Supplementary Material of "Learning Heuristics with Different Representations for Stochastic Routing"

Ya-Hui Jia, *Member, IEEE,* Yi Mei, *Senior Member, IEEE,* and Mengjie Zhang, *Fellow, IEEE*

TABLE A
EDGE COST AND VERTEX DEGREE OF THE INSTANCE.

Instance	Edge Cost	Vertex Degree
Ugdb1	11.43±6.45	3.50±0.67
Ugdb2	11.16±6.07	4.17±0.83
Ugdb3	10.52±6.15	3.50±0.90
Ugdb4	12.56±6.24	3.27±0.65
Ugdb5	11.88±6.45	3.85±0.69
Ugdb6	11.48±6.79	3.50±0.90
Ugdb7	11.90±6.47	3.50±1.09
Ugdb8	4.60±2.36	3.33±1.49
Ugdb9	4.32±2.40	3.70±1.46
Ugdb10	10.21±5.06	4.00±1.41
Ugdb11	7.75±5.53	4.00±1.35
Ugdb12	14.64±7.44	3.38±0.87
Ugdb13	18.11±20.65	5.40±1.65
Ugdb14	4.65±2.28	5.71±0.49
Ugdb15	2.50±1.36	5.71±0.49
Ugdb16	4.15±2.25	6.75±0.46
Ugdb17	2.85±1.61	6.75±0.46
Ugdb18	4.49±2.24	7.78±0.44
Ugdb19	3.90±2.64	2.50±1.41
Ugdb20	4.90±2.59	3.82±1.99
Ugdb21	4.44±2.61	5.82±1.72
Ugdb22	4.35±2.85	7.82±1.47
Ugdb23	4.06±2.33	9.82±0.40

I. PROBLEM COMPLEXITY ANALYSIS

After the comparison, we gain an insight into the circumstances under which the three representations suit for. First, except for the number of vertices, edges, and vehicles, we measure the tested instances from the other two aspects: edge cost and vertex degree in order to show the characteristics of the instances. The values are shown in the form of 'mean±std'.

Checking the results shown in Fig. 4 of the paper with the reference of Table A, we can find that:

- There are two circumstances under which the linear representation is powerful enough. 1) The road network is sparsely connected where the mean degree of vertex is smaller than 3 like Ugdb19. 2) The network is densely connected where the mean degree of vertex is larger than 5 and the variance of the edge cost is not very large, such as Ugdb14-Ugdb18 and Ugdb21-Ugdb23. For the former case, the priorities of different tasks vary greatly that even the linear representation can tell them apart easily since there are not so many good choices. A vehicle can basically choose the nearest task because choosing a task that is far away from the vehicle would cause a long distance detour. For the later case, vehicles always have many good candidate tasks to choose and choosing different tasks will not affect the final objective greatly.
- When the road network is moderately connected or the variance of the edge cost is large, the complexity of the problem is high and more complex representations like the tree representation are required. For a moderately connected network, a vehicle will have several good candidate tasks to choose each time and different choices will affect subsequent choices thus affecting the final objective value. When the variance of the edge cost is large, tasks need to be allocated carefully since different partitions of the tasks will lead to very different objective values and this is beyond the ability of the linear representation.
- Currently, only the applicable scope of ANN is a little unclear. It is definitely effective on the simple instances. However, for the complex situations, ANN is only effective on some of them. After analyzing the results and the network shape, we found that when the road network does not have neat structure (like clustered, symmetric, etc.), the ANN representation tends to degrade to the linear representation. Although this description is blur that we did not find a specific numeric measurement to indicate the applicable scope of ANN, the above observation gives us a direction to understand how the routing policy works and why they are different.

Due to the page limit, we have made detailed case study in the supplementary material to verify the above analysis. In summary, through increasing the complexity of the representation from linear to ANN, the performance of the heuristic can be improved. Assuming that for each decision situation there is an effective linear heuristic, the phenomenon of ANN degrading to a specific linear representation on some complex instance tells us that the ability of the ANN representation is not good enough to cover different decision situations. This may enlighten us to design more complex or flexible representations by deliberately partition the whole process into different decision situations.

