starts and ends at a central depot. The problem is analogous to the Dial-a-Ride problem (DARP)[10]. A request must be picked up from a pickup location to be delivered to a corresponding delivery location. Naturally, the pickup and delivery pair must be served by the same vehicle (the coupling constraint) and the pickup must precede the delivery (the precedence constraint). In addition, every request must be served within a predetermined time window interval (the time window constraint). If the vehicle arrives earlier than the allowed service time, it should wait until the beginning of the specified period. A solution to the problem should assign requests to vehicles and find a route for each vehicle, such that the total service cost is minimized and all problem constraints are adhered with.

The DARP problem is NP-hard with high complexity, and the solution algorithms can be divided into exact and heuristic approaches. It is both a grouping problem [5] (assigning requests to vehicles), and a routing problem [12] (finding the best route for each vehicle). The problem can be solved in two stages: the first stage constructs one or more initial solutions to the problem, while the second stage tries to improve these solutions using a heuristic [7] or a meta-heuristic approach.

To construct a solution for the DRT, each step of the algorithm usually selects an un-assigned request whose insertion is predicted to cause the least increase in the overall cost of the solution. The selected request is then inserted in its best (least cost) feasible insertion position found among all available routes. The rest of the paper is organized as follows: Section 2 summarizes some related work. Section 3 formally defines the model for DRT. Section 4 explains the routing algorithm embedded within the construction heuristics used. Section 5 explains details of the construction heuristics used. Finally, we conclude our work in Section 6.

II. Related Work

Solution construction can be done either sequentially or in parallel. A sequential construction builds routes one after another, while a parallel construction builds a number of routes simultaneously. To construct initial solutions sequentially, researchers usually adapted Solomons sequential insertion heuristics of the Vehicle Routing Problem with Time Windows (VRPTWs) (Solomon, 1987)[8]. A weighted sum of the extra travel distance and total time delay resulting from the insertion is often used to estimate the cost of the insertion. This type of construction was used by Li and Lim (2001) for the Multi Vehicle Routing [3], and was followed by a solution improvement phase called a tabu-embedded simulated annealing [2]. A parallel construction heuristic [1], on the other hand, was first introduced in Potvin and Rousseau (1993) for the VRPTWs. In a parallel construction, several routes are initialized with seed customers and requests are subsequently inserted into any of the initialized routes. Accordingly, the algorithm needs an initial estimate of the number of vehicles to be used. Routes are later added as needed if the initial estimate does not yield a feasible solution.

III. Model Formulation

Let G = (N, A) be a digraph. The node set is N = $\{n_i \in N | i = 0, 1, 2, ..., m\}$, such that m is an even index. The node n_0 denotes the depot, and each $n_i, i = 1, 2, ..., m$ denotes a customer location. Since for each customer request we have a pair of pickup and delivery locations, we can assume, without loss of generality, that the set $N^+ = \{n_i \in N | i = 1, 2, ..., m/2\}$ represents pickup locations, and the set $N^- = \{n_i \in N | i = (m/2) + 1, ..., m\}$ represents delivery locations, such that the pick-up location n i has the corresponding delivery location n_i + (m/2) . Thus, N = $N^+ \cup N^-$ and $|N^+| = |N^-| = m/2$. Each location n i is associated with:

1) A customer demand q_i , such that q_i ¿ 0 for a pickup location, q_i ¡ 0 for a delivery location and $q_i + q_j = 0$ for the same customers pickup and delivery locations (q_0 = 0).
2) A service time s_i (s_0 = 0), which is the time needed to load or unload a customer demand.

ISSN 2348 – 7968

3) A time window $[e_i , l_i]$ during which the location must be served, and $l_i \geq e_i$.

For each pair of nodes (n_i, n_j)a travel time t_{ij} and/or a travel distance d_{ij} are specified. Only edges satisfying the time window constraint are allowed. Thus the arc set is A = $\{(n_i , n_j) — n_i, n_j \in N, n_i \neq n_j, t_o i + s_i + t_i j < l_j\}$.

