Surrogate-Assisted Genetic Algorithms for the Travelling Salesman Problem and Vehicle Routing Problem

1st Muyao Fan
Division of Computing Science and Mathematics
University of Stirling
Stirling FK9 4LA, UK
muf1@stir.ac.uk

2nd Jingpeng Li
Division of Computing Science and Mathematics
University of Stirling
Stirling FK9 4LA, UK
Jingpeng.Li@stir.ac.uk

Abstract—The Travelling Salesman Problem and Vehicle Routing Problem are two similar famous NP-hard problems with simple evaluation functions. While these functions can underestimate the quality of solutions in some cases, surrogate models are widely used in many computationally expensive problems as a supplement and optimization of traditional evaluation methods. By studying two surrogate models based on the spatial structure of the solution, this paper dedicates to introducing an additional level of evaluation on the quality of the individuals' structure in the genetic algorithms designed for TSP and VRP, aiming to maintain the diversity of the population while optimizing the solution of the genetic algorithm in the meantime.

Index Terms—Surrogate, Genetic algorithms, Travelling Salesman Problem, Vehicle Routing Problem.

I. Introduction

Surrogate is a model that implements optimization through the data generated from the optimization process [10]. The purpose of using surrogate model is to approximate problems in the real situation. During the optimization process, this agent model cooperates with traditional models to evaluate individuals, and this mutual cooperation is called Surrogate Management. The agent model is trained on real problem data and has similarities to real problems [20, 16, 9]. The proxy model is used instead of the optimization problem to evaluate the solutions, so that a better solution of the real problem is selected to reduce the number of evaluations to the real problem. In addition, as the surrogate is not as accurate as the real problem, modification is required from the real evaluation to adjust the surrogate assessment. In this article, we try to use two types of surrogate, based on the structure of solutions for the Travelling Salesman Problem (TSP) and the Vehicle Routing Problem (VRP), respectively. We do not look for a quick evaluation of fitness, but use the surrogate as a complement of the GA to optimize the algorithm.

A. Travelling Salesman Problem

The TSP raises a question: given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city? It's a famous NP-hard problem in combinatorial optimization.

When we try to give out a solution for TSP, we either use a local search method to search the neighboring area of the initial solution to find a single new solution or use an evolutionary algorithm to generate a group of new solutions. For each newly-obtained solution, we need to evaluate its quality in terms of the total cost of the tour to visit every city.

Mathematically, the problem may be stated in the following equivalent way: given a "cost matrix" $D = (d_{ij})$, where d_{ij} = cost of going from city i to city j $(i, j = 1, 2, ..., n)$, find a permutation $P = (i_1, i_2, i_3, ..., i_n)$ of the integers from 1 to n that minimizes the quantity $d_(i_1, i_2) + d_(i_2, i_3) + ... + d_(i_n, i_1)$.

So far, various methods for solving TSP to optimal have been proposed. The branch cutting algorithm generally gives the most effective results. One can quickly get 5% solution quality assurance, but it takes a long time to get close to 1% [11]. As Rego and Glover said, the TSP's exact solver seems to "require calculations beyond the practical range" [7]. Therefore, various heuristic methods have been widely used to generate approximate optimal solutions within a reasonable computation time [1, 14, 24]. These heuristic algorithms provide a good balance between solution quality and computation time.

B. Vehicle Routing Problem (VRP)

The solution of VRP is very similar to the solution of TSP in form of chromosome in Genetic Algorithms (GAs), which makes it very convenient to apply the surrogate of TSP to VRP. However, VRP is a more complex combinatorial optimization problem than TSP. In transportation management, goods or services need to be provided to scattered points on map from a certain depot. VRP has received widespread attention in recent years

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

due to its broad applicability and economic importance in determining distribution strategies to reduce the operating costs of distribution systems.

The basic VRP refers to starting from a fixed depot and serving many customers with known needs in different locations. The transportation route of the vehicle needs to depart from a depot and return to the same depot. Each customer can only be visited once. There is also a limit on the maximum load and maximum travel distance. Further more, each time the customer is visited, the cost of the vehicle is also increased, so that the maximum driving distance of the vehicle after visiting a customer will be reduced accordingly. The optimization goal can be the minimization of the total distance traveled, and/or the minimization of the total number of vehicles used.

