What Makes The Dynamic Capacitated Arc Routing Problem Hard To Solve: Insights From Fitness Landscape Analysis

Hao Tong, Leandro Minku, Stefan Menzel, Bernhard Sendhoff, Xin Yao

2022

Preprint:

What Makes The Dynamic Capacitated Arc Routing Problem Hard To Solve: Insights From Fitness Landscape Analysis

Hao Tong
Leandro L. Minku
School of Computer Science,
University of Birmingham
Birmingham, UK
{hxt922,L.L.Minku}@cs.bham.ac.uk

Stefan Menzel
Bernhard Sendhoff
Honda Research Institute Europe
Offenbach, Germany
{stefan.menzel,bs}@honda-ri.de

Xin Yao*
School of Computer Science,
University of Birmingham, UK
Department of Computer Science and
Engineering, SUSTech, China
xiny@sustech.edu.cn

ABSTRACT

The Capacitated Arc Routing Problem (CARP) aims at assigning vehicles to serve tasks which are located at different arcs in a graph. However, the originally planned routes are easily affected by different dynamic events like newly added tasks. This gives rise to Dynamic CARP (DCARP) instances, which need to be efficiently optimized for new high-quality service plans in a short time. However, it is unknown which dynamic events make DCARP instances especially hard to solve. Therefore, in this paper, we provide an investigation of the influence of different dynamic events on DCARP instances from the perspective of fitness landscape analysis based on a recently proposed hybrid local search (HyLS) algorithm. We generate a large set of DCARP instances based on a variety of dynamic events and analyze the fitness landscape of these instances using several different measures such as fitness correlation length. From the empirical results we conclude that cost-related events have no significant impact on the difficulty of DCARP instances, but instances which require more new vehicles to serve the remaining tasks are harder to solve. These insights improve our understanding of the DCARP instances and pave the way for future work on improving the performance of DCARP algorithms.

KEYWORDS

Fitness Landscape Analysis, Dynamic CARP, Local Search Algorithm, Dynamic Events

ACM Reference Format:
Hao Tong, Leandro L. Minku, Stefan Menzel, Bernhard Sendhoff, and Xin Yao. 2022. What Makes The Dynamic Capacitated Arc Routing Problem Hard To Solve: Insights From Fitness Landscape Analysis . In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

*Corresponding author.

1 INTRODUCTION

The capacitated Arc Routing Problem (CARP) is a traditional combinatorial problem from real-world applications that aims at allocating a number of vehicles with limited capacity to serve a set of tasks in a graph [1, 10]. All vehicles start service from the depot according to the optimized routes and must return to the depot after completing their service. However, the service processes of the vehicles are likely to be influenced by dynamic events. For example, road congestion is likely to deteriorate the quality of the original plan.. Or, newly added tasks which are not in the original plan may be required to be served, such that the current service plan is no longer suitable for new tasks. Therefore, it is essential to consider the Dynamic CARP (DCARP) optimization in an uncertain environment.

In the literature, researchers proposed many algorithms for DCARP optimization considering different dynamic events, including cost change of roads, failure of vehicles, and newly added tasks. DCARP was first investigated in Handa et al. [3], considering dynamic changes of tasks in the salt-gritting scenario. Then, Tagmouti et al. [15] considered time-dependent service costs also in a winter gritting scenario. Monroy et al. [9] proposed rescheduling algorithms to deal with the failure of vehicles. Padungwech et al. [13] investigated the effect of update frequencies in DCARP scenario with newly added tasks. They concluded that the ratio of the number of newly added tasks to the number of all tasks would significantly influence the effect of update frequencies. Liu et al. [6] and Tong et al. [22] considered almost cost-related and task-related dynamic events. Tong et al. [22] proposed an efficient framework to deal with such general DCARP scenarios, enabling static algorithms to be used for DCARP optimization. Besides, Tong et al. [21] also proposed a Hybrid Local Search algorithm (HyLS) to provide better rescheduling within a short time, satisfying the requirement of quick rescheduling usually posed by dynamic scenarios in real-world applications.

Even though there are powerful algorithms for solving general DCARP instances, existing literature lacks information about which dynamic factors are likely to cause DCARP instances harder to solve. This seriously hinders the ability to improve the performance of DCARP algorithms. Therefore, in this paper, we focus on investigating the effect of different dynamic factors on the characteristics of DCARP instances by fitness landscape analysis. Our main goal is to analyze the basic fitness landscape of a DCARP instance and compare the landscape's features between

different DCARP instances produced by various dynamic factors. We provide the following novel contributions:

- The first systematic fitness landscape analysis for DCARP is performed in this paper, including several local and global optima related features. The analysis is based on the hybrid local search algorithm HyLS [21]. Multiple landscape properties are compared between instances generated with different dynamic events to investigate what factors make DCARP instances easier or harder to solve.
- We show that new tasks are the main factor influencing the difficulty of the instances, and instances requiring more new vehicles for the remaining tasks are harder to solve than instances with other features. These insights on the fitness landscape can potentially contribute towards improving the performance of algorithms in future .

The remainder of this paper is organized as follows. Section 2 introduces the related work on DCARP and the background on fitness landscape analysis. Section 3 provides details on the local search algorithm used in our analysis, here, HyLS. Section 4 presents the empirical results and the analysis of fitness landscape from several different perspectives. Section 5 concludes the paper and points out future work.

2 BACKGROUND AND RELATED WORK

2.1 Dynamic capacitated arc routing problem

Dynamic capacitated arc routing problems aim to update the service schedule when the quality of the original schedule deteriorates due to some dynamic events in an uncertain environment. When dynamic events occur, the remaining tasks, the map's information, and the status of all service vehicles at that time compose a DCARP instance. DCARP optimization is mainly targeted to re-optimize the DCARP instance and obtain an updated schedule for the new environment. When dynamic events happen, vehicles are usually located in different points and have served different tasks resulting in different remaining capacities. Therefore, besides CARP requiring the total demand being served by a vehicle not to exceed a vehicle's capacity, a DCARP instance has two additional constraints, i.e., (1) some vehicles are outside and have to start from these outside positions and (2) the remaining capacities of these outside vehicles are different from those of vehicles in the depot.

