STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **10**, February 2022, pp 263–281.
Published online in International Academic Press (www.IAPress.org)

Hybrid heuristics for the dial-a-ride problem with transfer and synchronization in central hub

Saïd Hammouda [*], Ali Lemouari [*], Mokhtar Taffar [*]

Department of Computer Science, LaRIA Laboratory, Jijel University, Jijel, Algeria

Abstract This paper deals with the dial-a-ride problem with transfer, where a set of customer demands are divided into two different regions, such as the pickup and delivery nodes must not be in the same region for a given request. Two vehicles are needed to meet customer demand which implies a synchronization process between vehicles in the transfer point called a central hub. In this paper, we provide a solution method based on an adaptive large neighborhood search algorithm (ALNS), to stretch the neighborhood of a solution, which gives the best search space exploration. Efficient constructions methods are proposed to repair a destroyed solution, giving a large possibility case. An intensification mechanism is ensured by blending the proposed ALNS with TS and SA algorithms. Implemented algorithms outperform all comparison methods such as memetic algorithms regardless of the quality solution and the run-time for all existing benchmarks sets.

Keywords Dial-A-Ride Problem, Transfer point, DARP, DARPT, ALNS, Simulated Annealing, Tabu search

AMS 2010 subject classifications 74Pxx, 68Rxx, 05C12

DOI: 10.19139/soic-2310-5070-1165

1. Introduction

This article proposes the resolution of transport on demand problem using *simulated annealing* (SA) and *Adaptive Large Neighborhood Search* (ALNS) as research metaheuristics. *The Dial-A-Ride problem* (DARP) is a variant of the *pick-up and delivery problem*(PDP), which allows delivering a set of customers from their collection sites to destination sites after planning a fleet of the vehicle of delivery service respecting some constraints such as vehicle capacity, the convenience of each customer, quality of service, etc.

Vehicle routing problem (VRP) is a PDP in which either all the origins or all the destinations are located at the depot, aims to find a set of routes at minimal cost by finding the shortest paths starting and ending at the depot, using a minimum number of vehicles, so that the known demand of all nodes are met. Each node is visited only once, by a single vehicle, and each vehicle has a limited capacity. The currently VRP models used are different from the one introduced by [1] and [2], as they increasingly aim to solve real-life problems, by adding constraints of complexities such as the maximum travel times, time windows for pickup and delivery, and requests information that dynamically changes over time. In VRP problems each vehicle only travels one route, each vehicle has the same characteristics using a single central depot. The purpose of the VRP is to arrange a set of vehicle routes in order that each customer is visited exactly once by a vehicle, each vehicle begins and ends its route at the depot, and therefore the vehicle's capacity isn't exceeded with minimum costs delivery.

In classical DARP, each customer is delivered by a single vehicle from their delivery site to their destination, and customers associated with separated requests can share the same vehicle as long as the capacity allows it to

[*]Correspondence to: Saïd Hammouda (Email: hammouda_said@univ-jijel.dz). Ali Lemouari (Email: lemouari_ali@yahoo.fr). Mokhtar Taffar (Email: Mokhtar.taffar@gmail.com).

ISSN 2310-5070 (online) ISSN 2311-004X (print)

minimize the cost of transport, but this is not enough as the vehicles travel great distances, and we can find two cars heading to two destinations in the same area to take different customers so that one vehicle can deliver the customers. To solve this problem, many researchers concluded that adding intermediate points called transfer points where customers are often transferred from one vehicle to a different, can reduce transportation costs; this problem is named the DARPT.

The principle of DARPT is to reduce costs by consolidating flows at transfer points. For large remote requests instead of transporting the request with a single vehicle from the collection site directly to the delivery site, we split the route into two parts, which reduces the cost of the visit and the cost of transporting. The planning of incoming and outgoing services at the transfer center level poses several problems that increase the complexity of the sub-adjacent routing of vehicles. An incoming passenger must arrive at the transfer center before the associated outbound service leaves the center as showing in Figure 1. Combined neighborhood search methods and k-opt heuristics with the simulated annealing algorithm are presented in [3], obtained results exceeded all proposed results in the literature for every instance problem of the dial-a-ride problems with transfer central hub. This work deals with the extension of the classical dial-a-ride problem through the addition of transfer points hub, this extension accords with the scenario on the distribution of a fleet of vehicles into two regions, each vehicle operates only in one specific region [3]. Therefore, customer trips are split into allowed regions with the need for vehicle exchanges in the hub center, which requires respect to the transfer planning constraints.

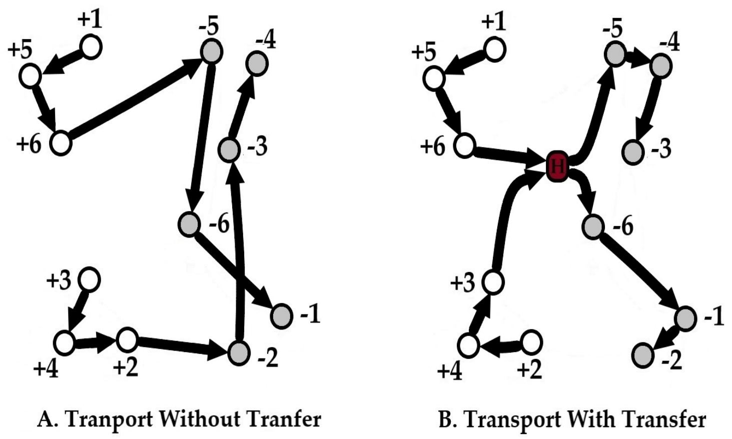

Figure 1. Transport of 6 orders without transfer and with transfer.

The remainder of this paper is organized as follows: First, we present the motivation of our works in Section 2. Then, in Section 3 we give some examples of applications of DARPT in real life. In Section 4, a literature review is presented. In Section 5, a problem statement and model of the DARPT are given. The solution approaches containing a description of the proposed ALNS and SA with Neighborhood search methods are detailed in Section 6. In the next section, we discussed and analyzed the results of complete experiments with DARPT. Finally, the last section concludes this paper and gives some perspectives of our work.

2. Motivation

In order to reduce the overall solution, and particularly in the cases where the distance between the pick-up and drop-off sites is too long, we are tended to use one or more centers to transfer customers between vehicles. The choice of this strategy is motivated firstly by the fact that real situations require the pointing of the trip of a customer between several vehicles of inter-region transport or inter-urban requests, and secondly for technical reasons when the founded solutions for the methods of resolution for the DARP problem does not always guarantee very good quality of service for priority requests and inter-urban requests where the distances are long as possible.

3. Objective and applications

In the real world of transport, the very long distances present complex and very expensive situations to deal in particular when no intermediate center is envisaged. This observation requires the pointing of the trip of a customer between several vehicles. Besides, technically the solution must always guarantee a very good quality of service for priority requests and inter-urban requests.

The purpose of our approach to using more centers for the very long distances separating the pick-up site and drop-off site in order to transfer customers between vehicles is to prove the feasibility of this approach and its efficiency to obtain a better solution for the long distances.

In reality, the transfer center between two regions is needed when the vehicles fleet size is limited and/or the distance of travels is too long, e.g., long trips between countries that take place by plane, most of them are divided into two trips, the first flight is covered by the airline company of the country from which the flight departs, to a transit country, where the passengers take a rest before the start of the second flight, which is covered by another company from the destination country as showing in Figure 2.

