# Deterministic Annealing for Depot Optimization: Applications to the Dial-A-Ride Problem

Ramesh Ramasamy Pandi*, Song Guang Ho†, Sarat Chandra Nagavarapu‡ and Justin Dauwels§
School of Electrical and Electronic Engineering
Nanyang Technological University, 50 Nanyang Avenue, Singapore, 639798.
Email: *ramesh006@e.ntu.edu.sg, †hosongguang@gmail.com, ‡saratchandra.nagavarapu@gmail.com, §jdauwels@ntu.edu.sg

***Abstract*—This paper introduces a novel meta-heuristic approach to optimize depot locations for multi-vehicle shared mobility systems. Dial-a-ride problem (DARP) is considered as a case study here, in which routing and scheduling for door-to-door passenger transportation are performed while satisfying several constraints related to user convenience. Existing literature has not addressed the fundamental problem of depot location optimization for DARP, which can reduce cost, and in turn promote the use of shared mobility services to minimize carbon footprint. Thus, there is a great need for fleet management systems to employ a multi-depot vehicle dispatch mechanism with intrinsic depot location optimization. In this work, we propose a deterministic annealing meta-heuristic to optimize depot locations for the dial-a-ride problem. Numerical experiments are conducted on several DARP benchmark instances from the literature, which can be categorized as small, medium and large based on their problem size. For all tested instances, the proposed algorithm attains solutions with travel cost better than that of the best-known solutions. It is also observed that the travel cost is reduced up to 6.13% when compared to the best-known solutions.**

***Index Terms*—Shared mobility system, depot location optimization, deterministic annealing, dial-a-ride problem, meta-heuristic.**

## I. Introduction

On-demand mobility has been an interesting and important area of research in today's era, as there is an ever-increasing demand for smart and efficient transportation systems. Effective fleet operations often require the flexibility to adjust and respond to customer demands, to ensure user satisfaction. Dial-a-ride problem (DARP) in the literature [1–7] addresses this issue by designing an optimal route scheme for a fleet of vehicles by minimizing the overall travel cost. Several constraints such as vehicle load, route duration, passenger ride time and time windows must be satisfied. The conventional dial-a-ride systems [1] consider a single depot, from which the vehicles get dispatched to serve the passengers. However, in most of the cases, due to the demographic distribution of the requests, vehicles may need to travel long distances from the depot to perform passenger pick-up/drop-off services. Thus, the single-depot vehicle dispatch mechanism greatly affects the cost and overall customer satisfaction. To address this issue, it is essential to incorporate multi-depot vehicle dispatch into ride-sharing systems and to develop an algorithmic approach that identifies optimal depot locations and enhances the operational efficiency.

Many real-time emergency services, namely, medical ambulance services, fire and rescue services, law enforcement, etc., require strategic placement of their infrastructure to minimize the response time. In recent years, one of the primary focuses of the automobile industry is to come up with alternative vehicular technologies such as electrical vehicles (EVs) to prevent carbon emissions and the use of fossil fuels [8]. The utilization of EVs for everyday activities would rise up, only if the charging infrastructure available to the public is made abundant and reachable. However, the existing infrastructure to charge these vehicles is either slender or disorganized. Therefore, optimizing the locations of charging stations is vital to promote their usage to achieve green transportation [9].

Despite the abundance of possible applications through depot location optimization, this problem has never been addressed in the context of passenger transportation. An earlier study addressing the depot location optimization for vehicle routing problem (VRP) proposed a particle swarm optimization based approach [10]. However, when compared to VRP, DARP is an advanced, user-oriented form of public transport characterized by flexible routing and scheduling of vehicles operating in shared-ride mode between pick-up and drop-off locations according to passengers needs. Recently, a new variant of DARP, multi-depot heterogeneous DARP [4] has been introduced, in which four depots are placed at the center of each quadrant in a specified service area, where the placement of depots is invariant of the problem size and geographical distribution of the customer demands.

Therefore, we introduce a deterministic annealing meta-heuristic to optimize the depot locations for the DARP. The major contributions of the paper are summarized as follows:

- A bilevel optimization framework is developed, in which the depot locations and route plans are optimized to minimize the travel cost for dial-a-ride problem.
- A deterministic annealing heuristic is proposed to optimize the depot locations, while a k-means clustering is designed to construct a set of good initial depot locations.
- The performance of the proposed algorithm is investigated on several DARP benchmark instances and compared with the best-known solutions.

The remainder of the paper is organized as follows: In Section II, we brief the problem description of the DARP considered in this paper. In Section III, we propose a deterministic annealing to solve DARP. We present the numerical results in Section IV and offer concluding remarks in Section V.

## II. Background

### A. Terminology

The terminology followed in this paper is presented here.

- $Vertex$: Pick-up, drop-off or the depot location.
- $Arc$: A route connecting two vertices.
- $n$: Number of requests, $i$ = 1, 2, $\cdots$, $n$.
- $m$: Fleet size or the number of vehicles in the fleet, $k$ = 1, 2, $\cdots$, $m$.
- $D_i$: Departure time; time at which a vehicle leaves a vertex $i$.
- $B_i$: Arrival time; time at which a vehicle begins its service at a vertex $i$.
- $d_k$: Service time; time taken by the $k^{th}$ vehicle to provide service at a vertex.
- $W_i$: Waiting time; duration for which a vehicle waits at a vertex $i$ before starting the service.
- $L$: Maximum ride time; total time spent by each passenger in a vehicle.
- $L_i$: Time spent by the $i^{th}$ customer in a vehicle.
- $Q_k$: Maximum vehicle load.
- $q_i$: Vehicle load available at the $i^{th}$ vertex.
- $l_i$: The latest time beyond which the $i^{th}$ vertex can not be served.
- $T_k$: Maximum route duration.
- $c_{ij}^k$: Routing cost for the $k^{th}$ vehicle to travel from the $i^{th}$ to the $j^{th}$ vertex.
- $e_{ij}^k$: An arc between two vertices $i$ and $j$ visited by $k^{th}$ vehicle.
- $r_{best}$: Best solution found so far $R$.
- $r$: Current solution of the algorithm.
- $r^*$: Current solution $r$ after perturbation.
- $c(r)$: Travel cost of a solution $r$.
- $upset$: Factor by which a coordinate is perturbed.
- $x_i$: Magnitude of x-axis in configuration coordinate $i$.
- $y_i$: Magnitude of y-axis in configuration coordinate $i$.
- $x_d$: Magnitude of x-axis of the depot location.
- $x_d$: Magnitude of y-axis of the depot location.
- $iteration$: Current number of iteration.
- $chain$: Current number of markov chain.
- $num\_of\_iteration$: Total number of iterations.
- $markov\_chain\_length$: Length of the *chain*.
- $T$: Current temperature of the system.
- $T_{max}$: Maximum temperature of the system.
- $T_{min}$: Minimum temperature of the system.
- $\Phi$: Cooling factor of the system.