Suppose a new DCARP instance at time point t_m, and that there are N_T tasks in total, including the previous unserved tasks as well as potentially newly added tasks. Assume there are N_{ov} outside vehicles when the DCARP instance is generated. The stop locations and the remaining capacities of these outside vehicles are represented as $\{v_1, v_2, ..., v_{N_{ov}}\}$ and $\{q_1, q_2, ..., q_{N_{ov}}\}$, respectively. The depot can be denoted as v_0. The maximum number of vehicles and the capacity of each vehicle are denoted as N_{veh} and Q, respectively. A DCARP solution can be represented as $S = \{R_1, R_2, ..., R_{N_{ov}}, ..., R_{N_K}\}$ containing K routes, where R_1 to $R_{N_{ov}}$ denote routes corresponding to outside vehicles and the remaining routes denote the newly deployed vehicles from the depot. Assume the i^{th} task in k^{th} route is $t_{k,i}$ and the demand of a task $t_{k,i}$ is $dm(t_{k,i})$. A route R_k can be represented as $R_k = (v_k, t_{k,1}, t_{k,2}, ..., t_{k,l_k}, v_0)$, where l_k denotes the number of tasks in the k^{th} route and v_k is the starting point of the route R_k. Therefore, the objective function and constraints for DCARP can be formulated by the following equations [22]:

$$\begin{aligned} \textbf{Min} \quad & TC(S) = \sum_{k=1}^{K} RC_{R_k} \\ \textbf{s.t.} \quad & \sum_{k=1}^{K} l_k = N_T \\ & t_{k_1,i_1} \neq t_{k_2,i_2}, \text{for all } (k_1, i_1) \neq (k_2, i_2) \\ & \sum_{i=1}^{l_k} dm(t_{k,i}) \leq q_k, \forall k \in \{1, 2, ...N_{ov}\} \\ & \sum_{i=1}^{l_k} dm(t_{k,i}) \leq Q, \forall k \in \{N_{ov}+1, ..., K\} \end{aligned} \tag{1}$$

where RC_{R_k} is the total cost of route R_k which is calculated by Equation (2):

$$\begin{aligned} RC_{R_k} = & \, mc(v_k, tail_{t_{k,1}}) + mc(head_{t_{k,l_k}}, v_0) + \\ & \sum_{i=1}^{l_k-1} mc(head_{t_{k,i}}, tail_{t_{k,i+1}}) + \sum_{i=1}^{l_k} sc(t_{k,i}) \end{aligned} \tag{2}$$

The $head_{t_{k,i}}$, $tail_{t_{k,i}}$ denote the task's head and tail vertices. The minimal total deadheading cost traversing from node v_i to node v_j is denoted as $mc(v_i, v_j)$, and $sc(t_{k,i})$ denotes the serving cost of task $t_{k,i}$. The first two constraints in Eq. (1) guarantee that all tasks are served only once and the other two constraints are formulated to satisfy the vehicles' capacity constraint.

In the above formulation, factors including the number of tasks, the demand of tasks, and the deadheading/serving cost of arcs will influence the objective function. These factors are all easily affected by the dynamic environment. For example, the cost of arcs will change if a road occurs to be congested, and new tasks are also probably added after the service starts. Therefore, we will investigate the influence of these dynamic factors on the characteristics of DCARP instances in this paper. More details will be introduced in Section 4.

2.2 Fitness landscape analysis

Fitness landscape analysis is a very popular approach for revealing the structure of search space. It helps to better understand the characteristics of problem instances and decide a more suitable algorithm for the specific problem [8]. A fitness landscape is defined by three elements, including a solution set X, a neighborhood structure V, and a fitness function f. A fitness landscape (X, V, f) is determined by solutions' fitness and their neighbors' fitness [4].

Fitness landscape analysis targets to investigate features of optimization problems, such as the fitness distribution in the search space and the structure of optima [8]. In the literature, there are two kinds of techniques proposed for for the characterization of these features, namely the random sampling based technique [7, 8] and the local optimal network based technique [12]. The random sampling based technique performs a local search algorithm in all sampled solutions and obtains a set of local optima

in the search space. Then, analysis metrics are computed based on the sampled solutions or local optima to characterize the target problem. For example, the correlation length technique operates a random walk in the search space and calculates a correlation coefficient based on the sampled solutions to measure the ruggedness of the fitness landscape [25]. In contrast, the local optimal network (LON) technique enumerates all solutions and formulates the fitness landscape as a network [11]. Each node represents a local optimum in the landscape, and each edge represents the basin-transition [24] or escape edges [23] between two local optima. Then, the properties of LON can directly reflect some characteristics of the problem, such as the number of vertices and density of edges in LON. Moreover, network analysis tools can also be used to analyze the fitness landscape. For example, community detection can be used to detect the clustering of local optima [12]. Furthermore, the fitness landscape based LON can be visualized through network visualization tools, helping people to better understand the connectivity between local optima.

The above two kinds of fitness landscape analysis techniques have been successfully applied to some combinatorial problems. Tayarani-N et al. [17] carried out fitness landscape analysis on four combinatorial optimization problems, based on randomly sampled solutions from the search space. After that, they focused on the fitness landscape of eleven different variants of the traveling salesman problem and analyzed their local optima-related features and global optima-related features, among others [18]. The LON technique was also widely used for analyzing the characteristics of combinatorial problems, such as the flow-shop problem [2], quadratic assignment problem [19] and traveling thief problem [26].