The VRP is obviously also an NP-hard problem. We can hardly get the exact solution to the problem within a reasonable time. Some effective heuristic algorithms have been developed [12], but for such complex multi-objective optimization combinatorial problems, meta-heuristic algorithms show more advantages. The tabu search algorithm has achieved good results in the application of VRP [19], and the simulated annealing has also got good results [8] although the tabu algorithm requires more computing resources and preset parameters. [18] The ant colony optimization algorithm is another very successful algorithm applied to VRP problems [21], with the results obtained being only slightly inferior than those of the tabu algorithm.

GAs are widely used metaheuristics. There are also many precedents for applying genetic algorithms to VRP problems [23, 2]. Compared with other metaheuristics, GAs have the advantage of a wider search range and are less likely to fall into the local optimal solution. Memetic algorithms, i.e. GAs combined with other heuristic algorithms that are good at finding the best solution in the neighborhood, often have better results [22]. Different from the domain search heuristic algorithm-assisted GA, this paper attempts to build a surrogate model based on the solution structure. By analyzing the structure of each solution, the fitness produced by the surrogate model, which differs from the one produced by the direct fitness evaluation function, is obtained. The surrogate is used to accelerate the convergence speed of the GA, while maintaining the diversity of the population and improving the solution quality.

II. Genetic Algorithms

GAs are inspired by natural selection and Darwinian evolution. They have been used in many disciplines, such as industrial design, computer automation design, social systems and more, to solve real-world problems and achieve optimization. In this case, the problems and standards could not be clearly stated [4]. A GA is an iterative process, and all solutions in each iteration are called a generation. A set of attributes (also known as genes) of the possible solutions is called a chromosome. There are several basic components in GAs: the number of chromosomes, the adaptive selection, the generation of new offspring, and random mutations in new offspring. A GA can basically be divided into the following three processes:

1) Selection, which selects a pool of candidates from the current generation by means of some fitness criteria. The fitter a candidate is, the more likely it is to get chosen. This ensures that only the more suitable candidates are used to create the next generation of individuals.

2) Crossover, which is one of the genetic operators used in the GA. A pair of candidates is selected from the pool generated in the first step to create new offspring. This involves combining the properties from the parents to construct new potential solutions. The selection of the properties can be completely random, or follow a set of rules, e.g. take the first X percent of one parent and the rest from the other parent. X can be randomized in each successive generation to minimize the chance of recession.

3) Mutation, which is another genetic operator. Its main purpose is to ensure genetic diversity. Some randomly selected bits in chromosomes in each generation are changed. The mutation rate is usually kept low to keep the evolution steady.

A. GA for TSP

1) Partially-Mapped Crossover (PMX): The PMX operator was suggested by Goldberg and Lingle (1985) [6]. It passes on ordering and value information from the parent tours to the offspring tours. A portion of one parent's string is mapped onto a portion of the other parent's string and the remaining information is exchanged.

The exchange mutation operator randomly selects two cities in the tour and exchanges them. δ is the consistent mutation rate and $\delta \in (0, 1)$, which is used to control the size of mutation rate in case we may saunter around the good solution instead of reach it. In the later experiment we chose $\delta = 0.5$, so the maximum mutation rate will be double of the minimum one.

Assume that d_H is the sum of Hamming distance of each solution to the best solution in this population. Then the mutation rate of the GA is:

$$P_m = \begin{cases} \delta & (d_H \geq k) \\ \delta\left(1 + \frac{k - d_H}{d_H}\right) & (\varepsilon k < d_H < k) \\ \frac{\delta}{\varepsilon} & (d_H \leq \varepsilon k) \end{cases} \quad (1)$$

Where k is a parameter to adjust the diversity of the GA.

2) Edge Assembly Crossover (EAX): EAX is a very effective algorithm for TSP [15], which focuses on retaining the edges that are inherited from the parents. It is a powerful GA comparable to the traditional LK-based heuristic algorithms for TSP.

EAX has the following 5 steps:

Step 1. Select parents tours A and B, and combine all edges of A and B into one big graph G_{AB}.

Step 2. Separate G_{AB} into small $AB-cycles$ which are formed by selecting cycles consisting of edges of tour A and tour B alternately.

Step 3. Select $AB-cycles$ to form an $E-set$ by specific strategy, and those $AB-cycles$ consisting of two edges are deleted. In our experiment, I chose one random subtour as the $E-set$.