### B. Problem description

The static multi-depot dial-a-ride problem is defined as follows. Several passengers place requests for the door-to-door transportation service between specific origin and destination locations. Each request $i$ may specify time-window either on their departure or arrival. Each vehicle $k$ has a route duration limit and a capacity constraint. In order to ensure mandatory high-quality service, a maximum ride time limit is set for each passenger. The depot location indicates the origin and destination for each vehicle. A vehicle $k$ must start from a depot, and end at a depot after completing the service. In addition, a vehicle can only leave the depot if a passenger request is assigned to it. A predetermined service time $d_k$ is set at both pick-up and drop-off locations. The main objective is to design vehicle routes and schedules in order to provide transportation service to all passengers at minimum travel cost while satisfying all the constraints.

In this DARP, $m$ vehicles are spread across different depot locations. Each vehicle can leave from a depot and return to any depot. In other words, a vehicle may have different starting and ending depots. A vehicle $k$ that leaves a depot, can return to the nearest depot location after serving the last passenger alloted to it. For the standard DARP [1], the search space can be defined as, $\mathbb{S} = \{r_1, r_2, \cdots\}$, where $r_1, r_2, \cdots$ are solutions. Every route for a vehicle $k$ starts and ends at the depot and the departure vertex $v_i$ and arrival vertex $v_{i+n}$ must belong to the same route, and the arrival vertex $v_{i+n}$ is visited after departure vertex $v_i$. The constraints to be satisfied are: (i) the load of vehicle $k$ cannot exceed the preset load bound $Q_k$ at any time; (ii) the total route duration of a vehicle $k$ cannot exceed the preset duration bound $T_k$; (iii) the ride time of any passenger cannot exceed the ride time bound $L$; the time window set by the customer must not be violated. Therefore, four major constraints exist in dial-a-ride problem: load, duration, time window, and ride time constraints. Load constraint violation $q(r)$ occurs when the number of passenger in a vehicle $k$ exceeds its load limit $Q_k$; duration constraint violation $d(r)$ happens when a vehicle $k$ exceeds its duration limit $T_k$; time window constraint violation $w(r)$ appears when time at start of service $B_i$ is earlier or later than the earliest or latest limit of time window $l_i$ respectively; ride time constraint violation $t(r)$ occurs when a passenger is transported longer time than the ride time limit $L$. The constraints are as follows:

$$q(r) = \sum_{\forall k} \max(q_{k,\max} - Q_k, 0) \tag{1}$$

$$d(r) = \sum_{\forall k} \max(d_k - T_k, 0) \tag{2}$$

$$w(r) = \sum_{\forall i} \left[\max(B_i - l_i, 0) + \max(B_{i+n} - l_{i+n}, 0)\right] \tag{3}$$

$$t(r) = \sum_{\forall i} \max(L_i - L, 0). \tag{4}$$

In the next section, we present a deterministic annealing meta-heuristic to optimize depot locations for DARP.

## III. Deterministic Annealing

In this section, a bilevel optimization framework is designed, in which two levels of optimizations are performed: the outer layer optimizes the depot locations and the inner layer optimizes the route plan. A new deterministic annealing (DA) meta-heuristic is developed for the depot location optimization. The objective here is to minimize the travel cost, which we consider as the travel time of the passengers. Here, we define a solution $r$ as a combination of two variables: depot locations (outer layer) and route plan (inner layer).

The idea of simulated annealing [11] originates from the annealing concept of metallurgy. The diffusion of atoms within a solid material by slow cooling to achieve equilibrium state in a metal is referred to as *annealing*. During the cooling process, at each temperature level, a certain amount of time lapses until the material progresses towards its equilibrium, to ensure stable crystalline structure of the metal. The translational idea of annealing into a meta-heuristic is to characterize the minimization problem of an objective function as the minimization of free energy of the system.

Deterministic annealing is a variant of simulated annealing meta-heuristic for solving optimization problems. In this approach, a temperature controlled random walk is performed to identify the optimal combination of depot locations that yield a route plan with minimum travel cost. On each iteration, the next solution is selected by performing a local search from the current solution, and the acceptance decision is based on the temperature of the system. There are two types of acceptance: metropolis and threshold criterion. Previous studies have shown that SA has higher computational complexity due to the utilization of a metropolis function. Therefore, we employ DA to deterministically optimize the system. A certain number of iterations is carried out at each temperature, in turn, the algorithm avoids getting trapped at local minimum solutions while exploring the vast amount of search space in an attempt to find the global optimal solution. An overview of the proposed deterministic annealing is outlined in Fig. 1, and the implementations details are described in Algorithm 1.

### A. Algorithm

The flow of the algorithm is as follows:

- The temperature $T$ of the system is set as the maximum temperature $T_{max}$.
- For each request $i$, a configuration coordinate $\{x_i, y_i\}$ is initialized as the midpoint of arrival vertex $v_i$ and departure vertex $v_{i+n}$.
- A construction heuristic is invoked to initialize a set of depot locations $\{x_d, y_d\}$, which depends on the configuration coordinates $\{x_i, y_i\}$ of each request $i$. The details of the construction heuristic is described in Section III-B.
- An initial route plan is constructed as follows. i) create $m$ empty set of routes, ii) insert the departure and arrival vertices of each request $i$ into the routes, while ensuring both the vertices belong to same route $k$, and the departure vertex precedes the arrival vertex.

