Disruption Management for Dial-A-Ride Systems

Abstract—Mobility on demand has been gaining more attention from the research community as a way to offer smart and efficient transportation services to people. Despite the advancements in vehicular technologies, vehicle breakdown (VB) remains one of the major contributors to the disruption of fleet operations, which may inflict large recovery costs and damage the service provider's reputation. However, modeling such dynamic events for dial-a-ride (DAR) systems has not been addressed until now; this is the research gap we attempt to address in this article. The following are the contributions of our work: 1) the formulation of a disruptive DAR problem (DDARP) with VB (DDARP-VB), 2) a GPU-based adaptive large neighborhood search (G-ALNS) algorithm to solve the DDARP-VB, and 3) a fleet size minimization (FSM) strategy that leads to reduced operational costs under disruptions. From our simulations, the proposed G-ALNS algorithm performs up to 52 times faster than its CPU counterpart and produces better solutions

Ramesh Ramasamy Pandi, Sarat Chandra Nagavarapu, and Justin Dauwels
Are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. Email: ramesh006@e.ntu.edu.sg; sarat@ntu.edu.sg; jdauwels@ntu.edu.sg

Song Guang Ho
Is with ST Engineering Land Systems, Singapore. Email: hosongguang@gmail.com

Twinkle Tripathy
Is with the Faculty of Aerospace Engineering, Technion, Israel Institute of Technology, Haifa. Email: tripathy.twinkle@gmail.com

Digital Object Identifier 10.1109/MITS.2020.3014429
Date of current version: 2 September 2020

*Corresponding author

when compared to the existing GPU approach for DARPs. Overall, our FSM strategy leads to a 59% reduction in fleet idle time under normal operations and a 15% reduction in operational cost under disruption.

Smart urban mobility [1]–[3] is currently one of the prime areas of focus for most countries as a way to fulfill people's daily transportation needs, enhance travel comfort, and provide superior user convenience. Rapid advancements in the areas of autonomous vehicle technology, parallel computing, and optimization techniques for vehicle routing work as the driving forces to achieve this objective. Many day-to-day services, such as emergency vehicles, door-to-door people transport, goods delivery, and so on need efficient fleet-management solutions developed using these emerging technologies to provide satisfactory service to customers.

DARPs [4], a special class of vehicle-routing problems (VRPs), belong to the category of people transportation, which offers pick-up and drop-off services to a group of users between their preferred origin and destination locations while minimizing the total travel cost, subject to a set of constraints. Some of these constraints include vehicle load, route duration, passenger ride time, and the time windows (TWs) associated with pick-up and drop-off. Several optimization algorithms exist in the literature to provide optimal [5] or near-optimal [6] solutions to solve this problem. However, in many real-life scenarios, fleet operations are often hindered by the occurrence of many unanticipated stochastic events, such as VB, accidents, drastic weather conditions, and so on, which, in turn, diminish the solution quality. According to [7], with advances in communications technologies, DAR systems have evolved from being based on advance reservation to real-time vehicle-dispatch systems.

Thus, fast online algorithms are needed to be able to insert new trips into routes that are already ongoing. To achieve the necessary computational speeds, parallel computation is a promising research direction. The technological advancements are constantly creating new trends to operate DAR services. Particularly, GPU technology has evolved significantly in the past decade with the introduction of the Compute Unified Device Architecture (CUDA) programming model. Much progress has been made in computing technologies over the years with the invention of high-performance parallel-computing hardware and its usage in nongraphical, computationally intensive, general-purpose applications.

Besides designing optimal routes, another important problem in the fleet-management system is the minimum fleet problem, which, in general, focuses on determining the minimum number of vehicles required to provide an uninterrupted and high-quality transportation service to users [8]. According to a recent study on FSM for taxi services, in New York City alone, taxis travel around the city without serving any passengers for about 40% of the time [9]. With the increased need for transportation, several other issues, such as traffic congestion, carbon emissions per vehicle, passenger ride times, and so on, also increase significantly. Furthermore, the fleet depreciation loss, maintenance costs, supporting infrastructure costs, and so on pose an additional burden to the service providers. Thus, there is a great need for FSM in DAR systems.

Regardless of how technology advances, VBs lead to major disruptions of transportation services across the globe. In the United States alone, a massive 32 million car breakdowns were recorded [10] in 2015, and the numbers are rising every year. In the context of DAR systems, breakdowns result in inflated operational costs and damaged reputations for the service providers (operators of taxis, private buses, and so on). Therefore, the development of efficient fleet-management strategies to tackle VB is vital to providing uninterrupted transportation service to passengers.

Although the problem of solving vehicle routing with breakdown is very important in many real-world applications, further research needs to be carried out to address this problem in greater detail. However, only a handful of methods are available in the literature to solve this problem. A tabu search algorithm [11] for disruption management is proposed to generate new route plans for the delivery problem. Optimization models based on networks for complex aircraft rerouting and crew scheduling are detailed in [12] and [13], and [14] discusses different formulations and existing methods in the literature to address disruption management in road freight transport systems. A savings-based heuristic [8] is proposed to evaluate the reliability of vehicle-routing schemes in vaccine delivery with a VB. A disruption-management mathematical model [9] and a Lagrange heuristic are developed to address the urgency VRP. An improved genetic algorithm [15] was proposed to address the VRP with breakdowns.

Our Contributions

- According to Eglese and Zambirinis (2018) [14]—a survey on disruptive VRPs—there is no existing model for the DARP-VB. The most relevant studies on VB in a VRP context can be found in Li et al. (2009) [17] and Mu et al. (2011) [11]. In our article, we extend the formulation of the VRP-VB by Mu (2011) [11] and the DARP by Cordeau and Laporte (2003) [4] to model the DDARP-VB with new additional realistic assumptions, such as immediate fleet rerouting after VB, the decision to serve all customers to avoid damage to service providers' reputations, the design of a realistic operational cost function, and dynamic updates in travel times between the newly generated nodes, that is, the location of vehicles during VB and all of the other nodes in the network.

- According to Psaraftis et al. (2016) [18]—a survey on dynamic VRP—the future development of parallel algorithms may help further reduce the computational time, especially for dynamic problems. In the literature, the DARP with traffic or demand uncertainty has been addressed. The common methodology is to forecast the demand [19] or traffic uncertainty [20] to perform anticipative routing. However, in the case of VB, it is not possible to rely on prediction outputs. Therefore, there is a great need to develop a technique for a recourse action as soon as possible in the event of VB. At this point, ALNS is one of the outstanding algorithms for the DARP. It was developed by Ropke and Pisinger (2006) [26] for VRP with TWs (VRP-TW). In our article, we extend this basic ALNS approach with GPU acceleration to address the real-time dynamic rescheduling for disruption in a DAR context. Our method is a novel approach since we do not find such a G-ALNS in the literature for any extension of the VRP-TW, including the DARP. We are able to better report results, both in terms of computational time and solution quality, for our GPU approach when compared to the CPU version of ALNS and a recent GPU-accelerated tabu search method.
- According to Ho et al. (2018) [7]—a survey of DARPs—"As every transport operator knows, things never go according to plan. For DAR systems, as for other transport systems, disruption management is essential." New modeling techniques and frameworks are required for disruption management. In our article, we attempt to address this research gap by modeling the DDARP-VB problem and presenting a G-ALNS solution algorithm. Our fleet-management strategy, which is based on minimizing the fleet size for normal routing operations and addressing the disruption using a backup fleet in combination with online optimization, is novel since we did not find such an approach in the literature. Our results suggest that the proposed strategy attains reduced operational costs and fleet idle time in both the normal and disruptive scenarios. We also investigate the influence of breakdown and the effect of the proposed approach on passenger experiences. We have discovered that the passenger experience in terms of user ride improves both before and after the breakdown.

The development of efficient fleet-management strategies to tackle VB is vital to providing uninterrupted transportation service to passengers.

Problem Formulation

The DARP addresses the transportation of people between specified pick-up and drop-off locations. The DARP is solved as a combinatorial optimization problem to minimize the total travel time while satisfying all of the constraints related to passenger convenience. The formulation of the DARP in [4] is often regarded as the standard version. In the standard DARP, a homogeneous set of requests is considered with a fixed number of vehicles located at a single depot. All vehicles are considered to have constant speed throughout the network, and the Euclidean distance formula gives the distance between any two nodes in the network. Since it is one of the most widely studied DARPs in the literature, our formulation of the DARP-VB is built upon it with new, additional realistic constraints related to the event of VB.