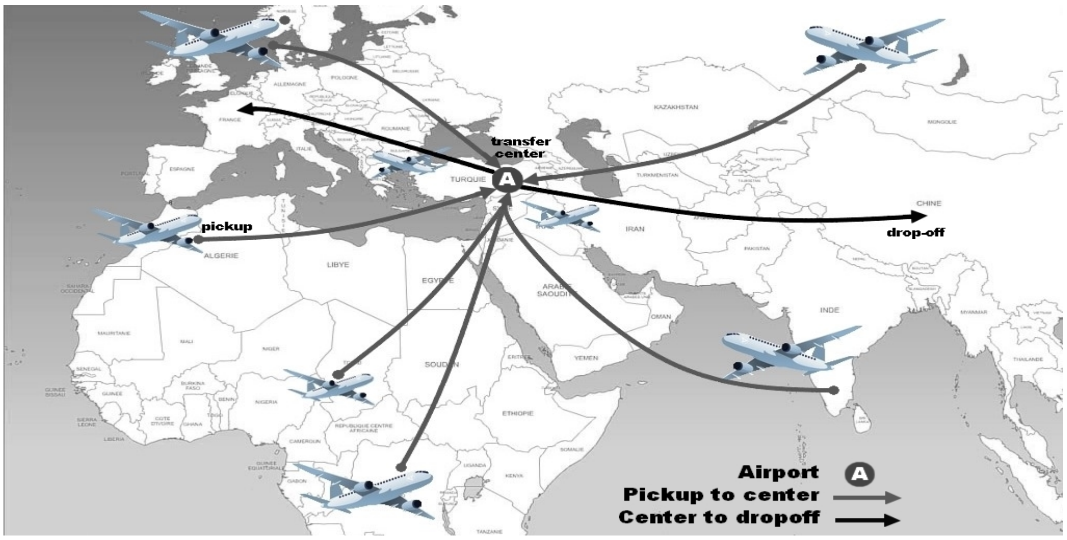

Figure 2. Passenger air transport.

In the case of transporting patients between hospitals, we can use this method to find the best transport routes using one or two hospitals as a transfer center to make consultation of the patients before starting their route. This transfer can ensure the health of patients especially for patients who cannot resist long distances and need good care.

4. Literature review

The DARP belongs to the class of VRP [4] and is a special case of the capacitated pickup and delivery problem surveyed by [5]. In DARP the fact that users may be grouped in a vehicle reduces the operating costs [6]. In addition, these vehicles may be stationed at different vehicle depots in their service area [7]. The main contribution of the paper is to allow transfer in the DARP. The authors of [8] have been published on this subject, They express the problem by DARPT where the location of their transshipments points are known before the resolution.

In the literature, many techniques and algorithms have been proposed to solve the DARPT, including heuristics, metaheuristics, exact methods, etc. First, the possibility of transferring customers between the vehicles in some transfer points is proposed in [9] that divides the system works in multiple sub-regions containing a bus that only travels there. A heuristic for the routing and programming of multi-objective vehicles for a *Pickup and Delivery*

Problem with Transfer (PDPT) proposed in [10], which the size of the fleet is not predetermined, customers are authorized to make transfers between vehicles and the transfer operation can be considered as a recourse policy because transfers are only used if the insertion of a request in the solution requires the use of an additional vehicle.

The heuristic proposed in [10] is adopted in a dynamic version of the problem by [11], they introduced the design and implementation of heuristic to solve the problems of *split delivery pickup and delivery time window with transfer* (SDPDTWT) of real-time customer data sets. In [12] a solution based on the decomposition of the Benders (branch and cut) proposed, where the transfer points are included where vehicles can interact with interchangeable passengers, with additional variables to follow these customers along their route, these results are compared with the results given by branch and bound. A new solution presented in [13], based on a graph formulation of the problem that treats each request independently to find the shortest path from pickup and delivery locations in the PDPT problem.

The PDPT is solved in [14] using a *greedy randomized adaptive search procedure* (GRASP) metaheuristic approach which uses ALNS in the improvement phase. The authors of [15] applied the ALNS metaheuristic explained how to check the feasibility of a query insertion where the location of their transfer points is known before resolution. An algorithm based on insertion techniques and constraints propagation proposed to solve the dial-a-ride problems [16], including the possibility of one transshipment from a dynamic transfer point by request, experiment on Cordeau instances [17].

The method proposed in [8] is evaluated on real and generated instances and their experiment shows that savings due to transfers can be up to 8% on real-life instances. A new deterministic annealing meta-heuristic to solve larger problem instances [18]. In [19] a *variable neighborhood search algorithm* (VNS) is developed to solve *the dial a ride problem with split requests and profits*(DARPSRP). Regarding the mathematical programming, a new heuristic is developed in [20] to obtain good quality solutions in a reasonable amount of time, consider the logistics business context propose using a fix, and optimize the metaheuristic algorithm for the full truckload version of the PDPT.

A problem-solving method using one hub center (transfer point) is proposed, which separates the region into two regions A and B such as for each customer his collection site and delivery site are located in a different area. *Transfer scheduling constraints* (TSC) is imposed and enhanced by a schedule building procedure that postpones waiting times at selected locations if necessary so that the transfer comfort remains at an acceptable level by avoiding too short or too long vehicle change times, but also to limit the total travel time between the collection site and the final delivery site [21].

Compared the results given by *large neighborhood search* (LNS) and *genetic algorithm* (GA) to solve the PDPT with benchmark instances from the literature. The GA outperforms the LNS because the LNS finds the optimal solution in 65 % of cases, and the GA finds the optimal solution in 74 % of cases [22]. In [23], The authors used a branch and cut algorithm for solving the *pickup and delivery problem with split loads and transshipments* (PDPSLT). The aim is to minimize the sum of travel costs and transshipment costs, so they present an arc-based mixed integer formulation for the problem and testing their implementation on randomly generated instances from [24] adapted to remark special characteristics of the problem or methodology.

5. Problem statement and model

Let N nodes (customers) distributed on both sides A and B, divided into origins and destinations sites, can change vehicles during their travels. The vehicle change is performed at a specific location called the hub center. So, the hub center can play a dual role, pick up or delivery center, as shown in Figure 1. The customer delivery of $i = 1$ is split into an inbound request, from the pickup site +1 to the temporary delivery node (hub center), and an outbound request from the temporary pickup node (hub center) to the delivery node -1. In this study, some constraints must ensure to guarantee a better quality of transport :

- For each request the pickup site must be visited before the corresponding drop off (constraint 1).
- The capacity of the vehicle must be respected (constraint 2).
- Each vehicle can deliver except customers located in their region (constraint 3).
- Each vehicle must return to the center after a maximum traveling time (constraint 4).

The planning of incoming and outgoing services at the transfer center level poses several problems that increase the complexity of the sub-adjacent routing of vehicles. Two constraints must be taken into account when planning the transfer :

- Minimum time of vehicle change T_{min} times should be considered (constraint 5).
- The total ride time of customer i measured between his pickup and arrival times r_{max} must not be exceeded (constraint 6).

Several models have been proposed in the literature for different variants of the standard dial a ride problem, we keep the same notation used on a three-index formulation model proposed in ([7]; [25]) with appropriate modification.

We define the DARPT on a directed graph $G(N, E)$, N denote a set of nodes where $N = N^+ \cup N^- \cup T$, where N^+: a set of collection nodes for requests, N^-: a set of delivery nodes for requests and *T*: a set of transfer points. Let E: a set of edges that considers all possible links except direct links from the collection nodes of requests to its delivery nodes are not allowed. Let *i* a request represented as a couple (i^+, i^-), such as each request is split into two sub-requests, an inbound requests from the collection node to the central hub noted (i^+, h^-) and an outbound requests from the central hub to the delivery nodes noted (h^+, i^-). Let *R* a set requests where $|R| = n$ and *K* a set of vehicles, each vehicle have a capacity Q_v, $K = K_a \cup K_b$ where K_a are a set of attributed vehicles to region A, K_b are a set of attributed vehicles to region B. For each node $i \in N^+ \cup N^- \cup T$ are associated a load q_i where $0 \leq q_i \leq 1$ for $i \in N^+ \cup N^-$ and $0 \leq q_h \leq Q_h$ where $h \in T = h^+, h^-$ and $q_{h^+} = q_{h^-} = q_h$, $Q_h$the maximum capacity of the central hub. Let us C_{ij}^k the edge costs between nodes *i* and *j* when using vehicle *k*. Likewise, we define t_{ij} as the travelling time between nodes *i* and *j*. The ride time for any requests is limited with an upper bound denoted as the maximal ride time T_{min}. In the central transfer hub, we note T_{min} as the least transfer service time ensuring relaxed changes from an inbound vehicle to the outbound one.