IV. ROUTING ALGORITHM

A crucial part of the DRT is the routing algorithm that will generate a feasible route for each individual vehicle. Our routing algorithm is based on an iterative improvement of individual routes, which is embedded in the overall constructive algorithm that could either be sequential or parallel. The main difference between our routing algorithm and other routing (insertion) heuristics [11] is that our algorithm does not try to find the best insertion position for each request in the route, but accepts any feasible insertion.
Our routing algorithm adopts a simple route representation. Rather than representing the visiting order of requests by a one-dimensional permutation of all the different locations, we treat both the pickup location and its associated delivery as one unit. In other words, we assign the same code (number) to both the pickup and its delivery. We then rely on a simple decoder to always identify the first occurrence as the pickup and the second as the delivery. An example of a route with four requests following this representation is: (2 1 1 3 4 2 3 4), where pickups are shown in boldface and deliveries in italics.
To deal with the hard time window constraint, our routing algorithm adopts an intelligent neighborhood [1] move that uses the time window as a guidance. The idea is to try to improve the current route by creating a new neighboring route. To avoid the frequent creation and evaluation of in feasible routes, though, our neighborhood move only swaps locations that are out of order in terms of their late time window bounds, i.e., if the latter location has a deadline that precedes the earlier one.This simple representation and neighborhood move are employed in a classical Hill-Climbing (HC) [9] route-improvement heuristic, which tries to gradually modify the current route until no further improvement is possible.
Algorithm 1 describes this simple heuristic.

Algorithm 1. The HC routing algorithm.

1) Given a route r
2) repeat
   a) for (Each possible pair of locations in r) do
   b) if (The latter location is more urgent in its upper time window bound) then
   c) Swap the current two locations in r to get a new route r_0
   d) $\Delta <$ cost(r_0) $\leftarrow$ cost(r)
   e) if ($\Delta < 0$) then
   f) r = r_0
3) until (Done) Stop when no improvement achieved in the previous pass

The cost function used in Step 6 of the HC algorithm to evaluate the quality of each route tries to minimize the total route duration as well as the degree of in feasibility in capacity and time windows constraints. The cost function of a route r is described by the following equation:

$$F(r) = w_1 \times D(r) + w_2 \times TWV(r) + w_3 \times CV(r), \quad (1)$$

where D(r) is the total route duration, including the waiting time and the service time at each location. TWV(r) is the total number of time window violations in the route, and CV(r) is the total number of capacity violations. The constants w_1, w_2, and w_3 are weights in the range [0, 1], and $w_1+w_2+w_3 = 1.0$. The choice of appropriate weights depends on the importance of each term in the objective function. We found that in order to get feasible solutions, the largest penalty should be imposed on the time window violations.

V. SOLUTION CONSTRUCTION HEUIRSTICS

In our construction heuristics we first start by sorting customers according to the distance from the depot (farthest first). However, since in our approach we deal with customers in pairs, where each pair consists of a pickup location and its associated delivery, the distance measure, in relation to the depot, could either be the distance between the depot and the pickup location, or the distance between the depot and the delivery location. We arbitrarily chose the distance separating the depot and the delivery location for the initial order of requests.

A. *The sequential construction algorithm*

The sequential construction heuristic [7] tries to build routes one after another. Requests are taken one by one in order, and each request (pickup and delivery pair) is initially inserted at the end of the current route. Our HC routing heuristic (Algorithm 1) is then called to try to improve the current route. If the HC algorithm returns an improved route that can feasibly accommodate the newly inserted pair, this insertion is accepted and we move on to the next request. However, if the improved route is still in feasible, the newly inserted pair is removed from the current route to wait for another insertion attempt in a new route.Thus, unlike the traditional insertion methods [6], our algorithm relies on the HC heuristic to improve the quality of the current route, without actually having to calculate the cost of each and every possible insertion position in order to select the best one among them.
Algorithm 2 describes the sequential construction procedure. It is important to note in Step 7 of this algorithm that, besides overcoming the precedence and the coupling issues [13], inserting a request (a pickup and delivery pair) at the end of the route has the added advantage of speeding up the insertion process, since two locations instead of one are simultaneously inserted.

Algorithm 2. The sequential construction

1) Let M $\leftarrow$ 0 { M is the number of vehicles used }
2) repeat
   a) Initialize an empty route r
   b) M = M + 1
   c) for (All unassigned requests) do
   d) Get the next unassigned request i
   e) Insert the request i at the end of the current route r

f) Call the HC routing heuristic (Algorithm 1) to improve r
g) if (r is a feasible route) then
h) Mark i as inserted
i) else
j) Remove i from r

3) until (All requests have been inserted)

VI. Numerical Results

We used Mobility Testbed (Simulator) to test our algorithms, we used several benchmarks scenario. Each scenario are of different problem sizes: 100, 200, 400, 600, 800, and 1000 customers.