In the standard DARP, n customer requests are served by m vehicles. Each request i consists of a TW for either the departure or arrival vertex or both. The objective is to minimize the travel cost subject to several constraints. Let $X = \{x_1, x_2, \ldots\}$ denote the solution space. All solutions $x_i \in X$ need to satisfy three basic constraints: 1) every route for a vehicle k starts and ends at the depot, 2) the departure vertex v_i and arrival vertex v_{i+n} must belong to the same route, and 3) the arrival vertex v_{i+n} is always visited after the departure vertex v_i. Any solution that violates this basic set of constraints becomes infeasible.

Additionally, four major constraints need to be satisfied: load, duration, TW, and ride time. A load constraint violation $q(x)$ occurs when the number of passengers in a vehicle k exceeds its load limit Q_k; a duration constraint violation $d(x)$ happens when a vehicle k exceeds its duration limit T_k; a TW constraint violation $w(x)$ appears when the time constraint on pick-up and drop-off is violated; and a ride time constraint violation $t(x)$ occurs when a passenger is transported for a longer time than ride time limit L. B_i represents the service starting time at vertex i; L_i represents the ride time of request i. For a detailed formulation, readers are referred to [5]. These four constraints can be formulated mathematically as follows:

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

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

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

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

The DARP addresses the transportation of people between specified pick-up and drop-off locations.

A common DARP application arises in door-to-door transportation services for the elderly and disabled. In this context, users often populate two requests per day: an outbound request from home to a destination and an inbound request for the return trip. Outbound requests, e.g., from home to the hospital, have a tight TW on the destination. Inbound requests, e.g., from the hospital back home, have a tight TW on the origin [21]. As in the work of Cordeau and Laporte (2003) [4], a more realistic way to model the DARP is to let users impose a TW of a prespecified width on the arrival time of their outbound trip and, similarly, a window on the departure time of their inbound trip. This approach seems to be in line with the practice of several North American transporters [5]. Therefore, we follow a similar assumption in our work along these lines.

According to the previously published works in [6], [21], and [22], the version of the DARP proposed in [4] is regarded as the "standard DARP," since it is widely treated in the literature. According to [4], "The information regarding TW widths, vehicle capacity, route duration and maximum ride time was provided by the Montreal Transit Commission (MTC), Canada. The instances are generated according to realistic assumptions." It can be inferred from the literature [21]–[23] that several recently studied new DARP variants generally extend the DARP formulation of [4]. Since our article is the first attempt to investigate VB in a DAR context, we have considered a similar approach and extend the formulation of [4].

The parameters related to vehicle capacity and speed may be defined as vehicle specific. Similarly, vehicle-specific depots can be defined. Nevertheless, in our DDARP-VB formulation, we have considered such parameters and sets to be homogeneous among all vehicles. This assumption is reasonable in the deployment of autonomous electric buses, as such vehicles currently mount the same commercial batteries and do not significantly differ in terms of dimensions, capacity, and weight [24]. Furthermore, the instances from [4] in which we conduct experiments have considered similar vehicle capacities, which reflects the situation of a transporter using mini buses for the transportation of individuals or groups of individuals.

The most relevant studies for VB in the VRP context can be found in [17] and [11]. We formulate the DDARP-VB by extending the existing VRP-VB model [11] and standard DARP model [4]. Additional assumptions made for this formulation are as follows: 1) dynamic vehicle rerouting happens at the time of disruption, without requiring vehicles to first arrive at their respective destinations; 2) all solutions must be complete, i.e., all requests must be served irrespective of service disruptions; and (3) the operational cost is computed based on a realistic estimation of today's transportation scenario. We formulate the DDARP-VB as follows:

1) All accepted travel requests must be served either feasibly or infeasibly.
2) During the event of a breakdown, any vehicle from the fleet must be in one of the following states:
   - has not left the depot
   - is traveling with no passenger onboard
   - is traveling with passenger(s) onboard
   - is at a service location with no passenger onboard
   - is at a service location with passenger(s) onboard
   - has returned to the depot.
3) A passenger i, affected by a breakdown event, can be served by another vehicle. For example, when a passenger i is inside a vehicle k_{VB} that is involved in a breakdown event, then passenger i can be served by another vehicle k_{VR} while ensuring that all of the constraints are met. If passenger i is dropped off by a vehicle that is involved in a breakdown event, then the TW for the subsequent pick-up will be set between the breakdown time and the end of the planning horizon. Therefore, if that passenger request is inbound, then it is very likely to be served by a new vehicle without constraint violation. However, if it is an outbound request, the previously tighter schedule produced by the algorithm decreases the chance of feasibly serving this user. In the event of VB, it is important to ensure uninterrupted transportation service for all passengers with minimal disruption to passenger convenience.
4) Ride time limit L, which is very important in the DARP, needs to be satisfied irrespective of VB to guarantee feasibility. If a passenger i is inside a vehicle k_{VB} that has broken down, then passenger i is dropped off at the location of the breakdown. Furthermore, a new request will be dynamically generated with the altered constraints from the previous problem, and the new ride time constraint will be set as the difference of the original ride time limit L and the travel time L_i of passenger i during the time of the breakdown.
5) When a vehicle k is involved in a breakdown event, it cannot serve any more passengers for that particular day. Therefore, it rejects all scheduled requests to be visited in the future while dropping off the onboard passengers immediately at the location of breakdown.

Methodology

In this section, we introduce an adaptive algorithm to solve the DDARP-VB.

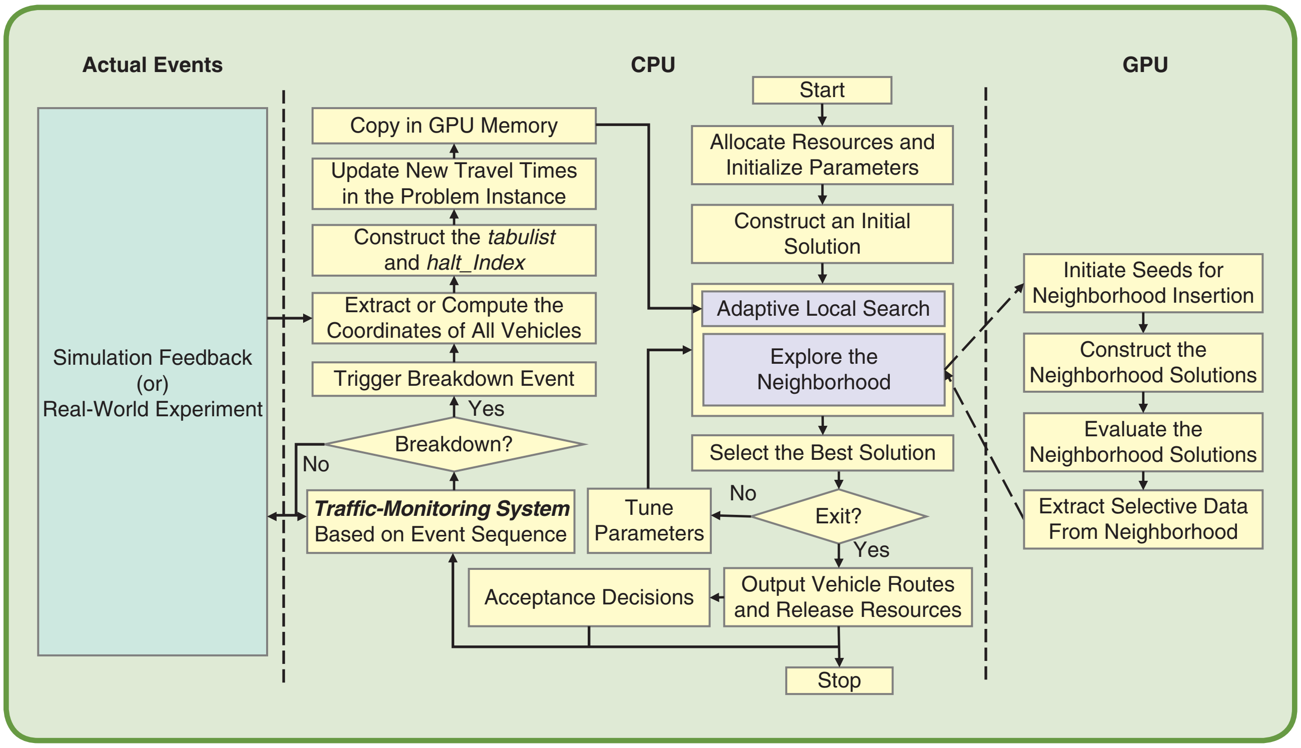
To perform a realistic evaluation of the solutions, we consider the actual vehicle operational cost structure observed in Singapore.