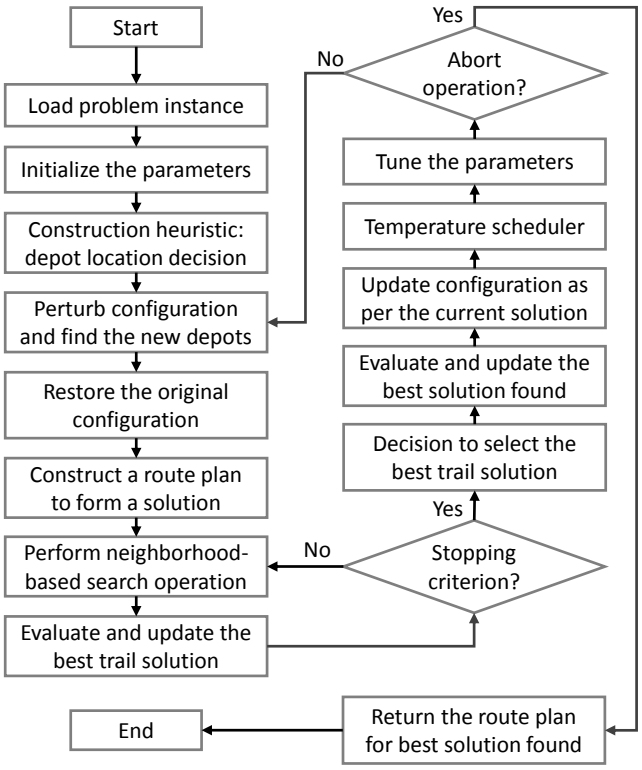



Figure 1: Flowchart: Deterministic Annealing.

**Algorithm 1** Deterministic Annealing

**Input:** $T_0$: Initial temperature; All constraints
**Output:** $\{x_d, y_d\}_{\text{best}}$: the best depot location obtained; $r_{\text{best}}$: the best route obtained

$T \leftarrow T_{max}$.
Initialize configuration coordinates $\{x_i, y_i\}$.
$\{x_d, y_d\} \leftarrow$ *Construct*$(\{x_i, y_i\})$.
$r_{\text{best}} \leftarrow \phi$ where $c(r_{\text{best}}) = \infty$.
Construct an initial solution $r$ with $\{x_d, y_d\}$
**for** *iteration* = 1 to *num_of_iterations* **do**
  **for** *chain* = 1 to *markov_chain_length* **do**
    $\{x_i, y_i\}$*,$\{x_d, y_d\}$* $\leftarrow$ *Perturb*$(\{x_i, y_i\},\{x_d, y_d\})$.
    $r$* $\leftarrow$ *OptimizeRoute*$(\{x_d, y_d\}$*, all constraints).
    **if** $c(r^*) < c(r_{\text{best}})$ **then**
      $\{x_d, y_d\}_{\text{best}} \leftarrow \{x_d, y_d\}^*$.
      $r_{\text{best}} \leftarrow r^*$.
    **end if**
    **if** $c(r^*) < c(r) + T$ **then**
      $\{x_d, y_d\} \leftarrow \{x_d, y_d\}^*$.
      $r \leftarrow r^*$.
    **end if**
  **end for**
  $T = T * \Phi$.
  **if** $T < T_{min}$ **then**
    $T = T_{max}$.
  **end if**
**end for**

- On each *iteration*, a *chain* of local search operation (Section III-C) is carried out for *markov_chain_length* number of times while keeping the temperature $T$ constant. On each *chain*, the depot locations are perturbed to generate a new set of depots, which is followed by the generation of route plan (Section III-D). At every *chain*, the procedure for the selection of a best admissible solution is as follows. If the current solution $r^*$ has the travel cost less than that of the best solution $r_{best}$ found so far, then the best solution object $r_{best}$ is updated with the current solution. However, the current solution is still accepted if its travel cost $c(r^*)$ is less than that of the summation of travel cost of the best solution $c(r_{best})$ and temperature $T$ of the system.
- At the end of each *iteration*, temperature $T$ of the system is multiplied by a factor $\Phi$. If the resulting temperature $T$ is less than the minimum temperature $T_{min}$, then it will be reset to the maximum temperature $T_{max}$ of the system.
- Once the stopping criterion is reached, the best solution $r_{best}$ that has depot locations and a set of routes with minimum travel cost is returned. Here, the stopping criterion is set as a fixed number of iterations.

### B. Construction heuristic: K-means clustering

A construction heuristic defines a set of rules to generate an initial set of depot locations. This algorithm requires the configuration coordinates $\{x_i, y_i\}$ of each request $i$ as the input, and generates a set of depot locations $\{x_d, y_d\}$ as the output. The construction heuristic is inspired by the k-means clustering approach. The procedure is as follows. 1) a set of depot locations $\{x_d, y_d\}$ are initialized randomly, 2) each request $i$ is assigned to its nearest depot $d$ based on the Euclidean distance from its configuration coordinate to the depot, 3) each depot coordinates $\{x_d, y_d\}$ are designated as a geographical mean of all configuration coordinates $\{x_i, y_i\}$ of the requests $i$ that are assigned to it, 4) repeat step 2-3 until the depot locations $\{x_d, y_d\}$ does not change for $N$ number of iterations. Implementation details are explained in Algorithm 2.