Even though it is much easier and more intuitive for users to obtain the analysis of the fitness landscape by the LON technique, most work in the literature required to enumerate all solutions in the search space to construct the LON, which is not suitable in problems with large search spaces. A few studies also proposed the sampled LON algorithms, such as Markov-Chain LON sampling and snowball LON sampling [19], but the metrics which can be used in these approaches are very limited. Therefore, we employed the sampling-based techniques to analyze DCARP instances in this paper so that we can apply a set of useful metrics to our instances, which all have a large search space.

3 HYBRID LOCAL SEARCH ALGORITHM

The fitness landscape is defined specifically for a neighborhood structure such that one problem instance can have different fitness landscapes for different neighborhood structures. In this paper, we focused on a hybrid local search algorithm (HyLS) proposed in [21], which was the best dynamic optimization algorithm for real scenarios, and hope the insights of DCARP instances by fitness landscape analysis can help to improve its performance. Therefore, in this paper, HyLS is used to find the local optimum for each sample in the landscape, and neighborhood moves in the HyLS represent neighborhood functions. The pseudo-code of HyLS is presented in Algorithm 1. The $OneStepMove(\cdot)$ in Line 7 applies one neighborhood move, such as single insertion, to a solution and obtains the best neighbor solution. More details are provided in [21]. The archive's size in this paper is set to 400.

Algorithm 1: The hybrid local search algorithm [21]

```
   Input: An initial solution S_0
   Output: A local optimum LO
1  Initialize the solution archive (set) SA = {S_0};
2  Set local optimal as LO ← S_0;
3  for each solution S_i in SA do
4      Step best solution S_sb ← S_i;
5      while true do
6          for each neighborhood move Move_j do
7              S_mj = OneStepMove_j(S_sb);
8              if improve and SA is not full then
9                  SA ← SA ∪ S_mj;
10             Update step best solution S_sb based on S_mj;
11         if not improve then
12             break;
13     if S_sb.cost < LO.cost then
14         LO ← S_sb;
```

4 FITNESS LANDSCAPE ANALYSIS

4.1 Dynamic factors and instance generation

The deployed CARP solution is likely to be affected by a series of dynamic events, and these dynamic events will generate DCARP instances with different characteristics. To compare the influence of different dynamic events, all DCARP instances are generated from the same initial static CARP instance and the same corresponding initial solution obtained by using a powerful meta-heuristic algorithm [16]. The $egl - E2 - A$ instance was selected as the basic static instance because it is the easiest instance among all static instances in the *egl* dataset, which has more tasks than other datasets [16]. An easier initial instance is likely to make the effect of different settings of dynamic events more obvious.

Four dynamic events which potentially affect the current CARP solution's quality are considered in this paper including ***cost increase***(CI), ***cost decrease***(CD), ***new demand***(ND) and ***new tasks***(NT). The influence of these dynamic events on the CARP solution is as follows:

- *Cost increase*: The cost of arcs belonging to the solution's path increases deteriorating the current solution.
- *Cost decrease*: The current solution's quality can potentially be improved when the cost of arcs which are not in the solution's path decreases.
- *New demand*: New demand in the current tasks potentially makes the current solution infeasible due to limited vehicles' capacities.
- *New tasks*: When there are new tasks, the current plan needs to be rescheduled to accommodate the new tasks.

Each dynamic event corresponds to one or multiple factors. We investigated two levels for each factor in this paper, which are all

presented in Table 1. When a dynamic event happens, a few outside vehicles are located at various positions with different remaining capacities. Thus, each DCARP instance will have two state factors, namely ***number of outside vehicles*** and ***value of remaining capacities for each outside vehicle***, which are also investigated in this paper. Their levels are presented in the bottom of Table 1.

For cost increase/decrease events, the two levels denote that we increase/decrease the costs of 20%(few) and 80%(many) of arcs in/not in the solution's path change. The arcs to suffer the change are selected uniformly at random and their cost is doubled/halved. In the new demand event, we selected uniformly at random 20%(few) and 80%(many) tasks among all remaining tasks to be changed. For the new task event, we also selected uniformly at random 20%(few) and 80%(many) of all available arcs which are not the remaining tasks as the new tasks. The value of new demand and the demand of new tasks are determined by the number of new vehicles required. If the value of new demand is set as small/large, we set that the total demands added to this instance one/four times of vehicle's full capacity, with the new demand added to each task being split equally among all tasks. For new task events, the positions of new tasks are also considered including *far from* and *close to* the depot. In the following subsections, we will use '**0**' and '**1**' to represent the first and second level for each factor.

For simplicity, each dynamic event was investigated independently in our experiment. Each DCARP instance was generated based on one dynamic event. Given a dynamic event, a DCARP instance configuration was created for each possible combination of levels for that dynamic event and levels for the state factors. Due to the page limitation, we put the pseudo-code of the instance generator in the supplementary file. The column "#Settings" in Table 1 presents the number of settings. Therefore, we generated DCARP instances with 64 (i.e., $(2 + 2 + 4 + 8) * 4$) different configurations of dynamic events and state factors. Moreover, 10 different DCARP instances were generated for each configuration in our experiment. The generated instances and the source code of the presented work is available in Github[1].

