Multi-atomic Annealing Heuristic for the Dial-a-ride Problem

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

***Abstract*—Dial-a-ride problem (DARP) deals with the transportation of users between pickup and drop-off locations associated with specified time windows. This paper proposes a novel algorithm called multi-atomic annealing (MATA) to solve the dial-a-ride problem. Two new local search operators (*burn* and *reform*), a new construction heuristic and two request sequencing mechanisms (*Sorted_List* and *Random_List*) are developed. Computational experiments conducted on various standard DARP benchmark instances prove that MATA is an expeditious meta-heuristic in contrast to other existing methods. In all experiments, MATA demonstrates the capability to obtain high quality solutions, faster convergence, and quicker attainment of a first feasible solution. It is observed that MATA attains a first feasible solution 29.8 to 65.1% faster, and obtains a final solution that is 3.9 to 5.2% better, when compared to other algorithms within 60 sec.**

***Index Terms*—Dial-a-ride problem, multi-atomic annealing, meta-heuristic, convergence.**

I. Introduction

Dial-a-ride problem belongs to a specific class of vehicle routing problem [1], in which passengers request for transportation between specified origin and destination. Each passenger usually defines time window for either pickup or drop-off.

DARP has been studied extensively since it was introduced by Wilson [2] in 1971; many attempts have been made to develop solution methodologies for DARP. Due to the high complexity of DARP, solvers generally require a long time to converge and obtain high quality solution, causing them impractical in real life, especially in a dynamic environment where changes happen frequently (i.e., requests, vehicles and traffic conditions). Exact methods such as branch-and-cut [3] have been developed, which require a long execution time especially for large-scale problems. Consequently, many heuristics and meta-heuristics have emerged to overcome this issue. In past decade, various techniques such as simulated annealing [4] and tabu search [5] [6] have been proposed to achieve near-optimal solutions in a shorter time.

An evolutionary local search approach [7] has been proposed, in which a dynamic probabilities management mechanism is used in the local search to improve convergence. A competitive variable neighborhood search [8] addresses the DARP using three classes of neighborhoods. Feasibility testing of dial-a-ride problem under maximum waiting time and maximum ride time has been investigated in [9].

Baugh et al. [4] were the first to employ simulated annealing to solve DARP. In 2003, Cordeau and Laporte [5] redefined the DARP formulation based on the real-life scenarios, and provided 20 benchmark instances. This formulation has been widely studied by the research community, and considered as the standard DARP, as detailed in [10].

In this paper, we propose a multi-atomic annealing heuristic for DARP. The major contributions of the paper are summarized as follows:

- A multi-atomic annealing (MATA) heuristic for DARP is proposed, in which a new construction heuristic is used to generate initial solution.
- Two new local search operators, *burn* and *reform*, are introduced to quickly explore the search space.
- The proposed MATA algorithm is tested on various standard DARP benchmark instances [5] and analysed for convergence.

The remainder of the paper is organized as follows: In Section II, we briefly discuss the standard DARP formulation. In Section III, we present the proposed multi-atomic annealing heuristic to solve DARP. In Section IV, we discuss the simulation results for the proposed algorithm. In Section V, we offer concluding remarks and ideas for future research.

II. Problem Formulation

The primary objective of dial-a-ride problem is to minimize the overall transportation cost besides providing superior passenger comfort and safety. DARP is mathematically formulated as an optimization problem subjected to several constraints. The DARP formulation addressed in this paper was proposed by Cordeau and Laporte [5]. A homogeneous set of customers and fleet are considered. It is assumed that the travel times between vertices are known.

In DARP, n customer requests are served using m vehicles, where each request i consists of time window either for departure or arrival vertex. Let $\mathbb{S} = \{x_1, x_2, \cdots\}$ denotes the search space, where $x_1, x_2, \cdots$ represent unique solutions. Every route for a vehicle k starts and ends at the depot; the departure vertex v_i and arrival vertex v_{i+n} must belong to the same route; the arrival vertex v_{i+n} is visited after departure vertex v_i. In addition, several other constraints need to be satisfied: (i) the load of vehicle k cannot exceed the vehicle capacity Q_k at any time; (ii) the total route duration of a vehicle k cannot exceed predefined duration bound T_k; (iii)

978-1-5386-4522-2/18/$31.00

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 $q(x)$ occurs when the number of passenger in a vehicle k exceeds its load limit Q_k; duration constraint violation $d(x)$ happens when a vehicle k exceeds its duration limit T_k; time window constraint violation $w(x)$ appears when time at the start of service B_i is later than l_i, the upper bound of time window; ride time constraint violation $t(x)$ occurs when a passenger is transported longer than ride time bound L. The constraints are defined as follows:

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

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

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

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

In the next section, we present the proposed multi-atomic annealing algorithm to solve DARP.