Step 4. Combine $E-set$ with tour A by deleting all edges from tour A in $E-set$ and adding all edges to form tour B in $E-set$. An offspring with some subtours will be generated.

Step 5. Merge the subtours into one valid tour by connecting subtours with each other. The process of connecting heuristically selects two edges in different subtours, removing them and adding two new edges.

To ensure diversity, EAX introduces entropy of the TSP population H to evaluate the individuals as follows;

$$H = -\sum_{e \in X} (F(e)/N) \log ((F(e)/N)) \tag{2}$$

where X is the set of all edges in the population, N is the size of the population, and $F(e)$ is the number of edges e in the current population. During the selection process, if there are children which increase the entropy of the whole group as well as decrease distance, the solution with the shortest path in the offspring is selected. If all offspring with shorter distance than parents will reduce the entropy of the population, choose the solution with the smallest $\delta L/\delta H$ where L is the distance of the tour. If all offspring are not better than tour A in distance, tour A will survive. The evaluation function is:

$$\text{eval}_{ENT}(y) = \begin{cases} \Delta L(y)/\Delta H(y) \text{ if } \Delta H(y) < 0 \\ -\Delta L(y)/\epsilon \text{ if } \Delta H(y) \geq 0 \end{cases} \tag{3}$$

Where ϵ is a small enough positive value. In our experiment, the surrogate fitness S is considered to replace the entropy every 5 generations.

$$\text{eval}_{SURRO}(y) = \begin{cases} \Delta L(y)/\Delta S(y) \text{ if } \Delta S(y) < 0 \\ -\Delta L(y)/\epsilon \text{ if } \Delta S(y) \geq 0 \end{cases} \tag{4}$$

Where $S(y)$ is the surrogate fitness of offspring y.

B. A GA for VRP

For the GA, this paper uses the the exact same method as the GA of TSP to represent chromosomes, which is a sequence containing all nodes once. This not only facilitates the use of the surrogate, but also makes all the generated offspring a legitimate solution, which greatly reduces useless searches. At the same time, this will not lose any information the chromosome contains, since a split algorithm (shown in figure1) is used to find the unique optimal solution corresponding to each chromosome [17]. After the chromosome is obtained, the split algorithm divides the chromosome into many different small sequences, and each sequence starts and ends with the same depot in the VRP. According to Bellman's algorithm[5], given the sequence order and direction, the optimal way of dividing can be found within $O(n^2)$. The crossover operator and mutation operator of the VRP population are consistent with the PMX method in TSP.

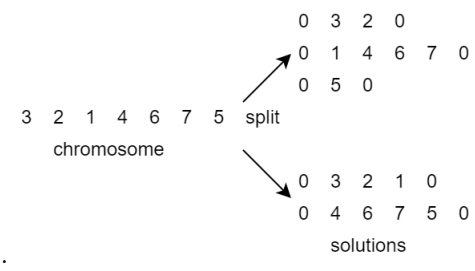

Fig. 1. How a TSP solution splits into two different VRP solutions

III. Surrogate Construction

Although the surrogate models are widely used in real number problems and simulation problems. In discrete combinatorial optimization problems, surrogate model applications are scarce. The reason is not that there are not enough problems in this field, but because the appropriate surrogate model has not been well learned and developed [3]. For combinatorial optimization problems with different properties, the surrogate models to be selected are totally different. Based on feature extraction and similarity comparison strategy, this part introduces several surrogate models suitable for TSP and VRP problems.

The workflow of the surrogate model in GA is as follows:

Step 1. Initialize the population.

Step 2. Use precise methods to evaluate the population.

Step 3. Build/update the surrogate model.

Step 4. Based on the surrogate model, revise the fitness and the population parameters.

Step 5. Generate offspring based on the new fitness.

Step 6. Return to step 2 or end the algorithm.

In both TSP and VRP, the total distance or cost is usually used to measure the quality of the solution, but there is no effective way to identify the feature of the solution. In other words, we get the fitness value of a solution, which cannot reflect the intrinsic properties of the solution itself. For example, in TSP, some solutions may have completely different paths, while the fitness is very close or even equal. Some solutions, even though their distance fitness are bad, can be very close to optimal solution in the view of structure that through one or two transformations it will be optimum. This makes it necessary to analyze the structure of the solution in additional to measuring the original fitness.