Overall Implementation of a DAR System

Figure 1 shows the flowchart of our real-time implementation of the DDARP-VB. As the service provider dispatches the fleet to serve passengers based on the solution provided by the algorithm, the *traffic-monitoring system* keeps track of the event sequence. In the case of unforeseen VB, the information corresponding to the broken down vehicle will be fed back to the optimizer, which includes vehicle coordinates (x_{VB}, y_{VB}), vehicle identity k_{VB}, and event occurrence time t_{VB}. In reflection, the algorithm dynamically optimizes routes and returns a new solution that has a minimal value of the objective function. All of the requests are served irrespective of feasibility to maintain the service provider's reputation.

However, in the real world, decisions on accepting passenger requests may often be based on several factors. After all of the passengers in the current demand pool have been served and all vehicles from the fleet have returned to the depot, the system can be shut down. To enhance the search efficiency of the optimization algorithm, we incorporate a tabu list that permanently forbids a set of requests that are already served at the time of disruption and *halt_Index* to check the feasibility of request insertions into the solution after the service disruption. Algorithm 1 shows the implementation details, as described:

1) Allocate GPU memory resources and transfer the DARP instance data from the CPU to GPU.
2) Generate an initial solution for the given instance using a construction heuristic. Here, a solution is encoded as a set of routes and schedules. Afterward, this initial solution is set as the current solution.
3) The route plan for the current solution is optimized using G-ALNS; then, the service provider can start to dispatch vehicles according to the best solution provided by the algorithm. Thereafter, on each iteration, the algorithm attempts to further minimize the objective function while monitoring any cases of VB based on the information provided in real time or simulation feedback.



Actual Events
CPU
GPU
Simulation Feedback (or) Real-World Experiment
Copy in GPU Memory
Update New Travel Times in the Problem Instance
Construct the tabulist and halt_Index
Extract or Compute the Coordinates of All Vehicles
Trigger Breakdown Event
Yes
Breakdown?
No
Traffic-Monitoring System Based on Event Sequence
Tune Parameters
Acceptance Decisions
Start
Allocate Resources and Initialize Parameters
Construct an Initial Solution
Adaptive Local Search
Explore the Neighborhood
Select the Best Solution
No
Exit?
Yes
Output Vehicle Routes and Release Resources
Stop
Initiate Seeds for Neighborhood Insertion
Construct the Neighborhood Solutions
Evaluate the Neighborhood Solutions
Extract Selective Data From Neighborhood

FIG 1 A flowchart of the real-time adaptive algorithm for the DAR system.

4) When the *traffic-monitoring system* detects a breakdown, the following steps are taken:
   - Extract the information of the broken vehicle k_{VB} and the event occurrence time t_{VB} from the system. For simulations, we randomly generate the values of k_{VB} and t_{VB}, provided that the vehicle k_{VB} has started service but not ended for the day.
   - The location of all vehicles during the event can be retrieved using GPS in real time or externally computed in simulation.
   - After the event, all of the previously served requests are updated in a list called T_{tabulist}. Such requests in the list will not be involved in the postbreakdown online route optimization process. Furthermore, we also record the *halt_Index* that contains previously visited nodes at the time of the breakdown to simplify the postoptimization algorithmic process.
   - If there are passengers inside the vehicle k_{VB}, then they are dropped off immediately at the breakdown location. Meanwhile, the travel times from the breakdown vertex to all other vertices are computed, and the constraints of affected users are adjusted.
   - The vehicle k_{VB} is discarded from service once the disruption occurs.
5) Once the stopping criterion is satisfied, the best solution that has the minimum travel cost is returned, and the GPU resources are released.

Algorithm 1. The overall implementation of the DAR system.

```
Input: DARP Instance
Output: BestSolution
 1: Allocate GPU resources
 2: Copy DARP instance from CPU to GPU
 3: Build an initial solution using construction heuristic
 4: Set current solution as the generated initial solution
 5: repeat
 6:   if check(trigger_breakdown) then
 7:     Choose a vehicle $k_{VB}$ to break down at time $t_{VB}$
 8:     Compute the coordinates for all vehicles $k \in m$
 9:     Construct the tabu list $T_{tabulist}$ and halt_Index
10:     if passengers $i_b \in n$ inside $k_{VB} > 0$ then
11:       Update coordinates of the passengers $i_b \in n$
12:       Update travel times between $i_b \in n$ and $i \in n$
13:       Disable the vehicle $k_{VB}$ after disruption
14:     end if
15:   else
16:     Copy the current solution from CPU to GPU
17:     while considering $T_{tabulist}$ and halt_Index do
18:       Perform removal operation
19:       Launch Neighborhood_Exploration kernel in GPU
20:     end while
21:     Copy selective data from GPU to CPU
22:     Select the next solution based on G-ALNS
23:     Adaptively adjust the ALNS operator weights
24:   end if
25: until stopping criterion is fulfilled
26: Release GPU resources
```

G-ALNS
The general idea of LNS by Shaw (1998) [25] is to repeatedly remove requests from the solution and to reinsert them at more profitable positions. Later, an adaptive weight adjustment scheme to LNS was introduced by Ropke and Pisinger (2006) [26] to develop ALNS for the VRP-TW. In this section, we extend this ALNS approach with GPU acceleration to solve the DDARP-VB. The idea of exploring neighborhoods using a GPU for the DARP was introduced in [27] and was applied for a tabu search algorithm. In this section, we develop a G-ALNS algorithm to solve the DDARP-VB.

Algorithm 2 details the CUDA kernel for neighborhood exploration in the GPU. Algorithm 3 details the proposed G-ALNS framework. The algorithm starts with an initial solution. For each iteration, a pair of destroy and repair ALNS operators are adaptively selected from a pool of operators using a roulette wheel mechanism [26]. First, a destroy operator is applied to remove some requests from the current solution. Then, a repair operator explores the neighborhood using a CUDA kernel (in Algorithm 2) based on [27] in the GPU and selects an appropriate solution in the CPU based on the simulated annealing acceptance criterion. The weights of the ALNS operators are adjusted after every segment according to predefined rules. For a detailed description of the ALNS heuristic, readers are referred to [26].

Neighborhood Search Operators
We implemented the following destroy and repair operators for ALNS:

Algorithm 2. Neighborhood_Exploration.

```
Input: DARP Instance; CurrentSolution
1: $t_{id} = threadIdx.x + blockIdx.x * blockDim.x$;
2: if $t_{id} < N_{size}$ then
3:   NeighborhoodSolution $x_{t_{id}} \leftarrow$ CurrentSolution
4:   For the request i, choose a new route and position
5:   Remove the request i from its current route in $N[t_{id}]$
6:   Insert the request i in the chosen route and position
7:   Evaluate the objective function value of solution $x_{t_{id}}$
8:   Extract selective data that will be copied back to CPU
9: end if
```

- Destroy operators
  - *Worst removal*: This operator iteratively removes requests that cause the biggest detour in the current solution.
  - *Random removal*: This operator randomly selects the requests to be removed among the planned requests.
  - *Related (or Shaw) removal*: This operator aims to remove requests that are alike so that their positions of service can somehow be interchanged.
- Repair operators
  - *Best insertion*: At each iteration, the best insertion cost is computed for each unplanned request, and the request with the lowest insertion cost is inserted at its best position. The heuristic stops when all requests are routed or none can be inserted anymore.
  - *Regret insertion*: This heuristic is based on the notion of regret, which is used by Potvin and Rousseau [28] for the VRP-TW and extended by Ropke and Pisinger [26]. It investigates the cost of inserting a request on the k best routes and finally inserts it on a route where the cost difference between inserting it into the best route and the $k-1$ best routes is largest. In our implementation, we consider regret-k heuristics with values of k between one and five.

FSM

We consider vehicle minimization as the first priority. Therefore, we use a simple two-stage algorithm. In the first stage, we start the algorithm with the number of vehicles specified for the instance. In the second stage, when the best solution does not improve anymore for a predefined number of iterations, we reduce the fleet size by one, greedily reinsert all of the requests from the disabled route into all other routes, and continue the optimization process. A similar approach can be found in the literature [26]. However, we have not seen such method applied to address disruptions for any variant of the VRP, including the DARP.

Implementation Details

To benchmark the efficiency of our GPU-based ALNS for rescheduling problems that require quick solutions, it is important to reimplement the same ALNS in the CPU, since Masson et al. (2014) [23] did not report the following: 1) solutions at various time instances < 30 s, 2) algorithmic parameters, 3) the CPU type, 4) the style of hash table implementation, and 5) the notion of critical vertex. In our approach, we use a separate chaining approach with hash function $f(x)$ mod 199 to avoid hash collisions, and the notion of critical vertex is not used, since it has the maximum potential to find the best solution during the search operation.