So the DARPT with synchronization process in the central hub consists to minimize the following objective function :

$$Min \sum_{k \in K} \sum_{i \in N} \sum_{j \in N} c_{ij}^k x_{ij}^k \tag{1}$$

Subject to :

$$\sum_{j \in N} X_{ij}^{k_a} - \sum_{j \in N} X_{i^- j}^{k_b} = 0, \qquad \forall k_a \in K_A \; \exists k_b \in K_B, \; \forall i \in N^+ \tag{2}$$

$$\sum_{k \in K} \sum_{j \in N} X_{ij}^k = 1, \qquad \forall k \in K, \; i \in N^+ \tag{3}$$

$$\sum_{j \in N} X_{ji}^k = \sum_{j \in N} X_{ij}^k \qquad \forall i \in N^+ \cup N^-, \; k \in K_A \cup K_B \tag{4}$$

$$\sum_{i \in N^+ \cup N^-} X_{hi}^k = \sum_{i \in N^+ \cup N^-} X_{ih}^k \qquad \forall k \in K_A \cup K_B, \; h \in T \tag{5}$$

The objective function of the model is given in Eq. (1). It is the reduction of the transport costs by the minimization of traveling distance. The goal of the model is to find the best sequence of vehicle trips and guarantee a very good quality of service for the customers. Eq. (2) and Eq. (3) ensure that each request is served by two different vehicles but not at the same time. Eq. (4) impose that, for each location in *N*, there is an equal number of vehicles arriving and leaving. Eq. (5) assure that each vehicle starts and ends its route at the central hub.

Let us A_i^k, B_i^k and L_i^k are variable denoting respectively the arrival time at *i* using vehicle *k*, the beginning service at node *i* using vehicle k and leaving time at node *i*, with $L_i^k = B_i^k + d_i$. We assume that completion service time and leaving time at any node denotes the same parameter, and the service starts just when the vehicle arrives $A_i^k = B_i^k$. To each vertex is associated a load q_i, with $q_{h^-} = q_{h^+} \geqslant 0, \; q_i \geqslant 0, \; i \in N^+$, and a service duration d_i, with $d_i \geqslant 0,\ i \in N$. Let us T_k the maximal route duration of $k \in K,\ w_i^k$ the load of vehicle k when leaving vertex i and r_i is the ride time associated to the request i, and as long as each order is made up of two requests requiring exactly two vehicles, then the total ride time of request is splitted into the two corresponding ride times of inbound and outbound requests $r_{i+h-}^k,\ r_{h+i-}^k$.

$$L_{h-}^{k_a} + T_{min} \leqslant B_{h+}^{k_b} \tag{6}$$

$$L_{i-}^{k_b} - B_{i+}^{k_a} \leqslant r_{max} \tag{7}$$

$$w_j^k = (w_i^k + q_j)X_{ij}^k \qquad i, j \in N,\ k \in K_A \cup K_B \tag{8}$$

$$A_{h-}^k - L_{h+}^k \leq T_k \qquad k \in K_A \cup K_B \tag{9}$$

$$r_{i+h-}^k \geq B_{h-}^k - (B_{i+}^k + d_{i+}) \qquad k \in K_A/K_B \tag{10}$$

$$r_{h+i-}^k \geq B_{i-}^k - (B_{h+}^k + d_{h+}) \qquad k \in K_A/K_B \tag{11}$$

$$r_{max} \geq r_i \geq r_{i+h-}^k + r_{h+i}^{k'} + T_{min}, \qquad (k \in K_A,\ k' \in K_B)/(k \in K_B,\ k' \in K_A) \tag{12}$$

$$L_{h-}^{k_a} + T_{min} \leq B_{h+}^{k_b} \qquad k_a \in K_A,\ k_b \in K_B \tag{13}$$

$$B_{h+}^{k_b} + d_{h+} \leq L_{h+}^{k_b} \qquad k_b \in K_B \tag{14}$$

$$B_{h-}^{k_a} + d_{h-} \leq L_{h-}^{k_a} \qquad k_a \in K_A \tag{15}$$

The vehicle tour is a path that begins in on center hub, serves all or some received requests, and terminates at the same hub. Eq.(6) preserves the minimal vehicle change time T_{min} or the ride time of passengers measured between pickup and delivery. No more than r_{max} time units are scheduled between the initial pickup time and the final leaving time of the passenger requests Eq.(7). The constraints (10), Eq.(11)ensure the organization of any inbound and outbound request, and Eq.(12), Eq.(13),Eq.(14) and Eq.(15) assure the relaxation in hub center.

6. Solution methodology

In this section, an overview of the different heuristics of resolution for the DARPT problem is presented. There is a similarity between ALNS and SA although the ALNS is deeper than SA. A solution is represented by a list of size m, m represents the number of roads in the processed instance. Each road is represented by a sub list that begins and ends with a zero which is the hub center and contains a series of requests visited by the latter's vehicle. For example, for the case of the figure 3, the solution contains 3 trips start and end with zero which represents the transfer center. The first contains three pickup nodes from the region A, (0, +5, +6, +4, 0). The second trip contains three pickup nodes and three drop off nodes from the region B, (0, +3, +1, -5, -4, -6, +2, 0). The last trips contains the last three drop off nodes in the region A, (0, -3, -1, -2, 0).

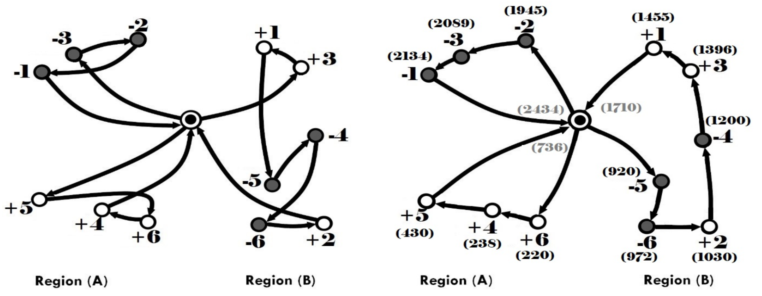

Figure 3. An initial solution and optimal solution of the small instance with $r_{max} = \infty$ time units and $T_{min} = 0$ time units.

6.1. Initialization and Insertion

In this study, the solution initial begins by initializing the pickup type nodes was chosen randomly in a route and at the same time their drop off type nodes in another route in the second region as long as the capacity of these vehicles has not been exceeded. In this case, the list of the first region placed in the big list of roads and the second list from the other region will be moved to the first list after emptying it. The process is repeated until the end of the list of nodes. This procedure guarantees the scheduling of roads by respecting the constraints Eq.(1), Eq.(2), Eq. (3) and Eq.(4). So, we can do this by following the steps of the Algorithm 1.

Algorithm 1 (Initialization)
Input problem data: List of Node *LN*; $i \leftarrow 0$; R_0, r1, r2 : list of routes

```
While non-empty (LN) Do
    Switch i Do
        case 0 :
            Repeat
                choose $P_i^+$ randomly node from LN belong to region A
                If (constraints 5, 6) respected Then
                    add $P_i^+$ to $r1$; add $P_i^-$ to $r2$
                    remove $P_i^+$, $P_i^-$ from LN
                End If
            Until not (constraint 2)
            add r1 to $R_0$
            $i \leftarrow 1$
        case 1 :
            Repeat
                choose $P_i^+$ randomly node from LN belong to region B
                If (constraints 5, 6) respected Then
                    add $P_i^+$ to $r2$; add $P_i^-$ to $r1$
                    remove $P_i^+$, $P_i^-$ from LN
                End If
            Until not (constraint 2)
            add r2 to $R_0$
            $i \leftarrow 0$
End Do
If non-empty(r1) Then
    add r1 to $R_0$
End If
If non-empty(r2) Then
    add r2 to $R_0$
End If
```