For the TSP and VRP, although the a solution is represented a series of nodes, what affects the evaluation value is actually the edges connecting these nodes. Therefore, each solution is no more than a set of edges, which can be

combined to form a solution to a legitimate problem. Analyzing the structure of the solution is the same as analyzing the edges of these sequences. Two different proxy models are proposed below. The first is based on the analysis of the frequency of connected edges of the population. The structural surrogate fitness of the solution is given by comparing the frequency of the edges contained in each individual in the population. The second is constructing an auto-encoder neural network, extracting the features of the solution sequence and mapping it in a lower-dimensional space. The structure-based surrogate fitness of the solution is obtained by analyzing the properties of the solution in this space.

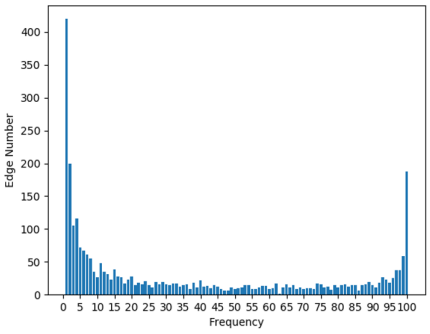

Fig. 2. Edge frequency distribution of 100 random heuristic solution of a random TSP

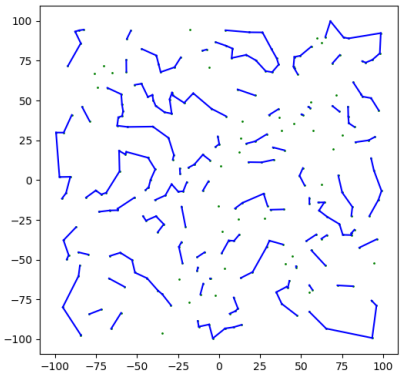

Fig. 3. Edges have frequency more than 80 in 100 random heuristic solutions in a two-dimensional plane

A. Edge Based Surrogate

After observing the common features of many good solutions, we find that as the quality of the solution increases, more solutions tend to have common edges. In other words, the entropy of the population of the entire solutions is decreasing. Those edges with the highest frequency in the entire population have a high probability of being part of the optimal solution, Which does make sense, as if all solutions are optimal, they will be the same solution. Those high-frequency edges are the pattern we are looking for. Therefore, an surrogate model is built based on the number of edges of the solution in the pattern (i.e. good edge) we choose. In the experiment part, we choose the high-frequency edges of 100 random 2-opt solutions as the surrogate model.

1) 3-edge surrogate: 3-edge surrogate model is based on the frequency of joint edges (shown in figure2). We select all the high-frequency edges (shown in figure3) that are connected together, and these connections contain at least 3 edges which have appeared in the high frequency end of the figure. Only those individuals who have the entire trilateral structure can get positive comments. This will raise the standard of "good structure" and avoid overestimating solutions with too many scattered edges. The more 3-edges it has, the higher the scores obtained from the surrogate model. The high frequency is specifically assigned depending on the problem.

2) Linear regression surrogate: Linear regression is a widely used statistical analysis method that uses regression analysis in mathematical statistics to determine the quantitative relationship between two or more variables. We use linear regression to find the relationship between edge frequency distribution and fitness. Take $i = 1, 2, ..., 10$, x_i represents the distribution of edges on different frequencies within an individual solution. For example, x_1 represents how many edges of the solution are not in the surrogate model we built earlier or in terms of the frequency of occurrences in the model that are less than 10%, and x_{10} represents how many edges belong to the frequency of occurrence of 90 to 100%.

$$y = b + \sum_{i=1}^{10} x_i * w_i \tag{5}$$

Where y is the fitness of the surrogate and $\hat{y}$ is the exact fitness function which is the total distance of the tour. The Loss function is

$$L = \sum_{n=1}^{N} \left[\left(\hat{y} \left(\boldsymbol{x}^n, \boldsymbol{w}^{\boldsymbol{n}} \right) - y^n \right)^2 \right] \tag{6}$$

By training this model with good solutions we get from heuristic methods, the $\boldsymbol{w}$ is determined and y is the surrogate fitness of the solution.

B. Auto-encoder based surrogate

Auto-encoder is a form of unsupervised learning neural network [27], which is widely used for dimension reduction or feature extraction of data and pattern recognition.