Algorithm 3. The G-ALNS framework.

```
Input: DARP Instance
Output: BestSolution
1:  BestSolution ← InitialSolution
2:  CurrentSolution ← InitialSolution
3:  while the termination condition is not satisfied do
4:     Select ALNS operators according to their past scores
5:     x ← CurrentSolution
6:     x ← Destroy(x)
7:     DataTransfer from CPU to GPU
8:     x ← Repair(x) ← Neighborhood_Exploration in GPU
9:     DataTransfer from GPU to CPU
10:    if x < BestSolution then
11:       BestSolution ← x
12:       CurrentSolution ← x
13:    else
14:       if AcceptanceCriterion(x, CurrentSolution) then
15:          CurrentSolution ← x
16:       end if
17:    end if
18: end while
19: return BestSolution
```

Table 1. The computational time and speed-up attained by ALNS on the GPU versus CPU for various instances.

Case	Nodes	m	CPU (s)	GPU (s)	Speed-up	Difficulty
R1a	48	3	5.61	1.15	4.89	5.61
R2a	96	5	18.83	1.5	12.54	18.83
R3a	144	7	42.42	2.11	20.11	42.42
R4a	192	9	80.24	2.86	28.09	80.24
R5a	240	11	119.3	3.6	33.1	119.3
R6a	288	13	185.43	4.24	43.77	185.43
R7a	72	4	10.81	1.39	7.77	10.81
R8a	144	6	45.99	2.24	20.53	45.99
R9a	216	8	98.19	3.44	28.53	98.19
R10a	288	10	243.9	4.66	52.37	243.9
VLS1	336	16	237.31	8.71	27.23	237.31
VLS2	480	22	513.6	16.41	31.3	513.6
VLS3	720	33	1,861.3	45.2	41.18	1,861.3
VLS4	1,008	45	2,171.5	83.3	26.07	2,171.5

The stopping criterion is 10^3 number of iterations.

Simulation experiments

Simulation Settings

The proposed G-ALNS algorithm for the DARP-VB is implemented in CUDA. The CPU version of the ALNS algorithm is implemented in C++. All simulations were carried out on a computer running a 2.1-GHz Intel Xeon E5-2620 v4 processor and Nvidia Tesla P100-PCI-E 16-GB GPU, interconnected with a Peripheral Component Interconnect express PCI-e bus. Visual Studio Integrated Development Environment 2015 with the CUDA 8.0 toolkit was used as the development platform.

The operational cost is one of the important parameters to assess in the fleet operations, which often refers to the overall cost incurred by the service provider and includes vehicle procurement cost, licensing fees, fuel, insurance,

Table 2. The R1a instance: results attained by the G-ALNS for the DDARP-VB.

Case	Breakdown Time (min)	Method	Before Breakdown n^*	Before Breakdown $\Phi(x)$ (km)	Before Breakdown $\alpha(x)$ (S\$)	After Breakdown n^*	After Breakdown $\Phi(x)$ (km)	After Breakdown $\alpha(x)$ (S\$)
Vehicle 1	Early (213.33)	NA	24	190.01	208	22	185.12	207
		CM	24	190.01	208	24	193.46	305
		G-ALNS	24	190.01	208	24	193.46	305
	Middle (286.77)	NA	24	190.01	208	24	190.01	208
		CM	24	190.01	208	24	190.01	208
		G-ALNS	24	190.01	208	24	190.01	208
	Late (360.20)	NA	24	190.01	208	24	190.01	208
		CM	24	190.01	208	24	190.01	208
		G-ALNS	24	190.01	208	24	190.01	208
Vehicle 2	Early (213.33)	NA	24	190.01	208	13	119.99	191
		CM	24	190.01	208	24	194.83	305
		G-ALNS	24	190.01	208	24	194.83	305
	Middle (286.76)	NA	24	190.01	208	16	129.17	194
		CM	24	190.01	208	24	192.5	304
		G-ALNS	24	190.01	208	24	192.5	304
	Late (360.20)	NA	24	190.01	208	18	144.86	197
		CM	24	190.01	208	24	200.35	306
		G-ALNS	24	190.01	208	24	200.35	306
Vehicle 3	Early (213.33)	NA	24	190.01	208	15	124.95	192
		CM	24	190.01	208	24	193.94	305
		G-ALNS	24	190.01	208	24	193.94	305
	Middle (286.76)	NA	24	190.01	208	19	143.99	197
		CM	24	190.01	208	23	194.4	305
		G-ALNS	24	190.01	208	24	201.03	306
	Late (360.20)	NA	24	190.01	208	18	151.18	198
		CM	24	190.01	208	24	195.66	305
		G-ALNS	24	190.01	208	24	195.66	305

The best results out of five random runs are reported. $\beta_{avg}(x) = 32.97\%$, and $\delta_{avg}(x) = 36.32\%$. CM: conventional method; NA: no recourse action taken; *: total number of served customers.

driver wages, maintenance, repairs, parking, toll taxes, and all other mileage-dependent depreciation costs. To perform a realistic evaluation of the solutions, we consider the actual vehicle operational cost structure [29] observed in Singapore.

The average operational costs for an owned and rented vehicle in Singapore [reported in Singapore dollars (S$)] are S$1,620/month and S$2,880/month, respectively, excluding fuel costs of S$0.24/km. The cleaning cost of a vehicle generally includes an average water wash cost of S$15 and an inside vacuuming cost of S$10 per use when considering a cost-effective cleaning agency based in Singapore. Since the DAR system involves passenger comfort, we set the cleaning cost to be a maximum value of S$25/day [30].

In Singapore, the parking cost for a vehicle is estimated to be S$0.04/min for a parking space provided by the Housing Board and Development Authority. The parking time in DAR service depends on the amount of time that the vehicle is parked in a customer location waiting to perform a pick-up or drop-off service [31]. For cars operating on Singapore roads, the average wear-and-tear cost is estimated to be around S$300/$10^4$ km of travel distance. Considering the upper bound, an average cost of S$0.03/km is incurred by the car owners for wear and tear.

The strategy for renting a vehicle in the case of a breakdown involves a full-day rental cost. The reported cost values are in Singapore dollars. The operational cost function is given by

$$\alpha(x) = (\nu + \Delta)\, m + (\gamma + \zeta)\, \Phi(x) + \delta \Gamma(x) + \mu \Upsilon(x), \quad (5)$$

where m is the total number of vehicles owned by the fleet operator, ν is the maintenance cost per vehicle per day, Δ is the cleaning cost per vehicle per day, $\Phi(x)$ is the total distance traveled by the vehicles in kilometers by both the driver's own and rental vehicles, ζ is the wear-and-tear cost per vehicle per kilometer, γ is the fuel cost per vehicle per kilometer, $\Gamma(x)$ is the number of additional vehicles added to the fleet after the breakdown, δ is the rental cost per vehicle per day, μ is the parking cost per vehicle per minute, and $\Upsilon(x)$ is the parking time of vehicles in minutes.

Table 3. The R1a instance: results attained by G-ALNS-FSM for the DDARP-VB.

			Before Breakdown			After Breakdown		
Case	Breakdown Time (min)	Method	n^*	$\Phi(x)$ (km)	$\alpha(x)$ (S$)	n^*	$\Phi(x)$ (km)	$\alpha(x)$ (S$)
Vehicle 1	Early (216.22)	NA	24	193.5	155	15	123.39	138
		CM	24	193.5	155	24	197.04	210
		G-ALNS-FSM	24	193.5	155	24	195.38	209
	Middle (315.48)	NA	24	193.5	155	19	147.48	198
		CM	24	193.5	155	23	197.88	210
		G-ALNS-FSM	24	193.5	155	24	197.69	209
	Late (414.75)	NA	24	193.5	155	20	168.77	148
		CM	24	193.5	155	24	197.87	210
		G-ALNS-FSM	24	193.5	155	24	197.87	210
Vehicle 2	Early (216.22)	NA	24	193.5	155	13	121.18	138
		CM	24	193.5	155	24	198.32	210
		G-ALNS-FSM	24	193.5	155	24	198.32	210
	Middle (315.48)	NA	24	193.5	155	18	138.4	141
		CM	24	193.5	155	23	195.99	209
		G-ALNS-FSM	24	193.5	155	24	200.56	210
	Late (414.75)	NA	24	193.5	155	19	154.28	145
		CM	24	193.5	155	23	197.87	209
		G-ALNS-FSM	24	193.5	155	24	204.89	211

The best results out of five random runs are reported. $\beta_{avg}(x) = 41.76\%$, $\delta_{avg}(x) = 35.38\%$, $\eta_{1(avg)}(x) = 25.48\%$, and $\eta_{2(avg)}(x) = 25.49\%$.

Table 4. The results for G-ALNS-FSM, G-ALNS, and CM on solving the DDARP-VB for five random runs with the stopping criterion as 30 s.