6.2. Simulated Annealing Heuristic

Simulated annealing is a metaheuristic designed to solve difficult optimization problems. The idea is to perform a movement according to a probability distribution that depends on the quality of the different neighbors as the best neighbors have a higher probability and the less good ones have a lower probability using a temperature parameter called T, which decreases during the search until its value reaches a stop sill S. The choice of the initial temperature T of the system is an essential point in the simulated annealing algorithm. The initial value assigned to T is reduced gradually by multiplying it on λ. The lambda λcoefficient is to be chosen carefully, indeed if it is too small, the temperature will drop very quickly and the algorithm could then block in local minima, otherwise, the temperature will drop very slowly and the calculation time will be quite substantial (λ is situated on average between 0.999

and 0.99999. As shown in Algorithm 2, we start by building an initial solution s_0 from a set of offers and then we initialize T, s and λ according to the problem to be dealt with. After the generalization of a solution close to $s\prime$ of s_0, this solution is accepted if it optimizes the objective function or if the probability of Exponential $(-\Delta f/T)$ is greater than a random number $u \in [0,1]$ such as Δf represents the difference between $s\prime$ and s_0. Then we repeat the same steps until we get the stop sill.

```
Algorithm 2 (SA Heuristic)
Input : $s_0$ solution, initial $T, \lambda$
$s \leftarrow s_0$
   While $T > S$ Do
      $s^{'} \leftarrow Neighborhoodsearch(s)$
      If $f(s^{'}) < f(s)$ and all constraints respected Then
         $s \leftarrow s\prime$
      Else
         $p \leftarrow Probability(T, \Delta f)$
         $u \leftarrow Random[0,1]$
         If $u < p$ and all constraints respected Then
            $s \leftarrow s\prime$
         End If
      End If
   End Do
```

6.3. *SA Neighborhood search*

A neighborhood of a solution s is a set of solutions that can be created by moving customers from a position to another one in its route or to another route. To avoid generating the complete space of solutions is better to mix many methods at the same time to get a better movement and expansion of solutions. In this study we used four methods with SA heuristic: Random inter-roads permutation, intra-road multiple permutations, Shuffle permutation, and two optimal as shown in the Figure 4.

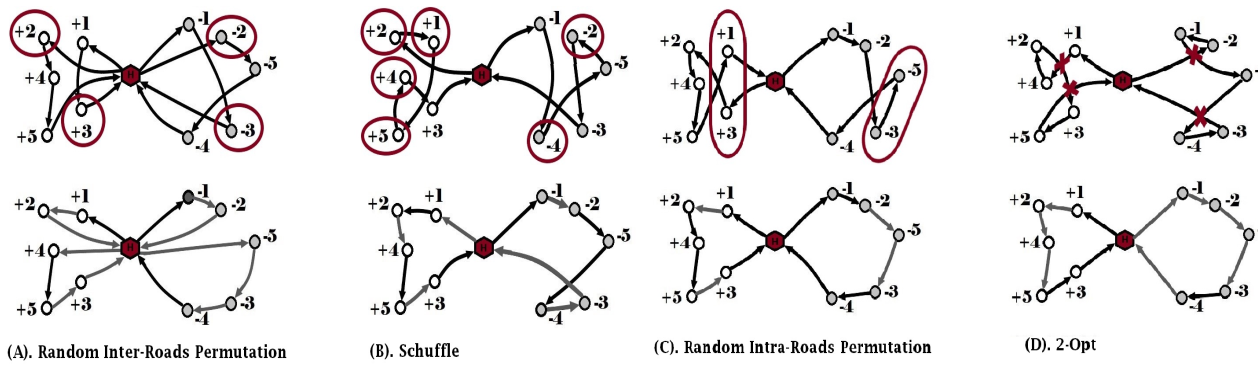
(A). Random Inter-Roads Permutation
(B). Schuffle
(C). Random Intra-Roads Permutation
(D). 2-Opt
+1
+2
+3
+4
+5
-1
-2
-3
-4
-5

Figure 4. Neighborhood search methods.

The case of inter-roads permutation consists in swap the positions of two customers randomly choosing and have the same type of request (collection or delivery) from two routes also choosing randomly from the same region respecting all constraints. An example is shown in figure 4(A). Secondly, the predefined shuffle function gave better results and quicker change of current results by mixed randomly the costumers between them as shown in figure 4(B). Another case is the random intra-road permutation that consists of choosing at the beginning two customers of the same route chosen randomly, after that we swap their positions as shown in figure 4(C), respecting all constraints. Finally, we used the local search and 2-opt heuristic to replace the roads that have an intersection between them with two other roads where the distance is lower and all constraints are respected as shown in figure 4(D).

6.4. *Large neighborhood search heuristic*

In this section, we describe the ALNS as an extension of *Large Neighborhood Search* (LNS) for the resolution of DARPT depending on the destroy and repair method to have a better movement between the solutions until obtaining the best global solution for each problem. The ALNS has been described extensively by [26] and applied to the PDPTW by [27]. We present the large neighborhood search algorithm proposed by [28]. The ALNS heuristic exploits a complex neighborhood that makes it possible to find better candidate solutions. In the LNS heuristic, the neighborhood is defined by the two operators of $destroy(s)$ and $repair(s)$, where s is the current solution.

Algorithm 3 (LNS Heuristic)
Input : s initial solution, $best \leftarrow s$

```
Repeat
    Select $D_i,\ R_i$
    $s^{'} \leftarrow Repair(D_i,\ Detroy(R_i,\ s))$
        If $accept(s,\ s^{'})$ Then
            $s \leftarrow s^{'}$
        End If
        If $f(s^{'}) < f(best)$ Then
            $best \leftarrow s^{'}$
        End If
Until stop criterion met
return best
```

As shown in Algorithm 3, the LNS algorithm starts by choosing two destruction and repair methods from the list of available methods and building a solution $s^{'}$ using these two methods. First, a simple accept criterion would be to accept all improving solutions $s^{'}$, Such a criterion has been used in earlier LNS implementations. Second, the current solution must accept if its objective value exceeds that of the best solution currently found.

6.5. *Adaptive Large neighborhood search Heuristic for DARPT with synchronization*

Adaptive large neighborhood search heuristic originally proposed by [27] using multiple destroy and repair operators within the same search process. At each iteration, the heuristic destroys a part of the current solution and repair it in a different way to perform a new solution. The same approach was used in [15] to solve the PDPT. The underlying principle of the ALNS is to destroy and repair a solution iteratively in order to improve it. The ALNS with two other metaheuristics proposed in [29], for solving the multi-depot and multi-trip heterogeneous dial a ride problem. The structure of the ALNS algorithm proposed by [30] and [31] presented in Algorithm 4.

Algorithm 4 (ALNS Heuristic)
Input : $S \leftarrow$ Create initial population ,
$best \leftarrow S$
P_i^D, P_i^R are vectors weight corresponding to the historical operators fulfillment rate

```
Repeat
    Select $D_i,\ R_i$ using Roulette selection and $P_i^D, P_i^R$
    $s^{'} \leftarrow Repair(D_i,\ Detroy(R_i,\ s))$
        If number of iteration has passed Then
            $s^{'} \leftarrow TSHeuristic(s^{'})$
        End If
        If $accept(s,\ s^{'})$ Then
            $s \leftarrow s^{'}$
        End If
        If $f(s^{'}) < f(best)$ Then
            $best \leftarrow s^{'}$
        End If
    Update $P_i^D, P_i^R$
Until stop criterion met
return best
```

We start building a solution $s^{'}$ by choosing two destruction and repair methods from the list of available methods using the roulette selection method. This choice is according to the weighted performance value of each destroy and repair method. Three modifications are added to the LNS to have an ALNS algorithm. First, after a certain number of iterations, we give the current solution to a TS heuristic to improve it quickly using the neighborhood methods implemented with simulated annealing. In this case, the current solution will be replaced by the one given by the TS. Second, the update of the score value of each destroy and repair method it's an important step for defines which methods will be used in the next steps. Last, simulated annealing accepts the criterion used to accept solutions in special cases.