Auto-encoder is basically a neural network trained with the same input and output [26]. The structure of the auto-encoder is symmetrical. The output of the short middle hidden layer is the code we are looking for, as it contain

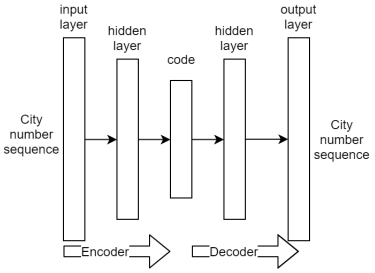

Fig. 4. Schematic diagram of auto-encoder surrogate

all the information of the input data. In TSP or VRP, the input and the output are node sequences, which is exactly the chromosome represents (shown in figure4. For example,in tour $0-2-3-5-6-8-1-7-4$, 0 is the first input and 4 is the last input. We set the dimension of the code to be around one quarter of the number of cities.

It should be noted that in the process of training this auto-encoder neural network, the input is a sequence of nodes, not the edges that really play a decisive role in the solution structure. Also, in both the TSP and VRP, any movement of the starting point of the sequence or any change in direction will not cause any substantial change in the total fitness. Therefore, when training this auto-encoder neural network, the input is a sequence starting from a fixed point, but the output needs to be listed as a sequence of random starting points and direction of the same sequence. This can ensure that the code in the middle of this auto-encoder can effectively extract the characteristics of the chromosome.

We select some random legal solutions generated by the heuristic algorithm and their retransformed sequences as training data for the auto-encode, and then put each individual in the population into the encoder to find their coordinates in low-dimensional space. In this way, the average coordinate of the population, C_p, and the coordinate of the individual with the best fitness in the population, C_b, can be obtained. For the coordinates C_s of a legitimate offspring obtained through crossover, if the offspring is close enough to the best individual, it means that the structure of this offspring is close to the optimal solution in the current population. Although the distance fitness of this offspring may not be necessarily nice, it has enough potential to become a high-quality solution after quite few mutations or inheritance. At the same time, if the coordinates of this offspring are far enough from the average coordinates of the population C_p, it means that this offspring has a structure that most individuals in the population do not have. Therefore, by retaining the offspring, the diversity of the population can be improved.

IV. Computational results

The TSP instances we use come from TSPlib [25], where the number in each instance name represents the number of cities. We use three different surrogates for the TSP GA-based on PMX. We set the size of population $= 300$ and the selective pressure $b = 1.90$. The mutation probability is adjusted according to Equation 1.

For the surrogate based on the structure of 3-edge, the three edges that are connected together and show up more frequently than 80 % are selected as the structure to find. Each generation will retain 5 individuals with the maximum number of 3-edge structures and 5 individuals with the minimum. After the surrogate of linear regression obtains fitness, it will only retain 5 individuals with the highest proxy evaluation value per generation into the offspring.

The auto-encoder uses a solution of 100 randomly generated heuristics and a random selection of 100 different permutations after the starting point for a total of 10,000 groups. Among them, 1,000 data were selected as testing set, and the rest were used as training set. Using Relu as the activation function and Adam as the learning rate algorithm, it stops when the test set accuracy rate does not improve for 20 consecutive epochs. In the selection phase, the auto-encoder also retains the 5 children closest to the optimal solution of the population C_b and the children most deviated from the population average C_m.

We stop the program after running 3,000 generations and get the following results in table I. It can be seen that the edge-based surrogate can effectively improve the quality of the GA's solution in the PMX algorithm with fewer cities, especially for the 3-edge surrogate. The auto-encoder based surrogate is of little help to GAs. In table II, for the TSP using the EAX algorithm, because the EAX algorithm itself is powerful enough, for the problem of a small number of cities, none of the three proxy models has been effective. For the problem with a large number of cities, auto-encoder and linear regression surrogate may improve the GA significantly.

The VRP instances we use come from [13]. Size of population $= 100$, and selective pressure $b = 1.90$. In table III, for more complex VRP problems, auto-encoder is the most effective surrogate, which can improve the quality of the solution by 1.09% on average compared to the standard GA. Edge-based surrogate cannot provide forward optimization steadily, and its effect on GAs is not obvious.

V. Conclusion

In this research we propose a number of surrogate models and test on the TSP and VRP instances of various sizes. The edge-based surrogate model has advantages in simple TSP problems, but it does not help the GA effectively in complex TSP and VRP. On the other hand, the auto-encoder-based surrogate model has significantly improved the quality of GA's solutions in more complex TSP and VRP.

