

Adaptive Algorithm for Dial-A-Ride Problem with Vehicle Breakdown

Ramesh Ramasamy Pandi, Song Guang Ho, Sarat Chandra Nagavarapu, and Justin Dauwels

Abstract—Vehicle breakdown in a centralized fleet operation can inflict large recovery costs and may damage service provider's reputation. We consider the disrupted dial-a-ride problem (DARP), where vehicle breakdown can occur at any point of time during the transportation service. The conventional method to handle vehicle breakdown is to own backup vehicles or rent additional vehicles when necessary, so that the impaired vehicles can be replaced if any vehicle breaks down. In this paper, we introduce an adaptive algorithm with GPU-acceleration, to quickly build routes and schedules for the DARP in the occurrence of unforeseen stochastic events. Computational experiments are conducted on various standard DARP instances, with realistic estimates of operational cost. The results show that the new adaptive algorithm reduces the operational cost when compared to the conventional method.

I. INTRODUCTION

Smart urban mobility is one of the prime areas of focus for most of the countries nowadays to fulfil people's daily transportation needs, enhance travel comfort and provide superior user convenience. Rapid advancements in the areas of autonomous vehicle technology, parallel computing, 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, etc. need efficient fleet management solutions developed using these emerging technologies to provide satisfactory service to the customers.

Dial-a-ride problem (DARP) [1], a special class of vehicle routing problem (VRP), belongs to the category of people transportation, 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 time windows associated with pick-up and drop-off. Several optimization algorithms exist in the literature to provide optimal [2] or near-optimal [3][4] solutions to solve the problem. However, in many real time scenarios, fleet operations are often hindered by the occurrence of many unanticipated stochastic events, such as vehicle breakdown, accidents, drastic weather conditions, etc., which in turn diminish the solution quality.

Regardless of how technology advances, vehicle breakdowns still remain as a major contributor to the disruption

Ramesh Ramasamy Pandi is a PhD candidate, Song Guang Ho is a research associate, Sarat Chandra Nagavarapu is a research fellow, and Justin Dauwels is an associate professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, 639798. E-mail: (ramesh006@e.ntu.edu.sg, nigelho@ntu.edu.sg, sarat@ntu.edu.sg, and jdauwels@ntu.edu.sg).

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

of transportation services across the globe. In US alone, a massive 32 million car breakdowns are recorded [5] for the year 2015 and the numbers are rising up further every year. In the context of dial-a-ride systems, this results in an inflated operational cost and damaged reputation for the service providers (operators of taxis, private buses, etc.). Therefore, the development of efficient fleet management strategies to tackle vehicle breakdown is vital to provide uninterrupted transportation service to the passengers.

Although the problem of solving vehicle routing with breakdown is very important in many real time applications, it has gained less attention from the research community. Only a hand-full of methods are available in the literature to solve this problem. A tabu search algorithm [6] for disruption management is proposed to generate new route plans for the delivery problem. Optimization models based on networks for complex aircraft re-routing and crew scheduling are detailed in [7] and [8]. [9] discusses different formulations and existing methods in the literature to address disruption management in road freight transport systems. A saving based heuristic [10] is proposed to evaluate the reliability of vehicle routing schemes in the vaccine delivery with vehicle breakdown. A disruption management mathematical model [11] and a Lagrange heuristic are developed to address the urgency vehicle routing problem. An improved genetic algorithm [12] was proposed to address VRP with breakdowns. However, to the best of our knowledge, there exists no method to address DARP with vehicle breakdown. Since demand responsive transportation is becoming a popular choice for daily commute, it is important to address disruption in DARP.

Therefore, this paper proposes an adaptive routing algorithm to solve the disrupted dial-a-ride problem. The major contributions of the paper are summarized as follows:

- An adaptive algorithm is introduced to address DARP with vehicle breakdown, in which a GPU kernel function and a tabu list are incorporated to enhance computing efficiency and search efficiency of the route optimization process in an event of vehicle breakdown.
- The performance of the proposed algorithm is evaluated on various standard DARP instances and compared against the conventional method.

The remainder of the paper is organized as follows: Section II discusses the problem formulation for the disrupted DARP. Section III presents the proposed adaptive algorithm for the DARP with vehicle breakdown. We report the simulation results and provide comparative analysis in Section IV, and offer concluding remarks in Section V.

II. PROBLEM FORMULATION

In this section, we describe the formulation of dial-a-ride problem with vehicle breakdown.