Table 1: Dynamic factors and state factors investigated in this paper

Dynamic Events	Fators	Acronyms	Levels	#Settings
Cost Increase	Number of arcs	CI-N	[few, many]	2
Cost Decrease	Number of arcs	CD-N	[few, many]	2
New Demand	Number of changed tasks	ND-N	[few, many]	4
	Value of new demand	ND-V	[small, large]	
New Tasks	Postion of new tasks	NT-P	[close, far]	8
	Number of new tasks	NT-N	[few, many]	
	Demand of new tasks	NT-D	[small, large]	

State Factors	Acronyms	Levels	#Settings
Number of outside vehicles	OV	[few, many]	4
Value of remaining capacities	RQ	[small, large]	

[1]Github link to be added

4.2 Auto-correlation

First, we focused on the auto-correlation of DCARP instances. The auto-correlation measures the local ruggedness of the fitness landscape based on random-walk sampling [25]. It calculates the expected correlation between the first $T - \tau$ and last $T - \tau$ subsequences of a fitness sequence $\{c_1, c2, ..., c_T\}$ corresponding to the solutions obtained by the random walk. The auto-correlation is estimated according to:

$$R(\tau) = \frac{\sum_{t=1}^{T-\tau}(c_t - \bar{c})(c_{t+\tau} - \bar{c})}{\sum_{t=1}^{T}(c_t - \bar{c})}$$

where $\bar{c}$ is the mean fitness of the whole sequence. The correlation length l is a single value reflecting the ruggedness of the fitness landscape measuring the auto-correlation when $\tau = 1$ [14] and is calculated by:

$$l = -\frac{1}{lnR(1)}$$

Due to space limits, here, we present the average correlation length over 10 DCARP instances for each setting (Figure 1), where a smaller value indicates a much more rugged landscape.

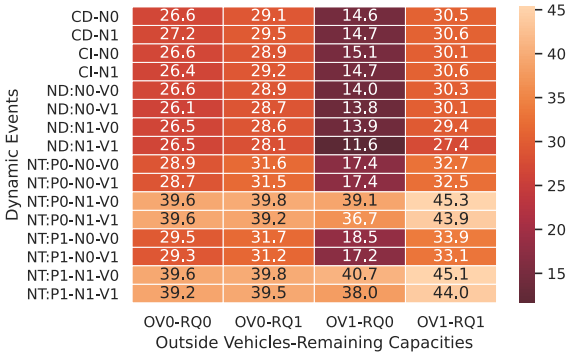

Figure 1: The average correlation length over 10 DCARP instances for each of 64 settings.

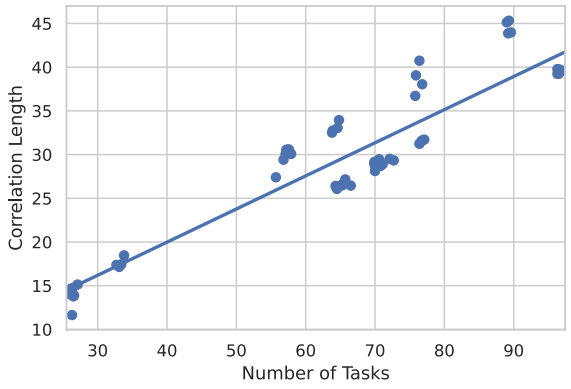

Figure 2: The correlation between correlation length and the number of tasks.

From Figure 1, for state factors, instances with more outside vehicles and fewer remaining capacities (OV1-RQ0) have the smallest correlation length, while the other three have similar correlation lengths to each other. For dynamic events, instances with new tasks have a larger correlation length, especially for settings with more new tasks (NT:P*-N1-V*), while the other three events have similar correlation lengths to each other. The main reason is the number of tasks in each instance. States with more outside vehicles and fewer remaining capacities indicate that many tasks have been served and only a few tasks remain to be served. In contrast, new task events obviously will cause the instance to have a large number of tasks. For instances with more tasks, the total cost is a relatively large value so that the cost change of one-step random walk has less effect on the cost of solutions. Thus, instances with more tasks will have a large correlation length. To further confirm the relationship between correlation length and number of tasks, we plot the correlation length and the number of tasks of all instances in Figure 2 and applied linear regression. It clearly demonstrates that they are approximately linearly correlated. Therefore, the conclusion motivates us that the algorithm designed for DCARP should focus much more on the number of tasks in the DCARP instances.

4.3 Local Optima

In this subsection, we focus on the local optima of the fitness landscape. We randomly sampled 100,000 solutions for each DCARP instance and performed the HyLS on each sampled solution to reach a local optimum. The cost and the solution of all local optima were recorded. The analysis for these local optima is presented in the following.

4.3.1 *Number of Local Optimum*.

Larger numbers of local optima can make the combinatorial optimization problem difficult because the optimization algorithm may be easily trapped in local optima [18]. In a landscape, if two solutions can converge to the same local optimum, the distance between these two solutions must be smaller than their corresponding local optimum's basin size. Therefore, the number of local optima is related to the basin size of the local optima. Thus, we calculate the average number of local optima for each setting Figure 3.

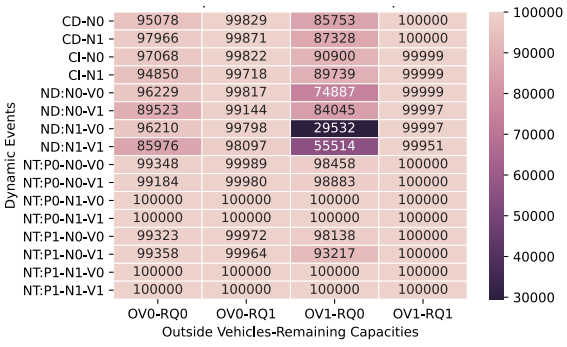

Figure 3: The number of local optima over 10 DCARP instances for each of 64 settings.

From Figure 3, instances with new tasks (NT) have more local optima than the other three dynamic events. In addition, the influence of the state factors is highlighted when the task's number becomes smaller as shown in instances without new tasks (CI, CD, ND) in the figure ($N_T < 70$ in our results). Instances with many outside vehicles and lower remaining capacities(OV1-RQ0) have the fewest local optima, indicating that the basin size of these instances might become large. In Figure 3, instances that only add demand to many existing tasks when there are many outside vehicles and lower remaining capacities(ND:N1-V*, OV1-RQ0) have a significantly smaller number of local optima. Therefore, from the perspective of the number of local optima, the new tasks are also the main factor which make instances much harder to solve. If there is no new task, the adding demand event can make an instance easier to solve.