Therefore, it is suggested that the idea of using surrogate models is not restricted to the complex problems

TABLE I
Results of different surrogate of PMX on TSP

Instance	Pure GA	3-edge			Linear Regression			Auto-encoder		
	best	best	improvement	optimum	best	improvement	optimum	best	improvement	optimum
qa194	20978.3	17456.4	0.17	×	19494.2	0.067	×	17488.7	-0.015	×
xqf131	1015.2	772.2	0.24	×	1005.4	0.0096	×	1084.3	-0.068	×
xqg237	1834.2	1636.5	0.11	×	1802.5	0.017	×	1654.5	0.098	×

TABLE II
Results of different surrogate of EAX on TSP

Instance	Pure GA	3-edge			Linear Regression			Auto-encoder		
	best	best	improvement	optimum	best	improvement	optimum	best	improvement	optimum
qa194	9352.2	9352.2	0	√	9352.2	0	√	9352.2	0	√
xqf131	564.5	564.5	0	√	564.5	0	√	564.5	0	√
xqg237	1019.5	1019.5	0	√	1019.5	0	√	1019.5	0	√
lu980	11340.3	11340.3	0	√	11340.3	0	√	11340.3	0	√
mu1979	86891.7	86891.7	0	×	86884.5	0.000082	×	86788.5	0.0011	×
nu3496	96282.5	96132.2	0.0015	×	96162.8	0.0012	×	95801.1	0.015	×
ei8246	206281	206231	0.00024	×	206351	-0.00034	×	205600	0.0043	×

TABLE III
Results of different surrogate on VRP

			Original GA	3-edge		Linear Regression		Auto-encoder	
cities	drop allowance	distance limit		best	%improvement	best	%improvement	best	%improvement
50	0	/	524.61	524.61	0.00	524.61	0.00	524.61	0.00
75	0	/	862.26	866.65	-0.51	864.41	-0.25	848.46	1.60
100	0	/	848.97	842.69	0.74	856.11	-0.84	844.05	0.58
150	0	/	1050.59	1055.53	-0.47	1047.97	0.25	1028.42	2.11
199	0	/	1325.15	1314.55	0.80	1326.35	-0.09	1295.41	2.24
50	10	200	576.40	584.30	-1.37	580.09	-0.64	567.76	1.50
75	10	160	909.68	912.23	-0.28	906.13	0.39	909.68	0.00
100	10	230	897.74	896.30	0.16	887.50	1.14	894.51	0.36
150	10	200	1196.36	1185.95	0.87	1182.60	1.15	1177.93	1.54
199	10	200	1432.21	1444.96	-0.89	1415.17	1.19	1416.17	1.12
120	0	/	1077.71	1083.96	-0.58	1080.94	-0.30	1042.11	3.30
100	0	/	851.13	842.70	0.99	846.53	0.54	848.92	0.26
120	50	720	1596.38	1581.53	0.93	1581.05	0.96	1586.16	0.64
100	90	1040	866.37	863.42	0.34	875.64	-1.07	866.37	0.00
Average					0.05		0.17		1.09

where the evaluation functions are difficult to define or expensive to calculate. For easy-to-evaluate problems like TSP and VRP, the surrogate model can also effectively improve the performance of GAs by analyzing structural features that are different from distance fitness. For many other similar problems, if the fitness of the solution does not fully reflect all the properties of the solution itself, using a surrogate model to extract more structural features and assisting the evaluation of the solution may be an effective way to improve the search ability of an optimization algorithm.

Acknowledgements

The work was supported by a PhD studentship from University of Stirling, as well as National Natural Science Foundation of China (Grants No. 71571076).

References

[1] E Aarts and JK Lenstra. “The traveling salesman problem: A case study”. In: Local Search in Combinatorial Optimization. Wiley, 1997, pp. 215–310.

[2] Guilherme Bastos Alvarenga, Geraldo Robson Mateus, and G De Tomi. “A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows”. In: Computers & Operations Research 34.6 (2007), pp. 1561–1584.

[3] Thomas Bartz-Beielstein and Martin Zaefferer. “Model-based methods for continuous and discrete global optimization”. In: Applied Soft Computing 55 (2017), pp. 154–167.

[4] James S Bergstra et al. “Algorithms for hyper-parameter optimization”. In: Advances in neural information processing systems. 2011, pp. 2546–2554.

[5] Thomas H Cormen et al. Introduction to algorithms. MIT press, 2009.