II. CASE STUDY

In the paper, we have made sufficient quantitative analyses about the performance of the three representations. However, we have not got enough insight about under what circumstance we should use simple representations like linear representation or complex representations like the tree representation. To achieve clearer guidelines than quantitative analyses only, we

choose four representative cases {Ugdb5, Ugdb12, Ugdb17, Ugdb19} to further analyze the routes generated by the tree representation heuristic and the linear representation heuristic. For each representation, the best heuristic among the 30 runs is chosen. According to the results shown in Fig. 5 of the paper, on Ugdb5 and Ugdb7, the tree representation heuristic achieved better than the linear representation. On Ugdb17 and Ugdb19, they two achieved similar results. The layouts of the four instances and the routes generated by the linear representation and tree representation heuristics are shown in Fig. A.

First, we analyze Fig. A(a) and (b) to see why the routes generated by the tree representation heuristic are much better than the linear representation heuristic. From Fig. A(a) we can see that the tree representation heuristic left the task (1,12) to the end of route 1 so that it can serve the tasks far away from the depot without violating the capacity constraint. If task (1, 12) is served in the first place like the linear representation heuristic did, the remaining capacity of the vehicle will be insufficient to handle the task (7,12) that is slightly farther than (1,12) from the depot. Fig. A(b) shows a complicated situation that except the route 1, the linear representation heuristic assigned the tasks in a very different way compared with the tree representation heuristic, which leads to higher cost. From Fig. A(a) and (b), we can see that Ugdb5 and Ugdb12 shares some similarities in terms of the layout of the road network. 1) The connectivity of the network is at a moderate level that is neither sparse nor dense. 2) It is very hard to decompose the networks into different routes manually.

Second, we check Ugdb17 and Ugdb19 corresponding to Fig. A(c) and (d) to see why even the linear representation is effective on these two instances. On Ugdb17, i.e. Fig. A, we can see that the connectivity of the network is very dense that basically a vehicle can serve tasks one by one without generating any deadheading cost (traversing without serving). Both the linear representation heuristic and the tree representation heuristic have left only one task to serve with refilling. Since the tasks (2,5) and (2,4) share the same cost, the total costs of the routes generated by the two heuristics are identical. In contrast with Fig. A(c), the network in Fig. A(d) is relatively sparse and we can even manually decompose the network into different areas and routes. Under this circumstance, the linear representation heuristic and the tree representation heuristic have generated identical routes.

Comparing the situations where the tree representation outperformed and did not outperform the linear representation, we can get some guidelines that:

- When the road network is densely connected and the vehicle is capable enough to serve the tasks assigned to it without refilling, or when the road network is sparsely connected and easy to be decomposed, a linear representation is good enough to provide similar performance to the tree representation.
- When the road network is moderately connected and hard to be decomposed, the linear representation is not recommended since the tree representation is more flexible, and thus more capable of generating significantly better routes.

III. Attribute Analysis

Corresponding to the analysis of the routes, we analyze the attribute frequencies and weights to see whether there is a clear relationship between the usage of the attributes and the complexity of the instances. The frequencies of attributes appearing in the tree representation heuristic and the weights of attributes in the linear representation heuristic are shown in Fig. B. The meaning of each feature can be found in Table I of the paper. For the sake of clarity, the absolute values of the weights are shown.

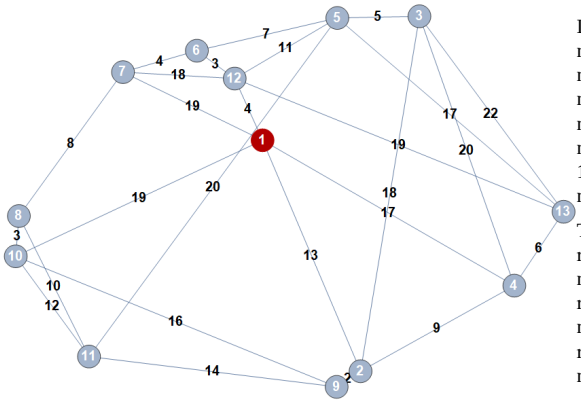
From Fig. B(a) we can see that generally, CFH and DEM generally have larger frequencies than the other attributes, but among these four instances, Ugdb17 shows an interesting pattern that CFH does not have a large proportion. This reason is related to the layout of the network. Fig. A(c) shows that the vehicle only needs to serve the successive tasks one by one without deadheading traversing for Ugdb17. Thus, in most cases, CFH is useless since the candidate tasks are just around the vehicle. Another confusing phenomenon is that the frequencies of the type2 attributes {FULL, FRT, FUT, RQ, CR} on Ugdb17 and Ugdb19 are not lower compared with that on Ugdb5 and Ugdb12. Since the linear representation on Ugdb17 and Ugdb19 shows competitive performance to the tree representation, the type2 attributes on Ugdb17 and Ugdb19 should not be important. The results indicate that although GP has a certain ability to select different attributes for different instances during the evolutionary process, it is not a good sign to use the frequencies of the type2 attributes as the measurement of the problem complexity.