4.3.2 *Time to reach the local optimum*.

The time for a solution to reach the local optimum reflects the funnel's depth of the corresponding local optimum. If the solution takes a long time to reach the local optimum, it indicates that this local optimum has a deep funnel. A deep funnel usually makes the optimization problem more difficult because the algorithm will take more time to reach the local optimum. Therefore, we recorded the average time to reach the local optimum by HyLS for all 100,000 solutions. The mean results over 10 DCARP instances for each setting are presented in Figure 4.

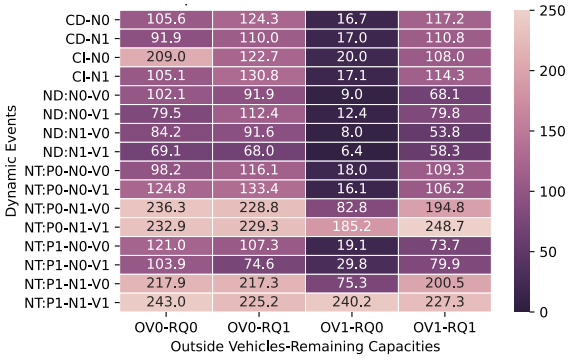

Figure 4: The average time over all 100000 solutions to reach the corresponding local optima (unit: s). The presented result for each configuration is the mean result over 10 DCARP instances.

Instances with a much deeper funnel require algorithms to spend more computational resources for exploitation. For these instances, the number of local optima searched within limited computational resources will be very small, as the algorithm is likely to take a longer time to exploit one local optimum. Therefore, such problem instances are more likely hard to solve. From Figure 4, the new task event is still the main event that influences the problem's difficulty. Specifically, instances with more new tasks take longer time to reach the local optima. The positions and demand value of new tasks have no significant influence on the results.

4.3.3 *Cost distribution*.

The probability of visiting a local optimum is an important metric

to analyze the difficulty of a problem instance. For an easy problem instance, the cost distribution will be more biased to high fitness so that the probability of finding a high-quality local optimum will be much higher. The fitness distribution is capable of well reflecting such characteristics. Therefore, we analyzed the cost distribution based on all recorded local optima. For a much fairer comparison, we normalized the cost of each DCARP instance by its expected cost, which is estimated by sampling a large number of random solutions from the search space.

Two statistical measures are applied to compare the cost distribution between DCARP instances with different configurations, including the mean cost and the skewness. The mean cost reflects the expected quality of a local optimum obtained by HyLS, and the skewness reflects the shape of the distribution compared with a normal distribution. A positive skewness value indicates that the bulk of costs lies to the left of the mean cost. Therefore, a lower mean cost and a higher skewness indicates that the probability of finding a high-quality local optimum is higher. The average values of these two statistics over 10 DCARP instances for each of the 64 settings are presented in Figure 5 and Figure 6.

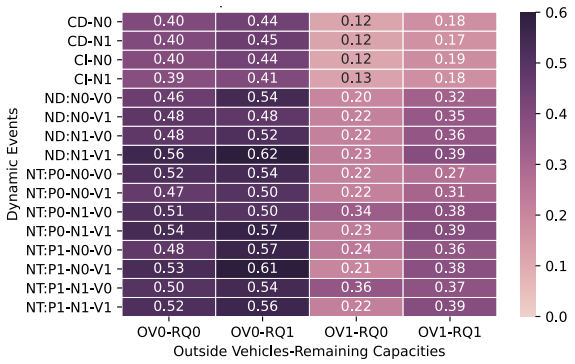

Figure 5: The averaged mean value of all reached local optima for each DCARP setting over 10 DCARP instances.

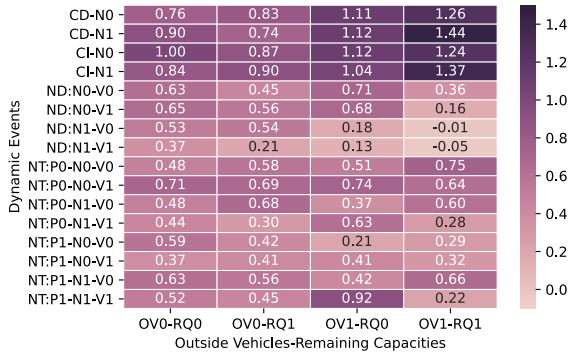

Figure 6: The averaged skewness value of all reached local optima for each DCARP setting over 10 DCARP instances.

From Figure 5, the values in the left two columns are larger than the ones in the right two columns. We conclude that instances with more outside vehicles (OV1) are much easier to solve. For the same dynamic event, instances with many outside vehicles and small remaining capacities (OV1-RQ0) are the easiest because the number of tasks is the smallest. Moreover, for the same state factor, adding new tasks makes instances harder than the other three events. The events of cost increase and decrease are similar to each other indicating that instances with only these two events will not become very difficult. The skewness presented in Figure 6 is able to reflect the probability of finding a local optimum in an instance. A much larger value indicates it has a high probability for finding a high-quality solution. From Figure 6, almost all instances have a positive skewness value, indicating that the quality of most reached local optima is much better than a random solution. This is also helpful for the effectiveness of HyLS. Specifically, the distribution of instances only with cost change events is more biased to the area with high-quality solutions meaning that an algorithm may find a good solution with fewer computing resources.

In summary, according to the mean cost and the skewness of cost distribution, we can conclude that cost change events will not make DCARP instances are harder to solve compared with new demand and new tasks events. A higher number of outside vehicles will also result in a higher mean quality of local optima to the DCARP instances.

4.3.4 *Distance between local optima.*

The distance between local optima is also an important property of the fitness landscape. If a local optimum with a low cost is far from other local optima, it will be hard for the algorithm to explore from a local optimal to a much better local optimum which makes the problem hard to optimize. Therefore, we analyzed the following two distance properties:

(1) The distance between local optima versus expected distance between two random solutions.
(2) The correlation between the cost of local optima and their distance.