[6] David E Goldberg, Robert Lingle, et al. "Alleles, loci, and the traveling salesman problem". In: Proceedings of an international conference on genetic algorithms and their applications. Vol. 154. Lawrence Erlbaum, Hillsdale, NJ. 1985, pp. 154–159.

[7] Gregory Gutin and Abraham P Punnen. The traveling salesman problem and its variations. Vol. 12. Springer Science & Business Media, 2006.

[8] DT Hiquebran et al. "A revised simulated annealing and cluster-first route-second algorithm applied to the vehicle routing problem". In: Engineering Optimization 22.2 (1993), pp. 77–107.

[9] Ilija Ilievski et al. "Efficient hyperparameter optimization for deep learning algorithms using deterministic rbf surrogates". In: Thirty-First AAAI Conference on Artificial Intelligence. 2017.

[10] Yaochu Jin. "Surrogate-assisted evolutionary computation: Recent advances and future challenges". In: Swarm and Evolutionary Computation 1.2 (2011), pp. 61–70.

[11] Michael Jünger, Gerhard Reinelt, and Giovanni Rinaldi. "The traveling salesman problem". In: Handbooks in operations research and management science 7 (1995), pp. 225–330.

[12] Gilbert Laporte. "The vehicle routing problem: An overview of exact and approximate algorithms". In: European journal of operational research 59.3 (1992), pp. 345–358.

[13] The OR Library. url: http://mscmga.ms.ic.ac.uk/jeb/orlib.

[14] Peter Merz and Bernd Freisleben. "Genetic local search for the TSP: New results". In: Proceedings of 1997 Ieee International Conference on Evolutionary Computation (Icec'97). IEEE. 1997, pp. 159–164.

[15] Yuichi Nagata. "Edge assembly crossover: A high-power genetic algorithm for the traveling salesman problem". In: Proc. of 7th Int. Conf. on Genetic Algorithms, 1997. 1997.

[16] Mahdi Arian Nik et al. "Surrogate-based multiobjective optimization of a composite laminate with curvilinear fibers". In: Composite Structures 94.8 (2012), pp. 2306–2313.

[17] Christian Prins. "A simple and effective evolutionary algorithm for the vehicle routing problem". In: Computers & Operations Research 31.12 (2004), pp. 1985–2002.

[18] Jacques Renaud, Fayez F Boctor, and Gilbert Laporte. "An improved petal heuristic for the vehicle routeing problem". In: Journal of the operational Research Society 47.2 (1996), pp. 329–336.

[19] Yves Rochat and Éric D Taillard. "Probabilistic diversification and intensification in local search for vehicle routing". In: Journal of heuristics 1.1 (1995), pp. 147–167.

[20] Liang Shi and Khaled Rasheed. "ASAGA: an adaptive surrogate-assisted genetic algorithm". In: Proceedings of the 10th annual conference on Genetic and evolutionary computation. ACM. 2008, pp. 1049–1056.

[21] Petr Stodola et al. "Using the ant colony optimization algorithm for the capacitated vehicle routing problem". In: Proceedings of the 16th International Conference on Mechatronics-Mechatronika 2014. IEEE. 2014, pp. 503–510.

[22] SR Thamgiah, R Vinayagamoorthy, and AV Gubbi. "Vehicle routing with time deadlines using genetic and local algorithm". In: Proceeding of The Fifth International Conference on Genetic Algorithms. 1993, pp. 506–513.

[23] Sam Rabindranath Thangiah, Kendall E Nygard, and Paul L Juell. "Gideon: A genetic algorithm system for vehicle routing with time windows". In: [1991] Proceedings. The Seventh IEEE Conference on Artificial Intelligence Application. Vol. 1. IEEE. 1991, pp. 322–328.

[24] Huai-Kuang Tsai et al. "Heterogeneous selection genetic algorithms for traveling salesman problems". In: Engineering Optimization 35.3 (2003), pp. 297–311.

[25] TSPlib. url: http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html.

[26] Handing Wang, Yaochu Jin, and Jan O Jansen. "Data-driven surrogate-assisted multiobjective evolutionary optimization of a trauma system". In: IEEE Transactions on Evolutionary Computation 20.6 (2016), pp. 939–952.

[27] Yasi Wang, Hongxun Yao, and Sicheng Zhao. "Auto-encoder based dimensionality reduction". In: Neurocomputing 184 (2016), pp. 232–242.