**Algorithm 2** Construct initial depot locations using k-means clustering: *Construct*()

**Input:** $\{x_i, y_i\}$: Configuration coordinates of all request $i \in I$
**Output:** $\{x_d, y_d\}$: Coordinates of all depot $d \in D$

**for all** depot $d \in D$ **do**
  $\{x_d, y_d\} \leftarrow$ random coordinates $\in$ (-10,10).
**end for**
**repeat**
  **for all** request $i \in I$ **do**
    Cluster request $i$ to the nearest depot $d$.
  **end for**
  **for all** depot $d \in D$ **do**
    $\{x_d, y_d\} \leftarrow$ mean of $\{x_i, y_i\}$ for all request $i$ clustered to depot $d$.
  **end for**
**until** no change in all $\{x_d, y_d\}$ for $N$ iterations

### C. Local search: Roulette wheel operator

Local search operator generates a new set of depot locations based on the current set of depots, and configuration coordinates, by applying local changes. In this algorithm, the configuration points $\{x_i, y_i\}$ of the current solution $r$ are the input, and a new set of configuration points $\{x_i, y_i\}$* and new depot locations $\{x_d, y_d\}$* are returned as the output.

A roulette wheel operator is introduced for local search, to be guided by the deterministic annealing meta-heuristic. The operator procedure is as follows. i) a depot $d$ is randomly chosen from the current set of depots, ii) for each request $i$ that is assigned to the chosen depot $d$, their respective configuration coordinates $\{x_i, y_i\}$ are altered by $\{x_u, y_u\}$, which are sampled from a uniform probability distribution function, iii) all the requests are assigned to their nearest depot locations based on the Euclidean distance, iv) for each depot $d$, its new coordinates $\{x_d, y_d\}$ are designated as a geographical mean of the configuration coordinates $\{x_i, y_i\}$ of all requests $i$ that are assigned to it. Algorithm 3 outlines the implementation details of the roulette wheel operator.

**Algorithm 3** Perturb depot locations using roulette wheel operator: *Perturb*()

**Input:** $\{x_i, y_i\}$: Configuration coordinates of all request $i \in I$ ; $\{x_d, y_d\}$: Coordinates of all depot $d \in D$
**Output:** $\{x_i, y_i\}$*: Perturbed configuration coordinates of all request $i \in I$; $\{x_d, y_d\}$*: Perturbed coordinates of all depot $d \in D$

$d_{\text{rand}} \leftarrow$ random $d \in D$.
$\{x_u, y_u\} \leftarrow$ random upset numbers $\in$ (-5,5).
**for all** request $i$ clustered to depot $d_{\text{rand}}$ **do**
  $\{x_i, y_i\}^* \leftarrow \{x_i, y_i\}$ + $\{x_u, y_u\}$.
**end for**
**for all** request $i \in I$ **do**
  Cluster request $i$ to the nearest depot $d$.
**end for**
**for all** depot $d \in D$ **do**
  $\{x_d, y_d\}^* \leftarrow$ mean of $\{x_i, y_i\}$ for all request $i$ clustered to depot $d$.
**end for**

### D. Route Optimization

The goal of route optimization is to generate a route plan that has minimum travel cost. Here, we employ a multi-atomic annealing heuristic [12] for the route optimization. Using the coordinates of all depots $d \in D$, the algorithm returns a solution $r^*$ with the best route plan. The procedure for route optimization is as follows. 1) a solution $r_{trial}$ is constructed randomly, 2) *burn* and *reform* operators [12] are applied to solution $r_{trial}$, 3) the solution $r_{trial}$ is evaluated using a scheduling heuristic [1], 4) the best solution $r^*$ is

updated if $r_{trial}$ has a travel cost lower than that of the best solution $r^*$, besides being both feasible and complete, 5) repeat steps 2-4 until the termination condition of route optimization is satisfied. The termination condition is set as a predefined number of optimization steps.

**Algorithm 4** Optimize route using multi-atomic annealing: *OptimizeRoute*()

**Input:** $\{x_d, y_d\}$: Coordinates of all depot $d \in D$; all constraints
**Output:** $r$*: best route found
1: Initialize internal temperature $T_{internal}$ and best route $r^*$
2: Construct an initial solution $r_0$
3: Set the solution $r_{trial}$ = $r_0$
4: **repeat**
5: Burn the solution $r_{trial}$ according to $T_{internal}$
6: Reform the solution $r_{trial}$
7: **if** $r_{trial} \in (\mathbb{F} \cap \mathbb{C})$ **and** $c(r_{trial}) < c(r^*)$ **then**
8: Update the best solution $r^*$ = $r_{trial}$
9: **end if**
10: Update internal temperature $T_{internal}$
11: **until** termination criterion

### *E. Temperature scheduler*

Temperature scheduler controls the cooling rate of the annealing process. In general, the temperature should be reduced gradually to attain thermodynamic equilibrium state. In DA, at each temperature $T$, a local search operation is performed several times. Therefore, this process can be referred to as markov chain. In this algorithm, the temperature scheduler initializes temperature $T$ of the system to $T_{max}$. The markov chain length represents the number of times that a local search is performed at a particular temperature, and it is set to *markov_chain_length*. The temperature $T$ is multiplied by a factor $\Phi$, whenever the *chain* reaches its limit *markov_chain_length*. The system is reheated to $T_{max}$ whenever the temperature $T$ reaches the minimum temperature $T_{min}$. The parameters $\Phi$, $T_{min}$ and $T_{max}$ are experimentally set for the algorithm before optimization process begins.

In the next section, we present the numerical results obtained for the proposed deterministic annealing meta-heuristic on the benchmark test instances from the literature.

## IV. Numerical Results

The proposed algorithm is implemented in C++. Numerical experiments have been conducted using a computer running 2.1 GHz Intel Xeon E-2620 v4 processor on various DARP instances from the literature.