	Benchmark Solution				Solution After Breakdown			CM				G-ALNS						
Case	n^*	m	$\Phi(x)$ (km)	$\alpha(x)$ (S$)	n^*	$\Phi(x)$ (km)	$\alpha(x)$ (S$)	n^*	$\Phi(x)$ (km)	$\alpha(x)$ (S$)	$\beta(x)$	n^*	$\Phi(x)$ (km)	$\alpha(x)$ (S$)	$\beta(x)$ (S$)	$\delta(x)$ (S$)	η_1 (%)	η_2 (%)
R1a	24	3	190.02	297	17.4	145.71	282	24	193.93	394	44.56	24	193.93	394	44.56	32.62	—	—
R2a	48	5	301.34	505	41.8	268.24	492	47.8	305.25	602	14.62	48	301.97	601	15.1	18.98	—	—
R3a	72	7	532	721	62	512.58	711	71.4	582	829	15.32	72	577.12	830	16.34	15.07	—	—
R4a	96	9	570.25	897	86.8	535.22	887	95	575.05	995	9.55	96	605.65	1,004	10.94	11.82	—	—
R5a	120	11	626.93	1,071	111.6	591.8	1,060	118.6	629.58	1,168	6.27	120	665.83	1,181	7.53	10.3	—	—
R6a	144	13	785.26	1,280	126.6	1,012.86	1,337	138.8	1,075.51	1,452	9.68	144	868.62	1,403	13.89	9.62	—	—
R7a	36	4	291.71	412	28.4	229.25	392	35.4	297.1	510	24.71	36	307.99	512	26.86	24.06	—	—
R8a	72	6	487.84	619	60.2	481.07	614	66.2	530.67	725	10.01	72	505.22	720	20.1	16.37	—	—
R9a	108	8	658.31	818	101	617.67	807	107	666.11	917	5.94	108	684.97	922	6.93	12.65	—	—
R10a	144	10	851.82	1,049	118	1,036.8	1,092	131	1,101.29	1,210	11.02	144	1,022.69	1,185	22.03	13.02	—	—
												G-ALNS-FSM						
R1a	24	2	193.5	215	15.6	127.85	196	23.4	199.38	312	52.73	24	198.11	312	56.98	45.07	**27.61**	**20.81**
R2a	48	4	305.5	415	36.4	232.29	392	47.8	311.54	512	31.64	48	317.04	514	32.16	23.82	**17.82**	**14.48**
R3a	72	6	542.98	637	62.8	481.08	618	71.8	546.72	734	14.77	72	543.92	733	15.06	15.1	**11.65**	**11.69**
R4a	96	6	638.44	652	83.4	562.84	629	94.4	642.33	749	13.06	96	722.85	771	15.23	18.35	**27.31**	**23.21**
R5a	120	8	688.552	829	98.8	694.91	827	110	763.2	944	11.83	120	782.13	950	22.86	14.63	**22.6**	**19.56**
R6a	144	9	862.18	955	129.2	777.82	932	141.8	864.67	1,052	9.76	144	977.55	1,081	11.47	13.12	**25.39**	**22.95**
R7a	36	3	302.97	325	27.2	242.67	307	35	311.24	423	29.57	36	304.55	422	33.27	30.04	**21.12**	**17.58**
R8a	72	5	511.43	538	59.4	421	513	71.4	517.74	636	20.36	72	527.07	638	21.35	18.66	**13.09**	**11.39**
R9a	108	8	658.31	818	101	617.67	807	107	666.11	917	5.94	108	684.97	922	6.93	12.65	**0**	**0**
R10a	144	9	921.2	973	132	840.75	951	142.6	926.3	1,071	10	144	999.31	1,089	9.15	11.89	**7.24**	**8.1**
Improvement (%)																	**17.38**	**14.98**

A reduction in operational cost is achieved by the G-ALNS-FSM both before and after VB. ∗: total number of served customers.

Table 5. The influence of VB and effect of the proposed algorithm on passenger experience.

	Before Breakdown				After Breakdown			
	G-ALNS		G-ALNS-FSM		G-ALNS		G-ALNS-FSM	
Case	Total (min)	Average (min)	Total (min)	Average (min)	Total (min)	Average (min)	Total (min)	Average (min)
R1a	1,094.99	45.62	1,205.42	50.23	1,186.57	49.44	1,232.5	51.35
R2a	1,929.19	40.19	2,297.71	47.87	1,937.94	40.37	2,339.73	48.74
R3a	3,505.72	48.69	3,538.91	49.15	3,469.18	48.18	3,350.99	46.54
R4a	4,997.11	52.05	4,638.47	48.32	4,976.65	51.84	4,402.21	45.86
R5a	6,002.53	50.02	5,810.07	48.42	5,971.77	49.76	5,589.79	46.58
R6a	7,643.21	53.08	6,938.14	48.18	7,061.58	49.04	6,821.72	47.37
R7a	1,508.93	41.91	1,521.94	42.28	1,456.1	40.45	1,662.36	46.18
R8a	3,692.65	51.29	3,648.65	50.68	3,802.11	52.81	3,716.62	51.62
R9a	5,222.37	48.36	5,222.37	48.36	5,262.71	48.73	5,262.71	48.73
R10a	7,560.65	52.5	7,188.66	49.92	6,987.08	48.52	7,049.9	48.96
Average	4,315.74		**4,201.03**		4,211.17		**4,142.85**	
Improvement (%)			**2.66**				**1.62**	

A comparison of user ride times for G-ALNS versus G-ALNS-FSM for the DARP is shown. The reduction in user ride time is achieved by our G-ALNS-FSM.

Table 6. The details of the solution attained by G-ALNS-FSM on the tested instances.

Case	v	m	m^*	Dist (km)	T_{dur} (min)	T_{idle} (min)	T_{ride} (min)	T_{early} (min)	T_{late} (min)	Δm	Δm (%)	Detour (%)
R1a	48	3	2	193.5	794.09	120.58	1,205.42	33.15	1,891.02	**1**	**33.33**	1.83
R2a	96	5	4	305.5	1,674.77	409.27	2,297.71	220.12	3,205.71	**1**	**20**	1.38
R3a	144	7	6	542.98	2,391.4	408.41	3,538.91	137.54	5,408.47	**1**	**14.29**	2.06
R4a	192	9	6	638.44	2,690.66	132.22	4,638.47	10.06	7,067.35	**3**	**33.33**	11.96
R5a	240	11	8	688.52	3,360.65	272.13	5,810.07	89.83	9,145.21	**3**	**27.27**	9.83
R6a	288	13	9	862.18	5,175.94	290.28	6,938.14	60.37	9,730.66	**4**	**30.77**	9.8
R7a	72	4	3	302.97	1,483.75	144.65	1,521.94	84.09	2,828.67	**1**	**25**	3.86
R8a	144	6	5	511.43	2,323.25	124.03	3,648.65	44.8	5,253.71	**1**	**16.67**	4.84
R9a	216	8	8	658.31	3,031.9	213.59	5,222.37	33.95	8,277.11	0	0	0
R10a	288	10	9	921.2	4,542.06	343.02	7,188.66	54.64	10,666.18	**1**	**10**	8.15
Total				5,625.04	27,468.46	2,458.17	42,010.35	768.55	63,474.09			

G-ALNS-FSM attains significant fleet savings at a reasonable increase in detour distance when compared to the benchmark solutions [7] without FSM. Detour (%): the percent total extra distance traveled by the fleet when compared to the benchmark solutions [7] without FSM; Dist: the total travel distance of all vehicles; m: the number of available vehicles; m^*: the number of used vehicles; Δm: the number of vehicles reduced by our FSM strategy; Δm (%): the percent vehicle reduction; T_{dur}: the total travel duration of all vehicles; T_{early}: the total time for vehicle early arrival; T_{idle}: the total idle time of all vehicles; T_{late}: the total time for vehicle late arrival; T_{ride}: the total ride time of all users; v: the number of vertices. The values in bold represent the reduction in fleet usage.

We test the proposed methodology on the standard DARP benchmark instances from R1a to R10a [4]. In these instances, the fleet size ranges from three to 13, and the corresponding number of requests ranges from 24 to 144. Each vehicle has a capacity of six seats, with a maximum route duration of 480 min. The maximum user ride time and service time are 90 and 10 min, respectively. The TW length for either the pick-up or drop-off vertex is 15 min, and the other subsequent vertex is 1,400 min.

Additionally, we have generated a set of four very large instances of a realistic scale to test the computational efficiency of our proposed approach. We name the instances very large scale (*VLS*)*1–VLS4*. The four instances, generated by merging multiple standard DARP instances from the literature, are as follows: $\text{VLS1} = {}_{\{R6a, R1a\}}$, $\text{VLS2} = {}_{\{R6a, R4a\}}$, $\text{VLS3} = {}_{\{R6a, R4a, R5a\}}$, and $\text{VLS4} = {}_{\{R6a, R4a, R5a, R10a\}}$. The numbers of nodes are 336, 480, 720, and 1,008, and the numbers of vehicles are 16, 22, 33, 45, respectively. Each vehicle has a maximum route duration of 1,440 min, and the user ride time limit is set as 180 min. Other operational constraints are set similarly to the standard DARP instances.