III. Multi-atomic Annealing (MATA)

Multi-atomic annealing (MATA) is inspired by the physical annealing process of solids, in which a solid is heated, and allowed to cool very slowly until it achieves a stable crystal lattice configuration with minimum lattice energy state. If the cooling schedule is sufficiently slow, the final configuration results in a solid with superior structural integrity.

The key algorithmic feature of MATA is that it provides means to escape local optima by melting a part of solid and allow the melted region to reform itself in a more stable configuration. At high temperature, a huge area of solid is melted, so more atoms are separated from the solid; at low temperature, a tiny area of solid is melted, so less atoms are separated from the solid. After melting, the structure is allowed to cool and reform into a more stable configuration with superior connecting bonds.

In literature, simulated annealing is another algorithm that mimics the annealing process. However, the concept of temperature defined in MATA is different from that of simulated annealing. In simulated annealing, the temperature is proportional to the energy of an atom involved in the diffusion process [11]. Whereas in MATA, temperature is proportional to the number of atoms separated and rearranged within the solid to improve the structure integrity.

It is evident that the temperature plays a vital role in determining the changes in structure. At high temperature, major changes to the solution are performed, allowing a wider exploration of search space; whereas at low temperature, minor changes to the solution are performed, allowing a deeper exploitation of the search space. Thus, a high initial temperature is preferred, so that a wider area of search space can be explored before a small area of search space can be exploited.

The notion of requests and solution quality in DARP is analogous to atoms and structural integrity of metals in metallurgy. To the best of our knowledge, this algorithm is proposed for the first time. The major challenge in this algorithm is to design a suitable cooling schedule, i.e. exploration and exploitation strategies, to obtain high quality solutions rapidly.

A. Algorithm

Multi-atomic annealing uses a construction heuristic to form an initial solution. The algorithm operates within a specified search space $\mathbb{S}$, where all solutions must be feasible without the need to be complete. A solution x is in feasible space $\mathbb{F}$, when all constraints ($q(x), d(x), w(x)$, and $t(x)$) are not violated; a solution x is in complete space $\mathbb{C}$, when all the requests are served. Search space, feasible space and complete space are defined as follows:

$$\mathbb{F} = \{x : q(x) = d(x) = w(x) = t(x) = 0\} \quad (5)$$

$$\mathbb{C} = \{x : r(x) = n\} \quad (6)$$

$$\mathbb{S} = \{x : x \in \mathbb{F}\}, \quad (7)$$

where $r(x)$ is the number of requests served in a solution. A solution x has a cost $f(x)$ which is equivalent to the total travel time of all vehicles. Solutions are evaluated using the three-level neighborhood evaluation method [12], which minimizes all the constraint violations while performing scheduling.

Algorithm 1 Multi-atomic Annealing (MATA)

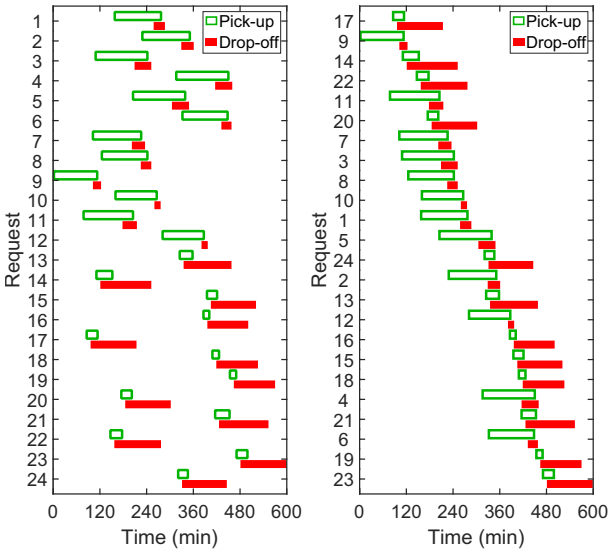
Input: T_{min}: minimum temperature; T_{max}: maximum temperature; λ_T: temperature drop rate; δ_{max}: no improvement limit