6.6. Destruction methods

The solution of the latest ALNS method is destroyed by five destruction heuristics and repaired by two insertion heuristics according to a probability, for this reason, the search is called adaptive. The probability of choosing each destroys and repair method is due to the importance of this latter in improving the result in the past iterations. We use five destruction heuristics :

1. Shaw Removal: this heuristic has been proposed by [32] for the PDP with time windows. The heuristic idea is to remove similar requests to have a new solution. The same idea has been used in [27]. The problem here is how to determine the similarity degree between requests, so shaw defined a related measure between requests i and j notes R(i,j) given by :

$$R(i,j) = \alpha(d_{+i,+j} + d_{-i-j}) + \beta(|T_{+i} - T_{+j}| + |T_{-i} - T_{-j}|) \tag{16}$$

   where $+i, -i, +j, -j$ present the pickup and delivery sites of request i, T_i indicates the time when site i is visited.
2. Random Removal: this removal method remove all q requests from the current solution at random. This heuristic can run faster than the other removal methods.
3. Worst Removal : this method remove all requests i with high cost and insert them in another position in the current solution s according to the $cost(i,s)$ [26] where $cost(i,s) = f(s) - f_{i^-}(s)$, and $f_{i^-}(s)$ is the solution cost of s without request i. From the example shown in Figure 5, the heuristic consists to remove a q request with a high cost. To do it, all planned requests sorted by descending order and eliminate the sub-requests $(i^+,h), (i^-,h), (j^+,h), (j^-,h)$

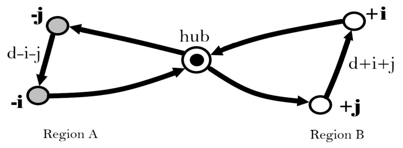

Figure 5. Delivery of two requests.

4. Inter Removal: this method allows to deletion of two nodes from the same trip at random on the condition of saving their positions to swap them in the repair step.
5. Shuffle Removal: this method removes all the nodes of a trip then reinserts this node randomly in the repair step.

6.7. Repair Methods

The repair phase allows the construction of a solution destroyed during the destroy phase. Let s be a solution, such that s composed from a set of route $\bigcup_{i=1}^{m} R_i$, where R_i is a route and m is the number of route in solution. Let n_A, n_B represents the number of route in each region, and (+i,-i) is a given request to be inserted in s.

So, the number of insertion positions of the pickup node +i and delivery node -i equal respectively to:

$$\sum_{i=1}^{x_A} |R_i| + x_A and \sum_{j=1}^{x_B} |R_j| + x_B \tag{17}$$

Let given R_i a route composed from n nodes. Any attempt to insert a request (+i,-i) in the best position gives us a complexity of $\theta(mn^2)$. We use two insertion heuristics based on best insertion.

1. Random insertion: This method allows us to insert the requests in trip randomly but the origin region of the request must be the same region of the insertion position.
2. Greedy insertion: The greedy heuristic is a simple construction heuristic. It inserts one request in each iteration at the same trips to find the best place for this request. At each iteration, we calculate the objective value, and the place which gives us the minimum value will be the best place to insert the request. So, The request is inserted at its minimum cost position.

6.8. Tabu Search

The authors in [17] have previously applied *Tabu Search*(TS) heuristic to solving the DARP. This is one of the main contributions to the origin of metaheuristics, which makes it possible to solve complex optimization problems instead of which classical methods are ineffective [33].

In the ALNS algorithm, we combined the TS heuristic as an important step to improve the current solution. This heuristic can work after a certain iteration has passed. The TS algorithm cannot guarantee the optimally of the solution found but shown great efficiency in s_0 randomly generate. For each iteration, we put the current solution in a taboo list Tl. So, each solution generated that does not belong to Tl and improves the value of the objective function to replace the current solution. We repeat the same steps until we get the stop sill or the generation of 50 solutions successively which does not improve the objective function.

In tabu search, a taboo list is needed to avoid cycles, something that happens when trying to instantiate the last remaining objects. In our implementation, we used a taboo list containing all the movements that are not allowed for a certain duration. This duration is defined dynamically according to the size of the taboo list initially equal to the number of possible movements, once the taboo list is complete, We delete the most recent solution.

6.9. Adaptive mecanism

Compared to the LNS method the metaheuristic ALNS uses a set of destroying and repair, the choice of the methods which will be used in the next generation is determined according to the weighted performance of each destroy and repair method. In algorithm 4 Select (D_i, R_j) is a function able to return the appropriate indices of selected operators, $detroy(R_j, s)$ and $repair(D_i, s^{'})$, where $s^{'}$ an incomplete solution, are picked operators according to their history. Two variables are used in the algorithm to save the historical performance of each proposed operator. Initially, all operators have the same weight, and a roulette selection method is used for the selection process, either for destroy or the repair operators, the method is known as law variance sampling. As shown in Figure 6, the adaptative mechanism allows selecting the appropriate method to be used in destroying heuristics. Only inter removal and shuffle removal dominate the selection process in the end. This is due that only the two heuristics

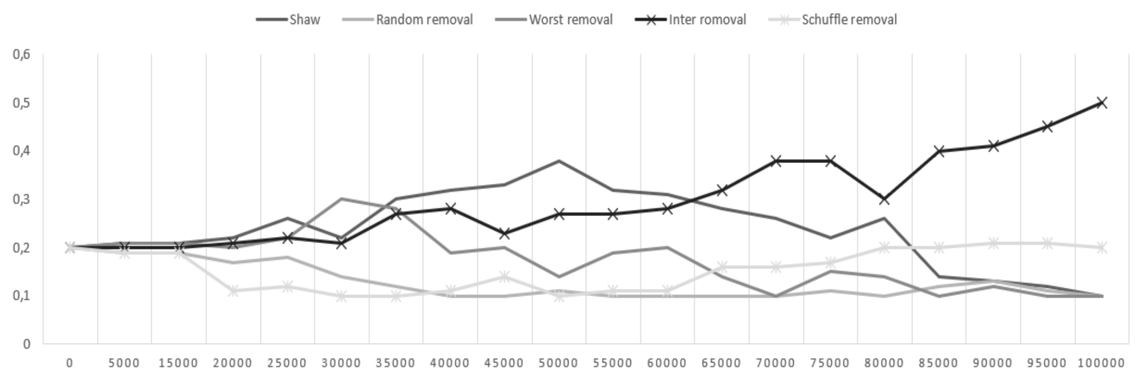

Figure 6. Performance of the five removal operators during the execution of the ALNS algorithm.

allow the modification in intra-routes. The method used in this paper gives an additional value to the scores of destroy and repair methods that improved the current solution. The scores for both methods are updated by the same amount, as we cannot tell whether it was the removal of the insertion that was the reason for the success. Indeed we build our weight array content a destroy and repair methods w_i, where $i \in n1 \cup n2$, n1 is the number of implemented methods of destroying and n2 is the number of implemented methods of repair. Following [27] for each iteration of the ALNS algorithm, the scores are updated according to the following equations:

$$P_i^D = P_i^D + \alpha \; and \; P_j^D = P_j^D - (\frac{\alpha}{\sum w_j}), \qquad j \neq i, i, j = 0, .., n1 \tag{18}$$

$$P_i^R = P_i^R + \alpha \; and \; P_j^R = P_j^R - (\frac{\alpha}{\sum w_i}), \qquad i \neq j, i, j = 0, .., n2 \tag{19}$$

7. Computational results

The application was coded in java with the IntelliJ programming environment IDEA 2018.3r plus JavaFX library and run on a PC with an Intel(R) Core (TM) i5-6300HQ Cpu@2.30ghz processor, 8GB RAM, and AMD FirePro W5130M graphics card.

7.1. Parameter setting and stopping criterion

To have better results and efficient algorithms it is necessary to choose the parameters' values correctly where that changes as the type of problem changes. In this study, the parameters are set as follows: T = 2000, S = 0.01, λ is situated on average between 0.999 and 0.99999, T_{min} is given by the user between [0, 100] with small instances and [0, 250] with medium and big instances, r_{max} is given by [2000, ∞] with small instances and [5000, ∞] with medium and big instances, The maximum time for a tour D_i = 10000 units distance, each instance are solved five times during 1000 iterations with Small instances, 10000 iterations with Medium instances and 100000 iterations with Big instances. we take the average of these five solutions.

7.2. benchmark instances

To illustrate the result, robustness, and discrimination of our SA and ALNS algorithms with the proposed neighborhood methods we use benchmark instances to randomly generate of the DARPT problem proposed by [21]. The proposed benchmark is structured as follow :

- Benchmark-caspt-small : composed of 6 problems, each problem contains 6 requests and 12 sites, 4 vehicles at most and one hub center.