A. DARP with Vehicle Breakdown

Dial-a-ride problem addresses the transportation of people between specified pickup and drop-off locations. DARP is solved as a combinatorial optimization problem with the aim of minimizing the total travel time while satisfying all the constraints related to passenger convenience. The formulation of DARP in [1] is often regarded as the standard version. In the standard DARP, a homogeneous set of requests are considered with fixed number of vehicles located at a single depot. All vehicles are considered to have constant speed throughout the network, and euclidean distance formula gives the distance between any two nodes in the network. Since it is one of the most widely studied DARP in the literature, our formulation of DARP with vehicle breakdown is built upon on this by incorporating additional constraints associated with the occurrence of breakdown.

In the standard dial-a-ride problem, n customer requests are served by m vehicles. Each request i consists of time window either for departure or arrival vertex or both. The objective is to minimize the travel cost subject to several constraints. Let $X = \{x_1, x_2, \dots\}$ 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) departure vertex v_i and arrival vertex v_{i+n} must belong to the same route, and 3) arrival vertex v_{i+n} is always visited after departure vertex v_i . Any solution that violates these basic set of constraints becomes infeasible. In addition to these, there are four major constraints that need to be satisfied: 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 the time constraint on pickup and drop-off is violated; ride time constraint violation $t(x)$ occurs when a passenger is transported for a longer time than ride time limit L . These four constraints can be formulated mathematically as follows:

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

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

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

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

There are two types of passenger requests considered often in DARP, trip from home, and return trip to home. The first one is considered to be critical and the second one is considered to be non-critical. When a passenger wants to

travel from home to office or airport, a tighter time-window will be requested for the drop-off service, while the pick-up time window is loosely constrained. On the other hand, when a passenger wants to travel back to home, the pick-up time window is tightly constrained than that of drop-off. Therefore, the time-window of either pick-up or drop-off service have tighter constraint than that of the other. The definition of DARP with vehicle breakdown can be extended as follows.

- 1) All accepted passenger requests must be served either feasible or infeasible.
- 2) During an event of breakdown, a vehicle may be at one of the following states:
 - Vehicle has not started the service.
 - Vehicle is traveling between two nodes.
 - Vehicle is traveling between two nodes, with a passenger on-board.
 - Vehicle has arrived at a particular node, and waiting for the passenger to show up.
 - Vehicle is currently serving a passenger at a particular node.
 - Vehicle has completed the service.
- 3) A passenger i affected by breakdown event can be served by another vehicle. For example, when a passenger i is inside a vehicle k_{VB} that involved in an event of breakdown, then the passenger i can be served by another vehicle k_{VR} while ensuring all the constraints are met. If that passenger request is critical, then they are most likely to be served by a new vehicle without constraint violation. However, if it is a non-critical request, the previously tighter schedule produced by the algorithm decreases the chance of feasibly serving this user. If the passenger i is dropped off by a vehicle that is involved in breakdown event, then the time window for the subsequent pick-up will be loosely constrained. In the event of vehicle breakdown, it is important to ensure uninterrupted transportation service for all passengers with minimum disruption to passenger convenience.
- 4) Ride time limit L , which is very important in DARP, needs to be satisfied irrespective of vehicle breakdown to guarantee feasibility. If a passenger i is inside a vehicle k_{VB} which is broken down, then the passenger i is dropped off at the location of breakdown. Furthermore, a new request will be dynamically generated with the altered constraints from the previous problem. The ride time constraint is set as the difference of original ride time limit L and the travel time L_i of the passenger during the time of breakdown.
- 5) When a vehicle k involves in a breakdown event, it cannot serve any more passengers for the particular day. Therefore, it rejects all the scheduled requests to be visited in the future, while dropping off passengers on-board immediately at the location of breakdown.

In the next section, we present the proposed adaptive algorithm for the DARP with vehicle breakdown.

III. ADAPTIVE ALGORITHM

In this section, we introduce an adaptive algorithm to solve DARP with vehicle breakdown. The key ideas are to dynamically optimize the route plan in an event of vehicle breakdown, eliminating the involvement of requests in local search which were already served at the time of disruption, dynamic re-insertion of affected passenger requests into the solution that have been dropped off by the vehicle, and exploiting GPU acceleration to quickly attain solutions. In overall, we provide a complete methodology to solve the disrupted DARP with vehicle breakdown, with an objective to minimize the distance traveled by the fleet. Fig. 1 shows the illustration of a vehicle breakdown interrupting the service provided by demand responsive transportation (DRT).