From Fig. B(b) we can see that CFH is still the most important attributes for Ugdb5 and Ugdb12, but for the linear representation, DEM (a demand of a customer) seems not that important compared with its frequency in tree representation heuristics. DEM is highly related to the fullness of the vehicle FULL. The tree representation heuristic can use these two attributes to estimate the customer sequence that can utilize the capacity of the vehicle as much as possible without violating the constraint. However, for the linear representation, FULL is useless as a type 2 attribute, which has a big influence to the importance of DEM. Fig. B(b) also shows that the linear representation may have the ability to undertake the attribute selection or heuristic explanation job since the weights of several attributes are close to zero on some instances, which indicates that they may be useless for the instances.

Overall, although the results show that the attributes have different importance on different instances, we have not found a clear pattern of attribute importance that can be used to indicate the complexity of the problem, but we do find that there is a potential to use the linear representation to select attributes or explain more complex heuristics.

IV. Theoretical Analysis of the Requirement of Training Samples

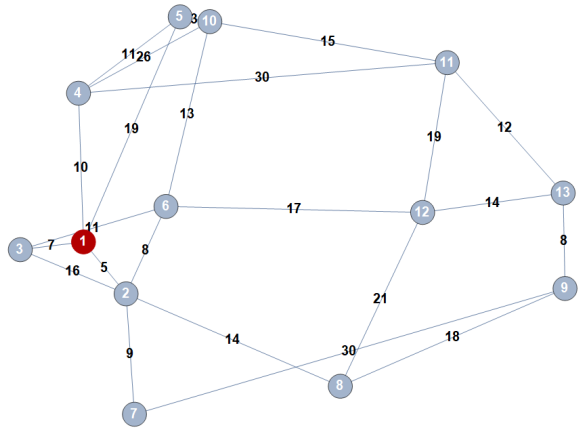
In this section, we analyze how many training samples are enough theoretically. From a high-level perspective, the hyper-heuristic algorithm considered in this paper also belongs to automatic algorithm design. There are already several works



Linear Route (cost 424) :
route 0: 1 ⇒ 10 ⇒ 11 ⇒ 5 → 6 ⇒ 12 → 1
route 1: 1 ⇒ 2 ⇒ 3 ⇒ 4 → 1
route 2: 1 → 12 ⇒ 13 ⇒ 3 ⇒ 5 ⇒ 12 → 1
route 3: 1 ⇒ 7 → 8 ⇒ 10 ⇒ 9 → 2 ⇒ 4 → 1
route 4: 1 ⇒ 12 → 6 → 7 ⇒ 8 ⇒ 11 ⇒ 9 ⇒ 2 → 1 → 12 ⇒ 7 → 6 → 12 → 1
route 5: 1 ⇒ 4 ⇒ 13 ⇒ 5 ⇒ 6 ⇒ 7 → 6 → 12 → 1

Tree Route (cost 402) :
route 0: 1 → 12 ⇒ 13 ⇒ 3 ⇒ 5 → 6 ⇒ 7→ 6→12 → 1
route 1: 1 ⇒ 4 ⇒ 13 ⇒ 5 ⇒ 6 ⇒ 12 → 1 ⇒ 12 → 1
route 2: 1 → 12 ⇒ 7 ⇒ 8 ⇒ 10 ⇒ 9 → 2 → 1
route 3: 1 ⇒ 2 ⇒ 4 ⇒ 3 ⇒ 2 → 1
route 4: 1 → 12 ⇒ 5 ⇒ 11 ⇒ 8 → 7 ⇒ 1
route 5: 1 ⇒ 10 ⇒ 11 ⇒ 9 ⇒ 2 → 1