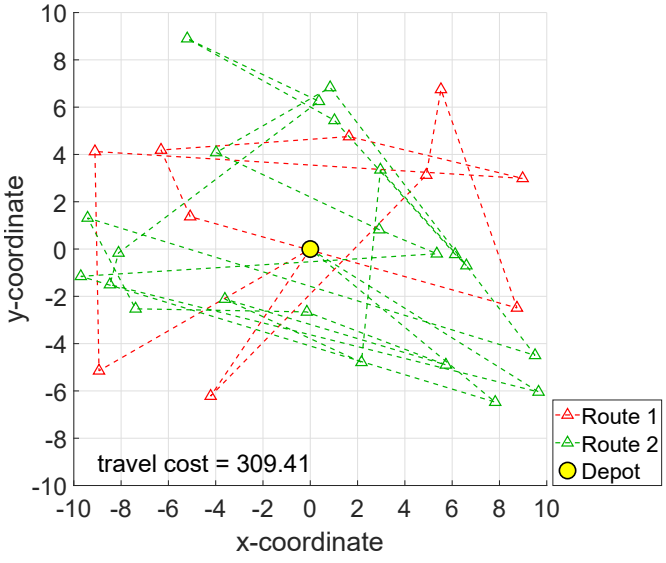
The DARP instances R1a-R3a and R1b-R3b are obtained from [1]. In these instances, the number of requests ranges from 24 to 72, and the number of vehicles ranges from 3 to 7. Each request is randomly generated in the Euclidean space of $[-10, 10]^2$. For R1a-R3a, the time window length is 15 min. For R1b-R3b, the time window length is 45 min. They have different geographical distribution of pick-up and drop-off locations, along with their time windows, which makes each problem instance unique in terms of difficulty. The service time for each vehicle at any location is set as 10 min. The capacity of each vehicle is 6, with a maximum route duration limit of 480 min. The maximum passenger ride time limit is 90 min. Furthermore, a set of instances from [13] is also considered for investigation. These instances are referred to as a and b sets, in which we considered the instances a2-16 to a2-24, and a2-20 to a2-24. The definition of the problem remains the same for these instances as well, except the following factors. The time-window length for the critical vertices is set as 15 min. For the a instances, the vehicle capacity is set as 1, the service time limit is 10 min, and the maximum ride time limit is 30 min. For the b instances, the vehicle capacity is set as a number uniformly distributed between 1 and 6, the service time limit is equal to vehicle capacity, and the maximum ride time limit is 45 min. Therefore, a significantly diverse set of instances are obtained from the literature to evaluate the performance of the proposed algorithm. A detailed discussion of the obtained results is as follows.

The DARP instances are categorized into small, medium and large, based on their problem size. Each instance corresponds to a different geographical distribution of the requests, and constraints associated with it. Figures 2a, 2c and 2e illustrate the route plan for the best-known solution for a small, medium and large-scale instance. Similarly, the Figures 2b, 2d and 2f illustrate the route plan attained the proposed deterministic annealing for those instances.

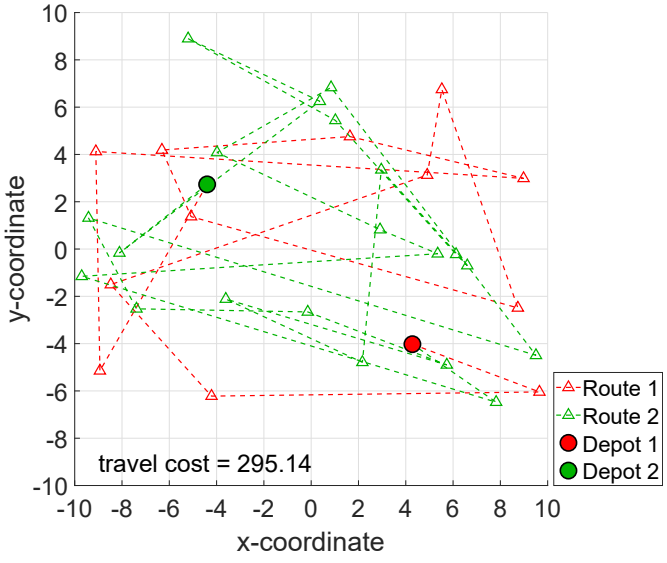
Tables I-III indicate the results attained for the proposed algorithm on various standard DARP instances from the literature. Table IV provides the comparison of results attained by the proposed algorithm against the best-known solutions from the literature. In this table, *Depots* indicate the total number of depots used to solve the given instance; *BKS* denotes the travel cost of the best-known solution without depot location optimization for a given instance; *Savings* denote the travel cost reduction in percentage attained by DA when compared to *BKS*. The definition for *Savings* is provided by:

$$\textit{Savings}\ (\%) \ = \ \frac{\textit{BKS} \ - \ \textit{travelcost}}{\textit{BKS}} \times 100. \qquad (5)$$

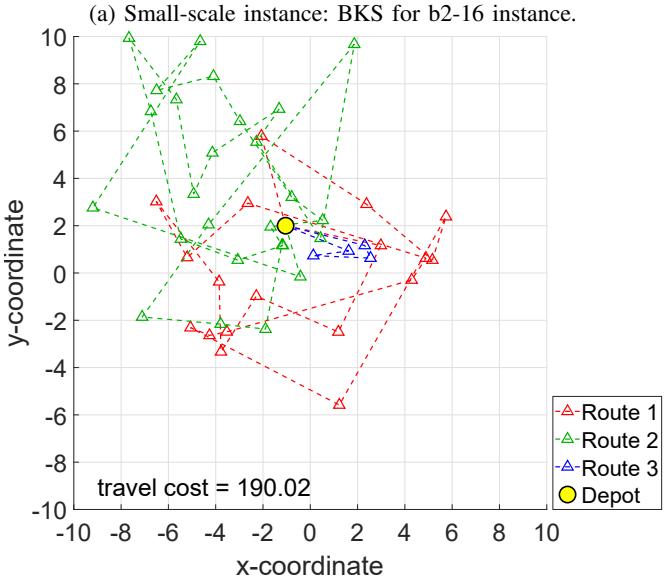
From the results, it can be observed that the depot locations are optimized to improve the overall travel cost. For the tested instances, the average travel cost percentage savings for small, medium and large-scale instances are 2.35%, 3.68%, and 4.49% respectively, when compared to the best-known solution. Notably, the proposed algorithm attained solutions up to 6.13% lesser cost when compared to the best-known solutions. From Table IV, it is evident that the proposed DA attains solutions with travel cost better than that of the best-known solutions for all the tested instances. It is also observed that the bigger the problem size, the more notable the *Savings* is. We believe that this is due to the flexibility that a larger problem offers in terms of more number of depots that in turn allows them to be widely scattered across the service area.