- Benchmark-caspt-medium : composed of 15 problems contain 25 requests and 50 sites, between 6 and 10 vehicles and one hub center.
- Benchmark-caspt-big : composed of 15 problems containing 100 requests and 200 sites, 20 vehicles at most, and one hub center.

In this study, there are three percentage cases P {0.50, 0.75, 1.00 } which represent the distribution of AB and BA orders in the two sides, wherein the case P = 0.50 the number of the pick-up sites equals the number of drop-off sites in two regions, in the case P = 0.75 the number of the pick-up sites is 75 % in one region and 25% in the other region and finally in the case P = 1.00 all pick-up sites in one region and their drop-off sites in the other region.

7.3. *ALNS vs.SA*

Table 1. Results of ALNS vs. SA with Small instances.

SMALL	r_{max}	∞		2000	
	T_{min}	0	100	0	100
caspt2v4o6hetp050	SA	2375	2375	2375	2375
	ALNS	2375	2375	2375	2375
caspt2v4o6hetp075	SA	2585	2585	2585	2585
	ALNS	2585	2585	2585	2585
caspt2v4o6hetp100	SA	2025	2025	2025	2025
	ALNS	2025	2025	2025	2025
caspt3v4o6hetp050	SA	2528	2528	2528	2528
	ALNS	2528	2528	2528	2528
caspt3v4o6hetp075	SA	2510	2510	2510	2510
	ALNS	2510	2510	2510	2510
caspt3v4o6hetp100	SA	1943	1943	1943	1943
	ALNS	1943	1943	1943	1943

The results of small instances problems are reported in Table 1, Columns 1 represent respectively the name of the instance (example: caspt0v2o6hetp050) which contains the number of vehicles ($v2 = 2$ vehicles), the number of orders ($o6 = 6$ orders) and at the end the percentage P (p050 equal to P = 50%). There are three percentage cases P (0.50, 0.75, 1.00) which represent the distribution of AB and BA orders in the two sides as shown in Figure 3, they are three orders AB and three orders BA so $P = 0.50$. The next four columns indicate the results obtained by our implementation in the four possible cases of T_{min} and r_{max} change.

We note that in this case there is no difference between the results when changing T_{min} and r_{max}, the result is often optimal. Taking the instance proposed in [21] of a DARPT with six requests $r = (1, 2, 3, 4, 5, 6)$ as shown in Figure 3 such as the collection and delivery nodes of every request are distributed in two separate regions AB request or BA request, $A = (-1, -2, -3, +4, +5, +6)$ and $B = (+1, +2, +3, -4, -5, -6)$. Each request is splitted into two other sub requests, for example (+1, -1) be (+1, center hub) and (center hub, -1). Here, the minimal travel distance sum is 2435 distance units for all the presented solutions, this result is confirmed mathematically by CPLEX in [21] who found the same result.

In the case of medium instances, the majority of the results given by the ALNS are better than the results given by SA in the four cases of temporal change T_{min} and r_{max}. If we talked about each problem instance alone, we note that the value of the solution increases directly with an increase of r_{max} or reduce of T_{min} or change them together, so each addition of a constraint increases the total problem-solving time. In terms of improving the result, the algorithm ALNS improves 80 % of medium instances compared to the SA algorithm.

Table 2. Results of ALNS vs. SA with Medium instances.

MEDIUM	r_{max}	∞		2000	
	T_{min}	0	100	0	100
caspt0v6o25hetp050	SA	4414	5006	6542	7000
	ALNS	4303	4514	6426	6430
caspt0v6o25hetp075	SA	4847	4977	6549	6615
	ALNS	4505	4587	5878	6004
caspt0v6o25hetp100	AS	3547	3460	7131	7557
	ALNS	3385	3459	5997	7390
caspt1v6o25hetp050	SA	5466	5689	7164	7297
	ALNS	4992	5112	7452	7052
caspt1v6o25hetp075	SA	5336	5233	7080	7198
	ALNS	5091	5239	6291	7069
Caspt1v6o25hetp100	SA	4304	4385	8084	8444
	ALNS	3993	4067	7052	7004
caspt2v6o25hetp050	SA	5209	5249	6845	6409
	ALNS	4840	4883	6532	6801
caspt2v6o25hetp075	SA	5303	5373	6384	7064
	ALNS	5136	5145	6642	6902
caspt2v6o25hetp100	SA	4038	3915	7063	7508
	ALNS	3988	4011	6668	6815
caspt3v6o25hetp050	SA	5422	5633	7500	7595
	ALNS	5039	5236	6953	7562
caspt3v6o25hetp075	SA	4822	4709	8126	8192
	ALNS	4525	4734	6815	7606
caspt3v6o25hetp100	SA	4617	4581	8804	8806
	ALNS	3799	3988	8092	7940
caspt4v6o25hetp050	SA	5218	5076	7046	6848
	ALNS	4807	4955	6960	7012
caspt4v6o25hetp075	SA	4800	4767	7847	7745
	ALNS	4752	4727	6396	7052
caspt4v6o25hetp100	SA	3966	3968	8157	8385
	ALNS	3751	4021	8015	8254

The results show that the ALNS heuristic on all four terms performs better than the LNS heuristic. The reason for this is that the Shaw removal and greedy insertion heuristics used by the ALNS heuristic is more performant compared to the other neighborhood methods using with SA.

Table 3 shows the number of generated trips. We observe an increase in the number of trips as a consequence of limiting the r_{max} time and as a result of the prolongation of the minimal transfer time T_{min}. Mixed trips are essential to minimize the number of trips. So, the reduction of the maximal allowed r_{max} time leads to an increase in the number of generated mixed trips. But the prolongation of the T_{min} time has no significant impact on the number of trips, except in rare cases. The competition between algorithms relates to the large problems that take a long time to solve. Researchers develop their algorithms by optimizing loops methods of finding the neighbors of the current solution, which they avoided moving to all solutions. Table 4 compiles the average observed travel distance in the different scenarios with our implementation of ALNS and SA algorithms. The results show the performance of SA compared to the ALNS with big instances. We have already discussed in [3] that the performance of SA due to the possibility of taking a nonfeasible solution according to a probability to avoid falling into a local minimum.

Table 3. Number of trips generated with Medium instances .

MEDIUM	r_{max}	∞		2000	
	T_{min}	0	100	0	100
caspt0v6o25hetp050	SA	3	3	5	7
	ALNS	3	3	6	6
caspt0v6o25hetp075	SA	3	3	6	6
	ALNS	3	3	6	6
caspt0v6o25hetp100	AS	2	3	8	7
	ALNS	2	2	8	8
caspt1v6o25hetp050	SA	3	3	7	7
	ALNS	3	3	7	7
caspt1v6o25hetp075	SA	3	3	8	8
	ALNS	3	3	7	8
Caspt1v6o25hetp100	SA	2	2	8	8
	ALNS	2	2	7	8
caspt2v6o25hetp050	SA	3	3	7	8
	ALNS	3	3	5	5
caspt2v6o25hetp075	SA	3	4	8	8
	ALNS	3	4	7	8
caspt2v6o25hetp100	SA	2	4	7	7
	ALNS	2	3	7	7
caspt3v6o25hetp050	SA	3	3	6	7
	ALNS	3	4	6	7
caspt3v6o25hetp075	SA	4	4	6	8
	ALNS	3	4	7	7
caspt3v6o25hetp100	SA	2	3	6	7
	ALNS	2	3	6	6
caspt4v6o25hetp050	SA	3	3	6	6
	ALNS	3	3	6	6
caspt4v6o25hetp075	SA	3	4	7	7
	ALNS	3	3	8	8
caspt4v6o25hetp100	SA	2	3	7	8
	ALNS	2	2	7	7

The adaptive mechanism of LNS can find just 16.66 % best results from all instances. So, the SA heuristic is more robust than the ALNS heuristics with large instances.