To ensure a fair comparison, each experiment was conducted five times, and the average values are reported. Table 1 shows detailed solutions for the DDARP-VB with different breakdown times, such as early, middle, and late during the service. First, we compute the average-over-service starting time and ending time of all vehicles for the benchmark solution and then divide them into $m+1$ slices to find the early, middle, and late time stamps, respectively.

In Mu et al. (2011) [11], for each DRVP instance, a time constraint of 12 s has been applied. In Li et al. (2009) [17], for disruptive VRP, 300 iterations for the rerouting problem are applied for delivery service. Here, we

Table 7. Policy implications: the average savings with FSM by G-ALNS and G-TS [27] when compared to the benchmark solutions [7] without FSM for all of the tested DARP instances.

	Savings (%)	
Quantifier	G-TS-FSM	G-ALNS-FSM
Reduction in fleet size	+21.05	**+21.05**
Reduction in travel distance	–9.83	**–6.22**
Reduction in total vehicle route duration	**+10.11**	+3.47
Reduction in total vehicle idle time	+57.77	**+58.19**
Reduction in total passenger ride time	**+5.02**	+2.66
Reduction in total vehicle early arrival time	+69.34	**+72.85**
Reduction in total vehicle late arrival time	–4.21	**–1.44**
Operational cost	+16.16	**+40.06**

The values in bold represent the more desirable value of the quantifier obtained by G-TS-FSM and G-ALNS-FSM.

Table 8. The evolution of the objective function over time for our G-ALNS approach versus a recent G-TS method [27] for the DARP.

	5 s		15 s		30 s		60 s	
Case	G-TS	G-ALNS	G-TS	G-ALNS	G-TS	G-ALNS	G-TS	G-ALNS
R1a	190.02	**190.02**	190.02	**190.02**	190.02	**190.02**	190.02	**190.02**
R2a	313.45	**309.62**	307.81	**305.28**	306.55	**304.11**	306.31	**303.99**
R3a	578.28	**576.63**	**555.56**	563.33	555.56	**548.97**	553.7	**546.52**
R4a	681.33	**618.25**	649.62	**599.08**	618.2	**590.36**	618.2	**584.52**
R5a	822.03	**709.1**	761.84	**679.46**	705.29	**666.43**	680.53	**655.73**
R6a	Inf	**882.15**	953.7	**856.76**	953.7	**835.38**	953.7	**827.65**
R7a	302.5	**297.29**	300.29	**291.74**	299.44	**291.71**	298.22	**291.71**
R8a	539.8	**519.27**	510.63	**509.84**	510.63	**503.99**	505.5	**497.34**
R9a	Inf	**808.83**	Inf	**763.46**	Inf	**719.54**	Inf	**704.37**
R10a	Inf	**1,031.01**	Inf	**941.04**	Inf	**924.05**	Inf	**916.26**

Inf corresponds to no solution found. The values represent travel distance (in kilometers). Intermediate objective values are recorded on solving the DARP within the first 60 s for direct comparison. The values in bold represent the more desirable value of the objective function between G-TS and G-ALNS at every timestamp.

report the average speed-up out of five runs after 10^3 iterations in Table 1, the best solution out of five runs after 30 s in Tables 2 and 3, the average out of five runs after 30 s in Table 4, the best out of five runs after 120 s in Tables 5–8, and the median values over five runs in Table 9.

Results and Discussion

In Table 1, we show the computational time attained by ALNS in the CPU and GPU. The *computational speed-up* is defined as the ratio of the computational time in the CPU to the computational time in the GPU. The difficulty of each instance is observed to be the computational time of the ALNS in the CPU. In Figure 2 illustrates the significance of G-ALNS in terms of computational speed-up when compared to its CPU counterpart for various problems. In Figure 3 demonstrates the computational efficiency of our approach in very-large-scale instances.

In Tables 2 and 3 depict the results for the R1a instance attained before and after breakdown by methods such as the conventional method (CM), G-ALNS, and G-ALNS-FSM. Note that no recourse action taken (NA) represents the solution when no recourse action has been taken in the event of a VB. In G-ALNS-FSM, we attain the benchmark solution before breakdown by incorporating an FSM scheme, whereas we do not include such a scheme in G-ALNS.

In the CM, we rent a new vehicle and dispatch it to follow the preplanned route of the broken vehicle. In G-ALNS, we rent an extra vehicle, apart from performing online optimization of routes after disruption. In G-ALNS-FSM, we dispatch a new vehicle from the depot and dynamically optimize the routes after the disruption. For the solved instance, we have only two vehicles for the G-ALNS-FSM benchmark solution and three vehicles for the G-ALNS benchmark solution. In Table 2, the vehicles 1, 2, and 3 serve 4, 26, and 18 nodes, respectively. For solution details, readers can refer to the benchmark solution [6]. In Table 3, the vehicles 1 and 2 serve 22 and 26 nodes, respectively.

In Table 4, we perform five executions with randomly selected breakdown vehicles and times at which the vehicles break down. From the results, we can see that the number of total passengers feasibly served β and percent cost hike of the rescheduled solution δ, when compared to benchmark solution, are in the favor of G-ALNS-FSM versus G-ALNS. In Table 4, η_1 shows the percent savings in the operational cost of the benchmark solution attained by G-ALNS-FSM when compared to G-ALNS, and η_2 shows the percent savings in the operational cost of the postbreakdown solution attained by G-ALNS-FSM when compared to G-ALNS. The average reduction in operational cost attained by G-ALNS-FSM, when compared to G-ALNS, corresponds to 17.38% before the breakdown and 14.98% after the breakdown, respectively. Therefore, the usage of FSM is beneficial in reducing operational costs.

In Table 5, we discuss the influence of VB and effect of the proposed approach on passenger experiences. In passenger transit services, user stress levels

Table 9. The effect of the proposed algorithm on fleet idle time.

	Fleet Idle Time			
Case	Benchmark	After Breakdown	CM	G-ALNS
R1a	211.14	151.98	204.84	204.84
R2a	725.98	607.8	725.98	719.66
R3a	607.27	502.39	581.48	618.27
R4a	812.88	784.69	809.38	826.81
R5a	813.83	770.69	817.68	909.18
R6a	1,015.99	902.35	965.31	1,131.02
R7a	440.34	351.78	440.03	410.59
R8a	322.33	262.81	291.82	336.69
R9a	213.59	213.59	221.69	221.19
R10a	716.66	554.28	669.94	577.08
Average	588	510.24	572.82	595.53
				G-ALNS-FSM
R1a	120.58	75.91	116.68	112.77
R2a	409.27	331.54	408.06	401.46
R3a	408.41	349.67	398.07	406.73
R4a	132.22	72.16	132.22	151.38
R5a	272.13	192.68	247.85	272.58
R6a	290.28	275.38	299.67	244.37
R7a	144.65	104.81	145.23	171.46
R8a	124.03	105.32	124.91	127.95
R9a	213.59	213.59	221.69	221.19
R10a	343.02	324.7	349.39	309.89
Average	245.82	204.57	244.38	241.98
Improvement (%)	**58.19**			**59.37**

The improvement attained by G-ALNS-FSM versus G-ALNS for the DARP before and after the breakdown is shown.

From our simulations, although the objective is not to increase passenger comfort, we have noticed that the passenger experience is improved by a reduction in commute time.

are significantly influenced by the commute time [16]. Empirical studies show that journey times affect passengers' quality of life. According to a psychological study [16], "the passenger stress effects in transit services seemed to be mediated by the time of the trip, i.e., reduced commuting time could account for some of the significant stress impacts of experimental alteration of the one commuting route." From our simulations, although the objective is not to increase passenger comfort, we have noticed that the passenger experience is improved by a reduction in commute time. When compared to G-ALNS, G-ALNS-FSM accounts for reductions in commute time by 2.66% before the breakdown and 1.62% after the breakdown, respectively.

In Table 6, we show the detailed solution attained by the G-ALNS-FSM method on DARP instances. In Table 7, we analyze the policy implication of FSM if disruption does not occur. We compare the results of our G-ALNS-FSM and an existing GPU-based tabu search (G-TS) algorithm [27] with FSM (G-TS-FSM). Finally, the reductions in quantifiers are computed with respect to the benchmark solutions [7] without FSM. We are able to reduce the fleet size by 21.05% with a reasonable additional detour distance of 6.22%. Moreover, the fleet idle time goes down significantly while minimizing the number of vehicles. The FSM strategy is justified by the reduction in fleet idle time, fleet cost savings, reduced user commute time, and reduced operational costs under disruption. It is also worth noting that our G-ALNS-FSM achieves lower operational costs than G-TS-FSM.