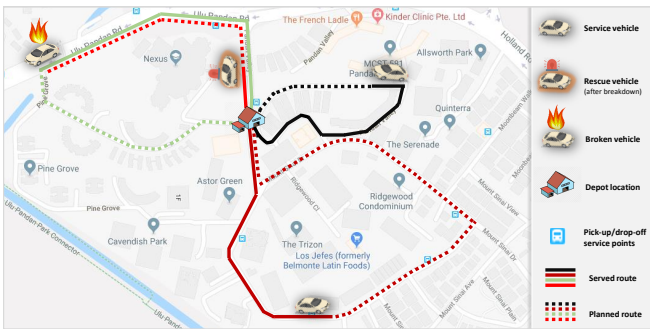


Fig. 1. Illustration of vehicle breakdown for DRT service in Singapore

A. Real-time implementation

The flow of the algorithm is as follows:

- 1) Allocate GPU memory resources, and transfer the DARP instance data from CPU to GPU.
- 2) A DARP solution is encoded as a set of routes and schedules. Construct an initial solution for the given instance using a construction heuristic. We use the construction heuristic provided by [3] in our algorithm. Afterwards, the generated initial solution is set as the current solution.
- 3) The route plan for the current solution is optimized based on step (5), and then the service provider can start to dispatch vehicles accordingly to the best solution provided by the algorithm. Thereafter, on each iteration, the algorithm attempts to minimize the objective function further, while monitoring any cases of vehicle breakdown based on the information provided in real-time or simulation feedback.
- 4) If the routing analyzer triggers a breakdown event,
 - Extract the information of broken vehicle k_{VB} and event occurrence time t_{VB} from the system. For simulations, we randomly determine the values for k_{VB} and t_{VB} , provided that the vehicle k_{VB} has begun the service, but not ended for the day.
 - The location of all vehicles during the event can be retrieved using the global positioning system in real-time or computed externally in simulation.

- As per the event, the requests that have been already served by the fleet is updated in a list, namely $T_{tabulist}$. Therefore, we ensure that such requests do not require relocation during online route optimization process. Furthermore, we also keep track of the *halt_Index* that contains the previously served nodes at the time of breakdown.
- If there are passengers inside the vehicle k_{VB} , then they will be dropped off immediately at the breakdown location, which will be updated in the system accordingly. Meanwhile, the new travel times from the breakdown vertex to all other vertices will be updated, besides modifying the constraints of affected passengers. Notably, the non-critical or return-to-home type of requests have wider time window for drop-off service and higher probability to be feasibly served when compared to the critical one's.

5) Otherwise,

- Asynchronously copy the current solution from CPU to GPU, reducing the data transfer latency.
 - While considering the $T_{tabulist}$ and *halt_Index*, perform local search operation on the current solution. The key idea to generate solutions quickly is to exploit GPU acceleration by parallel exploration of neighborhood, and employing a tabu list that permanently forbids the previously served requests narrowing down the neighborhood region. As a result, the adaptive algorithm improves both the computing efficiency and search efficiency of the optimization process. Section III-B has the implementation details on route optimization, and Algorithm 2 has the details of GPU kernel code.
 - After exploring the neighborhood solutions, the program control is transferred back to CPU. Afterwards, a data transfer request is initiated by CPU. The transferred data contain only a selective set of information i.e. request relocation details, and the value of objective function, instead of the entire solution object. By this way, the communication latency between CPU and GPU is reduced.
 - At the end of iteration, the best neighborhood move that yields the minimum objective function will be applied on the current solution. Finally, if necessary, the algorithmic parameters are tuned.
- 6) Once the stopping criterion is satisfied, the best solution that has minimum travel cost is returned and the GPU resources are released.