7.4. ALNS vs. SA and MA Algorithms

To verify the efficiency of algorithms ALNS and SA with medium and big instances, our results are compared with the memetic algorithm [21]. The results are reported in Table 5. Columns 1 represent the percentage P. Columns 3-6 represent the average results of the algorithms ALNS, SA, and MA in the four cases of the time change. The results given by SA and MA heuristic are compared in [3]. Compared results between proposed algorithms SA and proposed ALNS with MA algorithm, shown in Table 5 and Figure 7. The ALNS algorithm can outperform all implemented results proposed in [3] [21] for the MA and SA algorithms, this improvement can be easily found when trips are split into two regions and not constrained in minimum time T_{min} in-vehicle change on the hub center, and no constrained in a maximum ride time of customers r_{max}. Except for the last two results for constrained ride time, the MA algorithm outperforms SA and ALNS algorithms, this is due to the fact that the distribution parameter

Table 4. Results of ALNS vs. SA with BIG-10 instances.

BIG-10	r_{max}	∞		5000	
	T_{min}	0	250	0	250
caspt0v10o100hetp050	SA	12093	12296	14622	14709
	ALNS	12982	13584	16215	15987
caspt0v10o100hetp075	SA	12109	11250	16551	16600
	ALNS	13020	13006	18545	16988
caspt0v10o100hetp100	AS	9543	9629	16396	16423
	ALNS	**9502**	11128	17687	**16401**
caspt1v10o100hetp050	SA	12240	12054	15262	15268
	ALNS	13698	12987	16555	16785
caspt1v10o100hetp075	SA	11655	12220	15622	16980
	ALNS	12168	**12014**	17878	17991
caspt1v10o100hetp100	SA	9849	9593	17709	17644
	ALNS	10101	10050	19222	18121
caspt2v10o100hetp050	SA	12387	12818	15344	15461
	ALNS	14365	13601	17454	17001
caspt2v10o100hetp075	SA	12176	12199	16139	17429
	ALNS	**12165**	14009	16465	18211
caspt2v10o100hetp100	SA	10484	10098	17577	17604
	ALNS	10989	10189	17961	18529
caspt3v10o100hetp050	SA	13806	14032	16529	15531
	ALNS	14132	**14003**	16803	16919
caspt3v10o100hetp075	SA	11984	11919	16388	16670
	ALNS	**11901**	13163	**16291**	16902
caspt3v10o100hetp100	SA	9328	9583	16670	17079
	ALNS	10205	9978	18877	18015
caspt4v10o100hetp050	SA	12305	12272	15251	15618
	ALNS	12123	**12112**	15974	**15502**
caspt4v10o100hetp075	SA	11556	11653	16478	17165
	ALNS	13300	12004	16542	18255
caspt4v10o100hetp100	SA	9574	9842	17198	16726
	ALNS	10052	**9752**	17542	17440

between regions P is equal to 1.00, which means that each region contains either pickup or delivery sites, in this case, vehicles can visit only pickup or delivery sites at a time. This means that a route contains only pickup or delivery sites at a time. Adding to this the insertion operators used during the repair phase of the solution, here we only used two operators: Random and Greedy insertion heuristics which leads to less efficient results with reported results of MA algorithms. Figure 7 shows us that the curves represent the results of ALNS below the curve that represent the results of SA and MA algorithms in two cases P = 0.50 and P = 0.75 but not in the case P = 1.00. The total savings of ALNS are more than 7% compared to SA and more than 17% compared to MA in medium instances.

Compared results between proposed algorithms ALNS and proposed SA with MA algorithms in the case of BIG instances, an iteration number which can go up to one hundred thousand iterations and with several requests of one hundred requests, divided between two regions and two hundred geographic sites, our proposed ALNS algorithm shows the results obtained so far, exposed in Table 6 and Figure 8. The SA algorithm can outperform the ALNS and MA algorithm for instances with both distributions in regions P = 0.50 and P = 0.75. With a distribution parameter

Table 5. Total travels distances of ALNS, SA and MA with MEDIUM instances.

MEDIUM	r_{max}	∞		2000	
	T_{min}	0	100	0	100
P = 0.50	SA	5154	5330	8774	7029
	ALNS	4796	4940	6864	6971
	MA	6298	6800	9713	9580
P = 0.75	SA	5021	5011	7197	7362
	ALNS	4801	4886	6404	6926
	MA	6054	6069	8922	9120
P = 1.00	AS	4094	4061	7847	8140
	ALNS	3783	4098	7164	7480
	MA	4040	3945	6674	6398

Table 6. Total travels distances of ALNS, SA and MA with BIG-10 instances.

BIG-10	r_{max}	∞		5000	
	T_{min}	0	250	0	250
P = 0.50	SA	12459	12694	15401	15317
	ALNS	13460	13257	16600	16438
	MA	18003	18054	21114	21198
P = 0.75	SA	11896	11488	16215	16968
	ALNS	12510	12839	17144	17669
	MA	15581	15128	19294	19949
P = 1.00	AS	9755	9749	17110	17059
	ALNS	10169	10219	18257	17701
	MA	9803	10252	11933	12728

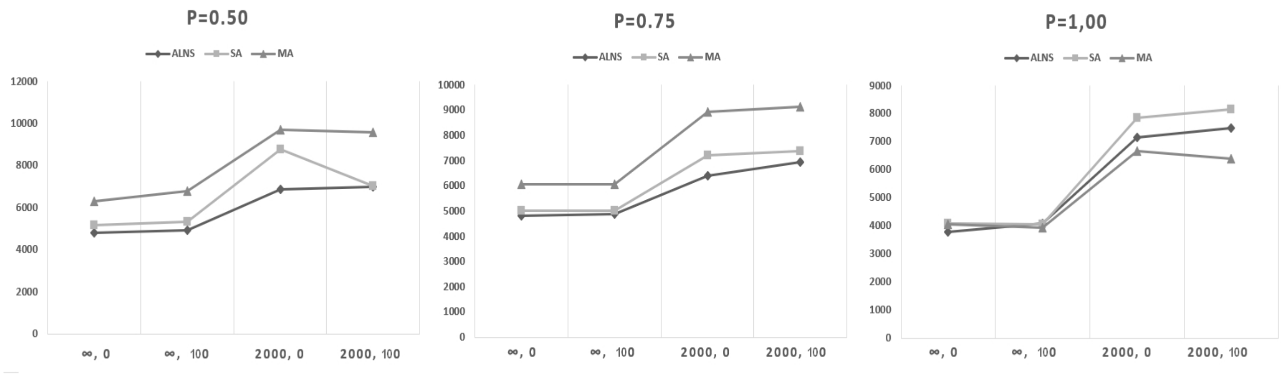

Figure 7. Line-Plot represent a comparison between ALNS, SA and MA heuristics with medium instances.

P equal to 1.00 the MA algorithm gives better results. The total savings of ALNS are more than 8% compared to Ma heuristic in big instances. The only loss of ALNS is -5% compared to the SA heuristic in big instances.

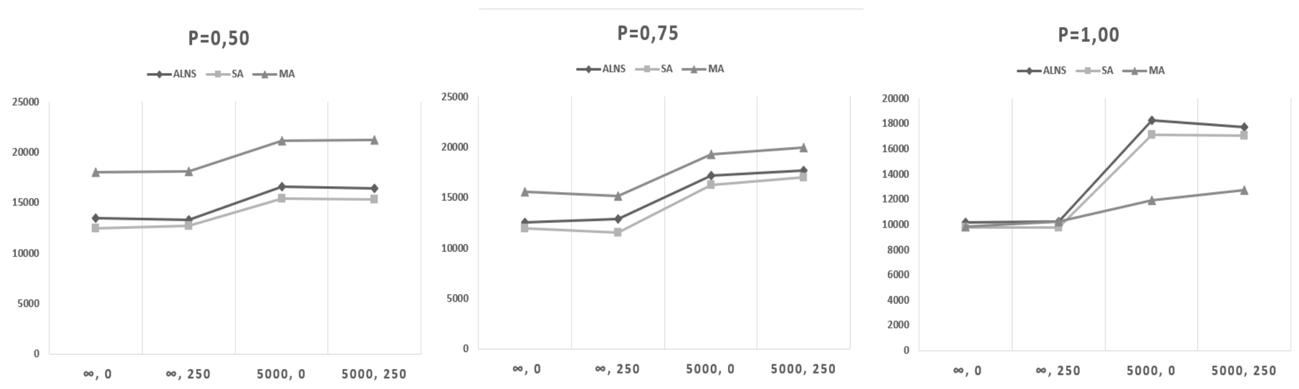

Figure 8. Line-Plot represent a comparison between ALNS, SA and MA heuristics with Big-10 instances.