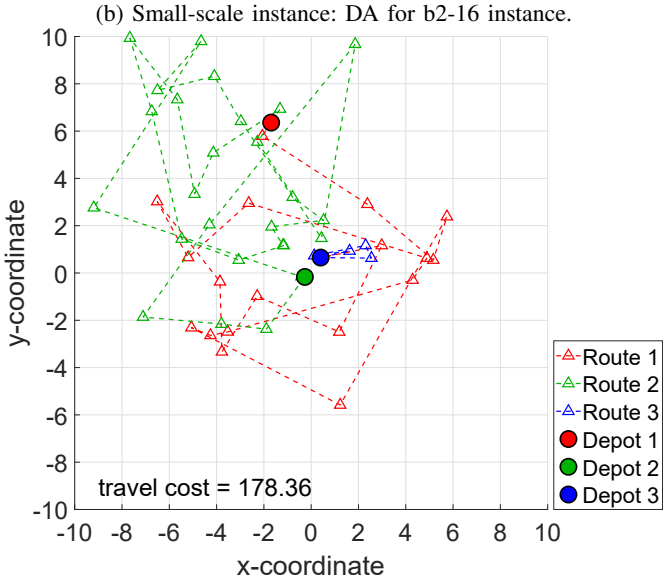

(a) Small-scale instance: BKS for b2-16 instance.



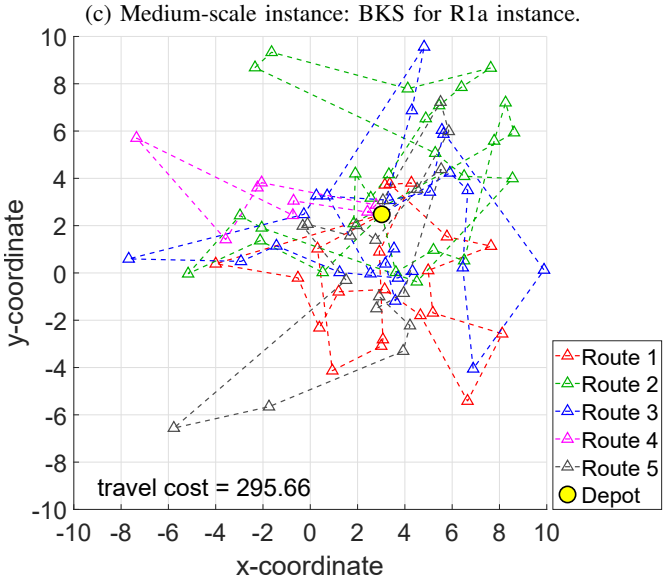

(b) Small-scale instance: DA for b2-16 instance.



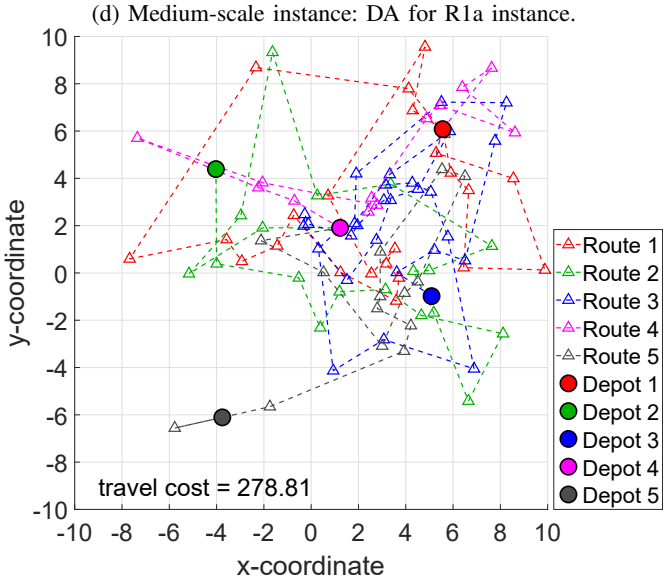

(c) Medium-scale instance: BKS for R1a instance.




(d) Medium-scale instance: DA for R1a instance.




(e) Large-scale instance: BKS for R2b instance.




(f) Large-scale instance: DA for R2b instance.

Figure 2: Comparison of route plans attained by the proposed DA vs BKS on various DARP instances.

| Instance | BKS | Deterministic annealing | | | | | | | | Travel cost |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Depot | | | Vehicle | | | | | |
| | | Label | Coordinate x | Coordinate y | Tag | Departure | Arrival | Passengers | Duration | |
| a2-16 [13] | 294.24 [13] | $d_1$ | 5.40 | -1.14 | $v_1$ | $d_1$ | $d_2$ | 10 | 395.06 | **289.18** |
| | | $d_2$ | -5.20 | -1.15 | $v_2$ | $d_2$ | $d_2$ | 6 | 377.44 | |
| a2-16 [13] | 309.41 [13] | $d_1$ | 4.28 | -4.02 | $v_1$ | $d_1$ | $d_2$ | 6 | 407.97 | **295.14** |
| | | $d_2$ | -4.40 | 2.73 | $v_2$ | $d_2$ | $d_1$ | 10 | 276.02 | |
| a2-16 [13] | 344.83 [13] | $d_1$ | -4.06 | 1.77 | $v_1$ | $d_1$ | $d_2$ | 14 | 565.83 | **335.68** |
| | | $d_2$ | 6.04 | -2.68 | $v_2$ | $d_2$ | $d_2$ | 6 | 467.03 | |
| a2-16 [13] | 332.64 [13] | $d_1$ | 2.34 | 3.18 | $v_1$ | $d_1$ | $d_2$ | 16 | 450.70 | **331.13** |
| | | $d_2$ | -2.80 | -3.05 | $v_2$ | $d_2$ | $d_2$ | 12 | 506.59 | |

Table I: Results attained by the proposed DA on small-scale DARP instances [13].