Fig. 2 shows flowchart of a real-time implementation. As the service provider dispatches the fleet to serve the passengers based on the solution provided by the algorithm, the routing analyzer constantly keeps track of the event sequence. In case of unforeseen vehicle breakdown, the information corresponding to the broken down vehicle will be fed back to the optimizer, which include vehicle coordinates (x_{VB}, y_{VB}) , vehicle ID k_{VB} , and event occurrence

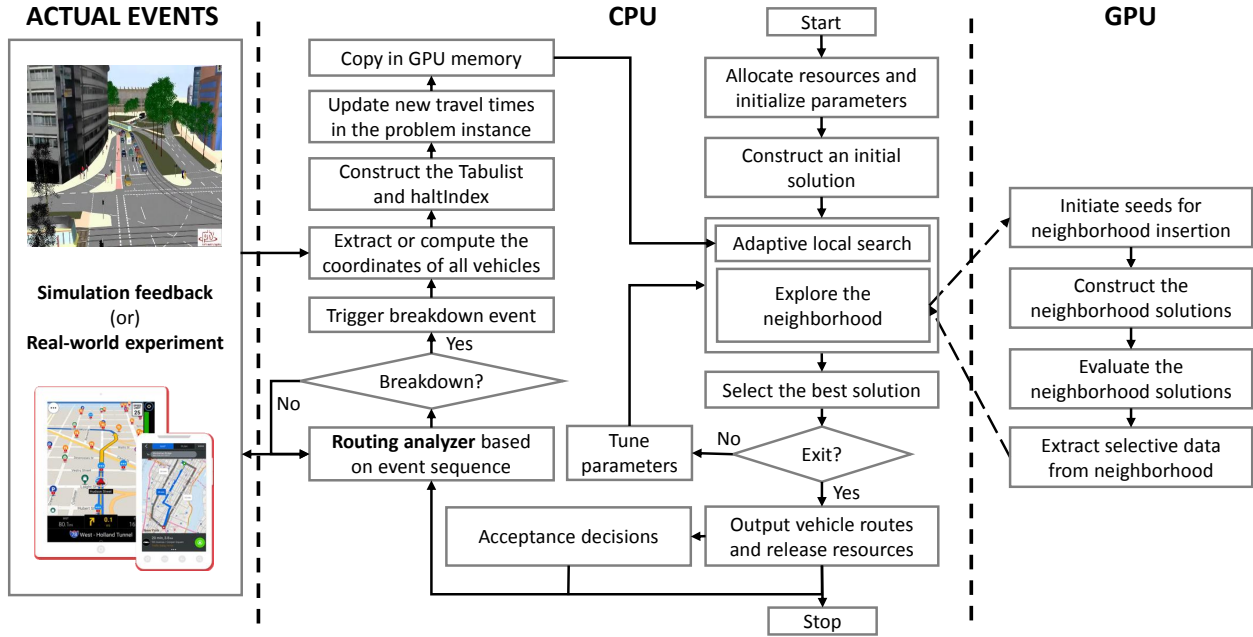


Fig. 2. Flowchart: Real-time adaptive algorithm for the DARP.

Algorithm 1 Adaptive Algorithm

Require: DARP Instance

Ensure: Best solution found

- 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 selected initial solution.
 - 5: **repeat**
 - 6: **if** *check(trigger_breakdown)* **then**
 - 7: Choose a vehicle k_{VB} to breakdown at time t_{VB} .
 - 8: Compute the coordinates for all vehicles
 - 9: Construct the tabu list $T_{tabulist}$ and $halt_Index$.
 - 10: **if** passengers inside $k_{VB} > 0$ **then**
 - 11: Update new coordinates for those passengers.
 - 12: Update new travel times between the nodes.
 - 13: **end if**
 - 14: **else**
 - 15: Copy the current solution from CPU to GPU.
 - 16: **while** considering $T_{tabulist}$ and $halt_Index$ **do**
 - 17: Perform *local search* operation.
 - 18: Launch *Parallel_N.Explore()* kernel in GPU.
 - 19: **end while**
 - 20: Copy selective data from GPU to CPU.
 - 21: Perform the best move on current solution.
 - 22: Tune the algorithmic parameters.
 - 23: **end if**
 - 24: **until** stopping criterion
 - 25: Release GPU resources.
-

time t_{VB} . As a reflection, the algorithm performs online route optimization and returns the vehicle routes that has minimum travel cost. In this implementation, we serve all the requests either feasible or infeasible to maintain service provider's reputation. However, in the real-world, decisions on accepting passenger requests are generally made based on the feasibility and the operational cost incurred. Once all the passengers in the current demand pool are served and all vehicles from the fleet return to the depot, the system can be shut down. Algorithm 1 has the implementation details.