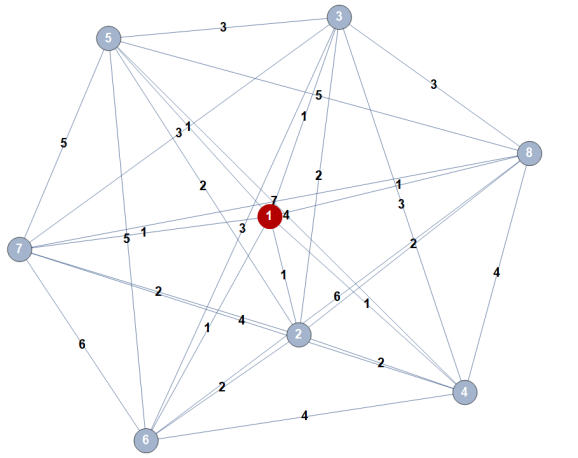
(a)



Linear Route (cost 585) :
route 0: 1 ⇒ 5 → 10 ⇒ 11 ⇒ 13 → 9 → 7 ⇒ 2 → 1
route 1: 1 → 4 ⇒ 10 ⇒ 6 ⇒ 2 → 1
route 2: 1 ⇒ 2 ⇒ 3 ⇒ 1 ⇒ 4 ⇒ 5 → 1
route 3: 1 → 2 ⇒ 8 ⇒ 9 ⇒ 13 → 11 → 10 ⇒ 5 → 1
route 4: 1 → 4 ⇒ 11 ⇒ 12 → 6 → 2 → 1
route 5: 1 → 3 ⇒ 6 ⇒ 12 ⇒ 8 → 2 → 1
route 6: 1 → 2 → 7 ⇒ 9 → 13 ⇒ 12 → 6 → 2 → 1

Tree Route (cost 564) :
route 0: 1 → 2 → 6 ⇒ 12 ⇒ 13 ⇒ 11 → 10 → 5 → 1
route 1: 1 → 2 ⇒ 8 ⇒ 12 ⇒ 11 → 10 → 5 → 1
route 2: 1 → 3 ⇒ 1 → 4 ⇒ 1 → 2 ⇒ 6 ⇒ 3 → 1
route 3: 1 → 2 ⇒ 7 ⇒ 9 ⇒ 13 → 9 ⇒ 8 → 2 → 1
route 4: 1 → 4 ⇒ 5 ⇒ 10 → 5 ⇒ 1 → 2 ⇒ 3 → 1
route 5: 1 → 4 ⇒ 11 ⇒ 10 → 5 → 1
route 6: 1 → 4 ⇒ 10 ⇒ 6 → 2 ⇒ 1

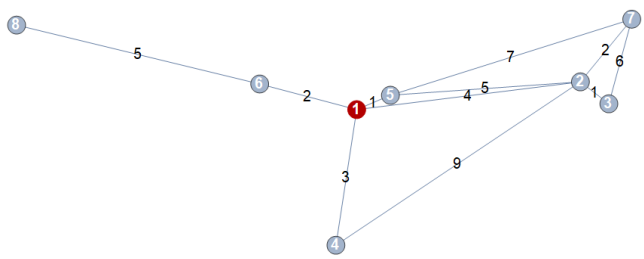
(b)



Linear Route (cost 91) :
route 0: 1 ⇒ 2 ⇒ 3 ⇒ 5 ⇒ 8 ⇒ 4 → 1 → 2 ⇒ 5 → 1
route 1: 1 ⇒ 4 ⇒ 5 ⇒ 7 ⇒ 4 ⇒ 2 ⇒ 6 → 1
route 2: 1 ⇒ 6 ⇒ 7 ⇒ 8 ⇒ 3 → 1
route 3: 1 ⇒ 5 ⇒ 6 ⇒ 8 ⇒ 2 ⇒ 7 → 1
route 4: 1 ⇒ 3 ⇒ 4 ⇒ 6 ⇒ 3 ⇒ 7 ⇒ 1 ⇒ 8 → 1

Tree Route (cost 91) :
route 0: 1 ⇒ 2 ⇒ 8 ⇒ 1 ⇒ 7 ⇒ 4 ⇒ 6 ⇒ 7 → 1
route 1: 1 ⇒ 4 ⇒ 8 ⇒ 6 ⇒ 3 → 1 → 2 ⇒ 4 → 1
route 2: 1 ⇒ 6 ⇒ 2 ⇒ 7 ⇒ 5 ⇒ 6 → 1
route 3: 1 ⇒ 5 ⇒ 2 ⇒ 3 ⇒ 7 ⇒ 8 → 1
route 4: 1 ⇒ 3 ⇒ 8 ⇒ 5 ⇒ 3 ⇒ 4 ⇒ 5 → 1