Output: x_{best}: best solution found

$T \leftarrow T_{\text{max}}$, $\delta \leftarrow 0$
$x_{\text{best}} \leftarrow \varnothing$ where $f(x_{\text{best}}) = \inf$.
generate *Sorted_List*.
$x \leftarrow$ *construct*(*Sorted_List*).
repeat
  $x \leftarrow$ *burn*(x, T,*Sorted_List*).
  $x \leftarrow$ *reform*(x).
  if $x \in (\mathbb{F} \cap \mathbb{C})$ **and** $f(x) < f(x_{\text{best}})$ **then**
    $x_{\text{best}} \leftarrow x$.
  else
    $\delta \leftarrow \delta + 1$.
  end if
  if $\delta > \delta_{\text{max}}$ **then**
    $x \leftarrow x_{\text{best}}$, $\delta \leftarrow 0$.
  end if
  $T \leftarrow T \times (1 - \lambda_T)$.
  if $T < T_{\text{min}}$ **then**
    $T \leftarrow$ random number in [T_{min}, T_{max}].
  end if
until termination criterion is fulfilled
return x_{best}.

Each iteration of the algorithm consists of three parts: In the first part, two local search operators, *burn* and *reform*, are used to alter the current solution to x_B and x_R respectively. In the second part, the current solution is compared against the best known solution x_{best}. A solution x is accepted as the best known solution, if and only if it is both feasible and complete ($x \in \mathbb{F} \cap \mathbb{C}$), besides having the least cost ($f(x) < f(x_{\text{best}})$). In the third part, the current temperature T is updated with respect to the temperature drop rate λ_T, and a restart mechanism may be triggered. The optimization process is restarted when no better solution is found after δ_{max} (no improvement limit) iterations. With the assumption that some better solutions are located near the current best known solution (x_{best}), the current solution x is reset to x_{best} to facilitate efficient exploitation of search space around the best known solution.

The algorithm is repeated until the termination criterion is fulfilled. In this paper, we use run time as the termination criterion. Algorithm 1 presents the structure of the multi-atomic annealing.



(a) Unsorted requests. (b) Sorted requests.

Fig. 1. Request sequencing mechanism for R1a instance [5].

B. Request Sequencing

Two different request sequencing mechanisms are used: random and sorted. The random sequence labeled as *Random_List*, is constructed by arranging all requests in random order. The sorted sequence labeled as *Sorted_List*, is constructed in two steps. In the first step, time window adjustment [12] is performed. In the second step, requests are sorted in an ascending order according to the value of $l_i + e_{i+n}$, where l_i is the upper bound of the time window for the departure vertex and e_{i+n} is the lower bound of the time window for the arrival vertex. Figs. 1a and 1b illustrate the request sequencing mechanism.

C. Construction Heuristic

Construction heuristic is used to form an initial solution by inserting all requests sequentially based on the *Sorted_List* (Section III-B). *Sorted_List* consists of requests arranged as per the time window, in which requests with time window approaching faster are kept ahead of others. Construction heuristic fetches requests from *Sorted_List* and inserts them into the solution at their best positions in a sequential order. During each insertion, a request is inserted into a route at a position with the lowest cost; if no feasible insertion is found, the request is not inserted. Two-step insertion, also known as neighborhood reduction [5], is used throughout this paper. Although construction heuristic does not guarantee a complete initial solution (i.e. all requests are served), *Sorted_List* ensures that the constructed initial solution serves more requests before the optimization process begins. Algorithm 2 presents the construction heuristic.