Unlike for continuous optimization problems, the Euclidean distance is not suitable for the CARP space. We applied Hamming distance to measure the distance between two CARP solutions. Suppose V_R is a set of vertices including the depot and all vertices incident with at least one task, a complete graph $G_R = (V_R, A_R)$ can be constructed, where A_R is the set of arcs between each pair of vertices in V_R. If one vertex i incidents with $d(i)$ tasks ($d(i) > 0$), we will duplicate $d(i)$ copies of this vertex in V_R and assign a unique id to each copy. The cost of an arc in A_R depends on the original (D)CARP instance. Therefore, all traversed paths in a CARP solution are arcs in G_R, and no arc in G_R will be used twice in a CARP solution. Consequently, a CARP solution can be represented by a binary vector in the space of arcs in G_R, where each bit denotes one arc in G_R and "1" denotes that the corresponding arc is used in the CARP solution. Thus, solutions will be converted to the binary representation first before calculating the distance.

The ratios of the expected distance between two local optima to that of two random solutions for all 64 settings are presented in Figure 7, where each result is the averaged value over 10 DCARP instances. A negative value indicates that the distance between two local optima is smaller than two random solutions. In Figure 7,

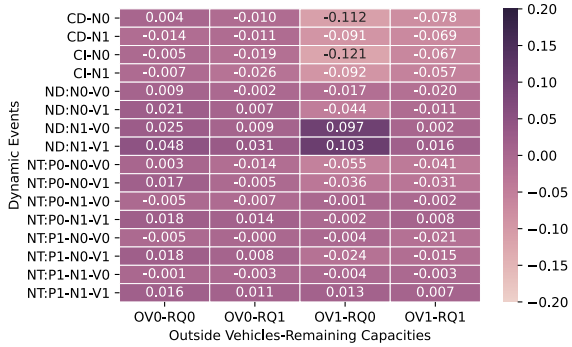

Figure 7: The ratio of the expected distance between two local optima to that of two random solutions. The presented result for each configuration is the averaged value over 10 DCARP instances.

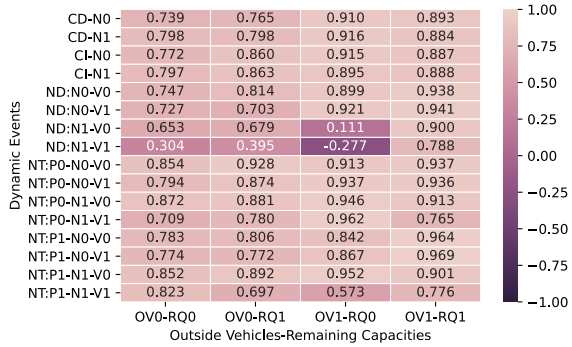

Figure 8: The correlation between the cost of local optima and their distance. The presented results for each configuration are the averaged values over 10 DCARP instances.

almost all ratios are very close to 0, reflecting that local optima in all DCARP instances are uniformly distributed instead of gathering in a specific area or far from each other. We conclude that all dynamic factors will not impact the distribution of the local optima, not further influencing the the difficulty of each instance.

The correlation between the cost of local optima and their distance for all 64 settings are presented in Figure 8, where each result is also the averaged value over 10 DCARP instances. From Figure 8, almost all correlations between the cost of local optima and their distance are close to 1. It indicates that better local optima are much closer to each other than the less fit local optima, which is beneficial to the optimization. Moreover, instances with more outside vehicles (OV1) have larger correlation values according to Figure 8 whereas the correlations for different dynamic factors under the same state factor are similar to each other. Therefore, instances with more outside vehicles might be slightly easier for optimization. Instances with adding demands to many tasks (ND:N1-V*) have the smallest correlation. This indicates that the distribution of local optima is not clustered based on their cost. Thus, the algorithm might require a bigger step for the exploration.

4.4 Global Optimum

In this subsection, we focus on the properties of the global optimum of the problem instance. As we cannot enumerate all samples in the search space in our DCARP instances, the real global optimum is unknown. Therefore, we applied a powerful meta-heuristic, i.e., Memetic Algorithm with Extended Neighborhood Search (MAENS) [16, 20], to optimize all DCARP instances with sufficient computational resources, i.e., with the same maximum iteration number as used in the static optimization [16]. The obtained best solution is then regarded the as the approximate global optimum.

4.4.1 *Fitness Distance Correlation.*

Fitness Distance Correlation (FDC) is also a useful measurement to characterize the problem difficulty [4, 5]. It measures the correlation of a solution's fitness and its distance to a global optimum. In minimization problems, a positive correlation indicates that the closer a solution distance is to the global optimum, the higher quality (i.e., lower cost) the solution has. The algorithm can benefit from such a fitness landscape and be more easily guided to the global optimum. In contrast, an instance with a negative correlation will potentially guide the algorithm towards the opposite direction of the global optimum. For DCARP instances, suppose $F = \{c_1, c_2, ..., c_n\}$ and $D = \{d_1, d_2, ..., d_n\}$ are the fitnesses of n solutions and the distances to the global optimum of these solutions. FDC is calculated by the following equation [4]:

$$FDC = \frac{\frac{1}{n}\sum_{i=1}^{n}(c_i - \bar{c})(d_i - \bar{d})}{\sigma_F \sigma_D}$$

where $\sigma_F, \sigma_D, \bar{c}, \bar{d}$ represent the standard deviations and means of F and D. The distance between two solutions are also measured by Hamming distance as introduced in Section 4.3.4. The mean FDCs over 10 DCARP of each configuration is presented in Figure 9.