(c)

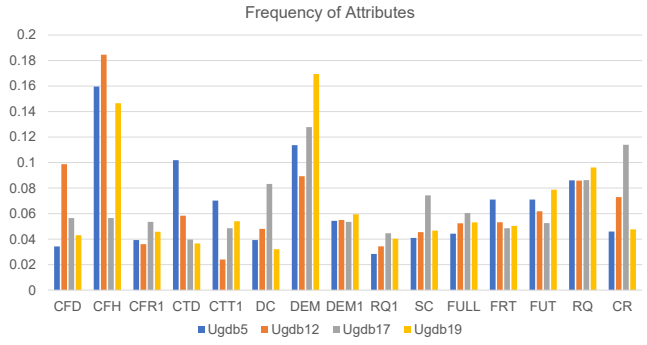


Linear Route (cost 57) :
route 0: 1 → 5 ⇒ 2 ⇒ 3 ⇒ 7 ⇒ 5 → 1
route 1: 1 → 6 ⇒ 8 → 6 ⇒ 1 ⇒ 5 → 1
route 2: 1 ⇒ 4 ⇒ 2 ⇒ 7 → 2 ⇒ 1

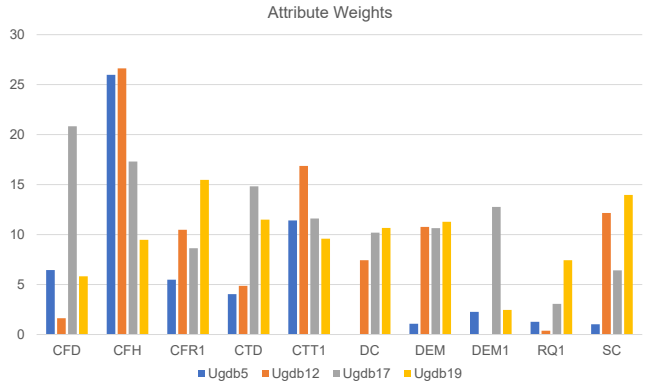
Tree Route (cost 57) :
route 0: 1 → 5 ⇒ 2 ⇒ 3 ⇒ 7 ⇒ 5 → 1
route 1: 1 ⇒ 4 ⇒ 2 ⇒ 7 → 2 ⇒ 1
route 2: 1 → 6 ⇒ 8 → 6 ⇒ 1 ⇒ 5 → 1

(d)

Fig. A. Layout of the instances and the routes generated by the linear representation heuristic and the tree representation heuristic. → means traversing without serving. ⇒ means serving.



(a)



(b)

Fig. B. Attribute frequencies of the tree representation heuristic and attribute weights of the linear representation.

that have made some theoretical analyses about how to allocate computational load to achieve better performance [1], [2]. Here, we start from the general framework of the analysis of automatic algorithm design to the specific hyper-heuristic algorithm that we considered in this paper and finally settle in stochastic problems.

First, assume there is a sample space I where a specific sample i can be selected/sampled with the probability of $P_I(i)$ and a generated algorithm that can at a certain probability $P_C(c_{ij}|i)$ give a solution with fitness value c_{ij} on the instance i. Then, for a scenario (c_{ij}, i) of the generated algorithm, we have the joint probability $P(c_{ij}, i) = P_C(c_{ij}|i)P_I(i)$. Thus, the expected value of the fitness value of the generated algorithm with respect to the scenario $P(c_{ij}, i) = P_C(c_{ij}|i)P_I(i)$:

$$\mu = E[c_{ij}] = \iint c_{ij} \mathrm{d}P_C(c_{ij}|i)\mathrm{d}P_I(i). \tag{1}$$