B. Route optimization: Tabu list for search efficiency

The goal of route optimization is to generate a route plan that has minimum travel cost. Here, the distance traveled by the fleet is considered as the travel cost. To perform that, we use a multi-atomic annealing heuristic [3] with GPU-acceleration [13]. In our implementation, the local search relocation operators in [3] defines the neighborhood structure besides adopting the temperature schedule. Furthermore, to enhance the search efficiency, we incorporate a tabu list that permanently forbids a set of requests that is already served at the time of breakdown event. Therefore, such requests need not to be involved in relocation operation, limiting the size of neighborhood being explored in the adaptive local search. Besides, the algorithm supports insertion of dynamic requests from the same demand pool i.e. re-insertion of dropped off requests into the solution pool again during the optimization process. Notably, the passengers affected by the broken vehicle can be served by another vehicle. Such a methodology is useful for those requests that have been dropped off by the vehicle that is broken down, as they will place a new transportation request with constraints defined according to the disrupted DARP formulation.

C. Parallel processing: GPU kernel for computing efficiency

From the preliminary tests, it was clear that neighborhood exploration for point-based meta-heuristic consumes long execution time. Since we implement a multi-atomic annealing that is point-based, we developed a GPU kernel function based on [13] to accelerate the process of local search operation. In the designed kernel, each thread t_{id} is responsible for exploration of each neighborhood solution from the pool of N_{size} solutions. For each thread, a request i will be relocated from its current vehicle k_1 to new vehicle k_2 at a predefined position, before evaluating the feasibility. Notably, the vehicle that is involved in the breakdown will be forbidden. To reduce communication latency between CPU and GPU, only a selective set of data including request relocation information and the value of objective function are placed in an array that will be later copied back to CPU.

Algorithm 2 GPU kernel function: *Parallel_N_Explore*

Require: DARP Instance; Current solution

- 1: $t_{id} = threadIdx.x + blockIdx.x * blockDim.x$;
 - 2: **if** $t_{id} < N_{size}$ **then**
 - 3: neighborhood solution $x_{t_{id}} \leftarrow$ current solution
 - 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 solution $x_{t_{id}}$ and objective function
 - 8: Extract selective data that will be copied back to CPU
 - 9: **end if**
-

In the next section, we evaluate the performance of the proposed adaptive algorithm and compare the results with conventional method for DARP with vehicle breakdown.

IV. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the adaptive algorithm on various standard DARP instances from the literature, and compare the results with conventional method to handle vehicle breakdown. The implementation details, definition of operational cost function based on realistic estimates, simulation settings for DARP with vehicle breakdown, and results and discussion, are provided as follows.

A. Implementation details

The proposed adaptive algorithm for DARP with vehicle breakdown is implemented in CUDA. All simulations are carried out on a computer running 2.1 GHz Intel Xeon E5-2620 v4 processor and Nvidia Tesla P100-PCI-E-16GB GPU, interconnected with a PCI-e bus. Visual studio IDE 2015 with CUDA 8.0 toolkit is used as the development platform.

B. Operational cost function

Operational cost is one of the important parameters to assess the fleet operations, which often refers to the overall cost incurred by the service provider or fleet owner that include: vehicle procurement cost, licensing fees, fuel, insurance, driver wages, maintenance, repairs, parking, toll taxes and all other mileage-dependent depreciation costs. In order to

perform realistic evaluation of our adaptive algorithm, we have considered the actual vehicle operational cost structure [14] observed in Singapore. The average operational cost for an owned and rented vehicle in Singapore are 1620\$/month and 2880\$/month respectively, excluding fuel cost being 0.24\$/km. The strategy involving renting a vehicle, in case of breakdown, is a full day rental cost.

The operational cost function is given by:

$$\alpha(x) = \beta m + \gamma \Phi(x) + \delta \Gamma(x), \quad (5)$$

where, m is the total number of own vehicles involved in the fleet operation, β is maintenance cost in \$ per own vehicle per day, $\Phi(x)$ is the total distance traveled in km by both own and rental vehicles, γ is the estimated petrol cost for a vehicle per day, $\Gamma(x)$ is the number of additional vehicles added to the fleet after breakdown, and δ is the estimated rental cost of a vehicle per day.

C. Simulation settings

For the simulation, we have considered standard DARP benchmark instances R1a, R2a, R3a, and R4a [1]. In these instances, the fleet size is 3, 5, 7, and 9, respectively and the corresponding number of requests is 24, 48, 72, and 96. Each vehicle has a capacity of 6 passengers with a maximum route duration of 480 min. Maximum user ride time and service time are 90 min and 10 min respectively. The time window length for either pick-up or drop-off vertex is 15 min, and for the other subsequent vertex is 1400 min.