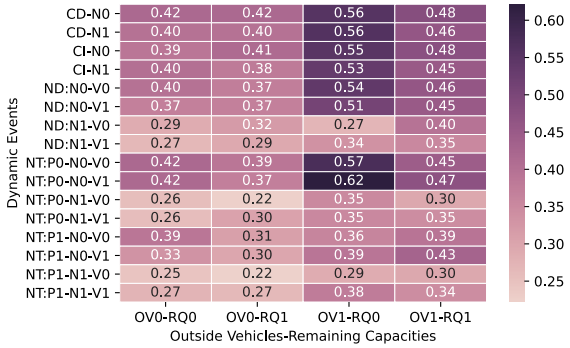

Figure 9: The fitness distance correlation of global optimum for all 64 settings. The presented result for each configuration is the mean result over 10 DCARP instances.

The high value in Figure 9 indicates that local optima that are closer to the global optima have higher quality, meaning that the instance is potentially easier to solve. In Figure 9, instances with new tasks (NT) have a relatively small FDC value compared with the other three events. For state factors, FDC values of instances with fewer outside vehicles (OV0) are also smaller than those with many

outside vehicles (OV1). We also observe that instances with adding demand to many tasks (ND:N1-V*) also have a small FDC value. These instances all require more vehicles to serve the remaining tasks. Therefore, an interesting potential conclusion is that when remaining tasks require more new vehicles, the instance will have a smaller FDC value, and the problem instance might become harder to solve.

4.4.2 *Return Probability*.

The final characteristic we investigate is the return probability of the global optimum. It is a property that empirically measures the basin of attraction of the global optimum by calculating the probability of returning to the global optimum starting from a solution with a fixed distance to it [18]. If the global optimum has a high return probability even though the starting solution is far from the global optimum, the basin of attraction will be large. As a result, it may be easier for the algorithm to find the global optimum, as the global optimum's basin of attraction is very large.

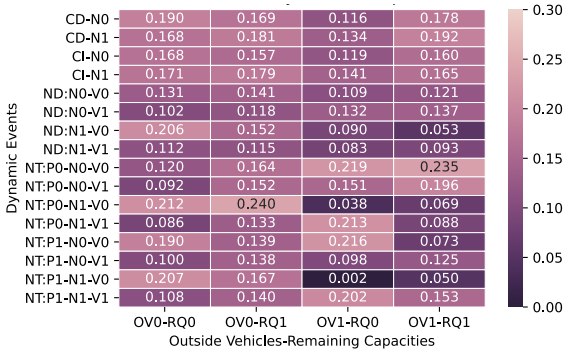

Figure 10: The return probability of global optimal with one random step of *Single Insertion*.

In our experiment, the return probability is measured by starting from solutions which are obtained by applying with a given number of *Single Insertion* operations. The *Single Insertion* is the most basic move for the CARP solution that move one task from one route to another position of this or another route in the solution. The return probability is estimated by starting from 1000 solutions generated by applying the same number of steps of *Single Insertion* from the global optimum. From our results, the return probability in all DCARP instances drops to 0 after about 6 steps of *Single Insertion*. For simplicity, we only present the return probability of the global optimum after one random step of *Single Insertion* averaged over 10 instances in Figure 10.

From Figure 10, the return probability for one-step neighborhood move was very small for all DCARP instances, which indicates that the global optimum's basin size is very small for the DCARP problem. Therefore, the DCARP instances are hard to solve, and a good algorithm should have a good exploration ability to find a high-quality solution. In addition, the bottom right part of Figure 10 has a relatively smaller return probability compared with other parts. In this area, instances have more remaining demands to be served and many outside vehicles. Therefore, the dynamic events causing more demands and the state with more outside vehicles are likely more challenging to solve.

4.5 Summary and Discussion

In this section, we performed fitness landscape analysis on a set of DCARP instances with different settings of dynamic events and state factors. A list of valuable conclusions are obtained from the empirical analysis and summarized as follows:

- DCARP instances with many remaining tasks and new tasks caused by the new task event had a smoother fitness landscape.
- The number of tasks was also the main factor affecting the number of local optima and the averaged time to reach the local optima. Instances with a larger number of tasks caused more local optima and consequently a longer time to reach the local optima.
- The local optimum's mean cost of DCARP instances was mainly affected by the number of outside vehicles and the number of tasks. Instances with more outside vehicles and fewer tasks had a relatively lower mean cost.
- The local optima in all DCARP instances were almost uniformly distributed. Their distances were similar to that of two random solutions.
- Local optima with higher quality were closer to each other than to less fit local optima for all DCARP instances.
- In terms of the global optimum, instances which require more vehicles for the remaining tasks had a smaller fitness distance correlation as well as a smaller return probability.

According to our insights on the fitness landscape analysis for DCARP instances, a potential direction about designing more effective algorithms for optimizing DCARP is to pay more attention to the new tasks and the total remaining demands. For example, for an instance being generated by new tasks, we can focus on designing strategies to allocate new tasks.

5 CONCLUSION

In this paper, we focused on Dynamic Capacitated Arc Routing Problems (DCARP) and investigated which factor(s) potentially make the DCARP instances hard to solve. We considered four common dynamic events: cost increase, cost decrease, new demand, and new tasks. We also considered two inherent state factors of the DCARP instance, i.e., the number of outside vehicles and the remaining capacities. Fitness landscape analysis techniques based on a local search algorithm, i.e., HyLS, were performed on a set of generated DCARP instances according to different configurations of dynamic and state factors. We investigated the autocorrelation, the number of local optima, the time to reach a local optimum, the mean cost of the local optima, the distance properties of local optima, the fitness distance correlation, and the return probability on all generated DCARP instances.

Based on the empirical results from the fitness landscape analysis, we found that the new task event had much more influence on the difficulty of the DCARP instances than the other three dynamic events. The number of remaining tasks was the main factor making DCARP instances hard to solve, with a larger number being harder.

There was no evidence that the cost increase and cost decrease significantly impacted the characteristics of the DCARP instances. In addition, our fitness distance correlation analysis found that instances requiring more new vehicles are likely more difficult to solve.