| Instance | BKS | Deterministic annealing | | | | | | | | Travel cost |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Depot | | | Vehicle | | | | | |
| | | Label | Coordinate x | Coordinate y | Tag | Departure | Arrival | Passengers | Duration | |
| a2-24 [13] | 431.12 [13] | $d_1$ | -4.18 | 1.62 | $v_1$ | $d_1$ | $d_2$ | 12 | 554.20 | **418.29** |
| | | $d_2$ | 6.80 | 3.14 | $v_2$ | $d_2$ | $d_1$ | 12 | 529.53 | |
| b2-24 [13] | 444.71 [13] | $d_1$ | -0.73 | -5.85 | $v_1$ | $d_1$ | $d_1$ | 12 | 526.77 | **435.33** |
| | | $d_2$ | 1.51 | 6.66 | $v_2$ | $d_2$ | $d_2$ | 12 | 558.44 | |
| R1a [1] | 190.02 [3] | $d_1$ | -1.69 | 6.36 | $v_1$ | $d_1$ | $d_3$ | 9 | 346.92 | **178.36** |
| | | $d_2$ | -0.26 | -0.16 | $v_2$ | $d_2$ | $d_2$ | 2 | 441.37 | |
| | | $d_2$ | 0.40 | 0.65 | $v_2$ | $d_3$ | $d_3$ | 13 | 81.20 | |
| R1b [1] | 164.46 [3] | $d_1$ | 2.80 | 2.53 | $v_1$ | $d_1$ | $d_1$ | 8 | 255.50 | **158.62** |
| | | $d_2$ | -5.12 | 6.28 | $v_2$ | $d_2$ | $d_3$ | 8 | 411.14 | |
| | | $d_2$ | -3.50 | -0.76 | $v_2$ | $d_3$ | $d_3$ | 8 | 363.02 | |

Table II: Results attained by the proposed DA on medium-scale DARP instances [1][13].

| Instance | BKS | Deterministic annealing | | | | | | | | Travel cost |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Depot | | | Vehicle | | | | | |
| | | Label | Coordinate x | Coordinate y | Tag | Departure | Arrival | Passengers | Duration | |
| R2a [1] | 301.33 [3] | $d_1$ | 1.22 | 1.90 | $v_1$ | $d_1$ | $d_3$ | 8 | 386.11 | **285.76** |
| | | $d_2$ | -3.75 | -6.10 | $v_2$ | $d_2$ | $d_1$ | 8 | 378.49 | |
| | | $d_3$ | 5.56 | 6.08 | $v_3$ | $d_2$ | $d_5$ | 14 | 406.29 | |
| | | $d_4$ | -4.02 | 4.39 | $v_4$ | $d_2$ | $d_5$ | 6 | 340.00 | |
| | | $d_5$ | 5.08 | -0.98 | $v_5$ | $d_2$ | $d_1$ | 12 | 457.46 | |
| R2b [1] | 295.66 [3] | $d_1$ | 5.56 | 6.08 | $v_1$ | $d_1$ | $d_1$ | 12 | 398.44 | **278.81** |
| | | $d_2$ | -4.02 | 4.39 | $v_2$ | $d_2$ | $d_4$ | 9 | 372.94 | |
| | | $d_3$ | 5.08 | -0.98 | $v_3$ | $d_3$ | $d_4$ | 14 | 373.46 | |
| | | $d_4$ | 1.22 | 1.90 | $v_4$ | $d_4$ | $d_1$ | 6 | 186.71 | |
| | | $d_5$ | -3.75 | -6.10 | $v_5$ | $d_5$ | $d_4$ | 7 | 186.05 | |
| R3a [1] | 532.42 [3] | $d_1$ | -4.75 | -2.90 | $v_1$ | $d_1$ | $d_7$ | 13 | 462.89 | **525.39** |
| | | $d_2$ | 0.80 | -3.88 | $v_2$ | $d_2$ | $d_5$ | 13 | 401.18 | |
| | | $d_3$ | -4.38 | 6.18 | $v_3$ | $d_3$ | $d_2$ | 11 | 370.31 | |
| | | $d_4$ | 6.45 | -4.88 | $v_4$ | $d_4$ | $d_7$ | 8 | 375.83 | |
| | | $d_5$ | 0.71 | 1.98 | $v_5$ | $d_5$ | $d_5$ | 14 | 393.58 | |
| | | $d_6$ | -7.43 | 2.01 | $v_6$ | $d_6$ | $d_2$ | 13 | 458.40 | |
| | | $d_7$ | 0.00 | 0.00 | $v_7$ | $d_7$ | $d_7$ | 0 | 0.00 | |
| R3b [1] | 484.82 [14] | $d_1$ | 2.87 | 0.10 | $v_1$ | $d_1$ | $d_6$ | 8 | 360.72 | **456.35** |
| | | $d_2$ | -0.77 | 3.24 | $v_2$ | $d_2$ | $d_5$ | 9 | 325.52 | |
| | | $d_3$ | -5.07 | 2.64 | $v_3$ | $d_3$ | $d_7$ | 13 | 372.26 | |
| | | $d_4$ | -0.22 | -5.20 | $v_4$ | $d_4$ | $d_5$ | 9 | 365.55 | |
| | | $d_5$ | -6.56 | 4.78 | $v_5$ | $d_5$ | $d_2$ | 9 | 362.48 | |
| | | $d_6$ | 6.19 | -5.56 | $v_6$ | $d_6$ | $d_3$ | 11 | 356.50 | |
| | | $d_7$ | 0.00 | 0.00 | $v_7$ | $d_7$ | $d_2$ | 13 | 383.05 | |

Table III: Results attained by the proposed DA on large-scale DARP instances [1].