Algorithm 2 Construction Heuristic: *construct*()

Input: *Sorted_List*
Output: x_0: initial solution

```
x_0 ← ∅.
for all request i in Sorted_List do
    insert request i into x_0 at the best position.
end for
return x_0.
```

D. Local Search Operators

Two new local search operators, namely *burn* and *reform* are proposed in this algorithm. During *burn* process, a band of requests is removed from the *Sorted_List*. As detailed in Section III, the number of requests to be removed, R, is proportional to the current temperature T, where $R \in [1, T]$ is a random integer with equal chance of being selected during each execution of *burn* operation. Algorithm 3 describes the *burn* operation in detail.

During the *reform* process, all requests are inserted in a random sequence *Random_List*, which results in an exhaustive search for all possible combinations of request insertions. If a

Algorithm 3 Burn Operator: *burn*()

Input: x: current solution; T: temperature; *Sorted_List*
Output: x_B: burned solution

```
x_B ← x.
R ← random integer in [1, T].
i_start ← random integer in [1, n].
i_end ← min(n, i_start + R)].
for request i ← i_start to i_end in Sorted_List do
    remove request i from x_B.
end for
return x_B.
```

request already exists in the solution, the request is removed before it is reinserted. During each insertion, a request is inserted into a route at a position with the lowest cost; if no feasible insertion is found, the request is not inserted. The process of solution reformation (*reform*) is detailed in Algorithm 4.

Algorithm 4 Reform Operator: *reform*()

Input: x: current solution
Output: x_R: reformed solution

```
x_R ← x.
generate Random_List.
for all request i in Random_List do
    if request i is served in x_R then
        remove request i from x_R.
    end if
    insert request i into x_R at the best position.
end for
return x_R.
```

E. Restart Mechanism

The optimization process is restarted when no better solution is found after k_{max} iterations. With the assumption that some better solutions are located near the current best known solution, the current solution is reset such that $x \leftarrow x_{best}$ to encourage more exploitation of search space near the best known solution.

We have tested the proposed MATA algorithm against other methods in the literature; the simulation results are presented in the next section.

IV. Simulation Results

The proposed multi-atomic annealing heuristic is implemented in C++. Simulations have been carried out on a computer running 2.1 GHz Intel Xeon E5-2620 v4 processor with 128 GB RAM. We consider the DARP benchmark instances R1a, R3a, R6a, R8a [5] to carry out the experiments. In these instances, the number of requests are 24, 72, 144, and 72, respectively; the fleet size is 3, 7, 13, and 6, respectively. The capacity of each vehicle is 6, the maximum ride time of a passenger is 90 min, and the maximum route duration is 480 min. We set the run time of 60 sec as the algorithm termination criterion. The maximum and minimum temperatures are set to $\frac{n}{2}$ and $\frac{m}{2}$ respectively, where n is the number of requests and m is the number of vehicles. The parameters δ_{max} and λ_T are set to 30 and 0.01 respectively.

Figs. 2 to 5 illustrate the convergence of the proposed MATA when compared to tabu search (TS) [5] and improved tabu search (ITS) [12]. The lines with triangle, circle and square markers in the plots correspond to the MATA, TS [5] and ITS [12] algorithms respectively. The x-axis of the plots represents the simulation run time, and the y-axis corresponds to the gap (%) as given by:

$$\text{Gap (\%)} = \frac{\text{cost} - BKS}{BKS} \times 100, \tag{8}$$

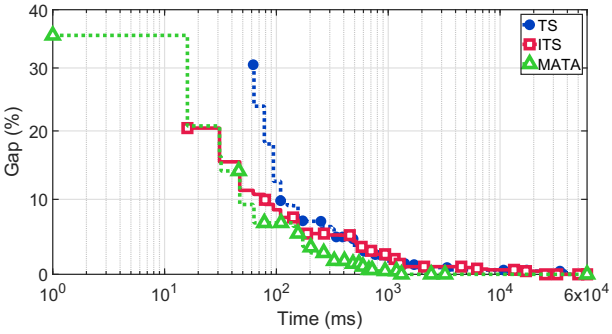


Fig. 2. Comparison of MATA with other methods for R1a instance [5].