For the hyper-heuristic algorithm considered in this paper, since the generated algorithm is a heuristic algorithm rather than meta-heuristic algorithm, there is nothing random about it. Applying a heuristic algorithm to a specific sample i will always generate the same c_{ij}. Thus, we can use c_i to represent the fitness value of a generated heuristic on a specific sample i. Then, (1) can be simplified as:

$$\mu = E[c_i] = \int c_i \mathrm{d}P_I(i). \tag{2}$$

To estimate μ, we evaluate the generate heuristic on a set of samples $\hat{I}$. For each evaluation $j \in \{1, 2, \ldots, |\hat{I}|\}$, we observe a fitness value c_j. Then, μ can be estimated by the estimator $\hat{\mu}$:

$$\hat{\mu} = \frac{1}{|\hat{I}|} \sum_{j=1}^{|\hat{I}|} c_j. \tag{3}$$

It is easy to know that $\hat{\mu}$ is an unbiased estimator of μ:

$$\int \hat{\mu} \mathrm{d}P(\hat{\mu}) = \frac{1}{|\hat{I}|} \sum_{j=1}^{|\hat{I}|} \int c_j \mathrm{d}P_I(j) = \mu. \tag{4}$$

Define the variance of the fitness value of the generated algorithm on the sample space I as:

$$\sigma^2 = \mathbb{E}_I[(c_i - \mu)^2]. \tag{5}$$

Although theoretically there is no upper bound of c_i for stochastic routing problems, in practice we can estimate the upper bound based on experiments. Assume there is a upper bound U and a lower bound L of c_i. Since $\hat{\mu}$ is an unbiased estimator of μ, according to Bernstein inequality, for any $\epsilon > 0$, we have:

$$\mathrm{Prob}\{\mu - \hat{\mu} \geq \epsilon\} \leq \exp(-\frac{N\epsilon^2}{2\sigma^2 + \frac{2(U-L)\epsilon}{3}}) \tag{6}$$

where N is the number of training samples. From (6), if we may know σ, U, L, we can estimate how many training samples are needed to ensure a certain level of deviation ϵ. However, in practice, these three values should all be estimated by generating enough training samples. Actually, since $\hat{\mu}$ is the unbiased estimator of μ, according to the law of large numbers:

$$\frac{1}{|I_s|} \sum_{j=1}^{|\hat{I}|} c_j \to \mu, (|\hat{I}| \to \infty), \tag{7}$$

we can get the similar conclusion that more training data will provide more accurate estimation. This theoretical analysis is in line with our intuition and our experiment in Section V.E especially for the linear and the ANN representations.

Then, we use the experimental results to do a post-hoc analysis about (6). Taking Ugdb1 as the example, we retrieved the best ANN-representation routing policy according to the test performance. Based on the fitness values this policy achieving on the 2000 test samples, we have estimated that $\sigma^2 = 304.21$, $U = 427.85$, and $L = 308.58$. Regarding ϵ, for the sake of convenience and to pursue an accurate estimation, we set a relatively small value $\epsilon = 1$ that is at most $1/300 = 0.33\%$ of the objective value. As to the confidence level, we use a common setting of 95%, which means that we want to have a estimation with deviation less than 1 at a confidence level of 95%. Substitute the above value for the symbols in (6), we have estimated $N \approx 2061$ that is very close to our experimental setting.

One thing should be noticed that according to the theoretically analysis, we should generate 2000 more training samples in advance and evaluate every policy on these training samples, but this training method is too expensive. In analogy with the relationship between the gradient descent method and the stochastic gradient descent method, we have used the batch simulation method in order to save the training time.

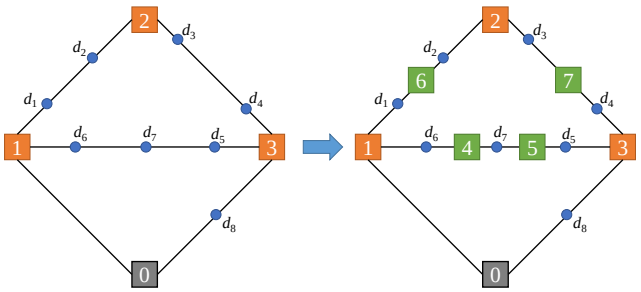


Fig. C. Transforming a CVRP into a CARP.

V. Transforming Node Routing into Arc Routing

In [3], [4], the capacitated arc routing problem (CARP) has been transformed into the capacitated vehicle routing problem (CVRP) by introducing more vertices. In this section, we show a simple inverse transformation from CVRP to CARP in order to verify that the observations we obtaining on UCARP can be extended to stochastic CVRPs.