As new task events and the requirements for new vehicles will potentially make DCARP instances harder to solve, future work could focus on designing new algorithms to better tackle these two key factors.

REFERENCES

[1] Ángel Corberán, Richard Eglese, Geir Hasle, Isaac Plana, and José María Sanchis. 2020. Arc routing problems: A review of the past, present, and future. *Networks* (June 2020).

[2] Fabio Daolio, Sébastien Verel, Gabriela Ochoa, and Marco Tomassini. 2013. Local optima networks of the permutation flow-shop problem. In *International Conference on Artificial Evolution (Evolution Artificielle)*. Springer, 41–52.

[3] Hisashi Handa, Lee Chapman, and Xin Yao. 2005. Dynamic salting route optimisation using evolutionary computation. In *2005 IEEE Congress on Evolutionary Computation*, Vol. 1. IEEE, 158–165.

[4] Terry Jones et al. 1995. *Evolutionary algorithms, fitness landscapes and search*. Ph. D. Dissertation. Citeseer.

[5] Terry Jones, Stephanie Forrest, et al. 1995. Fitness Distance Correlation as a Measure of Problem Difficulty for Genetic Algorithms.. In *ICGA*, Vol. 95. 184–192.

[6] Min Liu, Hemant Kumar Singh, and Tapabrata Ray. 2014. A memetic algorithm with a new split scheme for solving dynamic capacitated arc routing problems. In *2014 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 595–602.

[7] Katherine Mary Malan. 2021. A survey of advances in landscape analysis for optimisation. *Algorithms* 14, 2 (2021), 40.

[8] Katherine M Malan and Andries P Engelbrecht. 2013. A survey of techniques for characterising fitness landscapes and some possible ways forward. *Information Sciences* 241 (2013), 148–163.

[9] Marcela Monroy-Licht, Ciro Alberto Amaya, André Langevin, and Louis-Martin Rousseau. 2017. The rescheduling arc routing problem. *International Transactions in Operational Research* 24, 6 (2017), 1325–1346.

[10] M. Cândida Mourão and Leonor S. Pinto. 2017. An updated annotated bibliography on arc routing problems. *Networks* 70, 3 (Aug. 2017), 144–194.

[11] Gabriela Ochoa, Marco Tomassini, Sebástien Vérel, and Christian Darabos. 2008. A study of NK landscapes' basins and local optima networks. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*. 555–562.

[12] Gabriela Ochoa, Sébastien Verel, Fabio Daolio, and Marco Tomassini. 2014. Local optima networks: A new model of combinatorial fitness landscapes. In *Recent advances in the theory and application of fitness landscapes*. Springer, 233–262.

[13] Wasin Padungwech, Jonathan Thompson, and Rhyd Lewis. 2020. Effects of update frequencies in a dynamic capacitated arc routing problem. *Networks* 76, 4 (2020), 522–538.

[14] Peter F Stadler et al. 1995. Towards a theory of landscapes. In *Complex systems and binary networks*. Springer, 78–163.

[15] Mariam Tagmouti, Michel Gendreau, and Jean-Yves Potvin. 2011. A dynamic capacitated arc routing problem with time-dependent service costs. *Transportation Research Part C: Emerging Technologies* 19, 1 (2011), 20–28.

[16] Ke Tang, Yi Mei, and Xin Yao. 2009. Memetic algorithm with extended neighborhood search for capacitated arc routing problems. *IEEE Transactions on Evolutionary Computation* 13, 5 (2009), 1151–1166.

[17] Mohammad-H Tayarani-N and Adam Prügel-Bennett. 2013. On the landscape of combinatorial optimization problems. *IEEE Transactions on Evolutionary Computation* 18, 3 (2013), 420–434.

[18] Mohammad-H Tayarani-N and Adam Prügel-Bennett. 2016. An analysis of the fitness landscape of travelling salesman problem. *Evolutionary computation* 24, 2 (2016), 347–384.

[19] Sarah L Thomson, Gabriela Ochoa, Sébastien Verel, and Nadarajen Veerapen. 2020. Inferring future landscapes: sampling the local optima level. *Evolutionary computation* 28, 4 (2020), 621–641.

[20] Hao Tong, Leandro L Minku, Stefan Menzel, Bernhard Sendhoff, and Xin Yao. 2020. Towards Novel Meta-heuristic Algorithms for Dynamic Capacitated Arc Routing Problems. In *International Conference on Parallel Problem Solving from Nature*. Springer, 428–440.

[21] Hao Tong, Leandro L Minku, Stefan Menzel, Bernhard Sendhoff, and Xin Yao. 2021. A hybrid local search framework for the dynamic capacitated arc routing problem. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 139–140.

[22] Hao Tong, Leandro L. Minku, Stefan Menzel, Bernhard Sendhoff, and Xin Yao. 2021. A Novel Generalised Meta-Heuristic Framework for Dynamic Capacitated Arc Routing Problems. arXiv:2104.06585

[23] Sébastien Verel, Fabio Daolio, Gabriela Ochoa, and Marco Tomassini. 2011. Local optima networks with escape edges. In *International Conference on Artificial Evolution (Evolution Artificielle)*. Springer, 49–60.

[24] Sébastien Verel, Gabriela Ochoa, and Marco Tomassini. 2010. Local optima networks of NK landscapes with neutrality. *IEEE Transactions on Evolutionary Computation* 15, 6 (2010), 783–797.

[25] Edward Weinberger. 1990. Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biological cybernetics* 63, 5 (1990), 325–336.

[26] Mohamed El Yafrani, Marcella SR Martins, Mehdi El Krari, Markus Wagner, Myriam RBS Delgado, Belaïd Ahiod, and Ricardo Lüders. 2018. A fitness landscape analysis of the travelling thief problem. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 277–284.