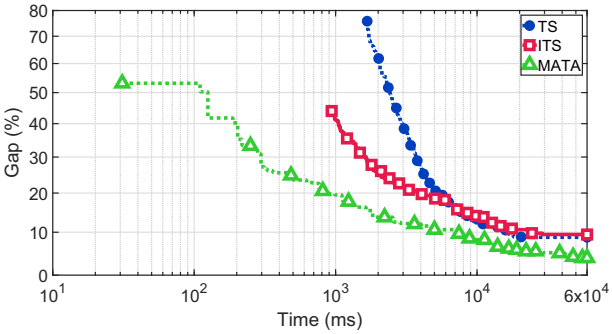


Fig. 3. Comparison of MATA with other methods for R3a instance [5].

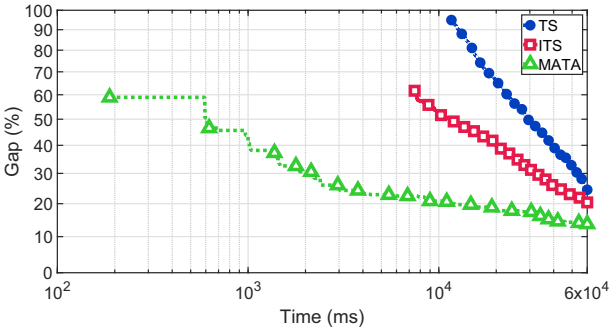


Fig. 4. Comparison of MATA with other methods for R6a instance [5].

where *BKS* denotes the travel cost of the best known solution for a given instance. We plot the median of the gap recorded from five independent simulations ran for a duration of one minute. It is evident from these plots that the proposed MATA outperforms TS [5] and ITS [12] algorithms in term of convergence.

In Tables I and II, we compare the median of the travel cost at various time instances {1, 2, 5, 15, 30, 60} sec for MATA, TS [5] and ITS [12] algorithms. From these tables, it can be seen that the MATA ensures better solution quality when compared to the other algorithms. Based on *BKS* listed in the tables, we can state that MATA attains near optimal solution in a shorter time for all the instances tested.

Test Instance	*BKS* [13]	Tabu Search (TS) [5]			Improved Tabu Search (ITS) [12]			Multi-atomic Annealing (MATA)		
		1 sec	2 sec	5 sec	1 sec	2 sec	5 sec	1 sec	2 sec	5 sec
R1a	190.02	193.26	191.66	191.11	193.77	191.88	191.66	190.84	190.02	190.02
R3a	532.00	-	863.26	644.51	750.64	670.69	632.84	635.77	605.06	588.60
R6a	785.26	-	-	-	-	-	-	1105.87	1025.38	965.20
R8a	487.84	-	-	614.78	-	642.78	574.11	617.85	591.54	579.57

Note: '-' corresponds to the scenarios where there is no solution obtained within the mentioned time instance.

TABLE I
COMPARISON OF THE TRAVEL COST AT VARIOUS TIME INSTANCES (1, 2 AND 5 SEC).

Test Instance	*BKS* [13]	Tabu Search (TS) [5]			Improved Tabu Search (ITS) [12]			Multi-atomic Annealing (MATA)		
		15 sec	30 sec	60 sec	15 sec	30 sec	60 sec	15 sec	30 sec	60 sec
R1a	190.02	191.05	190.79	190.02	191.11	190.02	190.02	190.02	190.02	190.02
R3a	532.00	592.15	578.66	578.66	593.06	582.25	582.25	565.98	559.20	552.76
R6a	785.26	1416.25	1171.76	977.61	1143.63	1034.10	945.55	934.98	922.40	893.00
R8a	487.84	548.96	544.45	544.45	549.66	536.74	536.74	541.51	528.25	519.52

TABLE II
COMPARISON OF THE TRAVEL COST AT VARIOUS TIME INSTANCES (15, 30 AND 60 SEC).

Test Instance	*BKS* [13]	Tabu Search (TS) [5]			Improved Tabu Search (ITS) [12]			Multi-atomic Annealing (MATA)		
		Cost	Gap (%)	Time (ms)	Cost	Gap (%)	Time (ms)	Cost	Gap (%)	Time (ms)
R1a	190.02	248.05	30.54	62	228.85	20.43	16	257.48	35.50	1
R3a	532.00	935.17	75.78	1672	765.95	43.98	938	814.69	53.14	31
R6a	785.26	1530.40	94.89	11642	1269.81	61.71	7469	1247.87	58.91	188
R8a	487.84	799.18	63.82	2562	731.00	49.84	1031	694.86	42.44	31

TABLE III
COMPARISON OF FIRST FEASIBLE SOLUTIONS.