In Table 8, we compare the evolution of the objective function attained by G-ALNS with an existing G-TS algorithm [27]. It can be observed that the proposed G-ALNS produces better solutions at various time stamps when compared to the G-TS method on all of the tested DARP instances. Figure 4 shows the evolution of the objective function by our G-ALNS when compared to the G-TS method for the R4a test instance.

In Table 9, we analyze the influence of breakdown on fleet idle time. The results show that the proposed G-ALNS-FSM, when compared to G-ALNS, reduces the fleet idle time by 58.19% before breakdown and 59.37% after breakdown, respectively. In the next section, we offer concluding remarks and suggest potential future research directions.

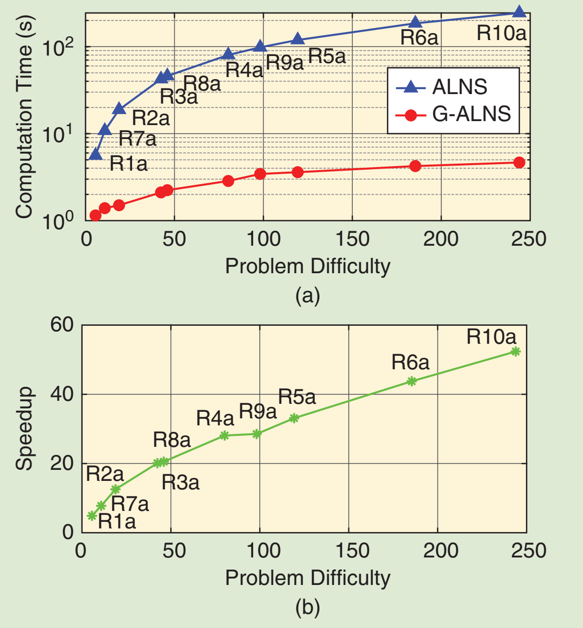


FIG 2 (a) A comparison of the average computational time attained by G-ALNS versus ALNS (CPU) for 10^3 total iterations. (b) A comparison of the average speed-up attained by G-ALNS versus ALNS (CPU).

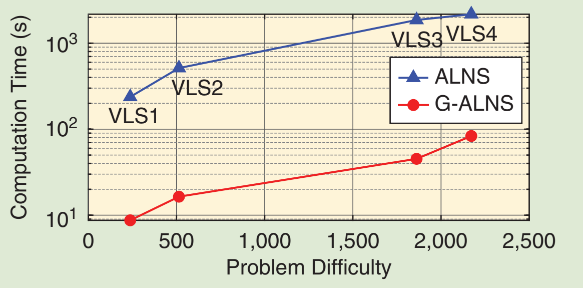


FIG 3 The computational efficiency of the proposed method on very large instances.

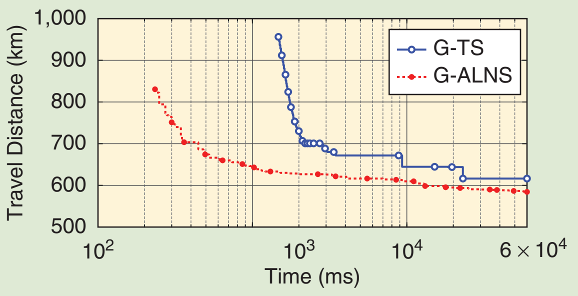


FIG 4 The evolution of the objective function over time for G-ALNS versus a recent G-TS method on test instance R4a.

Conclusions

Disruption in centralized fleet operations has been a big issue since the recovery cost is huge for the transportation industry. Despite its importance, this problem has never been addressed in the context of the advanced user-oriented forms of mobility service that are becoming increasingly relevant in today's ride-sharing transportation. In this article, we focused on the DDARP, in which VB may occur at any point of time during the transportation service. We formulated the DDARP-VB and proposed a novel G-ALNS algorithm to solve this problem. The general idea of ALNS is the removal and reinsertion of requests based on their past performances. During the reinsertion of a request, there are many combinations of possible solution outcomes. Since the sequential evaluation of all of these combinations is computationally expensive, we parallelized this operation using CUDA kernels in GPU.

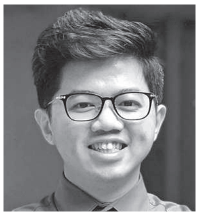
The results show that the proposed G-ALNS performs faster than its CPU counterpart while performing better than an existing G-TS algorithm in a shorter time. The G-ALNS with an FSM strategy attains solutions with a significant reduction in fleet cost at the expense of a negligible increase in travel cost. With realistic estimations of operational cost, the G-ALNS-FSM attains lower operational cost compared to G-ALNS when tested on various standard instances from the literature. Some potential future research directions include 1) an extension of G-ALNS-FSM to emergency DAR systems, such as medical, fire, and rescue services; 2) prediction techniques to tackle the DDARP with traffic congestion and investigate the influence of passenger comfort under congestion; and 3) investigation of the latest vehicular technologies, such as electric, hybrid, and autonomous vehicles, in the context of demand-responsive transportation.

Acknowledgments

This research was partially supported by the ST Engineering–Nanyang Technological University Robotics Corporate Laboratory through the National Research Foundation corporate lab@university scheme. We would like to thank the four anonymous referees and associate editor for their valuable comments and suggestions, which helped significantly improve the overall quality of this article.

About the Authors

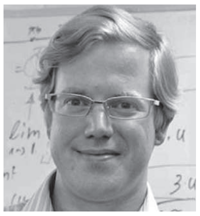
Ramesh Ramasamy Pandi (ramesh006@e.ntu.edu.sg) earned his B.E. degree in electronics and communication engineering from Anna University, India, in 2014 and his M.Sc. degree in signal processing from Nanyang Technological University (NTU), Singapore, in 2016. He is currently a Ph.D. candidate with the School of Electrical and Electronic Engineering at NTU. His research interests are at the intersection of combinatorial optimization, high-performance computing, and machine learning, with applications to intelligent transportation systems. He is a recipient of the prestigious Nanyang award, the Kwok Chin Yan Award, the NTU research scholarship, and the Best Paper Award.



Song Guang Ho (hosongguang@gmail.com) earned his B.Eng. degree in electrical and electronics engineering from Universiti Malaysia Sabah, in 2016 and his M.Sc. degree in computer control and automation from Nanyang Technological University (NTU), Singapore, in 2017. He was a research associate with the ST Engineering–NTU Robotics Corporate Laboratory until 2018. Since then, he has been a research engineer with ST Engineering Land Systems, Singapore. His research interests include intelligent transportation systems, operations research, combinatorial optimization, and computational algorithms.

Sarat Chandra Nagavarapu (sarat@ntu.edu.sg) earned his B.Sc. and M.Sc. degrees in electronics from Andhra University, Visakhapatnam, India, in 2006 and 2008, respectively, and his M.Tech. degree in information technology from the International Institute of Information Technology, Bangalore, Bengaluru, India, in 2010. He earned his Ph.D. degree in systems and control engineering from the Indian Institute of Technology, Bombay, Mumbai, India, in 2016. He is currently a research fellow with the Satellite Research Centre, Nanyang Technological University, Singapore. His research interests include multirobot systems, vehicle-routing algorithms, and designing cooperative exploration and mapping strategies for autonomous agents. He is a Member of IEEE.

Twinkle Tripathy (tripathy.twinkle@gmail.com) earned her B.Tech. degree in electronics and instrumentation engineering from the College of Engineering and Technology, Bhubaneswar, India, in 2011 and her M.Tech./Ph.D. dual degree in systems and control engineering from the Indian Institute of Technology, Bombay, Mumbai, India, in 2016. She is currently a postdoctoral fellow with the Faculty of Aerospace Engineering, Technion, Israel Institute of Technology, Haifa, Israel. Her current research interests include guidance and control of autonomous agents, cooperative control, and missile guidance. She is a Member of IEEE.



Justin Dauwels (jdauwels@ntu.edu.sg) earned his Ph.D. degree in electrical engineering from the Swiss Polytechnical Institute of Technology, Zurich, in 2005. He is currently an associate professor with the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore, where he is also the deputy director of the ST Engineering–NTU Corporate Lab. He has filed more than 10 U.S. patents related to data analytics. His research on intelligent transportation systems has been featured by the BBC, Straits Times, and Channel 5 as well as in various other media outlets. His research on the diagnosis of Alzheimer disease is featured at a five-year exposition at the Science Center, Singapore. He became a Japan Society for the Promotion of Science (JSPS) Post-Doctoral Fellow in 2007, a Belgian American Educational Foundation Fellow in 2008, and a Henri-Benedictus Fellow of the King Baudouin Foundation in 2008. He was a JSPS Invited Fellow in 2010 and 2011. He is a Senior Member of IEEE.