| Problem | Instance | Demand | Fleet | BKS | | | Deterministic annealing | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Depots | Travel cost | Savings (%) | Depots | Travel cost | Savings (%) |
| Small-scale | a2-16 [13] | 16 | 2 | 1 | 294.24 [13] | 0.00 | 2 | **289.18** | **1.72** |
| | b2-16 [13] | 16 | 2 | 1 | 309.41 [13] | 0.00 | 2 | **295.14** | **4.61** |
| | a2-20 [13] | 20 | 2 | 1 | 344.83 [13] | 0.00 | 2 | **335.68** | **2.65** |
| | b2-20 [13] | 20 | 2 | 1 | 332.64 [13] | 0.00 | 2 | **331.13** | **0.45** |
| Medium-scale | a2-24 [13] | 24 | 2 | 1 | 431.12 [13] | 0.00 | 2 | **418.29** | **2.97** |
| | b2-24 [13] | 24 | 2 | 1 | 444.71 [13] | 0.00 | 2 | **435.33** | **2.10** |
| | R1a [1] | 24 | 3 | 1 | 190.02 [3] | 0.00 | 3 | **178.36** | **6.13** |
| | R1b [1] | 24 | 3 | 1 | 164.46 [3] | 0.00 | 3 | **158.62** | **3.55** |
| Large-scale | R2a [1] | 48 | 5 | 1 | 301.33 [3] | 0.00 | 5 | **285.76** | **5.16** |
| | R2b [1] | 48 | 5 | 1 | 295.66 [3] | 0.00 | 5 | **278.81** | **5.69** |
| | R3a [1] | 72 | 7 | 1 | 532.42 [3] | 0.00 | 7 | **525.39** | **1.24** |
| | R3b [1] | 72 | 7 | 1 | 484.82 [14] | 0.00 | 7 | **456.35** | **5.87** |

*Note: '*' highlighted numbers indicate the best travel cost by which all passengers on the instance can be served.*

Table IV: Comparison of results attained by the proposed DA vs BKS [3][13][14] on DARP instances [1][13].

## V. Conclusion

Many algorithms exist in the literature to obtain a solution for the DARP with minimum travel cost. However, none of them has considered the depot optimization problem which is crucial in today's context of mobility-on-demand transportation systems. Therefore, we have proposed a deterministic annealing (DA) meta-heuristic to optimize depot locations for the dial-a-ride problem. In this algorithm, a roulette wheel operator is introduced for local search, and a k-means clustering-based construction heuristic is designed to construct an initial set of depot locations. The proposed DA algorithm is validated on various standard DARP benchmark instances from the literature. For all tested instances, the proposed algorithm attains solutions with travel cost better than that of the best-known solutions. The results demonstrate that the depot locations clearly impact the demand responsive transportation system in terms of travel cost.

Some possible directions for future work include development of exact methods to optimally solve depot optimization problem and apply our DA algorithm in other contexts of DARP such as emergency vehicle placement, fire and rescue operation. Besides, our method can be directly extended to design a centralized fleet dispatcher for on-demand mobility that, determines the choice of vehicles from those running on the roads to be dispatched and develops a route plan for the selected vehicles for serving all the passengers, both with an objective to minimize travel cost. While optimizing vehicle dispatch in dial-a-ride service, the travel cost is expected to considerably decrease a lot more than that of the earlier optimal routes as evident from our findings. As a result, this approach not only has the potential to save fuel consumption of vehicles that would otherwise be running on the roads but also encourages more people to utilize shared mobility systems.

## Acknowledgment

The research was partially supported by the ST Engineering-NTU Robotics Corporate Laboratory through the National Research Foundation (NRF) corporate lab@university scheme.

## References

[1] J.-F. Cordeau, M. Gendreau, and G. Laporte, "A tabu search heuristic for periodic and multi-depot vehicle routing problems," *Networks: An International Journal*, vol. 30, no. 2, pp. 105–119, 1997.

[2] J.-F. Cordeau, "A branch-and-cut algorithm for the dial-a-ride problem," *Operations Research*, vol. 54, no. 3, pp. 573–586, 2006.

[3] S. N. Parragh, K. F. Doerner, and R. F. Hartl, "Variable neighborhood search for the dial-a-ride problem," *Computers & Operations Research*, vol. 37, no. 6, pp. 1129–1138, 2010.

[4] 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*, vol. 67, pp. 166–186, 2014.

[5] M. A. Masmoudi, K. Braekers, M. Masmoudi, and A. Dammak, "A hybrid genetic algorithm for the heterogeneous dial-a-ride problem," *Computers & Operations Research*, vol. 81, pp. 1–13, 2017.

[6] M. Chassaing, C. Duhamel, and P. Lacomme, "An els-based approach with dynamic probabilities management in local search for the dial-a-ride problem," *Engineering Applications of Artificial Intelligence*, vol. 48, pp. 119–133, 2016.

[7] S. Ho, S. C. Nagavarapu, R. R. Pandi, and J. Dauwels, "Improved tabu search heuristics for static dial-a-ride problem: Faster and better convergence," *arXiv preprint arXiv:1801.09547*, 2018.

[8] "Electric vehicles - the future without internal combustion," https://disruptionhub.com/ev-olution-almost/.

[9] "8 reasons why electric vehicles are the future of transportation," https://homesteading.com/electric-vehicles-future-transportation/.

[10] Y.-M. Shen and R.-M. Chen, "Optimal multi-depot location decision using particle swarm optimization," *Advances in Mechanical Engineering*, vol. 9, no. 8, p. 1687814017717663, 2017.

[11] P. J. Van Laarhoven and E. H. Aarts, "Simulated annealing," in *Simulated annealing: Theory and applications*. Springer, 1987, pp. 7–15.

[12] S. G. Ho, R. R. Pandi, S. C. Nagavarapu, and J. Dauwels, "Multi-atomic annealing heuristic for the dial-a-ride problem," in *Proceedings of the 12th IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*, 2018, pp. 272–277.

[13] S. Ropke, J.-F. Cordeau, and G. Laporte, "Models and branch-and-cut algorithms for pickup and delivery problems with time windows," *Networks*, vol. 49, no. 4, pp. 258–272, 2007.

[14] S. N. Parragh and V. Schmid, "Hybrid column generation and large neighborhood search for the dial-a-ride problem," *Computers & Operations Research*, vol. 40, no. 1, pp. 490–497, 2013.