These results can be explained firstly by the minimum time of vehicle change and the total ride time constraints are considered and Secondly, the number of pickup and delivery points equal to 200 (200 requests).

8. Conclusion

This work deals with the extension of the classical dial-a-ride problem through the addition of a central hub. This extension accords with the scenario on the distribution of a fleet of vehicles into two regions, each vehicle operates only in one specific region. Therefore, customer trips are split into allowed regions with the need for vehicle exchanges in the hub center, which requires respect to the transfer planning constraints.

Combined destroy and repair neighborhood search methods and 2-opt heuristics with the adaptive large neighborhood search are presented. The obtained results exceeded some proposed results in the literature for each instance problem of the dial-a-ride problems with transfer central hub.

We look to future work to improve the results of big instances and use more than one transfer center to solve the proposed problem with other more dedicated heuristics to compare our obtained results. Moreover to generalize the solution of the problem with several regions instead of two regions.

REFERENCES

1. G.B.Bdantzig, B.George and H.J.Ramser, *The truck dispatching problem*, Management science, 1959, vol. 6, no 1, p. 80–91.
2. G.Clarke and J.W.Wright, *Scheduling of vehicles from a central depot to a number of delivery points*, Operations research, 1964, vol. 12, no 4, p. 568–581.
3. S.Hammouda, M.Taffar and A.Lemouari, *A Simulated Annealing for The Resolution of "Dial-A-Ride-Problem with Transfer" Using Hybrid Neighborhood Methods*, IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS). IEEE, 2020. p. 1-6.
4. B.L.Golden, S.Raghavan and E.A.Wasil, *The vehicle routing problem: latest advances and new challenges*, Springer Science and Business Media, 2008.
5. S.N.Parragh, K.F.Doerner and R.Hartl, *A survey on pickup and delivery problems*, Journal für Betriebswirtschaft, 2008, vol. 58, no 2, p. 81–117.
6. R.Chevrier, A.Liefooghe, L.Jourdan and C.Dhaenens, *Solving a dial-a-ride problem with a hybrid evolutionary multi-objective approach: Application to demand responsive transport*, Applied Soft Computing, 2012, vol. 12, no 4, p. 1247–1258.
7. J.F.Cordeau and G.Laporte, *The dial-a-ride problem: models and algorithms*, Annals of operations research, 2007, vol. 153, no 1, p. 29–46.
8. R.Masson, F.Lehuédé and O.Péton, *The dial-a-ride problem with transfers*, Computers and Operations Research, 2014, vol. 41, p. 12–23.
9. D.M.Stein, *Scheduling dial-a-ride transportation systems*, Transportation Science, 1978, vol. 12, no 3, p. 232–249.
10. J.S.Shang and C.K.Cuff, *Multicriteria pickup and delivery problem with transfer opportunity*, Computers and Industrial Engineering, 1996, vol. 30, no 4, p. 631–645.
11. S.R.Thangiah, R.SAM, A.Fergany and S.Awan, *Real-time split-delivery pickup and delivery time window problems with transfers*, Central European Journal of Operations Research, 2007, vol. 15, no 4, p. 329–349.

12. C.E.Cortés, M.Matamalat and C.Contardo, *The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method*, European Journal of Operational Research, 2010, vol. 200, no 3, p. 711–724.
13. P.Bouros, D.Sacharidis, T.Dalamagas and T.Sellis, *Dynamic pickup and delivery with transfers*, International Symposium on Spatial and Temporal Databases. Springer, Berlin, Heidelberg, 2011. p. 112–129.
14. Y.Qu and J.F.Bard, *A GRASP with adaptive large neighborhood search for pickup and delivery problems with transshipment*, Computers and Operations Research, 2012, vol. 39, no 10, p. 2439–2456.
15. R.Masson, F.Lehuédé and O.Péton, *An adaptive large neighborhood search for the pickup and delivery problem with transfers*, Transportation Science, 2013, vol. 47, no 3, p. 344–355.
16. S.Deleplanque and A.Quilliot, *Dial-a-ride problem with time windows, transshipments, and dynamic transfer points*, IFAC Proceedings Volumes, 2013, vol. 46, no 9, p. 1256–1261.
17. J.F.Cordeau and G.Laporte, *A tabu search heuristic for the static multi-vehicle dial-a-ride problem*, Transportation Research Part B: Methodological, 2003, vol. 37, no 6, p. 579–594.
18. K.Braekers, A.Caris and G.K.Janssens, *Exact and meta-heuristic approach for a general heterogeneous dial-a-ride problem with multiple depots*, Transportation Research Part B: Methodological, 2014, vol. 67, p. 166–186.
19. S.N.Parragh, J.Pinho de Sousa and B.Almada-Lobo, *The dial-a-ride problem with split requests and profits*, Transportation Science, 2015, vol. 49, no 2, p. 311–334.
20. F.Neves-Moreira, P.Amorim, L.Guimarães and B.Almada-Lobo, *A long-haul freight transportation problem: Synchronizing resources to deliver requests passing through multiple transshipment locations*, European Journal of Operational Research, 2016, vol. 248, no 2, p. 487–506.
21. J.Schönberger, *Scheduling constraints in dial-a-ride problems with transfers: a metaheuristic approach incorporating a cross-route scheduling procedure with postponement opportunities*, Public Transport, 2017, vol. 9, no 1, p. 243–272.
22. N.Danloup, H.Allaoui and G.Goncalves, *A comparison of two meta-heuristics for the pickup and delivery problem with transshipment*, Computers and Operations Research, 2018, vol. 100, p. 155–171.
23. D.Wolfiner and J.J.Salazar-González, *The pickup and delivery problem with split loads and transshipments: A branch-and-cut solution approach*, European Journal of Operational Research, 2021, vol. 289, no 2, p. 470–484.
24. H.L.M.Kerivin, M.Lacroix, A.R.Mahjoub and A.Quilliot, *The splittable pickup and delivery problem with reloads*, European Journal of Industrial Engineering, 2008, vol. 2, no 2, p. 112–133.
25. S.Ropke and J.F.Cordeau, *Branch and cut and price for the pickup and delivery problem with time windows*, Transportation Science, 2009, vol. 43, no 3, p. 267–286.
26. D.Pisinger and S.Ropke, *A general heuristic for vehicle routing problems*, Computers and operations research, 2007, vol. 34, no 8, p. 2403–2435.
27. S.Ropke and D.Pisinger, *An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows*, Transportation science, 2006, vol. 40, no 4, p. 455–472.
28. P.Shaw, *Using constraint programming and local search methods to solve vehicle routing problems*, International conference on principles and practice of constraint programming. Springer, Berlin, Heidelberg, 1998. p. 417–431.
29. M.A.Masmoudi, M.Hosny, K.Braekers and A.Dammak, *Three effective metaheuristics to solve the multi-depot multi-trip heterogeneous dial-a-ride problem*, Transportation Research Part E: Logistics and Transportation Review, 2016, vol. 96, p. 60–80.
30. Y.Li, H.Chen and C.Prins, *Adaptive large neighborhood search for the pickup and delivery problem with time windows, profits, and reserved requests*, European Journal of Operational Research, 2016, vol. 252, no 1, p. 27–38.
31. B.Li, D.Krushinsky, T.Van Woensel and H.A.Reijers, *An adaptive large neighborhood search heuristic for the share-a-ride problem*, Computers and Operations Research, 2016, vol. 66, p. 170–180.
32. 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.
33. F.Glover and M.Laguna, *Tabu search*, Handbook of combinatorial optimization. Springer, Boston, MA, 1998. p. 2093–2229.