Consider a CVRP with n customers and each customer i requires a certain demand of cargo d_i. These customers are scattered on a road network $G(V, E)$ where V and E represent the vertex set and the edge set. Fig. C shows a simple example where there are four vertices, five edges, and eight customers. The transforming method is very simple that if two or more customers are on the same edge, between each pair of successive customers, we add a virtual vertex to divide one edge into two parts. The virtual vertex can be allocated in any position between the two customers. After adding virtual vertices, on each edge, there is at most one customer. Thus, we can use the demand of the customer to represent the demand of that edge. In the example shown in Fig. C, we have added four virtual vertices in the map and turned the CVRP to a CARP with nine edges.

Corresponding to the transforming process, the stochastic characteristics of stochastic CVRPs can be transformed to UCARP as well such as the stochastic demand and stochastic traversing cost. Thus, considering both the transform from CVRP to CARP and the inverse one, we may have a inference that the experiments conducted on UCARPs can also represent some stochastic CVRPs.

VI. Investigation of the Search Space

In this experiment, we investigate whether the search space of the the linear representation is unimodal from the perspective of continuous numeric optimization. This investigation is helpful to choose or design suitable optimizer for the linear representation and the other numeric representations. Taking the weights of the linear representation $\psi \in \mathbb{P}^n \subset \mathbb{R}^n$ as the input and the objective value as the output, we can formulate the whole problem as a function $F : \mathbb{P}^n \to \mathbb{R}$. $\mathbb{P}^n$ in this paper is definitely a convex set. If F is a unimodal function, for $\forall \rho \in (0, 1)$ and $\forall \psi_1, \psi_2 \in \mathbb{P}^n$, the following situation will not happen:

$$F(\rho\psi_1+(1-\rho)\psi_2) > F(\psi_1) \wedge F(\rho\psi_1+(1-\rho)\psi_2) > F(\psi_2). \quad (8)$$

TABLE B
Test Performance of the Interpolation Points

case	two exemplars	ρ	fitness	ρ	fitness
Ugdb1	$F(\psi_1) = 344.205$ $F(\psi_2) = 344.175$	0.2 0.4	344.205 344.205	0.6 0.8	344.205 344.205
Ugdb23	$F(\psi_1) = 246.618$ $F(\psi_2) = 246.635$	0.2 0.4	246.377 249.967	0.6 0.8	251.64 251.844

For each instance, we randomly generate 10 pairs of solutions (ψ_1, ψ_2) and their middle point $(\psi_1 + \psi_2)/2$ and then test their objective values based on 2000 samples. If (8) happens, we think the search space is not unimodal. Due to the page limit, we do not show a table in the paper to give all the randomly generated solutions. Basically, the experimental results show that for every instance, the situation of (8) happens. There is no unimodal function among the 23 UCARP instances, which means the real-value search space indeed contains local optima. Thus, even a linear representation is used, mathematical optimization algorithms based on gradient cannot guarantee to find the global best solution.

Another important characteristic should be revealed is the landscape around the global optimum. Taking Ugdb1 and Ugdb23 as examples, for each instance, we choose two different final solutions that have achieved similar objective values as ψ_1 and ψ_2. Then, by setting four ρ values $\{0.2, 0.4, 0.6, 0.8\}$, we evaluate four interpolation points $\rho\psi_1 + (1 - \rho)\psi_2$. The results are shown in Table B.

- The results on Ugdb1 show that the four interpolation points have the same objective values as ψ_1, indicating that around the best solution, there is a flat fitness area. On the one hand, it is a good news that to reach the global optimum, we do not need very accurate weight values with high precision. On the other hand, the flat fitness area may also exist in the other area of the search space. Plus the multimodal characteristic of the search space, it is even harder to optimize the weights.
- The results on Ugdb23 shows the multimodal characteristic of the search space again that when $\rho = 0.4, 0.6, 0.8$, the objective values increase larger than both the objective values achieved by ψ_1 and ψ_2.

Regarding the neural network representation, previously we have stated that because the neurons of the hidden layer do not distinguish from each other, the search space must be multimodal. If this is the only factor which brings the multimodal characteristic, there may be several global optima in the search space and no poor local optima exist, which is a good thing. However, we now have confirmed that even for the linear representation, the search space is not unimodal. Thus, the search space of weights of the basic neural network representation may have many local optima. Thus, its optimization is not an easy job either.