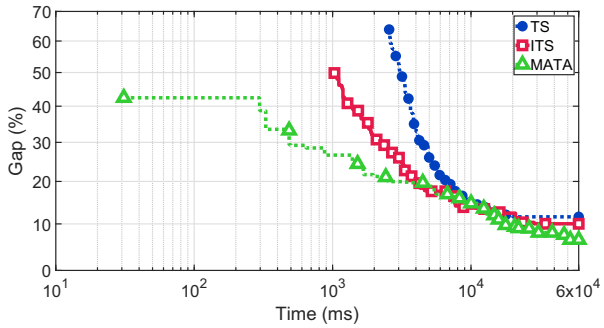


Fig. 5. Comparison of MATA with other methods for R8a instance [5].

Table III provides a comparison of the travel cost and execution time to attain first feasible solution for MATA, TS [5] and ITS [12] on various DARP test instances. From Table III, it is clear that MATA produces an initial feasible solution in less than a second, whereas the other algorithms require a considerable amount of time to achieve the same. As MATA obtains the first feasible solution much quicker, the quality of this solution might not be comparable to that of the other methods, which can be seen for instances R1a and R3a in Table III. On average, MATA attains a first feasible solution 65.1% faster than TS and 29.8% faster than ITS; in 60 sec, MATA obtains a final solution that is 5.2% better than TS and 3.9% better than ITS.

V. CONCLUSION

In this paper, we have proposed a novel multi-atomic annealing heuristic and applied it to solve the dial-a-ride problem. Many existing algorithms in the literature require long execution time to obtain a feasible solution. To overcome this limitation, we presented here two new local search operators (*burn* and *reform*), along with a construction heuristic, to provide a rapid first feasible solution. Furthermore, MATA shows faster convergence when compared to other algorithms. We conducted various numerical experiments on standard DARP benchmark instances to validate these claims. Some possible directions for future work are: (i) refinement of the construction heuristic through the exploration of different sequencing mechanisms to enhance the quality of the first feasible solution, (ii) development of new local search operators to widen the search horizon, and (iii) parallel implementation of the MATA algorithm to further reduce the computation time.

ACKNOWLEDGMENT

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

REFERENCES

[1] P. Toth and D. Vigo, *Vehicle routing: problems, methods, and applications*. SIAM, 2014.

[2] N. H. Wilson, J. Sussman, H.-K. Wong, and T. Higonnet, *Scheduling algorithms for a dial-a-ride system*. Massachusetts Institute of Technology. Urban Systems Laboratory, 1971.

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

[4] J. W. Baugh Jr, G. K. R. Kakivaya, and J. R. Stone, "Intractability of the dial-a-ride problem and a multiobjective solution using simulated annealing," *Engineering Optimization*, vol. 30, no. 2, pp. 91–123, 1998.

[5] J.-F. Cordeau and G. Laporte, "A tabu search heuristic for the static multi-vehicle dial-a-ride problem," *Transportation Research Part B: Methodological*, vol. 37, no. 6, pp. 579–594, 2003.

[6] D. Kirchler and R. W. Calvo, "A granular tabu search algorithm for the dial-a-ride problem," *Transportation Research Part B: Methodological*, vol. 56, pp. 120–135, 2013.

[7] 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.

[8] 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.

[9] M. Firat and G. J. Woeginger, "Analysis of the dial-a-ride problem of hunsaker and savelsbergh," *Operations Research Letters*, vol. 39, no. 1, pp. 32–35, 2011.

[10] J.-F. Cordeau and G. Laporte, "The dial-a-ride problem: models and algorithms," *Annals of Operations Research*, vol. 153, pp. 29–46, 2007, https://doi.org/10.1007/s10479-007-0170-8.

[11] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *science*, vol. 220, no. 4598, pp. 671–680, 1983.

[12] S. Ho, S. C. Nagavarapu, R. Ramasamy Pandi, and J. Dauwels, "Improved Tabu Search Heuristic for Static Dial-A-Ride Problem," *ArXiv e-prints, https://arxiv.org/abs/1801.09547*, 2018.

[13] 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.