To ensure fair comparison, each experiment is conducted five times and only average values of the results are reported. The result quantifiers include, total number of feasibly served customers, total distance traveled by all vehicles, total number of additional vehicles used, and operational cost for the entire transportation service. On each simulation run, we trigger breakdown when the iteration number reaches 10^4 . Once the breakdown is triggered, we continue to run the adaptive algorithm till the post-breakdown stopping criterion is satisfied, which is set as a predefined run time of 60 sec starting from the event of breakdown. Finally, we report the average percentage savings in operational cost $\eta(\%)$ attained by the adaptive algorithm when compared to the conventional method. The term η is defined as follows:

$$\eta(\%) = \frac{\alpha(x_{AA}) - \alpha(x_{CM})}{\alpha(x_{CM})} \times 100, \quad (6)$$

wher $\alpha(x_{AA})$ is the operational cost of solution x_{AA} attained by adaptive algorithm, $\alpha(x_{CM})$ is the operational cost of solution x_{CM} attained by conventional method.

D. Results and discussion

Table I shows the comparison of results attained by the Adaptive Algorithm (AA) and Conventional Method (CM) for DARP with vehicle breakdown. In the Table I, BKS indicates the travel cost for best-known solution for the particular DARP case, while n and m represents the number of passenger requests and fleet size respectively. Before

Case	n	m	BKS	Benchmark solution			Solution after breakdown			Conventional Method				Adaptive Algorithm			
				n^*	$\Phi(x)$	$\alpha(x)$	n^*	$\Phi(x)$	$\alpha(x)$	n^*	$\Phi(x)$	$\Gamma(x)$	$\alpha(x)$	n^*	$\Phi(x)$	$\alpha(x)$	η (%)
R1a	24	3	190.02	24	190.17	208\$	15.60	129.52	193\$	24.00	194.57	1	305\$	24.00	204.83	211\$	30.70
R2a	48	5	301.33	48	308.84	344\$	42.40	266.03	334\$	47.60	313.60	1	441\$	48.00	334.19	350\$	20.64
R3a	72	7	532.00	72	560.29	512\$	65.60	519.87	503\$	72.00	565.61	1	610\$	72.00	580.87	517\$	15.14
R4a	96	9	570.24	96	614.00	633\$	87.50	566.62	622\$	95.75	618.62	1	730\$	95.80	673.14	648\$	11.35

TABLE I
RESULTS ATTAINED BY THE ADAPTIVE ALGORITHM AND CONVENTIONAL METHOD FOR DARP WITH VEHICLE BREAKDOWN.