References

[1] S. Shaheen, A. Cohen, B. Yelchuru, and S. Sarkhili, "Mobility on demand: Operational concept report," Dept. of Transportation, Intelligent Transportation, Washington, D.C., Tech. Rep. FHWA-JPO-18611, 2017.

[2] S. Shaheen, A. Cohen, and I. Zohdy, "Shared mobility: Current practices and guiding principles," Federal Highway Administration, Washington, D.C., Tech. Rep. FHWA-HOP-16-022, 2016. [Online]. Available: https://ops.fhwa.dot.gov/publications/fhwahop16022/fhwahop16022.pdf

[3] R. Cervero, *Paratransit in America: Redefining Mass Transportation*. Westport, CT: Greenwood Publishing Group, 1997.

[4] J.-F. Cordeau and G. Laporte, "A tabu search heuristic for the static multi-vehicle dial-a-ride problem," *Transp. Res. B, Methodol.*, vol. 37, no. 6, pp. 579–594, 2003. doi: 10.1016/S0191-2615(02)00045-0.

[5] J.-F. Cordeau, "A branch-and-cut algorithm for the dial-a-ride problem," *Oper. Res.*, vol. 54, no. 3, pp. 573–586, 2006. doi: 10.1287/opre.1060.0283.

[6] M. A. Masmoudi, K. Braekers, M. Masmoudi, and A. Dammak, "A hybrid genetic algorithm for the heterogeneous dial-a-ride problem," *Comput. Oper. Res.*, vol. 81, pp. 1–13, May 2017. doi: 10.1016/j.cor.2016.12.008.

[7] S. C. Ho, W. Y. Szeto, Y.-H. Kuo, J. M. Leung, M. Petering, and T. W. Tou, "A survey of dial-a-ride problems: Literature review and recent developments," *Transp. Res. B, Methodol.*, vol. 111, pp. 395–421, May 2018. doi: 10.1016/j.trb.2018.02.001.

[8] W. Zheng, Z. Chun-yue, and H. Xiang-pei, "A simulation-based method for evaluating the reliability of vehicle routing schemes in the vaccine delivery with vehicle breakdown," in *Proc. Int. Conf. Management Science and Engineering (ICMSE)*, 2013, pp. 258–262. doi: 10.1109/ICMSE.2013.6586291.

[9] X. Wang, X. Wu, and X. Hu, "A study of urgency vehicle routing disruption management problem," in *Proc. WASE Int. Conf. Information Engineering (ICIE)*, 2010, vol. 3, pp. 452–455. doi: 10.1109/ICIE.2010.285.

[10] K. Korosec, "Vehicle breakdowns hit record high in U.S. despite new car tech," *Fortune*, July 21, 2016. [Online]. Available: http://fortune.com/2016/07/20/aaa-breakdown-new-tech-study/

[11] Q. Mu, Z. Fu, J. Lysgaard, and R. Eglese, "Disruption management of the vehicle routing problem with vehicle breakdown," *J. Oper. Res. Soc.*, vol. 62, no. 4, pp. 742–749, 2011. doi: 10.1057/jors.2010.19.

[12] J. Clausen, A. Larsen, J. Larsen, and N. J. Rezanova, "Disruption management in the airline industry: Concepts, models and methods," *Comput. Oper. Res.*, vol. 37, no. 5, pp. 809–821, 2010. doi: 10.1016/j.cor.2009.03.027.

[13] J. A. Filar, P. Manyem, and K. White, "How airlines and airports recover from schedule perturbations: A survey," *Ann. Oper. Res.*, vol. 108, no. 1/4, pp. 315–333, 2001. doi: 10.1023/A:1016079600083.

[14] R. Eglese and S. Zambirinis, "Disruption management in vehicle routing and scheduling for road freight transport: A review," *TOP*, vol. 26, no. 1, pp. 1–17, 2018. doi: 10.1007/s11750-018-0470-y.

[15] X. Wang, X. Wu, Z. Wang, and X. Hu, "A model and an improved genetic algorithm for the vehicle routing problem with breakdown vehicles," in *Proc. 4th Int. Conf. Innovative Computing, Information and Control (ICICIC)*, 2009, pp. 696–699. doi: 10.1109/ICICIC.2009.28.

[16] R. E. Wener, G. W. Evans, D. Phillips, and N. Nadler, "Running for the 7:45: The effects of public transit improvements on commuter stress," *Transportation*, vol. 30, no. 2, pp. 203–220, 2003. doi: 10.1023/A:1022516221808.

[17] J.-Q. Li, P. B. Mirchandani, and D. Borenstein, "Real-time vehicle rerouting problems with time windows," *Eur. J. Oper. Res.*, vol. 194, no. 3, pp. 711–727, 2009. doi: 10.1016/j.ejor.2007.12.037.

[18] H. N. Psaraftis, M. Wen, and C. A. Kontovas, "Dynamic vehicle routing problems: Three decades and counting," *Networks*, vol. 67, no. 1, pp. 3–31, 2016. doi: 10.1002/net.21628.

[19] M. van Engelen, O. Cats, H. Post, and K. Aardal, "Enhancing flexible transport services with demand-anticipatory insertion heuristics," *Transp. Res. E, Logistics Transp. Rev.*, vol. 110, pp. 110–121, Feb. 2018. doi: 10.1016/j.tre.2017.12.015.

[20] G. Kim, Y. S. Ong, T. Cheong, and P. S. Tan, "Solving the dynamic vehicle routing problem under traffic congestion," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 8, pp. 2367–2380, 2016. doi: 10.1109/TITS.2016.2521779.

[21] S. N. Parragh, K. F. Doerner, and R. F. Hartl, "Variable neighborhood search for the dial-a-ride problem," *Comput. Oper. Res.*, vol. 37, no. 6, pp. 1129–1138, 2010. doi: 10.1016/j.cor.2009.10.003.

[22] K. Braekers, A. Caris, and G. K. Janssens, "Exact and meta-heuristic approach for a general heterogeneous dial-a-ride problem with multiple depots," *Transp. Res. B, Methodol.*, vol. 67, pp. 166–186, Sept. 2014. doi: 10.1016/j.trb.2014.05.007.

[23] R. Masson, F. Lehuédé, and O. Péton, "The dial-a-ride problem with transfers," *Comput. Oper. Res.*, vol. 41, pp. 12–23, Jan. 2014. doi: 10.1016/j.cor.2013.07.020.

[24] C. Bongiovanni, M. Kaspi, and N. Geroliminis, "The electric autonomous dial-a-ride problem," *Transp. Res. B, Methodol.*, vol. 122, pp. 436–456, Apr. 2019. doi: 10.1016/j.trb.2019.03.004.

[25] P. Shaw, "Using constraint programming and local search methods to solve vehicle routing problems," in *Proc. Int. Conf. Principles and Practice Constraint Programming*, 1998, pp. 417–431. doi: 10.1007/3-540-49481-2_30.

[26] S. Ropke and D. Pisinger, "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows," *Transp. Sci.*, vol. 40, no. 4, pp. 455–472, 2006. doi: 10.1287/trsc.1050.0135.

[27] R. R. Pandi, S. G. Ho, S. C. Nagavarapu, T. Tripathy, and J. Dauwels, "GPU-accelerated tabu search algorithm for dial-a-ride problem," in *Proc. 21st Int. Conf. Intelligent Transportation Systems (ITSC)*, 2018, pp. 2519–2524. doi: 10.1109/ITSC.2018.8569472.

[28] J.-Y. Potvin and J.-M. Rousseau, "A parallel route building algorithm for the vehicle routing and scheduling problem with time windows," *Eur. J. Oper. Res.*, vol. 66, no. 3, pp. 331–340, 1993. doi: 10.1016/0377-2217(93)90221-8.

[29] J. Kwok, "Car, taxi and bus: How the costs stack up," *Straits Times*, June 29, 2014. [Online]. Available: https://www.straitstimes.com/business/invest/car-taxi-and-bus-how-the-costs-stack-up

[30] M. Chan, "No more queuing for car washes—This S'pore startup will come to you for just $14.90," *Vulcan Post*. [Online]. Available: https://vulcanpost.com/641548/shiokr-on-demand-car-wash-singapore/

[31] SGCarmart. Accessed: Aug. 15, 2019. [Online]. Available: https://www.sgcarmart.com/news/carpark_index.php?LOC=all&TYP=carpark&SRH

ITS