TABLE I. Results

Problem size	Vehicle	Distance	Time
100-Customers	11.78	2662.9	0.02
200-Customers	17.33	8887.08	0.08
400-Customers	33.56	22215.14	0.32
600-Customers	48.22	44949.4	0.72
Average	27.7225	19678.63	0.285

Table 1 shows the average number of vehicles, the average total distance, and the average processing time (in seconds), produced by the algorithm for each problem size separately.

TABLE II. Frequency of Solution

Min-Vehicle(%)	Min-Distance(%)
48	31

Table 2 shows the percentage of time the algorithm produced the minimum number of vehicles and the minimum total distance over all problem instances.

A. The SEQ algorithm: complexity analysis and implementation issues

We present in this section some remarks concerning the complexity and feasibility checking of the SEQ algorithm in relation to the common construction methods [4]. Analyzing our SEQ algorithm we find that: the routing heuristic in Algorithm 1 needs O(n^2) time for accessing each pair of locations in the route, where n is the number of requests in the problem instance. Also, the cost function (1), which checks the feasibility of the whole route as well, needs O(n) time. The SEQ algorithm (Algorithm 2) needs O(n) since its major iteration processes all requests in order. This will make the run-time. complexity of the whole algorithm O(n^4).

VII. Conclusion

The experimental results on a large number of benchmark instances indicate that the sequential construction heuristic (SEQ) seems to be the most favorable solution construction method, which can be by easily embedded in a heuristic or a meta-heuristic technique to reach final good quality solutions. With just a few simple lines of code, and without a predetermined number of vehicles or a solution evaluation mechanism, this algorithm produced good quality results, that are sometimes even better than the results obtained by the most sophisticated parallel algorithm tested. The SEQ algorithm also had an impressive speed, with a processing time that is at most 6% of the time needed by the parallel algorithm, making it even more suitable for population-based solution algorithms .

References

[1] Bent, R.Van Hentenryck, *A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows*, Computers and Operations Research 33 (4) 875893.

[2] Hosny, M.I., 2011, *Comparing genetic algorithms and simulated annealing for solving the pickup and delivery problem with time windows*, Arabnia, H.R., de la Fuente, D., Kozerenko, E.B., Olivas, J.A. (Eds.) Proceedings of the 2011 International Conference on Artificial Intelligence, ICAI11, Vol. II. CSREA Press, Las Vegas, Nevada, USA, pp. 513519

[3] Hosny, M.I., Mumford, C.L., 2009, *New solution construction heuristics for the multiple vehicle pickup and delivery problem with time windows* MIC2009, Metaheuristic International Conference.

[4] Lu, Q., Dessouky, M., 2006 *A new insertion-based construction heuristic for solving the pickup and delivery problem with time windows* European Journal of Operational Research 175 (2), 672687.

[5] Pankratz, G., 2005. A grouping genetic algorithm for the pickup and delivery problem with time windows. OR Spectrum 27, 2124.

[6] Ropke, S., Pisinger, D., 2006 *An adaptive large neighbourhood search heuristic for the pickup and delivery problem with time windows* Transportation Science 40 (4), 455472.

[7] Hosny, M.I., Mumford, C.L., 2010 *The single vehicle pickup and delivery problem with time windows: Intelligent operators for heuristic and metaheuristic algorithms* Journal of Heuristics, Special Issue on Advances in Metaheuristics 16 (3), 417439.

[8] Solomon, M.M., 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research 35 (2), 254265

[9] Campbell, A.M., Savelsbergh, M., 2004. Efficient insertion heuristics for vehicle routing and scheduling problems. Transportation Science 38 (3), 369378.

[10] Cordeau, J.-F., Laporte, G., 2003. The dial-a-ride problem (DARP): Variants, modeling issues and algorithms. 4OR: A Quarterly Journal of Operations Research 1 (2), 89101

[11] Derigs, U., Do hmer, T., 2008. Indirect search for the vehicle routing problem with pickup and delivery and time windows. OR Spectrum 30 (1), 149165.

[12] Bell, M.G.H., Wong, K.I., and Nicholson, A.J., 2005. A rolling horizon approach to the optimal dispatching of taxis. In: H.S. Mahmassani, ed. Transportation and traffic theory: flow,dynamics and human interaction. Oxford: Elsevier Science Ltd., 629648.

[13] Berbeglia, G., Cordeau, J.F., and Laporte, G., 2010. Dynamic pickup and delivery problems.European Journal of Operational Research, 202 (1), 815.