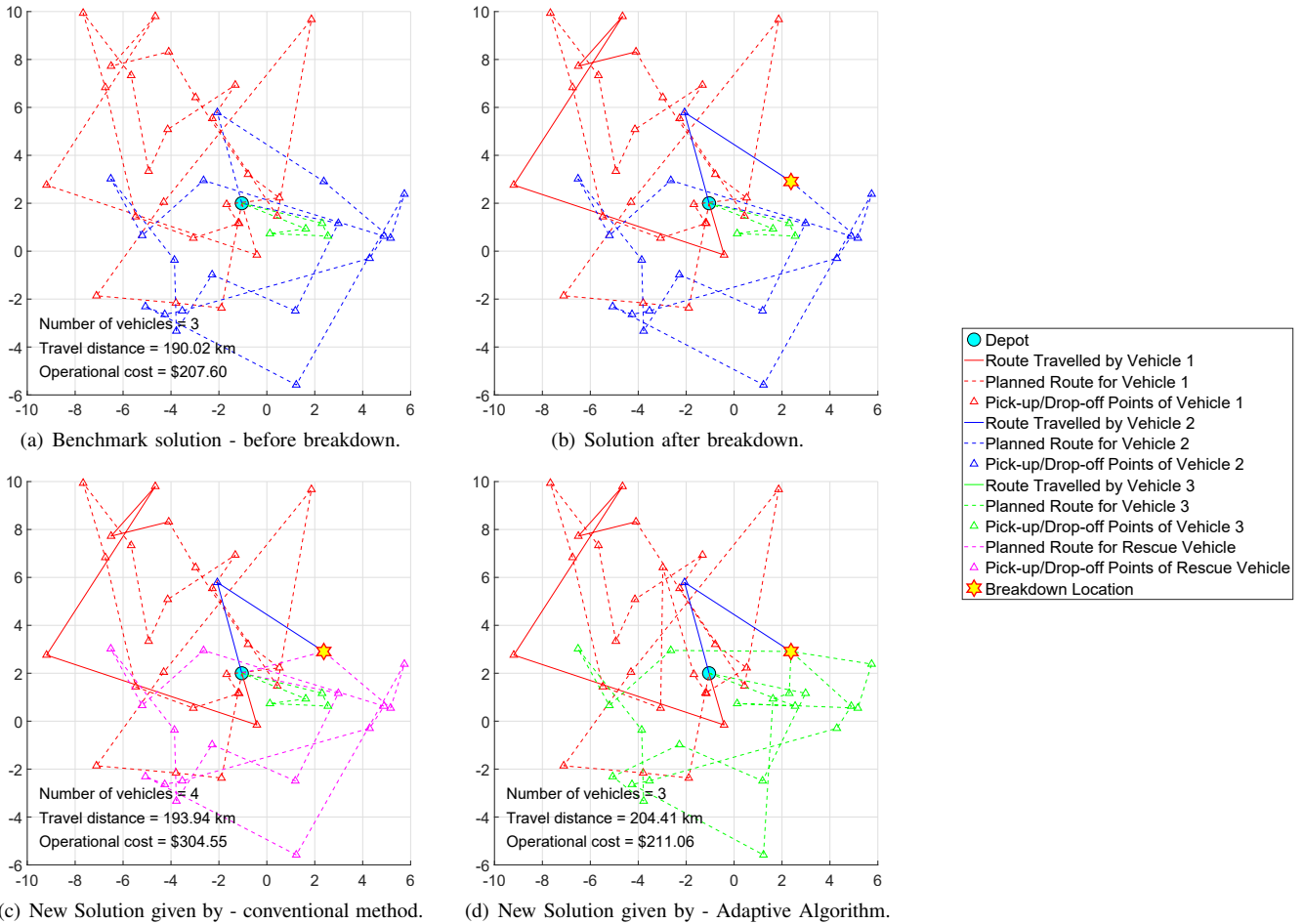


Fig. 3. Simulation results of DARP with vehicle breakdown (sample results are provided by simulation on R1a instance).

triggering the breakdown event, the solution provided by AA is in the column *Benchmark Solution*, and the altered solution after breakdown is given by the column *Solution after Breakdown*. The *Conventional method* column shows the new solution attained by CM after breakdown by dispatching a rental vehicle from the depot along the previously planned route with a new schedule provided by scheduling heuristic of [1]. Finally, the *Adaptive Algorithm* column provides the details of new solution generated by AA after breakdown. For the solutions attained by CM and AA, we show the average number of passengers feasibly served n^* , distance traveled by the fleet $\phi(x)$, number of additional

rental vehicles $\Gamma(x)$, and operational cost incurred $\alpha(x)$. Finally, we show the percentage savings in operational cost $\eta(\%)$ attained by AA when compared to CM. From the results, it can be identified that our adaptive algorithm clearly outperforms the conventional method in terms of percentage savings in operational cost $\eta(\%)$ of 30.70, 20.64, 15.14, and 11.35 for R1a, R2a, R3a and R4 respectively.

Figs. 3(a)-3(d) show the simulation results for DARP with vehicle breakdown. The results are obtained on a single simulation run by solving R1a DARP instance. Fig. 3(a) shows the planned route by AA before breakdown, Fig. 3(b) shows the solution after breakdown, Fig. 3(c) shows the new

solution generated by CM, and Fig. 3(d) shows the new solution generated by AA. It can be seen that the operational cost incurred by AA is lesser than that of CM, and slightly higher than that of the estimated cost of benchmark solution.

	CM	AA
Increase in $\alpha(x)$ when compared to benchmark (%)	22.89	1.69
Increase in number of passengers feasibly served (%)	21.32	21.57
Increase in number of rental vehicles used (%)	16.67	0.00
Increase in scheduled passenger waiting time (%)	30.31	31.26
Increase in scheduled passenger ride time (%)	7.73	8.75

TABLE II

COMPARISON OF RESULTS ATTAINED BY AA AND CM FOR DARP-VB.