VII. Optimizer Dependence

In order to show that the capacities of the numeric representations have been demonstrated by the CMA-ES optimization algorithm, we used another optimizer to evolve the ANN representation heuristic, called competitive swarm optimizer

TABLE C
COMPARISON OF CSO AND CMA-ES ON ANN

instance	CSO	CMA-ES
Ugdb1	345.86(2.33)	344.93(1.34)
Ugdb2	365.62(2.93)	364.72(2.30)
Ugdb3	307.01(1.07)	307.77(1.07)
Ugdb4	319.26(0.71)	319.53(1.01)
Ugdb5	432.80(7.32)	**419.66(5.91)**
Ugdb6	341.06(3.22)	**333.24(4.37)**
Ugdb7	350.53(4.56)	350.89(5.59)
Ugdb8	440.46(4.83)	**432.53(6.57)**
Ugdb9	397.79(5.72)	**393.34(5.09)**
Ugdb10	298.04(2.72)	**296.43(3.79)**
Ugdb11	431.44(5.60)	**426.59(4.81)**
Ugdb12	**620.57(6.18)**	623.50(6.68)
Ugdb13	582.00(2.25)	581.26(3.53)
Ugdb14	107.46(1.02)	106.36(0.49)
Ugdb15	58.20(0.21)	58.13(0.11)
Ugdb16	134.94(1.12)	134.76(1.50)
Ugdb17	91.17(0.16)	91.11(0.11)
Ugdb18	166.73(1.24)	166.18(0.99)
Ugdb19	61.16(0.51)	61.43(0.72)
Ugdb20	126.47(0.57)	126.94 (1.07)
Ugdb21	164.20(1.44)	163.06(0.56)
Ugdb22	208.86(0.81)	209.11(1.24)
Ugdb23	249.22(1.04)	248.51(1.26)

(CSO) [5]. CSO is one of the state-of-the-art EC algorithms proposed for large-scale global optimization problems with more than 200 variables. The population size of CSO is set to 200. Other settings are kept unchanged. The experimental results in Table C. The highlighted values mean that they are significantly better than their counterparts.

From the experimental results, we can see that CMA-ES shows better performance than CSO in optimizing the weights of the ANN representation heuristics. Only on Ugdb12, CSO shows better results than CMA-ES, but it does not affect the conclusion. Thus, our analysis in the paper is trustworthy.

Nevertheless, we can see that the optimizer indeed affects the final performance of the algorithm. This is inline with the analysis made in the previous section of the investigation of the search space. The search space of the numeric representation has the multimodal character. If the optimizer is not power, it may not be able to exhibit the real capacity of the representation.

As to the tree representation, currently GP may be the only option to use. Although there are several GP variants proposed to solve UCARP, basically they were proposed to reduce the tree size or to save the computation time [6], [7]. They did not improve the objective value significantly.

Overall, although the selection of optimizer would affect the experimental results, we think that the capacities of the three representations have been fully demonstrated in the paper. Thus, the analysis is optimizer-independent.

REFERENCES

[1] M. Birattari, "On the estimation of the expected performance of a metaheuristic on a class of instances," Tech. Rep., 2004.

[2] S. Liu, K. Tang, Y. Lei, and X. Yao, "On performance estimation in automatic algorithm configuration," in *Proc. AAAI2020*, vol. 34, no. 03, 2020, pp. 2384–2391.

[3] H. Longo, M. P. De Aragao, and E. Uchoa, "Solving capacitated arc routing problems using a transformation to the cvrp," *Comput. Oper. Res.*, vol. 33, no. 6, pp. 1823–1837, 2006.

[4] W.-L. Pearn, A. Assad, and B. L. Golden, "Transforming arc routing into node routing problems," *Comput. Oper. Res.*, vol. 14, no. 4, pp. 285–288, 1987.

[5] R. Cheng and Y. Jin, "A competitive swarm optimizer for large scale optimization," *IEEE Trans. Cybern.*, vol. 45, no. 2, pp. 191–204, 2014.

[6] M. A. Ardeh, Y. Mei, and M. Zhang, "A novel multi-task genetic programming approach to uncertain capacitated arc routing problem," in *Proc. GECCO2021*, 2021, pp. 759–767.

[7] S. Wang, Y. Mei, and M. Zhang, "A multi-objective genetic programming hyper-heuristic approach to uncertain capacitated arc routing problems," in *Proc. CEC2020*. IEEE, 2020, pp. 1–8.