Table II reports the average values of various performance quantifies obtained for CM and AA. While optimizing the routes using AA, we incur only 1.69% increase in operational cost when compared to the estimated cost of benchmark solution. In contrast, the CM leads to 22.89% increase in operational cost. The increase in number of customers feasibly served indicates the percentage difference between the number of customers served while handling breakdown using CM or AA, and when no action is taken. The number of rental vehicles used is 16.67% for CM, while obtaining 21.32% increase in number of customers feasibly served. Meanwhile, the AA achieves 21.57% increase, without exploiting the benefits of rental vehicles. Due to the breakdown event, the difference in scheduled waiting time and ride time for each passenger in a new solution drastically differs from the benchmark solution before breakdown. In that aspect, CM performs negligibly better than the adaptive algorithm.

Although the distance traveled $\phi(x)$ by the fleet for AA is higher than CM, the idea of renting an additional vehicle seriously increases the operational cost for CM. Besides, the importance of serving the affected passengers during breakdown can be seen from the fact that, on average, 12.04% of the customers goes unserved if no action is taken during the breakdown, which may potentially damage the reputation of service provider. For the tested instances [1], the location of the depot is equal to the average location of the seed points used to generate origin and destination locations. In real-world problems, the depot can be anywhere, the demand distribution varies according to the time of day. Hence, dynamically optimizing the routes of vehicles that are in the vicinity of affected demand pool can be advantageous over dispatching an extra vehicle.

Therefore, we conclude with the following points. While the distance traveled by the fleet for AA is higher than CM, higher passenger satisfaction without exploiting rental vehicles and lesser operational cost, are the key advantages of AA. Despite a slight delay incurred by the execution time of AA in decision-making, it can be still justified that AA is better based on the magnitude of savings in operational cost when compared to the conventional method.

V. CONCLUSION

We have considered a disrupted dial-a-ride problem in which vehicle breakdown occurs at any point of time during the transportation service. Disruption in centralized fleet operations has been big issue since the cost of recovery is huge for the transportation industry. Despite the importance, this type of problem has not been addressed in the context of advanced user-oriented form of mobility service that is becoming increasingly relevant in today's ride-sharing transportation. Therefore, we have proposed an adaptive algorithm to solve DARP with vehicle breakdown. With realistic estimates of operational cost, the AA outperforms CM in terms of operational cost for the disrupted DARP, when tested on various standard instances from the literature.

The future direction can be the investigation of multi-vehicle breakdowns in richer contexts of DARP with time-dependent travel times and heterogeneous demand. Extension of our method in emergency contexts of DARP such as medical, fire and rescue can be valuable provided the response time criticality of these applications.

REFERENCES

- [1] 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.
- [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. 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.
- [4] S. G. 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.
- [5] "Vehicle breakdowns hit record high in u.s. despite new car tech," <http://fortune.com/2016/07/20/aaa-breakdown-new-tech-study/>.
- [6] Q. Mu, Z. Fu, J. Lygaard, and R. Eglese, "Disruption management of the vehicle routing problem with vehicle breakdown," *Journal of the Operational Research Society*, vol. 62, no. 4, pp. 742–749, 2011.
- [7] J. Clausen, A. Larsen, J. Larsen, and N. J. Rezanova, "Disruption management in the airline industry—concepts, models and methods," *Computers & Operations Research*, vol. 37, no. 5, pp. 809–821, 2010.
- [8] J. A. Filar, P. Manyem, and K. White, "How airlines and airports recover from schedule perturbations: a survey," *Annals of operations research*, vol. 108, no. 1-4, pp. 315–333, 2001.
- [9] 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.
- [10] 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 *Management Science and Engineering (ICMSE), 2013 International Conference on*. IEEE, 2013, pp. 258–262.
- [11] X. Wang, X. Wu, and X. Hu, "A study of urgency vehicle routing disruption management problem," in *Information Engineering (ICIE), 2010 WASE International Conference on*, vol. 3. IEEE, 2010, pp. 452–455.
- [12] X. Wang, X. Wu, Z. Wang, and X. Hu, "A model and an improved genetic algorithm for the vehicle routing problem with break-down vehicles," in *Innovative Computing, Information and Control (ICICIC), 2009 Fourth International Conference on*. IEEE, 2009, pp. 696–699.
- [13] R. R. Pandi, S. G. Ho, S. C. Nagavarapu, T. Tripathy, and J. Dauwels, "GPU-Accelerated tabu search algorithm for the dial-a-ride problem," in *To appear in the Proceedings of the 21st IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2018.
- [14] "Car, taxi and bus - how the costs stack up," <https://www.straitstimes.com/business/invest/car-taxi-and-bus-how-the-costs-stack-up>.